The Fundamental Theorem of Algebra in Terms of Computational Complexity

– Preliminary Report –

Arnold Schönhage
Mathematisches Institut der Universität Tübingen

August 1982

Preamble. *For the time being (July 2004), this report is still "preliminary", the monograph announced on page 6 is still pending, awaiting its completion. But meanwhile there are many citations referring to this text, so it appears useful to make it available via the Internet. In editing the old typescript under LaTeX we have tried to decrease the number of typos, have eliminated two ambiguities in formula numbering, and corrected little inaccuracies in the references—but have withstood the temptation to include any mathematical 'updates' or to alter anything else, so this is still the authentic text in its 1982 version—except for an* Appendix *giving a few links to more recent literature.*

Abstract

The fundamental theorem of algebra gives rise to the following computational problem: Given any polynomial $P(z) = a_0 z^n + a_1 z^{n-1} + \cdots + a_n$ with complex coefficients and norm $\sum_k |a_k| \le 1$, and given any integer $s > 0$, how fast can we compute approximate linear factors $L_j(z) = u_j z + v_j$ $(1 \le j \le n)$ such that $|P - L_1 L_2 \cdots L_n| < 2^{-s}$ holds? — The main subject of this report is the existence of an algorithm for this task which always will compute a solution within the time bound of $O((n^3 \cdot \log n + sn^2) \cdot \log(ns) \cdot \log\log(ns))$ bit operations. The underlying *splitting circle method* combines well-known techniques from numerical analysis like Graeffe's method, discrete Fourier transforms, and Newton iteration with fast algorithms from complexity theory, especially for integer multiplication and its application to the numerical treatment of complex polynomials.

Contents

I Introduction

1 The problem

For many centuries the problem of solving polynomial equations was mainly studied in terms of *formulae*. After the early days of the 16th century, when Italian mathematicians had mastered the 3rd and 4th degree, it took more than 250 years until Gauss gave a rigorous[1] proof for the *existence* of the (possibly complex) roots of any real equation, and soon after that, Abel [1] could show that, in general, it is impossible to obtain them by merely applying arithmetic operations and successively solving *pure* equations. It is very informative to read the corresponding comments in § 9 of Gauss' dissertation [11], in particular that 'resolutio aequationis' and 'ipsius reductio ad aequationes puras' should be distinguished. Nevertheless the title of Abel's paper somehow carried on the former misunderstanding, at least verbally.

It was only after about another 100 years (and after many elegant existence proofs) that the 'solvability' of general equations with complex coefficients was adequately recognized, this time in terms of *algorithms*. In 1924, H. Weyl gave a constructive proof for the fundamental theorem of algebra [33]. In modern language of recursive analysis his reasoning essentially shows that the zeros of a complex polynomial recursively depend on its coefficients (see also [30]). In other words, there exists a Turing machine which, when fed with an integer s and sufficiently good approximations for the coefficients of some n-th degree polynomial, will output approximations for its zeros within an error bound of 2^{-s}. In terms of *computational complexity* now the main problem is: *how fast* can this be done?

In order to provide a basis for reliable answers some technical details need to be specified first. As a model of computation we choose that of multitape Turing machines; later on also *pointer machines* (cf. [23]) will be considered. Numbers shall be represented as binary integers, or as binary fractions $k/2^m$ with integers k and m. Complex numbers are pairs of binary fractions, and polynomials usually are given as coefficient vectors. For theoretical considerations the *norm* of a polynomial $P(z) = a_0 z^n + a_1 z^{n-1} + \cdots + a_n$ shall be $|P| = |a_0| + |a_1| + \cdots + |a_n|$; more about norms will be said in § 4.

As is well-known from perturbation theory, clustered zeros of a polynomial are less stable under small perturbations of the coefficients than isolated zeros. Therefore it seems to be more appropriate to understand any prescribed accuracy in the sense of backward analysis: the approximations for the zeros should be determined such that their elementary symmetric polynomials agree well enough with the coefficients of the given polynomial, provided its leading coefficient is $a_0 = 1$. Moreover also $a_0 \to 0$ shall be admitted, i. e. some of the zeros may tend to infinity. All these requirements are uniformly covered by the computational task of *approximate factorization*:

Given a polynomial $P(z) = a_0 z^n + \cdots + a_n$ *with complex coefficients and norm* $|P| \le 1$, *and given any integer* $s > 0$, *compute linear factors* $L_j(z) = u_j z + v_j$ $(1 \le j \le n)$ *such that*

$$|P - L_1 L_2 \cdots L_n| < 2^{-s} \tag{1.1}$$

is fulfilled.

[1] actually Gauss' first proof contains a gap; cf. Ostrowski's repair – Gauss, Werke, vol. 10, 2.

In analyzing the efficiency of algorithms for this task our main interest will be in their *worst case behavior*: for fixed n and s the inputs a_ν can be restricted to (complex) integer multiples of 2^{-s_0} without loss of generality, if, e. g., $s_0 = s + \lceil \log(2n+2) \rceil + 1$. With respect to these inputs there is a well-defined maximal running time $T(n, s)$ for any such algorithm.

Now the main problem can be restated more precisely: What is the true order of growth of the maximal running time $T(n, s)$ of (nearly) optimal algorithms for approximate factorization, what is the asymptotic behavior of $T(n, s)$ for $s \to \infty$, or for increasing degree n? — The answers may depend on the machine model, and it is also conceivable that there are much faster algorithms for restricted classes of inputs (e. g. for pure equations).

There are much simpler problems in computational complexity for which the corresponding 'true' asymptotics could not be determined so far. Therefore we must confine ourselves to partial answers here, for instance by deriving some good upper bound on the complexity of approximate factorization. Such a result will be presented in § 2, but first we should briefly discuss some of the partial answers which (at least implicitly) are contained in the vast literature on root-finding methods in numerical analysis.

In 1967 a symposium on constructive aspects of the fundamental theorem of algebra was held the proceedings of which [6] provide a good record of the state of the art at that time. Furthermore, chapter 6 of Henrici's book [12] gives an extensive account of existing techniques. Many of the numerical methods, however, are ruled out immediately (though perhaps widely used in practice), as they are not compatible with the strict requirements of our approach. Two examples may suffice to illustrate this point. The Jenkins-Traub method [13, 14], for instance, relies on choosing shifts ([14], p. 261) *at random*, not to speak of its other shortcomings. The other example is in [7]: at the end of the introduction, the authors comment on their 'never failing, fast convergent' root-finding algorithm in this way: "As round-off errors cannot be taken under control, no error bound for the computed approximate root can be given."

Inherent to all reliable algorithms for approximate factorization of polynomials is the use of multi-precision techniques, and the round-off errors clearly must be controlled. This may be achieved by suitable estimates or *dynamically* by interval or circular arithmetic ([10], or §§ 6.6, 6.13 of [12]), but in any case *a priori* estimates are necessary for obtaining a priori bounds on the running time. The same applies to all iterative procedures, which usually imply the additional problem of suitable initial values.

The most efficient root-finding algorithm among the reliable methods found in the literature is still based upon Weyl's exclusion technique. In § 6.11, IV of [12] Henrici describes an implementation of this idea which determines the zeros of an n-th degree polynomial within an error of less then ε by not more than $O\big(n^3 \log \frac{1}{\varepsilon}\big)$ applications of his *proximity test* T_3 (p. 508 of [12]), and for $\varepsilon = 2^{-s}$ each test requires at most $O(n^2)$ arithmetic operations, where an $O(ns)$-bit accuracy will suffice. Thus we get a rather coarse bound of $O(n^7 s^3)$ bit operations, but several improvements are possible. Application of fast integer and fast polynomial multiplication reduces to something like $O(n^{5+\delta} s^{2+\delta})$ for any $\delta > 0$. Another n^2 may be saved

by nearly sharp proximity tests, but nothing better than linear convergence can be achieved in this way, whence the factor s^2 inevitably occurs.

It is not clear whether methods based on Sturm sequences can possibly become superior. Lehmer [17, 18] and Wilf [34] both do not solve the extra problems which arise, if there is a zero on the test contour (circle or rectangle) or very close to it. There is, however, a rather efficient variant of such techniques, based on Vincent's theorem, by which Akritas [3, 3a] solves the related problem of isolating the real roots of integer polynomials. Approximate factorization of complex polynomials is certainly possible in *polynomial* time. A corresponding 'conjecture' in a recent paper ([9], p. 343) is, by the way, wrong, due to its poor formulation. The real challenge is to find more precise bounds.

2 The result

All the time bounds given in the sequel will depend on the time bound for the multiplication of long integers. In the case of multitape Turing machines multiplication of m-bit integers is possible (cf. [27, 24] or § 4.3.3 of [16]) in time

$$\psi(m) = c \cdot m \cdot \log(m+1) \cdot \log\log(m+2)\,; \tag{2.1}$$

it is not known whether this bound is asymptotically optimal. In the case of *pointer machines*, however, integer multiplication is possible in linear time [23], thus $\psi(m) = c \cdot m$. Any such bound shall fulfil some mild regularity conditions like the monotonicity of $\psi(m)/m$. Then our main result can be stated as follows.

Theorem 2.1. *There exists an algorithm for the approximate factorization of complex polynomials with maximal running time*

$$T(n,s) = O\left(n \cdot \psi(n^2 \log n) + n \cdot \psi(ns)\right). \tag{2.2}$$

Inserting (2.1) into (2.2) leads to the bound given in the Abstract.

The proof of Theorem 2.1 is by description of a corresponding algorithm. It is based upon a new approach called *splitting circle method* ("Trennkreisverfahren"). Estimating its running time is considerably simplified by using the linear bound $\psi(m) = O(m)$ of the pointer machines, just for convenience (the modifications for general ψ are rather obvious). Then our assertion reads

$$T(n,s) = O(n^3 \log n \,+\, n^2 s)\,. \tag{2.3}$$

It should be observed that small polynomials may have big factors. For $n = 2k$ the simple example

$$(z+1)^k \cdot 2^{-k}(z-1)^k = 2^{-k}(z^2-1)^k$$

shows $F \cdot G = P$ with $|P| = |G| = 1$, but $|F| = 2^{n/2}$, thus further factorization of G is required with its accuracy increased by at least $n/2$ bits; in other words, s should be of order n at least. In the light of this, the bound (2.3) is surprisingly small; any naive algorithm for multiplying n linear polynomials with n-bit precision, just in order to *check* the accuracy of a given approximate factorization, will require $O(n^4)$

bit operations. A similar bound would, for instance, result for a single application of the Schur-Cohn test.

Therefore our new method also possesses an enormous potential for the development of new efficient numerical procedures. Some basic applications like the approximate determination of the zeros, root isolation for integer polynomials and the computation of matrix eigenvalues are briefly discussed in §§ 19–21. It must be said, however, that the new method is rather involved and that many details of its implementation still need to be worked out carefully. In this report, also many of the proofs will be omitted. A full account of the new results shall be given in a monograph.

It is hard to say whether the general bound (2.2) is optimal. For increasing accuracy the second term $O(n\cdot\psi(ns))$ becomes dominant; in the case of many 'typical' polynomials it can be replaced by the smaller bound $O(\log n\cdot\psi(ns))$, but there are certain exceptionally 'bad' polynomials which may lead to the extra factor $n/\log n$. For any *fixed degree* n, however, our result is indeed optimal—except for the constants involved. This is trivial for pointer machines as our bound then becomes $O(s)$, which is not more than a constant multiple of the time needed for reading the input or printing the output. But even in the case of machine models for which the complexity of integer multiplication is unknown (e. g. for multitape Turing machines) we have the following surprising result.

Theorem 2.2. *For $s\to\infty$ and fixed $k,n\geq 2$ the complexity bounds for*

(a) *s-bit integer multiplication,*

(b) *taking k-th roots with relative error $<2^{-s}$,*

(c) *approximate factorization of n-th degree polynomials with error $<2^{-s}$,*

all have the same order of growth.[2]

Proof. In view of (a) it is sufficient to consider $k=n$. Moreover, (ms)-bit integer multiplication with any constant m has a complexity bound of the same order as (a). Thus Theorem 2.1 shows that (c) is bounded by (a), even if the error bound in (c) is replaced by 2^{-mns} with some constant m. Next we observe that approximate factorization of (z^n-u) with $2^{-n}\leq|u|\leq 2$ will yield (ms)-bit approximations for the n-th roots of u, hence (b) is bounded by (c).

For the final transition from (b) to (a) an idea due to H. Alt can be used (for $k=2$ see [4]). Assume there is a fast method for computing the n-th root of numbers u, for $\frac{1}{2}<u<2$. Then integer multiplication is reduced to squaring by means of the identity

$$xy=\tfrac{1}{2}\cdot\left(x^2+y^2-(x-y)^2\right),$$

and squaring an s-bit integer v is achieved by computing and rounding the quantity

$$\frac{n^2}{n-1}\cdot 2^{4s+2d}\left(2-\sqrt[n]{1+2^{-2s-d}\,v}\;-\;\sqrt[n]{1-2^{-2s-d}\,v}\right),$$

where d is a suitable constant and the n-th roots are computed within an error of less than 2^{-4s-4d}. □

[2]For bounding (c) by (a) again some regularity like $\psi(m)+\psi(m/2)+\psi(m/4)+\cdots\leq O(\psi(m))$ must be assumed.

With regard to the historical remarks of § 1 the conclusion is that, in terms of computational complexity, and for any fixed degree, there is no significant difference between solving pure or general polynomial equations with complex coefficients. Computationally, the solution of general equations is reduced to the more fundamental task of integer multiplication. Therefore, it also seems to be pointless to handle the 'casus irreducibilis' of Cardano's formula by means of trigonometric functions or to reduce the general equation of 5th degree to elliptic functions, since high precision evaluation of these functions is certainly not cheaper, but rather likely more expensive than integer multiplication.

3 The splitting circle method

The approximate factorization of a given n-th degree polynomial P is accomplished by approximately splitting P into two factors F of degree k, G of degree $n-k$, and then proceeding recursively (see § 5). The case $k = 1$ simply means approximate determination of a single zero and its deflation. High accuracy can be achieved fast by means of Newton iteration, if this zero is simple, well isolated, and if a suitable starting value is known. The case $k = 2$ is covered by Bairstow's method, quadratically convergent, if the corresponding 2 zeros are well separated from the $n-2$ other zeros. Here we will employ the general case (see [28, 29]) such that there are k zeros of P *inside* of a suitable 'splitting circle' while the $n-k$ other zeros lie *outside* of this circle, possibly some of them close to infinity. Moreover the zeros shall stay in some distance of the circle. It is a fundamental fact that such a circle can always be determined (§§ 7, 16, 17) except for the singular cases $P(z) \approx (az+b)^n$ which will be found out properly to be factored already.

There are stable transformations (cf. § 7) by which the general case can be reduced to the particularly clear situation of the 'unit splitting circle' $E = \{z \,\big|\, |z| = 1\}$. Due to the prior condition of a zero free annulus around E it is possible to obtain a reasonable lower bound on $\mu = \min_{z\in E} |P(z)|$, both theoretically and computationally with a rather moderate amount of work (§ 9). Based upon this quantity μ the analysis of Newton's method (§ 10) then leads to explicit bounds for its quadratic convergence. An *initial* approximate splitting with precision

$$|P - F_0 G_0| < \varepsilon_0 = \mu^4 \cdot 2^{-cn}, \tag{3.1}$$

for instance, turns out to be sufficient for that, where c is a suitable constant.

Finding such initial F_0 and G_0 is the most expensive part of our method, if no higher accuracy than $s = O(n)$ is required. It is based on the fact that the power sums $s_m = u_1^m + \cdots + u_k^m$ of the zeros $u_1, \ldots, u_k$ of P which lie inside of E are expressible by the formula

$$s_m = \frac{1}{2\pi i} \int_E \frac{P'(z)}{P(z)} z^m \, dz \,. \tag{3.2}$$

Again the zero free annulus and the lower bound μ enable us to give explicit bounds, here for the precision required for evaluating these integrals by means of N-point quadrature formulae which, of course, are accomplished by discrete Fourier transforms (§ 12). Approximations of $s_0 = k$, $s_1, \ldots, s_k$ then easily yield an approximate factor F_0 of P, and a suitable G_0 is found by polynomial divisison. This approach

has been described in [8] already, without a priori bounds, however. We will find a number of

$$N = O\left(\frac{1}{\delta} \cdot (n + \log(1/\mu))\right) \tag{3.3}$$

points to be sufficient for obtaining the initial accuracy of (3.1), where $\delta > 0$ measures the width of the zero free annulus in logarithmic scale, according to the condition

$$P(z) \neq 0 \quad \text{for} \quad e^{-\delta} < |z| < e^{\delta}. \tag{3.4}$$

It will always be possible to choose the splitting circle such that at least $\delta \geq \text{const}/n$ is guaranteed.

The time analysis of the whole factorization process by successive splittings will show that, with regard to the total running time, *balanced* splittings are most advantageous, i. e. k and $n-k$ should be of the same size, ideally $k = n/2$, but subject to the particular clustering of the zeros this may be impossible. Therefore the splitting circles are chosen such that, by combining k and δ, the number of points in (3.3) becomes smaller for smaller values of $\min\{k, n-k\}$ (§ 16). The diagnostics of the distribution of zeros necessary for this choice are obtained by means of Graeffe's method. Quite in the spirit of Ostrowski's paper [21] we will present a variant of this root squaring process especially suited for the application of fast algorithms (see §§ 14, 15). Contrary to the common use made of Graeffe's method, our primary interest is not in circles *on* which the zeros are located but in circles bounded away from all zeros, and for the latter purpose moderate precision is quite sufficient.

After this brief outline of the splitting circle method it should also be said that there are many new algorithms which had to be developed for auxiliary algorithmic problems, sometimes of considerable interest in their own right. In § 8 we list the results of [24] complemented with a fast numerical Taylor shift (analogous to the algebraic case in [2]). The Newton iteration algorithm in § 11 contains a very fast solution for a special case of partial fraction decomposition. The new algorithms for Graeffe's method and the search for splitting circles were mentioned already.

II Basic facts and techniques

4 Norms

In the linear space Π of complex polynomials (polynomial functions) and in its finite dimensional subspaces Π_n of polynomials of degree $\leq n$ the following norms will be used, where $f(z) = a_0 + a_1 z + \cdots + a_n z^n$:

$$\begin{aligned} |f|_E &= \max_{z \in E} |f(z)|, \\ |f|_2 &= \left(|a_0|^2 + \cdots + |a_n|^2\right)^{1/2}, \\ |f| &= |a_0| + |a_1| + \cdots + |a_n|. \end{aligned} \tag{4.1}$$

In some sense $|\cdot|_E$ is the most natural norm, while $|\cdot|_2$ and $|\cdot|$ are better for computations. This is even more true for

$$|f|_1 = |a_0|_1 + \cdots + |a_n|_1, \quad \text{where} \quad |a|_1 = |\text{Re } a| + |\text{Im } a| \quad \text{for} \quad a \in \mathbb{C}.$$

Though $|\cdot|_1$ is a norm with respect to real scalars only, it will certainly be useful for machine implementations. In the theoretical framework of this report $|\cdot|$ is mainly used. Observe the inequalities

$$|f|_2 \le |f|_E \le |f| \le |f|_1\,, \quad |f| \le \sqrt{n+1}\cdot|f|_2\,, \tag{4.2}$$

and the submultiplicativity of $|\cdot|_E$ and $|\cdot|$, i. e.,

$$|f{\cdot}g|_E \le |f|_E\cdot|g|_E\,, \quad |f{\cdot}g| \le |f|\cdot|g|\,, \tag{4.3}$$

also shared by $|\cdot|_1$, but not by $|\cdot|_2$. With respect to the factorization of polynomials there are several less trivial inequalities which we list here without proofs. Some of them seem to be new, others may be found in [20]. We start with the following quantitative supplement to the fundamental theorem of algebra.

