特異値分解とリフティング法に基づく多変数近似GCD計算とその最適化

Multivariate Approximate GCD Computation based on Singular Value Decomposition and Its Optimization

筑波大学医学医療系　　讃岐　勝 [*1]
Masaru Sanuki
Department of Clinical Medicine, Faculty of Medicine, University of Tsukuba

Abstract

Computing the multivariate approximate GCD, we consider to compute null space of $(k-1)$-th subresultant matrix within polynomials, where k is the degree of for multivariaapproximate GCD. For computing the null space, one is used the singular value decomposition and lifting technique, the other is used iterative method of matrix within polynomials. As the later, we show effective techiniques for matrix compuation within polynomials.

1　はじめに

浮動小数 $\mathbb{F}$ を係数に持つ多変数多項式全体を $\mathbb{F}[x, t_1, \ldots, t_\ell] = \mathbb{F}[x, \boldsymbol{t}]$ で表す．入力多項式 $F(x, \boldsymbol{t})$ と $G(x, \boldsymbol{t})$ は主変数 x に関する次数 k の近似 GCD を持つと仮定する．言い換えると，次を満たす次数 k の多項式 $C(x, \boldsymbol{t}) \in \mathbb{F}[x, \boldsymbol{t}]$ が存在する．

$$F(x, \boldsymbol{t}) = C(x, \boldsymbol{t})\tilde{F}(x, \boldsymbol{t}) + \Delta_F(x, \boldsymbol{t}), \quad G(x, \boldsymbol{t}) = C(x, \boldsymbol{t})\tilde{G}(x, \boldsymbol{t}) + \Delta_G(x, \boldsymbol{t}). \tag{1}$$

ここで，係数の大きさについて Δ_F と Δ_G は F と G に比べて十分小さい．

1 変数多項式および 2 変数多項式について，計算アルゴリズム開発に関する研究が数多く存在するが，3 変数以上の多項式の場合にはほとんど存在しない．2 変数多項式の場合には 1 変数多項式の場合に帰着できることが多いからである．このことから，純粋に多変数多項式の近似 GCD の計算アルゴリズムに関する研究はあまりなされていなことがわかる．本稿では 3 変数以上の多変数多項式の近似 GCD 計算について述べる．

最初に多変数多項式の近似 GCD の計算の歴史について少し触れる．Euclid の互除法 [9] から始まり，Hensel 構成および補間法と数式処理でよく知られた算法の適応が試みられたがたびたび不安定であった [18]．その後，多変数多項式の数係数を並べた一般化された部分終結式行列の特異値分解による方法が提案されたが [7, 17]，問題のサイズ (多項式の次数) に比べて行列サイズが非常に大きくなってしまい効率面で課題が残った．その後，数式処理でよく知られた算法の安定化のため，誤差伝搬の原因の 1 つである中間

[*1] 〒 305-8575 茨城県つくば市天王台 1-1-1　E-mail: sanuki@md.tsukuba.ac.jp

式膨張の回避をべき級数演算で行ったが高次に対しては適応が難しかった [12]．加えて，有効浮動小数を利用した完全誤差項の消去 [8] および多項式要素の Sylvester 行列への QR 法の適応によって一定の成果が出たが [11]，さらなる精度の向上のためには抜本的な算法の改良が必要なことを示す結果となった．

以上の状況のもと，筆者は精度の核となる部分は数値計算で行い，精度が落ちないことを保証できる場所は数式処理をベースとしたハイブリッドな方法を提案している．1 つは Bezout 行列と GCD の関係性を利用して Barnett の方法 [1, 2, 5] を多変数多項式に拡張を行った方法であり，full-rank な行列を作成して線型方程式系をリフティング法を用いて解くものである [13]．もう 1 つは本稿で紹介する正則でない行列の線型方程式系を解く方法であり，特異値分解 (null 空間の計算) とリフティング法による方法である [15]．

特異値分解 (null 空間の計算) とリフティング法による方法は，多項式を要素にもつ Sylvester 行列を数値要素の Sylvester 行列に射影させて特異値分解を行い，そこからリフティング法で特異値分解で得られた null 空間を復元する．本稿では数値要素に射影できない場合 (特異な場合) も含めて議論を行う．

