Joachim von zur Gathen & Igor E. Shparlinski (2008). Approximate polynomial gcd : small degree and small height perturbations. In *Proceedings of LATIN 2008*, Búzios, Rio de Janeiro, Brazil, Eduardo Sany Laber, Claudson Bornstein, Loana Tito Nogueira & Luerbio Faria, editors, volume 4957 of *Lecture Notes in Computer Science*, 276–283. Springer-Verlag, Berlin, Heidelberg. ISSN 0302-9743. URL http://dx.doi.org/10.1007/978-3-540-78773-0_24.

Approximate polynomial gcd: small degree and small height perturbations

Joachim von zur Gathen[1] and Igor E. Shparlinski[2]

[1] B-IT, Universität Bonn
53113 Bonn, Germany
gathen@bit.uni-bonn.de

[2] Department of Computing, Macquarie University
NSW 2109, Australia
igor@ics.mq.edu.au

Abstract. We consider the following computational problem: we are given two coprime univariate polynomials f_0 and f_1 over a ring $\mathcal{R}$ and want to find whether after a small perturbation we can achieve a large gcd. We solve this problem in polynomial time for two notions of “large” (and “small”): large degree (when $\mathcal{R} = \mathbb{F}$ is an arbitrary field, in the generic case when f_0 and f_1 have a so-called normal degree sequence), and large height (when $\mathcal{R} = \mathbb{Z}$).

Key words: Euclidean algorithm, gcd, approximate computation

1 Introduction

Symbolic (exact) computations of the gcd of two univariate polynomials form a well-developed topic of computer algebra. These methods are not directly applicable when the coefficients are “inexact” real numbers, maybe coming from physical measurements. The appropriate model here is to ask for a “large” gcd, allowing “small” additive perturbations of the inputs. Numerical analysis provides several ways of formalizing this, and “approximate gcd” computations are an emerging topic of computer algebra with a growing literature. We only point to Bini & Boito (2007) and its references.

The present paper considers two “exact” notions of approximate gcds. Namely, let $f_0, f_1 \in \mathbb{F}[x]$ be two univariate polynomials over a field $\mathbb{F}$, both of degree at most n, and d and e integers. We are interested in perturbations $u_0, u_1 \in \mathbb{F}[x]$ of degree at most e such that $\deg \gcd(f_0 + u_0, f_1 + u_1) \geq d$. We show that if $e < \min\{2d - n, n - d\}$, then the problem has at most one solution, and if one exists, we can find it in polynomial time. Then we also consider polynomials over $\mathbb{Z}$ and obtain similar results for perturbations $v \in \mathbb{Z}[x]$ of small height that achieve a $\gcd(f_0, f_1 + v)$ of large height (without any restrictions on their degree except that $\deg v \leq n$).

These results are natural polynomial analogues of those obtained recently by Howgrave-Graham (2001).

We prove that our algorithms solve the problem under rather restrictive assumptions. It remains an open question whether either a variant or some other algorithm can tackle a larger set of input values.

We also remark that finding multidimensional analogues, that is, constructing algorithms to find "small" perturbations $u_0, \dots, u_{s-1}$ of $f_0, \dots, f_{s-1}$ such that $\gcd(f_0 + u_0, \dots, f_{s-1} + u_{s-1})$ is "large" (in both number and polynomial cases) is another interesting direction of research.

2 Gcd of large degree

We write $f \operatorname{quo} g$ and $f \operatorname{rem} g$ for the quotient and remainder on division of f by nonzero g. Thus $f = (f \operatorname{quo} g) \cdot g + (f \operatorname{rem} g)$ and $\deg(f \operatorname{rem} g) < \deg g$.

