Technical Report CS74002-R

A TOTAL ALGORITHM FOR POLYNOMIAL ROOTS BASED UPON BAIRSTOW'S METHOD

David A. Ault

Department of Computer Science

College of Arts and Sciences

Virginia Polytechnic Institute and State University

Blacksburg, Virginia 24061

March 1974

Description.

Introduction. This program uses Bairstow's method to find the real and complex roots of a polynomial with real coefficients. There are several reasons for developing a routine based upon Bairstow's method. It is sometimes the case that all of the roots of a polynomial with real coefficients are desired. Bairstow's method provides an iterative process for finding both the real and complex roots using only real arithmetic. Further, since it is based on Newton's method for a system of two nonlinear equations in two unknowns, it has the rapid convergence property of Newton's method for systems of equations. The major drawback of this method is that it sometimes fails to converge [11, p. 110]. This is because it is difficult to find an initial starting guess which satisfies the strict conditions necessary to assure convergence. When these conditions are not satisfied, the sequence of approximations may jump away from the desired roots or may iterate away from the roots indefinitely.

Improvements. The basic Bairstow algorithm has been imbedded in a total algorithm which improves the reliability of this method. The total algorithm contains the following improvements over the basic Bairstow algorithm.

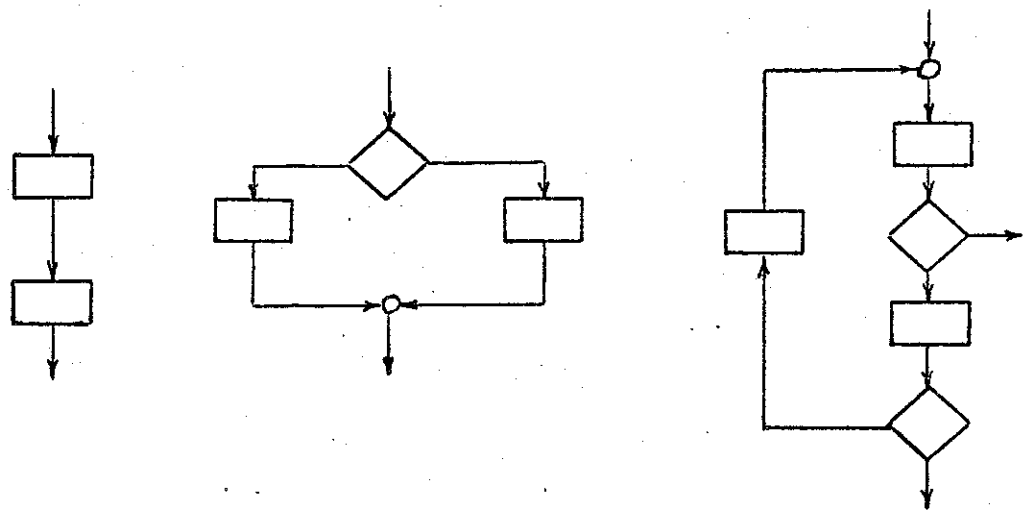
1. The program obtains an initial approximation to two roots of the polynomial and uses these two roots to estimate initial starting values for the Bairstow interation. These values are the coefficients p and q of an approximate quadratic factor, $z^2 - pz - q$, of the given polynomial.
2. During the Bairstow iteration, a check is made for nonconvergence due to interference of a real root. The real root is determined; the polynomial

is deflated; and initial starting coefficients for a new quadratic factor are determined.

3. To reduce the possibility that the Bairstow iteration process will produce an approximation which is a significant step away from the desired pair of roots, the convergence process is monitored by keeping a record of the "size" of the polynomial at the last approximation and comparing it to the "size" of the polynomial at the current approximation.
4. A form of Horner's method for the "synthetic" division of a polynomial by a quadratic polynomial is used which avoids the deliberate introduction of the loss of significant digits in the calculation of the constant term in the remainder.
5. A test is made to detect when the scheme is converging to a multiple root. In this case, convergence is slowed and only a limited number of significant digits in the root are obtained. The multiplicity of the root is determined, the root is found to the required number of significant digits and the polynomial is deflated to eliminate the multiple root.
6. The algorithm is written in modular form. This makes programming and debugging easier, as well as making it straight-forward to modify the algorithm by substituting a new module for a present one. For example, it may be that an efficient and yet more accurate method is developed to find initial approximations to the coefficients of the quadratic factor. This method could easily by incorporated into the algorithm by rewriting the subroutine START using this new method. The rest of the algorithm would remain unchanged.

