数理解析研究所講究録
986巻 1997年 127-135

近似根とその応用について

筑波大学数学系 北本卓也 (Takuya Kitamoto)
kita@math.tsukuba.ac.jp

1. 序論

多変数多項式の組が与えられたとき、この根全体のなす代数的多様体(0次元とは限らない)の性質(根、次元、種数等)を調べるにはグレブナー基底を求めればよいことが知られているが、グレブナー基底の計算はしばしば非常に重くなり、計算結果も複雑になることで知られている。そこでここではグレブナー基底の代わりに近似根を用いて、準素分解の計算ができることを示す。まず、始めに近似根について述べ、次にグレブナー基底を用いた従来の準素分解アルゴリズムについて述べたあと、近似根を用いた準素分解アルゴリズムを述べる。

2. 近似根

近似根とは多項式イデアルが与えられたとき、その零点を自由変数でべき級数展開したものである。n 変数多項式系 $f_1, f_2, ..., f_m$ $(n \geq m)$ が与えられたときに、近似根を求めるアルゴリズムは次のようである。

2.1. 基本的アルゴルズム

n 個の変数 $x_1, ..., x_n$ に対し m 個の多項式があるので、自由変数の数は $n-m$ 個である。よってここでは一般性を失うことなく、近似根は

$$\begin{array}{ccc} x_1 & = & g_1(x_{n+1}, ..., x_m) \\ \vdots & \vdots & \vdots \\ x_n & = & g_n(x_{n+1}, ..., x_m) \end{array}$$

(ただし、$g_i(x_{n+1}, ..., x_m)$ は $x_{n+1}, ..., x_m$ のべき級数) の形で表されるとする。

アルゴリズムは2つのステップからなる。

Step 1 $x_{n+1} = 0, ..., x_m = 0$ と置き、$f_1 = 0, ..., f_m = 0$ を解き(根の次元は0次元であるので何らかの数値的算法を用いる)、その数値根を $x_1^{(0)}, .., .x_n^{(0)}$ と置く。

Step 2 次の行列 H

$$H = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(x_1^{(0)}, ..., x_n^{(0)}, 0, ..., 0) & \cdots & \frac{\partial f_1}{\partial x_n}(x_1^{(0)}, ..., x_n^{(0)}, 0, ..., 0) \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1}(x_1^{(0)}, ..., x_n^{(0)}, 0, ..., 0) & \cdots & \frac{\partial f_n}{\partial x_n}(x_1^{(0)}, ..., x_n^{(0)}, 0, ..., 0) \end{bmatrix} \quad (1)$$

の逆行列 H^{-1} を計算する (H は数値行列であるので数値的算法により H^{-1} が求まる)。

Step 3 次式より、近似根を漸化的に求める。

$$\begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{bmatrix} \leftarrow -H^{-1} \begin{bmatrix} f_1(x_1^{(k)}, \cdots, x_n^{(k)}, x_{n+1}, ..., x_m) \\ \vdots \\ f_n(x_1^{(k)}, \cdots, x_n^{(k)}, x_{n+1}, ..., x_m) \end{bmatrix} \pmod{(x_{n+1}, ..., x_m)^{k+1}}$$

$$\begin{bmatrix} x_1^{(k+1)} \\ \vdots \\ x_n^{(k+1)} \end{bmatrix} \leftarrow \begin{bmatrix} x_1^{(k)} \\ \vdots \\ x_n^{(k)} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{bmatrix} \tag{2}$$

2.2. 数値例

f_1, f_2 を以下の様に置く。

$$f_1 = x^2 + xy - y + xz + 1$$
$$f_2 = -2x + y + z - 1$$

Step 1 $z = 0$ と置き、$f_1 = 0, f_2 = 0$ を解いて $(x, y) = (0, 1), (0.3333, 1.6667)$ を得る。$x^{(0)} = 0, y^{(0)} = 1$ と置く。

Step 2 H を次のように計算する。

$$H = \begin{bmatrix} \frac{\partial f_1}{\partial x}(x_1^{(0)}, y^{(0)}, 0) & \frac{\partial f_1}{\partial y}(x_1^{(0)}, y^{(0)}, 0) \\ \frac{\partial f_2}{\partial x}(x_1^{(0)}, y^{(0)}, 0) & \frac{\partial f_2}{\partial y}(x_1^{(0)}, y^{(0)}, 0) \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ -2 & 1 \end{bmatrix}$$

