A Global Bisection Algorithm for Computing the Zeros of Polynomials in the Complex Plane

HERBERT S. WILF

University of Pennsylvania, Philadelphia, Pennsylvania

ABSTRACT. A numerical method for solving polynomial equations is presented. Its basic module is a method, using Sturm sequences, for counting the zeros which lie in a given rectangle in the complex plane. The method is deflation-free, handles multiple complex zeros, and contains built-in safeguards against buildup of roundoff error.

KEY WORDS AND PHRASES: polynomial equations, numerical solution, Sturm sequences, roots, zeros of polynomials

CR CATEGORIES: 5.15

1. *Introduction*

We present here a method for finding the roots of polynomial equations on a digital computer. The algorithm is global, i.e. no initial guesses are required, and convergence, at least as far as significant digits allow, is assured. It finds all roots, real and complex, of equations with complex coefficients, together with their multiplicities. No deflation is done as roots are found, for none is needed, and the stability of working always with the original polynomial is thereby retained. The operating time is $O(n^3)$ for an equation of degree n, and the computer program is of just moderate length. An internal check against loss of significant digits is carried along automatically (see Section 3.3).

The underlying idea is this: Suppose we have a method of counting the number of zeros of a polynomial $P(z)$ inside an arbitrary rectangle R in the complex plane. We can then begin with some rectangle R_0, large enough to contain all of the zeros. R_0 can then be subdivided into four subrectangles $R_1, \ldots, R_4$. Next, we count the zeros inside each of the R_i $(i = 1, 4)$, and each R_i which actually contains at least one zero is put on a stack together with the number of zeros it contains. Thereafter we remove a rectangle from the stack and treat it as R_0 was treated, until its size has shrunk sufficiently, at which time we can output its center and the number of roots in its interior.

In Section 2 we give the mathematical basis for the method, which is a procedure based on the theory of Sturm sequences for counting the exact number of zeros of a polynomial which lie inside a given rectangle in the complex plane. In Section 3 we describe the precise implementation of these ideas as a computer algorithm. Section 4 contains a discussion of the precision which is attainable by the method as well as a numerical example, and in Section 5 we make some concluding remarks.

These ideas are related to other methods which are already known. A recent method of Pinkert [10] also uses Sturm sequences to determine regions which contain zeros. Our

This research was supported in part by the National Science Foundation.

Author's address: Department of Mathematics, University of Pennsylvania, Philadelphia, PA 19104.

implementation is quite different. Instead of working with infinite strips and biquadrants, we work always with a rectangle and its four subrectangles, a feature which makes the logic quite simple and modular. The interesting observation of Pinkert that Sturmian methods end themselves to *exact* rational arithmetic applies to our method also, and would lead to complete elimination of roundoff error, although we have not programmed this approach.

Another related circle of ideas is due to Henrici and his co-workers [2–5], who have implemented a construction of Weyl [11] to yield a method which produces small circles which contain the roots of a given equation by operating on all roots at once. Although the results are the same, that small regions containing the roots, the theory, and the practice are quite different from ours; our method, for example, is wholly unperturbed by multiple roots while theirs increases somewhat in complexity with the multiplicity.

2. *Mathematical Basis*

In this section we show how to calculate the number of zeros of a given polynomial $P(z)$ which lie inside a given rectangle R in the complex plane. The method is based on the argument principle, but it is a finite algorithm; more precisely, it is an $O(n^2)$ algorithm, where n is the degree of $P(z)$. The circle of ideas on which the method is based is classical, having been evolved in connection with the Routh-Hurwitz stability criteria (see [1]).

LEMMA 1 (the principle of argument). *Suppose that no zeros of $P(z)$ lie on the boundary ∂R of R. Then for the number of zeros N which lie inside R we have*

$$N = N(P; R) = (1/2\pi)\Delta_{\partial R}\,(arg\ P(z)). \tag{1}$$

Next we consider the computation of the right-hand side of (1) in the case of a polygon R. For this purpose we give:

Definition. By the Cauchy index $I_a^bR(x)$ of a real rational function $R(x)$ on the real interval $[a, b]$ we mean $I_a^bR(x) = N_-^+ - N_+^-$, where N_-^+ (respectively N_+^-) denotes the number of points of $[a, b]$ at which $R(x)$ jumps from $-\infty$ to ∞ (respectively ∞ to $-\infty$).