The logic of each module is constructed from three basic structures. Two of these structures are basic structures of smooth programming [9]. The third structure is an adaptation of a smooth programming structure. To only allow the structures which are currently being offered would complicate

the programming of many numerical procedures. This author believes that the ability to branch out of a loop is essential in numerical programming and, therefore, a structure which allows this type of logic is necessary. Knuth and Floyd [12] expressed the need for the same type of structure to be used in describing table-searching algorithms. The three structures are shown below. The first two are standard structures in smooth programming.



7. The program is written in ANSI Basic FORTRAN [1], except for the use of double precision constants, variables and functions. This sometimes reduces the clarity of the code and it often increases the number of source cards needed, but it makes the algorithm compatible with most FORTRAN compilers. Writing a program of this size in ANSI Basic FORTRAN requires substantial documentation. It is necessary both because of the size of the program and because the language itself cannot be used to clearly describe the algorithm.

Method. Suppose a polynomial $P(z)$ is defined by

$$P(z) = a_n z^n + a_{n-1} z^{n-1} + \ldots + a_1 z + a_0, \qquad (1)$$

where the a_i are real and z is a complex argument. Bairstow's method attempts to find a quadratic polynomial $F(z) = z^2 - pz - q$ which is a factor of $P(z)$. If such a polynomial can be found, then the roots of $F(z)$, and hence two roots of $P(z)$, can be found by the quadratic formula.

First, the polynomial is checked to make sure that the coefficient a_n is nonzero. If it is zero, the degree of the polynomial is reduced by one and the test is repeated. Second, any trivial roots are removed. That is, if the constant coefficient is zero, then zero is a root of the polynomial and we need only apply the Bairstow iteration to find the nonzero roots.

The next step is to reduce the effective degree of the polynomial. For example, the problem of finding the roots of a polynomial $a_4z^4 + a_2z^2 + a_0$ can be reduced to finding the roots $\bar{z}_1$ and $\bar{z}_2$ of the quadratic polynomial $a_4\bar{z}^2 + a_2\bar{z} + a_0$ and then solving for the roots of the original polynomial: $\pm \bar{z}_1$ and $\pm \bar{z}_2$. This process is carried out by the subroutine REDUC. REDUC uses GCD to find the greatest common divisor, say k, of the exponents of the terms of $P(z)$ with non-zero coefficients. If the greatest common divisor is greater than one, then the polynomial is reduced to a polynomial of degree n/k and the greatest common divisor is saved so that the roots of the original polynomial can be found as the kth roots of the reduced polynomial [11, p. 103].

The preprocessing of the polynomial is completed by calling the subroutine NORML to normalize its coefficients so that each coefficient is of maximum modulas one. This not only guards against the possibility of obtaining an arithmetic overflow during the calculation of the value of the polynomial, it permits a more accurate approximation of a zero of the polynomial. This is because the values of the polynomial around a zero fluctuate within a smaller range than would be the case without scaling.

We are now ready to determine the roots of the polynomial. The first step is to find initial approximations to p and q, the coefficients of a quadratic factor $z^2 - pz - q$ of $P(z)$. We can calculate initial values of p and q if we can find initial approximations to two roots of the polynomial. To limit the area of search for these initial approximations, we divide the complex plane into four regions and search these regions one at a time for initial approximations. The plane may be divided into two regions by noting that under the transformation $P(z) \to \overline{P}(z) = z^n P(1/z)$, each root of the transformed polynomial is the reciprocal of a root of $P(z)$.