2 章では，本稿で使用する定義およびその性質について述べる．3 章では，特異値分解とリフティング法を利用した多項式要素にする部分終結式行列の null 空間による近似 GCD 計算法について述べる．4 章では，5 章で利用する多項式を要素とする行列の算法について述べる．5 章では，3 章で述べる方法では計算できない場合について，拡張 Hensel 構成に基づく新たな計算法を述べる．

2　準備

本章では，全次数変数 T の導入と部分終結式とその性質について述べる．

2.1　全次数変数による重み付け

係数部を統一的に扱うため，全次数による重み付けを入力多項式に対して次に変換によって行う．

$$F(x,t_1,\ldots,t_\ell)\to\mathcal{F}(x,T,\boldsymbol{t})=F(x,Tt_1,\ldots,Tt_\ell),\quad G(x,t_1,\ldots,t_\ell)\to\mathcal{G}(x,T,\boldsymbol{t})=G(x,Tt_1,\ldots,Tt_\ell).\ (2)$$

2.2　部分終結式行列と近似 GCD

$(k-1)$ 次部分終結式行列 $\mathcal{S}_{k-1}(F,G)=\mathcal{S}_{k-1}\in\mathbb{F}[\boldsymbol{t},T]^{K\times K}$ は次で定義される行列である．

$$\mathcal{S}_{k-1}\ =\ \begin{pmatrix} \overbrace{\begin{matrix} f_m & & \\ f_{m-1} & f_m & \\ \vdots & \ddots & \ddots \end{matrix}}^{n-k+1} & \overbrace{\begin{matrix} g_n & & \\ g_{n-1} & g_n & \\ \vdots & \ddots & \ddots \end{matrix}}^{m-k+1} \end{pmatrix}=\delta\mathcal{S}_{k-1}^{(0)}+T\cdot\delta\mathcal{S}_{k-1}^{(1)}+\ldots+T^w\cdot\delta\mathcal{S}_{k-1}^{(w)}+\ldots$$

ここで，$K=m+n-2k+2$ かつ $\delta\mathcal{S}_{k-1}^{(i)}\in\mathbb{F}[\boldsymbol{t}]^{K\times K}$ である $(i\geq 0)$．

2.3　多項式に関する定義

定義 1 (多項式が特異)
多項式 $F(x,\boldsymbol{t})\in\mathbb{F}[x,\boldsymbol{u}]$ が特異であるとは，F の主係数について $f_m(\mathbf{0})=0$ または $F(x,\mathbf{0})=0$ を満たすことをいう．∎

3　特異値分解とリフティング法による構成

[15] では，リフティング法による方法を提案した．本章では，$\mathcal{S}_{k-1}^{(0)}$ と $\mathcal{S}_{k-1}$ において情報が落ちていないことを仮定する．すなわち，入力多項式 $F(x,\boldsymbol{t})$ と $G(x,\boldsymbol{t})$ は特異でないと仮定する．

3.1　1 変数多項式：特異値分解を基にした null 空間

数値行列 $\mathcal{S}_{k-1}^{(0)}$ の特異値分解とは次の分解である．

$$\mathcal{S}_{k-1}^{(0)} = U\Sigma V^T = \begin{pmatrix} \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_K \end{pmatrix} \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_K \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1^T \\ \vdots \\ \boldsymbol{v}_K^T \end{pmatrix}.$$

ここで，U および V はそれぞれ直交行列であり $\mathrm{span}_{\mathbb{F}}(\boldsymbol{u}_1,\ldots,\boldsymbol{u}_K) = \mathrm{span}_{\mathbb{F}}(\boldsymbol{v}_1,\ldots,\boldsymbol{v}_K) = \mathbb{F}^K$，また，$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{K-1} \geq \sigma_K \geq 0$ は特異値という．GCD の次数が k のとき，$\sigma_{K-1} \ggg \sigma_K \approx \epsilon$ であることが知られている．(ϵ は近似 GCD の許容度に依存する量である)．

$\mathcal{S}_{k-1}^{(0)}\boldsymbol{z}^{(0)} = \boldsymbol{0}$ の null 空間は，$\boldsymbol{v}_K \in \mathrm{Ker}(\mathcal{S}_{k-1}^{(0)})$ であり $\boldsymbol{z}^{(0)} = \boldsymbol{v}_K$ が解の一つである [4]．また，解は $\tilde{G}^{(0)}$ と $-\tilde{F}^{(0)}$ の係数が順に並んだ係数ベクトルとなる．