Theorem 4.1. *Let $f \in \Pi_n$ be factorized as $f = \alpha L_1\cdots L_n$, with linear factors $L_j(z) = u_jz + v_j$ of norm $|L_j| = |u_j| + |v_j| = 1$ and some $\alpha \in \mathbb{C}$. Then the (unique) number $|\alpha|$ is bounded by*

$$|f| \le |\alpha| \le 2^{n-1}\cdot|f|_E \le 2^{n-1}\cdot|f|\,, \quad |\alpha| \le 2^{n-1/2}\,|f|_2\,. \tag{4.4}$$

The non-trivial upper bounds are sharp for $f(z) = z^n - 1$. The next result is of a similar type.

Theorem 4.2. *Let $f = \beta L_1L_2\cdots L_n$ be a factorization of f with factors $L_j(z) = z - v_j$ and $|v_j| \le 1$ for the small zeros of f and factors $L_j(z) = u_jz - 1$ with $|u_j| \le 1$ for other zeros of f. Then $|\beta|$ is bounded by*

$$|f|\cdot 2^{-n} \le |\beta| \le |f|_2 \le |f|\,. \tag{4.5}$$

Frequently we will need the following corollary of Theorem 4.1.

Corollary 4.3. *Let $f \in \Pi_n$ be the product of polynomials $f_1, f_2, \ldots, f_k$. Then*

$$|f_1|\cdot|f_2|\cdots|f_k| \le 2^{n-1}\cdot|f|_E \le 2^{n-1}\cdot|f|\,.$$

Proof. There are $\alpha_1, \alpha_2, \ldots$ and linear factors $L_1, L_2, \ldots$ of norm 1 such that
$f_1 = \alpha_1L_1L_2\cdots L_m$, $f_2 = \alpha_2L_{m+1}L_{m+2}\cdots$ etc.,
thus $f = f_1f_2\cdots f_k = (\alpha_1\alpha_2\cdots\alpha_k)L_1\cdots L_n$, and by Theorem 4.1

$$|f_1|\cdot|f_2|\cdots|f_k| \le |\alpha_1|\cdot|\alpha_2|\cdots|\alpha_k| = |\alpha_1\cdots\alpha_k| \le 2^{n-1}\cdot|f|_E\,. \qquad \square$$

Finally we mention the following generalization of the last inequality.

Theorem 4.4. *Let $f \in \Pi_{n+m}$ be the product $f = f_1f_2\cdots f_k\cdot g$, where $g(z) = \gamma_0 + \gamma_1z + \cdots + \gamma_mz^m$ and $\gamma_0\gamma_m \ne 0$, $k \ge 1$, $m \ge 0$. Then*

$$|f_1|\cdot|f_2|\cdots|f_k| \le \frac{2^{n-1}}{\sqrt{|\gamma_0\gamma_m|}}\,|f|_E\,, \quad \textit{also} \quad \le \frac{2^{n-1/2}}{\sqrt{|\gamma_0\gamma_m|}}\,|f|_2\,.$$

The special case is obtained by choosing $g = 1$. The general result is especially powerful for large degree m, since the only information needed about g concerns the size of its extremal coefficients. If, for instance, g is an integer polynomial, then $|\gamma_0\gamma_m| \ge 1$, thus the above inequalities yield a somewhat refined version of one of Mignotte's theorems (cf. [20]).

5 Approximate factorization by successive splittings

Given a polynomial P of degree $n \geq 2$ and some $\varepsilon,\ 0 < \varepsilon < 1,$ a factorization of P into linear factors L_j with *relative* error bound ε,

$$|P - L_1 L_2 \cdots L_n| < \varepsilon \cdot |P|\,, \tag{5.1}$$

can be obtained by successive splittings in the following way. Initial rounding replaces P by some P_1 such that $|P - P_1| < \frac{\varepsilon}{n}\,|P|$ holds. Now assume inductively that some intermediate factorization

$$|P - P_1 \cdots P_k| < \frac{k}{n}\,\varepsilon \cdot |P| \tag{5.2}$$

with $k < n$ and $\deg P_1 = m > 1$ has been reached. For the next splitting, say

$$|P_1 - FG| < \varepsilon_k \cdot |P_1|\,, \tag{5.3}$$

it will be sufficient to choose

$$\varepsilon_k = \frac{\varepsilon}{n} \cdot \frac{|P|}{|P_1| \cdot |P_2| \cdots |P_k|}\,, \tag{5.4}$$

as this implies

$$|P - FG{\cdot}P_2 \cdots P_k| < \frac{k}{n}\,\varepsilon \cdot |P| \;+\; \varepsilon_k \cdot |P_1| \cdot |P_2 \cdots P_k| \;\leq\; \frac{k+1}{n}\,\varepsilon \cdot |P|\,.$$

The quotient of norms in (5.4) may be computed dynamically, but it is important to know that it will always be greater than 2^{-n}, since (5.2) and Corollary 4.3 yield

$$|P_1| \cdot |P_2| \cdots |P_k| \;\leq\; 2^{n-1}{\cdot}\,|P_1 \cdots P_k| \;<\; 2^{n-1}(1 + \frac{k}{n}\,\varepsilon) \cdot |P| \;<\; 2^n{\cdot}\,|P|\,.$$

Thus approximate factorization with prescribed accuracy 2^{-s_0} can be achieved by successive splittings with a *uniform* relative error bound 2^{-s}, where

$$s = s_0 + \lceil \log_2 n \rceil + n\,. \tag{5.5}$$

Correspondingly, the worst case analysis of the maximal running time $T(n, s_0)$ of our algorithm is simplified by assuming that the given s_0 is increased by $n + \lceil \log n \rceil$ initially, but then the new s is held fixed for all subsequent splittings.

Let $T_1(n, s)$ denote the maximal running time of the latter process. Then we have $T(n, s_0) \leq T_1(n, s)$ and recursively

$$T_1(n, s) \;\leq\; \max_{1 \leq k < n} \left(T_2(n, k, s) + T_1(k, s) + T_1(n - k, s)\right), \tag{5.6}$$

where $T_2(n, k, s)$ shall denote the maximal running time for splitting any $P \in \Pi_n$ into factors $F \in \Pi_k$ and $G \in \Pi_{n-k}$ within relative error 2^{-s} (plus some overhead for the factorization process). Clearly the norms of P, F, G should not exceed reasonable bounds here; intermediate normalizations can be included. Observe that, even for small values of s_0, we always have $s > n$, and in later stages of the factorization splittings of m-th degree polynomials will occur, where m is rather small compared to s. Thus high precision splitting, $s > m^3$, for instance, will be needed in any case.

Finally we have to analyze the recursion (5.6). In § 16 we will arrive at a bound for $T_2(n, k, s)$ symmetric in k and $n - k$. Therefore it suffices to consider $k \leq n/2$.

Lemma 5.1. *If* $T_2(n,k,s)$ *and* $T_2(n,n-k,s)$ *are bounded by*

$$T_2(n,k,s) \le c_1 \cdot kn^2 \cdot \log n + c_2 \cdot ns \quad (\textit{for } 1 \le k \le n/2\,), \tag{5.7}$$

then the recursion (5.6) implies

$$T_1(n,s) \le \tfrac{2}{3}\, c_1 \cdot n^3 \cdot \log n \,+\, c_2 \cdot n^2 s\,. \tag{5.8}$$

Proof. We use induction in n. The case $n = 1$ is void, as $T_1(1,s)$ may be taken to be zero. For $n \ge 2$ and any $k \le n/2,\ k \ge 1$ we have $(n-k)^2 + k^2 \le n^2 - kn$ and $(n-k)^3 + k^3 \le n^3 - \frac{3}{2}\,kn^2$. Therefore (5.6) with suitable k and the induction hypothesis (5.8) for k and $n-k$ yield

$$\begin{aligned} T_1(n,s) \;\le\;& c_1 \cdot kn^2 \cdot \log n + c_2 \cdot ns \\ &+ \tfrac{2}{3}\, c_1 \cdot k^3 \cdot \log k \,+\, c_2 \cdot k^2 s \\ &+ \tfrac{2}{3}\, c_1 \cdot (n-k)^3 \cdot \log(n-k) \,+\, c_2 \cdot (n-k)^2 s \\ \le\;& \tfrac{2}{3}\, c_1 \cdot n^3 \cdot \log n \,+\, c_2 \cdot n^2 s\,. \end{aligned}$$

□

Apparently the worst case can arise if always $k = 1$. Better bounds on the running time are possible, if the polynomials to be factorized are such that more balanced splittings can be guaranteed.

6 Root bounds

For any polynomial f of degree $n \ge 1$ we will denote the moduli of its roots in *decreasing* order by

$$r_1 \ge r_2 \ge \cdots \ge r_n\,,$$

sometimes more explicitly by $r_k(f)$. The maximal modulus $r_1(f)$ is called the *root radius* of f. If the coefficients of f are numbered according to

$$f(z) = a_0 z^n + a_1 z^{n-1} + \cdots + a_n \quad (a_0 \ne 0),$$

then we have the simple bounds

$$\left|\frac{a_m}{a_0}\right| \;\le\; \binom{n}{m} \cdot r_1^m, \tag{6.1}$$

$$|f| \;\le\; |a_0| \cdot (1 + r_1)^n\,, \quad |a_0| \;\ge\; \frac{|f|}{(1+r_1)^n}\,. \tag{6.2}$$

An extensive collection of root bounds derived from the coefficient moduli can be found in § 6.4 of [12]. Here we only mention such estimates which are needed in the sequel.

Theorem 6.1. *If* $q = \max\limits_{m \ge 1} \left|\dfrac{a_m}{a_0}\right|^{1/m}$, *then*

$$\frac{1}{n}\, q \;\le\; r_1 < 2q\,, \tag{6.3}$$

and in the case of $a_1 = 0$ *more precisely*

$$\sqrt{\frac{2}{n}} \cdot q \;\le\; r_1 < 1.63 \cdot q\,. \tag{6.4}$$

For the latter bound we refer to [32]. Analogous inequalities are obtained for $r_n(f)$ by switching to the *reverse* polynomial $f^*(z) = a_n z^n + \cdots + a_0 = z^n \cdot f(1/z)$, since $r_n(f)^{-1}$ is the root radius of f^* (provided $a_n \neq 0$).

In combination with Graeffe's method we will use the following generalization of the upper bound in (6.3).

Theorem 6.2. *If* $\ell + 1 \leq n$, *and* $|a_{\ell+m}| \leq cq^m \cdot |a_\ell|$ *for* $m = 1, 2, \ldots$, *then* $r_{\ell+1} < (\ell+1)(c+1)\, q$.

Proof. By Theorem 6.4n in [12], p. 459, we get

$$|a_\ell|\, r_{\ell+1}^{n-\ell} \leq \tbinom{n}{n-\ell} |a_n| + \tbinom{n-1}{n-\ell-1} |a_{n-1}|\, r_{\ell+1} + \cdots + \tbinom{\ell+1}{1} |a_{\ell+1}|\, r_{\ell+1}^{n-\ell+1},$$

hence $x = q/r_{\ell+1}$ fulfils (where $x < 1$ may be assumed)

$$1 \leq cx\tbinom{\ell+1}{\ell} + cx^2\tbinom{\ell+2}{\ell} + \cdots + cx^{n-\ell}\tbinom{n}{\ell}, \quad 1 + c \leq c/(1-x)^{\ell+1},$$

$$\frac{1}{x} \leq \frac{\sqrt[\ell+1]{c+1}}{\sqrt[\ell+1]{c+1} - \sqrt[\ell+1]{c}} < (\ell+1)(c+1). \qquad \square$$

7 Scalings and Taylor shifts

Substitutions $z = \rho z'$ and $z = z' + u$ with $\rho > 0$, $u \in \mathbb{C}$ will be used for transforming any splitting circle into the unit circle E, and for finding suitable splitting circles. Therefore we introduce *scaling* operators A_ρ for $\rho > 0$ and *shift* operators B_u for $u \in \mathbb{C}$, defined on the space Π of all polynomials by the identities

$$(A_\rho f)(z) = f(\rho z), \quad (B_u f)(z) = f(z + u). \tag{7.1}$$

Restricted to Π_n these operators have the norms

$$\beta_n(A_\rho) = \max\{|A_\rho f| \;|\; f \in \Pi_n \text{ and } |f| \leq 1\} = \max\{1, \rho^n\}, \tag{7.2}$$

$$\beta_n(B_u) = \max\{|B_u f| \;|\; f \in \Pi_n \text{ and } |f| \leq 1\} = (1 + |u|)^n. \tag{7.3}$$

In our strategy for choosing splitting circles two *stages* are to be distinguished. In the very beginning of the factorization process the given polynomial $p \in \Pi_n$ may have an arbitrary root distribution. Therefore initial tests are performed (the details are given in § 15) which guarantee

$$r_1(p) \leq 2 \quad \text{or} \quad r_1(p) \geq 1.8 \quad \text{(possibly both)} \tag{7.4}$$

and similarly,

$$r_n(p) \geq 0.5 \quad \text{or} \quad r_n(p) \leq 0.55. \tag{7.5}$$

Observe that this kind of 'decision' making requires an accuracy of just 10 percent.

In case of $r_1(p) \leq 2$ we can immediately proceed to stage two, where polynomials with root radius ≤ 2 are handled further. In case of $r_n(p) \geq 0.5$ the reverse polynomial p^* enters stage two. Otherwise the root moduli of p have a *minimum spread* measured by

$$\ln\left(\frac{r_1(p)}{r_n(p)}\right) \geq 1.1856\ldots, \tag{7.6}$$

and this is stage one, where the splitting circle will have its center in the origin and its radius ρ chosen in $0.5 < \rho < 2$. Then splitting $P = A_\rho\, p$ at the unit circle (§§ 9–13) will yield approximate factors F and G such that $r_1(F) < 1 < r_n(G)$ and, with respect to some relative error bound ε,

$$|P - FG| < \varepsilon \cdot |P| = \varepsilon \cdot |A_\rho\, p| \tag{7.7}$$

is fulfilled. By going back with $A_\rho^{-1} = A_{1/\rho}$ we find $f = A_{1/\rho}F$, $g = A_{1/\rho}G$ such that

$$|p - fg| = \left|A_{1/\rho}(P - FG)\right| \;<\; \varepsilon \cdot \beta_n(A_{1/\rho}) \cdot \frac{|A_\rho\, p|}{|p|} \cdot |p|$$

holds. Estimating the latter quotient by $\beta_n(A_\rho)$ shows that the amplification of the relative error cannot be greater than the *condition* number

$$\beta_n(A_\rho^{-1})\beta_n(A_\rho) = \max\left\{\rho^n,\, \rho^{-n}\right\}, \tag{7.8}$$

thus the following recipe has been obtained.

Lemma 7.1. *Stage one splitting with radius ρ $(0.5 < \rho < 2)$ of a polynomial $p \in \Pi_n$ with relative error less than ε_0 can be accomplished by unit circle splitting (7.7) with error bound*

$$\varepsilon = \varepsilon_0 \cdot \min\left\{\rho^{-n},\, \rho^n\right\} \;\geq\; \varepsilon_0 \cdot 2^{-n}. \tag{7.9}$$

Moreover the resulting factors f and g —and all their further factors—have a root radius $r_1(f) < \rho < 2$, or $r_1(g^*) < \frac{1}{\rho} < 2$, respectively, thus all further splitting is in stage two.

We have skipped the additional problems which arise from the fact that the numerical computations will yield some $\hat{P}$ instead of $P = A_\rho\, p$, and $\hat{f}, \hat{g}$ instead of f, g. Deriving the corresponding error bounds will cause no new difficulties; again the methods of § 5 apply.

Stage two splitting is reduced to unit circle splitting via several scalings and shifts. Given an n-th degree polynomial p_0 with root radius $r_1(p_0) \leq 2$, an initial scaling by A_ρ reduces to $p_1 = A_\rho\, p_0$ with $r_1(p_1) \leq \frac{1}{2}$, whence ρ can be chosen in $1 \leq \rho \leq 4$. The next step is a Taylor shift $p_2 = B_u p_1$, where u is chosen as the *center of gravity* of the zeros of p_1, easily obtainable from the highest two coefficients of p_1. Thus we have

$$p_2(z) = b_0 z^n + b_2 z^{n-2} + \cdots + b_n\,, \tag{7.10}$$

and $r_1(p_2) < 1$ due to the geometry of the preceding steps. In the singular case $b_2 = \cdots = b_n = 0$ the complete factorization of p_0 is $p_0(z) = b_0 \cdot (z/\rho - u)^n$. Otherwise (cf. (6.4)) we must have $r_1(p_2) > 0$, and then the following *blow-up* technique is used. By means of a one percent approximation

$$0.98\, r < r_1(p_2) < r \;\leq\; 1 \tag{7.11}$$

we proceed with $p_3 = r^{-n} A_r\, p_2$, hence $0.98 < r_1(p_3) < 1$ is satisfied. In § 17 we will describe how to find a suitable splitting circle with respect to such a p_3, with center v and radius R such that

$$|v| = 2\,, \qquad 1 < R < 3\,. \tag{7.12}$$

Correspondingly, we then set $p_4 = B_v\, p_3\,,\ \ p_5 = A_R\, p_4\,.$ Now unit circle splitting with a relative error bound ε will yield approximate factors F and G such that

$$|p_5 - FG| < \varepsilon \cdot |p_5| \tag{7.13}$$

is satisfied. In view of $p_5 = Sp_0$ with $S = A_R B_v\, r^{-n} A_r B_u A_\rho$ retranslation will produce approximate factors of p_0, namely $f = S^{-1}F,\ \ g = S^{-1}G\,,$ together with the error estimate

$$|p_0 - fg| < \beta_n(S^{-1}) \cdot \varepsilon \cdot |p_5|\,. \tag{7.14}$$

The details of the preceding construction combined with (7.2), (7.3) show that

$$\beta_n(S^{-1}) = \beta_n(A_{1/\rho} B_{-u}\, r^n A_{1/r} B_{-v} A_{1/R}) \ \le\ (9/2)^n\,.$$

This bound can be used in (7.14), where $|p_5|$ may be determined dynamically. It remains, however, to give an a priori bound for the quotient $|p_5|/|p_0|\,.$ The first step is $|p_1| \le \beta_n(A_\rho) \cdot |p_0| \le 4^n \cdot |p_0|\,.$ Next we observe that p_1 and p_2 have the same leading coefficient, thus $|b_0| \le |p_1|\,.$ As $r \approx r_1(p_2)$ may be very small it is decisive here to exploit the individual estimation (6.1), i. e.

$$|b_m| \ \le\ |b_0| \cdot \binom{n}{m} \cdot (r_1(p_2))^m \ \le\ \binom{n}{m} \cdot r^m \cdot |p_1|\,.$$

Hereby $p_5(z) = r^{-n}\, p_2(r(v + Rz))$ gives $|p_5| \le (|v| + R + 1)^n \cdot |p_1|$ and, because of (7.12), this additional amplification is bounded by 6^n. Altogether we have obtained the following result.

