2012

TR-2012003: Root-Finding and Root-Refining for a Polynomial Equation

Victor Y. Pan

Root-finding and Root-refining for a Polynomial Equation

Victor Y. Pan
Supported by NSF Grant CCF-1116736 and PSC CUNY Award 64512–0042.

Department of Mathematics and Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA
victor.pan@lehman.cuny.edu,
home page: http://comet.lehman.cuny.edu/vpan/

Abstract. Polynomial root-finders usually consist of two stages. At first a crude approximation to a root is slowly computed; then it is much faster refined by means of the same or distinct iteration. The efficiency of computing an initial approximation resists formal study, and the users rely on empirical data. In contrast, the efficiency of refinement is formally measured by the classical concept $q^{1/d}$ where q denotes the convergence order, whereas d denotes the number of function evaluations per iteration. In our case of a polynomial of a degree n we use $2n$ arithmetic operations per its evaluation of at a point. Noting this we extend the definition to cover iterations that are not reduced to function evaluations alone, including iterations that simultaneously refine n approximations to all n roots of a degree n polynomial. By employing two approaches to the latter task, both based on recursive polynomial factorization, we yield refinement with the efficiency 2^d, $d = cn/\log^2 n$ for a positive constant c. For large n this is a dramatic increase versus the record efficiency 2 of refining an approximation to a single root of a polynomial. The advance could motivate practical use of the proposed root-refiners.

KEYWORDS: Root-finders, Root-refiners, Efficiency, Polynomial factorization, Companion matrix methods

1 Two stages of iterative polynomial root-finding

The classical problem of polynomial root-finding is still a subject of intensive study because of its important applications to geometric modelling, financial mathematics, signal processing, control, and in particular to computer algebra, for which this is a fundamental task. We refer the reader to Bell (1940), Boyer (1968), and Pan (1997 and 1998) on the rich history of this subject and to McNamee (2002 and 2007) and McNamee and Pan (2012) on numerous old and new polynomial root-finders.

A typical iterative polynomial root-finder consists of two stages. At first substantial effort is invested into computing an initial point that lies much closer

to one of the roots than to any other of them. Then the same or another iteration refines this approximation. The following result provides formal support.

Theorem 1. *(Corollary 4.5 from Renegar (1987).) Assume a polynomial*

$$p(x) = \sum_{i=0}^{n} p_i x^i = p_n(x - z_1) \cdots (x - z_n), \ p_n \neq 0, \tag{1}$$

and Newton's iteration

$$x_{i+1} = x_i - p(x_i)/p'(x_i), i = 0, 1, \ldots \tag{2}$$

where $5n^2|x_0 - z_1| \leq \min_{j>1}|x_0 - z_j|$. *Then* $|x_i - z_1| \leq 2^{3-2^i}|x_0 - z_1|$, *that is the iteration conveges quadratically from the initial point* x_0.

On preceding works and on variations and extensions, which cover Newton's processes in Banach spaces and other iterative root-finders, see Kantorovich and Akilov (1982), Theorem V.4.3; Kim (1985), Smale (1986), Renegar (1987), Curry (1989), Petkovic and Herceg (2001), and the bibliography therein.

2 Divide-and-conquer factorization

Schönhage (1982), Neff and Reif (1994), and Pan (1995 and 2002) numerically factorize a polynomial $p(x) = \sum_{i=0}^{n} p_i x^i$ of (1) into the product of two non-constant factors and continue this splitting process recursively until factoriztion (1) of $p(x)$ into the product of n linear factors is closely approximated. Then the n approximate roots z_j are readily recovered such that $|x_j - z_j| \leq 1/2^b$ for a sufficiently large b and $j = 1, \ldots, n$. This process in Pan (1995) uses $\emptyset(n)$ ops with the precision $O(bn)$, that is $\emptyset(n^2 b)$ Boolean (that is bitwise) operations. Here and hereafter "ops" stand for "arithmetic operations" and $\emptyset(f(b,n))$ means $O(f(b,n))$ up to polylog factors in $b+n$. Pan (2002) decreases the Boolean cost bound by a factor n where we just seek $x_1, \ldots, x_n$ such that $||p(x) - p_n(x - x_1) \cdots (x - x_n)||_1 \leq 2^b ||p(x)||_1$, $||\sum_i u_i x^i||_1 = \sum_i |u_i|$.