The *degree sequence* of two univariate polynomials $f_0, f_1 \in \mathbb{F}[x]$ is the sequence of degrees $\deg f_0, \deg f_1, \deg f_2, \dots$ of the remainders $f_0, f_1, f_2, \dots$ in the Euclidean algorithm. Usually, but not always, $\deg f_{i-1} = 1 + \deg f_i$, and we say that f_0, f_1 *have a normal degree sequence* if that is the case for all i. We denote by M a polynomial multiplication time over $\mathbb{F}$, so that two polynomials of degree at most n can be multiplied with $O(\mathsf{M}(n))$ operations in $\mathbb{F}$. We may use $\mathsf{M}(n) = n \log n \log\log n$. In particular $\mathsf{M}(n) \in O\tilde{~}(n)$, where as usual $A \in O\tilde{~}(B)$ means that $|A| \leq c_1 B (\log(B+2))^{c_2}$ for some constants $c_1, c_2 > 0$; see von zur Gathen & Gerhard (2003, Chapter 8).

For our first result, we consider a field $\mathbb{F}$ and univariate polynomials $f_0, f_1 \in \mathbb{F}[x]$. We ask for perturbations $u_0, u_1 \in \mathbb{F}[x]$ of small degree so that the perturbed polynomials have a gcd of large degree. More precisely, we also have integers e_0, e_1, d, and we consider the set

$$\mathcal{U} = \{(u_0, u_1) \in \mathbb{F}[x]^2 \colon \deg u_i \leq e_i \text{ for } i = 0, 1,\ \deg \gcd(f_0 + u_0, f_1 + u_1) = d\}. \tag{1}$$

If e_i is negative, then the condition is meant to imply that $u_i = 0$. As an example, we can take $f_1, g, u_0 \in \mathbb{F}[x]$ of degrees n_1, m, e_0, respectively, with $e_0 < n_1 < m$, and $f_0 = g f_1 - u_0$, $d = n_1$, and $e_1 = n_1 - m - 1$. Then $\mathcal{U} = \{(u_0, 0)\}$, and the hypotheses in the theorem below are satisfied.

The algorithm below executes the Extended Euclidean Algorithm (EEA) for (f_0, f_1). It produces a finite series of "lines" (r_j, s_j, t_j) such that $s_j f_0 + t_j f_1 = r_j$, where $\deg r_j \leq n$ is strictly decreasing with growing j (see von zur Gathen & Gerhard 2003, Section 3.2). We have $s_1 = t_0 = 0$, and all other s_i and t_i are nonzero. Furthermore, since $\deg s_j$ and $\deg t_j$ are strictly increasing (see von zur Gathen & Gerhard 2003, Lemma 3.10), there is at most one "line" (r, s, t) with a prescribed degree for s (or t). We denote as $\operatorname{lc}(f)$ the leading coefficient of a polynomial f.

Algorithm 2. Approximate gcd of large degree.

Input: $f_0, f_1 \in \mathbb{F}[x]$ monic of degrees $n_0 > n_1$, respectively, coprime and with a normal degree sequence. Furthermore, integers d, e_0, e_1 with $d > 0$ and

$$e_0 < \min\{2d - n_1, n_0 - d\}, e_1 < \min\{2d - n_0, n_1 - d\}.$$

Output: $\mathcal{U}$ as in (1).

1. Execute the EEA with input (f_0, f_1).
2. Check if the EEA computes (r, s, t) with $sf_0 + tf_1 = r$ and $n_0 - \deg t = n_1 - \deg s = d$. If not, return $\mathcal{U} = \varnothing$.
3. Otherwise, if $s = 0$, then let $u_0 = -(f_0 \text{ rem } f_1)$ and return $\mathcal{U} = \{(u_0, 0)\}$ if $\deg u_0 \le e_0$, and else $\mathcal{U} = \varnothing$. If $t = 0$, then return $\mathcal{U} = \varnothing$.
4. {We now have $sf_0 + tf_1 = r$ and $st \ne 0$.} Compute
$$\begin{aligned} h_0 &= f_0 \text{ quo } t, \\ h_1 &= f_1 \text{ quo } s. \end{aligned}$$
If h_0 and h_1 are not associates, return $\mathcal{U} = \varnothing$.
5. Else, compute
$$\begin{aligned} &h = \mathrm{lc}(h_0)^{-1} h_0, \\ &\alpha = \mathrm{lc}(t)^{-1}, \\ &q_0 = \alpha t, \\ &q_1 = -\alpha s, \\ &u_i = q_i h - f_i \text{ for } i = 0, 1. \end{aligned}$$
6. If $\deg u_i \le e_i$ for $i = 0, 1$, then return $\mathcal{U} = \{(u_0, u_1)\}$, else return $\mathcal{U} = \varnothing$.