Thus, for $r > 0$, a root of $P(z)$ either lies in the region $\{z|\ 0 \leq |z| \leq r\}$ or in the region $\{z|\ 0 \leq 1/|z| \leq 1/r\}$. We need to find a suitable choice for r, so that both regions will contain some of the roots of $P(z)$. If each of the roots of $P(z)$ is real, then the product of all its roots is equal to $(-1)^n a_0/a_n$. If we let r be the geometric mean $\sqrt[n]{|a_0/a_n|}$ of all of its roots, then some roots will lie in each region. It is natural to ask if this same number is reasonable when some of the roots of $P(z)$ are complex. We show next that it is.

Each monic polynomial with real coefficients may be factored into the product of linear and quadratic terms, all with real coefficients, where each quadratic factor represents a pair of complex roots:

$$\frac{P(z)}{a_n} = \prod_{i=1}^{k} (z^2 - p_i z - q_i) \cdot \prod_{i=2k+1}^{n} (z - c_i).$$

Therefore, the constant term with absolute value $|a_0/a_n|$ is a product of the c_is and the q_is.

Each quadratic polynomial $z^2 - pz - q$ has roots

$$\frac{p + \sqrt{-(p^2 + 4q)}\,i}{2} \qquad \text{where } p^2 + 4q < 0 \text{ and } i = \sqrt{-1}.$$

The distance of each of these roots from the origin is

$$\left|\frac{p}{2} \pm \frac{\sqrt{-(p^2 + 4q)}\,i}{2}\right| = \sqrt{\left(\frac{p}{2}\right)^2 + \left(\frac{\pm\sqrt{-(p^2 + 4q)}}{2}\right)^2} = \sqrt{-q}.$$

The contribution of each term $-q_i$ $(i = 1,\ldots,k)$ to the term $|a_0/a_n|$ is the product of the distances of the corresponding roots from the origin. Therefore, it is also appropriate to define r to be the geometric mean $\sqrt[n]{|a_0/a_n|}$ in the case that some of the roots are complex.

Since complex roots occur in pairs, one being the complex conjugate of the other, it is sufficient to restrict the search to that portion of each of the above regions where the imaginary part of z is nonnegative. If we write $z = z + yi$, where x and y are real, then we need only search that portion of the above regions where $y \geq 0$. Each of these two regions can be further subdivided by first considering the half where $x \geq 0$ and then the half where $x \leq 0$. The search for approximate roots begins in the region

$$R_1 = \{z \mid z = x + yi,\ 0 \leq x \leq r \text{ and } 0 \leq y \leq r\}.$$

After as many roots as possible have been found in this region, the search is continued in the region

$$R_2 = \{z \mid z = x + yi,\ -r \leq x \leq 0 \text{ and } 0 \leq y \leq r\}.$$

After as many roots as possible have been found in this region, the transformation $P(z) \rightarrow z^n P(1/z)$ is made by reversing the order of the coefficients of $P(z)$. The regions R_1 and R_2 are searched for the roots of this new polynomial $\overline{P}(z)$ in the manner discussed above. Only the roots found here are the reciprocals of the roots of $P(z)$. Each time a root is found, the polynomial if deflated and the process continues with the new polynomial. When the fourth region yields as many roots as it can, the transformation is made which returns the deflated polynomial to the state of the original polynomial and the cycle through

the four regions begins again. This continues until a polynomial of degree two or less remains. In this case, the remaining roots are found directly.

Once one of the four regions has been selected, the subroutine START is called to calculate an initial guess (p_o, q_o) to the coefficients of a quadratic factor $z^2 - pz - q$. The subroutine START performs three functions. First, it locates those regions in which roots of the polynomial may lie. Second, it finds a pair of approximate roots. Third, after it has located a pair of roots, it provides an upper bound on the modulas of the polynomial at the two roots. This is used in the iteration subroutine CONV to monitor the convergence of Bairstow's method. This is important because any initial guess for p and for q, while they may be close to the true solution, may produce a denominator in the correction factors Δp and Δq which is near zero. This causes an unusually large iteration step, often away from the desired solution.