H^{-1} は

$$H^{-1} = \begin{bmatrix} -1 & -1 \\ -2 & -1 \end{bmatrix}$$

である。

Step 3

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \leftarrow -H^{-1} \begin{bmatrix} f_1(x^{(0)}, y^{(0)}, z) \\ f_n(x^{(0)}, y^{(0)}, z) \end{bmatrix} \pmod{z^2}$$

$$= \begin{bmatrix} z \\ z \end{bmatrix}$$

$$\begin{bmatrix} x^{(1)} \\ y^{(1)} \end{bmatrix} \leftarrow \begin{bmatrix} x^{(0)} \\ y^{(0)} \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{3}$$

$$= \begin{bmatrix} z \\ 1 + z \end{bmatrix} \tag{4}$$

より、$x^{(1)}=z, y^{(1)}=1+z$

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \leftarrow -H^{-1} \begin{bmatrix} f_1(x^{(1)}, y^{(1)}, z) \\ f_n(x^{(1)}, y^{(1)}, z) \end{bmatrix} \pmod{z^3}$$

$$= \begin{bmatrix} 3z^2 \\ 6z^2 \end{bmatrix}$$

$$\begin{bmatrix} x^{(2)} \\ y^{(2)} \end{bmatrix} \leftarrow \begin{bmatrix} x^{(1)} \\ y^{(1)} \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{5}$$

$$= \begin{bmatrix} z+3z^2 \\ 1+z+6z^2 \end{bmatrix} \tag{6}$$

より、$x^{(2)}=z+3z^2,\ y^{(2)}=1+z+6z^2$ を得る。

3. グレブナー基底を用いた準素分解アルゴリズム（0次元の場合）

3.1. 準素分解アルゴルズム

今、変数 $x_1, ..., x_n$ の多項式 $f_1, ..., f_n$ のなす 0 次元イデアル I を考える。
まず、Generic position を定義する。

定義 1 (Generic position) $\mathrm{rad}(I)$ の零点を $(x_1, ..., x_n)=(\alpha_{1,i}, ..., \alpha_{n,i})\ (i=1, ..., l)$ とする。全ての $j \neq k\ (1 \leq j, k \leq l)$ に対し$\alpha_{i,j} \neq \alpha_{i,k}$ が成り立つ時、I は i で generic position であるという。□

一言でいうと、重根でない根に置いて i 番目の要素が全て異なる時、I は i で generic position である。以後、一般性を失う事無く、$i=1$ であるとし、単に I は generic position であるという。I が generic position で無い時は、適当に $x_1, ..., x_n$ を線形変換する事により常に I を generic position にできる。
このとき、グレブナー基底を用いた準素分解アルゴリズムは以下の様に与えられる。

入力：0 次元イデアル $(f_1, ..., f_n)$ (ただし f_i は $x_1, ..., x_n$) の多項式)
出力：$(f_1, ..., f_n)$ の準素分解 $g_1 \cap \cdots \cap g_m$ と各 g_i の素イデアル h_i

Step 1 $(f_1, ..., f_n)$ の radical を求め、$(\tilde{f}_1, ..., \tilde{f}_n) \leftarrow \mathrm{rad}(f_1, ..., f_n)$ と置く。

Step 2 $(\tilde{f}_1, ..., \tilde{f}_n)$ を generic position にする。

Step 3 $(\tilde{f}_1, ..., \tilde{f}_n) \cap K[x_1]$ を求め、f と置く。

Step 4 f の既約因子を u_i とした時、$h_i=(f_1, ..., f_n, u_i)$ とする。

Step 5 十分大きな整数 q に対し、$g_i=(f_1, ..., f_n, h_i^q)$ とする。

3.2. 数値例

今、f_1, f_2 として

$$f_1=x_1^2-2,\ f_2=x_2^2-2 \tag{7}$$

と取り、(f_1, f_2) の準素分解を計算する。すぐ分かるように (f_1, f_2) の零点は

$$(\sqrt{2}, \sqrt{2}), (\sqrt{2}, -\sqrt{2}), (-\sqrt{2}, \sqrt{2}), (-\sqrt{2}, -\sqrt{2}) \tag{8}$$