3.2　多変数多項式：リフティング法による近似 GCD の構成

多変数多項式でも 1 変数多項式と同様の性質が成り立つ．すなわち，

$$\mathcal{S}_{k-1}\boldsymbol{x} = \boldsymbol{0}$$

の解 (null 空間)$\boldsymbol{x} \in \mathbb{F}[\boldsymbol{t}]^K$ の 1 つは $\tilde{G}$ と $-\tilde{F}$ の係数を並べた係数ベクトルになる．1 変数の情報を基に次のように構成できる．

$\boldsymbol{z}^{(w-1)} = \boldsymbol{z}^{(0)} + \sum_{i=1}^{w-1} T^i\delta\boldsymbol{z}^{(i)}$ まで計算できたと仮定するとき，$\boldsymbol{z}^{(w)} = \boldsymbol{z}^{(w-1)} + T^w \cdot \delta\boldsymbol{z}^{(w)}$ について

$$\mathcal{S}_{k-1}\boldsymbol{z}^{(w)} \quad \equiv \quad \boldsymbol{0} \pmod{T^{w+1}}$$

を T^w について整理することで

$$\mathcal{S}_{k-1}^{(0)}\delta\boldsymbol{z}^{(w)} \quad = \quad -\sum_{j=1}^{w} \mathcal{S}_{k-1}^{(j)}\delta\boldsymbol{z}^{(w-j)} = \delta\boldsymbol{p}^{(w)}$$

とかける．ここで，$\delta\boldsymbol{z}^{(w)}$ を $\boldsymbol{v}_i$ による基底変換，$\delta\boldsymbol{p}^{(w)}$ を $\boldsymbol{u}_i$ による基底変換を行う．

$$\delta\boldsymbol{z}^{(w)} = \begin{pmatrix} \delta z_1^{(w)} \\ \vdots \\ \delta z_N^{(w)} \end{pmatrix} = \delta\hat{z}_1^{(w)}\boldsymbol{v}_1 + \ldots + \delta\hat{z}_N^{(w)}\boldsymbol{v}_K, \delta\boldsymbol{p}^{(w)} = \begin{pmatrix} \delta p_1^{(w)} \\ \vdots \\ \delta p_N^{(w)} \end{pmatrix} = \delta\hat{p}_1^{(w)}\boldsymbol{u}_1 + \ldots + \delta\hat{p}_N^{(w)}\boldsymbol{u}_K.$$

このとき，$\delta\boldsymbol{z}^w$ は次でかける．

$$\delta\boldsymbol{z}^{(w)} = \delta\hat{p}_1^{(w)}/\sigma_1\boldsymbol{v}_1 + \ldots + \delta\hat{p}_{K-1}^{(w)}/\sigma_{K-1}\boldsymbol{v}_{K-1} + T \cdot 0 \cdot \boldsymbol{v}_K.$$

ゆえに，

$$\boldsymbol{z}^{(w)} = \boldsymbol{z}^{(w-1)} + \sum_{j=1}^{w} \delta\hat{p}_1^{(w)}/\sigma_1\boldsymbol{v}_1 + \ldots + \sum_{j=1}^{w} \delta\hat{p}_{K-1}^{(w)}/\sigma_{K-1}\boldsymbol{v}_{K-1} + \cdot\mathbb{F}[\boldsymbol{T},\boldsymbol{t}]_{[1,w]} \cdot \boldsymbol{v}_K \tag{3}$$

数値行列の特異値分解から効率的に構成ができる．

残る問題は，$\boldsymbol{v}_K$ だけ自由度があることであり，近似 GCD となりうる係数 $r \in \mathbb{F}[T,\boldsymbol{t}]$ を決定する必要がある．r の決定方法について，簡単な場合を次節で紹介する．

3.3 効率化のアイデア

実際の例をみながら説明をする．

GCD $C(x,t_1,t_2) = x^3 + (1 + t_2 - 2t_1 + {t_1}^2)x + 3$ をもつ次の多項式を考える．

$$\begin{aligned} F(x,t_1,t_2) &= (x^3 + ({t_2}^2 + t_1 + t_2 - 2)x^2 - 1) \times C(x,t_1,t_2), \\ G(x,t_1,t_2) &= (x^3 + (2{t_2}^2 - t_1 + 3)x^2 - 1) \times C(x,t_1,t_2). \end{aligned}$$

