

浮動小数グレブナー基底の悪条件性

佐々木 建昭 (Tateaki Sasaki) *

筑波大学 数学系

INSTITUTE OF MATHEMATICS, UNIVERSITY OF TSUKUBA

甲斐 博 (Hiroshi Kai) †

愛媛大学 理工学研究科

GRADUATE SCHOOL OF SCIENCE AND TECHNOLOGY, EHIME UNIVERSITY

Abstract

浮動小数係数多項式のグレブナー基底計算に対して、Sasaki-Kako は昨年、多倍長有効浮動小数を用いる実際的方法を呈示した。その成果を基に、本稿では浮動小数グレブナー基底に関する幾つかの課題に取り組む。まず、悪条件性を定量化するため条件数 (Condition Number) を定義し、条件数の実際的計算法を提案する。次に、自己簡約による桁落ちの低減化を目指して幾つかの手法を実験的に検討する。最後に、当面の最大の問題点について簡単に述べる。

1 はじめに

浮動小数グレブナー基底とは浮動小数を用いて計算したグレブナー基底であるが、それには2種類ある。第1種は、入力多項式の係数は正確だが、何らかの理由で浮動小数を用いる場合である。第2種は、入力式の係数が不正確なので、実用上、浮動小数を用いる場合である。本稿では第2種を扱う。第1種の基底は Shirayanagi & Sweedler [15, 16, 17] により研究された。第2種は、Stetter [18], Fortuna, Gianni & Trager [4], Traverso & Zanon [22, 21], Weispfenning [23], Kondratyev, Stetter & Winkler [7], Gonzalez-Vega, Traverso & Zanon [5], Stetter [20], Bodrato & Zanon [1], Mourrain と協力者 [8], 等により研究された。計算誤差が増大すれば、第一種は計算精度 (precision) を上げればよいが、第二種は結果の精度 (accuracy) を保つ必要がある。多くの研究にも拘らず、第2種はごく最近まで深刻な問題であった。実際、ヨーロッパでは浮動小数グレブナー基底計算が難しいので、境界基底 (border basis) なる基底が提案され [8]、大々的に研究されている。境界基底とは、イデアルの剰余環の単項式基底を取り巻く単項式集合の基底で、当然、0次元イデアルにしか定義できない。しかも、それが安定に計算できるか、筆者らは疑問に思っている。

一昨年、Sasaki-Kako [11, 12] は計算が不安定な原因を明らかにし、一つの安定化法を呈示して突破口を開いた。その方法は微小主係数を独立記号で置き換えて記号係数として扱うもので、計算が非常に重くて実用にならない。昨年、Sasaki-Kako [13, 14] は記号を用いない実際的方法を呈示した。それは入力多項式の係数を高精度有効浮動小数 [6] に変換して計算するもので、**高精度化法**と呼ぶことにする。

論文 [11, 12] は不安定性の原因として二つ挙げている。第一は**主要項のキャンセル** (主要項は大きな係数を持つ項) であり、第二は**完全誤差項** (誤差のみから成る項) の出現である。浮動小数による代数的計算を観察すると、多項式の加減算で主要な項が互いにキャンセルして、大きな誤差が生じることが頻繁にある。

*sasaki@math.tsukuba.ac.jp

†kai@cs.ehime-u.ac.jp

主要項は正確にキャンセルすることも多いが、浮動小数計算では正確なキャンセルは大抵0とならず、完全誤差項となって残る。完全誤差項が主項に現れた場合、以後の計算は全く誤ったものとなる。完全誤差項は係数を有効浮動小数で表現することで簡単に除去できる。Shirayanagi-Sweedlerのように区間数を用いてもよいが、区間数では桁落ち量の推定ができない。

論文 [11, 12] は主要項のキャンセルを二つの型、**自己簡約**と**本質的キャンセル**、に分類した。自己簡約は微小主項または巨大主項の多項式により引き起こされ、数値行列のガウス消去において微小ピボット行による消去が大きな桁落ちを引き起こすように、大きな桁落ち誤差を引き起こす。自己簡約は主要項の**正確なキャンセル**であり、自己簡約による精度低下は計算法の工夫で回避可能である。本質的キャンセルは主要項の**不完全なキャンセル**であり、悪条件数値行列の消去で起きるキャンセルと同様、除去が難しい。論文 [12] は本質的キャンセルが近似グレブナー基底計算の精度低下に深く関係することを指摘している。

