Computation and Validation of Root Clusters for Univariate Polynomials

Petru Pau, Josef Schicho
Research Institute for Symbolic Computation RISC - LINZ, Austria
[ppau,jschicho]@risc.uni-linz.ac.at

ABSTRACT

The problem of computing and validating root clusters for univariate polynomials has been considered by several authors. In this paper, we solve this problem by giving explicit formulae for the radii of disks containing at least k roots. The method is used in an algorithm for computing polynomial roots in the frame of "exact real computation".

1. INTRODUCTION

It is well-known that numerical root computation of univariate polynomials is ill-conditioned when the given polynomial has multiple zeroes, or when the given polynomial is close to a polynomial with multiple zeroes. The situation cannot be improved by making the input squarefree, because squarefree decomposition is in itself an ill-posed problem: an infinitesimally small change in the input causes a jump in the output. As it is not possible to compute the roots and their multiplicities individually (see [1]), several authors have suggested algorithms to compute clusters instead (see [12, 13, 8, 15]). The standard approach is to use pseudo-squarefree decompositions, which are a numerically stable variant of squarefree decomposition. The given polynomial is written as a product of powers of squarefree polynomials, plus a small error term. Each root of a squarefree pseudo-factor with exponent r corresponds to a cluster of r roots.

The location of an r-fold root, and likewise the location of an r-cluster of roots, depends Hölder continuously on the coefficients: if the coefficients are known up to an error ϵ, we may compute the location of the cluster up to an error of order $\epsilon^{1/r}$. This error can be bound a priori (estimating the errors that could be made by the algorithm) or a posteriori (using the computed result). The main contribution of this paper consists in two a posteriori bounds: Theorem 1 provides a formula in the coefficients, and the bound in Theorem 2 involves some combinatorial computation but is sharper. The bounds may be considered as generalizations of a result in [16] providing an infallible error bound for the error of a single root.

Another method for a posteriori validation of clusters is contained in [8]. This bound is usually tighter than ours (see the statistics in section 2). On the other hand, our bound is easier to compute. It is also infallible, in contrast to [8] which is just "highly confident" (as one has to estimate numerically the convergency radius of an infinite Taylor series).

Section 3 contains a numerically stable algorithm for computing clusters, similar to [13]. The main contribution in this part is a proof that the algorithm computes arbitrarily tight clusters as the accuracy approaches zero. This fact can be used – together with the bound in section 2 – to compute the roots of a polynomial in the sense of exact real number arithmetic (see [3, 11]); this is explained in section 4.

This research has been supported by the Austrian science fund (FWF) in the frame of the special research area "numerical and symbolic scientific computing" (SFB013).

The authors wishes to thank to Christian Weixlbaumer for his suggestions regarding the combinatorial properties of some recursions that occur in subsection 2.2, and Werner Krandick for useful remarks.

2. VALIDATION

Throughout this paper, we work with a polynomial p in $\mathbb{R}[x]$, of coefficients $p_0, \ldots, p_n$:

$$p = p_n x^n + p_{n-1} x^{n-1} + \cdots + p_0;$$

z will be a complex number, and the roots of p will be denoted by $z_1, \ldots, z_n$.

In [16], A. Strzebonsky gives an estimate for the error of an approximate of a root:

Lemma 1. *Let z be a complex number, and assume that*

$$|z - z_1| \leq |z - z_2| \leq \cdots \leq |z - z_n|.$$

Then

$$|z - z_1| \leq \left| \frac{n! p(z)}{(n-k)! p^{(k)}(z)} \right|^{1/k},$$

for all k such that $p^{(k)}(z) \neq 0$.

This lemma allows us to compute a bound for the distance from z to the nearest exact root of p. Practically, it gives the radius ε_z of a disk centered at z that contains at least one exact root of p.

If the polynomial is known to be squarefree, after computing some approximate values for its roots, we can use Strzebonsky's formula to get estimates of the accuracies. The case of overlapping disks corresponding to two distinct approximate roots is not acceptable. If this happens, the approximate roots must be computed more accurately, to be closer to the exact roots. If the coefficients of p are real numbers, the algorithm must also consider better approximates of these coefficients.

We cannot use this formula for polynomials about which we do not know whether they are squarefree or not. We can try to find some approximate values of the roots, and then produce a *matching*, a correspondence between exact roots and approximations; but if we do so, it may happen that two approximate roots, corresponding to two distinct exact roots, are detected to be closer to one of the exact roots, because their validity disks, with radii computed using Strzebonsky's formula, are not disjoint; thus they will be considered, incorrectly, as approximations of a double root – see Fig. 1. The second root will then be ignored completely.