特異値分解によって得た $\mathcal{S}_{k-1}^{(0)}\boldsymbol{z} = 0$ の解 $\boldsymbol{z} = \boldsymbol{z}^{(0)} = \boldsymbol{v}_K$ は次である．

$$\boldsymbol{v}_K = \begin{pmatrix} -0.242535625036333 \\ -0.727606875108999 \\ -2.24840273230668 \times 10^{-15} \\ 0.242535625036333 \\ \hline 0.242535625036333 \\ -0.485071250072665 \\ -1.32375311946987 \times 10^{-15} \\ -0.242535625036333 \end{pmatrix}.$$

次に，$w = 1$ の場合を計算する．$\boldsymbol{v}_K$ は自由に動くので，

$$\delta\boldsymbol{z}^{(1)} = \delta\hat{p}_1^{(1)}/\sigma_1\boldsymbol{v}_1 + \ldots + \delta\hat{p}_N^{(1)}/\sigma_{K-1}\boldsymbol{v}_{K-1} + r^{(1)}\boldsymbol{v}_K$$

について，$r^{(1)} = 0$ として解の候補を計算すると次を得た．

$$\delta\boldsymbol{z}^{(1)} = \begin{pmatrix} 0.0713340073636269t_1 + 0.0285336029454512t_2 \\ -0.0285336029454498t_1 + 0.0856008088363516t_2 \\ 3.65419500214514 \times 10^{-15}t_1 - 4.96824803519758 \times 10^{-15}t_2 \\ -0.0713340073636232t_1 - 0.0285336029454580t_2 \\ \hline -0.0713340073636273t_1 - 0.0285336029454504t_2 \\ -0.0998676103090790t_1 - 0.185468419145430t_2 \\ 4.77577504452709 \times 10^{-16}t_1 - 1.10469359354548 \times 10^{-15}t_2 \\ 0.0713340073636299t_1 + 0.0285336029454456t_2 \end{pmatrix}.$$

$r^{(1)}$ を実際に決定することによって近似 GCD の余因子を決定することが可能となる．いくつか決定の方法がある．本稿ではアイデアを紹介する．

1. モニックであることを仮定

   本例の場合，第 1,5 要素について $w \geq 1$ のときはすべて 0 である．そのため，すぐに適切な $r^{(1)}$ が決定できる．

2. どこかの係数を 0 と仮定

   $\delta \boldsymbol{z}^{(1)}$ の要素は $\tilde{G}$ と $-\tilde{F}$ の係数であることに注目する．多くの場合，係数がすべて 0 でないことがある．そのような場合，ある要素を 0 にすると他の要素も 0 になりそれは解となる可能性が高い．上記の例において，$\delta \boldsymbol{z}^{(1)} + \delta r_1 \boldsymbol{v}_k$ の第 1 要素を 0 にするよう δr_1 を決めるとき，第 3,4,5,8 要素も同時に 0 になる．これは期待している多項式の余因子である (注意 1 を参照)．

3. 一般の場合

   上の 2 つにに合致しない場合，高次の項を見ないで $\boldsymbol{r}^{(1)}$ を決めることは難しい．そのため，文字のまま計算を行い最後に試し割りおよび係数比較を行うことで係数を決定する．手間はかかるが，Hensel 構成のように主係数同士の GCD を再帰的に計算するようなステップをしなくてもよい．

注意 1 (疎な多項式の因子は疎な多項式か？)
係数を 0 にするアプローチは「疎な多項式の因子は疎な多項式」であることを前提にしているがこれは正しいのだろうか？ *[3, 6]* において，どのような場合に因子も疎になるかについて研究がなされている *(*逆にいうと，円分多項式のように密になりそうな場合が事前に予想できる*)*．それゆえ，アルゴリズムの中に因子が疎であるか否かチェックする仕組みは効率化の面で有効と考えられる．

4 多項式を要素とする行列の演算

本章では，多項式を要素とした行列の計算ついて，4.1 節では線型方程式を反復法で解く方法を紹介する [14]．これは null 空間の計算に応用できる．4.2 節では LU 分解であり数式処理の立場から何が起きるのか，実際の計算例で示す．

