数理解析研究所講究録
第1652巻 2009年 55-62

安定化剰余列算法の改良

讃岐 勝
MASARU SANUKI
筑波大学 数理物質科学研究科
GRADUATE SCHOOL OF PURE AND APPLIED SCIENCES, UNIVERSITY OF TSUKUBA *

Abstract

本稿では、浮動小数係数多項式の剰余列により近似 GCD を精度よく計算する安定化剰余列算法を QRGCD 法と比較しながら改良する。また、EZ-GCD 算法や PC-PRS 算法の悪条件性を回避する多変数多項式の近似 GCD 算法を提案する。

1 はじめに

1980年末、佐々木と野田らにより**近似代数**の概念が提唱されて以来、世界中で多種多様な研究が行われ、最近の計算機代数の国際会議での大きなトピックの1つとなっている。その中で、**近似 GCD(最大公約子)**の計算は基本的な算法の1つであり、これまで次の計算法が提案されている。

- 剰余列算法
  1変数多項式：効率的であるが、微小主係数の多項式が現れた場合に計算が不安定 [SN89]
  ⇒ 多項式のピボティングにより計算の安定化に成功 [SS07](次頁で解説)
  多変数多項式：次数と項数が爆発的に増えて時間がかかる (中間式膨張)[ONS91]
- 数値計算の算法に基づく算法
  - QR 分解に基づく算法 (QRGCD 法) : Sylvester 行列 S を $S = QR$ と分解 (Q : 直交行列、R : 上三角行列) [OST97, ZMF00, CWZ04]
  - 特異値分解に基づく算法：Sylvester 行列の部分行列 S_k の零空間を計算する
    1変数多項式：精度よく計算できる [CGTW95, Zen04]
    多変数多項式:次数が低くても行列のサイズが大きくなり計算に時間がかかる [GKMYZ04, ZD04]
  - 対称行列の displacement を利用した高速算法 [Zhi03, BB07]
- 打ち切りべき級数を用いる算法 (多変数多項式)
  - EZ-GCD 算法 : Hensel 法に基づく効率よい算法だが、計算が不安定になることが多くある [ZN00]
  - PC-PRS 算法：必要な項だけから剰余列算法によって効率よく計算できる [San05]
    ⇐ 中間式膨張による計算効率、精度の低下の問題を解決
  - PC-GivensGCD 法 : QRGCD 法と PC-PRS 算法の融合 [SS07]
- その他
  1変数多項式：quasi-GCD[Sch85]、ϵ-GCD[EGL97]、near-GCD[HS97]、Padé-GCD[Pan01]、構造行列による算法 [KYZ05]、Ruppert 行列を用いる算法 [長坂 08]、など
  多変数多項式：modular 算法 [CGTW95]、構造行列による算法 [KYZ06]、など

*sanuki@math.tsukuba.ac.jp

筆者と佐々木は、浮動小数係数多項式の剰余列計算における**微小主係数問題**に対して、行列計算における軸交換のテクニックを多項式演算に組み込むことによって剰余列による近似 GCD 計算を安定させた (**安定化剰余列算法**)[SS07]。しかし、アイデアの段階を述べたに過ぎず、精度と効率の面において改良する必要がある。また、多変数多項式の剰余列計算では PC-PRS 算法を用いて高次項を打ち切りながら計算を行い、近似 GCD の候補を得る。その候補から近似 GCD を定めるためには主係数のべき級数除算が必要となり、主係数の定数項が小さい時に大きな桁落ちを起こす (**主係数の特異性**)[SS07]。そこでリスクの少ない算法の開発が必要となる。目的を達成するため、鈴木 [Suz93] により提案された**拡張 Euclid 構成**を用いる。この算法は PC-PRS 算法および EZ-GCD 算法と同じくらい効率がよい。

本稿では、i) 安定化剰余列算法の改良、ii) 主係数の特異性を避ける算法の考察、を行う。**2** では、[SS07] で提案した安定化剰余列算法の改良を行う。**3** では、打ち切りべき級数を用いる算法の説明をし、拡張 Euclid 構成の近似化を行う。近似化した算法は主係数の特異性の回避ならず他の算法の悪条件性を回避する算法であることを示す。**4** でまとめる。

