近似 Groebner 基底の逐次算法に向けて（再訪）

Towards an iterative method for approximate Groebner basis, revisited

神戸大学 大学院 人間発達環境学研究科　　長坂 耕作 [*1]
Kosaku Nagasaka
Graduate School of Human Development and Environment, Kobe University

Abstract

In this talk, we briefly compare the NewtonSLRA/1 algorithm against the Cadzow's algorithm for computing a structured Groebner basis approximately, and discuss on a possibility of using signature-based algorithms for computing approximate Groebner basis.

1　はじめに

本報告では，拙稿 [Nag09, Nag11] で導入した構造化 Gröbner 基底を近似的に求める方法について，近年近似 GCD 向けの改良に取り組んできた NewtonSLRA/1[SS16] の亜種 [Nag21] 等により改善が見込めるかと，Vaccon らの研究 [VVY21, Vac18, VY17] で用いられた signature-based algorithm の多項式環での野呂・横山 [NY21] によるアルゴリズムが適用可能であるかについての現状を述べる。

まず，本報告で扱うのは，佐々木・加古 [SK08] による近似 Groebner 基底のクラス分けにおける第二種の問題であり，**先天的な誤差を持つ不正確な係数を持つ多項式**が与えられ，それを生成系とするイデアルの Gröbner 基底を求める問題である。このため，利用する演算の種類は問わないが，先天的な誤差への対応が不可避となり，理想としては，次のように先天的な誤差をなんとか排除したい。

例 1
次の多項式集合 F の場合，構造的に安定しており，イデアル $\langle F\rangle$ の *Gröbner* 基底 G の計算を行う際の先天的な誤差の影響は少ない。

$$F = \{2.000005x + 3.000001y,\ 0.999999xy - 2.000003\}$$

$$G = \left\{1.0x + 1.5y,\ 1.0y^2 + 1.33334\right\}.$$

ところが，それぞれの *1.2* 倍と *0.5* 倍の差を丸めた要素が追加されていると，次のように自明なイデアルになってしまう。この計算自体に間違いはないが，先天的な誤差により矛盾が生じていると捉えるならば，必要な情報が失われていることになる。

$$F = \{2.000005x + 3.000001y,\ 0.999999xy - 2.000003,\ -0.49995xy + 2.4001x + 3.59999y + 1.00001\}$$

$$G = \{1.0\}.$$

[*1] E-mail: nagasaka@main.h.kobe-u.ac.jp

そのため，何らかの方法で先天的な誤差をなるべく排除して安定化したいというのが，以前に試みていたことである。例えば，次のように多項式集合を多少となり摂動させ，無矛盾な系としての結果が得たい。

$$F = \{2.00005x + 2.99997y,\ 0.999981xy - 2.00000,\ -0.499986xy + 2.40006x + 3.60001y + 1.00001\}$$

$$G = \left\{1.0x + 1.49996y, 1.0y^2 + 1.3334\right\}.$$

◁

1.1 構造化 Gröbner 基底

まずは構造化 Gröbner 基底の定義を振り返るが，その前に項集合 $\mathcal{T}$ に制約した Macaulay 行列 $\mathcal{M}_{\mathcal{T}}(\cdot)$ の記法を導入しておく。これは，Gröbner 基底を計算する F_4 アルゴリズム [Fau99] などで用いられる行列と基本的には同じであり，考慮中の多項式集合に含まれる倍多項式であって，その項が $\mathcal{T}$ に含まれている倍多項式を対象に制限した Macaulay 行列になる。

定義 1 (構造化 Gröbner 基底)
G が以下の条件を満たすとき，G を多項式集合 F に対する，許容度 $\varepsilon \in \mathbb{R}_{\geq 0}$，階数落ち $d \in \mathbb{Z}_{\geq 0}$，項集合 $\mathcal{T}$，写像族 $\mathcal{S}$ の構造化 *Gröbner* 基底 という。

