Fourth Order Modified Laguerre's Method

Thomas R. Cameron
Davidson College

Abstract

We present a novel modification of Laguerre's method that results in a method for the concurrent approximation of all roots of a univariate polynomial. Our method has strong virtues including fourth-order convergence that is observed in practice and belonging to the class of embarrassingly parallel algorithms. A Fortran 90 implementation of our algorithm is available online and comparisons with several other software are provided to show the effectiveness of our approach.

Introduction

Let $p(\lambda)$ be a polynomial of degree m and denote by $(z_1, \ldots, z_m)$ the current approximations to the roots $r_1, \ldots, r_m$ of $p(\lambda)$. The jth approximation is updated via

$$\hat{z}_j = z_j - \frac{m}{G_j \pm \sqrt{(m-1)(mH_j - G_j^2)}}, \tag{1}$$

where

$$G_j = \frac{p'(z_j)}{p(z_j)} - \sum_{\substack{i=1\\i\neq j}}^{m} \frac{1}{(z_j - z_i)} \quad \text{and} \quad H_j = -\left(\frac{p'(z_j)}{p(z_j)}\right)' - \sum_{\substack{i=1\\i\neq j}}^{m} \frac{1}{(z_j - z_i)^2}. \tag{2}$$

On each iteration, z_j is updated for $j = 1, \ldots, m$, unless it was accepted on a previous iteration. In this sense, all roots of the polynomial are approximated concurrently, rather than sequentially.

Initial Estimates In essence, we select complex numbers along circles of suitable radii. What constitues suitable radii is formalized in [Bini] and can be computed via the upper envelope of the convex hull of the set $\{(i, \log|a_i|),\ i = 0, 1, \ldots, m\}$. We compute the convex hull via Andrew's Monotone Chain algorithm [Andrew].

Backward Error The backward error of an approximate root ξ is given by

$$\eta(\xi) = \frac{|p(\xi)|}{\alpha(\xi)}, \tag{3}$$

where $\alpha(\xi) = \sum_{i=0}^{m} |e_i||\xi|^i$.

Pseudocode

Concurrent Style

$(z_1, \ldots, z_m) \leftarrow$ initial estimates
while $i < itmax$ **do**
  for $j = 1$ to m **do**
    if z_j is not close enough to r_j **then**
      Update via (1) and (2)
    end if
  end for
  $i \leftarrow it + 1$
end while

Parallel Style

$(z_1, \ldots, z_m) \leftarrow$ initial estimates
while $i < itmax$ **do**
  parfor $j = 1$ to m **do**
    if z_j is not close enough to r_j **then**
      Compute G_j and H_j via (2) and store
    end if
  end parfor
  parfor $j = 1$ to m **do**
    if z_j is not close enough to r_j **then**
      Use G_j and H_j to compute z_j via (1)
    end if
  end parfor
  $i \leftarrow i + 1$
end while