である。

Step 1 (f_1, f_2) は重根を持たないので、これそのものが radical である。よって $(\tilde{f}_1, \tilde{f}_2) = (x_1^2 - 2, x_2^2 - 2)$ とおく。

Step 2 (8) よりわかるように、(f_1, f_2) は generic position ではない。よって $\tilde{x}_1 = x_1 + 2x_2$ とし、$x_1 = 2x_2 - \tilde{x}_1$ より、

$(\tilde{f}_1, \tilde{f}_2) = ((2x_2 - \tilde{x}_1)^2 - 2, x_2^2 - 2)$ を得る。

Step 3 $x_2 > \tilde{x}_1$ の項順序を用いて $(\tilde{f}_1, \tilde{f}_2)$ の lex order のグレブナー基底を求めると

$$(\tilde{f}_1, \tilde{f}_2) = (\tilde{x}_1^4 - 20\tilde{x}_1^2 + 36, \tilde{x}_1^3 - 26\tilde{x}_1 + 24x_2) \tag{9}$$

となるので、$f = \tilde{x}_1^4 - 20\tilde{x}_1^2 + 36$ とおく。

Step 4 f は、次のように因数分解されるので

$$f = (\tilde{x}_1^2 - 2)(\tilde{x}_1^2 - 18)$$

$$\begin{aligned} h_1 &= (\tilde{x}_1^4 - 20\tilde{x}_1^2 + 36, \tilde{x}_1^3 - 26\tilde{x}_1 + 24x_2, \tilde{x}_1^2 - 2) \\ &= (\tilde{x}_1^2 - 2, -\tilde{x}_1 + x_2) \\ h_2 &= (\tilde{x}_1^4 - 20\tilde{x}_1^2 + 36, \tilde{x}_1^3 - 26\tilde{x}_1 + 24x_2, \tilde{x}_1^2 - 18) \\ &= (\tilde{x}_1^2 - 18, -\tilde{x}_1 + 3x_2) \end{aligned}$$

ここで、$\tilde{x}_1 = x_1 + 2x_2$ を h_1, h_2 に代入し、$x_1 < x_2$ の項順序で h_1, h_2 のグレブナー基底を求めると

$$\begin{aligned} h_1 &= (x_1^2 - 2, x_1 + x_2) \\ h_2 &= (x_1^2 - 2, -x_1 + x_2) \end{aligned}$$

を得る。

Step 5 始めに述べたように、(f_1, f_2) は radical なので $g_1 = h_1,\ g_2 = h_2$ である。

3.3. Shape lemma

先ほどの数値例において、(9) のグレブナー基底が $(w_1(\tilde{x}_1), x_2 - w_2(\tilde{x}_1))$ (ただし、$w_i(\tilde{x}_1)$ は有理数係数の多項式) の形になっているが、これは偶然ではない。次の lemma が成り立つ。

補題 1 (Shape lemma)
$I=(f_1,...,f_n)$ が 0 次元イデアルであり、radical かつ generic position であるとする。この時、$x_1<x_2,...,x_n$ の項順序で計算された lex order のグレブナー基底は

$$(g_1(x_1),x_2-g_2(x_1),...,x_n-g_n(x_1))$$

の形に書ける。ただし、$g_i(x_1)$ は有理数係数の多項式、g_1 はモニックかつ無平方であり、g_i $(i=2,...,n)$ は $\deg(g_i)<\deg(g_1)$ を満たす。□

よって $I=(g_1(x_1),x_2-g_2(x_1),...,x_n-g_n(x_1))$ と書ける事になる。先ほどの準素分解アルゴリズムの *Step* 4 において

$$h_i=(g_1(x_1),x_2-g_2(x_1),...,x_n-g_n(x_1),u_i(x_1))\ \text{(ただし、}u_i\text{ は }g_1(x_1)\text{ の既約因子)}$$

である事に注意すると、

$$h_i=(u_i(x_1),x_2-\tilde{g}_2(x_1),...,x_n-\tilde{g}_n(x_1))\ \text{(}\tilde{g}_i(x_1)\text{ は }g_i(x_1)\text{ を }u_i(x_1)\text{ で簡約化したもの)} \tag{10}$$

