Finding the roots of a real polynomial simultaneously

using Bairstow's method

Wai-Shing Luk

# Abstract

Aberth's method of finding the roots of polynomials has been shown to be robust. However, in this method, complex arithmetic is required even if the polynomial is real, since it starts from an initial approximation of the complex numbers. For real polynomials, we propose a new method that does not require any complex arithmetic in the iterative process. It is based on the observation that Aberth's method is a systematic use of Newton's methods. Then, in the proposed method, a similar technique is applied to Bairstow's procedure. As a result, the method requires half the amount of computation per iteration as Aberth's method. Numerical experiments show that the new method exhibits competitive overall performance in the polynomials tested.

## Introduction

Consider a problem of finding the roots of a real polynomial . As described in many numerical textbooks, deflation is a standard technique. An alternative approach is to use the formula given by Weierstrass (see [10] and references therein).

where is the degree of the polynomial and denotes the iteration index. Note that the order of convergence of this method is quadratic. A related method for finding quadratic factors was given by Dvorciuk [6]. In 1973, Aberth proposed another method with a cubic rate of convergence [1]:

where denotes the dervative of . These methods have recently been studied for parallel implementations [5, 4, 7]. In a previous discussion [3], it was pointed out that Aberth's method can be seen as a modification of Maehly's proceduce [11, p. 259], only replacing the computed zeros with all other iterates. We refer to this generalized idea as the *Parallel Anticipatory Implicit Deflation* (PAID) method.

As mentioned by Aberth, the symmetry problem can be overcome by using asymmetric iterates for Aberth's method [1]. However, in this configuration, complex arithmetic cannot be avoided even if the polynomial is real. In this paper, we will not consider the possibility of using symmetric iterates. Instead, we try to use the PAID idea and Bairstow's approach to overcome this drawback. This idea is similar to the one in the literature [13], but the obtained method is simpler, which makes it more competitive with Aberth's approach.

Recall that Bairstow's method avoids complex arithmetic by seeking quadratic factors of [11, pp. 301-303]. Let be the coefficients of the linear remainder of such that:

and the coefficients of the linear residuals of such that:

Bairstow's method can be written as:

Horner-type scheme is used to evaluate and . By following the PAID approach, we construct a parallel method as follows. First, we use a method to suppress the computation of quadratic factors. Details will be given in Section 2. Next, by replacing the computed quadratic factors with the trial factors of the suppression method, we obtain a simultaneous version of the Bairstow method, which will be described in Section 3. We will also discuss a simple method for selecting the initial guess in Section 4. Numerical results will be presented in Section 5. For simplicity, we will omit the superscripts in the following sections if it is understood.

## Suppressing the computed quadratic factor

The first step in developing the novel algorithm is to find a method of suppressing, which is described in [8]. Assume that has been found to be a factor of . Let be that deflated polynomial. The goal of suppression is to perform the Bairstow process without explicitly consructing . Let be the coefficients of the linear remainder of and be the coefficients of the linear remainder of . The relation between and can be expressed in the form [8]:

where , , , , , , , and are intermediate variables. By repeating this process, a second quadratic factor can be suppressed, starting with and so on.

## Modified Bairstow method

A simultaneous version of Bairstow's method can now be obtained by replacing in (2.3) with the trial factors in the case of even-degree polynomials, i.e., . Starting with trial factors , , the Bairstow iteration is applied to each trial factor in parallel, treating all other factors as computed and performing the suppression process according to (2.3). Figure 3.1 shows an overview of the modified Bairstow method.

For polynomials of odd degree, we need to do the following special treatment. Before applying to the formula, an extra root is to be added to the polynomial so that the resulting polynomial is of even degree. In our scheme, for convenience, we choose the root of the origin as the extra root. Thus, if there are any roots of the origin in the original polynomial, they should be removed at the beginning and remembered as part of the solution. The entire algorithm is summarized below.

1. Remove any roots of the origin from the polynomial and remember them as part of the solution.
2. If the resulting polynomial is odd, insert a root at the origin to make it even.
3. Iterate according to the algorithm shown in Figure 3.1 to find the root.
4. If the original polynomial is of odd degree, delete the root of the origin inserted in step 2.
5. Add the roots at the origin stored in step 1.

At first glance, the method appears to be inferior to Aberth's method because of the high cost of suppressing the quadratic factor. However, as we mentioned before, the method is actually more economical due to the avoidance of complex arithmetic. Table 3.1 shows the amount of computation required per iteration for both methods. Only the leading terms are counted.

Table 3.1

|  | Aberth’s method | The proposed method |
| --- | --- | --- |
| Horner |  |  |
| Suppression |  |  |
| Total |  |  |

The first row of the table shows the cost of the Horner type of evaluation. It includes the evaluation of and of the evaluation, while in Aberth's method the evaluation of and of the evaluation. The second row indicates the cost of suppressing the zero/quadratic factor. It includes the evaluations of in Aberth’s method and the evaluations of equation (2.3) in the proposed method. If we count addition, subtraction, multiplication and division as a *flop* (floating point operation), then the total flop for each iteration of Aberth’s method is and the proposed method is . Thus, an improvement of about a factor of two is expected.

## Select Initial Guess

We follow Aberth’s suggestion that the initial guesses should be evenly distributed on a circle with the center which euqals to the centroid of zeros [1]:

where . can be determined by the formula , where and represent the coefficients of and respectively. The angle is used to break the symmetry with respect to the real aixs, which is taken as in Aberth’s discussion. For the parallel Bairstow method, we simply take it to be zero. is taken as the effective radius proposed by Chen [2]:

for Aberth's method. Note that is a complex number. For the proposed method, since must be a real number, we use instead. As a result, the initial is given by the following equation.

## Numerical Experiment

We wrote experimental programs in MATLAB to implement Aberth's method and the modified Bairstow's method described in Section 3. The number of iterations (iter.) and the number of floating-point operations (flops) were used to measure performance. Since these two methods have different stopping criteria, we list the maximum value of the actual residual error ( ), which is defined as for comparison purposes. We follow [7] and the test polynomials were taken from Table 1 of [9] (see also [12]).

These programs were run on a DECstation 5000/133 from Digital Equipment Corporation, and the results are listed in Table 5.1. We observe that, with the exception of some polynomials, the convergence rates of the two methods are very similar. In the following, we will discuss them. In problems 3, 5, 6, 7, and 9, Aberth's method converges more slowly. This is because the corresponding polynomials either contain multiple roots, or only real roots, or both. However, the modified Bairstow's method does not necessarily exhibit fast convergence in all cases. For example, in problem 10, slow convergence rates were also found. In Problem 31, the modified Bairstow method required 50 iterations to converge. We observed that for the first 44 iterations, one pair of iterates just wandered, but suddenly approached the root quickly in the last few iterations. An appropriate explanation remains unknown.

## Conclusions

In this paper, the general idea of the proposed method is presented. The advantage of this method is that it avoids complex arithmetic and is inherently parallelizable. Numerical results show the robustness and efficiency of the method.