Computing such a polynomial factorization is important in its own right because of the applications to time series analysis, Weiner filtering, noise variance estimation, covariance matrix computation, and the study of multi-channel systems (see Wilson (1969), Box and Jenkins (1976), Barnett (1983), Demeure and Mullis (1989 and 1990), and Van Dooren (1994)). Theoretical impact includes extension to the isolation of the roots of a polynomial $p(x)$ with integer coefficients of length at most l and with distinct roots. (Isolation means computation of n disjoint discs, each containing a single root of $p(x)$.) Assuming the equation $b = \lceil (2n+1)(l + 1 + \log(n+1)) \rceil$, which links l to the precision b of the factorization, the extension is proved in Section 20 of Schönhage (1982) based on the gap theorem of Mahler (1964). Combination with the estimates of Pan (2002) yields the following important result.

Theorem 2. *Let polynomial $p(x)$ of (1) have n distinct simple zeros and integer coefficients in the range $[-2^\tau, 2^\tau]$. Then one can isolate the n zeros of $p(x)$ from each other at by using $\tilde{O}(n^2\tau)$ Boolean operations.*

The cited arithmetic and Boolean cost estimates are optimal up to polylogarithmic factors (see Pan (2002)), but the users prefer to employ alternative functional iteration algorithms such as the Weierstrass–Durand–Kerner (hereafter WDK) and Ehrlich–Aberth algorithms (see Weierstrass (1903), Durand (1960), Kerner (1966), Ehrlich (1967), and Aberth (1973)). The known upper estimates for the complexity of these algorithms is no match to the ones of Pan (2002), but the gap disappears if we rely on informal empirical data on excellent global convergence of these functional iterations, that is their convergence right from the start (see our further comments on this behavior in the Appendix). Our next sections show, however, that at the stage of the refinement of approximate roots even these highly successful algorithms remain by far inferior to the recursive factorization approach.

3 Efficiency of refinement

Assume d function evaluations per iteration that refines an initial approximation and converges with order q. Then it is customary to measure the efficiency of the refinement by

$$\text{eff} = q^{1/d}. \tag{3}$$

For example, we have $q = d = 2$ and $\text{eff} = \sqrt{2}$ for Newton's iteration $x_{i+1} = x_i - p(x_i)/p'(x_i)$, $i = 0, 1, \dots$, whereas $d = 1$ and $q = \text{eff} \approx 1.839$ for the root-finder of Muller (1956). More generally, we write $d = 0.5f/n$ provided the iteration uses f ops and the input polynomial has degree n, and therefore can be evaluated at a point in $2n$ ops. The record efficiency of the known root-refiners for a single root of a polynomial is 2 (see McNamee and Pan (2012)), in particular achieved by combining a linearly convergent root-finder with the Δ^2 convergence acceleration of Aitken (1926). In the next sections, however, we refine all n roots of $p(x)$ with the much greater efficiency

$$\text{eff} = 2^d,\ d = cn/\log^2 n \text{ for a positive constant } c. \tag{4}$$

The supporting algorithms are numerically stable, their precision is controlled, and their Boolean cost stays nearly optimal. In particular we avoid using polynomial evaluation at n point in $O(n \log^2 n)$ ops, which saves ops but is numerically unstable and has inferior Boolean complexity.

4 Root-refining via recursive divide-and-conquer factorization

Suppose we are given the coefficients of a polynomial $p(x)$ of (1) and approximations $z_1, \dots, z_n$ to its n simple roots $x_1 \dots, x_n$. This defines an approximate

factorization

$$p = p(x) \approx f(x) = p_n(x - z_1) \cdots (x - z_n), \tag{5}$$

and Schönhage (1982) (by extending Ostrowski (1940 and 1966)) bounds the approximation errors $|x_j - z_j|$ for $|z_j| \le 1$ and $|\frac{1}{x_j} - \frac{1}{z_j}|$ for $|z_j| \ge 1$, $j = 1, \dots, n$ in terms of the norm $||p(x) - f(x)||_1$. By applying Newton's multivariate iteration to refine this factorization, we arrive at the WDK algorithm (surprisingly this link has not been observed until Pan and Zheng (2011b)).