Lemma 7.2. *Stage two splitting of a polynomial $p_0 \in \Pi_n$ with root bound $r_1(p_0) \le 2$ and with prescribed relative error bound ε_0 can be accomplished by unit circle splitting (7.13) with error bound*

$$\varepsilon \ =\ \varepsilon_0 \cdot \frac{2^n \cdot |p_0|}{9^n \cdot |p_5|} \ \ge\ \frac{\varepsilon_0}{108^n}\,. \tag{7.15}$$

There are obvious variations in constructing such a reduction to the unit circle case by which the pessimistic bound of 108^n may possibly be decreased to a certain extent. For the proof of Theorem 2, it is quite sufficient to know that, for the reduced task of splitting at the unit circle, only $O(n)$ bits of additional precision are required, in accordance with the increase (5.5) of the same order.

8 Basic algorithms

In [24] a new method is described by which the multiplication and division of polynomials with complex coefficients can be reduced to the multiplication of long integers in a very efficient way. The primary result of this kind concerns the numerical multiplication of polynomials (given here with the *linear* time bound for integer multiplication).

Theorem 8.1. *Multiplication of m-th degree polynomials with complex coefficients of modulus less than one in s-bit precision, where $s \ge \log(m+1)$, is possible in linear time $O(ms)$.*

An example shall illustrate the size of the integers which are involved. In factoring a polynomial of degree $n = 100$ it may occur that two polynomials of degree $m = 50$ are to be multiplied in 500-bit precision, for instance. Such a task is then reduced to one big integer multiplication $\bmod (2^N+1)$, where $N = 204800 = 25 \cdot 2^{13}$ will suffice.

The numerical *division* of polynomials, say F by G, is to be understood as the task of computing *some* Q and some R such that (for prescribed s-bit precision)

$$|F - (QG + R)| < 2^{-s} \tag{8.1}$$

is satisfied, where $\deg R < \deg G$ and $\deg Q < \deg F - \deg G$. Reasonable assumptions are $|F| \le 1$ and $1 \le |G| \le 2$. Moreover, it must be guaranteed that $G \in \Pi_m$ really does have the specified degree in a quantitative sense. In [24] we have demonstrated that this is properly achieved by bounding the root radius of G.

Theorem 8.2. *Given $F \in \Pi_{2m}$ with $|F| \le 1$ and $G \in \Pi_m$ with $1 \le |G| \le 2$ and a given bound $r \ge r_1(G)$, numerical division of F by G up to an error less than 2^{-s} is possible in time*

$$O(m(s + m + m \cdot \log(1 + r))). \tag{8.2}$$

Here this result will only be applied with some uniform bound $r = O(1)$ and for $s \ge m$. Then (8.2) simply gives the linear time bound $O(ms)$.

The main part of the algorithm behind Theorem 8.2 is the computation of a sufficiently good approximation to the exact quotient Q. Its coefficients appear in an initial segment of a Laurent series for $F(z)/G(z)$. They are obtained by means of several discrete Fourier transforms (DFTs), and any DFT

$$(x_0, \ldots, x_{n-1}) \mapsto (\hat{x}_0, \ldots, \hat{x}_{n-1}), \qquad \hat{x}_k = \sum_{j=0}^{n-1} x_j \omega^{kj}, \tag{8.3}$$

where $\omega = \exp(2\pi i/n)$, can be reduced to a convolution (cf. [16], Ch. 4.3.3—Ex. 8), i. e., to polynomial multiplication.

Theorem 8.3. *For complex inputs x_j of modulus less than one and $s \ge \log(n+1)$, the discrete Fourier transform (8.3) in s-bit precision is possible in time $O(ns)$.*

Similar bounds hold for the transformations of the preceding paragraph. While this is trivial with scaling, the *Taylor shift* requires more careful examination. For the algebraic models of computational complexity it is known that Taylor shifts are achievable within a time bound of the same order as that of polynomial multiplication (cf. [2]). The numerical treatment, however, is a bit more subtle. Given the coefficients of a polynomial $f(z) = a_0 + a_1 z + \cdots + a_n z^n$ and a complex number v, the coefficients b_j of the shifted polynomial $B_v f$ are determined by the formula

$$b_j = \sum_{k=j}^{n} a_k \binom{k}{j} \cdot v^{k-j} \qquad (0 \le j \le n).$$

By introducing the quantities $j!\cdot b_j$, $k!\cdot a_k$, and $v^m/m!$ it can be rewritten as a convolution, but these factorial factors are rather large. Therefore we introduce powers of an additional scaling parameter q and obtain the identity

$$(q^j j!)\cdot b_j = \sum_{k=j}^{n} \Big((q^k k!)\cdot a_k\Big) \cdot \Big(v^{k-j}/(q^{k-j}(k-j)!)\Big) \qquad (0 \le j \le n). \tag{8.4}$$

We choose q as an integer power of two (for practical reasons), uniquely determined by the inequality $\frac{1}{n} \le q < \frac{2}{n}$. Then the auxiliary factors $q^m m!$ in (8.4) all are bounded by $\exp(O(n))$. In this way the computation of the b's is reduced to a multiplication of two polynomials of moderate size (with regard to § 7 we may assume $|v| \le 2$).

Theorem 8.4. *Taylor shift $f \mapsto B_v f$ for $f \in \Pi_n$, $|f| \le 1$ and for $|v| \le 2$ in s-bit precision is possible in time $O(n(n+s))$.*

III Unit circle splitting

9 Lower bounds for $\mu = \min_{z \in E} |P(z)|$

The general assumptions about the n-th degree polynomial P to be splitted into two factors are as follows. P shall have k zeros inside of E and $n-k$ zeros outside of E; we may assume $k \le n/2$, otherwise the reverse polynomial P^* can be splitted. Furthermore there shall exist a zero-free annulus

$$\mathfrak{A}_\delta = \left\{ z \,\middle|\, e^{-\delta} < |z| < e^{\delta} \right\}, \tag{9.1}$$

where δ is bounded by $\frac{1}{2}\ln 3 > \delta \ge \text{const}/n$. The moduli of the zeros therefore fulfil (cf. § 6)

$$r_n \le \cdots \le r_{n+1-k} \le e^{-\delta}, \qquad e^{\delta} \le r_{n-k} \le \cdots \le r_1 . \tag{9.2}$$

The prescribed relative error bound can be made absolute by normalizing $|P| = 1$ (practically something like $1 \le |P| \le 2$ will be more convenient). Then (9.2) and Theorem 4.2 yield

$$|P(z)| \ge 2^{-n} \prod_{j=1}^{k} (1 - r_{n+1-j}) \cdot \prod_{j=1}^{n-k} (1 - r_j^{-1}) \quad \text{for } |z| = 1, \tag{9.3}$$

hence the minimum μ in the heading of this paragraph has the theoretical lower bound

$$\mu \ge 2^{-n} \left(1 - e^{-\delta}\right)^n \ge (2 \cdot e^{\delta})^{-n} \cdot \delta^n . \tag{9.4}$$

Improvements are possible, if additional information about the distribution of the r_j is available. We will come back to this point in § 16.

Very often the bound (9.4) will be much too pessimistic. For practical implementation it is therefore desirable to find out the value of μ at least approximately in any actual instance, just by computation. The decisive quantity in the time analysis of our splitting algorithm will be $\log(\frac{1}{\mu})$, which is bounded by $n \cdot \log n + O(n)$. Thus it will be quite satisfactory to determine $\log(\frac{1}{\mu})$ up to an error of size $O(\log n)$. This is achieved in the following way.
Choose $m = \lceil (\delta \cdot \log n)^{-1} \rceil$, $N = mn$, and consider the points ω^j for $0 \le j < N$ on the unit circle, where $\omega = \exp(2\pi i/N)$. Then the discrete minimum

$$\mu_0 = \min_j \left| P(\omega^j) \right| \tag{9.5}$$

is an approximation for μ, which can easily be determined by means of a discrete Fourier transform of size $N = O(n/(\delta \log n))$. In view of (9.4) and $\delta \geq \text{const}/n$ this DFT does not require more than $O(n \cdot \log n)$-bit precision. Due to Theorem 8.3, the minimum μ_0 can thus be determined sufficiently precise in time $O(n^2/\delta)$.

At this point a digression on *space requirements* seems to be useful. In the preceding algorithm storing of up to $O(n^3)$ bits may become necessary, but with respect to space a much more economic variant is at hand, which will be important for the practical implementation of our method. We substitute $j = \kappa + \lambda m$ $(0 \leq \kappa < m,\ 0 \leq \lambda < n)$ in (9.5); then the search for the minimal modulus of the values

$$P(\omega^{\kappa+\lambda m}) = \sum_{\nu=0}^{n} (a_\nu \omega^{\kappa\nu})(\omega^m)^{\lambda\nu} \tag{9.6}$$

can be performed such that κ is held fixed while λ is running. As $\omega^m = \exp(2\pi i/n)$, it is therefore possible to compute all the values (9.6) by means of m DFT's each of size n successively. In this way the space bound is reduced to $O(n^2 \log n)$, while the time bound remains $O(n^2/\delta)$.

In order to estimate $\log(\frac{1}{\mu}) - \log(\frac{1}{\mu_0}) = \log(\frac{\mu_0}{\mu})$ assume that $z \in E$ and $j < N$ are such that $\mu = |P(z)|$ and that the arc γ of E joining z and ω^j has minimal length, hence $|\gamma| \leq \pi/N$. For $t \in E$ the logarithmic derivative of P is bounded by

$$\left|\frac{P'(t)}{P(t)}\right| \leq \frac{k}{1-e^{-\delta}} + \frac{n-k}{e^{\delta}-1} \leq \frac{n}{\delta} \operatorname{ch} \delta \qquad \text{(cf. (9.2))}\,. \tag{9.7}$$

Because of $\mu \leq \mu_0$ we therefore obtain (as announced)

$$\left|\ln\left(\frac{\mu_0}{\mu}\right)\right| = \ln\left(\frac{\mu_0}{\mu}\right) \leq \ln\left|\frac{P(\omega^j)}{P(z)}\right| = \left|\operatorname{Re}\int_\gamma \frac{P'(t)}{P(t)}dt\right| \leq O\Big(\frac{\pi}{N}\cdot\frac{n}{\delta}\Big) \leq O(\log n)\,.$$

In a similar way $\ln(\mu/|P(z)|)$ can be estimated for any $z \in \overline{\mathfrak{A}}_{\delta/2}$, i. e., for $e^{-\delta/2} \leq |z| \leq e^{\delta/2}$, by integration of a bound for the logarithmic derivative analogous to (9.7) along the radial segment joining z and $z/|z| \in E$. We obtain, for instance,

$$\begin{aligned} |P(z)| &\geq \mu/(e^{\delta/2}+1)^n \qquad \text{for} \quad |z| = e^{-\delta/2}, \\ |P(z)| &\geq \mu/(1+e^{-\delta/2})^n \qquad \text{for} \quad |z| = e^{\delta/2}, \quad (\delta \leq \tfrac{1}{2}\ln 3). \end{aligned}$$

These bounds imply the following result, which will be applied in § 12.

Lemma 9.1. *Assume that $P \in \Pi_n$ has no zeros in the annulus $\mathfrak{A}_\delta$ and let $|P(z)| \geq \mu$ for $z \in E$. Then in the half width annulus $\overline{\mathfrak{A}}_{\delta/2}$ any $\hat{P} \in \Pi_n$ is bounded away from zero according to*

$$|\hat{P}(z)| \geq \mu/(e^{\delta/2}+1)^n - |\hat{P}-P| \qquad \text{for} \quad e^{-\delta/2} \leq |z| \leq e^{\delta/2}.$$

10 High accuracy splitting by Newton iteration

We keep to the assumptions about P as stated in § 9. Splitting P into two factors will be achieved by first computing an initial approximate splitting and then increasing its accuracy by Newton's method. We begin our analysis by describing such a

Newton iteration step. Corresponding error estimates will then show which initial accuracy is sufficient for subsequent quadratic convergence. Assume that there are approximate factors $F \in \Pi_k$ and $G \in \Pi_{n-k}$. $F(z) = z^k + \varphi_1 z^{k-1} + \cdots + \varphi_k$ shall have all its zeros inside of E, and $G(z) = a_0 z^{n-k} + \cdots$ shall have the same leading coefficient as P. Moreover let

$$|FG - P| \le \varepsilon, \quad \text{where} \quad 8\varepsilon \le \mu = \min_{z \in E} |P(z)| . \tag{10.1}$$

Then, by Rouché's theorem, P and FG must have the same number of zeros inside of E, whence G has all its zeros outside of E. Because of $\mu \le |P|_E \le |P| = 1$, (10.1) implies

$$|FG| \le 1 + \varepsilon \le \tfrac{9}{8} , \tag{10.2}$$

and our assumptions about the leading coefficients of F and G combined with Theorem 4.2 yield

$$|F| < 2^k, \quad |G| < \tfrac{9}{8}\, 2^{n-k}. \tag{10.3}$$

The *Newton correction* is a pair $(f, g) \in \Pi_{k-1} \oplus \Pi_{n-k-1}$ producing $F_{\text{new}} = F + f$ and $G_{\text{new}} = G + g$ (observe that the leading coefficients are held fixed). It is determined by the condition that the first order Taylor approximation of the underlying error mapping becomes zero. The image of (f, g) under this mapping is

$$F_{\text{new}} G_{\text{new}} - P = (FG - P) + fG + gF + fg , \tag{10.4}$$

lying in the space Π_{n-1}. As the only higher order term on the right-hand side of (10.4) is the second order term fg, the error mapping has a constant second derivative, and f and g are to be chosen such that the condition

$$P - FG = fG + gF \tag{10.5}$$

is fulfilled, which is equivalent to the (incomplete) partial fraction decomposition

$$\frac{P - FG}{FG} = \frac{f}{F} + \frac{g}{G} \qquad (\deg f < \deg F, \quad \deg g < \deg G) . \tag{10.6}$$

Thereby f and g are uniquely determined, since F and G have no common zero. The algorithmic problem of (approximately) *computing* f and g will be dealt with in § 11. Here we continue by first giving an outline of the theoretical error estimate. In view of (10.4), (10.5), the new error has the simple bound

$$\varepsilon_{\text{new}} = |fg| \le |f| \cdot |g| . \tag{10.7}$$

Upper bounds for $|f|$ and for $|g|$ can be derived by means of the following technical lemma, also to be used in § 11.

Lemma 10.1. *Under the previous assumptions about $P, F, G, \mu, \varepsilon$, let K, p, q be polynomials such that $K = pG + qF$ and $\deg p < \deg F$. Then*

$$|p| \le \frac{|K|}{\mu - \varepsilon} \cdot |F'| \le \frac{8}{7} \cdot \frac{|K|}{\mu} \cdot k\, |F| . \tag{10.8}$$

Proof. Consider the elementary integral representation

$$p(z) = \frac{1}{2\pi i}\int_E \frac{p(t)}{F(t)}\cdot\frac{F(z)-F(t)}{z-t}\,dt \;=\; \frac{1}{2\pi i}\int_E \frac{K(t)}{(FG)(t)}\cdot\frac{F(z)-F(t)}{z-t}\,dt\,,$$

which can be taken coefficientwise. Then estimating $|K(t)| \le |K|,\ |(FG)(t)| \ge \mu-\varepsilon$ for $t \in E$ yields (10.8). □

From (10.5) and (10.1) we thus obtain the bound

$$|f| \;\le\; \frac{8}{7}\cdot\frac{\varepsilon}{\mu}\cdot k\,|F|\,. \tag{10.9}$$

In order to find an upper bound for $|g|$ we multiply (10.5) by F and estimate

$$|gF^2| \;\le\; |F(P-FG)| \,+\, |f(FG)| \;\le\; \varepsilon\,|F| \,+\, \frac{9}{7}\cdot\frac{\varepsilon}{\mu}\cdot k\,|F|\,.$$

Now we apply Corollary 4.3 to $gF^2 \in \Pi_{n+k-1}$ and get

$$|g|\cdot|F|^2 \;\le\; 2^{n+k-2}\cdot\varepsilon\,|F|\cdot\left(1+\frac{9}{7}\cdot\frac{k}{\mu}\right).$$

Combining this with (10.7), (10.9) finally gives (after some obvious simplifications)

$$\varepsilon_{\text{new}} \;\le\; \varepsilon^2\cdot k^2\cdot 2^{n+k}/\mu^2\,. \tag{10.10}$$

In order to ensure $\varepsilon_{\text{new}} \le \varepsilon$ we should have at least $\varepsilon \le \mu^2/(k^2\cdot 2^{n+k})$. For such ε's it is also guaranteed that F_{new} again has all its zeros inside of E, since we may apply Rouché's theorem to FG compared with

$$F_{\text{new}}G = (F+f)G = FG + fG\,,$$

provided $|fG| < |(FG)(t)|$ for all $t \in E$. The latter condition is fulfilled as (10.9) and Corollary 4.3 yield

$$|fG| \;\le\; |f|\cdot|G| \;\le\; \frac{8}{7}\cdot\frac{\varepsilon}{\mu}\cdot k\,|F|\cdot|G| \;\le\; \frac{8}{7}\cdot\frac{\varepsilon}{\mu}\cdot k\cdot\tfrac{9}{8}\,2^{n-1}$$

which will be less than $\mu-\varepsilon$.

Except for the extra factors on the right-hand side, (10.10) expresses what is usually called *quadratic* convergence. We prefer to use the notion of *exponential convergence*, expressed by the condition

$$\log\left(\varepsilon_{\text{new}}^{-1}\right) \;\ge\; \lambda\log\left(\varepsilon^{-1}\right) \qquad \text{with some constant } \ \lambda > 1\,. \tag{10.11}$$

With $\lambda = 3/2$, for instance, this is equivalent to $\varepsilon_{\text{new}}^2 \le \varepsilon^3$, and squaring (10.10) thus shows that an initial approximate splitting according to

$$|FG-P| \le \varepsilon \;\le\; \mu^4/(k^4\cdot 2^{2n+2k}) \tag{10.12}$$

will be sufficient for the exponential convergence of the Newton iteration process.

11 The Newton iteration algorithm

Given P, F and G, the solution (f, g) of equation (10.5) or the partial fraction decomposition (10.6) could be found algebraically by means of Euclid's algorithm; it seems to be rather difficult, however, to implement this approach numerically in a stable way. Even if we could overcome these difficulties in case of the fast versions of Euclid's algorithm known in algebraic complexity theory, and admitted the fast algorithms of § 8 were used, still the best we could hope for would be a time bound of $O(n \cdot (\log n) \cdot s)$ for obtaining a solution in s-bit precision. It is a remarkable fact that here, in the case of Newton's method, all these problems can be circumvented by a special technique which, moreover, comes out with the better time bound $O(ns)$. The basic idea is to maintain an auxiliary polynomial H such that, approximately, $HG \equiv 1 \bmod F$, which is also subject to the iteration process, i. e. H_{new} will be computed such that—then in better approximation— $H_{\text{new}} G_{\text{new}} \equiv 1 \bmod F_{\text{new}}$ will hold.

