The GPGCD Algorithm with the Bézout Matrix for Multiple Univariate Polynomials

Boming Chi[1] and Akira Terui[2][0000−0003−0846−3643]

[1] Faculty of Pure and Applied Sciences,
University of Tsukuba, Tsukuba, 305-8571, Japan
hakumei-t@math.tsukuba.ac.jp
[2] Faculty of Pure and Applied Sciences,
University of Tsukuba, Tsukuba, 305-8571, Japan
terui@math.tsukuba.ac.jp
https://researchmap.jp/aterui

Based on the Greatest Common Divisor (GCD) computation, computing approximate GCD is one of the oldest studied areas in symbolic-numeric computation. We propose a new algorithm for computing approximate GCD of univariate polynomials with real coefficients [2] in a series of GPGCD algorithms ([1], [6], [7], [8]) we have studied.

For polynomial $F(x) = f_m x^m + \cdots + f_0 x^0 \in \mathbb{R}[x]$, the norm $\|F\|$ denotes the 2-norm defined as $\|F\|_2 := (f_m^2 + f_{m-1}^2 + \cdots + f_0^2)^{\frac{1}{2}}$. For a vector $(a_1, \ldots, a_n) \in \mathbb{R}^n$, the norm $\|(a_1, \ldots, a_n)\|$ denotes the 2-norm defined as $\|(a_1, \ldots, a_n)\|_2 := (a_1^2 + \cdots + a_n^2)^{\frac{1}{2}}$.

Definition 1 (Approximate GCD). *For polynomials*

$$F_i(x) = f_{im} x^m + \cdots + f_{i0} x^0 \in \mathbb{R}[x], \quad i = 1, \ldots, n, \quad f_{1m} \neq 0, \tag{1}$$

which are relatively prime in general, a positive integer d, and a real number $\epsilon > 0$, if there exist polynomials $\tilde{F}_1(x), \ldots, \tilde{F}_n(x)$ such that $\tilde{F}_1(x), \ldots, \tilde{F}_n(x)$ have the GCD $\tilde{H}$ of degree d, and $\left\|(\|F_1 - \tilde{F}_1\|, \ldots, \|F_n - \tilde{F}_n\|)\right\| \leq \epsilon$, we call $\tilde{H}$ an approximate GCD of polynomials $F_1(x), \ldots, F_n(x)$ with tolerance ϵ.

Algorithms for computing approximate GCD are classified into two categories: 1) for a given tolerance of $\left\|(\|F_1 - \tilde{F}_1\|, \ldots, \|F_n - \tilde{F}_n\|)\right\|$, make the degree of approximate GCD as large as possible, and 2) for a given degree d, minimize the magnitude $\left\|(\|F_1 - \tilde{F}_1\|, \ldots, \|F_n - \tilde{F}_n\|)\right\|$.

In both categories, algorithms based on various methods have been proposed, including the Euclidean algorithm, low-rank approximation of the Sylvester matrix or subresultant matrices, Padé approximation, and optimizations (for references, see those in our previous paper [1] and a recent paper [4]).

We have proposed a series of algorithms called "GPGCD algorithm" based on optimizations ([1], [6], [7], [8]). Our methods focus on the second category of the approximate GCD algorithms, solving the following problem.

Problem 1 (The approximate GCD problem for multiple polynomials). Let $F_1(x), \ldots, F_n(x)$ be univariate polynomials as shown in (1). For a given integer d with $m > d > 0$, find polynomials $\tilde{F}_1(x), \ldots, \tilde{F}_n(x)$ with making the

perturbation

$$\sqrt{\sum_{i=1}^{n} \|F_i(x) - \tilde{F}_i(x)\|^2} \tag{2}$$

as small as possible, such that there exists the exact GCD $\tilde{H}(x)$ of polynomials $\tilde{F}_1(x), \ldots, \tilde{F}_n(x)$ satisfying

$$\tilde{F}_i(x) = \tilde{f}_{im}x^m + \cdots + \tilde{f}_{i0}x^0 = \bar{F}_i(x) \times \tilde{H}(x), \tag{3}$$

where $\tilde{H}(x)$ is a polynomial of degree d and $\bar{F}_1(x), \ldots, \bar{F}_n(x)$ are relatively prime polynomials.