$F(x,u)$ を主変数 x、従変数 $(u)=(u_1,\ldots,u_\ell)$ の多変数多項式とする。、$\mathrm{lc}(F)$ と $\deg(F)$ は F の主係数と次数をそれぞれ表す。多項式ノルムを $||F||$ と表し、数係数の絶対値の最大値で定義する。なお、与える多項式 F は $||F||=1$ と規格化されているものとする。$\mathrm{appgcd}(F,G;\varepsilon)$ は F と G の許容度 ε の近似 GCD を表す。$\mathrm{quo}(F,G)$ と $\mathrm{rem}(F,G)$ は F を G で割った商と余りをそれぞれ表す。$\mathrm{elim}(F,G)$ は G による F の主項消去を表す ;

$$\mathrm{elim}(F,G)=\begin{cases} F-\mathrm{lc}(F)/\mathrm{lc}(G)x^{\deg(F)-\deg(G)}G & \textit{if } \deg(F)\geq\deg(G),\\ F & \textit{otherwise.}\end{cases}$$

2 安定化剰余列算法

F と G を多項式とする ($\deg(F)-\deg(G)=d\geq 0$)。$|\mathrm{lc}(G)|/|\mathrm{lc}(F)|\ll 1$ のとき、F の G による剰余 H は $H=\mathrm{rem}(F,G)=F-QG\approx \text{const.}\times\mathrm{rest}(G)$ となる。ただし、$\mathrm{rest}(G)$ は G から主項を除いたものを表す。H は G にほぼ等しいため、$\mathrm{rem}(G,H)$ の計算で大きな桁落ちが起きる。この桁落ちメカニズムを**自己簡約**と呼ぶ。安定化剰余列算法では自己簡約を次のように回避した : Step 1) $H=(x^d-a)G-\mathrm{lc}(G)/\mathrm{lc}(F)F$ を計算する。ただし、a は $\deg(a)<d$ の多項式である。Step 2) F と H の剰余を計算する ($|\mathrm{lc}(H)|/|\mathrm{lc}(F)|\ll 1$ ならば、G を H で置き換え同様の操作を行う)。このとき自己簡約による桁落ちは起きない。しかし、上のアルゴリズムには大きな問題がある。

1. いつピボッティングを行うか? ($|\mathrm{lc}(G)|/|\mathrm{lc}(F)|\ll 1$ の判定)
   [SS07] では、$|\mathrm{lc}(G)|/|\mathrm{lc}(F)|<\mathcal{S}=0.25$ のときピボッティングを行った。$\mathcal{S}$ の大きさにより次の弊害が起きることを実験より確認している。
   - $\mathcal{S}$ を 1 に近づけた場合 : ピボッティングの回数が多くなるため計算が遅くなる。
   - $\mathcal{S}$ を小さくした場合 : 精度が悪くなる。
2. a をどのように定めるか?
   $a=0$ または a が小さい場合、定数項の小さい多項式 H が生成され計算が不安定になる。計算を安定させるため、x^d-a と F が近似共通因子を持たないように $|a|\in[1.2,1.5]$ である数によって定めた ; $m=n+1$ のとき、$a\in\mathbb{C}$ は QRGCD 法の計算のときに現れることを確認している。

本稿では、効率 (停止性) を保証しながら $\mathcal{S}$ を 1 に近づけるため、QRGCD 法と比較しながら改良を行う。同時に a の構成ついて考える。

2.1 安定剰余列の改良

$f(x), g(x) \in \mathbb{C}[x]$ は次で表される 1 変数多項式とする。

$$f(x) = f_m x^m + \cdots + f_0, \quad g(x) = g_n x^n + \cdots + g_0 \quad (f_m, g_n \neq 0, m \geq n). \tag{1}$$

f と g からなる Sylvester 行列 $S(f,g)$ の各行にそれぞれ名前をつける：

$$S(f,g) = \begin{pmatrix} f_m & f_{m-1} & \cdots & \cdots & \cdots \\ & f_m & f_{m-1} & \cdots & \cdots \\ & & f_m & f_{m-1} & \cdots \\ & & & \ddots & \cdots \\ g_n & g_{n-1} & \cdots & \cdots & \cdots \\ & g_n & g_{n-1} & \cdots & \cdots \\ & & g_n & g_{n-1} & \cdots \\ & & & \ddots & \cdots \end{pmatrix} \begin{matrix} \longleftarrow f^{(1,0)}\text{-row} \\ \longleftarrow f^{(2,0)}\text{-row} \\ \longleftarrow f^{(3,0)}\text{-row} \\ \vdots \\ \longleftarrow g^{(1,0)}\text{-row} \\ \longleftarrow g^{(2,0)}\text{-row} \\ \longleftarrow g^{(3,0)}\text{-row} \\ \vdots \end{matrix} \tag{2}$$

