Computers and Mathematics with Applications 61 (2011) 1305–1334

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

New progress in real and complex polynomial root-finding☆

Victor Y. Pan [a,b,*], Ai-Long Zheng [b]

[a] Department of Mathematics and Computer Science, Lehman College of the City University of New York, Bronx, NY 10468, USA

[b] Ph.D. Programs in Mathematics and Computer Science, The Graduate Center of the City University of New York, New York, NY 10036, USA

ARTICLE INFO

Article history:
Received 27 December 2010
Accepted 27 December 2010

Keywords:
Polynomial root-finding
Real roots
Companion matrices
DPR1 matrices
Eigenvalues
Eigenvectors
Rayleigh quotients
Secular equation
Homotopy continuation methods

ABSTRACT

Matrix methods are increasingly popular for polynomial root-finding. The idea is to approximate the roots as the eigenvalues of the companion or generalized companion matrix associated with an input polynomial. The algorithms also solve secular equation. QR algorithm is the most customary method for eigen-solving, but we explore the inverse Rayleigh quotient iteration instead, which turns out to be competitive with the most popular root-finders because of its excellence in exploiting matrix structure. To advance the iteration we preprocess the matrix and incorporate Newton's linearization, repeated squaring, homotopy continuation techniques, and some heuristics. The resulting algorithms accelerate the known numerical root-finders for univariate polynomial and secular equations, and are particularly well suited for the acceleration by using parallel processing. Furthermore, even on serial computers the acceleration is dramatic for numerical approximation of the real roots in the typical case where they are much less numerous than all complex roots.

1. Introduction

1.1. Background on root-finding

The solution of a univariate polynomial equation is the classical problem of mathematics and numerical mathematics, extensively studied for four millennia (since the Sumerian times) and is still a research area with highly important applications to numerical, algebraic and geometric computations (see, e.g., [1–8], and the bibliography therein).

The increasingly popular matrix methods approximate the roots as the eigenvalues of the associated companion and generalized companion matrices. Matlab's function "roots" applies the QR algorithm to companion matrices. The algorithms in [9,10] alternate the steps of Weierstrass' polynomial root-finding iteration (also called Durand–Kerner's) and of the QR algorithm applied to diagonal plus rank-one generalized companion matrices (hereafter we refer to them as *DPR1 matrices*) associated to polynomial and secular equations. (See our Theorem 7.1, the papers [11–13,8], and the bibliography therein on secular equation.)

Neither of these algorithms exploits the structure of input matrices, but nonetheless for the task of approximation of all n roots of a polynomial of degree n, the Fortune's package EIGENSOLVE [9] competes with the other current best root-finder MPSOLVE by Bini and Fiorentino [14], based on Börsch–Supan's iteration (also called Aberth's or Aberth–Ehrlich's).

☆ Supported by PSC CUNY Awards 61406-0039 and 62230-0040. Some results of this paper have been presented at the International Symposium on Symbolic and Algebraic Computation (ISSAC 2010) in Münhen, Germany, in July 2010.

* Corresponding author at: Department of Mathematics and Computer Science, Lehman College of the City University of New York, Bronx, NY 10468, USA. Tel.: +1 914 737 2637; fax: +1 718 960 8969.

E-mail addresses: v_y_pan@yahoo.com, victor.pan@lehman.cuny.edu (V.Y. Pan), azheng@yahoo.com (A.-L. Zheng).

URL: http://comet.lehman.cuny.edu/vpan/ (V.Y. Pan).

0898-1221/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2010.12.070

Empirically these and various other celebrated iterative root-finders and eigen-solvers rapidly converge to the solution right from the start and with rare exceptions need a rather small constant number of iteration loops per root or eigenvalue. It follows that in practice one needs just the order of bn^2 bit operations (up to a polylog factor) to approximate all the n roots of an input polynomial of a degree n within the relative error bound 2^{-b} (cf. [3,15]).

This is a nearly optimal number of bit operations. Indeed for the worst case input one must process at least bn^2 input bits and therefore must perform at least $0.5bn^2$ bit operations to ensure the above bound on the output errors because in the worst case the input errors are magnified by the factor of n in the output. (Compare, e.g., the roots of the two polynomials $(x-4/7)^n-2^{-n}$ and $(x-4/7)^n-3^{-n}$.)

Such a magnification is not typical for random inputs, and one can decrease the computational cost on the average by tuning the precision of computing to each specific input and output. Such tuning has been incorporated in the MPSOLVE and EIGENSOLVE and can be included into most of the popular root-finders and eigen-solvers as well.