We shall relate the change in argument around the boundary of a polygon R to the Cauchy indices of a set of rational functions, one for each side of the polygon.

Consider a point z which moves counterclockwise around ∂R, and the curve $w = P(z)$ in the w-plane, which is the image of ∂R under the polynomial mapping. We focus attention on the traversal of a single bounding edge of R, say that joining $z = a$ to $z = b$ by a straight line, and we assume that $P(z)$ is never zero on ∂R.

As z moves from a to b we have $z = a + (b - a)t$ $(0 \le t \le 1)$, and so

$$P(z) = P(a + (b - a)t) = \sum_{\nu=0}^{n} (\alpha_\nu + i\beta_\nu)t^\nu = P_R(t) + iP_I(t).$$

If the image point $P(z)$ crosses from the first quadrant of the w-plane to the second quadrant, the real rational function $P_I(t)/P_R(t)$ will jump from $+\infty$ to $-\infty$. Likewise if $P(z)$ crosses from the third quadrant to the fourth quadrant, $P_I(t)/P_R(t)$ also jumps from $+\infty$ to $-\infty$.

Hence N_+^- counts the number of counterclockwise crossings of the y-axis which the image curve makes as t goes from 0 to 1. Similarly N_-^+ counts the clockwise crossings, and $N_+^- - N_-^+$ is the net excess of the counterclockwise over the clockwise crossings as z traverses the edge ab.

Next we sum the $N_+^- - N_-^+$ around all edges of the polygon, obtaining the excess of counterclockwise crossings of the image curve around the entire ∂R. Then

$$\Delta_{\partial R} \arg P(z) = \pi \sum_{\partial R} (N_+^- - N_-^+),$$

since each extra counterclockwise crossing of the imaginary axis advances arg $P(z)$ by π. This, with (1), proves

THEOREM 1. *Let the complex polynomial $P(z)$ have no zeros on the boundary of the polygonal region R, of m sides. Then in the interior of R the number of zeros of $P(z)$ is exactly*

$$N = N(P, R) = -\frac{1}{2} \sum_{i=1}^{m} I_0^1 \{P_I^{(i)}/P_R^{(i)}\}, \tag{2}$$

where I_0^1 is the Cauchy index of the real rational function $P_I^{(i)}(t)/P_R^{(i)}(t)$ on the i-th edge of ∂R.

Finally, following [1, 12], we describe the computation of the Cauchy index.

Definition. A sequence $f_1, f_2, ..., f_p$ of real polynomials is said to form a *Sturm sequence for a real interval* $[a, b]$ if

(a) f_p is of constant sign on $[a, b]$, and

(b) for each i $(2 \le i \le p - 1)$ and each zero $x^* \in [a, b]$ of $f_i(x)$ we have $f_{i+1}(x^*)f_{i-1}(x^*) < 0$. □

The use of Sturm sequences in calculating the Cauchy index is described by

THEOREM 2. *Let $f_1(x), f_2(x), ..., f_p(x)$ be a Sturm sequence for $[a, b]$, and let $V(x)$ denote the number of sign changes in this sequence at the point x. Then*

$$I_a^b(f_2(x)/f_1(x)) = V(a) - V(b). \tag{3}$$

To summarize the ideas, then, let $P(z)$ be a polynomial and R a rectangle in the complex plane. Let Q_1, Q_2, Q_3, Q_4 $(Q_5 = Q_1)$ be the vertices of R arranged in counterclockwise order around R. On side k of R, containing Q_k, Q_{k+1} $(k = 1, 4)$, we construct a Sturm sequence S_k as follows:

(i) Expand $P(z)$ as a polynomial about the point Q_k. Replace z by $Q_k + i^{k-1}t$, where t is a real variable (this takes the direction $\overline{Q_kQ_{k+1}}$ as the positive real axis, origin at Q_k). Let

$$\tilde{P}(t) = \sum_{\nu=0}^{n} (\alpha_\nu + i\beta_\nu)t^\nu \quad (\alpha_\nu, \beta_\nu \text{ real}(\nu = 0, n))$$

denote the resulting polynomial in t. Take

$$f_1(t) = \sum_{\nu=0}^{n} \alpha_\nu t^\nu, \qquad f_2(t) = \sum_{\nu=0}^{n} \beta_\nu t^\nu. \tag{4}$$

(ii) Construct a Sturm sequence whose first two elements are $f_1(t), f_2(t)$ by the usual negative-remainder algorithm

$$f_i(x) = q_i(x)f_{i+1}(x) - f_{i+2}(x) \quad (i = 1, 2, ..., p - 2), \tag{5}$$

in which each f_{i+2} is the negative remainder left after dividing f_i by f_{i+1}. The algorithm halts after finitely many steps with $f_p(x) = \gcd(f_1, f_2)$, which is a constant under our hypothesis that $P(z)$ has no zeros on the boundary of R.

(iii) Let $V_k(t)$ denote the number of sign variations in the kth Sturm sequence $(k = 1, 2, 3, 4)$ evaluated at the point t. The number of zeros of the polynomial $P(z)$ which lie inside R is exactly

$$N(P; R) = \frac{1}{2} \sum_{k=1}^{4} (V_k(|Q_{k+1} - Q_k|) - V_k(0)). \tag{6}$$

3. *Algorithmic Implementation*

In this section we discuss the various phases of a computer program which carries out the bisection procedure. We treat first a generic step in the calculation, second the start-up procedure, and third the termination criteria.

3.1 A TYPICAL STEP. We maintain a stack of rectangles, each of which contains at least one zero of $P(z)$. We suppose, recursively, that for each rectangle R on the stack we have stored the following information:

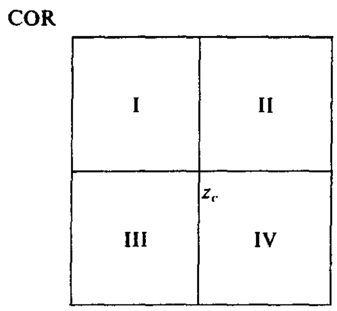
(i) NZ, the number of zeros of $P(z)$ which lie inside R;

(ii) COR, the northwest corner of the rectangle R;

(iii) D1, D2, the length and breadth of R;
(iv) the Sturm sequences $S_1, \ldots, S_4$ on the four sides of R.

These sequences are stored by storing the quotients $q_i(x)$ $(i = 1, \ldots, p-2)$, which appear in (5) because then $2n$ or fewer registers suffice to store a sequence. Evaluation of the sign variation function $V(x)$ at a point x is done by reading the recurrence (5) backward, so that we find $V(x)$ in $O(n)$ operations. A complete description of a sequence requires also its origin and its constant term f_p, both of which are stored.

The information on the stack which relates to the next rectangle R is read into working storage. We locate the center z_c of R, and divide R into four subrectangles through z_c, numbered as shown below:



We form the Taylor expansion of $P(z)$ about z_c, by means of algorithm Taylor of [9]. The Sturm sequence which governs the horizontal line through z_c, with $\mathrm{Re}(z_c)$ as its origin, is formed as in (4) and (5). We rotate the expansion 90° counterclockwise (i.e. find the coefficients of $P(z_c + it)$), and repeat (4) and (5) to obtain the Sturm sequence which lives on the vertical line through z_c, with $\mathrm{Im}(z_c)$ as its origin. We have now all six of the necessary sequences.