論文 [13, 14] で呈示された高精度化法の要点は次の定理である。

定理 1 項の正確なキャンセルによる桁落ち誤差は、計算精度を増大すると、その分だけ小さくなる。 ◇

この定理は、論文 [14] では自己簡約に関して証明されたが、証明に必要なのは自己簡約ではなく正確な項キャンセルである。入力時に全係数を高精度浮動小数に変換すれば、入力精度以上の桁は全て意味のない数となるが、計算機は意味のない数も正当に扱い、幾つかの項が正確にキャンセルすると、その最下位桁のゴミが他の同類項に誤差として伝搬する。結果として上位桁の意味ある数の精度が保たれるのである。

高精度化法は極めて実地的な方法だが、本質的桁落ちには無力で、次の課題を抱えている。

1. 正確な項キャンセルを起こすメカニズムを全て列挙すること。
2. 本質的桁落ちを定量化できないか（条件数を定義できないか）？
3. 本質的桁落ち量を見積もる簡単で実地的な方法はないか？
4. 高精度計算は高価なので、自己簡約による桁落ち量を減らせないか？
5. さらに、本質的桁落ち量を減らすことは可能か？

本稿はこれらの全課題に取り組むものだが、研究はまだ道半ばである。しかし、方向は見えているので、幾つかの成果と方向性を記述する。また、上記 3. の課題に関しては、論文 [13] に記述した方法が不十分なことが判明したので、よりよい方法を呈示する。

2 グレブナー基底計算における桁落ちメカニズムの復習

条件数を定義したり桁落ちの低減を図るには、桁落ちメカニズムを理論的に知ることが必要であるので、Sasaki-Kako の理論を復習する。初めに、本稿では主項消去のみによってグレブナー基底を計算することを強調しておく。このことは本章の議論で非常に重要である。

F, G 等は多変数多項式を表すとする。多項式 F のノルムは $\|F\|$ で表す；本稿では 1 ノルム $\|F\|_1$ (係数の絶対値の和) を用いる。係数なしの項をべき積と呼ぶ。lt(F), lc(F), rt(F) は項順序 $\succ$ に関する F の主項、主係数、残余項をそれぞれ表す： $F = \text{lt}(F) + \text{rt}(F)$ 、 $\text{lt}(F) \succ \text{rt}(F)$ 。Spol(F, G) は F と G の **S** 多項式を、Lred(F, G) は lt(F) の G による **M** 簡約を表す。Lred(F, G) は $F \xrightarrow{G} \tilde{F}$ と表される。 $F \xrightarrow{G} \tilde{F}$ は F を G により可能な限り、すなわち lt($\tilde{F}$) が G に関して **M** 既約になるまで簡約することを表す。

F と G は次の関係式を満たす多項式とする（満たさない場合は、 F あるいは G あるいは双方に 0 係数の項を付加し、強引に満たすようにする）。

$$\left. \begin{aligned} F &= f_1 S_1 + f_2 S_2 + \cdots + f_m S_m \\ G &= g_1 T_1 + g_2 T_2 + \cdots + g_n T_n \end{aligned} \right\} \begin{aligned} S_i, T_i &: \text{べき積} \\ S_i &= T T_i \quad (\forall i) \end{aligned} \quad (2.1)$$

F は G により先頭項から順に k 回M簡約できるとする: $F \xrightarrow{G} \dots \xrightarrow{G} \tilde{F}$ 。部分終結式理論 [2] によれば、 $\tilde{F}$ は F と G の係数により、次式のように行列式で表すことができる (定数倍の不定性は除く)。

$$\tilde{F} = \sum_{i=1}^{m-k} \begin{vmatrix} g_1 & \cdots & g_k & g_{k+i} \\ & \ddots & \vdots & \vdots \\ & & g_1 & g_{1+i} \\ f_1 & \cdots & f_k & f_{k+i} \end{vmatrix} S_{k+i}. \quad (2.2)$$

$\tilde{F}$ は $\tilde{F} = aF + bG$ と表すことができる。ここで、 $b = c_1(S_1/T_1) + \dots + c_k(S_k/T_1)$ ($c_1, \dots, c_k$ は数値) であるが、 $a = (-1)^{k+1}g_1^k$ とすれば、 b は次のように行列式で表すことができる。

$$a = (-1)^{k+1}g_1^k, \quad b = \begin{vmatrix} g_1 & \cdots & g_k & S_1/T_1 \\ & \ddots & \vdots & \vdots \\ & & g_1 & S_k/T_1 \\ f_1 & \cdots & f_k & 0 \end{vmatrix}. \quad (2.3)$$