1. G は以下で定義される $F_{st} = \{f_{st,1}, \ldots, f_{st,k}\} \in \mathbb{C}[\vec{x}]$ の生成するイデアルの *Gröbner* 基底である。
2. F と F_{st} は $\mathcal{S}$ による構造化多項式集合である。即ち，写像族 $\mathcal{S}$ に対しパラメータ $\vec{p_i}, \vec{p_{st}}_i \in \mathbb{C}^{n_i}$ が存在して，$f_i(\vec{x}) = \mathcal{S}_i(\vec{p_i})$ と $f_{st,i}(\vec{x}) = \mathcal{S}_i(\vec{p_{st}}_i)$ を満たす。
3. $\|\cdot\|$ をベクトルノルムとして，$\|(\vec{p_1} \ldots \vec{p_k}) - (\vec{p_{st}}_1 \ldots \vec{p_{st}}_k)\| = \varepsilon$ を満たす。
4. $\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F_{st})) = \mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F)) - d$ を満たす。 ◁

なお，実際の計算において，定義おける多項式集合 F と写像族 $\mathcal{S}$ は容易に与えることができるものの，それ以外の許容度 $\varepsilon \in \mathbb{R}_{\geq 0}$，階数落ち $d \in \mathbb{Z}_{\geq 0}$，項集合 $\mathcal{T}$ は容易に決定することはできない。これらを一定の条件下で解決したのが，拙稿 [Nag11] で導入した次のアルゴリズムである。

入力: 項順序 $\succ$, 写像族 $\mathcal{S}$，多項式集合 $F = \{f_1(\vec{x}), \ldots, f_k(\vec{x})\}$.
出力: 構造化 Gröbner 基底 G，許容度 ε，階数落ち d，項集合 $\mathcal{T}$，摂動後の多項式集合 F_{st}，または「失敗」

1. F の Gröbner 基底を計算（近似か厳密）し，項集合 $\mathcal{T}$ を推定（F_4 準拠であり項順序や sugar に依存）。
2. 行列 $\mathcal{M}_{\mathcal{T}}(F) \in \mathbb{C}^{m \times n}$ の特異値 σ_i $(i = 1, \ldots, r_{org})$ を計算。
3. 階数落ち d を推定（「失敗」の場合も）。
4. SLRA を解くアルゴリズムで，$\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F_{st})) = \mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F)) - d$ を満たす構造化多項式集合 F_{st} と許容度 ε を計算。そのような F_{st} が見つかられなければ，「失敗」を出力。
5. F_{st} の近似 Gröbner 基底 G を既知の方法で計算し，$\{G,\ \varepsilon,\ d,\ \mathcal{T},\ F_{st}\}$ を出力。

2 構造化 Gröbner 基底の計算と SLRA

構造化 Gröbner 基底計算の最終段階手前では，多項式集合 F，階数落ち $d \in \mathbb{Z}_{\geq 0}$，項集合 $\mathcal{T}$ などが与えられている状態で，最後の条件である「$\mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F_{st})) = \mathrm{rank}(\mathcal{M}_{\mathcal{T}}(F)) - d$」を満たすようなパラメー

タ $\vec{p_i}, \vec{p_{st\,i}} \in \mathbb{C}^{n_i}$ を求めることになる（そのパラメータの $\mathcal{S}$ による像が F_{st} であり，その生成するイデアルの Gröbner 基底が求めようとしている構造化 Gröbner 基底である）。この計算（パラメータの計算）は，NewtonSLRA/1[SS16] やその亜種 [Nag21] 等が対象としている SLRA と呼ばれる問題である。

定義 2 (SLRA: Structured Low Rank Approximation)
Given a structure specification $\mathcal{S} : \mathbb{R}^{n_\alpha} \to \mathbb{R}^{m \times n}$ *, a parameter vector* $\vec{p} \in \mathbb{R}^{n_\alpha}$*, a vector norm* $\|\cdot\|$*, and an integer* r*,* $0 < r < \min\{m, n\}$*, find a vector* $\vec{p^*}$ *such that*

$$\min_{\vec{p^*}} \|\vec{p} - \vec{p^*}\| \ \ \textit{and}\ \operatorname{rank}(\mathcal{S}(\vec{p^*})) \leq r.$$