である事が分かり、結局、素イデアル h_i は (10) の形で求まる事が分かる。

4. 数値根を用いた準素分解

4.1. アルゴルズム

まず、次の事実に注意する。代数的閉体 K 上で、0 次元イデアル $I=(f_1,...,f_n)$ の準素分解を求めると、その素イデアルは常に $(x_1-\alpha_1,...,x_n-\alpha_n)$ (ただし、$\alpha_i\in K$ は I の零点) と簡単な形になる。つまり、素イデアルは x_i に関して線形になる。先ほどの数値例で何故この様にならなかったかというと、係数が整数であり、代数的閉体でなかったからである。この場合には、先ほどの数値例の計算結果にあるように $x-\alpha_1$ の所へは α_1 の最小多項式が入る。この事実は (10) からも見て取れる (既約多項式とはその根の最小多項式である)。故に数値根からその最小多項式を格子算法を用いて構成すれば、グレブナー基底を用いること無く素イデアルの始めの部分を求めることが出来る。素イデアルの残りの部分は (10) の形をしている事が分かっているので、これも格子算法で整数係数を決めてやる事により求まる。アルゴリズムを簡単に述べると以下のようになる。

入力：0 次元イデアル $(f_1,...,f_n)$ (ただし f_i は $x_1,...,x_n$) の多項式)
出力：$(f_1,...,f_n)$ の準素分解 $w_1\cap\cdots\cap w_m$ と各 w_i の素イデアル h_i

Step 1 $(f_1,...,f_n)$ の全ての零点を求め (重複度を含む)、S とおく。

Step 2 $(f_1,...,f_n)$ を generic position に変換し、それに応じて S を $\tilde{S}$ に変換する。

Step 3 $\tilde{S}$ より零点を一つ取り出し (零点の重複度を m とする)、$\alpha_1,\cdots,\alpha_n$ と置き、α_1 の最小多項式 $g_1(x)$ を格子算法により求める。最小多項式の次数を e と置き、その α_1 以外の根に対応する零点を $\tilde{S}$ から除く。

Step 4 次式を満たす整数 k_i を

$$\alpha_i = k_0 + k_1\alpha_1 + \cdots + k_{e-1}\alpha_1^{e-1}$$

格子算法により求め、

$$g_i(x) = k_0 + k_1 x + \cdots + k_{e-1}x^{e-1}$$

とおく。

Step 5 h_i を次のように置き、

$$h_i = (g_1(x_1), x_2 - g_2(x_1), ..., x_n - g_n(x_1))$$

w_i を

$$w_i = h_i^m$$

とおく。

Step 6 $\tilde{S}$ が空で無ければ *Step* 3 へ行く。そうでなければ、アルゴリズムを終了する。

4.2. 数値例

f_1, f_2 を (7) とする。

Step 1

$$S \leftarrow \{(1.414, 1.414), (1.414, -1.414), (-1.414, 1.414), (-1.414, -1.414)\}$$

とする。

Step 2 前の数値例同様に $\tilde{x}_1 = x_1 + 2x_2$ とする。この時、

$$\tilde{S} \leftarrow \{(4.242, 1.414),\ (-1.414, -1.414),\ (1.414, 1.414),\ (-4.242, -1.414)\}$$

となり、generic position となる。

Step 3 $\tilde{S}$ より、$(\alpha_1, \alpha_2) = (4.242, 1.414)$ を取り出す ($m = 1$ と置く)。

$$\begin{bmatrix} 10000 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 10000\alpha \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 10000\alpha^2 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

に格子算法を適用することにより、$\begin{bmatrix} 0 & -18 & 0 & 1 \end{bmatrix}^T$ を得るので α_1 の最小多項式は $g_1(x) = x^2 - 18$ である。よって $e = 2$ と置き、$\tilde{x} = -4.24264 (= -\sqrt{18})$ に対応する零点 $(-4.24264, -1.41421)$ を $\tilde{S}$ から除く。

Step 4 格子算法を

$$\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \alpha_1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} \alpha_2 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

上の格子に適用すると、$\begin{bmatrix} 0 & 0 & -1 & 3 \end{bmatrix}^T$ を得るので、

$$g_1(x) = 3x$$

とおく。

Step 5 h_1 を

$$h_1 = (\tilde{x}_1^2 - 18, \tilde{x}_1 - 3x_2)$$

と置き、w_1 を $w_1 = h_1$ と置く。

Step 6 $\tilde{S}$ が空でないので *Step* 3 へ行く。

ここから先ほどと同様にして、

$$h_2 = (\tilde{x}_1^2 - 2, \tilde{x}_1 - x_2), \quad w_2 = h_1$$

を得る。以上より、グレブナー基底を用いた算法と同じ計算結果を得た。

5. グレブナー基底を用いた準素分解アルゴリズム（一般の場合)