$|\mathrm{lc}(g)|/|\mathrm{lc}(f)| < S$ を仮定する (微小主係数と判断)。$f^{(1,0)}$-row を軸とし、Givens 回転により $g^{(1,0)}$-row の非零先頭要素を消去する。

$$\begin{pmatrix} \tau_1 & \sigma_1 \\ -\sigma_1 & \tau_1 \end{pmatrix} \begin{pmatrix} f^{(1,0)}\text{-row} \\ g^{(1,0)}\text{-row} \end{pmatrix} = \begin{pmatrix} f_m^{(1,1)} & f_{m-1}^{(1,1)} & f_{m-2}^{(1,1)} & \cdots \\ 0 & g_{n-1}^{(1,1)} & g_{n-2}^{(1,1)} & \cdots \end{pmatrix} \begin{matrix} \longleftarrow f^{(1,1)}\text{-row} \\ \longleftarrow g^{(1,1)}\text{-row} \end{matrix}.$$

ここで $\tau_1 = f_m/\sqrt{|f_m|^2 + |g_n|^2}$ かつ $\sigma_1 = g_n/\sqrt{|f_m|^2 + |g_n|^2}$ である。$g^{(1,1)}$-row は $h_1 = \mathrm{elim}(x^{m-n}g, f)$ に対応する。次に、$g^{(1,1)}$-row を軸に選び次の消去を行う。

Step 2-1 : $g^{(1,1)}$-row と $f^{(2,0)}$-row の Givens 回転による消去を行う。

$$\begin{pmatrix} \tau_2 & \sigma_2 \\ -\sigma_2 & \tau_2 \end{pmatrix} \begin{pmatrix} f^{(2,0)}\text{-row} \\ g^{(1,1)}\text{-row} \end{pmatrix} = \begin{pmatrix} 0 & 0 & f_{m-1}^{(2,1)} & f_{m-2}^{(2,1)} & \cdots \\ 0 & g_n^{(1,2)} & g_{n-1}^{(1,2)} & g_{n-2}^{(1,2)} & \cdots \end{pmatrix} \begin{matrix} \longleftarrow f^{(2,1)}\text{-row} \\ \longleftarrow g^{(1,2)}\text{-row} \end{matrix}.$$

$f^{(2,1)}$-row と $g^{(1,2)}$-row はそれぞれ $\mathrm{elim}(f, xh_1)$ と $\tau_2 x h_1 + \sigma_2 f$ に対応する。このとき、$|\mathrm{lc}(h_1)|/|\mathrm{lc}(f)| > S$ ならば h_1 と $f^{(2,1)}$-row に対応する多項式の近似 GCD を計算する。そうでないならば **step 2-2** に進む。
Step 2-2 : $g^{(1,2)}$-row と $g^{(2,0)}$-row の Givens 回転による消去を行い、それぞれ $g^{(1,3)}$-row と $g^{(2,1)}$-row に変換される。$g^{(2,1)}$-row は $\tau_3(\tau_2 x h_1 + \sigma_2 f) - \sigma_2 x^{m-n-1} g = \tau_3 \sigma_2 f - \mathrm{elim}(\mathrm{elim}(f, x^{m-n}g), x^{m-n-1}g)$ に対応する。$\mathrm{elim}(\mathrm{elim}(f, x^{m-n}g), x^{m-n-1}g) = \mathrm{elim}(h_1, x^{m-n-1}g) = h_2$ とおく。$m = n+1$ のとき、この $h_2 = \mathrm{rem}(f,g)$ $(\deg(h_2) < \deg(g))$ である。この場合、f と h_2 の近似 GCD を計算する。$m > n+1$ のとき **step 3-1** および **step 3-2** に進む。