Now suppose we are given an initial approximate factorization of the polynomial p into the product of two factors of comparable degrees,

$$p \approx f = f_1 f_2, \tag{6}$$

$$\deg f_1 < c \deg f_2 < c' \deg f_1 \tag{7}$$

for two positive constants c and c'; furthermore assume that the root sets of these factors are separated by a root-free annulus $A(z, r, R) = \{x : \ r \le |x - z| \le R\}$ bounded by two circles with a center z and radii r and R, respectively, such that $R/r > 1 + c/n^d$ for two constants $c > 0$ and d; we call the ratio R/r the *relative width* of the annulus and call $A(z, r, R)$ a (c, d) *annulus*. Then one can recursively refine these factors by computing the polynomials

$$f_j^{(\text{new})} = f_j + t_j$$

for $j = 1, 2$ as well as Newton's correction polynomials t_j satisfying

$$\frac{r}{f} = \frac{t_1}{f_1} + \frac{t_2}{f_2} \tag{8}$$

where $r = p - f$, $\deg t_j < \deg f_j$ and $f_j = \frac{f}{f_j}$ for $j = 1, 2$ (cf. Schönhage (1982)). The well known algorithms compute such a partial fraction decomposition (hereafter referred to as PFD) by using $O(n \log^2 n)$ ops (cf. Bini and Pan (1994), Problem 4.2c (PART·FRAC), pages 30–31). The computation is prone to numerical stability problems, but the modification in the next section avoids them.

The iterative updating process converges quadratically and define approximate factorization (6) provided we are given a (c, d) annulus with a positive c and $d \le 1$, e.g., a $(1, 1)$ annulus (see Kirrinnis (1998)).

As soon as we closely approximate the factors f_1 and f_2, we recursively factorize both of them in similar fashion, stopping when we arrive at a refined complete approximate factorization (5) where $z_1, \dots, z_n$ approximate the n roots $x_1, \dots, x_n$ with a desired accuracy.

In the next sections we support estimate (4) in two ways, by employing the algorithms of Kirrinnis (1998) and Bini and Pan (1996), respectively.

5 Polynomial root-refining based on Kirrinnis' iteration

Kirrinnis (1998) has extended the above techniques to the refinement of an initial factorization $p \approx f = f_1 \cdots f_s$ into the product of s nonconstant factors for any

integer s from 2 to n, that is he assumed $1 < s \le n$, $\deg f_j > 0$ for all j, and $\deg f_1 + \cdots + \deg f_s = n$. Furthermore, he has proved quadratic convergence of the iteration as well as of its variant in which he improved the efficiency and numerical stability of the refinement. In this variant he confined the most expensive and numerically unstable stage of the PFD computation to the first iteration. At all subsequent iterations he updated the corrections t_j and new factors f_j as follows (we decrypt his formulas a little):

$$f = f_1 \cdots f_s, \tag{9}$$

$$t_j^{\text{new}} = (2 - t_j \frac{f}{f_j}) t_j \bmod f_j, \ j = 1, \ldots, s, \tag{10}$$

$$f_j^{\text{new}} = f_j + (t_j^{\text{new}} p \bmod f_j), \ j = 1, \ldots, s. \tag{11}$$

In the above variations, polynomial multiplications replace the computation of PFDs; this improves numerical stability and decreases the number of ops per refinement iteration to $O(n \log n)$. Kirrinnis (1998) also estimates precision and Boolean cost of these computations: they stay at nearly optimal level.

We assume the iteration for $s = 2$ and the balanced splitting condition (7), extend splitting recursively, and arrive at the bound (4).