Theorem 3. *Let f_0, f_1, $n = n_0$, n_1, d, e_0, e_1 satisfy the input specification of Algorithm 2. Then the set $\mathcal{U}$ contains at most one element, and Algorithm 2 computes it with $O(\mathsf{M}(n) \log n)$ operations in $\mathbb{F}$.*

Proof. We have noted above that there is at most one "line" (r, s, t) in the EEA with $sf_0 + tf_1 = r$ and $n_0 - \deg t = n_1 - \deg s = d$. If there is no such line, then our algorithm returns $\mathcal{U} = \varnothing$. Otherwise we take that line.

We first have to check that any (u_0, u_1) returned by the algorithm is actually in the set $\mathcal{U}$. This is clear in Step 3. For an output in Step 6, we note that

$$\gcd(f_0 + u_0, f_1 + u_1) = \gcd(q_0 h, q_1 h) = h \gcd(s, t) = h,$$

since $\gcd(s, t) = 1$ (see von zur Gathen & Gerhard 2003, Lemma 3.8 (v)),

$$\deg h = \deg h_0 = \deg f_0 - \deg t = d,$$

and indeed $(u_0, u_1) \in \mathcal{U}$.

To show correctness of the algorithm it remains to show that if $\mathcal{U} \ne \varnothing$, then the algorithm indeed returns this set $\mathcal{U}$, and that $\mathcal{U}$ has at most one element.

So we now suppose that $\mathcal{U} \ne \varnothing$, let $(u_0, u_1) \in \mathcal{U}$, and $h = \gcd(f_0 + u_0, f_1 + u_1)$, so that $\deg h = d$. One first checks that the algorithm deals correctly with the two special cases $d = n_0$ and $d = n_1$. In the other cases, there exist uniquely determined $q_0, q_1 \in \mathbb{F}[x]$ such that

$$f_i = q_i h - u_i \quad \text{for} \quad i = 0, 1, \tag{4}$$

since $\deg u_i < 2d - n_{1-i} < d = \deg h$. Eliminating h from these two equations, we find

$$q_1 f_0 - q_0 f_1 = q_0 u_1 - q_1 u_0, \tag{5}$$

and call this polynomial $g = q_0 u_1 - q_1 u_0$. We have $\deg q_0 = n_0 - d < n_0$. Now g is nonzero, because otherwise f_0 would divide q_0, a polynomial of smaller degree than f_0, which would imply that $q_0 = 0$, a contradiction.

We have

$$\deg q_0 + \deg g \le n_0 - d + \max\{(n_0 - d) + e_1, (n_1 - d) + e_0\} < n_0,$$

since $e_i < 2d - n_{1-i}$ for $i = 0, 1$.

Thus (5) satisfies the degree inequalities of the EEA, and by the well-known uniqueness property of polynomial continued fractions (see, for example, von zur Gathen & Gerhard (2003, Lemma 5.15)), there exist a remainder r and corresponding Bézout coefficients s, t in the EEA for f_0 and f_1, and nonzero $\alpha \in \mathbb{F}[x]$ so that

$$sf_0 + tf_1 = r \text{ and } (g, q_1, -q_0) = \alpha(r, s, t).$$

Furthermore, since the Euclidean degree sequence is normal, α is a constant. We have $n_0 - \deg q_0 = n_0 - \deg t = d$, similarly $n_1 - \deg q_1 = d$, and $\deg u_i \le e_i < n_i - d = \deg q_i$, so that u_i equals the remainder of f_i on division by q_i, for $i = 0, 1$. It follows from (4) that indeed (u_0, u_1) is returned by the algorithm.