Step 3-1 : $\mathrm{elim}(f, h_2)$ を計算する。もし、$|\mathrm{lc}(\mathrm{elim}(f,h_2))|/|\mathrm{lc}(f)| > S$ ならば、f と h_2 の近似 GCD を計算する。そうでないならば、**step3-2** に進む。
Step 3-2 : $h_3 = \mathrm{elim}(h_2, x^{m-n-2}g)$ を計算する。$m = n+2$ のとき、$h_3 = \mathrm{rem}(f,g)$ なので f と h_3 の近似 GCD を計算する。そうでないならば、次の steps に進む。

以上が QRGCD 法を基とした安定化剰余列算法の改良であり、アルゴリズムは次のように書ける。

アルゴリズム 1 (改良された安定化剰余列算法)

Input:	f *and* $g \in \mathbb{C}[x]$ *with* $\deg(f) \geq \deg(g)$, *and* $0 \leq \varepsilon < 1$.
Output:	$\mathrm{appgcd}(f, g; \varepsilon)$.

$h = \mathrm{elim}(f, g)$;
if $||h||/||g|| \leq \varepsilon$ *then return* g;
if $|\mathrm{lc}(g)|/|\mathrm{lc}(f)| < \mathcal{S}$ *then goto LP*
else compute $\mathrm{appgcd}(h, g; \varepsilon)$;
LP: *if* $|\mathrm{lc}(h)|/|\mathrm{lc}(f)| \geq \mathcal{S}$ *then* $\tilde{f} = \mathrm{elim}(f, h)$; *compute* $\mathrm{appgcd}(\tilde{f}, g; \varepsilon)$;
else $h = \mathrm{elim}(h, g)$;
if $\deg(h) < \deg(g)$ *then compute* $\mathrm{appgcd}(f, h; \varepsilon)$;
goto LP;
end;

命題 1 (停止性)
アルゴリズム 1 は停止する。

証明 はじめに $h = \mathrm{elim}(f, g)$ を計算する。$|\mathrm{lc}(h)|/|\mathrm{lc}(f)| \geq \mathcal{S}$ ならば、$\mathrm{appgcd}(\tilde{f}, g; \varepsilon)$ を計算する。このとき、$\deg(\tilde{f}) < \deg(f)$ であるから f の次数が低くなった。そうでない場合、$\tilde{h} = \mathrm{elim}(g, h)$ を計算する。ただし、$\deg(\tilde{h}) < \deg(h)$ である。この計算後、f と $\tilde{h}$ の主係数の大きさに注意して同様の計算を進める。このとき、f の次数を減らす多項式 $\tilde{f}$ か、次数条件 $\deg(\hat{h}) < \deg(g)$ を満たす多項式 $\hat{h}$ が生成できる。この場合、f と $\hat{h}$ の近似 GCD を計算する。いずれの場合にも、f か g より次数の低い多項式を自己簡約無しで構成できる。故に、命題は正しい。 ∎

注意 1
上のアルゴリズムの中で、$\mathcal{S}$ の大きさは停止性にそれほど多くの影響をあたえず、大きい方が計算の精度が安定する。故に $\mathcal{S} = 1$ として近似 *GCD* を計算する。 ∎

3 打ち切りべき級数を用いる多変数近似 GCD 算法

$F(x, u)$ と $G(x, u)$ を $\mathbb{C}$ 上のの多変数多項式とする。打ち切りべき級数を用いる算法は、次の手順で計算を近似 GCD 行う。

1. 展開点 $s \in \mathbb{C}^\ell$ を選び、多項式イデアルを $I = \langle u_1 - s_1, \ldots, u_\ell - s_\ell \rangle$ とする。近似 GCD の候補 $\tilde{C}(x, u) \equiv C(x, u) \pmod{I^{j+1}}$ を計算する (j は C の従変数 u に関する全次数より大きい)。
2. 主係数によるべき級数除算を行い、主係数同志の近似 GCD を掛け合わせる :

$$C \equiv \mathrm{appgcd}(\mathrm{lc}(F), \mathrm{lc}(G); \varepsilon) \times \tilde{C}/\mathrm{lc}(\tilde{C}) \pmod{I^{j+1}}. \tag{3}$$