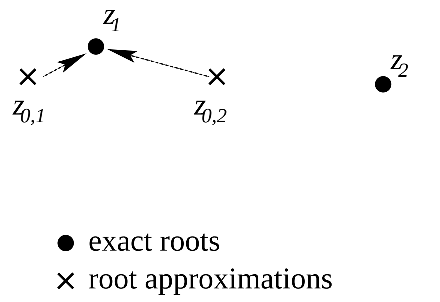

Figure 1: Although the approximate root $z_{0,2}$ corresponds to z_2, it is closer to z_1; z_1 will be (erroneously) considered a double root.

Clearly, if we had a formula for estimating the distance to the second nearest exact root, we would be able to handle these cases more gracefully: instead of two overlapping disks containing at least one root, we would produce one disk containing at least two roots. In general, if we had a formula for estimating the distance to the k'th nearest exact root, we would be able to produce disks labelled by a positive integer, which is greater than or equal to the number of roots contained. The next two subsections do provide such formulae.

It remains the question of how we could come up with reasonable guesses for the midpoints and for the integer labels. This will be the content of section 3.

2.1 Estimating the distance to the k'th nearest root

In this subsection we assume that the roots of p are arranged in the increasing order of their absolute values: $|z_1| \leq \cdots \leq |z_n|$. We try first to find an estimate for the absolute value of the k'th root, with $1 \leq k \leq n$. In other words, we want a (small) neighborhood of 0 that contains (at least) k exact roots of p. For the moment, we assume that p_0 is different from 0.

In the following, we look for a positive value M with the property that the assumption "only $k-1$ roots of p have absolute values smaller than M" leads to a contradiction.

We make this assumption, and rewrite the relations between coefficients and roots for the reciprocal polynomial $\bar{p} = x^n p(1/x)$. Denoting by $y_1, \ldots, y_n$ the roots of $\bar{p}$, we get $|y_1|, \ldots, |y_{k-1}| \geq \frac{1}{M}, |y_k|, \ldots, |y_n| < \frac{1}{M}$. We use the following symbol for the sums of products of roots:

$$\begin{aligned}
\underset{s,t}{\overset{k}{\mathrm{S}y}} &:= \sum_{s \leq i_j \leq t} y_{i_1} y_{i_2} \ldots y_{i_k}, && (1)\\
\underset{s,t}{\overset{0}{\mathrm{S}y}} &:= 1 \\
\underset{s,t}{\overset{l}{\mathrm{S}y}} &:= 0 \text{ if } l = 0 \text{ or } l > t - s + 1.
\end{aligned}$$

Clearly,

$$|\underset{k,n}{\overset{l}{\mathrm{S}y}}| < \binom{n-k+1}{l} \frac{1}{M^l} \leq \left(\frac{n}{M}\right)^l.$$

For $i = 0, \ldots, k-1$, let

$$R_i := \sum_{j=0}^{i} \left|\frac{p_j}{p_0}\right| \left(\frac{2n}{M}\right)^{i-j}.$$

We claim that

$$\left|\underset{1,k-1}{\overset{i}{\mathrm{S}y}}\right| < R_i$$

for $i = 0, \ldots, k-1$. Indeed, by using the formula

$$\left|\frac{p_i}{p_0}\right| = |\underset{1,n}{\overset{i}{\mathrm{S}y}}| = \left|\sum_{j=0}^{i} \underset{1,k-1}{\overset{i-j}{\mathrm{S}y}} \underset{k,n}{\overset{j}{\mathrm{S}y}}\right| \qquad (2)$$

and induction on i, we obtain

$$\begin{aligned}
\underset{1,k-1}{\overset{i}{\mathrm{S}y}} &\leq \left|\frac{p_i}{p_0}\right| + \sum_{j=0}^{i-1} |\underset{1,k-1}{\overset{j}{\mathrm{S}y}} \underset{k,n}{\overset{i-j}{\mathrm{S}y}}| \leq \left|\frac{p_i}{p_0}\right| + \sum_{j=0}^{i-1} R_j \left(\frac{n}{M}\right)^{i-j} \\
&= \left|\frac{p_i}{p_0}\right| + \sum_{j=0}^{i-1} \left|\frac{p_j}{p_0}\right| (2^{i-j} - 1) \left(\frac{n}{M}\right)^{i-j} \leq R_i.
\end{aligned}$$