At first we describe a *nested iteration* by which the accuracy of such an H can be increased while F and G are held fixed, with all the assumptions of § 10. Let H be given such that

$$HG \equiv 1 - D \bmod F \quad \text{with} \quad D \in \Pi_{n-1},\ H \in \Pi_{k-1}, \tag{11.1}$$

where $|D|$, of course, should be small. Then $\hat{H} \equiv H(1 + D) \bmod F$, uniquely determined by the additional condition $\hat{H} \in \Pi_{k-1}$, fulfills

$$\hat{H}G \equiv 1 - D^2 \equiv 1 - \hat{D} \bmod F \tag{11.2}$$

with a unique $\hat{D} \in \Pi_{k-1}$. Computing $\hat{H}$ and $\hat{D}$ can be done mod F; nevertheless it is advantageous to admit a D of higher degree in (11.1) at the outset of the iteration. By (11.2) we have $\hat{D} \equiv D^2 \bmod F$; hence there exists some q such that $\hat{D}G + qF = D^2 G$. Now we apply Lemma 10.1 and obtain

$$|\hat{D}| \;\le\; \frac{8}{7} \cdot \frac{|D|^2 |G|}{\mu} \cdot k\,|F| \;\le\; \frac{9}{7} \cdot \frac{k}{\mu} \cdot 2^{n-1} \cdot |D|^2.$$

From this estimate we conclude that $H \mapsto \hat{H}$ is the first step of an iteration process with exponential convergence $|\hat{D}| \le |D|^{1.5}$, provided the initial approximation in (11.1) is at least, say,

$$|D| \;\le\; \mu^2 / (k^2 \cdot 2^{2n})\,. \tag{11.3}$$

Furthermore, there is a $Q \in \Pi_{n-k-1}$ implicit in (11.1) such that $HG + QF = 1 - D$. By Lemma 10.1 and Corollary 4.3, all the norms $|H|, |HG|, |Q|$ are certainly bounded by $2^{2n} \cdot k/\mu$, which implies that carrying out such an iteration step in precision 2^{-s}, where (in view of (11.3)) s should be at least of order $O(n + \log(\frac{1}{\mu}))$, will involve only numbers whose integer parts also have length $O(s)$ at most. Finally we refer to the algorithms of § 8 and to the well-known technique of performing exponentially converging iterations in gradually increasing precision such that the whole execution time can be estimated by a geometric series.

Lemma 11.1. *Given F, G with all the assumptions of § 10, and given an H with (11.1) and (11.3), it is possible to compute a better approximation $\tilde{H}$ such that $\tilde{H}G \equiv 1 - \tilde{D} \bmod F$ with $|\tilde{D}| < 2^{-s}$ for any prescribed precision s in time $O(ns)$.*

Now we are ready to present the Newton iteration in its *algorithmic form*. The initial assumptions shall be a bit more generous than (10.12). Let F_0, G_0, H_0 be initial approximations for F, G, H as described before such that (with some ε_0)

$$|F_0G_0 - P| \;\le\; \varepsilon_0 \;\le\; \mu^4/(k^4\cdot 2^{3n+k+1})\,, \tag{11.4}$$

$$H_0G_0 \equiv 1 - D_0 \bmod F \quad \text{with} \;\; |D_0| \le \mu^2/(k^2\cdot 2^{2n})\,. \tag{11.5}$$

We explain the first step of the iteration producing F_1, G_1, H_1. At first H_0 is replaced by a better approximation H_1 such that (with some Q)

$$H_1G_0 + QF_0 = 1 - D \quad \text{with} \;\; |D| \le \varepsilon_0\,. \tag{11.6}$$

According to Lemma 11.1 this is possible in time $O(n\cdot\log(1/\varepsilon_0))$. Then (10.5) with f_0, G_0 is solved approximately by computing $f_0 \in \Pi_{k-1}$, $g_0 \in \Pi_{n-k-1}$ uniquely determined by the conditions

$$f_0 \equiv H_1P \;\equiv\; H_1(P - F_0G_0) \bmod F_0\,, \tag{11.7}$$

$$(P - F_0G_0) - f_0G_0 = g_0F_0 + R \quad \text{with} \;\; R \in \Pi_{k-1}\,. \tag{11.8}$$

This requires additions, subtractions, multiplications of polynomials and divisions by F_0, which has a root radius less than one. A $(2\log\varepsilon_0^{-1})$-bit precision should be sufficient for these computations, thus Theorems 8.1, 8.2 give the time bound $O(n\cdot\log(1/\varepsilon_0)$ again. Finally, $F_1 = F_0 + f_0$ and $G_1 = G_0 + g_0$ are obtained.

Estimating the new error is a bit more involved than in § 10. Instead of (10.7) we now get

$$|F_1G_1 - P| = \varepsilon_1 \;\le\; |R| + |f_0|\cdot|g_0|\,. \tag{11.9}$$

Combining (11.6) and (11.7) gives

$$f_0G_0 \equiv (P - F_0G_0)(1 - D) \bmod F_0\,,$$

hence $|f_0| \le \frac{8}{7}\,\frac{k}{\mu}\,\varepsilon_0(1+\varepsilon_0)|F_0|$ by Lemma 10.1. Similarly, (11.8) yields

$$RG_0 \equiv -D(P - F_0G_0)\,G_0 \bmod F_0\,,$$

hence $|R| \;\le\; \frac{8}{7}\,\frac{k}{\mu}\,\varepsilon_0^2\cdot|F_0|\cdot|G_0| \;\le\; \frac{8}{7}\,\frac{k}{\mu}\,\varepsilon_0^2\cdot 2^{n-1}(1+\varepsilon_0)\,.$
Next we multiply (11.8) by F_0 and obtain

$$\left|g_0F_0^2\right| \;\le\; |R|\cdot|F_0| \;+\; \varepsilon_0\,|F_0| \;+\; |f_0|\cdot|F_0G_0|\,,$$

and Corollary 4.3 gives $|g_0|\cdot|F_0|^2 \;\le\; 2^{n+k-2}\left|g_0F_0^2\right|$. Putting all this together (observe that $\varepsilon_0 \le \mu/2^{3n+k+1}$ by (11.4)) enables us to estimate

$$|g_0| \;\le\; \frac{6}{11}\cdot\frac{k}{\mu}\cdot 2^{n+k}\cdot\frac{\varepsilon_0}{|F_0|}\,,$$

and in accordance with (10.10),

$$\varepsilon_1 < \varepsilon_0^2\cdot k^2\cdot 2^{n+k}/\mu^2. \tag{11.10}$$

Finally we have to study how well H_1 is suited for F_1, G_1. From (11.6) we conclude that

$$H_1G_1 + QF_1 = 1 - D_1 \quad \text{with} \quad D_1 = D - H_1g_0 - Qf_0\,,$$

hence $D_1 \in \Pi_{n-1}$. In order to find an upper bound for $|D_1|$ we apply Lemma 10.1 to (11.6) and find

$$|H_1| \;\le\; \frac{8}{7}\cdot\frac{k}{\mu}\,(1+\varepsilon_0)|F_0|\,.$$

After multiplying (11.6) by F_0 we then proceed by

$$|Q|\cdot|F_0|^2 \le\; 2^{n+k-2}\,\left|QF_0^2\right| \;\le\; 2^{n+k-2}\,\big((1+\varepsilon_0)|F_0| \;+\; |H_1|\cdot|F_0G_0|\big)$$

etc., such that, after some computation,

$$|D_1| \;\le\; |D| \;+\; |H_1|\cdot|g_0| \;+\; |Q|\cdot|f_0| \;\le\; 2^{n+k+1}\cdot(k^2/\mu^2)\cdot\varepsilon_0 \tag{11.11}$$

is obtained. In virtue of (11.4) this latter bound implies $|D_1| \le \mu^2/(k^2\cdot 2^{2n})$. Altogether our analysis of the first step has thus established (11.4) and (11.5) for $F_1, G_1, H_1, \varepsilon_1$, and (10.10) guarantees exponential convergence $\varepsilon_1 \le \varepsilon_0^{1.5}$; after t steps of the iteration, we will have $\varepsilon_t \le \varepsilon_0^{1.5^t}$. For the t-th step (11.11) reads $|D_t| \le 2^{n+k+1}\cdot(k^2/\mu^2)\,\varepsilon_{t-1}$, which shows that the accuracy of H_t with respect to F_t, G_t also becomes better and better, though with some delay, and typically one or two steps of the nested iteration will suffice.

Again the execution time is bounded by a geometric series. In order to compute a splitting $|F_tG_t - P| < 2^{-s}$ the iteration is carried on until $\varepsilon_t < 2^{-s}$. Then the time bound is

$$\sum_{j=0}^{t-1} O(n\log\varepsilon_j^{-1}) \;=\; O(ns).$$

Lemma 11.2. *Given an initial splitting F_0, G_0 together with an auxiliary H_0 such that the conditions (11.4) and (11.5) are fulfilled, a more precise splitting with error bound 2^{-s} can be computed in time $O(ns)$.*

12 Initial splitting by discrete Fourier transforms

Suitable polynomials F_0, G_0 for (11.4) can be found by means of the power sum method (cf. [8]) already mentioned in the Introduction (§ 3, formula (3.2)). More precisely it will suffice to determine a good initial F_0. Then G_0 is simply found by polynomial division

$$P - F_0G_0 = R \quad \text{with} \quad R \in \Pi_{k-1}\,. \tag{12.1}$$

Let $P = FG$ be the corresponding *exact* splitting, with

$$F(z) \;=\; \prod_{j=1}^{k}(z-u_j)\,, \qquad |u_j| \le e^{-\delta} \quad \text{for} \quad j \le k\,, \tag{12.2}$$

$$G(z) \;=\; \beta\prod_{j=k+1}^{n}\left(\frac{z}{u_j}-1\right), \qquad \left|\frac{1}{u_j}\right| \le e^{-\delta} \quad \text{for} \quad j > k\,. \tag{12.3}$$

At first we want to analyze how small $|F_0 - F|$ should be. By (10.3) we get

$$|F_0G - P| \;\le\; |G|\cdot|F_0 - F| \;\le\; \tfrac{9}{8}\,2^{n-k}\cdot|F_0 - F|\,; \tag{12.4}$$

assuming this bound to be less than $\mu/8$ enables us to use (10.1) and Lemma 10.1 with F_0 instead of F. By Rouché's theorem, F_0G has exactly k zeros inside of E, hence F_0 has all its zeros inside of E, as all zeros of G lie outside of E. Furthermore (12.1) shows that $R \equiv P \equiv P - F_0G \bmod F_0$, thus we have $RG + qF_0 = (P - F_0G)G$ with some suitable q. Now Lemma 10.1, (12.4), and Corollary 4.3 yield

$$|F_0G_0 - P| = |R| \;\le\; \frac{9}{7}\cdot\frac{k}{\mu}\cdot 2^{n-k}\cdot|F_0 - F|\cdot|G|\cdot|F_0| \;\le\; \frac{9}{8}\cdot\frac{9}{7}\cdot\frac{k}{\mu}\cdot 2^{2n-k-1}\cdot|F_0 - F|\,.$$

Comparing this with (11.4) shows that

$$|F_0 - F| \;\le\; \tfrac{1}{2}\,\mu^5/(k^5\cdot 2^{5n}) \tag{12.5}$$

will suffice, also leaving a margin for the error in computing G_0. It should be remarked that here the application of Lemma 10.1 is redundant to some extent, due to the unnecessary blow-up by the factor $|G|$, but we want to avoid a tedious extra discussion of the quantity $\min_{z\in E}|F_0(z)|$.

A suitable F_0 is determined by approximately computing the power sums

$$s_m = u_1^m + \cdots + u_k^m\,, \qquad |s_m| \le k e^{-\delta m} \tag{12.6}$$

for $0 \le m \le k$. With regard to (12.3) we also consider

$$S_m = \left(\frac{1}{u_{k+1}}\right)^m + \cdots + \left(\frac{1}{u_n}\right)^m, \qquad |S_m| \le (n-k)e^{-\delta m}. \tag{12.7}$$

For $e^{-\delta} < |z| < e^{\delta}$ the Laurent expansion of $P'(z)/P(z)$ is

$$\frac{P'(z)}{P(z)} = \sum_{\nu\in\mathbb{Z}} c_\nu z^\nu = \sum_{m=0}^{\infty}\frac{s_m}{z^{m+1}} - \sum_{m=1}^{\infty} S_m z^{m-1}. \tag{12.8}$$

The desired power sums can be determined approximately by means of a DFT. Let N be greater than k and set $\omega = \exp(2\pi i/N)$. For $p = m+1$, $0 \le m \le k$, the quantities

$$W_p \;=\; \frac{1}{N}\sum_{j=0}^{N-1}\frac{P'(\omega^j)}{P(\omega^j)}\cdot\omega^{pj} \;=\; \sum_{\kappa\in\mathbb{Z}} c_{-p+\kappa N} \tag{12.9}$$

deviate from the s_m by not more than

$$\begin{aligned} |W_{m+1} - s_m| \;&\le\; \sum_{\kappa=1}^{\infty}|s_{\kappa N+m}| + \sum_{\kappa=1}^{\infty}|S_{\kappa N-m}| \\ &\le\; \left(k\cdot e^{-\delta N-\delta m} \;+\; (n-k)\cdot e^{-\delta N+\delta m}\right)/(1-e^{-\delta N})\,. \end{aligned} \tag{12.10}$$

The elementary symmetric functions of $u_1, \cdots, u_k$ appear as the coefficients of the reverse polynomial $F^*(z) = 1 - \sigma_1 z + \sigma_2 z^2 - \cdots + (-1)^k\sigma_k z^k$. They are related to the power sums by the well-known identity

$$F^*(z) \;=\; \exp\left(-\sum_{m=1}^{\infty}\frac{s_m}{m}\,z^m\right) \qquad \text{for } |z| < e^{\delta}\,. \tag{12.11}$$

The σ's are completely determined by $s_1, \ldots, s_k$. Correspondingly we may form the auxiliary function

$$A(z) = \exp\left(- \sum_{m=1}^{k} \frac{W_{m+1}}{m}\, z^m - \sum_{m=k+1}^{\infty} \frac{s_m}{m}\, z^m \right) = \sum_{j=0}^{\infty} \varphi_j\, z^j\,, \tag{12.12}$$

and our candidate for F_0 then is (observe $\varphi_0 = 1$)

$$F_0(z) = z^k + \varphi_1 z^{k-1} + \cdots + \varphi_k\,. \tag{12.13}$$

Obviously, $A = F^* \cdot \exp(D)$, where

$$D(z) = \sum_{m=1}^{k} \frac{1}{m}\,(s_m - W_{m+1})\, z^m\,,$$

hence

$$\begin{aligned} |F_0^* - F^*| \;&\le\; |F^*| \cdot \left(|D| + \tfrac{|D|^2}{2!} + \cdots + \tfrac{|D|^k}{k!} \right), \\ |F_0 - F\,| \;&\le\; 2^k \cdot |D|/(1 - |D|/2)\,. \end{aligned}$$

From (12.10) we derive $|D| \le n \cdot k \cdot e^{-\delta(N-k)}/(1 - e^{-\delta N})$, and in view of the intended accuracy (12.5) we may assume $|D| < 0.01$ and $e^{-\delta N} < 0.01$, whence

$$|F_0 - F| < 2^k \cdot n \cdot k \cdot e^{-\delta(N-k)} \cdot 100/98\,.$$

In order to guarantee (12.5) it is therefore sufficient to choose

$$N \;\ge\; k + \frac{1}{\delta}\left((5n + k + 2)\ln 2 \;+\; \ln(nk^6) \;+\; 5\ln\tfrac{1}{\mu} \right), \tag{12.14}$$

where again a margin is left for the error in computing F_0.

The preceding analysis shows us that F_0 can be computed by performing the following major algorithmic steps:

(i) choose N minimal according to (12.14),

(ii) compute $\alpha_j = P'(\omega^j)$ for $0 \le j < N$ by a DFT,

(iii) compute $\beta_j = P(\omega^j)$ for $0 \le j < N$ by a DFT,

(iv) form $\gamma_j = \alpha_j/\beta_j$ for $0 \le j < N$,

(v) compute $W_2, \ldots, W_{k+1}$ according to (12.9) from the γ's by means of a DFT,

(vi) compute $\varphi_1, \ldots, \varphi_k$ from $W_2, \ldots, W_{k+1}$.

The last step can, for instance, be furnished by the analogue of Newton's formula which follows from (12.12) and (12.13), namely

$$m\varphi_m = -(W_2\varphi_{m-1} + \cdots + W_m\varphi_1 + W_{m+1}) \qquad (1 \le m \le k)\,. \tag{12.15}$$

This recursion requires not more than $O(k^2)$ arithmetical steps. In virtue of $|P(\omega^j)| \ge \mu$, $\left|\frac{P'}{P}(\omega^j)\right| \le n/(1 - e^{-\delta})$ and with regard to the error bound (12.5) all these computations will be sufficiently precise if an s_0-bit precision is used throughout, where s_0 can be chosen such that $s_0 = O(n + \log\frac{1}{\mu})$ is maintained. As step

(i) chooses an N of order $O(\frac{1}{\delta}(n+\log\frac{1}{\mu}))$, Theorem 8.3 shows that the DFT's in steps (ii), (iii), and (v) are possible in time $O\left(\frac{1}{\delta}(n+\log\frac{1}{\mu})^2\right)$. Therefore F_0 and G_0 can be computed in time

$$O\left((k+1/\delta)\cdot(n+\log(1/\mu))^2\right). \tag{12.16}$$

In addition we have now to explain how a suitable H_0 can be found such that (11.5) is satisfied. Given F_0 and G_0, the *exact* $H\in\Pi_{k-1}$ of the partial fraction decomposition

$$\frac{1}{F_0G_0}=\frac{H}{F_0}+\frac{Q}{G_0}$$

has the integral representation

$$H(z)=\frac{1}{2\pi i}\int_E\frac{1}{(F_0G_0)(t)}\cdot\frac{F_0(z)-F_0(t)}{z-t}\,dt\,.$$

Inserting the coefficients from (12.13) yields

$$H(z)\;=\sum_{\kappa=0}^{k-1}z^\kappa\left(\sum_{\ell=1+\kappa}^{k}\varphi_{k-\ell}\,v_{\ell-\kappa-1}\right), \tag{12.17}$$

where

$$v_m=\frac{1}{2\pi i}\int_E\frac{t^m}{(F_0G_0)(t)}\,dt\qquad(0\le m\le k-1).$$

Again these quantities can approximately be determined by DFT's. As an analogue to (12.9) we introduce the quantities

$$V_p=\frac{1}{N}\sum_{j=0}^{N-1}\frac{1}{(F_0G_0)(\omega^j)}\,\omega^{pj}=\sum_{\nu\in\mathbb{Z}}d_{-p+\nu N}, \tag{12.18}$$

where the d's are the coefficients in the Laurent expansion

$$\frac{1}{(F_0G_0)(t)}=\sum_{\nu\in\mathbb{Z}}d_\nu t^\nu.$$

In view of $v_m=d_{-m-1}$ we then define

$$H_0(z)=\sum_{\kappa=0}^{k-1}z^\kappa\left(\sum_{\ell=1+\kappa}^{k}\varphi_{k-\ell}V_{\ell-\kappa}\right). \tag{12.19}$$

Apparently, the coefficients of H_0 can be obtained by a convolution of the φ's with $(V_1,\dots,V_k)$. With regard to (11.5) we estimate $|D_0|=|HG_0-H_0G_0|\le|G_0|\cdot|H-H_0|$, whence $|H-H_0|\le\mu^2/(k^2\cdot 2^{2n}\cdot|G_0|)$ will be sufficient. As the convolution can amplify the errors $|V_{m+1}-v_m|$ by a factor of $\sum|\varphi_j|=|F_0|<2^n/|G_0|$ at most, we see that

$$\sum_{m=0}^{k-1}|V_{m+1}-v_m|<\mu^2/(k^2\cdot 2^{3n}) \tag{12.20}$$

is sufficient for (11.5). Therefore everything goes as before with the computation of F_0, provided we can supplement a suitable estimate for the Laurent coefficients d_ν. Let us apply Lemma 9.1 to $\hat{P} = F_0 G_0$, where (11.4) guarantees $|\hat{P} - P| < \mu/2^{3n+2}$, hence (for $\delta \le \frac{1}{2}\ln 3$)

$$|(F_0 G_0)(t)| \;\ge\; \mu\cdot(e^{\delta/2}+1)^{-n} - \mu\cdot 2^{-3n-2} \;\ge\; \mu/3^n \quad \text{for } |t| = e^{\pm\delta/2}.$$

Contour integrations along these circles then yield

$$|d_\nu| \le \frac{3^n}{\mu}\cdot\exp(-|\nu|\cdot(\delta/2))\,.$$

Comparing this with (12.10) etc. shows that again $N = O(\frac{1}{\delta}(n+\log\frac{1}{\mu}))$ and $s_0 = O(n+\log\frac{1}{\mu})$ will suffice.