このとき、C は $F(x, u)$ と $G(x, u)$ の近似 GCD である。

$\mathrm{lc}(\tilde{C})$ の定数項 c_0 とする。$|c_0| \ll ||\mathrm{lc}(\tilde{C})||$ ならば、べき級数除算 $1/\mathrm{lc}(\tilde{C})$ で得られる各項のノルムは次数が上がるごとに大きくなる。$\tilde{C}$ および C の係数のノルムは $O(1)$ であるが、$\tilde{C}/\mathrm{lc}(\tilde{C}) \pmod{I^{j+1}}$ の各係数のノルムも $\leq O(1)$ であるため、主係数によるべき級数除算において桁落ちが起きることがわかる (主係数の特異性)。展開点 $s \in \mathbb{C}^\ell$ を変更によって桁落ちは避けられるが、上のようなべき級数除算を用いずに始めに決めた展開点から近似 GCD の候補 (モニックな近似 GCD の候補) を計算したい。本稿では、鈴木 [Suz93] による**拡張 Euclid 構成**を用いる。この算法による GCD 計算は PC-PRS 算法および EZ-GCD 算法と同じくらい効率がよい。

3.1 拡張 Euclid 構成

展開点 $s \in \mathbb{C}^\ell$ を選び、PC-PRS 算法によって modulo I^j での F と G の GCD $\tilde{C}^{(j-1)}(x,u)$ および余因子 $A^{(j-1)}(x,u)$ と $B^{(j-1)}(x,u)$ を計算する。

$$\tilde{A}^{(j-1)}(x,u)\,F(x,u) + \tilde{B}^{(j-1)}(x,u)\,G(x,u) \equiv \tilde{C}^{(j-1)}(x,u) \pmod{I^j}. \tag{4}$$

このとき、次を満たす $\tilde{C}^{(j)}(x,u)$ と $\tilde{A}^{(j)}(x,u), \tilde{B}^{(j)}(x,u)$ を再計算することなく計算したい：

$$\tilde{A}^{(j)}(x,u)F(x,u) + \tilde{B}^{(j)}(x,u)G(x,u) \equiv \tilde{C}^{(j)}(x,u) \pmod{I^{j+1}}. \tag{5}$$

(4) から拡張 Euclid 構成によって $\tilde{C}^{(j)}(x,u)$ と $\tilde{A}^{(j)}(x,u), \tilde{B}^{(j)}(x,u)$ を除算から簡単に計算することができる。求める多項式を次のように表す。

$$\begin{cases} \tilde{C}^{(j)}(x,u) = \tilde{C}^{(j-1)}(x,u) + \delta\tilde{C}^{(j)}(x,u), & \deg(\delta\tilde{C}^{(j)}) < \deg(\tilde{C}^{(0)}) \\ \tilde{A}^{(j)}(x,u) = \tilde{A}^{(j-1)}(x,u) + \delta\tilde{A}^{(j)}(x,u), \\ \tilde{B}^{(j)}(x,u) = \tilde{B}^{(j-1)}(x,u) + \delta\tilde{B}^{(j)}(x,u). \end{cases} \tag{6}$$

ただし、$\delta\tilde{C}^{(j)}$ および $\delta\tilde{A}^{(j)}, \delta\tilde{B}^{(j)}$ は全次数 j の項の和である。これらの多項式を (5) に代入すると、

$$\begin{aligned} (\tilde{A}^{(j-1)} + \delta\tilde{A}^{(j)})F + (\tilde{B}^{(j-1)} + \delta\tilde{B}^{(j)})G &\equiv \tilde{C}^{(j-1)} + \delta\tilde{C}^{(j)} \pmod{I^{j+1}}, \\ [\tilde{A}^{(j-1)}F + \tilde{B}^{(j-1)}G]_j^j + [\delta\tilde{A}^{(j)}F^{(0)} + \delta\tilde{B}^{(j)}G^{(0)}]_j^j &= \delta\tilde{C}^{(j)}. \end{aligned} \tag{7}$$