Using formula 2 again, and taking into account that $\underset{1,k-1}{\overset{k}{\mathrm{S}y}} = 0$, we get

$$\begin{aligned}
\left|\frac{p_k}{p_0}\right| &\leq \sum_{i=0}^{k-1} R_i \left(\frac{n}{M}\right)^{k-i} = \sum_{i=0}^{k-1} \left|\frac{p_i}{p_0}\right| (2^{k-i} - 1) \left(\frac{n}{M}\right)^{k-i} \\
&< \sum_{i=0}^{k-1} \left|\frac{p_i}{p_0}\right| \left(\frac{2n}{M}\right)^{k-i}.
\end{aligned}$$

We are now in position to choose a value for M:

$$M := 2n \cdot \max_{0 \leq j < k} \sqrt[k-j]{\frac{k|p_j|}{|p_k|}}, \qquad (3)$$

which would then lead to the contradictory $\left|\frac{p_k}{p_0}\right| < \left|\frac{p_k}{p_0}\right|$.

If we assume that only $k-1$ roots of p have their absolute values less than M, the few steps from above lead us to a contradiction. Thus, we conclude the following Lemma.

Lemma 2. *The polynomial p has at least k roots in the disk centered at the origin, of radius M given by formula 3.*

Observe that the final bound does not contain p_0 in the denominator. This allows us to get rid of the assumption $p_0 \neq 0$. Let $p := p_n x^n + \cdots + p_1 x$, and let M be as defined in 3. For small α, the polynomial $p_\alpha := p + \alpha$ has at least k roots in the disk with radius M around the origin. Because the set of roots of a polynomial depends continuously on the coefficients, the disk also contains at least k roots of p.

Clearly, theorem 2 gives the optimal radius 0 if 0 happens to be a k-fold root. If p is a polynomial with distance ϵ to a polynomial q with a k-fold root, then Lemma 2 produces a radius of order $\epsilon^{1/k}$. Considering the polynomials $p := x^k - \epsilon$, we see that this order is optimal.

In order to obtain a disk of center $z_0 \in \mathbb{C}$ containing at least k roots, we apply formula (3) to $p(x+z_0)$. By Taylor expansion, we get the following theorem:

Theorem 1. *Let $p \in \mathbb{R}[x]$ of degree n, let $k \in \mathbb{N}$ and $z \in \mathbb{C}$ such that $p^{(k)}(z) \neq 0$. Then there are at least k roots of p in the disk centered at z of radius*

$$M := 2n \cdot \max_{0 \leq j < k} \sqrt[k-j]{\frac{k \cdot k! \cdot |p^{(j)}(z)|}{j! \cdot |p^{(k)}(z)|}}. \qquad (4)$$

2.2 A tighter estimation

We use the same notations as in the previous subsection. Again, we try to find a positive value M such that the assumption "only $k-1$ roots of p have absolute values smaller than M" leads to a contradiction.

We have:

$$\begin{aligned}
-\frac{p_1}{p_0} = \underset{1,n}{\overset{1}{\mathrm{Sy}}} \quad &\text{thus} \quad \underset{1,k-1}{\overset{1}{\mathrm{Sy}}} = -\frac{p_1}{p_0} - \underset{k,n}{\overset{1}{\mathrm{Sy}}}; \\
\frac{p_2}{p_0} = \underset{1,n}{\overset{2}{\mathrm{Sy}}} \quad &\text{thus} \quad \underset{1,k-1}{\overset{2}{\mathrm{Sy}}} = \frac{p_2}{p_0} - \underset{1,k-1}{\overset{1}{\mathrm{Sy}}} \cdot \underset{k,n}{\overset{1}{\mathrm{Sy}}} - \underset{k,n}{\overset{2}{\mathrm{Sy}}} \\
&= \frac{p_2}{p_0} + \frac{p_1}{p_0} \underset{k,n}{\overset{1}{\mathrm{Sy}}} + \left(\underset{k,n}{\overset{1}{\mathrm{Sy}}}\right)^2 - \underset{k,n}{\overset{2}{\mathrm{Sy}}} \\
&\ldots
\end{aligned}$$