To locate a root of the polynomial in the chosen region, the modulas of the polynomial is approximated by a bivariate polynomial of degree 2 over that region. In particular, the modulas of the polynomial is calculated at the six points 0, r/2, r, $|r|i/2$ $|r|i$ and $r + |r|i$, where r is plus or minus the geometric mean of the roots of the polynomial. The sign of r indicates whether the chosen region is to the right or to the left of the imaginary axis. The coefficients of the bivariate polynomial

$$f(x,y) = a_0 + a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2$$

which interpolates to $|P(z)|$ at these six points can be solved for directly because of the selection of the interpolating points. The minimum of this polynomial occurs at the point at which both of its partial derivatives are zero. This leads to a system of two linear equations in two unknowns, which can be solved by Cramer's rule, provided the denominator in Cramer's rule is

nonzero. If the denominator is zero or if the minimum does not lie in the region, then the initial approximation to a root is taken to be the minimum of this bivariate polynomial along the boundary of the region. A check is made to be sure that the modulas of the given polynomial at this root is "sufficiently" near zero. By this we mean that the modulas of the polynomial at the approximate root must be small compared to the width of the region.

If the approximation is complex, then it and its complex conjugate are used to calculate the initial values p_o and q_o. If it is nearly zero, then we choose $p_o = 0$ and $q_o = 0$. If it is real and nonzero, then we must find another root to be used to determine p_o and q_o. One could use START to approximate another root. If we used the same region, we would obtain the same approximation. We could use another region, but that would require a considerable amount of extra computation. Besides, the next approximation may be complex, not real. To obtain another real root, the assumption was made that there exists a real root of such magnitude that the geometric mean of the two real roots would equal the geometric mean of all the roots. While it is possible to construct polynomials which do not have this property, this method usually gives a reasonable guess at a root. Even in cases where this guess is inaccurate, Bairstow's method converges, although in a greater number of iterations. In the case that the polynomial is of odd degree and only one real root exists, the singular real root is detected early in the Bairstow iteration and the Birge-Vieta method [6, p. 34] is used to find it rapidly. The remaining problem is to determine the sign of the approximation to the second real root. Since the magnitude of the first approximation is less than the geometric mean, the magnitude of the second approximation will be greater than the geometric mean. First check the sign of a_{n-1}/a_n. This number is the sum of all the roots and its sign will be dominated by the sign of the

larger roots. If it is negative, then the larger roots are positive. If it is positive, then the larger roots are negative. If it is zero, then either the roots are paired about the same size, only opposite in sign, or there are several smaller roots of one sign and a few larger roots of opposite sign. In either case we want to choose an approximation of opposite sign and on the other side of the geometric mean from the first approximate root. These two approximations are used to calculate the initial values p_0 and q_0.

The subroutine CONV is called to determine if a quadratic polynomial $z^2 - pz - q$ is a factor of $P(z)$. A generalization of Horner's method is used to determine the coefficients of the linear remainder obtained by dividing $P(z)$ by $z^2 - pz - q$. If we denote the quotient polynomial by $Q(z)$, then $P(z)$ may be written

$$P(z) = Q(z)\ (z^2 - pz - q) + r_1 z + r_0, \tag{2}$$

where

$$Q(z) = r_n z^{n-2} + r_{n-1} z^{n-3} + \cdots + r_3 z + r_2.$$

We use a form of Horner's method which avoids the certain introduction of error into the calculation of r_0 due to loss of significant digits. For $n \geq 3$, the iterative scheme to calculate the coefficients given in (2) is

$$\begin{aligned} r_n &= a_n \\ r_{n-1} &= a_{n-1} + pr_n \\ r_k &= a_k + pr_{k+1} + qr_{k+2} \quad \text{for } k = n-2,\ldots,1 \\ r_0 &= a_0 + qr_2 \end{aligned} \tag{3}$$

We briefly explain this important variation in the calculations. A different form of the remainder term, namely

$$P(z) = Q(z)(z^2 - pz - q) + r_1(z - p) + r_0$$

leads to the iterative equations [4, p. 60]

$$r_n = a_n$$
$$r_{n-1} = a_{n-1} + pr_n$$
$$r_k = a_k + pr_{k+1} + qr_{k+2} \quad \text{for } k = n-2,\ldots,0.$$