ただし、$[P]_n^m$ は多項式 P の従変数 u に関する全次数 $n \sim m$ の項の和を表す。$\tilde{C}^{(0)}(x) = \tilde{A}^{(0)}(x)F^{(0)} + \tilde{B}^{(0)}(x)G^{(0)}$ なので、$\tilde{C}^{(0)}$ の除算によって次式を得る。

$$\begin{aligned} \delta\tilde{C}^{(j)} &= \mathrm{rem}([\tilde{A}^{(j-1)}F + \tilde{B}^{(j-1)}G]_j^j, \tilde{C}^{(0)}), \\ (\delta\tilde{A}^{(j)}, \delta\tilde{B}^{(j)}) &= -Q^{(j)}(x,u) \times (\tilde{A}^{(0)}, \tilde{B}^{(0)}). \end{aligned} \tag{8}$$

ただし、$Q^{(j)}(x,u) = \mathrm{quo}([\tilde{A}^{(j-1)}F + \tilde{B}^{(j-1)}G]_j^j, \tilde{C}^{(0)})$ である。このとき、次がわかる。

補題 2
非負整数 j について、$\mathrm{lc}(\tilde{C}^{(j)}(x,u)) \in \mathbb{C}$ である。 ∎

PC-PRS 算法を用いて GCD の計算をするとき、打ち切り次数 E は大きく見積もられることが多い (ほぼ、与えられた多項式の従変数に関する全次数)。そこで、打ち切り次数を小さく設定し (実用上、$E = 3$ か $E = 4$ で十分)、小さすぎた場合に GCD の候補の全次数を上げる目的で算法は開発された。しかし、近似 GCD 計算では打ち切り次数が大きくなると項数も増え誤差が積もりやすくなる。そこで、次の手順で近似 GCD を計算する。

1. 1 変数多項式の近似 GCD を計算する。
   余因子も必要となるが、必ずしも次数条件 (すなわち一意性) は満たさなくともよい。
2. 拡張 Euclid 構成によって従変数の全次数を 1 つずつ上げる。
3. 近似 GCD の主係数を定める。

補題 3 (近似 GCD の許容度)
$\tilde{C}^{(0)}(x)$ を $F^{(0)}$ と $G^{(0)}$ の許容度 ε の近似 GCD とする。このとき、$\tilde{C}^{(j)}(x,u) \pmod{I^{j+1}}$ は許容度 $O(\varepsilon)$ の近似 GCD の候補である。

証明 $\tilde{C}^{(j')}(x,u)$, $\tilde{A}^{(j')}(x,u)$, $\tilde{B}^{(j')}(x,u)$ およびそれぞれの摂動項 $\Delta_A^{(j')}(x,u)$, $\Delta_B^{(j')}(x,u)$, $\Delta_C^{(j')}(x,u)$ が次の式を満たすと仮定する。

$$(\tilde{A}^{(j')} + \Delta_A^{(j')})F + (\tilde{B}^{(j')} + \Delta_B^{(j')})G \equiv \tilde{C}^{(j')} + \Delta_C^{(j')} \pmod{I^{j'+1}}.$$

このとき、$Q^{(j)}(x,u)$ と $\delta\tilde{C}^{(j)}$ は次式で表される。

$$\begin{aligned} Q^{(j)}(x,u) &= \mathrm{quo}([\tilde{A}^{(j-1)}F + \tilde{B}^{(j-1)}G]_j^j, \tilde{C}^{(0)} + \Delta_C^{(0)}), \\ \delta\tilde{C}^{(j)} &= \mathrm{rem}([\tilde{A}^{(j-1)}F + \tilde{B}^{(j-1)}G]_j^j, \tilde{C}^{(0)} + \Delta_C^{(0)}). \end{aligned}$$

故に、$\delta\tilde{A}^{(j)}, \delta\tilde{B}^{(j)}$ および $\delta\tilde{C}^{(j)}$ は許容度 $O(\varepsilon)$ の多項式の積および商によって定まることがわかる。積と商も許容度 $O(\varepsilon)$ の多項式であるので [Sas93]、主張は正しい。 ∎

例 1 (近似 GCD の計算 [ZN00])
次の 2 変数多項式 $F(x,u)$ と $G(x,u)$ の近似 GCD を計算する。

$$\begin{aligned} F(x,u) &= (x^2+u^2+1.01)(x^2+xu+u^2+1.12), \\ G(x,u) &= (x^2+u^2+1.01)(x^2+xu+1.02) + 10^{-4}(x+u). \end{aligned}$$