Theorem 3. *Assume n close initial approximations to n distinct roots of a polynomial $p(x)$ of (1) and refine them by recursively applying equations (9)–(11) for $s = 2$ and balanced splitting condition (7), also extended recursively. Then the refinement has efficiency (4).*

Proof. Represent the above recursive refinement process by a binary tree whose root p has two children f_1 and f_2, each of them in turn has at most two children such that $f_1 \approx f_{11} f_{12}$ and $f_2 \approx f_{21} f_{22}$, and so on. At every level of the tree its nodes represent polynomials whose degrees sum to $n - l$ where l denotes the number of linear factors output at the previous levels. The tree has $O(\log n)$ levels because we balance degrees of the factors in every splitting by extending bound (7). It follows that computing Newton's corrections for all factor polynomials at each level takes $O(n \log n)$ ops per iteration. This is translated into $d = O(\frac{\log n}{n})$ per level, $d = O(\frac{\log^2 n}{n})$ for all the $O(\log n)$ levels, and thus into (4) because $q = 2$ for Newton's iteration.

6 Polynomial root-refining based on splitting about a line

An alternative recursive factorization process of Cardinal (1996) employs the companion matrix techniques; its refinement in Bini and Pan (1996) also supports efficiency (4). Cardinal (1996) factorizes a polynomial $p(x)$ of a degree n based on recursive application of the matrix sign iteration

$$y_{i+1} = \frac{1}{2}(y_i + y_i^{-1}), \ i = 0, 1, \ldots \tag{12}$$

to the associated companion matrix C of the polynomial $p(x)$. The iteration begins with $y_0 = C$ and defines a quadratically converging process that splits an input polynomial into two factors with the two root sets separated by the imaginary axis $\{x:\ \Re(x) = 0\}$ provided no roots lie on this axis. The following theorem specifies convergence estimate; they show faster convergence for the images of the roots that lie closer to the two points ± 1.

Theorem 4. *(See Bini and Pan (1996), page 500.) Write* $\gamma = |\frac{y_0-1}{y_0+1}|$ *and assume (12) and* $\Re(y_0) \neq 0$. *Then* $|y_i - 1| \leq \frac{2\gamma^{2^i}}{1-\gamma^{2^i}}$ *for* $i = 0, 1, \dots$ *if* $\Re(y_0) > 0$, *whereas* $|y_i + 1| \leq \frac{2}{\gamma^{2^i}-1}$ *for* $i = 0, 1, \dots$ *if* $\Re(y_0) < 0$.

As soon as the root images converge to the points 1 and -1, one can readily split out a respective factor $f_1 = f_1(x)$ of $p(x)$ (see Pan, Qian and Zheng (2012)); then one can approximate the factor $f_2 = p/f_1$ by applying approximate division (cf. Pan (1995) or Kirrinnis (1998)).

Iteration (12) involves additions and inversions in the Frobenius algebra $\mathcal{A}_C$ generated by the companion matrix C of the input polynomial $p(x)$; an addition takes n ops, inversion $O(n \log^2 n)$ ops (see Cardinal (1996) or Pan(2005)).

Bini and Pan (1996) replace (12) with the iteration

$$y_{i+1} = (15 - 10y_i^2 + 3y_i^4)y_i/8,\ i = 0, 1, \dots \tag{13}$$

By virtue of the following theorem (see Bini and Pan (1996), Proposition 4.1), the iteration converges cubically to one of the two points $z = \pm 1$ from any initial point in the discs $D_{z,1/2} = \{x:\ |x - z| \leq 1/2\}$.

Theorem 5. *Write* $\gamma_i = |y_i - \text{sign}(\Re(y_i))|$ *for* $i = 0, 1, \dots$. *Assume (13) and* $\gamma_0 \leq 1/2$. *Then* $\gamma_i \leq \frac{32}{113}(\frac{113}{128})^{3^i}$ *for* $i = 1, 2, \dots$

Transition from iteration (12) to (13) replaces matrix inversions by multiplications in the algebra $\mathcal{A}_C$, each reduced to six FFTs a n points, and so the cost decreases to $O(n \log n)$ ops per step, translated into $O(\log n)$ function evaluations per iteration, similarly to Kirrinnis (1998). Consequently the iterative process again supports bound (4), although in a way distinct from Kirrinnis (1998). Moreover, the algorithm avoids the computation of a PFD, required at the first iteration of Kirrinnis (1998).

7 Computation of (1, 1) annuli

Next we complete the refinement algorithms by computing $(1, 1)$ annuli; the first of the annuli defines the factors f_1 and f_2 satisfying (7), and the next annuli define similar splittings of the polynomials f_1, f_2, and their factors at the subsequent recursive steps.