4.1 反復法

本説では次の線形連立方程式の解法を考える．

$$\mathcal{M}\boldsymbol{x} = \boldsymbol{b} \text{ where } A \in \mathbb{F}[T, \boldsymbol{t}]^{n\times n}, \boldsymbol{b} \in \mathbb{F}[T, \boldsymbol{t}]^{n}. \tag{4}$$

ここで，T は全次数変数であり変数 $\boldsymbol{t}$ で決まる（これまでと用法は一緒である)．

数値要素

行列 $\mathcal{M}$ を $\mathcal{M} = \mathcal{L} + \mathcal{D} + \mathcal{U}$ と下三角，対角，上三角成分の和で分解するとき，Gasuu-Seidel 法および Jacobi 法は次の反復式によって実行される．

- Gasuu-Seidel 法：$(\mathcal{L} + \mathcal{D})\boldsymbol{x}^{(t+1)} = \boldsymbol{b}_i - \mathcal{U}\}\boldsymbol{x}^{(t)}$.
- Jacobi 法：$\mathcal{D}\boldsymbol{x}^{(t+1)} = \boldsymbol{b}_i - (\mathcal{L} + \mathcal{U})\boldsymbol{x}^{(t)}$.

そして，$||\boldsymbol{x}^{(t+1)} - \boldsymbol{x}^{(t)}||$ が十分小さくなったとき反復を終了する．ここで 2 つの算法が収束するためには行に関して優対角行列であることが必要である：各 i に対して，

$$|m_{i,i}| > \sum_{j\neq i} |m_{i,j}|. \tag{5}$$

多項式を要素とする場合：直接法のアイデアを利用

各 w に対して，式 (4) を利用する方法を考える．

直接法を用いた方法のキーは計算済みの数値行列 $\mathcal{M}^{(0)}$ の逆行列を再利用することである [13]．一方，Jacobi 法・Gauss-Seidel 法または Krylov 部分空間法による方法は逆行列そのものでなく $\mathcal{M}^{(0)}\boldsymbol{r}$ を小さくするまたは $(\mathcal{M}^{(0)})^i\boldsymbol{r}$ で張ることで収束性を保証する方法である（ここで，$\boldsymbol{r}$ は残差を表す）．しかし，各全次数 w に対して $\boldsymbol{r}$ は変化するので全次数 w を求める際に $w-1$ までの結果を利用することはできず，w 回だけ反復法を繰り返すことになる．

(4) の代わりに $c_1\boldsymbol{m}_1 + \sum_{j=2}^{n}(T^w x_j)\boldsymbol{m}_j$ を考えると，次のように変形できる．

$$x_1\boldsymbol{m}_1 + \sum_{j=2}^{n}(u^w x_j)\boldsymbol{m}_j = T^w\boldsymbol{b} + (1-T^w)x_1\boldsymbol{m}_1. \tag{6}$$

- 全次数 1 のとき（$w=1$）
  式 (6) の第 1 要素について全次数 1 の斉次式のみを集めることで，k に関する反復式を得る．

$$\delta x_1^{(1,k+1)}m_{11}^{(0)} + x_1^{(0)}\delta m_{11}^{(1)} + \sum_{j\neq i} T x_j^{(0)}m_{1,j}^{(0)} = \delta b_1^{(0)} + \delta x_1^{(1,k)}m_{11}^{(0)} + x_1^{(0)} + x_j^{(0)}m_{1,j}^{(0)}. \tag{7}$$

  ここで，多項式 $x^{(w,t)} \in \mathbb{F}[T, \boldsymbol{t}]$ は k 回目の反復で得られる全次数 w の斉次多項式である．

- 全次数 1 のとき（$w>1$）
  式 (6) の第 1 要素について全次数 w の斉次式のみを集めることで，k に関する次の反復式を得る．

$$\delta x_1^{(w,k+1)}m_{1,1}^{(0)} + \sum_{j\neq i} T^w x_j^{(0)}m_{1,j}^{(0)} = \delta b^{(0)} + \delta x_1^{(w,k)}m_{1,1}^{(0)} + \sum_{j=0}^{w}\delta x_1^{(j)}\delta m_{1,1}^{(w-j)} + c_j^{(0)}a_{1,j}^{(0)}. \tag{8}$$