変数 $x_1, ..., x_m$ の多項式 $f_1, ..., f_n \in K[x_1, ..., x_m]$ のなす I を考える。I は 0 次元であるとは限らない。グレブナー基底を用いたアルゴリズムでは I の準素分解は I を 0 次元のイデアルに変換して行なわれる。

今、$x_1, ..., x_m$ のうち自由変数を $x_{l+1}, ..., x_m$ とする。$f_1, ..., f_n \in K[x_1, ..., x_m]$ を $f_1, ..., f_n \in K(x_{l+1}, ..., x_m)[x_1, ..., x_l]$ であると考えたとき、これを $K[x_1, ..., x_m]$ から $K(x_{l+1}, ..., x_m)[x_1, ..., x_l]$ への extension といい、この時 $f_1, ..., f_n \in K(x_{l+1}, ..., x_m)[x_1, ..., x_l]$ がなすイデアルを I^e と書く。また、逆に $f_1, ..., f_n \in K(x_{l+1}, ..., x_m)[x_1, ..., x_l]$ に係数部の有理関数の分母の LCM $h_1, ..., h_n$ をかけ、分母を全てはらって $f_1h_1, ..., f_nh_n \in K[x_1, ..., x_m]$ とすることを contraction といい、$J = f_1, ..., f_n \in K(x_{l+1}, ..., x_m)[x_1, ..., x_l]$ の時、$J^c = (f_1h_1, ..., f_nh_n)$ と書く。

$K(x_1, ..., x_l)$ は体であり、$x_1, ..., x_m$ のうち自由変数を $x_1, ..., x_l$ であるので I^e は 0 次元イデアルである。よって イデアル I を extraction し、$K(x_{l+1}, ..., x_m)[x_1, ..., x_l]$ で準素分解を 0 次元の場合のアルゴリズムを用いて行ない、その結果を contraction して $K[x_1, ..., x_m]$ に戻せば良いように思える。しかしながら、実は $I \supset I^{ec}$ は常に成り立つが $I = I^{ec}$ は一般には成り立たない。例えば、$I = (xy)$ とし、I の $K(x)[y]$ への extraction を考えると

$$\frac{1}{x}xy = y = \in I^e \cap K[x, y] = I^{ec}$$

であるが、明らかに $y \notin I$ である。

しかしながら次の定理が成り立つ。

定理 1 (I と I^{ec} の関係)
I の要素 f_i を $f_i \in K(x_{l+1},...,x_m)[x_1,...,x_l]$ の要素と考えて、extention し、それをまた contraction した イデアルを I^{ec} とする。$x_1,...,x_l > x_{l+1},...,x_m$ を満たす項順序で求めた I のグレブナー基底を G とする。f を次式のように置く。

$$f = LCM\{HC(g) \mid g \in G\} \tag{11}$$

ただし、$HC(g) \in K[x_{l+1},...,x_m]$ は $g \in G$ を $x_1,...,x_l$ の多項式と見たときの Head Coefficient を表わす。この時、十分大きな自然数 s に対して

$$I = I^{ec} \cap (I, f^s)$$

が成り立つ。

今 I の最大独立変数を $x_1,...,x_l$ とすると I は係数を $K(x_{l+1},...,x_m)$ とする 0 次元イデアルとみなせる。よって 0 次元イデアルの準素イデアル分解アルゴリズムを用いて、準素分解をおこない、結果を contraction すれば 0 次元以外のイデアルの準素分解が行える。ただし、この場合は途中でイデアルを extension した後 contraction しているので I^{ec} を準素分解したことになり、先に述べたように必ずしも I の準素分解の結果と一致するとは限らない。そこで、上の定理を用いて再帰的に準素分解を行う。つまり、

$$I \text{ の準素分解} = I^{ec} \text{ の 0 次元準素分解} \cap (I, f^s) \text{ の準素分解}$$

を用いて計算を行う。ここで イデアル (I, f^s) の次元は イデアル I の次元より常に小さいので、いつかは右辺の (I, f^s) が 0 次元になり、計算が終了する。以上をアルゴリズムにまとめると、次のようになる。