We can compute the desired annuli by following Pan (1995 and 2002) and based on approximating the roots of higher order derivatives of $p(x)$, but this

process is rather complicated and expensive; we proceed more efficiently by using close initial approximations to the roots of $p(x)$ available for refinement. We first recall the following results (cf. Pan (2002)).

Theorem 6. *Given a polynomial $p(x)$ of (1) two real constants $c > 0$ and d, and a complex value z, we need $O(n \log^2 n)$ ops to approximate within relative errors of $1 + c/n^d$ the distances $|z - x_j|$ between z and all roots x_j of $p(x)$ for $j = 1, \dots, n$.*

Corollary 1. *Given a polynomial $p(x)$ of (1) with n distinct roots, a fixed real $u > 1$ and a complex v_0, we need $O(n \log^2 n)$ ops to compute either (i) a required wide separating annulus for $p(x)$ or (ii) a disc $D(v_1, \rho_1) = \{x : |x - v_1| \le \rho_1\}$ containing at least $n/12$ roots of $p(x)$ where $|v_0 - v_1| \ge u\rho_1$.*

It remains to extend computations from case (ii) to arrive at case (i). Before delving into this, however, we decrease the estimates of the theorem and consequently corollary by a factor $\log^2 n$ by using approximate roots $z_1, \dots, z_n$, not assumed to available in Pan (2002). Furthermore, by reapplying the amended corollary for v replacing z, we obtain a disc $D(v_2, \rho_2) = \{x : |x - v_2| \le \rho_2\}$ containing at least $n/12$ roots of $p(x)$ and such that $|v_2 - v_0|/\rho_2$ has order of u^2, unless we yield the desired case (i). By reapplying the corollary $t = O(\log n)$ times we use $O(n \log n)$ ops overall and ensure either the desired case (i) or the bound $|v_t - v_0|/\rho_t \ge s = cn^d$ for any fixed real $c > 0$ and d. It remains to treat the latter case where it is sufficient for us to choose c and d such that $s = cn^d \ge 3$ and thus

$$|v_t - v_0|/\rho_t \ge 3. \tag{14}$$

Let n_s denote the number of the roots of $p(x)$ in the disc $D(v_k, s\rho_k) = \{x : |x - v_k| \le s\rho_k\}$. By assumption (ii) we have $n_s \ge n/12$ for $s \ge 1$. Consider n_s for $s = s_h = (1+1/n)^h$ for $h = 0, 1, \dots$. If $s_h = s_{h-1}$, then $A(v_k, s_{h-1}\rho_k, s_h\rho_k)$ is a root-free annulus with relative width $1+1/n$, that is a $(1, 1)$ annulus. Clearly, $s_h > s_{h-1}$ for at most $n - n/12$ integers h because at most $n - n/12$ roots lie outside the disc $D(v_t, \rho_t)$. Therefore we have a $(1, 1)$ annulus $A(v_t, s_{h-1}\rho_t, s_h\rho_t)$ for $h \le n - n/12$. It supports approximate factorization (6). Note that

$$s_h/\rho_t \le (1 + 1/n)^h \le (1 + 1/n)^n < 3. \tag{15}$$

Now it remains to ensure the degree bound (7).

We achieve this by computing a complex v_0 such that application of the avove construction produces the factor f_1 of degree at most $3n/4$. In our procedure of computing v_0 assume that we have sufficiently close approximations z_j to the n distinct roots x_j of $p(x)$, $j = 1, \dots, n$. Furthermore assume that $n = 4k$ is divisible by 4, $\Re(z_j) \le \Re(z_{2k}) < \Re(z_{2k+1}) \le \Re(z_l)$ for $j < 2k$ and $l > 2k + 1$, $\Im(z_j) \le \Im(z_k) < \Im(z_{k+1}) \le \Im(z_l)$ for $j < k$ and $k + 1 < l \le 2k$, and $\Im(z_j) \le \Im(z_{3k}) < \Im(z_{k+1}) \le \Im(z_l)$ for $2k < j < 3k$ and $3k < l$. We can yield the latter bounds by properly enumerating the roots if all their projections on the real and imaginary axis are distinct; the latter property holds with probability 1 under random rotation of the complex plane.