In particular, since at most one (u_0, u_1) is returned by the algorithm and it equals each element of $\mathcal{U}$ (if $\mathcal{U} \neq \varnothing$), $\mathcal{U}$ contains at most one element.

The cost for computing a single line in the Extended Euclidean Scheme is $O(\mathsf{M}(n) \log n)$; see von zur Gathen & Gerhard (2003, Algorithm 11.4). All other operations are not more expensive. □

In particular the cost of Algorithm 2 is in $O\tilde{}(n)$.

Figure 1 indicates at the bottom the triangle of values in the e_0-d-plane satisfying the restriction required for e_0, with large $n_0 = n_1 + 1$. There are trivial solutions $u_i = -f_i \operatorname{rem} h$ for $i = 0, 1$ when $e_0, e_1 \ge d - 1$, for any h of degree d; these form the area above the diagonal. We ran experiments with "random" polynomials, with and without a planted perturbed gcd. Values in the bottom triangle were, of course, correctly dealt with. We also ran the algorithm without any of the bounds d, e_0, e_1. Then it would typically compute $(u_0, u_1) \in \mathcal{U}$ with $e_0 = n_0 - d$ and $1 \le d \le n_1$, which is the dotted line in Figure 1. Planted gcds with $d < n_0/2$ were usually not detected.

3 Gcd of large height

We now look at the same problem in a different setting which we consider only for polynomials over $\mathbb{Z}$ (although it can be extended to polynomials over other fields and rings). Namely, we consider the case where the height $H(f) = \max\{|f_j| \colon 0 \le j \le n\}$ of a polynomial

$$f = \sum_{j=0}^{n} f_j x^j \in \mathbb{Z}[x]$$

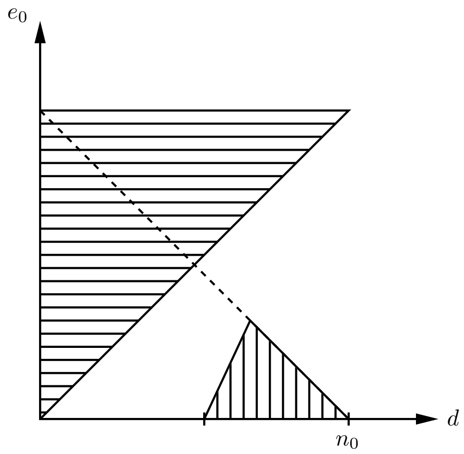

Fig. 1. The three areas – bottom triangle, half-plane, dotted line – are explained in the text.

is the measure of interest.

We first need to know that a large polynomial takes a small value only very rarely. Our bound is in fact the same as for the number of roots of the polynomial.

Lemma 6. *Let $h \in \mathbb{Z}[x]$ have degree $d \geq 3$, let $A \geq 2$ be an integer, and*

$$\mathcal{A} = \{a \in \mathbb{Z}\colon -A \leq a \leq A,\ |h(a)| \leq H(h)2^{-d}A^{-d^2}\}.$$

Then $\#\mathcal{A} \leq d$.

Proof. Let $a_0, \ldots, a_d \in \{-A, \ldots, A\}$ be $d+1$ distinct integers, and let $V = (a_i^j)_{0\leq i,j\leq d}$ be the corresponding $(d+1)\times(d+1)$ Vandermonde matrix. Each column of V has L_2-norm at most

$$(\sum_{0\leq i\leq d} A^{2i})^{1/2} \leq 2^{1/2}A^d.$$

We write $h = h_d x^d + \cdots + h_1 x + h_0$. Then

$$V \cdot (h_0, \ldots, h_d)^T = (h(a_0), \ldots, h(a_d))^T$$

The determinant of V is a nonzero integer, therefore from Cramer's rule and Hadamard's inequality we find

$$\begin{aligned} H(h) = \max_{0\leq k\leq d} |h_k| &\leq (2^{1/2}A^d)^d \left(\sum_{0\leq j\leq d} h(a_j)^2 \right)^{1/2} \\ &\leq (d+1)^{1/2}(2^{1/2}A^d)^d \max_{0\leq j\leq d} |h(a_j)| \leq 2^d A^{d^2} \max_{0\leq j\leq d} |h(a_j)|, \end{aligned}$$

which proves the claim. □

The bound of Lemma 6 can be improved slightly by estimating the determinant of V more carefully.