命題 2
収束するための十分条件は

$$|m_{1,i}| > \sum_{j\neq i}|m_{1,j}| \text{ and } |m_{1,i}| > \sum_{j\neq i}||\delta m_{1,j}^{(w)}||. \tag{9}$$

ここで，f は多項式 $f=\sum_{i=0}^{n}=f_i x^i$ のノルムであり，数値の場合は絶対値に一致する．$f=\max\{||f_n||,\ldots,||f||\}$．

証明 反復式より，

$$m_{1,1}^{(0)}(\delta x_1^{(w,k+1)} - \delta x_1^{(w,k)}) = (1-T^w)\sum_{j\neq i} x_j^{(0)}m_{1,j}^{(0)} + \sum_{j=0}^{w}\delta x_1^{(j)}\delta x_{1,1}^{(w-j)}.$$

上式が収束するためには，条件式 *(9)* を満たす必要がある． ∎

4.1.1 数値実験

反復法を Maple15 で実装し，CPU Intel core-i5 (2.6GHz)，メモリ 8GB の Windows 7 (64bit) の PC 上で実験を行った．

4.1 において，数値要素と異なるのは右辺の要素が多項式であるかだけである．**4.1.1** では 2 つの場合における反復回数について考察を行う．

反復回数の比較

$\mathcal{M}^{(0)}\boldsymbol{x}=\boldsymbol{b}$ について，$\mathcal{M}^{(0)}\in\mathbb{F}^{n\times n}$ を優対角要素，$\boldsymbol{b}$ を数値要素・多項式要素とした場合の収束までの反復回数・計算時間について比較を行う．収束条件は $||\boldsymbol{x}^{(t+1)}-\boldsymbol{x}^{(t)}||<0.0001$ とした．

次の表は $\boldsymbol{b}\in\mathbb{F}[u,v]$，各要素の項数は最大 5，全次数 5 の場合についての実験である．

サイズ	数値要素		多項式要素	
5×5	32	0.001	16	0.005
10×10	33	0.008	15	0.046
100×100	30	0.546	5	0.625

全次数 5 の場合

多項式要素場合の方が反復回数が少なくな傾向があった．これは $\boldsymbol{b}$ の各要素の項数が最大 5 であることが起因する．2 変数・全次数 5 の場合 $\{u^5,u^4vu^3v^2,u^2v^3,uv^4,v^5\}$ の 6 項であり，$\mathcal{M}^{(0)}\boldsymbol{x}_1=u^5\tilde{b}_1$, $\mathcal{M}^{(0)}\boldsymbol{x}_2=u^4v\tilde{b}_2$, …, $\mathcal{M}^{(0)}\boldsymbol{x}_6=v^5\tilde{b}_6$ をそれぞれ解いていると考えることができる ($\tilde{b}_i$ は数値ベクトル)．項数を限定していることから，$\tilde{b}_i$ の要素には 0 が多く，解として 0 が多く入る．そのため収束回数が少ない．ただし，1 つ 1 つの演算は多項式の加算なので計算時間はかかっている．

各項について，それぞれ計算すればさらに反復回数が減らせることが期待できる．反復法は null 空間を計算しているに他ならいないため，この例は一般の線型方程式および null 空間の計算法を示している．

4.2 LU 分解

ここでは，多項式を要素とする行列の LU 分解について実際に解く際に何がおきるかコメントする．

2 つの行列 $\mathcal{M}_1$ および $\mathcal{M}_2$ は, それぞれ正則および特異という条件を持つ（3 列目のみが異なる）．次はそれぞれの行列について LU 分解をした結果を示している．