Now we proceed as follows.

Flowchart 1. Computing a $(1,1)$ annulus.

COMPUTATIONS:

1. *Compute the half-sum* $a = 0.5(\Re(z_{2k}) + \Re(z_{2k+1}))$.
2. *Compute the three half-sums* $a_1 = 0.5(\Im(z_k) + \Im(z_{k+1}))$, $a_2 = 0.5(\Im(z_{3k}) + \Im(z_{3k+1}))$ *and* $b = 0.5(a_1 + a_2)$.

OUTPUT $v_0 = a + b\sqrt{-1}$.

Theorem 7. *Suppose that Flowchart 1 has output a complex* v_0 *and that the inequality* $|v_k - v_0| > \rho_k$ *holds for some complex value* v_k *and positive* ρ_k. *Then the disc* $D(v_k, \rho_k)$ *contains at most* $3n/4$ *roots of* $p(x)$.

Proof. Partition the complex plane into four domains D_1, D_2, D_3, and D_4 bounded by the straight line $L = \{x : \Re(x) = a\}$ and the two half-lines $R_1 = \{x : \Re(x) \le a \text{ and } \Im(x) = a_1\}$ and $R_2 = \{x : \Re(x) \ge a \text{ and } \Im(x) = a_2\}$, both half-lines being orthogonal to the line L. The point v_0 lies on the line L at the same distance from the half-lines R_1 and R_2. By definition of these four domains, each of them contains exactly $k = n/4$ roots of $p(x)$. Combining (14) and (15) obtain the bound $|v_t - v_0| > \rho_t$, which implies that the disc $D(v_k, \rho_k)$ can have nonempty intersections with at most three of the domains D_1, D_2, D_3, and D_4. Therefore it contains neither of $n/4$ roots of $p(x)$ lying in the fourth domain.

Remark 1. Bound (4) on the efficiency of polynomial root-refinement has been stated in McNamee and Pan (2012) where a supporting recursive factorization was outlined, although the splitting steps relied on the computation of PFDs, as in our Section 4. We add the supporting algorithms of our Sections 5 and 6, which enable us to avoid numerical stability problems, thus decreasing the precision of computing and the Boolean (bitwise operation) complexity bounds, and we cover the issue of computing (c, d) annuli, skipped in McNamee and Pan (2012).

8 On the implementation of root refinement via recursive factorization

So far the implementation of recursive factorization for root-finding had only moderate success (due to X. Gourdon), the hardest stages been the computation of the wide separating annuli and initial factorizations. Refinement does not include the latter stage, whereas the former stage is dramatically simplified in the algorithm of the previous section. For a large class of inputs various further heuristic simplifications of this algorithm work, e.g., Flowchart 1 can immediately define the desired $(1,1)$ annuli, without invocation of Theorem 6.

The algorithm is amenable to parallel acceleration, but in that respect may be surpassed by the iterations directed to a single root such as Newton's root-finder or Inverse Power Iteration for eigen-solving for the companion matrix. Namely, assume that we concurrently apply such iteration at h crude but sufficiently

close initial approximations to h distinct roots for $1 < h \leq n$ (see Pan and Zheng (2011b)). Then the parallel processing requires no data exchange among the h processors, thus allowing acceleration of the refinement by a factor h.

Appendix

A Can one benefit from expanding the system of constraints and variables?

Pan and Zheng (2011b) suggest that the reduction of root-finding for a univariate polynomial of a degree n to the multivariate polynomial system of n Viète's (Vieta's) polynomial equations with n unknowns can explain the empirical strength of global convergence of the WDK and Ehrlich–Aberth iterations, based on such a system. The authors argue that multiple additional constraints keep the iterative process on its course to convergence stronger than the single polynomial equation can do. If true, this suggests a more general recipe of properly expanding an original system of constraints to support more reliable convergence of its iterative solution. Empirical global convergence of iterative solution of a polynomial equation is also quite strong for some companion matrix algorithms (see Pan and Zheng (2011a)). Then again the algorithms compute the solution to n constraints defining n unknowns: namely, an eigenvalue and an eigenvector of dimension n (defined up to scaling), versus the single constraint defined by a univariate polynomial equation. Yet another example is the known effect of using the duality in linear and nonlinear programming and in the solution of a multivariate system of polynomial equations (see Mourrain and Pan (2000) and Faugére (2002)). Should these examples motivate further attempts of improving global convergence of iterative solution to a system of constraints (in particular a system of multivariate polynomial equations) by means of their proper expansion with additional constraints and varaibles? The idea has strong support from many proverbs such as "One's as good as none", "There's strength in numbers", "One man does not make a team" (see more in Pan and Zheng (2011b)), but does not seem to be yet explicitly proposed in sciences.