このSLRAを解くために拙稿 [Nag11] では，単純な Lift-and-Project 法（直交射影でなく最近行列の要素平均）を用いたが，今回は直交射影による Lift-and-Project 法（Cadzow algorithm）や NewtonSLRA/1[SS16] などで改善が見られるかの実験を行った。なお，近似 GCD 向けに開発した亜種 [Nag21] である Relaxed NewtonSLRA については，NewtonSLRA/1 の時点で後述のとおりに改善が見込めなかった為に実験は行っていない。

2.1 実験内容とその結果の一部

次の多項式集合 F を入力とし，全次数辞書式順序で項集合 $\mathcal{T} = \{x^2y, y^3, x^2, xy, y^2, y, 1\}$ の場合を考える。

$$F = \{0.002 + 1.01x^2 - 2.09y^2,\ 3.06xy + 4.03x^2y,\ 0.504x^2 + 1.504xy + 2.04x^2y - 1.02y^2\}$$

このとき，SLRA の対象となる Macaulay 行列は次の通り（想定される階数は 3 なので，階数落ちは $d = 1$）。

$$\mathcal{M}_{\mathcal{T}}(F) = \begin{pmatrix} 0 & 0 & 1.01 & 0 & -2.09 & 0 & 0.002 \\ 1.01 & -2.09 & 0 & 0 & 0 & 0.002 & 0 \\ 4.03 & 0 & 0 & 3.06 & 0 & 0 & 0 \\ 2.04 & 0 & 0.504 & 1.504 & -1.02 & 0 & 0 \end{pmatrix}$$

構造（写像族 $\mathcal{S}$）が線形の場合，与えられた行列の変動は線形空間を張る（変動後の行列はアフィン空間を動く）。この問題における当該線形空間の正規直交基底の 1 組として次を使用した。なお，この線形空間の内積はフロベニウスノルムを誘導する転置行列との積のトレースを用いている。

$$\left\{ \begin{pmatrix} 0&0&0&0&0 & 0 & 0.707107 \\ 0&0&0&0&0 & 0.707107 & 0 \\ 0&0&0&0&0 & 0 & 0 \\ 0&0&0&0&0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 0&0&0.707107&0&0 \\ 0&0.707107&0&0 & 0 & 0&0 \\ 0 & 0 & 0&0 & 0 & 0&0 \\ 0 & 0 & 0&0 & 0 & 0&0 \end{pmatrix}, \begin{pmatrix} 0 & 0&0.707107&0&0&0&0 \\ 0.707107&0 & 0 & 0&0&0&0 \\ 0 & 0 & 0 & 0&0&0&0 \\ 0 & 0 & 0 & 0&0&0&0 \end{pmatrix}, \right.$$

$$\left. \begin{pmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0 \end{pmatrix}, \begin{pmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \end{pmatrix}, \begin{pmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0 \end{pmatrix}, \begin{pmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0 \end{pmatrix}, \begin{pmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \end{pmatrix}, \begin{pmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0 \end{pmatrix} \right\}$$

Cadzow algorithm や NewtonSLRA の収束先が，以前に用いた Lift-and-Project 法よりも優れているかの実験のため，速度は考慮せずに最大反復回数を 1024 として計算を行った結果が以下の行列である。

$$\text{Cadzow algorithm}: \begin{pmatrix} 0 & 0 & 1.01198 & 0 & -2.08901 & 0 & 0 \\ 1.01198 & -2.08901 & 0 & 0 & 0 & 0 & 0 \\ 4.03864 & 0 & 0 & 3.04856 & 0 & 0 & 0 \\ 2.02274 & 0 & 0.49605 & 1.52686 & -1.02398 & 0 & 0 \end{pmatrix}$$

$$\text{NewtonSLRA/1}: \begin{pmatrix} 0 & 0 & 2.50269 & 0 & -8.79613 & 0 & -5.61233 \\ 2.50269 & -8.79613 & 0 & 0 & 0 & -5.61233 & 0 \\ 5.95215 & 0 & 0 & 3.51882 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\text{NewtonSLRA/2}: \begin{pmatrix} 0 & 0 & 1.10807 & 0 & -1.82471 & 0 & 0.386553 \\ 1.10807 & -1.82471 & 0 & 0 & 0 & 0.386553 & 0 \\ 4.28703 & 0 & 0 & 2.25747 & 0 & 0 & 0 \\ 2.59889 & 0 & 0 & 1.36853 & 0 & 0 & 0 \end{pmatrix}$$

