Revisiting QRGCD and Comparison with ExQRGCD*

Kosaku Nagasaka†and Takaaki Masui‡
†Kobe University, 3-11 Tsurukabuto, Nada-ku, Kobe 657-8501 JAPAN
‡Kobe-Takatsuka High School, 9-1 Mikatadai, Nishi-ku, Kobe 651-2277 JAPAN
† nagasaka@main.h.kobe-u.ac.jp, ‡ masui.takaaki@gmail.com

1 Introduction

In this poster, we are interested in computing "approximate polynomial GCD": for the input polynomials $f(x), g(x) \in \mathbb{R}[x]$, we call the polynomial $d(x) \in \mathbb{R}[x]$ "approximate polynomial GCD" of tolerance $\varepsilon \in \mathbb{R}_{\geq 0}$ if it satisfies

$$f(x) + \Delta_f(x) = f_1(x)d(x),\ g(x) + \Delta_g(x) = g_1(x)d(x)$$

for some polynomials $\Delta_f(x), \Delta_g(x), f_1(x), g_1(x) \in \mathbb{R}[x]$ such that $\deg(\Delta_f) \leq \deg(f)$, $\deg(\Delta_g) \leq \deg(g)$, $\| \Delta_f \|_2 < \varepsilon \ \| f \|_2$ and $\| \Delta_g \|_2 < \varepsilon \ \| g \|_2$ where $\| \cdot \|_2$ denotes the 2-norm. Although there are many studies, we revisit the QRGCD algorithm[2] which is one of algorithms based on matrix decompositions and is also implemented as a part of the SNAP package of Maple. It is notable that the QRGCD algorithm is very simple and has been used as the benchmark algorithm for newly proposed algorithms. The framework of the QRGCD algorithm is as follows. For details, please refer the original paper[2].

1. Compute the QR decomposition of $Syl(f, g)$: $Syl(f, g) = QR$.
2. Find the gap between the k-th and $(k+1)$-th row vectors $\vec{r}_k, \vec{r}_{k+1}$ of R and form the polynomial with coefficients $\vec{r}_k$, which is an approximate polynomial GCD (or its factor).
3. Apply the same procedures to the reversal polynomials of cofactors since R may not have the approximate common divisor whose roots are outside the unit circle in the complex plane.

However, since QRGCD was proposed in the early stage of approximate GCD, its theoretical background is not enough analyzed from the current theoretical point of view and the official implementation is different from the paper. For example, Bini and Boito[1] reported that QRGCD failed to recognize the correct degree of GCD for polynomials with small leading coefficients. This result is caused by the preconditioning routine in the official implementation hence QRGCD works well for such polynomials. Therefore, our aim consists of two parts: 1) verifying the efficiency of QRGCD with much theoretical considerations, and 2) improving the framework and algorithm to be more accurate and able to satisfy the given tolerance.

2 Notable Facts on QRGCD and its Implementation in SNAP

Recently, the concept of "structured perturbation" is important in the theory of approximate GCD. However, at the time of QRGCD proposed, this concept is not widely discussed hence there are unclear statements in the original paper from this point of view. Analyzing their theory from this concept could be interesting. For example, any relationship between the QR factoring and structured perturbation, the reason that the QR factoring can not detect the roots outside the unit circle and so on.

*This work was supported in part by JSPS KAKENHI Grant Number 22700011.

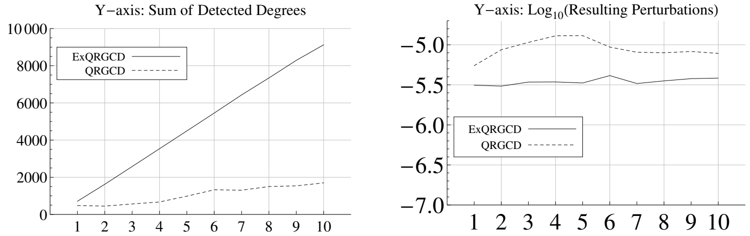
Moreover, we found the 4 significant differences between the original algorithm and the SNAP implementation. According to our personal conversations, some of them are implemented by the original authors and others may be by H. Kai, the person implemented it in the SNAP package. The differences are 1) the preconditioning routine “find non-zero terms”, 2) the matrix norm used, 3) the polynomials to be applied to the algorithm “Split”, and 4) the fail-safe retry loop. The first one may be the cause that many people think QRGCD is weak for polynomials with small leading coefficients. Without this, QRGCD works well for such polynomials. Other differences seem to be some techniques to make QRGCD working well.

3 Improved QRGCD Algorithm (ExQRGCD)

We refine the framework of QRGCD from the different approach with recent theoretical results of approximate polynomial GCD and propose the improved algorithm called “ExQRGCD”. The most notable difference is that our algorithm detects a row vector of R by estimating relative distance from the expected approximate GCD while QRGCD detects by estimating absolute distance. As a result, ExQRGCD works more accurately. For example, it works for the following polynomials (QRGCD does not work well for this kind of polynomials unfortunately). For $i = 1, \ldots, 10$, we have generated 100 pairs of (f, g) such that

$$f(x) = d(x)\prod_{j=1}^{2i}(x-\omega_{f,j})\prod_{j=1}^{2i}(x-\hat{\omega}_{f,j}),\ g(x) = d(x)\prod_{j=1}^{2i}(x-\omega_{g,j})\prod_{j=1}^{2i}(x-\hat{\omega}_{g,j})$$

where $d(x) = \prod_{j=1}^{3i}(x-\omega_{d,j})\prod_{j=1}^{3i}(x-\hat{\omega}_{d,j})$, $\omega_{\cdot,j} = O(10^{-2}), \hat{\omega}_{\cdot,j} = O(10^{2})$ is randomly chosen, $f(x), g(x)$ are normalized (i.e. $\|f(x)\|_2 = \|g(x)\|_2 = 1$) and rounded with $Digits := 10$. We computed with tolerance 10^{-5}. Figure 1 shows the result that ExQRGCD is explicitly better than QRGCD though as for computing time, ExQRGCD is 39.8 times slower than QRGCD (note that QRGCD outputs failure for 62% pairs so computing time is very fast for the rest easy cases). The average of resulting perturbations of ExQRGCD is also better. For other random generated examples, ExQRGCD is almost 2 times slower than QRGCD since ExQRGCD is more conservative than QRGCD for detecting approximate GCD hence it computes QR decompositions several times (this is more than that of QRGCD in general).



Number of fail events or wrong Δ_* (QRGCD)											
i=1	2	3	4	5	6	7	8	9	10	Δ_f	Δ_g
29	63	68	71	64	60	66	66	69	68	110	119

Number of fail events or wrong Δ_* (ExQRGCD)											
i=1	2	3	4	5	6	7	8	9	10	Δ_f	Δ_g
0	0	0	0	0	0	0	0	0	0	0	0

Figure 1: Sum of Detected Degrees and Resulting Perturbations (failure is not counted)

We note that our preliminary implementations on Maple and written in C, and generated polynomial data are available: “`http://wwwmain.h.kobe-u.ac.jp/~nagasaka/research/snap/issac2013/`”.

References

[1] D. A. Bini and P. Boito. A fast algorithm for approximate polynomial GCD based on structured matrix computations. In *Numerical methods for structured matrices and applications*, volume 199 of *Oper. Theory Adv. Appl.*, pages 155–173. Birkhäuser Verlag, Basel, 2010.

[2] R. M. Corless, S. M. Watt, and L. Zhi. *QR* factoring to compute the GCD of univariate approximate polynomials. *IEEE Trans. Signal Process.*, 52(12):3394–3402, 2004.