REFERENCES

Aberth, O. (1973), Iteration Methods For Finding All Zeros of a Polynomial Simultaneously, *Mathematics of Computation*, **27**, **122**, 339–344

Aitken, A.C. (1926), On Bernoulli's numerical solution of algebraic equations, *Proc. Roy. Soc. Edin.* **46**, 289–305

Barnett, S. (1983), *Polynomial and Linear Control Systems*, Marcel Dekker, New York

Bell, E. T. (1940), *The Development of Mathematics*, McGraw-Hill, New York

Bini, D. and Pan, V. Y. (1994), *Polynomial and Matrix Computations, Vol. 1: Fundamental Algorithms*, Birkhäuser, Boston

Bini, D. and Pan, V. Y. (1996), Graeffe's, Chebyshev, and Cardinal's processes for splitting a polynomial into factors, *J. Complexity*, **12**, 492–511

Box, G. E. P. and Jenkins, G. M. (1976), *Time Series Analysis: Forecasting and Control*, Holden-Day, San Francisco

Boyer, C. A. (1968), *A History of Mathematics*, Wiley, New York

Cardinal, J. P. (1996), On two iterative methods for approximating the roots of a polynomial, *Proceedings of AMS-SIAM Summer Seminar: Mathematics of Numerical Analysis: Real Number Algorithms* (J. Renegar, M. Shub, and S. Smale, editors), Park City, Utah, 1995. *Lectures in Applied Mathematics*, **32**, 165–188, American Mathematical Society, Providence, Rhode Island

Curry, J. H. (1989), On zero finding methods of higher order from data at one point, *J. of Complexity* **5**, 219–237

Durand, E. (1960), Equations du type F(x) = 0: Racines d'un polynome, In *Solutions numeriques des equations algebriques*, volume 1, Masson, Paris

Demeure, C. J. and Mullis, C. T. (1989), The Euclid algorithm and the fast computation of cross–covariance and autocovariance sequences, *IEEE Trans. Acoust., Speech, Signal Processing* **37**, 545–552

Demeure, C. J. and Mullis, C. T. (1990), A Newton–Raphson method for moving-average spectral factorization using the Euclid algorithm, *IEEE Trans. Acoust., Speech, Signal Processing* **38**, 1697–1709

Ehrlich, L. W. (1967), A modified Newton method for polynomials, *Comm. of ACM*, **10**, 107–108

Faugére, J. C. (2002), A new efficient algorithm for computing Gröbner bases without reduction to zero (F5), in *Proceedins of the International Symposium on Symbolic and Algebraic Computation (ISSAC 2002)*, 75–83, ACM Press, NY

Gathen, J. von zur and Gerhard, J. (2003), *Modern Computer Algebra*, Cambridge University Press, Cambridge, UK (second edition)

Kantorovich, L.V. and Akilov, G.P. (1982), *Functional Analysis*, Pergamon, Oxford

Kerner, I. O. (1966), Ein Gesamtschrittverfahren zur Berechnung der Nullstellen von Polynomen, *Numerische Mathematik* **8**, 290–294

Kim, M.-H. (1985), Computational complexity of the Euler type algorithms for the roots of complex polynomials, PhD Thesis, City University of New York

Kirrinnis, P. (1998), Polynomial factorization and partial fraction decomposition by simultaneous Newton's iteration, *J. of Complexity* **14**, 378–444

McNamee, J.M. (2002), A 2002 update of the supplementary bibliography on root of polynomials, *J. Comput. Appl. Math.* **142**, 433–434; also at web-site `www.yorku.ca/~mcnamee/`

McNamee, J.M. (2007), *Numerical Methods for Roots of Polynomials (Part 1)*, Elsevier, Amsterdam