a と b の対 (a, b) は $\tilde{F}$ の **syzygy** と呼ばれる。

F に加え $F' = f'_1S'_1 + f'_2S'_2 + \dots + f'_mS'_m$ があり、 $S'_i = SS_i$ ($i = 1, \dots, m$) を満たすとする。このとき、 F' も G により k 回M簡約することができるので、簡約した結果を $\tilde{F}'$ と表す: $F' \xrightarrow{G} \dots \xrightarrow{G} \tilde{F}'$ 。当然、 $\tilde{F}'$ も $\tilde{F}$ と同様の行列式で表現できる。 $\tilde{F}$ と $\tilde{F}'$ のS多項式 $\text{Spol}(\tilde{F}, \tilde{F}') = \text{ht}(\tilde{F}')\tilde{F} - \text{ht}(\tilde{F})\tilde{F}'$ を考える。行列式表現 (2.2) によれば、 Spol の第 i 項の係数は次のようになる。

$$\begin{vmatrix} g_1 & \cdots & g_k & g_{k+1} \\ & \ddots & \vdots & \vdots \\ & & g_1 & g_{1+1} \\ f'_1 & \cdots & f'_k & f'_{k+1} \end{vmatrix} \cdot \begin{vmatrix} g_1 & \cdots & g_k & g_{k+i} \\ & \ddots & \vdots & \vdots \\ & & g_1 & g_{1+i} \\ f_1 & \cdots & f_k & f_{k+i} \end{vmatrix} - \begin{vmatrix} \text{same} \\ \vdots \\ \text{same} \\ f'_j \leftrightarrow f_j (\forall j) \end{vmatrix} \quad (2.4)$$

$$= \begin{vmatrix} g_1 & \cdots & g_k \\ & \ddots & \vdots \\ & & g_1 \end{vmatrix} \cdot \begin{vmatrix} g_1 & \cdots & g_k & g_{k+1} & g_{k+i} \\ & \ddots & \vdots & \vdots & \vdots \\ & & g_1 & g_{1+1} & g_{1+i} \\ f'_1 & \cdots & f'_k & f'_{k+1} & f'_{k+i} \\ f_1 & \cdots & f_k & f_{k+1} & f_{k+i} \end{vmatrix}. \quad (2.5)$$

この恒等式は Sylvester の恒等式の特別な場合である。

上記の恒等式と syzygy の行列式表現 (2.3) から、次のことが言える。

- (2.4) で g_1^k に比例しない項は多数出現するが、それらは互いに正確にキャンセルする。それらの項は、浮動小数計算では完全誤差項となって計算を破壊することが多い。
- $|g_1| \ll \|G\|$ のとき、主要項が互いに正確にキャンセルする。このキャンセルが自己簡約である。
- 自己簡約は syzygy 計算でも起き、Spol 計算とほぼ同じ大きさの桁落ちが起きる。
- $\text{ht}(\tilde{F}) \mid \text{ht}(\tilde{F}')$ のときS多項式はM簡約に他ならず、自己簡約はM簡約でも起きる。
- 主要項の自己簡約で生じた完全誤差項が他の小さな同類項に加わると、相対的に大きな誤差となってその項の精度 (accuracy) を下げる。計算精度 (precision) を上げると、自己簡約による誤差は上げた精度分だけ下位桁に移動する。結果として、上位桁の精度が保たれる。

3 浮動小数グレブナー基底の条件数

数値解析では、浮動小数による計算誤差が増大して計算結果の精度 (accuracy) が低下する問題を**悪条件問題** (ill-conditioned problem) と呼び、悪条件性を数値的に表す指標として**条件数** (condition number) を導入している。ただし、計算法の簡単な工夫で回避可能な精度低下は条件数には含めず、本質的な精度低下だけを含まないように定義する。たとえば数値行列のガウス消去では、行列の行の間に近似線形従属関係が成立すれば、どのような順序で行消去を行おうとも近似線形従属分だけの桁落ちが起きる。一方、(1,1) 要素が微小な場合、第1行で消去を行うと2回目の消去で大きな精度低下が起きるが、この現象は行の入れ替え (pivoting) で簡単に回避できる。したがって、数値行列の条件数は近似線形従属性だけを取り込んで、かつ計算法に依存しない形で定義される。

浮動小数グレブナー基底の条件数も同じように定義されるべきである。自己簡約には幾つかの型があるが、今まで見つけた型ではいずれも自己簡約は正確な項キャンセルをひき起こし、正確な項キャンセルが起こす精度低下は高精度化法で回避できるので、条件数には自己簡約による桁落ちを含めるべきではない。一方、本質的桁落ちは不正確な項キャンセルで引き起こされる。典型的な例は、入力多項式を係数ベクトル表示したとき、係数ベクトルの間に近似線形従属関係が成立する場合である。したがって、グレブナー基底の条件数は本質的桁落ちだけを取り込む形で定義すべきである。