The following general formula can be proved by induction:

$$\underset{1,k-1}{\overset{l}{\mathrm{Sy}}} = (-1)^k \cdot \left(\frac{p_l}{p_0} + \frac{p_{l-1}}{p_0} D_1 + \cdots + \frac{p_1}{p_0} D_{l-1} + D_l\right),$$

where

$$D_l := \det \begin{pmatrix} \underset{k,n}{\overset{1}{\mathrm{Sy}}} & 1 & 0 & \ldots & 0 \\ \underset{k,n}{\overset{2}{\mathrm{Sy}}} & \underset{k,n}{\overset{1}{\mathrm{Sy}}} & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \underset{k,n}{\overset{l-1}{\mathrm{Sy}}} & \underset{k,n}{\overset{l-2}{\mathrm{Sy}}} & \underset{k,n}{\overset{l-3}{\mathrm{Sy}}} & \ldots & 1 \\ \underset{k,n}{\overset{l}{\mathrm{Sy}}} & \underset{k,n}{\overset{l-1}{\mathrm{Sy}}} & \underset{k,n}{\overset{l-2}{\mathrm{Sy}}} & \ldots & \underset{k,n}{\overset{1}{\mathrm{Sy}}} \end{pmatrix}.$$

Now, we can write:

$$\frac{p_k}{p_0} = (-1)^k \left(\underset{1,k-1}{\overset{k-1}{\mathrm{Sy}}}\ \underset{k,n}{\overset{1}{\mathrm{Sy}}} + \cdots + \underset{1,k-1}{\overset{1}{\mathrm{Sy}}}\ \underset{k,n}{\overset{k-1}{\mathrm{Sy}}} + \underset{k,n}{\overset{k}{\mathrm{Sy}}}\right);$$

also by induction, it can be proved that:

$$\frac{p_k}{p_0} = (-1)^k \left(\frac{p_{k-1}}{p_0} D_1 + \cdots + \frac{p_1}{p_0} D_{k-1} + D_k\right). \qquad (5)$$

On the other hand, we put the condition that $|y_i| < \frac{1}{M}$, for $i = k, ..., n$, therefore

$$\left|\underset{k,n}{\overset{l}{\mathrm{Sy}}}\right| < \binom{n-k+1}{l} \frac{1}{M^l}.$$

For the determinants we have:

$$\begin{aligned}
|D_1| &= |\underset{k,n}{\overset{1}{\mathrm{Sy}}}| \\
&< \frac{n-k+1}{M}; \\
|D_2| &= |(\underset{k,n}{\overset{1}{\mathrm{Sy}}})^2 - \underset{k,n}{\overset{2}{\mathrm{Sy}}}| \leq (\underset{k,n}{\overset{1}{\mathrm{Sy}}})^2 + |\underset{k,n}{\overset{2}{\mathrm{Sy}}}| \\
&< \frac{(n-k+1)^2}{M^2} + \frac{(n-k+1)(n-k)}{2M^2} \\
&= \frac{(n-k+1)(3n-3k+2)}{2M^2}
\end{aligned}$$

and so on.

Now formula (5) becomes:

$$\left|\frac{p_k}{p_0}\right| < \left|\frac{p_{k-1}}{p_0}\right| \frac{n-k+1}{M} + \left|\frac{p_{k-2}}{p_0}\right| \frac{(n-k+1)(3n-3k+2)}{2M} +$$

$$\left|\frac{p_{k-3}}{p_0}\right| \frac{(n-k+1)(13n^2 - (26k-17)n + (13k^2 - 17k + 6))}{6M^3} +$$

$$\left|\frac{p_{k-4}}{p_0}\right| \frac{1}{8M^3} (n-k+1) \left(25n^3 - (75k-49)n^2\right.$$

$$\left.(75k^2 - 98k + 34)n - (25k^3 + 49k^2 - 34k + 8)\right) + \ldots,$$

the polynomials in n in the numerators becoming more and more complicated. We denote these polynomials by $P_{i,k}(n)$, so that we get

$$\left|\frac{p_k}{p_0}\right| < \sum_{i=0}^{k-1} \left|\frac{p_i}{p_0}\right| \cdot \frac{P_{k-i,k}(n)}{M^{k-i}}.$$

The authors are not aware of a closed form for the polynomials $P_{i,k} \in \mathbb{Z}[n]$. An algorithm for computing $P_{i,k}(n)$ is given below:

Algorithm P(i,k):
1. compute

$$dP := \det \begin{pmatrix} R_1 & 1 & 0 & \ldots & 0 \\ R_2 & R_1 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R_i & R_{i-1} & R_{i-2} & \ldots & R_1 \end{pmatrix},$$

as a polynomial in $\mathbb{Z}[R_1, R_2, \ldots, R_i]$;
2. substitute all integer coefficients c in dP by $|c|$;
3. substitute $\binom{n-k}{j} M^{-j}$ for R_j in dP, for $j = 1, \ldots, i$;
4. return $dP \cdot M^i$ as a polynomial in n.