$$
\begin{aligned}
\mathcal{M}_1 &= \begin{pmatrix} u^2 & uv & uw \\ v^2 & vw & uv \\ w^2 & wu & wv \end{pmatrix} \\
&= \begin{pmatrix} 1 & & \\ \frac{v^2}{u^2} & 1 & \\ \frac{w^2}{u^2} & \frac{w(u^2-vw)}{v(uw-v^2)} & 1 \end{pmatrix} \begin{pmatrix} u^2 & uv & uw \\ 0 & \frac{v(uw-v^2)}{u} & \frac{v(u^2-vw)}{u} \\ 0 & 0 & -\frac{w(u^3-3uvw+v^3+w^3)}{uw-v^2} \end{pmatrix} \\
\mathcal{M}_2 &= \begin{pmatrix} u^2 & uv & uw \\ v^2 & vw & uv \\ uw & wv & w^2 \end{pmatrix} \\
&= \begin{pmatrix} 1 & & \\ \frac{v^2}{u^2} & 1 & \\ \frac{w}{u} & 0 & 1 \end{pmatrix} \begin{pmatrix} u^2 & uv & uw \\ 0 & \frac{v(uw-v^2)}{u} & \frac{v(u^2-vw)}{u} \\ 0 & 0 & 0 \end{pmatrix}
\end{aligned}
$$

2 つの結果について，大差がないように思うが特異な行列 $\mathcal{M}_2$ の場合は複雑になっていないが，正則な行列 $\mathcal{M}_1$ の場合は上三角行列の最後が複雑になっている．これは Euclid の互除法を思い浮かべてもらうとわかりがよく，互いに素な場合に結果が複雑になることと一緒の現象である[1)]．

[1)] 行列の場合，Smith の標準形が LU 分解の操作に一番近く，この場合も結果は複雑になることが知られており，計算を困難にする要因となっている

5 Newton 多角形を利用した全次数変数の導入

多項式が特異な場合には，拡張 Hensel 構成 [16] で利用される Newton 多角形を利用した重み付けを利用する．この重み付けは主係数が特異，多項式そのものが特異なときに利用される．次の手順で重み付けは行われる．

アルゴリズム 1 (Newton 多角形による重み付け)

① $H = \sum_i h_i {x^{e_x^{(i)}}} {u_1}^{e_{u_1}^{(i)}} \cdots {u_\ell}^{e_{u_\ell}^{(i)}}$ と表す時，各項 ${x^{e_x^{(i)}}} {u_1}^{e_{u_1}^{(i)}} \cdots {u_\ell}^{e_{u_\ell}^{(i)}}$ の指数部なる点 $(e_x^{(i)}, e_{u_1}^{(i)} + \cdots + e_{u_\ell}^{(i)})$ を平面上にプロットし，この点集合からなる凸包 *(Newton* 多角形*)* を構成する．

② *Newton* 多角形の下包において，各辺 $\mathcal{L}_1, \ldots, \mathcal{L}_d$ を *Newton* 線と呼ぶ．任意に選んだ *Newton* 線 $\mathcal{L}_i$ 上の点に対応する多項式の和を $H^{(0)}$ とおき，この多項式を *Newton* 線 $\mathcal{L}_i$ に対する *Newton* 多項式 $N_{\mathcal{L}_i}$ と呼ぶ（通常，下包の最右点を含む *Newton* 線 $N_{\mathcal{L}}$ を選ぶ）．

③ F と G について，$\mathcal{L}$ 上の点に対応する多項式 $N_{\mathcal{L}}(F)$ と $N_{\mathcal{L}}(G)$ の *GCD* を $C^{(0)}$，$D^{(0)} = \mathrm{quo}(H^{(0)}, C^{(0)})$ とおく．ここで，$\gcd(C^{(0)}, D^{(0)}) = 1$ & $\deg_x(C) = \deg_x(C^{(0)})$ を満たす必要がある（満たさない場合の対応策は *[10]* で解説）．

上の条件を満たすとき，次の変換によって全次数変数 T を導入する．

$$F(x, \boldsymbol{u}) = \frac{F(tx, T^{w_1}u_1, \ldots, T^{w_\ell}u_\ell)}{T^{n-\lambda d}}$$

ここで，w_i は各変数につけた重みで通常 $w_i = 1$，$\lambda = d/n$ は *Newton* 線の傾きで d, n は互いに素な数である．

このとき，Sylvester 行列は次のように分解される．

$$\mathcal{S}_{k-1} = \mathcal{P}_{k-1}^{(0)} + T \cdot \delta\mathcal{P}_{k-1}^{(1)} + T \cdot \delta\mathcal{P}_{k-1}^{(2)} + \ldots T \cdot \delta\mathcal{P}_{k-1}^{(w)} + \ldots \tag{10}$$