Our original GPGCD algorithms ([6], [7], [8]) use the Sylvester (or subresultant) matrices. Then, like other algorithms for computing GCD or approximate GCD using the Bézout matrix ([3], [5]), we have proposed a GPGCD algorithm using the Bézout matrix for two polynomials with real coefficients [1]. It aims to make the algorithm more efficient by taking advantage of the fact that the size of the Bézout matrix is smaller than that of the Sylvester matrix for the same inputs. Now, in the complete version of this paper [2], we give an extension for our algorithm using the Bézout matrix, which can accept more than two polynomials with real coefficients as inputs.

The proposed algorithm is shown as follows (see the definitions and details of the notations in the complete version of this paper [2]).

Algorithm 1 (The GPGCD algorithm for multiple polynomials with the Bézout matrix).
Inputs:

- $F_1(x), \ldots, F_n(x) \in \mathbb{R}(x)$: the given polynomials with $\max(\deg(F_i(x)), i = 1 \ldots n) = \deg(F_1(x)) > 0$,
- $d \in \mathbb{N}$: the given degree of approximate GCD with $d \leq \min(\deg(F_i(x)), i = 1 \ldots n)$,
- $\epsilon > 0$: the stop criterion with the modified Newton method,
- $0 < \alpha \leq 1$: the step width with the modified Newton method.

Outputs:

- $\tilde{H}(x)$: the approximate GCD, with $\deg(\tilde{H}) = d$,
- $\tilde{F}_1(x), \ldots \tilde{F}_n(x)$: the polynomials which are close to $F_1(x), \ldots, F_n(x)$, respectively, with the GCD $\tilde{H}$.

Step 1 Calculate the Bézout matrix $B = \mathrm{Bez}(F_1(x) \ldots F_n(x))$, the initial values $\boldsymbol{y}_0$, and the Jacobian matrix.
Step 2 Set the initial values $(\boldsymbol{s}_0, \boldsymbol{y}_0)$.
Step 3 Solve the linear system to find the search direction $\boldsymbol{d}_k$.

Step 4 If $\|\boldsymbol{d}_k\| < \epsilon$, obtain the $(\boldsymbol{s}^*, \boldsymbol{y}^*)$ as $(\boldsymbol{s}_k, \boldsymbol{y}_k)$, calculate the Bézout matrix $\tilde{B} = \mathrm{Bez}(\tilde{F}_1(x), \ldots, \tilde{F}_n(x))$ from polynomials $\tilde{F}_i(x),\ i = 1, \ldots, n$, then go to Step 5. Otherwise, let $(\boldsymbol{s}_{k+1}, \boldsymbol{y}_{k+1}) = (\boldsymbol{s}_k, \boldsymbol{y}_k) + \alpha \boldsymbol{d}_k$ and calculate the Bézout matrix and the Jacobian matrix with $(\boldsymbol{s}_{k+1}, \boldsymbol{y}_{k+1})$, then go to Step 3.

Step 5 Calculate the approximate GCD $\tilde{H}(x)$ with the Barnett's theorem. Calculate the approximate polynomials $\tilde{F}_i(x),\ i = 1, \ldots, n$. Return $\tilde{F}_i(x),\ i = 1, \ldots, n$ and $\tilde{H}(x)$.

We have implemented the proposed algorithm on the computer algebra system Maple 2021 and have executed experiments with the proposed algorithm and the original GPGCD algorithm for multiple polynomials [7] (abbreviated as mGPGCD-Syl algorithm) and the SNTLS-based algorithm [5]. We have generated 5 test groups of test polynomials, each group comprising 100 tests. We set the degree of the input polynomials for all groups to 10 and the degree of the approximate GCD to $3, 4, 5, 6, 7$, respectively.