A closer analysis shows that the leading coefficient of $P_{i,k}$ depends only on i; in fact, it is $A000670(i+1)/i!$, where

$A000670$ is the sequence of preferential arrangements (see [7]).

We can choose now a value for M:

$$M := \max_{0 \le l < k} \sqrt[k-l]{k \frac{|p_l|}{|p_k|} P_{k-l,k}(n)}.$$

If we make the assumption that only $k-1$ roots of p have absolute values smaller than M, the previous considerations will lead us to a contradiction of the form $\frac{|p_k|}{|p_0|} < \frac{|p_k|}{|p_0|}$.

When we work with an arbitrary complex number z instead of 0, an upper bound for the radius of a disk centered at z which covers (at least) k roots of p is:

$$M := \max_{0 \le l < k} \sqrt[k-l]{k \frac{k!}{l!} \frac{|p^{(l)}(z)|}{|p^{(k)}(z)|} P_{k-l,k}(n)}. \tag{6}$$

Note that if z approximates an exact root $\bar{z}$ of p of multiplicity k, then M decreases as z converges to $\bar{z}$. Also, if the coefficients of p are only known with accuracy ε, then the bound given before, which is nothing else than an estimate of the accuracy of the root, is $O(\varepsilon^{1/k})$.

We finalize this discussion with the theorem which, in fact, we have just proved:

Theorem 2. *Let $p \in \mathbb{R}[x]$, let $k \in \mathbb{N}$ and $z \in \mathbb{C}$ such that $p^{(k)}(z) \neq 0$. Then there are at least k roots of p in the disk centered at z of radius*

$$\max_{1 \le l \le k} \sqrt[l]{k \frac{k!}{(k-l)!} \frac{|p^{(k-l)}(z)|}{|p^{(k)}(z)|} P_{l,k}(n)}.$$

2.3 Experimental comparison between formulas

Next table contains lists of values obtained by implementing in Maple the two formulas presented in this paper, and the method of Hribernig and Stetter (H&S). The polynomials were randomly chosen. Each polynomial has a cluster of k roots around 0: its last k coefficients are very small. For each polynomial, an approximate root close to 0 has been considered. The number of digits of precision was chosen randomly, ranging between 4 and 16. The "exact distance", in the last column, was computed by extracting the roots with high accuracy (around 300 digits).

It can be seen that, in most of the cases, the formulas (4) and (6) provide values which are one order of magnitude higher than the values computed with the method of Hribernig and Stetter. On the other hand, our formulas are much faster, and also infallible.

There are two possible reasons for the failures reported by the method of Hribernig and Stetter. First, their method involves the computation of a Taylor series for a rational function. It may have happened that the criterion for stopping the generation of its coefficients was not fulfilled after a reasonable number of them had been computed. Secondly, this Taylor series contributed to a polynomial equation, whose smaller real positive solution is the value reported by the algorithm. It may have happened that the equation did not have any real (or positive) solution.

Degree	Multi-plicity	Formula (4)	Formula (6)	H&S method	Exact distance
12	7	3.5 (17 msec)	.88 (0.3 sec)	.24e-1 (0.6 sec)	.59e-2
10	5	.66e-1 (11 msec)	.20e-1 (0.2 sec)	.71e-3 (0.5 sec)	.25e-3
7	2	.11e-6 (2 msec)	.48e-7 (7 msec)	.40e-8 (90 msec)	.40e-8
9	4	.49e-2 (14 msec)	.16e-2 (45 msec)	.72e-4 (5.2 sec)	.34e-4
13	8	.18e3 (0 msec)	41. (0.4 sec)	failure (0.2 sec)	.39
10	5	.94e-1 (12 msec)	.28e-1 (71 msec)	.10e-2 (0.6 sec)	.36e-3
7	2	.16e-4 (5 msec)	.67e-5 (10 msec)	.56e-6 (1.8 sec)	.56e-6
10	5	.63e-1 (19 msec)	.19e-1 (80 msec)	.68e-3 (7.6 sec)	.24e-3
14	9	6.7 (0 msec)	1.9 (0.7 sec)	failure (0 msec)	.19
8	3	.23e-2 (3 msec)	.88e-3 (18 msec)	.49e-4 (0.3 sec)	.28e-4

