Journal of Computational Mathematics, Vol.8, No.3, 1990, 202–211.

COMPARATIVE TESTING OF FIVE NUMERICAL METHODS FOR FINDING ROOTS OF POLYNOMIALS*

Glenn R. Luecke
(*Department of Mathematics, Computation Center, Iowa State University, U.S.A.*)
James D. Francis
(*Department of Mathematics, University of Washington, U.S.A.*)

Abstract

This paper summarizes the results of the comparative testing of (1) Wilf's global bisection method, (2) the Laguerre method, (3) the companion matrix eigenvalue method, (4) the companion matrix eigenvalue method with balancing, and (5) the Jenkens-Traub method, all of which are methods for finding the zeros of polynomials. The test set of polynomials used are those suggested by [5]. The methods were compared on each test polynomial on the basis of the accuracy of the computed roots and the CPU time required to numerically compute all roots.

Introduction

This paper summarizes the results of comparative testing of five methods that find all zeros of a polynomial. Twenty-five polynomials were used which were designed to test potential weakness in such algorithms; see [5]. All computer runs were made on a National Advanced Systems AS/6 computer using the SVS operating system and using the WATFIV FORTRAN compiler in double precision, which means about fifteen decimal digits of accuracy.

Methods tested

1. The Jenkins–Traub Method (JTM)

A description of this method can be found in [3]. The IMSL Library [2] routine ZRPOLY was used to implement this method for polynomials with real coefficients, and the IMSL Library routine ZCPOLY was used for polynomials with complex coefficients.

2. The Laguerre Method (LM)

A description of this method can be found in [1]. The IMSL Library routine ZPOLR is an adaptation of the program ZERPOL developed by B.T.Smith [7]. This routine will only find approximations to the roots of polynomials with real coefficients.

3. The Eigenvalue Method (EM)

It is well known that if A is the $n \times n$ companion matrix of a polynomial p of degree n, then the characteristic polynomial os A is a known scalar multiple of p; see [6]. Thus, the eigenvalues of A are the roots of p. Since A is an Upper Hessenberg matrix, the IMSL Library routines EQRH3F (for real A) and ELRHIC (for complex A) were used to compute the eigenvalues of A.

4. The Eigenvalue Method with Balancing (EMB)

This method is identical to the companion matrix eigenvalue method mentioned above, except that the matrix A was balanced (see [2]) before the eigenvalues of A were computed.

* Received October 4, 1987.

The IMSL Library routines EBALAF (for real A) and EBALAC (for complex A) were used to balance A.

5. The Wilf Method (WM)

A description of this method can be found in [8]. The computer algorithm for this method was kindly sent to us by Dr. Herbert Wilf. In the process of testing this routine, several bugs were found in the FORTRAN code received and appropriate corrections were made.

Test Polynomials

Test polynomials used are those suggested by [5]. Each polynomial was designed to test for a specific potential problem. Even though this report gives some of the test polynomials in factored form, the factors were multiplied out exactly and the coefficients of the polynomial in the form $a_0 + a_1x + a_2x^2 + \cdots a_nx^n$ were used by each method to determine the roots. The following is a list of fourteen polynomials $p_1(z), \cdots, p_{14}(z)$ from which the twenty-five test polynomials $Ex1, \cdots, Ex25$ are derived.

(1) $p_1(z) = B(z-A)(z+A)(z-1)$ with $A = 10^{-2}$, $B = 1$ (Ex 1); $A = 10^{-6}$, $B = 1$ (Ex 2); $A = 10^{8}$, $B = 1$ (Ex 3); $A = 2, B = 10^{-6}$ (Ex 4); $A = 2, B = 10^{8}$ (Ex 5). $p_1(z)$ is designed to test whether relatively large or small zeros or whether large or small coefficients cause difficulty for the method; see [5].

(2) $p_2(z) = (z-1)(z-2)(z-3)\cdots(z-N)$ with $N = 5$ (Ex 6) and $N = 10$ (Ex7). This polynomial is ill-conditioned for large values of N in the sense that the magnitudes of the coefficients vary considerably. The leading coefficient is one while the constant term is $N!$. This causes extreme variation of the polynomial between consecutive roots which can affect the convergence of some methods. N should be chosen small enough so that all coefficients can be represented exactly.

(3) $p_3(z) = (z-10^{-1})(z-10^{-2})\cdots(z-10^{-N})$ with $N = 3$ (Ex 8) and 4 (Ex 9). For this polynomial the coefficient of z^n is one and the constant coefficient is $10^{-N!}.p_3(z)$ is designed to test for underflow in its evaluation and the ability of a method to distinguish zeros that are close together.

(4) $p_4(z) = (z-.1)^3(z-.5)(z-.6)(z-.7)$, (Ex 10). This polynomial, along with p_5 through p_8, have one or more multiple roots and/or "nearly" multiple roots (i.e. distinct, but nearly equal roots), which can cause convergence difficulties for many algorithms.

(5) $p_5(z) = (z-.1)^4(z-.2)^3(z-.3)^2(z-.4)$, (Ex 11).

(6) $p_6(z) = (z-.1)(z-1.001)(z-.998)(z-1.00002)(z-.99999)$, (Ex 12).

(7) $p_7(z) = (z-.001)(z-.01)(z-.1)(z-1)(z-10)(z-(.1+Ai))(z-(.1-Ai))$ with $A = 0$, (Ex13), $A = .1$ (Ex14), $A = .001$ (Ex15). This polynomial has a multiple root at 0.1 when $A = 0$ and a "nearly" multiple root when A is small.

(8) $p_8(z) = (z+1)^5$, (Ex 16).

(9) $p_9(z) = z^5 - 1$, (Ex 17). The five roots of this polynomial are equimodular roots which can cause convergence difficulties, especially for algorithms which use power techniques to separate such roots.

(10) $p_{10}(z) = (z^5 - A^{-1})(z^5 + A)$ with $A = 10^5$ (Ex 18) and 10^{10} (Ex19).

(11) $p_{11}(z) = (z-A)(z-1)(z-A^{-1})$ with $A = 10^3$ (Ex20) and 10^6 (Ex21). p_{11} is designed to test the accuracy of methods that compute roots one (or one complex pair) at a time and then use deflation to calculate the rest.