The experimental results are shown as follows. Table 1 shows the number of test polynomials for which approximate GCD has not been computed in each algorithm. Cases in which the computation failed are those in which the calculation time exceeded approximately 2 hours, or the built-in function of Maple was determined not to converge for some reason. The result shows higher stability of the GPGCD algorithms. Table 2 shows the average computing time of each algorithm. The average computing time of the proposed algorithm is approximately from 1/10 to 1/8 of that of the mGPGCD-Syl algorithm and is approximately from 1/18 to 1/8 of that of the SNTLS-based algorithm. This shows the efficiency of the proposed algorithm. Table 3 shows the average of the norm of perturbation in each algorithm. The average of the norm of perturbation in the proposed algorithm is from 1 to 3 times of that of the mGPGCD-Syl algorithm and is sufficiently smaller than that of the SNTLS-based algorithm.

Table 1. Numbers of cases of test polynomials with no outputs for each group

	Numbers of cases of test polynomials with no outputs for each group		
Group	mGPGCD-Bézout	mGPGCD-Sylvester	SNTLS-based
1	0	0	13
2	0	0	35
3	0	0	19
4	0	0	37
5	1	0	34

References

1. Chi, B., Terui, A.: The GPGCD algorithm with the Bézout matrix. In: Computer Algebra in Scientific Computing (CASC 2020), Lecture Notes in Computer Sci-

Table 2. Comparison of the average computing time

	Average computing time (sec.)		
Group	mGPGCD-Bézout	mGPGCD-Sylvester	SNTLS-based
1	21.02	185.81	379.86
2	21.81	179.95	194.02
3	15.71	124.89	156.03
4	13.26	105.22	106.07
5	8.806	91.706	105.89

Table 3. Comparison of the average of the norm of perturbation

	Average of the norm of perturbation		
Group	mGPGCD-Bézout	mGPGCD-Sylvester	SNTLS-based
1	7.957	4.663	3.337×10^{10}
2	8.913	9.143	8.309×10^{7}
3	6.731	5.534	2.090×10^{9}
4	6.144	2.141	7.933×10^{10}
5	3.534	2.305	1.964×10^{16}

ence, vol. 12291, pp. 170–187. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-60026-6_10
2. Chi, B., Terui, A.: The GPGCD algorithm with the Bézout matrix for multiple univariate polynomials (2022), arXiv:2205.02984, preprint, 12 pages.
3. Diaz-Toca, G.M., Gonzalez-Vega, L.: Barnett's Theorems About the Greatest Common Divisor of Several Univariate Polynomials Through Bezout-like Matrices. J. Symb. Comput. **34**(1), 59–81 (2002). https://doi.org/10.1006/jsco.2002.0542
4. Nagasaka, K.: Relaxed NewtonSLRA for approximate GCD. In: Computer Algebra in Scientific Computing (CASC 2021), Lecture Notes in Computer Science, vol. 12865, pp. 272–292. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-85165-1_16
5. Sun, D., Zhi, L.: Structured low rank approximation of a bezout matrix. Math. Comput. Sci. **1**(2), 427–437 (Dec 2007). https://doi.org/10.1007/s11786-007-0014-6
6. Terui, A.: An iterative method for calculating approximate GCD of univariate polynomials. In: ISSAC '09, pp. 351–358. ACM Press, New York, New York, USA (2009). https://doi.org/10.1145/1576702.1576750
7. Terui, A.: GPGCD, an iterative method for calculating approximate GCD, for multiple univariate polynomials. In: Computer Algebra in Scientific Computing (CASC 2010), Lecture Notes in Computer Science, vol. 6244, pp. 238–249. Springer Berlin Heidelberg, Berlin, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15274-0_22
8. Terui, A.: GPGCD: An iterative method for calculating approximate GCD of univariate polynomials. Theor. Comput. Sci. **479**, 127–149 (2013). https://doi.org/10.1016/j.tcs.2012.10.023