具体的に定義する前に、数値行列の条件数の定義を参考にする。 A は与えられた $n \times n$ 正方数値行列で、 $\mathbf{b}$ は与えられた n 次元数値ベクトルとして、線形方程式 $A\mathbf{x} = \mathbf{b}$ を考える。 $\mathbf{b}$ に微小摂動 $\delta\mathbf{b}$ を加えたとき $\mathbf{x}$ は $\delta\mathbf{x}$ だけ変動するとする。

$$A\mathbf{x} = \mathbf{b} \longrightarrow A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}. \quad (3.1)$$

このとき、 $\|\delta\mathbf{x}\|/\|\mathbf{x}\|$ を $\|\delta\mathbf{b}\|/\|\mathbf{b}\|$ で上から押さえる数 $\text{Cond}(A)$ が A の条件数である：

$$\frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \text{Cond}(A) \frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|} \implies \text{Cond}(A) \stackrel{\text{def}}{=} \|A\| \cdot \|A^{-1}\|. \quad (3.2)$$

入力基底とそのグレブナー基底をそれぞれ $\text{IB} = \{F_1, F_2, \dots, F_r\} \subset K$, $\text{GB} = \{G_1, G_2, \dots, G_t\} \subset K$ とし、各 F_i に摂動 δF_i を加えたとき、各 G_j は δG_j だけ変動するとする ($i \in \{1, \dots, r\}$, $j \in \{1, \dots, t\}$)：

$$\begin{array}{ccc} \text{initial basis} & & \text{Gröbner basis} \\ \{F_1, F_2, \dots, F_r\} & \implies & \{G_1, G_2, \dots, G_t\} \\ \downarrow & & \downarrow \\ \{F_1 + \delta F_1, \dots, F_r + \delta F_r\} & \implies & \{G_1 + \delta G_1, \dots, G_t + \delta G_t\} \end{array} \quad (3.3)$$

行列の条件数に倣い、グレブナー基底の条件数 $\text{Cond}(\text{GB})$ は次の不等式を満たすように定義する。

$$\text{Cond}(\text{GB}) \cdot \max \left\{ \frac{\|\delta F_1\|}{\|F_1\|}, \dots, \frac{\|\delta F_r\|}{\|F_r\|} \right\} \geq \max \left\{ \frac{\|\delta G_1\|}{\|G_1\|}, \dots, \frac{\|\delta G_t\|}{\|G_t\|} \right\}. \quad (3.4)$$

摂動 $\delta F_1, \dots, \delta F_r$ を大きくするとどこかでイデアルの次元が変化し、グレブナー基底自体がガラリと変わるだろうが、上記はそのような大きな変化が生じない範囲での定義であることに注意されたい。

不等式 (3.4) を満たす $\text{Cond}(\text{GB})$ を本質的桁落ちで表すことが課題なので、まず本質的桁落ちを定式化する。初期基底 IB とグレブナー基底 GB の間には各要素間の線形関係が存在する。たとえば、各 G_j は $F_1, \dots, F_r$ により次のように表すことができる ($(a_{j1}, \dots, a_{jr})$ は G_j の **syzygy** といわれる)：

$$G_j = a_{j1}F_1 + \dots + a_{jr}F_r \quad (j = 1, \dots, t), \quad a_{ji} \in K \quad (i = 1, \dots, r). \quad (3.5)$$

IB と GB の各要素には定数倍の不定性があるから、次のように規格化する。

$$\|F_i\|_1 = 1 \quad (i = 1, \dots, r), \quad \max\{\|a_{j1}\|_1, \dots, \|a_{jr}\|_1\} = 1 \quad (j = 1, \dots, t). \quad (3.6)$$

このとき、 G_j に生じた本質的桁落ち $\text{Ic}(G_j)$ は次式で表される（桁落ちが生じれば $\|G_j\| \ll 1$ となる）。

$$\text{Ic}(G_j) \stackrel{\text{def}}{=} \frac{1}{\|a_{j1}F_1 + \cdots + a_{jr}F_r\|_1}. \quad (3.7)$$

ここで、ノルムとして 1 ノルムを用いるのは、条件数の定義に便利だからである。

定理 2 グレブナー基底の条件数 $\text{Cond}(\text{GB})$ を次式で定めれば、不等式 (3.4) が満たされる。

$$\text{Cond}(\text{GB}) \stackrel{\text{def}}{=} r \times \max\{\text{Ic}(G_1), \dots, \text{Ic}(G_t)\}. \quad (3.8)$$