3. COMPUTATION

Let us turn now to the problem of computing estimations for the roots and their multiplicities. For polynomials with rational coefficients, we may compute squarefree decomposition by GCD computations. We will show that if we replace the GCDs in this algorithm by pseudo-GCDs, we obtain a pseudo-squarefree factorization (pseudo-SFF for short).

There are various concepts of pseudo-GCDs available in the literature [14, 5, 6, 4, 9]. Most of these functions (in fact all except [9]) depend on numerical tests for zero and are therefore *nondeterministic*: there is a certain gray zone where the function value is not unique.

We call a nondeterministic function *continuous* iff the inverse image of any open set is open. If f is a continuous nondeterministic function, then we may compute *one* value $y \in f(x)$ approximately when we know x approximately.

A very simple continuous nondeterministic function is the pseudo-zerotest, defined as

$$\text{pseudo-ZT}_\epsilon(x) = \begin{cases} 1 & \text{if } |x| < \epsilon; \\ 0 & \text{if } x \neq 0. \end{cases}$$

The argument of the nondeterministic function is x, and $\epsilon > 0$ is considered a parameter. To compute one value of pseudo-ZT$_\epsilon(x)$, it suffices to know x up to an accuracy of ϵ.

Let $f_\epsilon : A \to B$ be a family of nondeterministic function parametrized by positive reals. Then we say that f converges to a deterministic function $\overline{f} : A \to B$ iff

$$\lim_{x' \to x, \epsilon \to 0} f_\epsilon(x') = \overline{f}(x)$$

for all $x \in A$. For instance, the family of pseudo-zerotests given above converges to the zero test (which is 1 for $x = 0$ and 0 otherwise).

Lemma 3. *The composition of two continuous nondeterministic functions is again continuous.*

The composition family of two convergent nondeterministic function families is again convergent. Moreover, the limit of the composition is the composition of the limits of the components.

Proof. Obvious. □

We denote by $\mathcal{M}_n$ the space of all monic polynomials of degree n with real coefficients, with the natural topology. Furthermore, we define $\mathcal{M} := \bigcup_{i=0}^{\infty} \mathcal{M}_n$ (the space of all monic polynomials), setting the distance between elements from different subsets to ∞. The GCD is now a function from $\mathcal{M} \times \mathcal{M}$ to $\mathcal{M}$.

Lemma 4. *There is a family of continuous nondeterministic functions from $\mathcal{M} \times \mathcal{M}$ to $\mathcal{M}$ that converges strongly to the GCD.*

Proof. We follow the construction [5]. Suppose that the degrees of p, q are n, m, respectively. First, we estimate the rank of the Sylvester matrix by pseudo-zerotesting its singular values. Suppose this rank guess is $m+n-k$. Second, we compute polynomials u, v, r of degree $m-k-1$, $n-k-1$, k, with r monic, such that $||pu + qv - r||_2$ is minimal. We return the monic polynomial r.

By properties of the pseudo-zerotest, the rank guess may be too low but it cannot be higher than the actual rank. Therefore, the second step is a least square problem with a matrix of maximal rank, which is continuous. As the pseudo-zerotest and the computation of the singular values are also continuous, we see that the whole construction is a composition of continuous nondeterministic functions, and therefore itself continuous.

The family of pseudo-zerotests is convergent, therefore the whole construction is a convergent family by lemma 3. To see what the limit is, we have to replace all pseudo-zerotests by zerotests. Then the rank guess is correct, and the least squares problem has exactly one solution which makes the norm zero; the component r is then the exact GCD. □

Let us call the above construction *pseudo-GCD*.

For $n = 0, 1, 2, \ldots$, the function SFF^n (squarefree factorization) is a function from $\mathcal{M}_n$ to $\mathcal{M}^n$, mapping $t_1 t_2^2 \cdots t_n^n$ to the tuple $(t_1, \ldots, t_n)$. Here is a recursive algorithm for computing a family of nondeterministic functions converging to SFF, also called *pseudo-SFF*.