We also need the following statement which has essentially been proved in Howgrave-Graham (2001). For the sake of completeness we present a succinct proof. The gcd of two integers, at least one of which is nonzero, is taken to be positive.

Lemma 7. *Let F_0 and F_1 be integers. Then the set of all integers V with $|V| < |F_1|$ and*

$$\gcd(F_0, F_1 + V) \geq 2\sqrt{|F_0 V|}$$

can be computed in time polynomial in $\log(|F_0 F_1| + 1)$.

Proof. For an integer V we write

$$\Delta = \gcd(F_0, F_1 + V), \qquad G_0 = \frac{F_0}{\Delta}, \qquad G_1 = \frac{F_1 + V}{\Delta}.$$

We have $|F_1 + V| < 2|F_1|$. Then one verifies that

$$F_0 G_1 - F_1 G_0 = \frac{F_0 V}{\Delta} = \frac{(F_1 + V_1)(F_0 V_1 - F_1 V_0)}{G_1 \Delta^2}.$$

Hence

$$\left| \frac{F_0}{F_1} - \frac{G_0}{G_1} \right| \leq \frac{2|F_1|(|F_0 V|)}{|F_1| G_1^2 \Delta^2} \leq \frac{1}{2G_1^2}.$$

Thus G_0/G_1 is one of the convergents in the continued fraction expansion of F_0/F_1, and can be found in polynomial time. Thus $\Delta = F_0/G_0$ can take only polynomially many values. For each of them, we verify whether $V = G_1 \Delta - F_1$ satisfies the condition of the lemma. □

The gcd of polynomials f_0 and f_1 in $\mathbb{Z}[x]$ is monic if one of f_0 or f_1 is. We now consider for given $f_0, f_1 \in \mathbb{Z}[x]$ and integers D, E the set

$$\mathcal{V} = \{v \in \mathbb{Z}[x] \; : \; H(v) \leq E, \; H(\gcd(f_0, f_1 + v)) \geq D\}. \tag{8}$$

Algorithm 9. Approximate gcd of large degree.

Input: $f_0, f_1 \in \mathbb{F}[x]$ monic of degrees $n \geq n_1$ and heights H_0 and H_1, respectively, and such that $\gcd(f_0, f_1) = 1$. Furthermore, we are given a positive $\varepsilon < 1$ and positive integers D and E.

Output: $\mathcal{V}$ as in (8).

1. Initialize $\mathcal{V} = \varnothing$. Put $A = \lceil 4\varepsilon^{-1} n^2 \rceil$ and choose $n + 1$ distinct integers $a_0, \ldots, a_{n+1}$ uniformly at random in the interval $\{-A, \ldots, A\}$.
2. Evaluate $f_i(a_j)$ for $j = 0, \ldots, n$ and $i = 0, 1$.
3. For each $j = 0, \ldots, n$, compute continued fraction expansions of $f_0(a_j)/f_1(a_j)$ and find the set of all V_j with

$$\gcd\left(f_0(a_j), f_1(a_j) + V_j\right) \geq D 2^{-n} A^{-n^2}.$$

4. For each possible choice $(V_0, \ldots, V_n)$ compute the unique interpolation polynomial $v \in Q[x]$ of degree at most n with $v(a_j) = V_j$ for all j. If v satisfies the conditions in (8), then add v to $\mathcal{V}$.
5. Return $\mathcal{V}$.