証明 規格化 (3.6) より、(3.4) 左辺の $\max$ は $\text{Acc}(\text{IB}) \stackrel{\text{def}}{=} \max\{\|\delta F_1\|_1, \dots, \|\delta F_r\|_1\}$ となる。1 ノルムの特性 $\|PQ\|_1 \leq \|P\|_1 \|Q\|_1$ と G_j の規格化より、

$$\begin{aligned} \|\delta G_j\|_1 &= \|a_{j1}\delta F_1 + \cdots + a_{jr}\delta F_r\|_1 \\ &\leq \|a_{j1}\|_1 \cdot \|\delta F_1\|_1 + \cdots + \|a_{jr}\|_1 \cdot \|\delta F_r\|_1 \\ &\leq \|\delta F_1\|_1 + \cdots + \|\delta F_r\|_1 \leq r \times \text{Acc}(\text{IB}) \end{aligned}$$

となる。この式と $\text{Ic}(G_j) = 1/\|G_j\|_1$ ($j = 1, \dots, t$) から (3.4) が得られる。◇

例 1 本質的キャンセルが起きる例（論文 [13] で与えた例）。

係数ベクトル表示から、 P_1, P_2, P_3 の間には近似度 $O(10^{-4})$ の近似線形関係が成立することが分かる。

$$\begin{cases} P_1 = 57/56 x^2 y + 68/67 x z^2 - 79/78 x y + 89/88 x \\ P_2 = x y z^3 - x y^2 z + x y z \\ P_3 = 56/57 x y^2 - 67/68 y z^2 + 78/79 y^2 - 88/89 y \end{cases} \quad (3.9)$$

与式 P_1, P_2, P_3 の数係数を倍長浮動小数に変換して係数に誤差を入れたのち、30 桁精度の多倍長浮動小数でグレブナー基底を計算すると、下記の結果が得られる（下線は正しい数字を表す）。

$$\begin{cases} P_1, P_2, P_3 \text{ are unchanged,} \\ P_6 = y^2 z^2 - \underline{2.995436947732552644538319700370} x y^2 \\ \quad - \underline{1.0020782165123748257674951096740} y^3 \\ \quad + \underline{1.9983254691737245140192885621560} x y + \dots, \\ P_7 = x z^2 - \underline{1.764316342370426661429391997320e-3} y z^2 \\ \quad - \underline{9.947232450186805419457332443380e-1} x y \\ \quad + \underline{1.7679829737261936385647927531480e-3} y^2 + \dots \end{cases}$$

高精度計算でも幾つかの係数に $O(10^4)$ の桁落ち誤差が発生しているのが分かる。◇

4 本質的桁落ち量の実際的评价法

条件数を評価するには、各 G_j に生じた本質的桁落ち量 $\text{Ic}(G_j)$ を評価する必要がある。(3.5) によれば、 G_j の syzygy を計算すれば $\text{Ic}(G_j)$ が得られるが、syzygy の計算はグレブナー基底自体の計算よりも重い。そこで、syzygy を計算しないで $\text{Ic}(G_j)$ を実際的に評価したい。実は、これは論文 [13] で既に考えたことであり、個々の自己簡約のステップでは計算法は分かっている。しかしながら、その方法をグレブナー基底の計算全体に適用するとき、考えが不十分だったことが判明した。不十分な点は桁落ちの伝搬過程である。たとえば、多項式の和 $P_1 + P_2 = P'$ において、項 T_1, T_3 に自己簡約桁落ち S_1 と本質的桁落ち I_1 が発生

し、次に和 $P' + P_3 = P''$ において、項 T_2, T_4 に自己簡約桁落ち S_2 と本質的桁落ち I_2 が発生したとする。このとき、全体の和 $P_1 + P_2 + P_3$ において発生する桁落ちは $S_1 \times S_2$ と $I_1 \times I_2$ では決していない。この例は極端で、自己簡約では通常、主要項が全てキャンセルするので、個々の係数毎に見る必要はないのでは、との意見もあろう。しかし、実験してみると、桁落ちを個々の係数毎に処理しない場合、グレブナー基底の桁落ち評価が実際値と全くかけ離れてしまうのである。

個々の係数毎に自己簡約による桁落ちと本質的桁落ちを区別するには、多項式の表現法を変更する必要がある、それはやりたくないのが当初は困っていた。しかし、真に必要なのは本質的桁落ちの評価であり、自己簡約桁落ちは大雑把な評価でよい（自己簡約桁落ち量は、計算に必要な精度 (precision) の見積もりに使われるだけ）、また、自己簡約の発生は事前に予知可能である（ただし、6章の問題点を参照）、の3点から、次のような簡単な工夫を考案した。