Algorithm pseudo-GCD(n, ϵ, p):
 if $n \leq 1$ return the n-tuple $(p, 1, \ldots, 1)$ and exit;
 $p' :=$ derivative of p;
 $q :=$ pseudo-GCD$(\epsilon, p, \frac{1}{n} p')$;
 $m := \deg(q)$;
 $r :=$ pseudo-SFF(m, ϵ, q);
 for i from 1 to m
  $p :=$ quotient(p, r_i^{i+1});
 return (p, r) (filled up with 1-components at the end)

Theorem 3. *The above algorithm is correct.*

Proof. It is easy to see that the degree of the pseudo-GCD is less than or equal to the degree of each of the arguments. Since taking derivative reduces the degree by one, the recursion is finite, and we may proceed by induction on n. For $n = 0$, the assertion is trivial.

Pseudo-GCD is a continuous family converging to GCD. By induction hypothesis, pseudo-SFF is a continuous family converging to SFF. Computing derivative, degree (of monic polynomials), power, and quotients are all continuous. By lemma 3, the whole construction is continuous and converges to the function obtained by replacing pseudo-GCD by GCD and pseudo-SFF by SFF (using the induction hypothesis again). But this is obviously the SFF. □

Remark 1. *With similar techniques, one could also prove a similar result for the stabilized square-free decomposition algorithm in [13].*

Once we have proved the correctness by induction, we may as well remove n from the input parameters, as it can be inferred from the input polynomial.

Having a continuous algorithm converging to squarefree factorization, we may compute clusters by computing the roots of the pseudo-factors, in the following way. The algorithm returns a list of triples (z, r, i), $z \in \mathbb{C}$, $r \in \mathbb{R}_+$, $i \in \mathbb{Z}_+$, representing a disk with midpoint z with radius r containing i roots.

Algorithm clusters(ϵ, p):
 CLUSTERS := empty list of disks;
 $(t_1, \ldots, t_n) :=$ pseudo-SFF(ϵ, p);
 for i from 1 to n do
  for each root z of t_i do
   if pseudo-ZT$(p^{(i)}(z))$ then
    exit with error;
   $r :=$ an upper bound for the i-th next root,
    computed by theorem 2
   if the disk $\mathcal{C}(z, r)$ overlaps with disks
    in CLUSTERS then exit with error;
   add (z, r, i) to CLUSTERS;
 return CLUSTERS

The algorithm is correct: by theorem 2, the number of roots inside the disk $\mathcal{C}(z, r)$ contains at least i roots; the disks are disjoint, and by counting it is clear that this minimum is always obtained. The algorithm may return an error at two steps, corresponding to a situation where the numerical insecurity is critical for the clustering process. In this case, the easiest way to obtain a result is to re-run the algorithm with a different parameter ϵ.

The clustering algorithm above forms again a family of continuous non-deterministic functions. It is easy to find its limit (up to the order of the roots): when we replace all pseudo-operations by their exact counterparts, then the radii of all disks have the value zero. Therefore, the limit function is the computation of all roots together with their multiplicities.

4. AN IMPLEMENTATION FOR EXACT REALS

In exact real computation, a real number is represented by a program producing arbitrarily accurate approximations. An approximation is given by a pair of floating point numbers, the value and the accuracy. The distance between the approximated real and the value must not be greater than the accuracy.

The set of all representable real numbers is the set of all *computable reals*. It is well-known that the complex roots of a monic polynomial with computable real coefficients are again computable, or in other words, that the fundamental theorem of algebra is constructive (see [2]). In this section, we present an efficient algorithm for computing the roots of a monic polynomial with exact real coefficients.

It is necessary to mention two features of our implementation of exact reals [3]. First, for every real number, we keep the best approximation computed so far. Second, whenever a real number is constructed, we compute an initial approximation (possibly with quite low accuracy). Both features turned out to be useful; see [3] for details.

Complex numbers are represented analogously to real numbers, by a best known approximation and a program that computes arbitrarily accurate approximations. (We could have also represented a complex number as a pair of reals, but this would have been less compact.)

Here is the input-output specification for root computation in the frame of exact real computation.

Input: n real numbers, denoting the coefficients of a monic polynomial p of degree n;

Output: n complex numbers, denoting the roots of p. Multiple roots occur multiply in this list.

As we will compute the approximations for the complex roots by clusters, we will always have the situation that the approximation disks – centered at the value with the accuracy as the radius – for two roots in the list are either identical or disjoint. The refinements of the individual roots are therefore not independent: any refinement will affect all roots simultaneously.