展開点を原点に選ぶ : $s=(0)$。最初に 1 変数多項式 $F^{(0)}$ と $G^{(0)}$ の近似 GCD $C^{(0)}$ を計算する。

$$C^{(0)} = 1.000000000 + .9900994526\,x^2 - 0.0009358224912\,x.$$

拡張 $Euclid$ 構成によって $C^{(i)}(x,u)$ $(i=1,2)$ が計算できる。

$$\begin{aligned} C^{(1)} &= 1.000000000 + 0.9900994526\,x^2 - 0.0009358224912\,x - 0.0009900989321\,u, \\ C^{(2)} &= 1.000000000 + 0.9900994526\,x^2 - 0.0009358224912\,x - 0.0009900989321\,u \\ &\quad + 0.009385357494\,xu^2 + 0.990103163\,u^2. \end{aligned}$$

$C^{(2)}$ は $F(x,u)$ と $G(x,u)$ の近似 GCD である。 ∎

上の例において、EZ-GCD 法により計算すると、初期因子に近似共通因子が存在するので大きな桁落ちを起こす [SY98]。一方、拡張 Euclid 構成は除算の積の計算のため、不安定になることなく近似 GCD を計算できる。

4 まとめ

QRGCD 法と比較することによって、安定化剰余列算法の改良を行った。係数の大きさによって算法を停止させる以前の方法より、次数によって算法の停止性を保証したことによって算法は安定した。しかし、これ以上の QRGCD 法と比較しながら算法を改良しても QRGCD 法に近付くだけで計算効率に優れた剰余列算法の良さが失われる。剰余列算法をさらに安定化させるには別のアプローチから研究を進める必要がある。筆者は剰余列算法と displacement を基とする高速 QR 法 [Zhi03] の関係や他の算法の関係を調べることで精度を保証しながら剰余列算法を改良できることを確認している。

除算を基とする拡張 Euclid 構成の近似化した算法は、a) 主係数の特異性を回避、b) EZ-GCD 法より安定している、という 2 点では評価できる。しかし、余因子も補間する必要があり次数が高くなると計算に時間がかかり、誤差も膨らむなどの弊害が起きることが容易に想像がつく。しかし、全次数をあげることにより多変数多項式の Hensel 因子を作成できるので、拡張 Euclid 構成単独で近似 GCD を計算するより EZ-GCD 法と融合させるなどの工夫が考えられる。以上の点を今後の課題としたい。

謝　辞

本研究を遂行するにあたり、多岐にわたってアドバイスをして頂いた筑波大学 佐々木建昭教授に感謝致します。

参 考 文 献

[BB07] D. Bini and P. Boito, *Structured matrix-based methods for polynomial ϵ-gcd: analysis and comparisons*, Proc. of ISSAC'07, ACM Press, 2007, 9–16.

[CGTW95] R. Corless, P. Gianni, B. Trager and S. Watt, *The singular value decomposition for polynomial systems*, Proc. of ISSAC'95, ACM Press, 1995, 195–207.

[CWZ04] R. Corless, S. Watt and L. Zhi, *QR factoring to compute the GCD of univariate approximate polynomials*, IEEE Trans. Signal Proces., **52(12)** (2004), 3394–3402.

[EGL97] I. Emiris, A. Galligo and H. Lombardi, *Certified approximate univariate GCDs*, J. Pure and Applied Alge., **117&118** (1997), 229–251.

[GKMYZ04] S. Gao, E. Kaltofen, J. P. May, Z. Yang and L. Zhi, *Approximate factorization of multivariate polynomials via differential equations*, Proc. of ISSAC'04, ACM Press, 2004, 167–174.

[HS97] V. Hribernig and H. J. Stetter, *Detection and validation of clusters of polynomial zeros* , J. Symb. Comput., **24**(1997), 667–681.