ここで，$\delta\mathcal{P}_{k-1}^{(i)} \in \mathbb{F}[x, \boldsymbol{t}]^{K\times K}$ である $(i \geq 0)$．重みを形式的につけたので，先に紹介した方法を適応することができる．異なる点は次の 2 点である．

1. $\mathcal{P}_{k-1}^{(0)} \in \mathbb{F}[x, \boldsymbol{t}]^{K\times K} \backslash \mathbb{F}^{K\times K}$ なので，特異値分解が計算できない (解決法は次を参照)．
2. （必須ではないが効率面で）$\mathcal{P}_{k-1}^{(0)} \in \mathbb{F}[x, \boldsymbol{t}]^{K\times K} \backslash \mathbb{F}^{K\times K}$ なので，三角化が困難．リフティングの際に何度も積を計算するので簡単な方がよい（計算法は 4.2 節を参照）

アイデア（洗練されていない）

$\mathcal{P}_{k-1}^{(0)} \in \mathbb{F}[x, \boldsymbol{t}]^{K\times K} \backslash \mathbb{F}^{K\times K}$ なので，特異値分解が計算できないケースについて，算法をよくみると null 空間が求められればよいことがわかる．rank は 1 しか落ちていないことを仮定しているので，何か 1 つ見つかればそれを利用することができる．4.1 節において，算法はすでに紹介済みであり計算できることがわかる．一般に $\mathcal{P}_{k-1}^{(0)}$ は複雑でない行列を扱うことが多いので[2)]，計算することは困難ではない．

6 まとめ

本稿では，多項式を要素とする Sylvester 行列の null 空間を利用した近似 GCD 計算のアイデアについて紹介した．算法が適応できないケースについても，拡張 Hensel 構成のアイデアを利用して計算できることを示したが，一般的な計算例を示す必要がある．これを次の課題としたい．

[2)] 疎な多項式の場合に起きる

謝　辞

講演について，有用なコメントをいただくことができました．感謝いたします．

参 考 文 献

[1] S. Barnett. *Greatest common divisor of two polynomials*. Linear Algebra Appl., **3**, 1970, 7–9.

[2] S. Barnett. *Greatest common divisor of several polynomials*. Proc. Camb. Phil. Soc., **70**, 1971, 263–268.

[3] V. Bhargava, S. Saraf and I. Volkovich, *Deterministic Factorization of Sparse Polynomials with Bounded Individual Degree*, 2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS), 485–496, 2018.

[4] 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.

[5] G. M. Diaz-Toca and L. Gonzalez-Vega. *Barnett's theorems about the greatest common divisor of several univariate polynomials though Bezout-like matrices. J. Symb. Compu.*, **34**, (2002), 59–81.

[6] J. zur Gathen and E. Kaltofen, *Factoring sparse multivariate polynomials*, J. of Comput. Syst. **31(2)**, 1985, 265–287.

[7] 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.

[8] F. Kako and T. Sasaki. Proposal of "effective floating-point number" for approximate algebraic computation. *Preprint of Tsukuba Univ.*, 1997.

[9] 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.

[10] M. Sanuki, D. Inaba and T. Sasaki: Computation of GCD of sparse multivariate polynomials by extended Hensel construction, *Proc. of SYNASC 2015*, IEEE, 2015, 34–41.

[11] M. Sanuki and T. Sasaki, *Computing approximate GCDs in ill-conditioned cases*, Proc. of Symbolic-Numeric Computation 2007 (SNC 2007), 2007, 170–179.

[12] 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.

[13] M. Sanuki. *Computing multivariate approximate GCD based on Barnett's theorem*, Proc. of Symbolic-Numeric Computation 2009 (SNC 2009), 2009, 149–157.

[14] 讃岐勝．*Jacobi* 法を基にした多項式要素の線形方程式の解法，第 42 回数値解析シンポジウム講演予稿集, 2013, 144–147

[15] 讃岐勝．多変数近似 *GCD* 計算のための *Sylvester* 部分行列の *null* 空間の効率的計算，日本数式処理学会第 28 回大会にて発表, 2019 年 5 月 31 日–6 月 2 日 (学会誌「数式処理」に掲載予定)

[16] T. Sasaki and F. Kako, *Solving multivariate algebraic equation by Hensel construction*, Japan J. Indust. Appl. Math., 16(2), 1999, 257–285.

[17] 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.

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