From (6), translated to the correct origins, we can readily compute the number of zeros inside each subrectangle I, II, III, and IV. Each one of these four subrectangles which actually contains 1 or more zeros of $P(z)$ is written on the stack, in the sense that the information (i)–(iv) above, all of which is available in working storage, is placed on the stack.

3.2 Start-Up Procedure. To begin the calculation, we choose an initial square S, whose center is at a randomly chosen point in the unit square of the complex plane and whose side length is a random number. We compute the number of zeros inside S by the methods already described. If this number is less than n, we double the side length and repeat. Otherwise we record the stack information described in Section 3.1, all of which is available in working storage, on the stack, and proceed as described in Section 3.1.

3.3 Termination Procedure. The stopping criteria are quite interesting. The left-hand side of eq. (6) is an integer; hence the sum which appears on the right-hand side must be an *even* number. If this should ever fail to happen, it must be that a loss of significant digits in the calculation has introduced spurious sign changes into the Sturm sequences.

Continuous monitoring of the parity of this sum provides an excellent check on loss of significance, as follows: The large rectangle R, from the stack, is subdivided into four subrectangles as described above. All four of the sums just mentioned are checked for evenness. If all are even we proceed as previously described. If one or more are odd we have lost significance, possibly because a zero of $P(z)$ lies on or close to the new lines of subdivision.

Our response is to rechoose the "center" z_c of the large rectangle R by selecting it at random in a suitable small zone which contains the true center (it is here that the rectangles, which had been squares, became nonsquare for the first time). We examine the four new sums to see if they are even, and repeat this process up to three times, if necessary, in an attempt to obtain four even sums. If we fail three times, we output the rectangle R and the number of zeros in its interior. Otherwise the calculation continues. Our experience is that this allows about an additional reduction by a factor of eight in the size of the output rectangle over what we would have obtained by stopping as soon as loss of significance was found for the first time.

Of course, the calculation can also be halted when the rectangles become small enough.

4. *Accuracy, and an Example*

The method relies on accurate evaluation of $P(z)$ for its operation. Consider a zero z^* of multiplicity p. For z near z^* we have $P(z) \sim c_p(z - z^*)^p$ $(c_p \neq 0)$. Suppose further that we carry d decimal digits in our calculation, and that K is the modulus of the largest coefficient of $P(z)$ about the origin.

If it happens that $|c_p(z - z^*)^p| < 10^{-d}K$, then we shall surely have lost all significant digits in z^*. Hence each such zero z^* is surrounded by a "fog zone" of approximate radius

$$R_f \sim 10^{-d/p}|K/c_p|^{1/p}. \tag{7}$$

For example, in a calculation which retains 16 digits, if we assume that $|K/c_p|^{1/p} \approx 1$, then a zero of multiplicity 4 cannot be approached nearer than .0001, and therefore could not be calculated by this method to better than ±.0001 accuracy without resorting to higher precision arithmetic. As discussed in Section 3, the method will itself realize that significance is lost, and will cease subdivision and inform the user of the size of the domain of uncertainty.

Following the suggestions of [7] for vigorous testing of new methods, we computed in double-precision arithmetic (16 digits) the zeros of the polynomial

$$\begin{aligned} P(z) &= z^5 - (13.999 + 5i)z^4 + (74.99 + 55.998i)z^3 \\ &\quad - (159.959 + 260.982i)z^2 + (1.95 + 463.934i)z \\ &\quad + (150 - 199.95i) \\ &= (z - (1 + i))^2(z - (4 - 3i))(z - (4 + 3i))(z - (3.999 + 3i)), \end{aligned} \tag{8}$$

which appears in [6]. Shown below are, for each of the five roots, its real part (the center of the last rectangle), its uncertainty (half the width of the last rectangle), its imaginary part, and its uncertainty. We show 12 digits.

