Root-Finding with Eigen-Solving

Victor Y. Pan, Dmitriy Ivolgin, Brian Murphy, Rhys Eric Rosholt, Yuqing Tang, Xinmao Wang and Xiaodong Yan

Abstract. We survey and extend the recent progress in polynomial root-finding via eigen-solving for highly structured generalized companion matrices. We cover the selection of eigen-solvers and matrices and show the benefits of exploiting matrix structure. No good estimates for the rate of global convergence of the eigen-solvers are known, but according to ample empirical evidence it is sufficient to use a constant number of iteration steps per eigenvalue. If so, the resulting root-finders are optimal up to a constant factor because they use linear arithmetic time per step and perform with a constant (double) precision. Some by-products of our study are of independent interest. The algorithms can be extended to solving secular equations.

Mathematics Subject Classification (2000). Primary 65H05; Secondary 65H17; Tertiary 65F15.

Keywords. Polynomial root-finding, eigenvalue, generalized companion matrix, secular equation.

1. Introduction

1.1. Background

Polynomial root-finding is a classical and highly developed area but is still an area of active research [McN93, McN97, McN99, McN02, NAG88, P97, P01/02, PMRTa]. The divide-and-conquer algorithms in [P95, P96, P01/02] (cf. [S82], [G52/58, CN94, NR94, K98] on some important related works) approximate all roots of a polynomial by using arithmetic and Boolean time which is optimal up to polylogarithmic factors (under both sequential and parallel models of computing). The algorithm, however, is quite involved, and the users prefer more transparent iterative algorithms, such as Newton’s, Jenkins-Traub’s [JT70, JT72], Müller’s, Laguerre’s, and Halley’s, which use linear arithmetic time per iteration and approximate a single root, and Durand-Kerner’s (actually Weierstrass’) and

Supported by PSC CUNY Awards 66437-0035 and 67297-0036.
Aberth/Ehrlich’s (actually Börsch-Supan’s), which use quadratic time per iteration and approximate all roots of a polynomial (see Tables 4–6). The iterations converge superlinearly if the approximations are close to the roots.

Computing close initial approximations is still an unsettled area. A popular approach is to seek them as approximations to the eigenvalues of the Frobenius companion matrix, whose spectrum is precisely the set of the roots of the polynomial. This property characterizes the more general class of generalized companion (hereafter we say GC) matrices of a polynomial, which can be used instead of the Frobenius matrix and, like it, can be chosen highly structured. Thus one can first approximate the eigenvalues of a GC matrix numerically, by exploiting its structure and employing the highly effective software of numerical eigen-solvers, and then refine the approximations rapidly, by applying the cited polynomial root-finders. Such a combination of the power of numerical techniques of structured matrix computations and symbolic/algebraic methods of computations with polynomials naturally continues the extensive study in [P92, BP94, BP94, P98, P98/01, MP00, P01, EP02, BGP02/04, EMP04, BGP03/05] and the references therein. We contribute to this area once again, although we only cover eigen-solving, not the refining stage.

1.2. The QR DPR1 Approach

Matlab approximates polynomial roots by applying the QR eigen-solver to the Frobenius matrix. This works quite well except that the output approximations to the eigenvalues are frequently too crude and need refinement.

Malek and Vaillancourt in [MV95, MV95a] and Fortune in [F01/02] apply the QR algorithm to the diagonal plus rank-one (hereafter we say DPR1) GC matrices, defined by the polynomial and the root approximations, which we call the companion knots. As soon as the QR algorithm stops and outputs the updated knots, the matrix is updated as well, and the QR algorithm is reapplied to it. According to the extensive tests reported in the three papers and some theory in [F01/02], this process indeed improves the approximations rapidly until they initialize the cited popular root-finders.

In [BGP03/05, BGP04] the rank structure of the DPR1 input matrix has been exploited to accelerate the QR stage of the algorithms in [MV95, MV95a, F01/02] by the order of magnitude. The resulting algorithm uses linear (rather than quadratic) memory space and linear arithmetic time per iteration, but otherwise performs as the classical QR algorithm, remaining as robust and converging as rapidly. The acceleration, however, is achieved only where the companion knots are real or, with the amendment in [BGP04] based on the Möbius transform of the complex plane, where they lie on a line or circle. Thus the algorithms in [BGP03/05, BGP04] use linear space and linear time per step only for the original DPR1 matrix, but not for its updates.