The j-th exact complex number in the output list consists then of a program that computes arbitrarily fine approximations to the j-th root. As input information it takes the known approximations for all roots. If the list of known approximations is $(z_1, \ldots, z_n)$, we assume that $z_1 = \cdots = z_{i_1}$, $z_{i_1+1} = z_{i_1+2} = \cdots = z_{i_2}$, $\ldots$, $z_{i_{r-1}+1} = z_{i_{r-1}+2} = z_{i_r}$, $i_r = n$, and that $z_{i_1}, z_{i_2}, \ldots, z_{i_r}$ are distinct (and therefore disjoint approximation disks).

1. The first and largest step is to compute clusters for p with radii as small as demanded. This can be done using the algorithm "clusters".
2. For each $j = 1, \ldots, n$ compute the unique l such that $i_{l-1} < j \leq i_l$ (setting $i_0 = 0$).
3. Compute the clusters that intersect the approximation disk of z_{i_l}. Suppose they are (w_1, r_1, k_1), $\ldots$, (w_m, r_m, k_m). We necessarily must have $k_1 + \cdots + k_m = i_l - i_{l-1}$.
4. Compute s such that $k_1 + \cdots + k_{s-1} < j - i_{l-1} \leq k_1 + \cdots + k_s$.
5. Return the value w_s with the accuracy r_s.

The steps 2-5 ensure that the approximations after a single refinement are again either disjoint or identical. Moreover, the number of identical approximations corresponds to the number of roots in the corresponding cluster.

As initial approximation for all roots of $x^n + p_{n-1}x^{n-1} + \cdots + p_0$, we take the value 0 and the accuracy

$$B = 2 \max_{0 \leq i < n-1} |p_i|^{1/(n-i)}.$$

It is well-known that B is a bound for the absolute value of any root (see [10], theorem 3).

5. REFERENCES

[1] Aberth, O. *Precise numerical methods using C++.* Academic Press Inc., San Diego, CA, 1998.

[2] Bishop, E. *Foundations of constructive analysis.* McGraw-Hill, 1967.

[3] Bodnar, G., Pau, P., and Schicho, J. Exact real computation in computer algebra. Tech. Rep. 00-33, RISC-Linz, Univ. Linz, A-4040 Linz, 2000.

[4] Corless, R., Chin, P., and Corliss, G. Optimization strategies for the approximate GCD problem. In *Proc. ISSAC'98* (1998), ACM Press, pp. 228–235.

[5] Corless, R., Gianni, P., Trager, B., and Watt, S. The singular value decomposition for polynomial systems. In *Proc. ISSAC'95* (1995), ACM Press, pp. 195–207.

[6] Emiris, I. Z., Galligo, A., and Lombardi, H. Certified approximate univariate GCDs. *J. Pure Appl. Alg. 117/118* (1997), 229–251.

[7] Gross, O. A. Preferential arrangements. *Amer. Math. Monthly 69* (1962), 4–8.

[8] Hribernig, V., and Stetter, H.-J. Detection and validation of clusters of polynomial zeroes. *J. Symb. Comp. 24* (1997), 667–682.

[9] Kamarkar, N. K., and Lakshman, Y. N. On approximate GCDs of univariate polynomials. *J. Symb. Comp. 26* (1998), 653–666.

[10] Krandick, W. Isolierung reeller Nullstellen von Polynomen. In *Wissenschaftliches Rechnen*, J. Herzberger, Ed. Akademie Verlag, Berlin, 1995, pp. 105–154.

[11] Ménissier-Morain, V. Arbitrary precision real arithmetic: design and algorithms. *J. Symb. Comp.* (2001). to appear.

[12] Neumaier, A. An existence test for root clusters and multiple roots. *ZAMM 68* (1988), 257–259.

[13] Noda, M. T., and Sasaki, T. Approximate square-free decomposition and rootfinding for ill-conditioned algebraic equations. *J. Inf. Proc. 12* (1989), 159–168.

[14] Schönhage, A. Quasi-gcd computations. *J. Complexity 1* (1985), 118–137.

[15] Stetter, H.-J. Condition analysis of overdetermined algebraic problems. In *Proc. CASC 2000* (2000), pp. 345–365.

[16] Strzeboński, A. Computing in thr field of complex algebraic numbers. *J. Symb. Comp. 24* (1997), 647–656.