1. 自己簡約が起きる計算（多項式の和）を事前に検出し、その和計算を特別扱いする。
 2. その和計算では、個々の係数で桁落ちをモニターして、桁落ちが発生したら、元の精度に戻してやる。と同時に、その多項式和全体で生じた桁落ちの最大値を結果式の自己簡約桁落ちとみなす。
 3. グレブナー基底の要素多項式の各係数に残る桁落ちはすべて本質的桁落ちとみなす。
 4. 2. で定めた自己簡約桁落ちは“一つの多項式に一つの桁落ち量”の形で伝搬させる；多項式 P_1 と P_2 の自己簡約桁落ちが S_1 と S_2 ならば、和 $P_1 + P_2$ の自己簡約桁落ちは $S_1 \times S_2$ と計算する。
2. では本質的桁落ちを自己簡約桁落ちとみなす危険があり、本質的桁落ちが過小評価される危険性がある。また、桁落ち伝搬に対する 4. の計算法では、自己簡約桁落ちが過大評価される危険性が多々ある。

例 2 巨大な誤差を引き起こす簡単な系 ([11] 等で与えた例)。左が初期基底で、右がグレブナー基底。 P_2 と P_5 には $O(10^9)$ の自己簡約桁落ちが、 P_5 には $O(10^2)$ の本質的桁落ちが生じることが分かっている。

$$\left\{ \begin{array}{l} P_1 = x^3/10.0 + 3.0x^2y + 1.0y^2 \\ P_2 = 1.0x^2y^2 - 3.0xy^2 - 1.0xy \\ P_3 = y^3/10.0 + 2.0x^2 \end{array} \right\} \Rightarrow \left\{ \begin{array}{l} P_2 = y^2 \\ P_4 = xy + \dots \\ P_5 = x^2 + \dots \end{array} \right\}$$

上述の“本質的桁落ち評価法”を適用すると、 P_2, P_4, P_5 の自己簡約桁落ちと本質的桁落ちが次表のように評価された。

P_2	Self-Red = 333395852.3	Intr-Can = 8.078
P_4	Self-Red = 89.3	Intr-Can = 8.078
P_5	Self-Red = 429353390.1	Intr-Can = 9.040

今の場合、自己簡約桁落ちは非常に良く評価されており、本質的桁落ちは小さめに評価されている。大きな自己簡約桁落ちを差し引いた事を考慮すると、なかなかよい評価と言えると思う。◇

5 自己簡約の低減を目指して

応用面での多くの計算では、桁落ちの大部分は自己簡約桁落ちが占める。実際、本質的桁落ちが入力精度を超えると、得られたグレブナー基底は全く意味を失う。一方、自己簡約桁落ちは時として10進100桁を超えることも珍しくないのが、桁落ちを低減して計算のスピードアップを図るためには、自己簡約桁落ちを低減する工夫が必要である。

以下では、主項が(相対的に)微小あるいは巨大な多項式を異常と呼ぶ。自己簡約は異常な多項式により引き起こされるから、自己簡約を低減化するには、異常な多項式をうまく処理する必要がある。筆者らは

現在までに下記の三つの方策を試みたが、そのうち一つはかなり成功を収めた。今後、さらに実験を続け、よりよい方策に高めていく予定である。

方策 1 アルゴリズムの前段では、異常な多項式によるM簡約を実行せず、前段でのS多項式生成が終了したら後段に移り、後段では異常な多項式も含めて全ての多項式でM簡約を行う。

結果 0に簡約される多項式が減り、S多項式生成が増えて無限ループに陥る危険性がある。

方策 2 S多項式生成に際して、前段では異常度が異なる組合せを後回しにし、後段でそれらのS多項式を生成する；後回しにしている間に別の多項式で簡約されて、異常多項式が消えることを期待。

結果 その期待は空しく、単に桁落ちが後段に先送りされるだけのことが多い。

方策 3 異常性の低い多項式から順に並べておき、S多項式生成とM簡約においては、前方に並べられた多項式から順に選ぶ。

結果 下記の例が示すように、ある程度の成功を収めた。

例 3 例 2 の系に上記の方策 3 を適用する。

$$\left\{ \begin{array}{l} P_1 = x^3/10.0 + 3.0x^2y + 1.0y^2 \\ P_2 = 1.0x^2y^2 - 3.0xy^2 - 1.0xy \\ P_3 = y^3/10.0 + 2.0x^2 \end{array} \right\} \Rightarrow \left\{ \begin{array}{l} P_5 = y^2 \\ P_2 = x^2 + \dots \\ P_4 = xy + \dots \end{array} \right\}$$

基底計算の戦略が変わったので、得られる基底要素の名前付けも変わったが、項順序は同じ (全次数順序) なので、基底自体は変わらない。自己簡約桁落ちと本質的桁落ちは、次表のように評価された。

P_5	Self-Red = 1010.5	Intr-Can = 0.252
P_2	Self-Red = 89.3	Intr-Can = 0.505
P_4	Self-Red = 1010.1	Intr-Can = 0.432