McNamee, J.M. and Pan, V.Y. (2012), Efficient Polynomial Root-refiners: a Survey and New Record Estimates, Computers and Mathematics with Applications, **63**, 239–254

Mourrain, B. and Pan, V. Y. (2000), Multivariate polynomials, duality and structured matrices, *J. of Complexity*, **16**, **1**, 110–180. (Proceedings Version in STOC'98)

Muller, D.E. (1956), A method for solving algebraic equations using an automatic computer, *Math. Tables Aids Comput.* **10**, 208–215

Neff, C. A. and Reif, J. H. (1994), An $o(n^{1+\epsilon})$ algorithm for the complex root problem, *Proc. 35th Ann. IEEE Symp. on Foundations of Computer Science (STOC'94)*, 540–547, IEEE Computer Society Press

Ostrowski, A. M. (1940), Recherches sur la méthode de Graeffe et les zéros des polynomes et des series de Laurent, *Acta Math.* **72**, 99–257

Ostrowski, A. M. (1966), *Solution of Equations and Systems of Equations*, Academic Press, New York (second edition)

Pan, V. Y. (1995), Optimal (up to polylog factors) sequential and parallel algorithms for approximating complex polynomial zeros, *Proc. 27th Ann. ACM Symp. on Theory of Computing*, 741–750, ACM Press, New York

Pan, V. Y. (1996), Optimal and nearly optimal algorithms for approximating polynomial zeros, *Computers and Math. with Applications* **31, 12**, 97–138

Pan, V. Y. (1997), Solving a polynomial equation: some history and recent progress, *SIAM Review*, **39, 2**, 187–220

Pan, V. Y. (1998), Solving polynomials with computers, *American Scientist*, **86**, January-February 1998

Pan, V. Y. (2002), Univariate polynomials: nearly optimal algorithms for factorization and rootfinding, *Journal of Symbolic Computations* **33, 5**, 701–733 (Proc. version in *ISSAC '01*, 253–267)

Pan, V. Y. (2005), Amended DSeSC Power Method for polynomial root-finding, *Computers and Math. (with Applications)*, **49**, **9**–**10**, 1515–1524

Pan, V. Y., Qian, G. and Zheng, A.–L. (2012), Randomized Matrix Methods for Real and Complex Polynomial Root-finding, Tech. Report TR 2012002, *PhD Program in Comp. Sci., Graduate Center, CUNY*, 2012

Available at http://www.cs.gc.cuny.edu/tr/techreport.php?id=429

Pan, V. Y. and Zheng, A.–L. (2011a), New progress in real and complex polynomial root-finding, *Computers and Mathematics with Applications* **61**, 1305–1334 (Proceedings version in ISSAC'10), ACM Press, New York)

Pan, V. Y. and Zheng, A.–L. (2011b), Root-finding by expansion with independent eonstraints, *Computers and Mathematics with Applications* **62**, 3164–3182 (Proceedings version in SNC'11, ACM Press, New York)

Petkovic, M.S. and D. Herceg (2001), Point estimation of simultaneous methods for solving polynomial equations: a survey, *Computers and Mathematics with Applications* **136**, 183–207

Renegar, J. (1987), On the worst-case arithmetic complexity of approximating zeros of polynomials, *J. of Complexity* **3**, 90–113

Schönhage, A. (1982), The fundamental theorem of algebra in terms of computational complexity, *Department of Math., University of Tübingen*, Germany

Smale, S. (1986), Newton's method estimates from data at one point, in *the Merging Disciplines: New Directions in Pure, Applied and Computational Math.* (edited by R.E. Ewing, K.I. Cross and C.F. Martin), 185–196, Springer

Van Dooren, P. M. (1994), Some numerical challenges in control theory, *Linear Algebra for Control Theory, IMA Vol. Math. Appl.* **62**

Weierstrass, K. (1903), Neuer Beweis des Fundamentalsatzes der Algebra, *Mathematische Werker*, Tome **I**II, Mayer und Müller, Berlin, 251–269

Wilson, G. T. (1969), Factorization of the covariance generating function of a pure moving-average process, *SIAM J. on Numerical Analysis* **6**, 1–7