Let us now summarize the main result of this chapter on unit circle splitting by combining the DFT techniques of this paragraph with Lemma 11.2 on Newton's method as presented in §§ 10, 11.

Theorem 12.1. *Given $P \in \Pi_n$, k, δ, s, such that P has k zeros bounded by $e^{-\delta}$ and $n-k$ zeros of modulus $\ge e^{\delta}$ (where $0 < \delta \le \frac{1}{2}\ln 3$), an approximate splitting $|P - FG| < 2^{-s}\cdot|P|$ with $F \in \Pi_k$ having all its zeros inside of E and with $G \in \Pi_{n-k}$ having all its zeros outside of E can be computed in time*

$$O\left((k+1/\delta)\cdot(n+\log(1/\mu))^2\right),$$

where $\mu = \min_{z\in E}|P(z)|/|P|$.

Inserting here the lower bound (9.4) for μ produces a term $\frac{n^2}{\delta}\cdot(\log\frac{1}{\delta})^2$ which, for $\delta \sim \frac{1}{n}$, may become of order $n^3(\log n)^3$. Therefore refined methods for controlling μ will be necessary in order to obtain the better bound (2.3).

13 Algorithmic extras

In this paragraph we will comment some finer points concerning the algorithms for computing F_0 and H_0. Though not necessary for the global proof of Theorem 2.1, such additionals will greatly enhance the practical efficiency of the splitting circle method.

Our time bound on the DFT's in steps (ii), (iii), (v) was $O(Ns_0)$, where $Ns_0 \ge n^2/\delta$. With $\delta \sim 1/n$ this could imply the need for storing $O(n^3)$ bits. As in § 9 we can, however, partition these big DFT's into collections of smaller DFT's, which then can be carried out sequentially, if the whole computation of F_0 is rearranged correspondingly. In step (i) N should be chosen as a product $N = KL$, where L is of order n, for instance $n < L \le 2n$. (If convenient, L may be chosen as a power of two.) Steps (ii)–(v) are modified in the following way, where

$$P(z) = a_0 z^n + \cdots + a_n\,, \quad \omega^K = \exp(2\pi i/L)\,, \quad j = \kappa + \lambda K\,.$$

At first set $U_2 = U_3 = \ldots = U_{k+1} = 0$. Then for $\kappa = 0, 1, \ldots, K-1$, perform the following steps:

(ii) DFT: $\alpha_\lambda = P'(\omega^j) = \sum_{\nu=0}^{n-1} ((\nu+1)a_{n-\nu-1}\,\omega^{\kappa\nu})\left(\omega^K\right)^{\lambda\nu}, \quad 0 \le \lambda < L\,;$

(iii) DFT: $\beta_\lambda = P(\omega^j) = \sum_{\nu=0}^{n} (a_{n-\nu}\omega^{\kappa\nu})\left(\omega^K\right)^{\lambda\nu}, \quad 0 \le \lambda < L\,;$

(iv) $\gamma_\lambda \leftarrow \alpha_\lambda/\beta_\lambda \qquad \text{for} \quad 0 \le \lambda < L\,;$

(v) DFT: $u_p = \sum_{\lambda=0}^{L-1} \gamma_\lambda \left(\omega^K\right)^{p\lambda}, \quad 0 \le p < L\,;$

(v)′ $U_p \leftarrow U_p + u_p\,\omega^{\kappa p} \quad \text{for} \;\; 2 \le p \le k+1\,.$

Finally $W_p \leftarrow \frac{1}{N}\,U_p$ for $2 \le p \le k+1$ according to (12.9).

Hereby the space requirements are reduced to $O(ns_0) = O(n(n+\log(1/\mu)))\,.$ The computation of H_0 can be organized in the very same way.

The other point concerns step (vi) in the computation of $F_0\,.$ The time bound $O(k^2 s_0)$ for solving the recursion (12.15) is small enough not to spoil our overall bound $O\left(\frac{1}{\delta}(n+\log(1/\mu))^2\right),$ as the worst case analysis in §§ 16, 17 will show that $\frac{1}{\delta}$ can be of order $O(k)$. On the other hand k may be of order n, and then $k^2 s_0$ is of order n^3 at least, which appears to be rather disproportionate to the time bound for the other steps of that computation. There is, in fact, a faster method for solving this particular task.

Lemma 13.1. *Given the power sums $s_1, \ldots, s_k$ of the zeros of a polynomial $F(z) = z^k + \varphi_1 z^{k-1} + \cdots + \varphi_k$, which is known to have all its zeros inside of E, the coefficients $\varphi_1, \ldots, \varphi_k$ can be computed with error less than 2^{-s} in time $O(k(s+k))$.*

(In § 12 we had F_0 instead of F and the W's instead of the power sums $s_m\,.$)

Proof. We translate a formal power series approach well-known in algebraic complexity theory into numerical computations. In accordance with (12.11) the coefficients of F are determined by the condition

$$\frac{(F^*)'(z)}{F^*(z)} \;\equiv\; -s_1 - s_2 z - \cdots - s_k z^{k-1} \mod z^k\,. \tag{13.1}$$

Instead of the corresponding recursion (12.15) a Newton type iteration is used. Its last step, for instance, starts from an approximation $g(z) = \varphi_1 z + \cdots + \varphi_p z^p$ of half degree precision $p = \lfloor k/2 \rfloor$ such that

$$\frac{g'(z)}{1+g(z)} \;\equiv\; -s_1 - \cdots - s_p z^{p-1} \mod z^p \tag{13.2}$$

is fulfilled. The correction shall have the form $(1+g(z))(1+h(z)),$ where

$$h(z) = c_{p+1} z^{p+1} + \cdots + c_k z^k$$

is determined by the condition

$$\frac{g'(z)}{1+g(z)} + \frac{h'(z)}{1+h(z)} \equiv \frac{g'(z)}{1+g(z)} + h'(z) \equiv -\sum_{m=1}^{k} s_m z^{m-1} \mod z^k. \tag{13.3}$$

Here the first congruence follows from $2p+1 \geq k$, the second one represents the recipe for computing the coefficients of h', or h, respectively. Whence this last step of the iteration essentially requires one polynomial division and one polynomial multiplication both of order k in the degree.

Previous steps of the iteration will involve approximations of order z^m, where $m = \lfloor k/2^t \rfloor$. They will cause round-off errors such that, e. g., (13.2) will hold approximately only. We skip this routine matter and turn to another more delicate point. The time bound for the divisions depends on root bounds for the denominator polynomials (cf. Theorem 8.2), more precisely such root bounds must be *given* explicitly in order to achieve the time bound (8.2). Let $u_1, \ldots, u_k$ denote the zeros of F. The denominators in question are the polynomials

$$f_m(z) = 1 + \varphi_1 z + \cdots + \varphi_m z^m \quad \text{for} \quad m = \lfloor k/2^t \rfloor .$$

In particular,

$$f_k(z) = F^*(z) = \prod_{j=1}^{k} (1 - u_j z) .$$

As we are dealing with the division of power series here, the parameter r of Theorem 8.2 is to be replaced by $1/r_m$, where r_m is a lower bound for the moduli of the zeros of f_m. From the assumption $|u_j| < 1$ for all j we derive $|\varphi_\kappa| \leq \binom{k}{\kappa}$ and $|f_k(z)| \geq (1-\rho)^k$ for $|z| = \rho$ while on the other hand

$$|f_k(z) - f_m(z)| \leq \sum_{\kappa=m+1}^{k} \binom{k}{\kappa} \rho^\kappa \quad \text{for} \quad |z| = \rho .$$

It can be shown that for $\rho = r_m = \frac{1}{4} \cdot \frac{m+1}{k}$, for instance, the inequality for Rouché's theorem is valid, hence f_m has only zeros of modulus greater than r_m. In (8.2) we can therefore use $\log\left(1 + \frac{1}{r_m}\right) \leq \log \frac{5k}{m+1} \leq \log 5 + t$, and the whole iteration for computing $\varphi_1, \ldots, \varphi_k$ will be possible within time

$$O\left(\sum_t \frac{k}{2^t}\left(s + \frac{k}{2^t}\,(1 + \log 5 + t)\right)\right) = O(k(s+k)) . \qquad \square$$

IV Suitable splitting circles

14 Two variants of Graeffe's method

The nucleus of Graeffe's method is the (negative) *root squaring step* transforming any given n-th degree polynomial

$$f(z) = a_0 z^n + \cdots + a_n = a_0 \prod_{j=1}^{n} (z + u_j) \tag{14.1}$$

into a polynomial $g(z) = b_0 z^n + \cdots + b_n$ determined by the identity

$$g(-z^2) = \gamma\, f(z) f(-z) = \gamma\, a_0^2 \prod_{j=1}^{n} (-z^2 + u_j^2), \tag{14.2}$$

where $\gamma > 0$ is a scaling factor for computational convenience. The new coefficients are given by

$$b_m = \gamma(a_m^2 - 2a_{m-1}a_{m+1} + 2a_{m-2}a_{m+2} - \cdots) \tag{14.3}$$

(set $a_\nu = 0$ for $\nu < 0$ or $\nu > n$). By suitably numbering the zeros $-u_j$ of f (or $-u_j^2$ of g) according to the conventions of § 6 we see that

$$r_k(g) = (r_k(f))^2 \quad \text{for } 1 \le k \le n\,. \tag{14.4}$$

Usually this device is applied for approximately determining all moduli $r_k(f)$ at once. Starting from $f = f_0$, iteratively $f_1, f_2, \ldots$ are generated. As f_t has the zeros $-u_j^{2^t}$, the ratios r_k/r_{k+1} are rapidly amplified for increasing values of t, provided they are not too close to one. At the same time, however, the coefficients of the f_t are extremely diverging in size, which makes it rather difficult to handle the convolution (14.3) by means of the fast algorithms mentioned in § 8. Therefore we will present a modified approach focussing on one particular r_k or on a fixed radius R in order to decide which of the $r_j(f)$ are (approximately) smaller (or greater) than R.

Numerically the root squaring steps are to be carried out in some bounded precision. In order to get the errors under control we will keep to the convention (by suitable choice of the γ's in (14.2)) that the norms of the polynomials involved in the process will always stay within the bounds $\frac{1}{2} \le |f|, |g| \le O(2^n)$. Rounding to integer multiples of some 2^{-s} shall be considered as separate step in its own right, while the actual root squaring computation is carried out exactly. Correspondingly, (14.2) is replaced by the following steps:

(i) round the coefficients of f to integer multiples of 2^{-s},
thereby obtaining some $\hat{f}$ such that

$$|f - \hat{f}| < \varepsilon = (n+1)\, 2^{-s}\,; \tag{14.5}$$

(ii) compute the coefficients of $\hat{f}(z)\hat{f}(-z)$ exactly;

(iii) choose a suitable γ (now g is *defined*);

(iv) round g to precision $2^{-s'}$, thereby *computing* $\hat{g}$;

———etc.———

By Theorem 8.1, step (ii) is possible in time $O(n(s+n))$; there exists an elegant reduction to one long integer multiplication (in the framework of [24]) which nicely exploits the fact that the result contains even powers of z only. With regard to (i) and (iv) we have to find estimates for the perturbations $r_k(f) - r_k(\hat{f})$ and $r_k(g) - r_k(\hat{g})$; observe that different precision parameters $s' \neq s$ may be used. In step (iii) we will sometimes include the additional application of a scaling operator A_ρ as defined in § 7. Then we have $r_k(g) = (r_k(f))^2/\rho$.

Our strategy for finding suitable splitting circles is based on a fast algorithm for the following computational problem.

Task R. *Given an n-th degree polynomial f, a radius $R > 0$ and an error bound $\tau > 0$, find k such that*

$$r_k(f) > R\,e^{-\tau} \quad \text{and} \quad r_{k+1}(f) < R\,e^{\tau} \tag{14.6}$$

are satisfied (where formally $r_0 = \infty$, $r_{n+1} = 0$).

An initial scaling immediately reduces to the case $R = 1$, i. e., we transform f into $f_0 = \gamma_0 A_R f$, including a scaling for size by some $\gamma_0 > 0$. When applied in the sequel, typical values of τ will range from small constants like $\tau = 0.01$ to $\tau = 0.001/n$, for instance. Observe that there may exist several correct values of k satisfying (14.6), due to the built-in redundancy of this condition, in contrast to naive versions usually treated by Sturm sequence techniques like the Schur-Cohn algorithm.

By rounding the coefficients of f_0 to integer multiples of 2^{-s_0} we obtain a polynomial $\hat{f}_0$ such that

$$|\hat{f}_0 - f_0| < \varepsilon_0 = (n+1)2^{-s_0}. \tag{14.7}$$

The appropriate choice of s_0 (depending on τ) will be discussed below. Now we have to solve the problem for $\hat{f}_0$ with $R = 1$ and some $\hat{\tau} < \tau$. Possibly a suitable k can be found simply by inspection of the coefficient size of $\hat{f}_0(z) = a_0 z^n + a_1 z^{n-1} + \cdots + a_n$ in the following way. Let us measure the size of a_j by its *binary exponent* defined as

$$\beta_j = \begin{cases} \min\{\beta \in \mathbb{Z} \,\big|\, |\mathrm{Re}\ a_j| < 2^\beta \ \text{ and } \ |\mathrm{Im}\ a_j| < 2^\beta\} & \text{for } a_j \neq 0\,, \\ -\infty \quad \text{for } a_j = 0\,. \end{cases} \tag{14.8}$$

Furthermore assume that the scaling for size was by a γ_0 chosen as a power of two such that $\max_j \beta_j = 0$ is achieved. Then there exists some $k \geq 0$ such that $\beta_k = 0$ (for definiteness, choose k minimal), whence $|\hat{f}_0| \geq |a_k| \geq \frac{1}{2}$, and $|a_j| < \sqrt{2}$ for all j. In case of $k = n$, the second part of (14.6) is trivially satisfied, since $r_{n+1}(\hat{f}) = 0 < e^{\hat{\tau}}$. For $k < n$ we can apply Theorem 6.2; by setting $\ell = k$, $c = 2\sqrt{2}$, $q = 1$ we obtain $r_{k+1}(\hat{f}) < (2\sqrt{2}+1)(k+1) < 4n$, certainly bounded by $e^{\hat{\tau}}$, provided $\hat{\tau} \geq \ln(4n)$. Such a value of $\hat{\tau}$ (or τ) seems to be unreasonably large, but the point is that our algorithm will iteratively increase the value of τ by successive root squaring steps such that, eventually, $\ln(4n)$ will be surpassed. It may happen, however, that we are successful for a smaller $\hat{\tau}$ already, since Theorem 6.2 can be applied with the improved value of $q = 2^{-\lambda}$, where $\lambda = \min_{j>k}(-\beta_j/(j-k))$, which may be small enough to guarantee

$$r_{k+1}(\hat{f}) < (2\sqrt{2}+1)(k+1)\,2^{-\lambda} \leq e^{\hat{\tau}}.$$

The first part of (14.6) is treated in the very same way after switching to the reverse polynomial $\hat{f}^*$. Here and in the sequel we assume $a_0 \neq 0$ and $a_n \neq 0$. Otherwise obvious reductions to smaller degree apply.

If the preceding test fails to guarantee both parts of (14.6) for $\hat{f}_0$ and $R = 1$, then a root squaring step is performed which yields $f_1(-z^2) = \gamma_1 \hat{f}_0(z)\hat{f}_0(-z)$. Thereby the former problem is reduced to Task R for f_1 with $R = 1$ and $\tau_1 = 2\hat{\tau}$. Rounding the coefficients of f_1 to integer multiples of 2^{-s_1} produces some $\hat{f}_1$, and τ_1 is to be replaced by some $\hat{\tau}_1 < \tau_1$. Again the maximal binary exponent shall be zero by suitable choice of γ_1, etc. iteratively, producing polynomials $f_2, \hat{f}_2, \ldots, f_m, \hat{f}_m \ldots$.

In order to achieve progress by this iterative method the decisive parameters $s_m, \tau_m, \hat{\tau}_m$ shall be chosen such that $\hat{\tau}_m \geq \frac{3}{4}\tau_m$ and such that for the iterated polynomials $f_m, \hat{f}_m$ the conditions

$$(14.9) \qquad \begin{cases} r_k(\hat{f}_m) > e^{-\hat{\tau}_m} & \text{implies} \quad r_k(f_m) > e^{-\tau_m}, \\ r_{k+1}(\hat{f}_m) < e^{\hat{\tau}_m} & \text{implies} \quad r_{k+1}(f_m) < e^{\tau_m} \end{cases}$$

are satisfied for any k. Because of $\hat{\tau}_m \geq \frac{3}{4} \cdot \left(\frac{3}{2}\right)^m \cdot \tau$ the above test based on Theorem 6.2 then must succeed after at most M steps of the iteration, where $M = O(\log\log n + \log\frac{1}{\tau})$.

Let us now study the *perturbation problem* in (14.9), e. g. for $|\hat{f}_0 - f_0| < \varepsilon_0$, $\hat{\tau} = \frac{3}{4}\tau$. Let $r_1 \geq \ldots \geq r_n$ denote the moduli of the zeros v_j of f_0 and assume we had $r_k(\hat{f}_0) > e^{-\hat{\tau}}$ and $r_k \leq e^{-\tau}$ for some k. For sufficiently small values of $\varepsilon_0 = (n+1)2^{-s_0}$ this assumption will lead to a contradiction. In this way we find out how large s_0 should be.

Consider the continuous family of polynomials

$$(14.10) \qquad h_t = f_0 + t\cdot(\hat{f}_0 - f_0) \quad \text{for} \;\; 0 \leq t \leq 1,$$

all of which satisfy $|f_0 - h_t| < \varepsilon_0$. As $r_k(h_t)$ is continuous (!) in t, varying from $r_k \leq e^{-\tau} = p$ to $r_k(\hat{f}) > e^{-\frac{3}{4}\tau} = q$, we conclude that for any $r \in [p, q]$ there exists some t such that h_t has a zero z of modulus $|z| = r$, hence

$$|f_0(z)| = |f_0(z) - h_t(z)| \;\leq\; |f_0 - h_t| < \varepsilon_0 .$$

This implies (let the leading coefficient of f_0 be c)

$$\varepsilon_0 > |c| \cdot \prod_{j=1}^{n} |z - v_j| \;\geq\; |f_0| \cdot \prod_{j=1}^{n} \left(\frac{|r - r_j|}{1 + r_j}\right) > \left(\tfrac{1}{2} - \varepsilon_0\right) \cdot \prod_{j=1}^{n} \left(\frac{|r - \rho_j|}{1 + \rho_j}\right),$$

where $\rho_j = \min\{1, r_j\}$, and therefore we obtain

$$(14.11) \qquad \prod_{j=1}^{n} |r - \rho_j| < 2^{n+1} \cdot \frac{\varepsilon_0}{1 - 2\varepsilon_0} \qquad \text{for all} \quad r \in [p, q].$$

By a well-known extremum property of the n-th Chebyshev polynomial there must exist an $r \in [p, q]$ such that the left-hand side in (14.11) is not less than $(q-p)^n \cdot 2^{1-2n}$. Moreover, $e^{-\frac{3}{4}\tau} - e^{-\tau} > \frac{\tau}{4}\, e^{-\tau}$ can be used, and in this way we find that

$$(14.12) \qquad s_0 \;\geq\; n\,(5 + \log(e^{\tau}/\tau)) \;+\; \log(n+2)$$

will suffice to guarantee the first part of (14.9) for $m = 0$. The second line follows by considering reversed polynomials again.