No adequate formal support has been provided so far for the empirical data on the fast convergence of the cited iterations, but this has not been a serious issue for the users, who gladly employ iterative algorithms as soon as their iteration loop is performed fast and their fast convergence has empirical support.

Nearly optimal (up to a polylog factor) upper bounds on the parallel and sequential Boolean and arithmetic time-complexity of the approximation of all roots of a polynomial have been proved based on the divide-and-conquer root-finder in [16,4,17]. These bounds, however, slightly exceed the empirical bounds supported by the other cited iterations, and since the users rely on empirical bounds, the implementation work for the algorithm in [16,4,17] has never had sufficient motivation.

1.2. Advancing the RQ iteration

We devise polynomial root-finders based on the inverse Rayleigh quotient iteration [15,18–20], which is a variant of Newton's iteration [21,22,19], but its power is enhanced because it approximates both eigenvalues and the associated eigenvectors. (Hereafter we use the abbreviation "RQ" for "Rayleigh quotient".) The iteration is a popular means for fast refinement of an approximate eigenvector and empirically has good global convergence to matrix eigenpairs.

It was first applied to polynomial root-finding in [23], then in [24,25]. In these applications the iteration exploited the input matrix structure, used linear arithmetic time $c_{\mathrm{RQ}}n$ per step (for a scalar c_{RQ}) and linear memory space, empirically converged in a few steps [15], and readily incorporated the techniques for tuning the precision of computing for each specific input and output. One can extend the iteration to approximating all eigenvalues via deflation, by applying the iteration concurrently at sufficiently many distinct initial points, or by combining these two techniques. Even for the task of the approximation of all roots the iteration competes with the Börsch–Supan and Weierstrass algorithms according to the tests in [26,23], although its strength is in approximating a single root and all roots in a fixed region. Under appropriate implementation it should become the method of choice for these tasks, and with some further advance can become such also for approximating all roots.

The QR algorithm in [11] also exploits the input matrix structure, and for companion and DPR1 matrices uses linear memory space and only $c_{\mathrm{QRR}}n$ arithmetic operations per step provided that the associated polynomial has only real roots [8]. The papers [27,28] remove this restriction and still support a linear time bound $c_{\mathrm{QR}}n$, but the constant c_{QR} noticeably exceeds c_{RQ} and c_{QRR}. Furthermore unlike the RQ iteration, the QR algorithm in the papers [11,27,28] has numerical problems in exploiting matrix structure and also restricts concurrency in the approximation of distinct eigenvalues.

In the present paper we pushed the advantages of the RQ approach further by incorporating our novel techniques, which simplify every iteration step and avoid or minimize application of deflation techniques for the approximation of all roots. Even under the sequential model of computing we yield noticeable progress versus the algorithms in [23] (see our tables in Sections 5 and 6). Furthermore our preprocessing turns the shifted companion matrices into bidiagonal matrices and turns the DPR1 matrices into diagonal ones, which allows significant parallel acceleration of our iteration steps.

To improve the chances for fast convergence one can apply the iteration to both input polynomial and its reverse and can alternate it with other root-finders such as iterative factorization algorithms in Section 10.

1.3. Real eigen-solving and root-finding

Our another achievement is a novel numerical algorithm that approximates all real roots of a polynomial with real coefficients where real roots are much less numerous than the nonreal ones. The latter case is typical both for random input polynomials [29] and in the practice of algebraic–geometric computations, but the known numerical algorithms approximate all real roots not much faster than they approximate all complex roots. This holds in terms of both theoretical estimates and the actual CPU time [30].

To yield our acceleration we combine our simplified RQ iteration with repeated squaring of the matrix functions $M^{(0)}=I+2\sqrt{-1}(M-\sqrt{-1}\,I)^{-1}$ and $(M^{(0)})^{-1}$ where M is the input matrix. On the one hand, such squaring is inexpensive in the case of companion matrices M (see [31,32]) and DPR1 matrices M (see our Theorems 2.1, 7.8 and 7.9). On the other hand, the smallest eigenvalues of the matrices $\frac{(M^{(0)})^k+(M^{(0)})^{-k}}{\|(M^{(0)})^k+(M^{(0)})^{-k}\|}$ converge to zero as k grows large, so that we can readily approximate the eigenspace associated with these eigenvalues, which (as one can easily prove) is precisely the eigenspace associated with the real eigenvalues of the input matrix M.