Note that convergence to the desired quadratic factor necessarily introduces a loss of significant digits in the calculation of $r_1 = a_1 + pr_2 + qr_3$. This is because a_1 is a fixed number and r_1 approaches zero as $z^2 - pz - q$ converges to a quadratic factor of $P(z)$. Thus the quantity $pr_2 + qr_3$ must approach $-a_1$ and the result of the addition is a loss of significant digits. This results in an error in the calculation of $r_0 = a_0 + pr_1 + qr_2$ because p is multiplied by a factor with few, if any, significant digits [8, p. 646]. We use the iteration scheme (3) which avoids the deliberate introduction of this error.

The original choices (p_0, q_0) for p and q will most likely not result in the coefficients r_1 and r_0 being equal to zero. Thus, we must solve the system of nonlinear equations

$$r_0(p,q) = 0$$

$$r_1(p,q) = 0.$$

Newton's method for systems of nonlinear equations gives an iteration scheme to improve the original approximation to p and q as follows:

$$p_{n+1} = p_n + \Delta p,\ q_{n+1} = q_n + \Delta q,\ n = 0, 1, 2, \ldots$$

where

$$\Delta p = \left(r_0\frac{\partial r_1}{\partial q} - r_1\frac{\partial r_0}{\partial q}\right)\Big/ d,\ \Delta q = \left(r_1\frac{\partial r_0}{\partial p} - r_0\frac{\partial r_1}{\partial p}\right)\Big/ d, \tag{4}$$

and

$$d = \frac{\partial r_1}{\partial p}\frac{\partial r_0}{\partial q} - \frac{\partial r_0}{\partial p}\frac{\partial r_1}{\partial q}.$$

Here r_1 and r_0 and their partial derivatives are evaluated at (p_n,q_n). The problem is to find values for the partial derivatives of both $r_1(p,q)$ and $r_0(p,q)$ with respect to p and q. The method proposed by Bairstow was to take the partial derivatives of each equation in the iterative scheme (3) and to use the resulting equations to calculate the partial derivatives in (4) numerically.

If we calculate the partial derivatives of each of the equations (3) with respect to p and set $s_{i+1} = \partial r_i/\partial p$, $i = 0,\ldots,n-1$, we have the iterative scheme

$$\begin{aligned} s_n &= r_n \\ s_{n-1} &= r_{n-1} + ps_n \\ s_k &= r_k + ps_{k+1} + qs_{k+2} \quad \text{for } k = n-2,\ldots,2 \\ s_1 &= qs_3. \end{aligned}$$

If we find the partial derivatives of each of the equations (3) with respect to q and set $t_{i+2} = \dfrac{\partial r_i}{\partial q}$, $i = 0,\ldots,n-2$, we have the iterative scheme

$$\begin{aligned} t_n &= r_n \\ t_{n-1} &= r_{n-1} + pt_n \\ t_k &= r_k + pt_{k+1} + qt_{k+2} \quad \text{for } k = n-2,\ldots,3 \\ t_2 &= r_2 + qt_4. \end{aligned} \tag{6}$$

Note that the calculations to find s_i and t_i, $i = n,\ldots,3$, are exactly the same, so only one set of the equations needs to be used. The set of equations (6) may be used to calculate each of the four partial derivatives needed in (4). This is because

$$\frac{\partial r_1}{\partial q} = t_3, \quad \frac{\partial r_0}{\partial q} = t_2, \quad \frac{\partial r_1}{\partial p} = s_2 = t_2 + pt_3, \text{ and } \frac{\partial r_0}{\partial p} = s_1 = qt_3. \tag{7}$$

Using (7), Δp and Δq in (4) can be written

$$\Delta p = (r_0t_3 - r_1t_2)/d, \quad \Delta q = (r_1qt_3 - r_0(t_2 + pt_3))/d \tag{8}$$

where

$$d = (t_2 + pt_3)t_2 - qt_3^2.$$

New approximations to p and q can now be calculated from (4).