The preceding argument based on (14.11) actually has established an error bound of order $O(\sqrt[n]{\varepsilon_0})$ for the perturbation of r_k, provided r_k is close to one. The weaker bound $O(n \cdot \sqrt[n]{\varepsilon_0})$ commonly used (cf. [22], Appendices A, B) would have led to an extra term of order $n \log n$ in (14.12).

Rewriting (14.12) for s_m and $\tau_m \geq \left(\frac{3}{2}\right)^m \cdot \tau$ and observing the time bound $O(n \cdot s_{m-1})$ for the m-th root squaring step shows us that there are two different phases of the iteration. Starting from some small τ we have

$$O\left(\log(e^{\tau_m}/\tau_m)\right) \leq O\left(\log \tfrac{1}{\tau_m}\right) \leq O\left(\log \tfrac{1}{\tau} - m \log \tfrac{3}{2}\right)$$

until $\tau_m \geq 1$ is reached. Then the second phase is entered, which is dominated by an exponential growth of the $s_{m+\nu}$ according to

$$O\left(\log(e^{\tau_{m+\nu}}/\tau_{m+\nu})\right) \leq O\left(\tau_{m+\nu}\right) \leq O\left(\left(\tfrac{3}{2}\right)^\nu\right).$$

Correspondingly, the time bounds for phase one and phase two are $O(n^2(\log \frac{1}{\tau})^2)$ and $O(n^2 \log n)$, respectively.

Theorem 14.1. *Task R with $\tau \leq 1$ is solved by the preceding algorithm in time $O\left(n^2\left(\log n + (\log \frac{1}{\tau})^2\right)\right)$.*

For $\tau = c_1 n^{-c_2}$ the second term is dominating and we have the bound $O(n^2 (\log n)^2)$. For less accuracy, however, say $\tau = c_1 (\log n)^{-c_2}$, the smaller bound $O(n^2 \log n)$ is obtained.

The other natural problem to be treated by the methods of this paragraph is, in some sense, dual to Task R, namely to find an approximation for $r_k(f)$. Clearly a search algorithm could be designed based on Task R and bisection, but there exists a more direct algorithm.

Task S. *Given an n-th degree polynomial f, an index k $(1 \leq k \leq n)$ and an error bound $\tau > 0$, find an $R > 0$ such that $R\, e^{-\tau} < r_k(f) < R\, e^{\tau}$.*

We give a brief outline of the algorithm. The main idea is to shift the focus each time before the next root squaring step is performed by application of a scaling operator A_ρ such that, naively speaking, the coefficient moduli $|a_{k-1}|$ and $|a_k|$ become equal and maximal. More precisely we start from $f = f_0$, and given f_m we set $\tilde{f}_m = \gamma_m A_{\rho_m} f_m$, and then $\hat{f}_m$ is obtained by rounding $\tilde{f}_m$ to integer multiples of 2^{-s}, where a constant s is used for all steps of the iteration. In order to explain how γ_m and ρ_m shall be chosen, let $\beta_0, \ldots, \beta_n$ denote the binary exponents of the coefficients of f_m, $\tilde{\beta}_0, \ldots, \tilde{\beta}_n$ those for $\tilde{f}_m$. Then one way to proceed would be to choose γ_m and ρ_m as rational powers of two such that $\tilde{\beta}_j \leq 0$ for all j and $\tilde{\beta}_\ell = 0$ for some $\ell \leq k-1$ and $\tilde{\beta}_h = 0$ for some $h \geq k$ is achieved. It may be more feasible, however, to choose γ_m and ρ_m as *integer* powers of two such that there exist an $\ell \leq k-1$ and an $h \geq k$ with the property that $\tilde{\beta}_\ell = 0$, $0 \leq \tilde{\beta}_h < h - \ell$, and none of the lattice points $(j, \tilde{\beta}_j)$ lies above the straight line through $(\ell, \tilde{\beta}_\ell)$ and $(h, \tilde{\beta}_h)$. Such ℓ and h can be found as vertices of the 'Newton diagram' (cf. [21]) for $\beta_0, \ldots, \beta_n$ by a little auxiliary algorithm in time $O(n \log n)$.

After this shift of focus we have $\frac{1}{2} \le |\tilde{f}_m| \le O(2^n)$, and by Theorem 6.2 applied to $\tilde{f}_m^*$ and $\tilde{f}_m$ we obtain

$$\frac{1}{(2\sqrt{2}+1)\,n} < r_h(\tilde{f}_m) \le r_k(\tilde{f}_m) \le r_{\ell+1}(\tilde{f}_m) < (2\sqrt{2}+1)\,n \cdot 2\,. \tag{14.13}$$

If the iteration would be carried out without rounding (i. e. $\hat{f}_m = \tilde{f}_m$ and $f_{m+1}(-z^2) = \tilde{f}_m(+z)\tilde{f}_m(-z)$), then $r_k(f)$ could be expressed exactly as

$$r_k(f) = \rho_0 \cdot \rho_1^{\frac{1}{2}} \cdot \rho_2^{\frac{1}{4}} \cdots \rho_m^{2^{-m}} \cdot (r_k(\tilde{f}_m))^{2^{-m}}.$$

In view of (14.13) the last factor differs from one by not more than a factor $e^{\pm\tau/2}$ after $M = \log\log n + \log\frac{1}{\tau} + O(1)$ steps of the iteration. Another $\tau/2$ is left for estimating the perturbations caused by round-off errors. It turns out that an s of order $O\left(n\,(\log n + \log\frac{1}{\tau})\right)$ will suffice. Thus

$$R = \rho_0 \cdot \rho_1^{\frac{1}{2}} \cdot \rho_2^{\frac{1}{4}} \cdots \rho_M^{2^{-M}} \tag{14.14}$$

is a solution for Task S.

Theorem 14.2. *Task S with $\tau \le 1$ is solved by the preceding algorithm in time*

$$O\left(n^2\left(\log n + \log\tfrac{1}{\tau}\right)\left(\log\log n + \log\tfrac{1}{\tau}\right)\right).$$

Corollary 14.3. *For $\tau = c\,n^{-d}$, $c, d > 0$, Task S can be solved in time $O(n^2 \log^2 n)$.*

15 A root radius algorithm

Computing an approximation for the root radius of a polynomial f or of its reverse f^* means to solve Task S for $k = 1$, or $k = n$, respectively. In § 7 we have mentioned several points (e. g. , (7.4), (7.11)) where the root radius was needed by a few percent approximation only. In that case Theorem 14.2 gives the bound $O(n^2 \log n \log\log n)$ which, however, is not good enough for the proof of Theorem 2.1. Fortunately our algorithm for Task S can be modified for $k = 1$ such that a precision s of order $O(n \log\frac{1}{\tau})$ is sufficient. For the applications of § 7 we thus get the improved bound $O(n^2 \log\log n)$.

Now all iterated polynomials f_m shall have the leading coefficient $a_0 = 1$. Therefore f_0 shall be the given f divided by its leading coefficient. Again we set $\tilde{f}_m = \gamma_m A_{\rho_m} f_m$, rounding $\tilde{f}_m$ to integer multiples of 2^{-s} gives $\hat{f}_m$, and $f_{m+1}(-z^2) = \hat{f}_m(z)\hat{f}_m(-z)$. Necessarily we must set $\gamma_m = \rho_m^{-n}$, and this time ρ_m shall be chosen as an integer power of two such that the coefficients of $\tilde{f}_m(z) = z^n + a_1 z^{n-1} + \cdots + a_n$ will have the properties

$$|\mathrm{Re}\ a_j| < 2^j \binom{n}{j}, \quad |\mathrm{Im}\ a_j| < 2^j \binom{n}{j} \quad \text{for } 1 \le j \le n\,, \tag{15.1}$$

$$|a_h| \ge \binom{n}{h} \quad \text{for some } h \ge 1\,. \tag{15.2}$$

Observe that the binomial coefficients $\binom{n}{j}$ for $0 \le j \le n$ can be computed as substrings of the binary expansion of $(2^n+1)^n$ in linear time $O(n^2)$ by means of fast integer multiplication.

The conditions (15.1), (15.2) imply $|\tilde{f}_m| < 2 \cdot 3^n$, and by (6.1) and (6.3) we obtain $1 \le r_1(\tilde{f}_m) < 4\sqrt{2} \cdot n$, whence

$$\left(r_1(\tilde{f}_M)\right)^{2^{-M}} < \left(\sqrt{32}\, n\right)^{2^{-M}} \le e^{\tau/2} \tag{15.3}$$

will be reached after $M = \log\log n + \log\frac{1}{\tau} + O(1)$ steps again. The R in (14.14) will be a sufficient approximation for $r_1(f)$ if the perturbations can be estimated such that $r_1(\tilde{f}_m)$ and $r_1(\hat{f}_m)$ differ by a factor $e^{\pm\tau/4}$ at most. Assume $r_1(\hat{f}_m) > r_1(\tilde{f}_m)$, for instance, and let z be a zero of $\hat{f}_m$ of maximal modulus. Then we estimate

$$|\tilde{f}_m(z)| = |\tilde{f}_m(z) - \hat{f}_m(z)| \le |z|^n \cdot |\tilde{f}_m - \hat{f}_m| < (r_1(\hat{f}_m))^n \cdot (n+1)\, 2^{-s},$$

$$|\tilde{f}_m(z)| \ge \prod_{j=1}^{n} \left(|z| - r_j(\tilde{f}_m)\right) \ge \left(r_1(\hat{f}_m) - r_1(\tilde{f}_m)\right)^n,$$

therefore $(n+1)\, 2^{-s} \le (\tau/4)^n$ will suffice, i. e.,

$$s \ge n \cdot \log\tfrac{4}{\tau} + \log(n+1)\,. \tag{15.4}$$

Theorem 15.1. *Computing the root radius of an n-th degree polynomial within relative error $e^{\pm\tau}$ is possible in time $O\left(n^2 \left(\log\frac{1}{\tau} + \log\log n\right) \log\frac{4}{\tau}\right)$.*

An example shall illustrate the bounds given above. For $e^\tau = 1.01$ and $n = 50$ we obtain from (15.3) and (15.4) that $M = 11$ and $s = 438$ are sufficient. This s may appear rather high at first sight, but observe that the polynomial in question could be $f(z) = (z - 1.37)^{48}(z - 0.92)^2$, for instance.

16 A balanced splitting scale

Specifying a suitable splitting circle means specifying its *center* and its *radius*, and it is in this order that these parameters are chosen when the algorithm is performed. For analyzing the effects of such choices, however, the reverse order is preferable. We begin by discussing how to choose the radius, while choosing the center will be studied in § 17. Here we assume that the center is zero and that the moduli $r_1 \ge r_2 \ge \cdots \ge r_n$ of the zeros of the given n-th degree polynomial f possess a minimum spread $\Delta = \ln(r_1/r_n)$. For stage one splitting as explained in § 7 we have $\Delta > 1.18$ by (7.6). For stage two splitting we will give a construction by which $\Delta \ge 0.3$ is guaranteed.

The time bound for unit circle splitting in Theorem 12.1 supplemented by Lemma 13.1 is

$$O\left((1/\delta) \cdot (n + \log(1/\mu))^2\right) + O(ns)\,. \tag{16.1}$$

In the worst case we must accept $\delta = O(\frac{\Delta}{n})$; on the other hand, $1/\delta = O(\frac{n}{\Delta})$ will always be achievable, and then (9.4) implies $\log\frac{1}{\mu} = O(n \log n)$. For small values

of n we can, therefore, simply determine all r_j by solving Task S for $j = 1, \ldots, n$ and then choose a maximal gap $\ln(r_k/r_{k+1}) = 2\delta$.

For general n we have to use more refined techniques in order to achieve the time bound for $T_2(k, n, s)$ indicated by Lemma 5.1. Comparing the first term on the right-hand side of (5.7) with the first term of (16.1) shows that δ should have a lower bound related to k. Correspondingly the search for a suitable radius r will be based on an appropriate scale of standard gap size. Formally such a *splitting scale* is a sequence $(\delta_1, \ldots, \delta_{n-1})$ such that $\sum(2\delta_j) \le \Delta$ is satisfied, and k together with a radius r are acceptable, if $\ln(r_k/r)$ and $\ln(r/r_{k+1})$ are bounded by $q{\cdot}\delta_k$, where $q = 0.9$, for instance, leaves a margin for small errors in solving Task S.

As a rough rule we want $1/\delta_j$ to be proportional to j, hence we could choose $\delta_j = \alpha/j = \delta_{n-j}$ for $j \le \frac{n}{2}$, where $\alpha = \text{const} \cdot \Delta/\log n$ would be a suitable factor. Then, however, δ_j would be unnecessarily small in the middle range, e. g. $\delta_{n/2} = O(\Delta/n \log n)$ instead of $O(\frac{\Delta}{n})$. Therefore we construct our splitting scale in a modified way, namely:

$$\delta_{n-j} = \delta_j = \frac{\alpha}{j} \qquad \text{for} \quad 1 \le j \le K\,, \tag{16.2}$$

$$\delta_j = \frac{\beta}{n - 2K - 1} \qquad \text{for} \quad K < j < n - K\,, \tag{16.3}$$

where the parameters K, α, β are chosen as

$$K = \left\lfloor \frac{n}{2\lfloor \log n \rfloor} \right\rfloor, \tag{16.4}$$

$$\alpha = \frac{\Delta}{8\lfloor \log n \rfloor}\,, \qquad \beta < \frac{\Delta}{4}\,. \tag{16.5}$$

We should have $n \ge 4$ at least. Summation of the gaps in the *middle range* (16.3) yields $\sum(2\delta_j) = 2\beta < \Delta/2$, and for each of the *tails* (16.2) we find the estimate

$$\sum_{j=1}^{K}(2\delta_j) = \frac{\Delta}{4\lfloor \log n \rfloor} \sum_{j=1}^{K} \frac{1}{j} \;<\; 0.7 \cdot \frac{\Delta}{4}\,. \tag{16.6}$$

The redundant factor 0.7 in (16.6) and choosing β smaller than $\Delta/4$ will turn out to be useful for obtaining a finer bound for $1/\mu$, namely

$$\log(1/\mu) = O(n)\,, \qquad \mu \;\ge\; \exp(-O(n))\,. \tag{16.7}$$

With these δ's and the bound (16.7) the first term of (16.1) indeed becomes $O(kn^2{\cdot}\log n)$, in agreement with (5.7); the uniform bound $O(n^3)$ for the middle range is even better, if k is precisely of order $O(n)$. Actually, when comparing (5.7) with (16.1) we have to observe that, with regard to the transformations of § 7, the s of (5.7) is to be increased by $O(n)$. The effect in (16.1) is only a harmless extra term of order $O(n^2)$.

Thus we are left with the problem how to *find* an acceptable pair (k, r) within time $O(kn^2 \log n)$ such that, in addition, (16.7) is satisfied. This is a typical trade-off situation: we should find a large gap (relative to the splitting scale) in order

to reduce the time needed for unit circle splitting, but we should not spend too much time on the search procedure. In the sequel we present a search algorithm mainly designed for giving an existence proof with regard to the worst case bounds mentioned above. There are several refinements useful for practical implementation which, however, must be omitted here.

The very first decision is whether k should be chosen in the middle range or in one of the tails. Let us assume that approximations $R_1 \le r_1$ and $R_n \ge r_n$ are given with $\ln(R_1/R_n) \ge \Delta$. Solving Task R for $\tau = 0.01\Delta/\lfloor \log n \rfloor$ and the two radii

$$R' = R_1 \cdot \exp(-\tfrac{1}{4}\Delta + \tau)\,, \qquad R'' = R_1 \cdot \exp(-\tfrac{3}{4}\Delta - \tau)$$

will yield corresponding indices k', k''; by Theorem 14.1 this is possible in time $O(n^2 \log n)$, which is sufficiently fast even for $k = 1$. If $k' \le K$, then we have

$$r_{K+1} \le r_{k'+1} < R'e^{\tau} = R_1 \cdot \exp(-\tfrac{1}{4}\Delta + 2\tau) \le R_1 \cdot \exp(-0.24 \cdot \Delta)\,, \tag{16.8}$$

and in this case a $k \le K$ shall be chosen. If $k' > K$ and $k'' \ge n - K$, then k will be chosen in the other tail $k \ge n - K$. In the remaining case, $k' > K$ and $k'' < n - K$ imply the inequalities

$$r_{K+1} \ge r_{k'} > R_1 \cdot \exp(-\tfrac{1}{4}\Delta)\,, \qquad r_{n-K} \le r_{k''+1} < R_1 \cdot \exp(-\tfrac{3}{4}\Delta)\,, \tag{16.9}$$

and then k will be chosen in the middle range.

Next we explain the further search for $k \le K$. The other tail is handled quite analogously. We define

$$R_m = R \cdot \exp(-2\delta_1 - 2\delta_2 - \cdots - 2\delta_{m-1}) \quad \text{for } m = 2, \ldots, K+1\,,$$

whence $R_{K+1} > R_1 \cdot \exp(-0.175 \cdot \Delta)$ by (16.6). There must exist some $k \le K$ such that $r_k \ge R_k$ and $R_{k+1} > r_{k+1}$ are satisfied. The search shall yield an index k where these inequalities are true within suitable error bounds. Observe that Task R can be used for such approximate comparisons between R_j and r_j. At first these comparisons are performed for $j = 2, 3, \ldots, M$, where $M = \min\{K+1, \lceil \log n \rceil^2\}$, for instance with precision $\tau = 0.01 \cdot \Delta/\lfloor \log n \rfloor^3$, until for the first time $R_j e^{\tau} > r_j$ is observed. In this case $k = j - 1$ is chosen, and by Theorem 14.1 the whole search then has the time bound $O(k \cdot n^2 \log n)$. Otherwise, if $r_j > R_j e^{-\tau}$ is found for all $j \le M$, then the range $M < j \le K+1$ is subdivided by $O(\log n)$ steps of binary search until $(R_{k+1}e^{\tau}, R_k e^{-\tau})$ is found as a suitable gap, where $\tau = 0.01 \cdot \Delta/n$ will suffice. In this case the search time is bounded by $O(n^2(\log n)^3)$ for some $k \ge (\log n)^2$. The radius of the splitting circle is chosen as $r = \sqrt{R_k R_{k+1}}$. Then the previous estimates imply

$$\ln(r/r_j) > 0.24 \cdot \Delta - 0.175 \cdot \Delta = 0.065 \cdot \Delta \quad \text{for } j > K\,. \tag{16.10}$$

In case of the middle range the further search for a suitable k will also be based on binary search, this time by means of Task S. Precision $\tau = 0.01 \cdot \Delta/n^2$ will be sufficient, but for simplicity let us skip these details now. We use a modified technique in order to achieve the bound (16.7). After t steps the search procedure will recursively have found an interval $[u, v]$ and indices $\ell < h$ such that the conditions

$$r_\ell \ge R_1 e^{-u}, \qquad R_1 e^{-v} \ge r_h\,, \tag{16.11}$$

$$v - u \;\geq\; \tfrac{1}{2}\Delta \cdot \frac{h-\ell}{n-2K-1} \cdot \prod_{\nu=1}^{t}(1-\theta\,2^{-\nu}) \tag{16.12}$$

are satisfied (approximately), where $\theta = 1 - 4\beta/\Delta$ (cf. (16.5)). The initial values of these parameters for $t = 0$ are given by (16.9). If $h - \ell \geq 2$, then the next step of the recursion is based on the (approximate) determination of r_m, where $M = \ell + \lfloor (h-\ell)/2 \rfloor$. Correspondingly set $w = u + \frac{m-\ell}{h-\ell}\,(v-u)$. If $r_m < R_1 e^{-w}$, then m shall replace h (otherwise ℓ), but instead of simply replacing v (otherwise u) by w we replace v by the slightly smaller value

$$v' = u + (w-u)(1-\theta\,2^{-t-1})\,,$$

or otherwise u by the greater value

$$u' = v - (v-w)(1-\theta\,2^{-t-1})\,,$$

respectively. In this way (16.11) and (16.12) are preserved inductively. When $h = \ell + 1$ has been reached, the final choice is $k = \ell$ and $r = R_1 \cdot \exp(-(u+v)/2)$.