入力：イデアル $(f_1,...,f_n)$ (ただし f_i は $x_1,...,x_m$) の多項式)
出力：$(f_1,...,f_n)$ の準素分解 $w_1 \cap \cdots \cap w_k$ と各 w_i の素イデアル h_i

Step 1 もし、$1 \in (f_1,...,f_n)$ ならば ϕ を返す。

Step 2 最大独立変数 $x_{l+1},...,x_m$ を求める。

Step 3 f_i を $f_i \in K(x_{l+1},...,x_m)[x_1,...,x_l]$ に extension し、0 次元準素分解アルゴリズムを用いて $(f_1,...,f_n)$ の準素分解を行う。その結果を contraction し、準素イデアルを $\tilde{w}_i$、その素イデアルを $\tilde{h}_i$ と置く。

Step 4 定理 1 の f を求める。

Step 5 入力を $(f, f_1,...,f_n)$ として *Step* 1 へ行き、$(f, f_1,...,f_n)$ の準素分解を求め、準素イデアルを $\hat{w}_i$、その素イデアルを $\hat{h}_i$ と置く。

Step 6 準素イデアルを $\{w_1,...,w_k\} = \{\tilde{w}_1,...,\tilde{w}_{k_1},\hat{w}_1,...,\hat{w}_{k_2}\}$ その素イデアルを $\{h_1,...,h_k\} = \{\tilde{h}_1,...,\tilde{h}_{k_1},\hat{h}_1,...,\hat{h}_{k_2}\}$ とし、帰る。

6. 近似根を用いたアルゴリズム

与えられたイデアルが 0 次元の場合は、Grobner 基底の代わりに数値根と格子算法を用いて primary component が計算できることを示した。イデアルの次元が正の場合も、近似根と格子算法を用いて primary component が計算できる（ただし Grobner 基底の計算が全くいらないというわけではない)。

具体的には、先ほどのアルゴリズムの *Step* 3 で extension したイデアルの準素分解を行う場合に近似根と格子算法を用いて primary component を計算する。基本的なアイデアは、 0 次元のイデアルの準素分解を数値根と格子算法を用いて行ったときと同じである。Shape lemma を用いて Grobner 基底の形を決め、格子算法で係数を決めていく。

0 次元の場合の Grobner 基底

$$(a_{1,0} + a_{1,1}x_1 + \cdots + a_{1,q}x_1^q,$$
$$a_{2,q}x_2 - (a_{2,0} + a_{2,1}x_1 + \cdots + a_{2,q-1}x_1^{q-1}),$$
$$\cdots,$$
$$a_{l,q}x_l - (a_{l,0} + a_{l,1}x_1 + \cdots + a_{l,q-1}x_1^{q-1}))$$

では $a_{i,j}$ は整数であるが、今回は K は有理数を係数とする 自由変数 $x_{l+1}, ..., x_m$ の有理式からなる体であるので $a_{i,j}$ は $x_{l+1}, ..., x_m$ の多項式になる。よって $x_1, ..., x_l$ を近似根として $x_{l+1}, ..., x_m$ のべき級数で展開し、上式が 0 になるように係数を決めればよい。

7. 結論

近似根の準素分解への応用の基本的なアイデアを示した。近似根を用いれば、グレブナー基底の計算の一部を格子算法で置き換えることができ、効率的な計算が可能になる（格子算法は多項式時間アルゴリズムである）と思われる。いくつか計算した数値例ではうまく計算が行えるようであるが、今回はあくまでも基本的なアイデアであり、計算結果の正当性を保証する近似根の精度等についてはまだ結果が得られていない。これらが今後の課題である。

参 考 文 献

[BW 93] T. Becker and V. Weispfenning: *Grobner Basis*, Springer-Verlag, 1993.

[GCL 92] K.O.Geddes, S.R.Czapor, G.Labahn: *Algorithms for Computer Algebra*, Kluwer Academic Pub., 1992.

[Sas 88] 佐々木: 近似的代数計算法数理研講究録 (Collection of Research Reports, Research Institute of Mathematical Study, Kyoto University) **676**, 1988, 307–319.