どの結果も階数は落ちているが，NewtonSLRA は収束先が Cadzow algorithm や以前の Lift-and-Project 法に比べて，かなり摂動が大きく改善に用いることは難しい結果となっている。他にも ISSAC2011 で用いた例で計算を行ったものの，同様の挙動を示しており，局所最適解による構造化 Gröbner 基底の計算を，NewtonSLRA やその亜種で改善することは難しいと考えられる。

3　構造化 Gröbner 基底の計算と signature-based algorithm

以前のアルゴリズムでは，最初に項集合 $\mathcal{T}$ を推定して（確定させて），その後に階数落ちの判断や実際の摂動を求める 2 段階（最終の Gröbner 基底計算も入れれば 3 段階）で計算を行っている。このため，最初に行う項集合の計算がうまくいないと破綻する（失敗が返る）。これを改善するため，項集合の計算を進める段階で同時に階数落ちの判断や実際の摂動を求める逐次的な算法を考えたい。逐次的に計算を行うこと自体は以前の方法でも可能であるが，途中の計算の正当性を保証することが F_4 準拠では難しかった。そこで，signature-based algorithm の利用を検討している。なお，ISSAC2011 のアルゴリズムで用いている項集合の決定方法は次のとおりである。

1. $F := F_{init}$, $\mathcal{T} := \{\}$, $\mathcal{T}_F := \{\mathrm{supp}(f_i) \mid f_i \in F_{init}\}$, P:=F のペア集合
2. while $P \neq \phi$
   (a) $\{i,j\} \in P$, $P := P \setminus \{i,j\}$
   (b) $\mathcal{T}_{ij}$:=`TermUpdate`$(F, \mathcal{T}_F, \{i,j\})$　（F_4 の Symbolic-Preprocessing のような処理）
   (c) F_{init} の係数行列 $M_{T_{ij}}$ を $\mathcal{T}_{ij}$ に基づいて構成
   (d) $M_{T_{ij}}$ の QR 分解で誤差を考慮しつつ基底多項式を検出し，結果を $F_{new} := \{\hat{f}_1, \ldots, \hat{f}_r\}$
   (e) $F := F \cup F_{new}$, P の更新, $\mathcal{T}_F := \mathcal{T}_F \cup \{\mathcal{T}_{ij}, \ldots, \mathcal{T}_{ij}\}$
3. $\mathcal{T}$ を出力

3.1 手計算による可能性の考察

実際に次の F を入力として，signature-based algorithm の手順でもって，全次数順序と POT（加群順序）の構造化 Gröbner 基底を求めることを試みた。即ち，signature-based algorithm の手順で逐次的に項集合を確定させながら，SLRA を繰り返し解いていく形になる。なお，下線部は頭項を表している。

$$F=\{f_1=0.002+1.01\underline{x^2}-2.09y^2,\ f_2=3.06xy+4.03\underline{x^2y},\ f_3=0.504x^2+1.504xy+2.04\underline{x^2y}-1.02y^2\}$$

この段階での擬正則なペアやその情報は次の通り。余項の項は，$\mathfrak{S}$ 簡約で消去に用いることが可能な多項式やこの段階の項集合の決定に必要なので掲載している。

擬正則なペア	S 多項式で消去の頭項	guessed signature	余項の項
(f_1,f_2)	$(y\vec{e}_1,\vec{e}_2)$	$\vec{e}_2$	$\{y^3,xy,y\}$
(f_1,f_3)	$(y\vec{e}_1,\vec{e}_3)$	$\vec{e}_3$	$\{y^3,x^2,xy,y^2,y\}$
(f_2,f_3)	$(\vec{e}_2,\vec{e}_3)$	$\vec{e}_3$	$\{x^2,xy,y^2\}$