Now we come to the proof of (16.7). By translating (9.3) into the language of this paragraph we obtain the lower bound

$$\mu \;\geq\; 2^{-n} \prod_{j=1}^{n} (1 - \exp(1 - |\ln r - \ln r_j|)\,)\,. \tag{16.13}$$

If some $k \leq K$ has been chosen, then these factors are greater than $1 - \exp(-0.065 \cdot \Delta)$ for $j > K$, due to (16.10), and for $j \leq K$ the trivial bound

$$|\ln(r/r_j)| \;\geq\; 0.9 \cdot \delta_k \;\geq\; 0.9 \cdot \frac{\Delta}{4n}$$

yields another estimation factor

$$\left(1 - \exp\left(-\frac{\text{const}}{n}\right)\right)^{K} \;\geq\; \left(\frac{\text{const}}{n}\right)^{O(n/\log n)} \;\geq\; \exp(-O(n))\,,$$

whence (16.7) is satisfied with an implicit constant depending on Δ.

In case of a k in the middle range we have to trace the search procedure which led to its choice. Let n_t denote the number of indices j between ℓ and h after the t-th step, i. e., $n_t = h - \ell - 1$ and $n_0 = n - 2K - 2$. The construction given above implies $n_{t+1} \leq n_t/2$, $n_{t+1} + 1 \geq \lfloor (n_t+1)/2 \rfloor$, hence $n_t + 1 \geq (n_0+1)/2^{t+1}$. In the t-th step the contraction of the search interval by the factor $(1 - \theta\,2^{-t})$ generates an extra gap between $\ln r$ and $\ln r_j$ for $n_{t-1} - n_t$ indices j which are excluded by the t-th step, and the size of this gap has the lower bound

$$\frac{n_t + 1}{n_0 + 1} \cdot 2\beta\theta \cdot 2^{-t} \geq \beta\theta \cdot 2^{-2t},$$

as can be deduced from (16.12). Eventually all j's between $k+1$ and $n-K$ will be excluded in this way, therefore

$$\prod_{j=K+2}^{n-K-1} |\ln(r/r_j)| \;\geq\; (\beta\theta)^{n_0} \prod_{t \geq 1} 2^{-2t(n_{t-1}-n_t)} \;\geq\; (\beta\theta \cdot 2^{-4})^{n_0}.$$

For the other values of j again the trivial bound $|\ln(r/r_j)| \geq 0.9 \cdot \delta_k \geq 0.9 \cdot \beta/n$ can be used. Combining these estimates with (16.13) finally yields (16.7).

Altogether we have now established the bound (5.7) for $T_2(k,n,s)$ provided we can always find a suitable center for the splitting circle such that a minimum spread Δ is guaranteed.

17 How to find a suitable center

In § 7 we have already explained that the stage one splitting circle has its center in the origin. Now we will supply the details of choosing the center v of a suitable circle for stage two splitting, as announced in (7.12). In view of the discussion after (7.11) we may assume that the given polynomial p has a root radius $r_1(p) \in (0.98, 1)$ and that the center of gravity of its zeros $u_1, \ldots, u_n$ lies in the origin. Consequently the set of the u's has a *diameter* greater than 0.98, hence it should be possible to find a new center v such that

$$\Delta = \ln \max_j |v - u_j| - \ln \min_j |v - u_j| \tag{17.1}$$

is not too small. There are many obvious variations in constructing such a center v. Our algorithm does not yield the optimal constants. It is mainly designed for clarity and ease of presentation including the condition (7.12).

As candidates for v the four points $v_0 = 2$, $v_1 = 2i$, $v_2 = -2$, $v_3 = -2i$ are considered. Four Taylor shifts (see (7.1) and Theorem 8.4) yield $f_m = B_{v_m} p$ for $0 \leq m \leq 3$. Next $r_1(f_m)$ and $r_n(f_m) = 1/r_1(f_m^*)$ are computed for each m by the method of § 15 within relative error $e^{\pm\tau}$, say with $\tau = 0.001$. By Theorem 15.1 this is possible in time $O(n^2 \log\log n)$. Finally we choose m ($v = v_m$) such that $r_1(f_m)/r_n(f_m)$ is maximal.

Let u_1 be a zero of maximal modulus and let $\nu \in \{0,1,2,3\}$ be such that $|u_1 - v_\nu|$ is minimal. Then an easy geometric argument (see Fig. 17.1) shows that our algorithm will determine a center v_m ($v_m = v_\nu$, for instance) such that $\Delta = \ln(r_1(f_m)/r_n(f_m)) > 0.30$ is guaranteed because of $r_n(f_\nu) < R_0 < 1.48$ and $r_1(f_\nu) \geq 2$ (otherwise the origin could not be the center of gravity).

By a similar argument it can be shown that only considering the candidates v_0, v_1 will also yield a minimum spread Δ. On the other hand, testing an increasing number of v's will guarantee a greater lower bound for Δ. This is another trade-off situation that needs further inspection with regard to practical implementation of the splitting circle method.

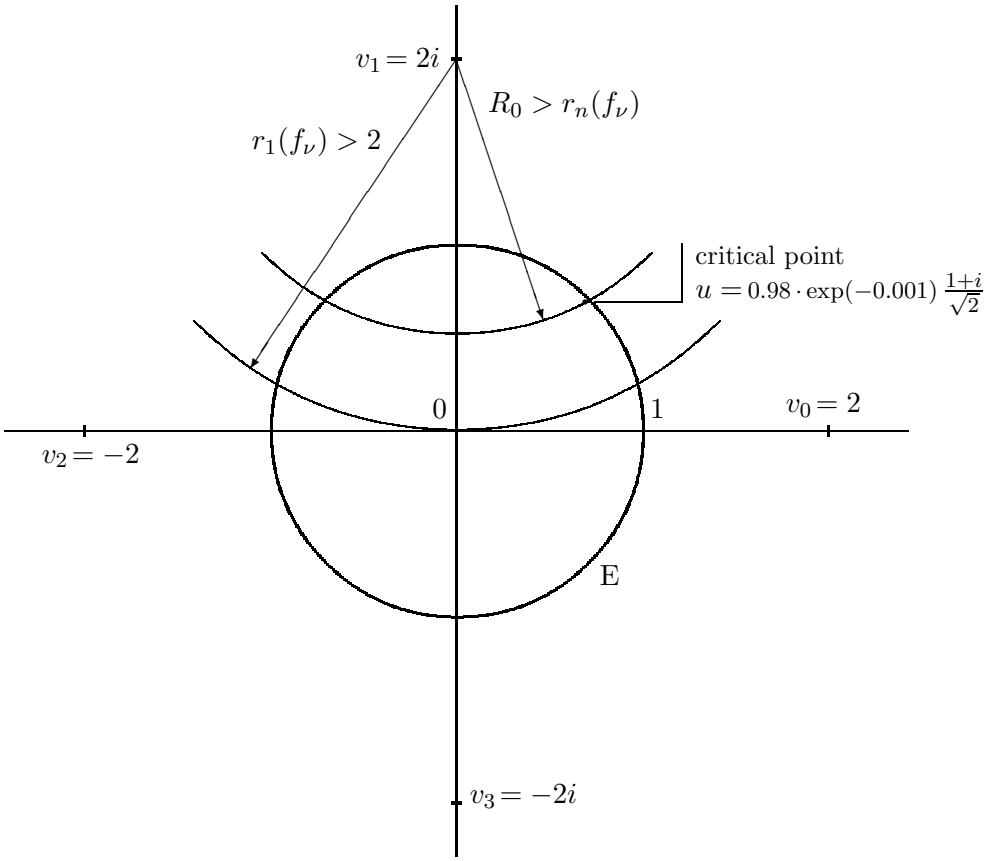

Figure 17.1. New center $v \in \{2, 2i, -2, -2i\}$ for obtaining minimum spread $\Delta = \ln(r_1/r_n) > 0.30$.

18 Conclusion

By the preceding paragraphs we have completed the description of the splitting circle method and the proof of Theorem 2.1 according to the brief outline given in § 3. Our extra discussions on space requirements have established the bound $O(n^2 \log n)$ in § 9, the bound $O\,(n(n + \log(1/\mu))) = O(n^2)$ in § 13 for initializing the Newton iteration, which itself is certainly possible in space $O(n(s+n))$. The diagnostics by Graeffe's method (§§ 14, 15) as applied in §§ 16, 17 require space $O(n^2 \log n)$ again. More precisely we have to relate these bounds to the underlying machine model and to the method used for multiplying long integers. Thus we have the following result.

Supplement 18.1. *The time bound (2.2) of Theorem 2.1 can be achieved by an algorithm with a space bound of the same order as the space bound for any algorithm for multiplying integers of length* $(ns + n^2 \log n)$*, which has the time bound* ψ*.*

In the case of multitape Turing machines, for instance, the Schönhage-Strassen integer multiplication can be implemented in linear space. In the case of pointer machines possibly another factor $\log n$ can be saved by counting nodes instead of bits.

Reexamination of the proof of Lemma 5.1 and the details of our constructions in § 16 show that balanced splitting is indeed favorable. If, for instance, all successive splittings occur with a degree ratio in $[\frac{1}{4}, \frac{3}{4}]$, i. e. $\frac{n}{4} \le k \le \frac{3}{4}n$ etc., then we will get the improved time bound $O(n^3 + sn \log n)$. On the other hand there do exist hard polynomials enforcing rather unbalanced splittings. A whole family of such examples is given by

$$f(z) = \prod_{j=0}^{n-1}(z + x_j \cdot 4^{-j}), \quad \text{where } 1 \le x_j \le 1.01\,, \tag{18.1}$$

for instance; here $s = 2n^2$ specifies a precision well tuned with respect to the coefficient size. Any strategy for choosing the splitting circles will end up with some $(k, n-k)$ and gap size δ, where n^2/δ is of order $n^2\, 4^k$ at least. As $O(sn^2) = O(n^4)$ should not be exceeded, we must have $k \le O(\log n)$. Moreover both factors are of the same type (18.1) again.

The rigidity of a polynomial expressing itself by the impossibility of balanced splittings forms a completely new phenomenon in the numerical treatment of polynomials. At present it cannot be decided whether this property of being *strongly nested* represents a profound invariant of complexity theory. Possibly it is just an artificial byproduct of the splitting circle method. In any case it should properly be distinguished from the occurrence of clustered zeros which, in itself, does not cause extra difficulties for the approximate factorization of a polynomial, due to the blow-up technique of § 7 (cf. the transition from p_2 to p_3). It is the stability of the zeros which is affected by the presence of clusters. This will be discussed in § 19.

It is rather unlikely that there exists a variant of the splitting circle method essentially faster for the subclass of real polynomials. Examples like

$$f(z) = (z^2 - 2z + 2)^{30} + 2^{-300}$$

having two conjugate clusters of zeros show on the contrary that the best way of dealing with such a polynomial may be its decomposition into two non-real factors.

V Applications

19 Approximate determination of the zeros

In the Introduction we have already mentioned that, from a general point of view, the adequate computational form of the fundamental theorem of algebra seems to be the task of approximate factorization (see (1.1)). Let us now study its applicability to the approximate determination of the zeros.

For a polynomial $P \in \Pi_n$ with leading coefficient one and bounded root radius, say $r_1(P) < 1$, the determination of its zeros can be understood as the following task: Given an error bound $\eta > 0$, compute complex numbers $v_1, \dots, v_n$ such that the zeros $z_1, \dots, z_n$ of P (after suitable numbering) satisfy

$$|v_j - z_j| < \eta \quad \text{for } 1 \le j \le n\,. \tag{19.1}$$

Such v's can, of course, be found by computing an approximate factorization F of P, where $F(z) = (z - v_1) \cdots (z - v_n)$ and $|P - F| < \varepsilon$. The crucial question is how to choose ε in dependence of η and of P.

For general P, possibly having a very small leading coefficient and rather large zeros, we should perhaps relate the task of zero determination to the normal form of linear factors as used in Theorem 4.2, i. e., for any zero z_j of modulus $|z_j| > 1$ there should be a factor $(u_j z - 1)$ in the approximate factorization with $|u_j - 1/z_j| < \eta$. This more general task can be reduced to the special case mentioned before either by application of a scaling operator A_ρ with $0.5 \le \rho \le 2$ or by one splitting step, called stage one splitting in § 7. The analysis of § 10 shows that an approximate splitting

$$|P - FG| < \varepsilon \le \mu^2/(k^2\, 2^{n+k+1}) \tag{19.2}$$

will imply the first Newton correction of F to be bounded by $|f| \le \frac{8}{7} \cdot \frac{\varepsilon}{\mu} \cdot k\,|F|$ (see (10.9)), and further corrections can be estimated by a geometric series; whence in the limit the factor F_∞ of P is obtained such that

$$|F - F_\infty| < \tfrac{16}{7} \cdot \tfrac{\varepsilon}{\mu} \cdot k \cdot 2^k < \varepsilon \cdot 2^{O(n)} \tag{19.3}$$

holds, provided $\mu > \exp(-O(n))$ is observed (see (16.7)). The precision in (19.2) is less than that required by the factorization process itself. In view of (19.3) an additional precision of $O(n)$ bits is required, also staying within the previous bounds. The factors of F give the approximations for the zeros of P lying inside of the splitting circle. By considering the reverse polynomial P^* the reciprocals of the large zeros are approximately found from the factors of G, correspondingly. The error estimate its then to be based on a bound for $|G^* - G^*_\infty|$ similar to (19.3).

Let us now return to the special case of $P(z) = \prod(z - z_j)$ with $|z_j| < 1$ and $F(z) = \prod(z - v_j)$, where $|P - F| < \varepsilon$. The zeros z_j are elements of the open set

$$W(\varepsilon, F) = \{z \,\big|\, |F(z)| < \varepsilon \ \text{ and } \ |z| < 1\}. \tag{19.4}$$

By a continuity argument applied to $F + t(P - F)$, $0 \le t \le 1$, we have the more precise statement that, with regard to the decomposition of $W(\varepsilon, F)$ into its *components* $W_1, \ldots, W_k$, each W_κ contains the same number of v's, or z's, respectively, namely

$$m_\kappa = \#\{j \,\big|\, z_j \in W_\kappa\} = \#\{j \,\big|\, v_j \in W_\kappa\}. \tag{19.5}$$

In the sequel we assume a numbering of the z's such that $v_j \in W_\kappa$ implies $z_j \in W_\eta$. Then $|v_j - z_j|$ is certainly bounded by the diameter of W_κ, which can be estimated in the following way. Any two points $u, z \in W_\kappa$ can be connected by an arc within W_κ. For any $x \in [0, r]$, $r = |z - u|$, there is some point $u + xe^{i\alpha}$ on this arc, hence

$$\varepsilon > |F(u + xe^{i\alpha})| = \prod_{j=1}^{n} |(u - v_j) + xe^{i\alpha}| \ge \prod_{j=1}^{n} |x - |u - v_j|\,|\,. \tag{19.6}$$

Due to the extremum property of the n-th Chebyshev polynomial there is some $x \in [0, r]$ for which the right-hand side of (19.6) is not less than $r^n/2^{2n-1}$. In this way we obtain $|z - u| < \left(\varepsilon \cdot 2^{2n-1}\right)^{1/n}$, thus the diameter of W_κ is less than $4\sqrt[n]{\varepsilon}$. By choosing u as any of the v's in W_κ a slightly sharper bound can be derived.

Theorem 19.1. *If a polynomial P with root radius $r_1(P) \le 1$ has an approximate factorization $F(z) = (z - v_1) \cdots (z - v_n)$ such that $|P - F| < \varepsilon$, then the zeros $z_1, \ldots, z_n$ of P can be numbered such that, for $j = 1, 2, \ldots, n$,*

$$|v_j - z_j| < \gamma_n \sqrt[n]{\varepsilon}\,, \quad \text{where } \gamma_n = (1 + \cos(\pi/2n)) \cdot 2^{1-1/n} < 4\,. \tag{19.7}$$

If the bound on the root radius of P is some $\rho > 1$, then $\sqrt[n]{\varepsilon}$ is to be replaced by $\rho \sqrt[n]{\varepsilon}$. Therefore $\rho \le 2$ will cause another constant factor at most.

Consider a prescribed accuracy $\eta = 2^{-\ell}$. The preceding analysis has shown that an approximate factorization within error bound $\varepsilon = 2^{-s}$ will suffice, if we choose $s = n(\ell + O(1))$. By means of Theorem 2.1 we thus get the following result in the sense of a worst case analysis.

Theorem 19.2. *Approximate determination of the zeros (or of the reciprocals of the large zeros, respectively) of any n-th degree polynomial within an error bound of $2^{-\ell}$ is possible in time $O(n^3 \log n + n^3 \ell)$. In case of multitape Turing machines an extra factor $\log(n\ell) \log\log(n\ell)$ comes in.*

Observe that the second term $O(n^3\ell)$ will be a crude overestimate for many polynomials that permit more balanced splittings. Moreover (19.7) can be improved considerably if the zeros are not clustered too much, i. e., if the 'multiplicities' m_κ in (19.5) are small in comparison with n. We give a brief outline of a method for deriving corresponding *a posteriori* bounds, e. g. for $|v_1 - z_1|$.