In the Birge-Vieta method, the value of the derivative of the polynomial at the iteration point x_n is found by taking the quotient polynomial obtained by dividing P(x) by $(x - x_n)$ and evaluating it at x_n. In a similar manner, the evaluation of the variables t_i, $i = n,\ldots,2$ in (6) is equivalent to dividing the quotient polynomial

$$Q(z) = r_n z^{n-2} + r_{n-1} z^{n-3} + \cdots + r_3 z + r_2$$

by the quadratic factor $z^2 - pz - q$. The polynomial may now be written

$$P(z) = (t_n z^{n-4} + t_{n-1} z^{n-5} + \cdots + t_4)(z^2 - pz - q)^2 + (t_3 z + t_2)(z^2 - pz - q) + r_1 z + r_0. \tag{9}$$

If the solution $z^2 - pz - q$ is a multiple factor of P(z), equation (9) shows that t_3 and t_2 must also approach zero. The calculations (8) indicate that the denominator in Newton's method for nonlinear equations in two variables approaches zero. Thus, the rate of convergence will be slowed as is the case when converging to a multiple real root using the Birge-Vieta method.

Because Bairstow's method has the problem that it sometimes fails to converge or that it even diverges, it is necessary to monitor its progress at each iteration. Before the next values of p and q can be determined by (4), two checks must be made. First a test must be made to see if the current approximate values of p and q are at least as good as the last approximations, that is, whether or not the iteration is wandering away from the desired solution. This test is made by calculating the absolute value of the product of the polynomial evaluated at each of the two roots of the equation $z^2 - pz - q$, [11, p. 110]. The value is denoted by NORM in the program. If $z_1, z_2 = (p \pm \sqrt{p^2 + 4q})/2$, then

$$\text{NORM} = |P(z_1)P(z_2)| = |(r_1 z_1 + r_0)(r_1 z_2 + r_0)| = |-r_1^2 q + r_0 r_1 p + r_0^2|. \tag{10}$$

This value is compared with the value of NORM from the previous iteration. If the values do not increase, then it is assumed that the iteration scheme is not diverging and a new p and q are calculated from (4). If the present value of NORM is larger than the previous value, then half the previous values of Δp and Δq are used as corrections to the previous values of p and q instead of continuing the usual iteration process. That is, if the last step moved too far away, then make a step which is only half the distance in the same direction.

Even if the steps are not diverging, it has been shown by Gabler [10] that the Bairstow iteration may continue indefinitely without converging to a quadratic factor of P(z). The problem is determined by the following theorem, due to Gabler [10].

Theorem. If P(z) has a real root $z = k$ and if for some index n,

$$k^2 - p_n k - q_n = 0 \tag{11}$$

is true, then it is also true for the following index (and therefore for all further indices):

$$k^2 - p_{n+1} k - q_{n+1} = 0.$$

Geometrically, this theorem says that all of the iterations from some point on will be confined to the line $\{(p,q) \mid k^2 - pk - q = 0\}$ if the iteration (4) happens to land on that line. In practice, the iterations will parallel this line if they come close to it. The use of this theorem in our algorithm depends upon the number of real roots in P(z).

If the polynomial has only one real root, then the iterations will continue indefinitely because complex roots are found in complex conjugate pairs and $z^2 - px - q$ always has the factor $z - k$ corresponding to the real root k whenever $k^2 - pk - q = 0$.

If there is more than one real root, there is a line in the p,q-plane determined by each distinct real root and the iteration will follow any such line if it finds it. If the real root k is a multiple root, then the line (11) at the point $p_n = 2k$ and $q_n = -k^2$ yields a quadratic factor of P(z). This is because $(z - k)^2 = z^2 - 2kz + k^2$. For any two distinct real roots, there is some point in the p,q-plane where the lines generated by these two roots will intersect. To see this, suppose that k_1 and k_2 are distinct real roots of P(z). The line in the p,q-plane corresponding to k_i is $\{(p,q) \mid k_i^2 - k_i p - q = 0\}$. Solving for the intersection of these two lines leads to a system of two linear equations in two unknowns. The determinant of this system is $k_1 - k_2$, which is nonzero, since $k_1 \neq k_2$. Therefore, there exists a unique solution (the point of intersection is $p = k_1 + k_2$ and $q = -k_1 k_2$).