擬正則なペアの中で最も guessed signature の低い (f_1,f_2) が選択され，その S 多項式とその $\mathfrak{S}$ 簡約での正規形を求めるが，この S 多項式は既に $\mathfrak{S}$ 既約であり，多項式集合や擬正則ペアは次のように更新される。なお，余項の項は完全に構造的な処理をしたいが，今回の考察では QR 分解による方法を採用している（それでも実際の $\mathfrak{S}$ 簡約結果とは多少異なる）。

$$G=\{f_1,f_2,f_3,f_4=spoly(f_1,f_2)\}\ \text{where}\ sig(f_4)=\vec{e}_2$$

擬正則なペア	S 多項式で消去の頭項	guessed signature	余項の項
(f_1,f_4)	$(y^3\vec{e}_1,x^2\vec{e}_2)$	$x^2\vec{e}_2$	$\{y^5,x^3y,x^2y,y^3\}$
(f_2,f_4)	$(y^2\vec{e}_2,x^2\vec{e}_2)$	$x^2\vec{e}_2$	$\{x^3y,xy^3,x^2y\}$
(f_1,f_3)	$(y\vec{e}_1,\vec{e}_3)$	$\vec{e}_3$	$\{y^3,x^2,xy,y^2,y\}$
(f_2,f_3)	$(\vec{e}_2,\vec{e}_3)$	$\vec{e}_3$	$\{x^2,xy,y^2\}$
(f_3,f_4)	$(y^2\vec{e}_3,x^2\vec{e}_2)$	$y^2\vec{e}_3$	$\{x^3y,x^2y^2,xy^3,y^4,x^2y\}$

続いて，(f_1,f_4) の擬正則ペアの計算に移る。このペアの S 多項式は $\mathfrak{S}$ 既約ではないため，余項を消去可能な多項式から項集合を求める。そのため，現時点で $\mathfrak{S}$ 簡約に利用可能な多項式を再掲しておく。

$$LM(f_1)=x^2,\ LM(f_2)=x^2y,\ LM(f_3)=x^2y,\ LM(f_4)=y^3$$
$$sig(f_1)=\vec{e}_1,\ sig(f_2)=\vec{e}_2,\ sig(f_3)=\vec{e}_3,\ sig(f_4)=\vec{e}_2$$

余項が $\{y^5,x^3y,x^2y,y^3\}$ であることや guessed signature に注意して，$\mathfrak{S}$ 簡約に順次用いることが可能な多項式を求めると，次のようになる。なお，f_3 は signature が大きいため利用することはできない。

$$\{y^3f_1,x^2f_4=(x^2yf_1,x^2f_2),y^2f_4=(y^3f_1,y^2f_2),xyf_1,xf_4=(xyf_1,xf_2),yf_1,f_4=(yf_1,f_2)\}$$

ここで丸括弧「$(\cdot,\cdot)$」は，入力の多項式集合 F を用いてどのように生成されるかを表している（逐次的に SLRA を解いていくので，全ての要素は入力の多項式を用いて表現する必要がある）。最終的に整理した結果，次の多項式で $\mathfrak{S}$ 簡約の計算を実現可能なことがわかる。

$$\{f_2,\ xf_2,\ x^2f_2,\ y_1,\ xyf_1,\ x^2yf_1,\ y^2f_2,\ y^3f_1\}$$

その Macaulay 行列（大きさは 8×9）を構成し，その最小特異値 σ_8 を求めた。結果の最小特異値は非常に小さく，誤差のために階数が大きくなっていると判断することにする。即ち，(f_1,f_4) の擬正則ペアは $\mathfrak{S}$

簡約により 0 になったことを意味する。

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 4.03 & 0 & 3.06 & 0. \\ 0 & 0 & 0 & 4.03 & 0 & 3.06 & 0 & 0 & 0. \\ 4.03 & 0 & 0 & 3.06 & 0 & 0 & 0 & 0 & 0. \\ 0 & 0 & 0 & 0 & 0 & 1.01 & -2.09 & 0 & 0.002 \\ 0 & 0 & 0 & 1.01 & -2.09 & 0 & 0. & 0.002 & 0. \\ 1.01 & -2.09 & 0 & 0 & 0 & 0.002 & 0. & 0 & 0. \\ 0 & 4.03 & 0 & 0 & 3.06 & 0 & 0. & 0 & 0 \\ 0 & 1.01 & -2.09 & 0 & 0 & 0 & 0.002 & 0 & 0 \end{pmatrix}, \ \sigma_8 = 1.92835\mathtt{e}{-16}$$