本質的桁落ち量の評価が小さすぎるのが気になるが、自己簡約桁落ちはかなり低減された。

◇

6 未解決の問題

主な未解決問題は次の 2 点である。

第一点：正確な項キャンセルを起こす全てのメカニズムの列挙。

第二点：本質的桁落ちの低減化。

第一点 4章で与えた本質的桁落ち評価法は、自己簡約桁落ちを検出して除くことが前提になっている。これは、自己簡約を起こすメカニズム、正しく言えば、正確な項キャンセルを起こすメカニズム、の全てが解って初めて可能なことである。しかしながら、そのメカニズムは現在、具体的なタイプがいくつか解っているだけで、全貌が解っているわけではない。論文 [11] では簡単な同一簡約型 (下記参照) を指摘し、論文 [13] では複雑な同一簡約型と対自己簡約 (paired self-reduction) を指摘した。しかし、対自己簡約は不正確な項キャンセルを引き起こすので、対象から外すべきである。

現在は、正確な項キャンセルを引き起こすメカニズムを次の 3 タイプに分類している。

1. 同一簡約型 $F_i \xrightarrow{G_1} \dots \xrightarrow{G_r} \tilde{F}_i$ ($i = 1, 2$) のとき、 $\text{Spol}(\tilde{F}_1, \tilde{F}_2)$ で生じる項キャンセル。ここで、 $G_1, \dots, G_r$ は同じ多項式でもよく、正常な多項式でもよいが、 G_1 は異常多項式でなければならない。このタイプは論文 [11, 12, 13, 14] で取り上げたものである。

2. 剰余列型 $F_1 \xrightarrow{F_2} F_3, F_2 \xrightarrow{F_3} F_4$ のように、多項式剰余列に類似の型。ここで、 F_2 は異常多項式である。部分終結式理論 [2] が示すように、この場合にも正確な項キャンセルが起きる。
3. 多重剰余列型 3 個以上の多項式 $F_1, \dots, F_r$ から出発して、勝手に商多項式を選びながら（商多項式の主変数に関する次数は被除多項式の次数と同じか低い必要がある）、 $r-1$ 個の剰余列を作ることができて、多重多項式剰余列と命名されている [10]。多重多項式剰余列計算では正確な項キャンセルが起きるので [9, 10]、グレブナー基底計算でも同じ状況では正確な項キャンセルが起きる。

第二点 数値行列の場合、行列の悪条件性は変えようがない（近似特異な行列を特異行列と摂動行列の和に分解するなどの離れ業を使えば別だが）。一方、グレブナー基底計算の場合、多項式を係数ベクトル表示して、ある項順序でみると近似線形従属であっても、別の項順序でみると近似従属性が成立しない場合は多々ある。この性質を使えば本質的桁落ちを低減できそうである。

例 4 次の系のグレブナー基底を、全次数順序と辞書式順序の二通りで計算してみる。

$$\begin{cases} F_1 = x^2(2yz + 1), \\ F_2 = yz(3xz - 2), \\ F_3 = x(3xz - 2) - (2yz + 1). \end{cases}$$

全次数順序で計算すると次の結果が得られる（ $\#E[f, e]$ は値部が f 、誤差部が e の有効浮動小数である）。

$$(GB) = \begin{cases} G_1 = x + \#E[1.33333333333333, 1.01e-11] y, \\ G_2 = yz + \#E[0.058139534883721, 1.47e-13] x \\ \quad + \#E[0.077519379844961, 1.96e-13] y + \#E[0.5, 2.00e-13]. \end{cases}$$

$G_1 = a_1 F_1 + a_2 F_2 + a_3 F_3$, $\max\{\|a_i F_i\|_\infty\} \approx 2000$ ゆえ、 G_1 には $O(10^3)$ の本質的桁落ちが起きている。一方、同じ系を辞書式順序で計算すると次の結果が得られる。

$$(GB) = \begin{cases} G_2 = yz + \#E[0.5, 4.0e-15], \\ G_1 = x - \#E[4.0, 4.0e-15] y^3 z^3 - \#E[2.0, 4.0e-15] y^2 z^2 \\ \quad - \#E[2.666, 2.66e-15] y^2 z + yz + \#E[0.5, 5.0e-16]. \end{cases}$$

$G_1 = a'_1 F_1 + a'_2 F_2 + a'_3 F_3$, $\max\{\|a'_i F_i\|_\infty\} \approx 10$ ゆえ、本質的桁落ち量が大幅に減少した。ただし、順序が異なるので得られるグレブナー基底も異なる。一方のグレブナー基底から他方を導出する良く知られた方法があるが、その方法を適用したとき桁落ちがどのようになるかは、まだ調べていない。◇