To identify the case of a single real root k, note from (2) that for any pair p,q such that $k^2 - pk - q = 0$, $0 = P(k) = r_1(p,q)k + r_0(p,q)$. Therefore, if $-r_0(p,q)/r_1(p,q)$ remains relatively constant for several successive approximations to p and q, then it might be the case that the iteration scheme (4) is stuck on or near a line in the p,q-plane corresponding to a real root. Moreover, the real root is nearly equal to this ratio and so it provides a good initial approximation for the Birge-Vieta method. Once the real root has been determined, the polynomial may be deflated and the search continued with this new polynomial. In this way, future calculations will not be influenced by the real root.

If there is more than one real root, we would like to have Bairstow's method converge, possibly along the line determined by one of the real roots, to a quadratic factor which will yield two roots. To prevent the technique explained in the above paragraph from interrupting the Bairstow iteration in the case of convergence to two real roots, the modulas of the polynomial at the two roots of $z^2 - p_n z - q_n$ is considered. If the measure given by (10) indicates

that convergence to a quadratic factor is possible, then a real root is not isolated. When two roots are found, the polynomial is deflated and the process is repeated with this new polynomial.

If one or two roots were not found, then a test is made to detect if the iterative scheme is converging to a multiple root. The fact that the denominator in the calculation of Δp and Δq approaches zero as p and q approach a quadratic factor corresponding to a multiple root introduces roundoff error into the calculation. This prevents convergence to more than a limited number of significant digits. If the polynomial has a multiple root α, either real or complex, then α will be a root of the derivative of the polynomial. The subroutine MULT is called to find the multiple roots. It uses s straight-forward calculation to find the algebraic derivative of a polynomial according to the definition and to test to determine if α is a root of this derivative. If $P^{(i)}(\alpha) = 0$, $i = 0,\ldots,j-1$, and $P^{(j)}(\alpha) \neq 0$, for $j > 0$, then α is a root of multiplicity j and $P^{(j-1)}(z)$ may be used to find the root without the problems of roundoff error due to a near zero denominator in the calculation of Δp and Δq. When the root is found to the number of significant digits desired, use the root to deflate the polynomial P(z) j times. A check is made at each step of the deflation to be sure that α is truly a zero of the remaining polynomial.

If at least one root was found by either the Bairstow iteration, the implementation of the theorem of Gabler or the subroutine MULT, then begin the search for another root in the same region. Otherwise, select the next region and search there for a root.

Several auxiliary routines were used in the program. Some are available as part of the FORTRAN library. The remainder are described briefly here. Two functions included here are KROOT and VALUE. KROOT determines the kth root of

the absolute value of a given number. The algorithm is a generalization of the algorithm in Conte [4, p. 39] to calculate the square root of a number. Given a real number α and an integer $k > 1$, the number α is divided by 2^k until for some integer $j \geq 0$, $\beta = \alpha/(2^k)^j$ and $0 \leq \beta \leq 1$. Newton's method for nonlinear equations is used to find the kth root of β, say γ. Then the kth root of α is $(\beta \cdot 2^{kj})^{1/k} = \beta^{1/k} \cdot 2^j = \gamma \cdot 2^j$. The function VALUE uses Horner's method to calculate the value of a polynomial at a given point. The modulus of this value is returned by the function.

The subroutine BIRGE is the Birge-Vieta method to find the real root of a polynomial [6, p. 34]. Given the polynomial and an initial approximation to a root, the root is returned. The routine uses the subroutine HORNR to find the value of the polynomial and its derivative at the approximate root by repeated application of Horner's method of polynomial evaluation. These values are used in Newton's method to find a closer approximation to the root. As was the case with Bairstow's method, the sequence of iterations must be monitored to be sure that they do not step substantially away from the root. If the "size" of the polynomial at the iterations increases, a smaller step away from the previous approximation is taken instead.