この結果（(f_1, f_4) が 0 に $\mathfrak{S}$ 簡約される）から，擬正則ペアは次のように更新される（signature-based algorithm では，擬正則ペアが 0 に簡約された場合に，guessed signature の倍多項式の guessed signature を持つ擬正則ペアが削除される）。

擬正則なペア	S 多項式で消去の頭項	guessed signature	余項の項
(f_1, f_3)	$(y\vec{e}_1, \vec{e}_3)$	$\vec{e}_3$	$\{y^3, x^2, xy, y^2, y\}$
(f_2, f_3)	$(\vec{e}_2, \vec{e}_3)$	$\vec{e}_3$	$\{x^2, xy, y^2\}$
(f_3, f_4)	$(y^2\vec{e}_3, x^2\vec{e}_2)$	$y^2\vec{e}_3$	$\{x^3y, x^2y^2, xy^3, y^4, x^2y\}$

続いて，(f_1, f_3) の擬正則ペアの計算に移るが，現時点で $\mathfrak{S}$ 簡約に利用可能な多項式を再掲しておく。

$$LM(f_1) = x^2,\ LM(f_2) = x^2y,\ LM(f_3) = x^2y,\ LM(f_4) = y^3$$
$$sig(f_1) = \vec{e}_1,\ sig(f_2) = \vec{e}_2,\ sig(f_3) = \vec{e}_3,\ sig(f_4) = \vec{e}_2,\ x^2\vec{e}_2 \in LM(syz)$$

余項が $\{y^3, x^2, xy, y^2, y\}$ であることや guessed signature に注意して，$\mathfrak{S}$ 簡約に順次用いることが可能な多項式を求めると，次のようになる。

$$\{yf_1,\ f_3,\ f_4 = (yf_1, f_2),\ f_1\} \xrightarrow[\text{整理すると}]{} \{yf_1,\ f_3,\ f_2,\ f_1\}$$

その Macaulay 行列（大きさは 4×7）を構成し，その最小特異値 σ_4 を求めたものが以下である。

$$\begin{pmatrix} 1.01 & -2.09 & 0. & 0 & 0 & 0.002 & 0. \\ 2.04 & 0 & 0.504 & 1.504 & -1.02 & 0. & 0. \\ 4.03 & 0 & 0. & 3.06 & 0 & 0. & 0. \\ 0 & 0 & 1.01 & 0 & -2.09 & 0 & 0.002 \end{pmatrix}, \ \sigma_4 = 0.0293402$$

ところが，この行列には先程の結果（(f_1, f_4) が 0 に $\mathfrak{S}$ 簡約される）が加味されていない。そのため，(f_1, f_4) の際に用いた（未確定の）多項式らと，今回の (f_1, f_3) の際に用いた多項式らが異なり，それぞれが独立してしまう。これは無意味な計算をしていることを意味する。そこで，今回の計算に必要な $\{yf_1,\ f_3,\ f_4 = (yf_1, f_2),\ f_1\}$ に加えて，(f_1, f_4) の計算に必要であった以下の結果を考慮することにする。

$$\{y^3f_1, x^2f_4 = (x^2yf_1, x^2f_2), y^2f_4 = (y^3f_1, y^2f_2), xyf_1, xf_4 = (xyf_1, xf_2), yf_1, f_4 = (yf_1, f_2)\}$$

結果として得られる，整理した $\mathfrak{S}$ 簡約の計算に必要な多項式は次のとおりである。

$$\{f_1,\ f_2,\ f_3,\ xf_2,\ x^2f_2,\ yf_1,\ xyf_1,\ x^2yf_1,\ y^2f_2,\ y^3f_1\}$$

この結果に基づき，その Macaulay 行列（大きさは 10×12）を構成し，その最小特異値等 σ_9, σ_{10} を求めた。σ_{10} は (f_1, f_4) の擬正則ペアが 0 に $\mathfrak{S}$ 簡約されることに対応しており，その値は非常に小さい。一方，σ_9 が今回の (f_1, f_3) に対応しており，0 には簡約されないことがわかる。