The numbering of the v's shall be such that the numbers $d_j = |v_j - v_1|$ are in non-decreasing order. By (19.7) we find as an initial bound $r_1 = \gamma_n \sqrt[n]{\varepsilon}$. Now assume that there exists an $m < n$ such that $d_m \le \frac{5}{4} \cdot r_1 < d_{m+1}$. Then we can restrict the Chebyshev polynomial argument to the factors with $j \le m$ in (19.6), while the other factors are replaced by $(d_j - r_1)$. In this way we obtain the smaller (!) bound

$$r_2 = \gamma_m \cdot \left(\frac{\varepsilon}{\prod_{j=m+1}^{n} (d_j - r_1)} \right)^{1/m}\,. \tag{19.8}$$

If now d_m is greater than $\frac{5}{4} \cdot r_2$, then it will be possible to use some $m' < m$ and r_2 instead of r_1 for computing another improved bound $r_3 < r_2$ in the very same way, etc. We should, however, not spend too much computational effort on finding 'sharp' a posteriori bounds (whatever that means), since it may be cheaper to find a more accurate factorization. Increasing its accuracy by $2^{-O(n)}$ will require additional time $O(n^3)$ at most, and then we can afford to use a simplified version of the idea behind (19.8). We choose the minimal $m \ge 1$ such that (approximately)

$$\left(\tfrac{1}{4}\, d_{m+1}\right)^m \cdot \left(\prod_{j=m+1}^{n} d_j \right) \ge \varepsilon \cdot 2^{n-1} \tag{19.9}$$

is satisfied. Then the error bound is

$$|v_1 - z_1| < (1 + \cos(\pi/2n)) \left(\frac{\varepsilon \cdot 2^{n-1}}{\prod_{j=m+1}^{n} d_j} \right)^{1/m}\,.$$

20 Isolating the zeros of integer polynomials

Let us now consider polynomials $P(z) = a_0 z^n + \cdots + a_n$ with integer coefficients $a_j \in \mathbb{Z}[i]$ and *simple* zeros $z_1, \ldots, z_n$. Isolating these zeros means to compute n disjoint disks each containing exactly one of the z's. Akritas [3, 3a] gives a solution for isolating the real zeros of (real) integer polynomials with time bound $O(n^5\ell^3)$, where ℓ denotes the maximal coefficient length. By means of fast integer multiplication this bound can immediately be reduced to $O(n^4\ell^2)$. In the sequel we describe a solution by means of the splitting circle method with the asymptotically better time bound $O(n^3(\ell + \log n))$.

It will suffice to specify a suitable relative accuracy ε and a radius r such that any approximate factorization F of P with

$$|P - F| < \varepsilon \cdot |P| \tag{20.1}$$

will provide a solution of the isolation problem by the n disks obtained from the zeros $v_1, \ldots, v_n$ of F and from r according to

$$D_j = \begin{cases} \{z \,|\, |z - v_j| < r\} & \text{for } |v_j| \le 1\,, \\ \{z \,|\, |1/z - 1/v_j| < r\} & \text{for } |v_j| > 1\,. \end{cases} \tag{20.2}$$

Let z_1 denote an arbitrary zero of P with $|z_1| \le 1$. The discriminant $\operatorname{dis}(P) \ne 0$ is an integer, whence $|\operatorname{dis}(P)| \ge 1$. Thus an inequality given at the end of [19] (and slightly improved by using $|\cdot|_2$ in Theorem 4.2 instead of $|\cdot|$ as in [19]) yields

$$|P'(z_1)| \;\ge\; \tau = \Big((n-1)^{(n-1)/2} \cdot |P|_2^{n-2}\Big)^{-1}. \tag{20.3}$$

For $|z - z_1| \le 8r$ and small r we have $|z| \le 1 + 8r \le 1 + 1/n^2$, and then $|P''(z)| \le n^2\,|P|$ can be shown. For $|z - z_1| = tr$ with $0 \le t \le 8$ Taylor's formula now yields the lower bound

$$|P(z)| \;\ge\; tr\,|P'(z_1)| - \tfrac{1}{2}t^2r^2 \cdot n^2\,|P| \;\ge\; tr \cdot (\tau \,-\, \tfrac{1}{2}tr \cdot n^2\,|P|)\,. \tag{20.4}$$

Hereby it becomes clear that we can, for instance, choose

$$r = \frac{\tau}{4n^2\,|P|}\,, \qquad \varepsilon = \frac{\tau^2}{8n^2\,|P|^2}\,. \tag{20.5}$$

Then (20.1) and (20.4) imply that there exists exactly one zero of F in $|z - z_1| < 4r$, say v_1, and more precisely $|v_1 - z_1| < \frac{2}{3}r$. Therefore $z_1 \in D_1$, even in case of $|v_1| > 1$, where the second line of (20.2) applies. Moreover (20.4) shows $|z_1 - z_j| \ge 8r$ for all other zeros. That guarantees that the disks (20.2) based on (20.5) are pairwise disjoint. For the large zeros $|z_j| > 1$ the same analysis holds for the reverse polynomials P^*, F^*.

A simple measure for the size of P is its maximal coefficient length defined as the minimal integer ℓ such that $|\operatorname{Re} a_j| < 2^\ell$ and $|\operatorname{Im} a_j| < 2^\ell$ for all j. We can use the bounds $|P|_2^2 \le (2n+2)\,2^{2\ell}$ and $|P| \le (2n+2)\,2^\ell$. Then the ε in (20.5) is to be replaced by some

$$2^{-s} \le \Big(8n^2(n-1)^{n-1} \cdot (2n+2)^n\,2^{(2n-2)\ell}\Big)^{-1},$$

hence $s = \lceil (2n+1)(\ell + 1 + \log(n+1)) \rceil$ will suffice. Theorem 2.1 now gives the final result.

Theorem 20.1. *Isolating the zeros of any squarefree n-th degree integer polynomial with maximal coefficient length ℓ is possible in time*

$$O(n^3 \log n + n^3 \ell) \cdot \log(n\ell) \cdot \log\log(n\ell)\,.$$

Again the second term $n^3\ell$ may be an overestimate in cases that permit more balanced splittings. A more detailed study of the possible patterns in the distribution of zeros of such integer polynomials could perhaps reveal that 'many' of the splittings will necessarily be balanced such that in this way an even better time bound could be derived. We leave this as an open problem to the reader.

21 Eigenvalues of matrices

A full treatment of the computational problems concerning the structural invariants of general complex matrices is beyond the scope of this report, but clearly the methods of § 19 can be applied for computing the *eigenvalues* of any complex $n \times n$ matrix A provided the coefficients of the *characteristic polynomial* $f(z) = \det(zI - A)$ are available with sufficient precision. So the main subject of this last paragraph is the approximate determination of f (i. e., of its coefficients).

Recently W. Keller (Zürich [15]) has discovered that in terms of *algebraic* complexity the transition from a matrix A to its characteristic polynomial is of about the same complexity as matrix multiplication (up to logarithmic factors). In the sequel we will mimic his idea *numerically*. Let ω denote the exponent of matrix multiplication over fields of characteristic zero (cf. [26], Theorem 2.8 in particular). Meanwhile the first non-trivial bound $\omega < 2.81$ has been reduced to $\omega < 2.5$ (see [31, 26, 5]), while there is only the trivial lower bound $\omega \geq 2$. Despite the rumor about numerical instability of fast matrix multiplication algorithms it can be shown that, for any $\beta > \omega$, multiplication of complex $n \times n$ matrices in s-bit precision is possible in time $O(n^\beta s)$, admittedly with a 'constant' depending on β. Without knowing the true value of ω, we could even design one global algorithm with time bound $o(n^\beta s)$ for any $\beta > \omega$.

By the methods of [25] *unitary* transformation of a given matrix A into *upper Hessenberg form* B (i. e. $b_{j+m,j} = 0$ for $m \geq 2$) can therefore be accomplished in time $O(n^\beta s)$, if s-bit precision is sufficient. B has the same eigenvalues $\lambda_1, \ldots, \lambda_n$ as A, but the result of the numerical transformation will be some $\hat{B}$ with eigenvalues $\hat{\lambda}_1, \ldots, \hat{\lambda}_n$. In order to estimate these perturbations some norm for vectors and matrices is needed, e. g. $|y| = \sum |y_j|$ for $y^t = (y_1, \ldots, y_n)$ and $|A| = \max_\nu \sum_\mu |a_{\mu,\nu}|$ for matrices. Then an analogue of Theorem 19.1 can be derived, namely:

$$|B - \hat{B}| < \varepsilon \quad \text{implies} \quad |\lambda_j - \hat{\lambda}_j| < \text{const} \cdot |B| \cdot \sqrt[n]{\varepsilon} \tag{21.1}$$

(by suitably numbering the λ's).

After multiplication with a suitable factor 2^{-d} we may assume that $|B| \leq \frac{1}{2n}$ (actually with $\hat{B}$ instead of B, but let us skip these technicalities now). Furthermore, let $b_{j+1,j} \neq 0$ for $j = 1, \ldots, n-1$; otherwise the problem splits up into

several smaller eigenvalue problems. Then the next step is *scaling* by a similarity transformation with a diagonal matrix D such that $M = D^{-1}BD$ has only ones in its subdiagonal, i. e., $M = S + R$, where R denotes the upper triangle of M, and the subdiagonal S represents a shift operator mapping the j-th unit vector e_j upon e_{j+1}, and with $Se_n = 0$. For $1 \le j \le j+m \le n$ the new entries are given by

$$r_{j,j+m} = b_{j,j+m} \cdot \prod_{\nu=0}^{m-1} b_{j+\nu+1,j+\nu}\,. \tag{21.2}$$

They can be computed in a stable way by $O(n^2)$ multiplications of complex numbers bounded by one. In s-bit precision this requires not more than time $O(n^2 s) \le O(n^\beta s)$. Moreover (21.2) implies $|R| \le \frac{1}{2n}$. The coefficients of $f(z) = \varphi_0 + \varphi_1 z + \cdots + \varphi_{n-1}z^{n-1} + z^n$ are determined in the following way. Since f is also the characteristic polynomial of M, we have $f(M) = 0$, $\varphi_0 I + \varphi_1 M + \cdots + \varphi_{n-1}M^{n-1} = -M^n$. This equation applied to the first unit vector e_1 yields

$$\sum_{j=1}^{n} \varphi_{j-1} x_j = -x_{n+1}\,, \qquad X\varphi = -x_{n+1}\,, \tag{21.3}$$

where $x_j = M^{j-1}e_1$, and X denotes the $n \times n$ matrix built up from the columns $x_1, \ldots, x_n$. In view of $e_j = S^{j-1}e_1$ we can estimate

$$|x_j - e_j| \;=\; \big|\big((S+R)^{j-1} - S^{j-1}\big)\, e_1\big| \;\le\; \big(1 + \tfrac{1}{2n}\big)^n - 1 \;<\; \sqrt{e} - 1 < 0.65\,,$$

which implies $|X - I| < 0.65$ and the existence of X^{-1} with $|X^{-1}| < 1/(1-0.65) < 3$. Thus the coefficient vector can be obtained from (21.3) as $\varphi = X^{-1}(-x_{n+1})$, numerically in time $O(n^\beta s)$.

Keller's main idea concerns the efficient computation of the iterates $x_j = M^{j-1}e_j$ for $j \le n+1$. By $\lceil \log n \rceil$ matrix multiplications the powers $M^2, M^4, \ldots$ can be obtained; then the crucial step is

$$M^{2^m}(x_1, \ldots, x_{2^m}) = (x_{2^m+1}, \ldots, x_{2^{m+1}})$$

for $0 \le m \le \lceil \log(n+1) \rceil - 1$ recursively, which requires another $\lceil \log(n+1) \rceil$ matrix multiplications.

The whole process can be summarized by the statement that (for any $\beta > \omega$) time $O(n^\beta s)$ suffices for approximately computing the characteristic polynomial of a given $n \times n$ matrix A such that the perturbations of the eigenvalues are bounded by $\mathrm{const} \cdot |A| \cdot 2^{-s/n}$ in the sense of worst case analysis covered by (21.1). Thus application of Theorem 19.2 yields the final result.

Theorem 21.1. *Approximate determination of the eigenvalues of any complex $n \times n$ matrix A of norm $|A| \le 1$ within an error bound of $2^{-\ell}$ is possible in time*

$$O(n^{\beta+1}\ell) \;+\; O(n^3 \log n + n^3 \ell) \tag{21.4}$$

for any $\beta > \omega$, e. g. $\beta = 2.5$.

For practical applications only Strassen's exponent $\beta = \log_2 7$ seems to be feasible, and clearly one should not start with the pessimistic bound (21.1). It is better to begin with a moderate precision of s_0 bits. Then the bound is $O(n^{2.81}s_0 + n^3 \log n)$. If the a posteriori analysis (see § 19) based on the actual distribution of the eigenvalues does not yet guarantee the desired accuracy, then the whole computation is repeated in $2s_0$-bit precision, in $4s_0$-bit precision, etc., until satisfactory accuracy is obtained. QR fans, by the way, should be aware of the fact that each QR iteration step is at least as expensive as one matrix multiplication.

Theoretically the worst case bound (21.4) reveals that, according to the present state of our knowledge, the most expensive part in computing the eigenvalues of a general matrix is the computation of its characteristic polynomial. Substantial improvements of this complexity bound seem to be possible only by further progress in the art of matrix multiplication.

References

[1] N.H. Abel, Beweis der Unmöglichkeit algebraische Gleichungen von höheren Graden als dem vierten allgemein aufzulösen.
J. reine u. ang. Math. 1, 65–84 (1826).

[2] A.V. Aho, K. Steiglitz and J.D. Ullman, Evaluating polynomials at fixed sets of points. SIAM J. Comp. 4, 533–539 (1975).

[3] A.G. Akritas, An implementation of Vincent's theorem.
Numer. Math. 36, 53–62 (1980).

[3a] A.G. Akritas, The fastest exact algorithms for the isolation of the real roots of a polynomial equation. Computing 24, 299–313 (1980).

[4] H. Alt, Square rooting is as difficult as multiplication.
Computing 21, 221–232 (1979).

[5] D. Coppersmith and S. Winograd, On the asymptotic complexity of matrix multiplication. SIAM J. Comp. 11, 472–492 (1982).

[6] B. Dejon and P. Henrici (editors),
Constructive aspects of the fundamental theorem of algebra.
Proc. Symp. Zürich-Rüschlikon 1967, J. Wiley, London 1969.

[7] B. Dejon and K. Nickel, A never failing, fast convergent rootfinding algorithm.
In [6], pp. 1–35.

[8] L.M. Delves and J.N. Lyness, A numerical method for locating the zeros of an analytic function. Math. Comp. 21, 543–560 (1967).

[9] H. Friedman and K.-I. Ko, Computational complexity of real functions.
Theor. Comp. Sci. 20, 323–352 (1982).

[10] I. Gargantini and P. Henrici, Circular arithmetic and the determination of polynomial zeros. Numer. Math. 18, 305–320 (1972).

[11] C.F. Gauss, Demonstratio nova theorematis omnem functionem algebraicam rationalem integram unius variabilis in factores reales primi vel secundi gradus resolvi posse. Gesammelte Werke III, 1–30 (1799).

[12] P. Henrici, Applied and computational complex analysis, vol. 1.
J. Wiley, New York 1974.

[13] M.A. Jenkins and J.F. Traub, An algorithm for an automatic general polynomial solver. In [6], pp. 151–180.

[14] M.A. Jenkins and J.F. Traub, A three-stage variable-shift iteration for polynomial zeros and its relation to generalized Rayleigh iteration.
Numer. Math. 14, 252–263 (1970).

[15] W. Keller, Private Communication.

[16] D.E. Knuth, The art of computer programming, vol. 2.
Addison-Wesley, Reading, Mass. 1969, 2nd edition 1981.

[17] D.H. Lehmer, A machine method for solving polynomial equations.
JACM 8, 151–162 (1961).

[18] D.H. Lehmer, Search procedures for polynomial equation solving.
In [6], pp. 193–208.

[19] K. Mahler, An inequality for the discriminant of a polynomial.
Michigan Math. J. 11, 257–262 (1964).

[20] M. Mignotte, An inequality about factors of polynomials.
Math. Comp. 28, 1153–1157 (1974).

[21] A. Ostrowski, Recherches sur la méthode de Graeffe et les zéros de polynômes et les séries de Laurent. Acta Math. 72, 99–257 (1940).

[22] A. Ostrowski, Solution of equations and systems of equations, 2nd edition.
New York 1966.

[23] A. Schönhage, Storage modification machines.
SIAM J. Comp. 9, 490–508 (1980).

[24] A. Schönhage, Asymptotically fast algorithms for the numerical multiplication and division of polynomials with complex coefficients.
Proc. EUROCAM '82, Marseille 1982.

[25] A. Schönhage, Unitäre Transformationen großer Matrizen.
Numer. Math. 20, 409–417 (1973).

[26] A. Schönhage, Partial and total matrix multiplication.
SIAM J. Comp. 10, 434–455 (1981).

[27] A. Schönhage and V. Strassen, Schnelle Multiplikation großer Zahlen.
Computing 7, 281–292 (1971).

[28] J. Schröder, Über das Newton'sche Verfahren.
Arch. Rat. Mech. Anal. 1, 154–180 (1957).

[29] J. Schröder, Factorization of polynomials by generalized Newton procedures.
In [6], pp. 295–320.

[30] E. Specker, The fundamental theorem of algebra in recursive analysis.
In [6], pp. 321–329.

[31] V. Strassen, Gaussian elimination is not optimal.
Numer. Math. 13, 354–356 (1969).

[32] A. Van der Sluis, Upper bounds for roots of polynomials.
Numer. Math. 15, 250–262 (1970).

[33] H. Weyl, Randbemerkungen zu Hauptproblemen der Mathematik, II. Fundamentalsatz der Algebra und Grundlagen der Mathematik. Math. Z. 20, 131–150 (1924).

[34] H.S. Wilf, A global bisection algorithm for computing the zeros of polynomials in the complex plane. JACM 25, 415–420 (1978).

[35] S. Smale, The fundamental theorem of algebra and complexity theory. Bull. AMS (New Ser.) 4, 1–36 (1981).

Added in proof: After completion of this report we have received notice of the paper [35]. Despite of the striking coincidence of the titles there is no substantial overlap with the material of our report. As far as can be seen, the algorithm described in [35] cannot handle multiple zeros; a certain subset of polynomials is excluded. There is no discussion of round-off errors. Instead of counting bit operations, the complexity of root-finding is studied in terms of the degree only.

Appendix 2004. We add a few selected links for 'transitive' access to more recent literature on root finding and related topics:

[Ke] W. Keller-Gehrig, Fast algorithms for the characteristic polynomial. Theor. Comp. Sci. 36, 309–317 (1985). — *supplementing reference [15]*.

[Ki1] P. Kirrinnis, Partial fraction decomposition in $\mathbb{C}(z)$ and simultaneous Newton iteration for factorization in $\mathbb{C}[z]$. J. Complexity 14, 378–444 (1998).

[Ki2] P. Kirrinnis, Fast computation of invariant subspaces and bit complexity in numerical linear algebra. Habilitationsschrift Universität Bonn, November 1999, about 200 pages. — *hopefully getting published soon*.

[Ne] C.A. Neff, Specified precision root isolation is in NC. J. Comp. Syst. Sci. 48, 429–463 (1994).

[NR] C.A. Neff and J.H. Reif, An efficient algorithm for the complex roots problem. J. Complexity 12, 81–115 (1996).

[Pa1] V.Y. Pan, Optimal and nearly optimal algorithms for approximating polynomial zeros. Computers & Math. 31, 97–138 (1996).

[Pa2] V.Y. Pan, Univariate polynomials: Nearly optimal algorithms for numerical factorization and root-finding. J. Symb. Comp. 33, 701–733 (2002).

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

[Sch] R. Schätzle, On the perturbation of the zeros of complex polynomials. IMA Journal of Numerical Analysis 20, 185–202 (2000). — *improving our Theorem 19.1, by replacing the bound 4 in (19.7) with the optimal value 2*.

[Sc1] A. Schönhage, Quasi-GCD computations. J. Complexity 1, 118–137 (1985).

[Sc2] A. Schönhage, Equation solving in terms of computational complexity. Proc. Int. Congress of Mathematicians, Berkeley 1986, vol. 1, 131–153.