Theorem 10. *Let f_0, f_1, ε, D, E be inputs to Algorithm 9. If*

$$E < H_1 2^{-n-1}(4\varepsilon^{-1}n^2 + 1)^{-n^2-n}$$

and

$$D \geq 2^{n+2}(4\varepsilon^{-1}n^2 + 1)^{n^2+n}(H_0 E)^{1/2},$$

then Algorithm 9 computes $\mathcal{V}$ with probability $1 - \varepsilon$ in time polynomial in $(\log(DH_1\varepsilon^{-1}))^n$.

Proof. Let $v \in \mathcal{V}$ as in (8), $h = \gcd(f_0, f_1 + v)$, and $d = \deg h$. We want to show that with probability at least $1 - \varepsilon$, v is found in step 4.

For $a_0, \ldots, a_n$ chosen in step 1, by Lemma 6 we see that with probability at least

$$\left(1 - \frac{4n}{2A+1}\right)^n > \left(1 - \frac{\varepsilon}{2n}\right)^n > 1 - \varepsilon,$$

we have simultaneously

$$|h(a_j)| \geq H(h)2^{-d}A^{-d^2} \geq D2^{-n}A^{-n^2} \qquad \text{and} \qquad |f_i(a_j)| \geq H_i 2^{-n}A^{-n^2}$$

for each $j = 0, \ldots, n$ and $i = 0, 1$, since each a_j has to avoid the at most $d + 2n \leq 3n$ "small" values of h, f_0 and f_1, and also the values $a_0, \ldots, a_{j-1}$. We also have

$$|f_1(a_j)| \geq H_1 2^{-n}A^{-n^2} > 2EA^n \geq |v(a_j)|$$

for each j, so that $f_1(a_j) + v(a_j) \neq 0$. Since the value of a polynomial gcd divides the gcd of the polynomial values, we find

$$\gcd\left(f_0(a_j), f_1(a_j) + v(a_j)\right) \geq |h(a_j)| \geq D2^{-n}A^{-n^2}.$$

On the other hand,

$$|f_i(a_j)| \leq 2H_i A^n \qquad \text{and} \qquad |v(a_j)| \leq 2EA^n$$

for each $j = 0, \ldots, n$ and $i = 0, 1$. Thus, under the conditions of the theorem we have

$$\begin{aligned} 2(|f_0(a_j)v(a_j)|)^{1/2} &\leq (16H_0 E A^{2n})^{1/2} \\ &\leq (16D^2 2^{-2n-4}(4\varepsilon^{-1}n^2 + 1)^{-2n^2-2n}A^{2n})^{1/2} \\ &\leq (D^2 2^{-2n}A^{-2n^2})^{1/2} = D2^{-n}A^{-n^2}. \end{aligned}$$

The above inequalities show that Lemma 7 applies and step 3 indeed finds the value $V_j = v(a_j)$. Thus Algorithm 9 works correctly. For any j, the set of all V_j in step 3 can be computed in time polynomial in $n \log(H_0 H_1 \varepsilon^{-1})$, by Lemma 7. Finally, the number of possibilities for the vector $(V_0, \ldots, V_n)$ is polynomial in $(\log DH_1\varepsilon^{-1})^n$. □

4 Acknowledgements

The first author's work was supported by the B-IT Foundation, and the second author's by ARC grant DP0556431. Thanks go to Daniel Loebenberger for help with the figure, and to Mark Giesbrecht.

References

1. DARIO A. BINI & PAOLA BOITO (2007). Structured Matrix-Based Methods for Polynomial ϵ-gcd: Analysis and Comparisons. *Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation ISSAC2007,* Waterloo, Ontario, Canada 9–16. URL `10.1145/1277548.1277551`.
2. JOACHIM VON ZUR GATHEN & JÜRGEN GERHARD (2003). *Modern Computer Algebra*. Cambridge, UK, 2nd edition. ISBN 0-521-82646-2, 800. URL `http://cosec.bit.uni-bonn.de/science/mca.html`. First edition 1999.
3. NICK HOWGRAVE-GRAHAM (2001). Approximate integer common divisor. 51–66. URL `http://www.springerlink.com/content/ak783wexe7ghp5db/`.