Root	Real (error)	Imaginary (error)
1	4.00000000000 (±.3D − 15)	−3.00000000000 (±.3D − 15)
2	3.99999999999 (±.4D − 11)	2.99999999999 (±.2D − 11)
3	3.99899999999 (±.1D − 11)	3.00000000000 (±.1D − 11)
4	0.99999998109 (±.5D − 8)	0.99999997949 (±.5D − 8)
5	1.00000002282 (±.3D − 8)	1.00000002142 (±.3D − 8)

We observe that the simple zero and the "close" pair give the method little difficulty, as does the fact that these three have nearly equal modulus. The double zero is determined up to the size of the natural fog which surrounds it, given that the calculation was done in double precision. Note also that the double zero was actually split into two, and that the final sizes of the rectangles do not give strict upper bounds on the errors, but only their orders of magnitude.

5. *Conclusions*

For those who need guaranteed root-finding ability, this method is recommended. It seems quite unstoppable, and will converge as near to the roots as the significant digits carried will allow. The absence of deflation is another plus, as is the internal check on loss of significance.

Such luxuries come only at some cost, in this case time. For large degree n, the operation time will be $\sim Cn^3$, compared with Cn^2 of several competitors. The time could be considerably reduced by switching to some local iterative method when the rectangles become moderately small.

Compared with the Lehmer-Schur algorithm [8], ours is of about the same complexity and running time, but avoids deflation because of the fact that our rectangles do not overlap whereas circles must. Compared with the Jenkins-Traub procedure, once again the present method avoids deflation whereas theirs does not. Also our algorithm has one stage compared with their three, which simplifies the program. Against this must be placed the $O(n^3)$ operating time as opposed to their $O(n^2)$. Variations on root-squaring and related algorithms offer similar global convergence in $O(n^2)$ time with, however, both deflation and sensitivity to roots of equal or nearly equal modulus, which leads to the necessity for complicated countermeasures.

ACKNOWLEDGMENTS. My thanks are due to Professor A. Nijenhuis for several stimulating discussions of the ideas presented herein and to Professor Peter Henrici for several suggestions which have resulted in improvements in the manuscript.

REFERENCES

1. GANTMACHER, F.R. *Applications of the Theory of Matrices.* Wiley-Interscience, New York, 1959, Ch. V.
2. GARGANTINI, I., AND HENRICI, P. Circular arithmetic and the determination of polynomial zeros. *Numer. Math. 18* (1972), 305–320.
3. HENRICI, P. Uniformly convergent algorithms for the simultaneous determination of all zeros of a polynomial. *Studies in Numer. Analy. 2* (1968), 1–8.
4. HENRICI, P., AND GARGANTINI, I. Uniformly convergent algorithms for the simultaneous determination of all zeros of a polynomial. Proc. Symp. on Constructive Aspects of the Fundamental Theorem of Algebra, B. Dejon and P. Henrici, Eds., Wiley-Interscience, London, 1969, pp. 77–114.
5. HENRICI, P., AND WATKINS, B.O. Finding zeros of a polynomial by the Q-D algorithm. *Comm. ACM 8* (1965), 570–574.
6. JENKINS, M.A., AND TRAUB, J.F. A three-stage variable-shift iteration for polynomial zeros. *Numer. Math. 14* (1970), 252–263.
7. JENKINS, M.A., AND TRAUB, J.F. Principles for testing polynomial zerofinding programs. *ACM Trans. Math. Software 1* (1975), 26–34.
8. LEHMER, D.H. A machine method for solving polynomial equations. *J. ACM 8* (1961), 151–162.
9. NIJENHUIS, A., AND WILF, H.S. *Combinatorial Algorithms.* Academic Press, New York, 1975.
10. PINKERT, J.R. An exact method for finding the roots of a complex polynomial. *ACM Trans. Math. Software 2* (1976), 351–363.
11. WEYL, H. Randbemerkungen zu Hauptproblemen der Mathematik, II. Fundamentalsatz der Algebra und Grundlagen der Mathematik. *Math. Z. 20* (1924), 131–150.
12. WILF, H.S. *Mathematics for the Physical Sciences.* Wiley, New York, 1962.

RECEIVED DECEMBER 1976; REVISED AUGUST 1977