The subroutine NEST uses the sets of equations (3) and (6) to calculate the coefficients r_1 and r_0 of the linear remainder of the polynomial divided by $z^2 - pz - q$, the coefficients r_i $(i = n,\ldots,2)$ of the quotient polynomial obtained and the coefficients t_2 and t_3 of the linear remainder of this quotient polynomial divided by $z^2 - pz - q$.

Two utility routines are EDGE and GCD. EDGE is used by the subroutine START to locate the minimum of the bivariate quadratic polynomial along a specified edge of the region being searched for a root. It is called four times, once for each edge of the region. The other routine GCD, for greatest

common divisor, is used by the subroutine REDUC to find the greatest common divisor of the exponent of those terms of the polynomial having nonzero coefficients. The Euclidean algorithm is used to find the greatest common divisor of two given positive integers [2, p. 159].

Since the effective degree of the polynomial may be reduced by the subroutine REDUC, the roots of the original polynomial may be the k kth roots of this reduced polynomial. These roots are found by the subroutines ROOT1 and ROOT2. If a single real root of the reduced polynomial is determined, the subroutine ROOT1 is called to find the k kth roots of this root. If a quadratic factor is determined, the subroutine ROOT2 is called to find the 2k roots associated with the two roots of the quadratic factor. The algorithm is an application of de Moivre's theorem and the details can be found in Hamming [11, p. 103]. Further, if the roots found by either subroutine are the roots of the transformed polynomial $\overline{P}(z)$, then the reciprocals of these roots are the roots of $P(z)$. This function is also provided by ROOT1 and ROOT2.

Several double precision functions are needed from the FORTRAN library. They are DABS, DMIN1, and DATAN2. DABS is an inline function which finds the double precision absolute value of the argument. DMIN1 returns the minimum of the two double precision arguments. DATAN2 calculates the arctangent of a number in the form a/b. The result is in radians, between $-\pi$ and π.

References

1. American National Standard Basic FORTRAN (ANSI X3.10-1966), American National Standards Institute, Inc., 1430 Broadway, New York, N.Y. 10018, 1966.

2. Birkhoff, Garrett and Saunders MacLane, Algebra, The MacMillan Company, London, 1967.

3. Carrano, F. M., A Modified Bairstow Method for Multiple Zeros of a Polynomial, Mathematics of Computation 27 (1973), 781-792.

4. Conte, S. D., Elementary Numerical Analysis, McGraw-Hill Book Company, New York, 1965.

5. Dijkstra, E. W., Notes on Structured Programming, APIC Studies in Data Processing No. 8, Academic Press, 1972, 1-82.

6. Dorn, William S. and Daniel D. McCracken, Numerical Methods with FORTRAN IV Case Studies, John Wiley and Sons, Inc., New York, 1972.

7. Ellenberger, Kenneth W., Algorithm 30: Numerical Solution of the Polynomial Equation, Comm. ACM 3, 12 (1960), 643.

8. Ellenberger, Kenneth W., On Programming the Numerical Solution of Polynomial Equations, Comm. ACM 3, 12 (1960), 644-647.

9. Foulk, C. R., Smooth Programs, First Computer Science Conference (Columbus, Ohio), Feb., 1973.

10. Gabler, W., Entry Areas for Pairs of Roots Using Bairstow's Method, Zeitschrift Fur Angewandte Mathematik und Mechanik 49, 4 (1969), 249-250.

11. Hamming, R. W., Numerical Methods for Scientists and Engineers, 2nd edition, McGraw-Hill Book Company, New York, 1973.

12. Knuth, D. E. and Floyd, R. W., Notes on Avoiding "GO TO" Statements, Information Processing Letters 1, 1971, 23-31.

13. USA Standard FORTRAN (USAS X3.9-1966), United States of America Standards Institute, 10 East 40th Street, New York, N.Y. 10016, 1966.