[KYZ05] E. Kaltofen, Z. Yang and L. Zhi, *Structured low rank approximation of a Sylvester matrix*, International Workshop on Symbolic-Numeric Computation 2005 (SNC 2005), D. Wang & L. Zhi (Eds.), 2005, 188–201; full paper appear in Symbolic-Numeric Computation (Trends in Mathematics), D. Wang & L. Zhi (Eds.), Birkhäuser Verlag, 2007, 69–83.

[KYZ06] E. Kaltofen, Z. Yang and L. Zhi, *Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials*, Proc. of ISSAC'06, ACM, 2006, 169–176.

[LYZ05] B. Li, Z. Yang and L. Zhi, *Fast low rank approximation of a Sylvester matrix by structure total least norm*, J. JSSAC (Japan Society for Symbolic and Algebraic Computation), **11 (3&4)** 2005, 165–174.

[長坂 08] 長坂 耕作, *Ruppert* 行列による近似 *GCD* の算出, 本予稿集, 2008.

[ONS91] M. Ochi, M-T. Noda and T. Sasaki, *Approximate greatest common divisor of multivariate polynomials and its application to ill-conditioned systems of algebraic equations*, J. Inform. Proces., **14** (1991), 292–300.

[OST97] H. Ohsako, H. Sugiura and T. Torii, *A stable extended algorithm for generating polynomial remainder sequence (in Japanese)*, Trans. of JSIAM (Japan Society for Indus. Appl. Math.) **7** (1997), 227–255.

[Pan01] V.Pan, *Univariate polynomials: nearly optimal algorithms for factorization and rootfinding*, Proc. of ISSAC'01, ACM Press, 2001, 253–267.

[San05] M. Sanuki, *Computing approximate GCD of multivariate polynomials (Extended abstract)*, International Workshop on Symbolic-Numeric Computation 2005 (SNC 2005), D. Wang & L. Zhi (Eds.), 2005, 308–314; full paper appear in Symbolic-Numeric Computation (Trends in Mathematics), D. Wang & L. Zhi (Eds.), Birkhäuser Verlag, 2007, 55–68.

[Sas93] T. Sasaki, *A study of approximate polynomials, I — representation and arithmetic —*, Japan J. Indust. Appl. Math. **12** (1995), 137–161.

[Sch85] A. Schönhage, *Quasi-GCD*, J. Complexity, **1**, 1985, 118–147.

[Suz93] M. Suzuki, *Improvements of the power-series coefficient polynomial remainder sequence GCD algorithm*, Japan J. Indust. Appl. Math. **10(1)** (1993), 41–67.

[SN89] T. Sasaki and M-T. Noda, *Approximate square-free decomposition and root-finding of ill-conditioned algebraic equations*, J. Inform. Proces., **12** (1989), 159–168.

[SS92] T. Sasaki and M. Suzuki, *Three new algorithms for multivariate polynomial GCD*, J. Symbolic Comput., **13**(1992), 395–411.

[SS07] M. Sanuki and T. Sasaki, *Computing approximate GCDs in ill-conditioned cases*, Proc. of Symbolic-Numeric Computation 2007 (SNC 2007), J. Verschelde & S. M. Watt (Eds.), 2007, 170–179, London, Ontario, Canada, 25-27 July, 2007.

[SY98] T. Sasaki and S. Yamaguchi, *An analysis of cancellation error in multivariate Hensel construction with floating-point number arithmetic*, Proc. of ISSAC'98, ACM Press, 1998, 1–8.

[Zen04] Z. Zeng, *The approximate GCD of inexact polynomials part I: a univariate algorithm*, to appear, 2004.

[Zhi03] L. Zhi, *Displacement structure in computing the approximate GCD of univariate polynomials*, Proc. of ASCM2003, World Scientific, 2003, 288–298.

[ZD04] Z. Zeng and B. H. Dayton, *The approximate GCD of inexact polynomials part II: A multivariate algorithm*, Proc. of ISSAC'04, ACM Press, 2004, 320–327.

[ZMF00] C. J. Zarowski, X. Ma and F. W. Fairman, *QR-factorization method for computing the greatest common divisor of polynomials with inexact coefficients*, IEEE Trans. Signal Proces., **48(11)** (2000), 3042–3051.

[ZN00] L. Zhi and M-T. Noda, *Approximate GCD of multivariate polynomials*, Proc. of ASCM2000, World Scientific, 2000, 9–18.