参 考 文 献

- [1] M. Bodrato and A. Zanoni. Intervals, syzygies, numerical Gröbner bases: a mixed study. *Proc. of CASC2006 (Computer Algebra in Scientific Computing)*, Springer-Verlag LNCS 4194, 64–76, 2006.
- [2] J.E. Collins. Subresultant and reduced polynomial remainder sequence. *J. ACM*, **14**, 128–142, 1967.
- [3] D. Cox, J. Little and D. O’Shea. *Ideals, Varieties, and Algorithms*. Springer-Verlag New York, 1997.
- [4] E. Fortuna, P. Gianni and B. Trager. Degree reduction under specialization. *J. Pure Appl. Algebra* **164** (2001), 153–164.
- [5] L. Gonzalez-Vega, C. Traverso and A. Zanoni. Hilbert stratification and parametric Gröbner bases. *Proc. of CASC2005 (Computer Algebra in Scientific Computing)*; Springer-Verlag LNCS 3718, 220–235, 2005.

- [6] F. Kako and T. Sasaki. Proposal of “effective” floating-point number. Preprint of Univ. Tsukuba, May 1997 (unpublished).
- [7] A. Kondratyev, H.J. Stetter and S. Winkler. Numerical computation of Gröbner bases. *Proc. of CASC2004 (Computer Algebra in Scientific Computing)*, 295–306, St. Petersburg, Russia, 2004.
- [8] B. Mourrain. Pythagore’s dilemma, symbolic-numeric computation, and the border basis method. *Symbolic-Numeric Computations (Trends in Mathematics)*, 223–243, Birkhäuser Verlag, 2007.
- [9] T. Sasaki and A. Furukawa. Secondary polynomial remainder sequence and extension of subresultant theory. *J. Inf. Proces.*, **7**, 175–184, 1984.
- [10] T. Sasaki and A. Furukawa. Theory of multiple polynomial sequence. *Publ. of RIMS (Research Inst. for Mathematical Sciences)*, **20**, 367–399, 1984.
- [11] 佐々木建昭、加古富志雄. 浮動小数 Gröbner 基底の計算法. 数理研研究集会「数式処理の理論と応用の研究」、2006 年 12 月。
- [12] T. Sasaki and F. Kako. Computing floating-point Gröbner base stably. *Proc. of SNC2007 (Symbolic Numeric Computation)*, 180–189, London, Canada, 2007.
- [13] 佐々木建昭、加古富志雄. 悪条件性を推定する浮動小数グレブナー基底の計算法. 数理研研究集会「Computer Algebra – Design of Algorithms, Implementations and Applications」、京都、2007 年 11 月。
- [14] T. Sasaki and F. Kak.: Floating-point Gröbner Basis Computation with Ill-conditionedness Estimation. *Proc. of 8th ASCM (Asian Symposium on Computer Mathematics)*, pp. 395–409, 2007; see also *LNAI 5081, Computer Mathematics*, D. Kapur (Ed.), pp. 278–292, 2008.
- [15] K. Shirayanagi. An algorithm to compute floating-point Gröbner bases. *Mathematical Computation with Maple V. Ideas and Applications*, Birkhäuser, 95–106, 1993.
- [16] K. Shirayanagi. Floating point Gröbner bases. *Mathematics and Computers in Simulation* **42** (1996), 509–528.
- [17] K. Shirayanagi and M. Sweedler. Remarks on automatic algorithm stabilization. *J. Symb. Comput.*, **26** (1998), 761–765.
- [18] H.J. Stetter. Stabilization of polynomial systems solving with Gröbner bases. *Proc. of ISSAC’97 (Intern’l Symposium on Symbolic and Algebraic Computation)*, 117–124, ACM Press, 1997.
- [19] H.J. Stetter. *Numerical Polynomial Algebra*. SIAM Publ., Philadelphia, 2004.
- [20] H.J. Stetter. Approximate Gröbner bases – an impossible concept *Proc. of SNC2005 (Symbolic-Numeric Computation)*, 235–236, Xi’an, China, 2005.
- [21] C. Traverso. Syzygies, and the stabilization of numerical Buchberger algorithm. *Proc. of LMCS2002 (Logic, Mathematics and Computer Science)*, 244–255, RISC-Linz, Austria, 2002.
- [22] C. Traverso and A. Zanoni. Numerical stability and stabilization of Gröbner basis computation. *Proc. of ISSAC2002 (Intern’l Symposium on Symbolic and Algebraic Computation)*, 262–269, ACM Press, 2002.
- [23] V. Weispfenning. Gröbner bases for inexact input data. *Proc. of CASC2003 (Computer Algebra in Scientific Computing)*, 403–411, Passau, Germany, 2003.