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.01 & 0 & -2.09 & 0 & 0.002 \\
0 & 0 & 0 & 0 & 0 & 4.03 & 0 & 0. & 3.06 & 0 & 0. & 0. \\
0 & 0 & 0 & 0 & 0 & 2.04 & 0 & 0.504 & 1.504 & -1.02 & 0. & 0. \\
0 & 0 & 0 & 4.03 & 0 & 3.06 & 0 & 0. & 0 & 0 & 0. & 0. \\
4.03 & 0 & 0 & 3.06 & 0 & 0 & 0 & 0 & 0 & 0 & 0. & 0. \\
0 & 0 & 0 & 0 & 0 & 1.01 & -2.09 & 0. & 0 & 0 & 0.002 & 0. \\
0 & 0 & 0 & 1.01 & -2.09 & 0 & 0. & 0 & 0.002 & 0 & 0. & 0. \\
1.01 & -2.09 & 0 & 0 & 0 & 0.002 & 0. & 0. & 0 & 0 & 0. & 0. \\
0 & 4.03 & 0 & 0 & 3.06 & 0 & 0. & 0. & 0 & 0 & 0 & 0. \\
0 & 1.01 & -2.09 & 0 & 0 & 0 & 0.002 & 0. & 0 & 0 & 0 & 0.
\end{pmatrix},$$

$$\sigma_9 = 0.0253285,\ \sigma_{10} = 7.29561\mathtt{e}\text{-}17$$

この計算では，先程は signature の関係から $\mathfrak{S}$ 簡約に用いることができなかった f_3 が使われるように見えるが，signature-based algorithm の性質上問題とはならない。また，この性質をうまく使うことで，F_4 準拠の構成方法では難しい，階数が落ちることと 0 に簡約されることが必要十分であることを保証可能ではないかと期待している。今後は，これらの考察に基づき，signature-based algorithm に基づく逐次的に構造化 Gröbner 基底を計算するアルゴリズムを確立したいと考えている。

謝　辞

This work was supported by JSPS KAKENHI Grant Number 19K11827.

参 考 文 献

[Fau99] Jean-Charles Faugére. A new efficient algorithm for computing Gröbner bases (F_4). *J. Pure Appl. Algebra*, 139(1-3):61–88, 1999. Effective methods in algebraic geometry (Saint-Malo, 1998).

[Nag09] Kosaku Nagasaka. A study on Gröbner basis with inexact input. In *Proceedings of CASC 2009*, volume 5743 of *Lecture Notes in Comput. Sci.*, pages 247–258. Springer, Berlin, 2009.

[Nag11] Kosaku Nagasaka. Computing a structured Gröbner basis approximately. In *ISSAC 2011—Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, pages 273–280. ACM, New York, 2011.

[Nag21] Kosaku Nagasaka. Relaxed newtonslra for approximate gcd. In *Proceedings of CASC 2021*, volume 12865 of *Lecture Notes in Comput. Sci.*, pages 272–292. Springer, Berlin, 2021.

[NY21] Masayuki Noro and Kazuhiro Yokoyama. Implementation of a signature based algorithm in risa/asir. *RIMS Kôkyûroku*, 2185:139–148, 2021.

[SK08] Tateaki Sasaki and Fujio Kako. Floating-point Gröbner basis computation with ill-conditionedness estimation. In *Proceedings of ASCM 2007*, volume 5081 of *Lecture Notes in Comput. Sci.*, pages 278–292. Springer, Berlin, 2008.

[SS16] Éric Schost and Pierre-Jean Spaenlehauer. A quadratically convergent algorithm for structured low-rank approximation. *Found. Comput. Math.*, 16(2):457–492, 2016.

[Vac18] Tristan Vaccon. Matrix-F5 algorithms and tropical Gröbner bases computation. *J. Symbolic Comput.*, 89:227–254, 2018.

[VVY21] Tristan Vaccon, Thibaut Verron, and Kazuhiro Yokoyama. On affine tropical F5 algorithms. *J. Symbolic Comput.*, 102:132–152, 2021.

[VY17] Tristan Vaccon and Kazuhiro Yokoyama. A tropical F5 algorithm. In *ISSAC'17—Proceedings of the 2017 ACM International Symposium on Symbolic and Algebraic Computation*, pages 429–436. ACM, New York, 2017.