数物科学概論/計算代数入門

野呂 正行

(神戸大学理学部)

(pdf ファイルが

http://www.math.kobe-u.ac.jp/HOME/~noro

より入手可)

計算代数（Computer Algebra）

文字通り，計算機上で代数的計算を行う

より具体的には，

- 種々の代数的対象を計算するアルゴリズムの開発
- 計算機上での効率的実装

キーワード： アルゴリズム，記号計算，厳密な（誤差がない）計算，厳密計算のための近似計算

計算数学とふつうの数学の違い

例: 多項式の既約判定，因数分解

- ふつうの数学

  K を体とするとき，$K[X]$ は UFD である．よって多項式は既約多項式の積に一意的に分解する．

  分解できない多項式が既約多項式である．

- 計算数学

  既約性をどうやって判定するか．

  既約でない場合，どうやって分解するか．

  どうやれば効率よく分解できるか．

計算の舞台

- 数

  整数，有理数，代数的数，有限体 etc.

- 多項式環

  種々の体上，一変数，多変数

- 剰余環，加群

  非可換環も扱う (微分作用素環，行列環，Lie 環 etc.)

- 群

  有限群 (置換群)，行列群 etc.

計算の対象

- 種々の分解

  整数，多項式の素因子分解，イデアルの準素分解
- 方程式求解

  多項式 GCD，グレブナー基底，終結式による変数消去，微分方程式の求解
- 種々の記号的計算

  微分，初等関数の不定積分，Taylor 展開 etc.
- 数学，特に代数学，代数幾何学における実験

  次元，コホモロジーの計算 etc.
- 暗号，符号

例： 整数

CPU（ハードウェア）が提供する整数計算 ：（例えば)32bit 整数の四則

⇒ 誤差なし計算を続けるとすぐに範囲を逸脱

⇒ 任意の大きさの整数を扱うアルゴリズム，ソフトウェアが必要

⇒ 既に自明ではない（cf. Knuth ： TAOCP 第 2 巻)

最近では gmp というライブラリを使うのが主流

数値計算との違い

完全に排反するものではないが，方向性の違いがある

- **誤差のない計算**

  $3x - 1 = 0$ **の解は** $1/3$ **であって** $0.333\cdots3$ **ではない**

- **式そのものを計算対象にする**

  多項式も純代数的に扱う ： 多項式関数とは明確に区別する

- **独特な近似計算**

  元に戻れる近似計算

  有限体から有理数値へ，多項式から有理式へ etc.

計算代数固有の困難： 中間式膨張

例： 互除法による GCD 計算

K : **体**, $f, g \in K[x]$

$f = qg + r \Rightarrow \mathrm{GCD}(f, g) = \mathrm{GCD}(g, r)$

この性質により，剰余を次々と計算して，割り切れたらそれが $\mathrm{GCD}(f, g)$

$\Rightarrow$ **工夫なしに実行すると，途中の剰余の係数が巨大化**

($\mathrm{GCD}(f, g) = 1$ **の場合が最悪)**

$\Rightarrow$ **なんらかの工夫が必要**

キーワード： 近似計算

計算代数における近似計算

原理

ある整数 x が $|x| < M$ であることがあらかじめ分かっている場合

1. $M' > 2M$ なる整数を用意する.
2. $y \equiv x \bmod M'$, $|y| < M$ となる整数 y を求めれば $x = y$.

   ($y \equiv x \bmod M'$ となる y は $y = x + kM'$ と書けるが, $|y| < M$ の範囲には一つしかない.)

y の求め方

- 十分多くの p_i に対し $y_i \equiv x \bmod p_i$ なる y_i を求め，中国剰余定理（CRT）を適用
- ある p に対し $y_1 \equiv x \bmod p_i$ なる y_1 を求め，これから Hensel lifting により $y_k \equiv x \bmod p^k$ なる y_k を求める．

p_i，p としては通常素数を使う $\Rightarrow$ 有限体 $\mathrm{GF}(p)$ 上の計算

CRT (Chinese Remainder Theorem)

p_i $(i = 1, \ldots, r)$ **が二つずつ互いに素とする.**

$M = \Pi_{i=1}^{r} p_i$**,** $M_i = M/p_i$ **とおく.**

$B_i = (M_i \bmod p_i)^{-1} \cdot M_i$ **とおけば**

$$y \equiv y_i \bmod p_i \Leftrightarrow y = y_1 B_1 + \cdots + y_r B_r + kM$$

(k **は整数;** $\bmod M$ **で一意的という意味)**

有限体を使う理由

- **中間膨張が起きない**

  常に p で割った剰余を保持するから

  例えば $\mathrm{GCD}(f, g) = 1$ は極めて高速に判定できる

- **有限体固有のアルゴリズムが適用可**

  例： 多項式因数分解における，Berlekamp アルゴリズム

有理数の場合

有理数を復元するにはどうするか?

整数–有理数変換

1. **有理数解を** $x = n/d$ **とする．** d **と互いに素な数** M **に対し，整数**y **が** $y \equiv x \bmod M$ **を満たすとする．(これは** $dy \equiv n$ **を意味する.)**
2. $y \equiv u/v \bmod M$**，** $|u|, |v| < \sqrt{M/2}$ **なる整数** u**,** v **を探す.**
3. **もし** u**,** v **が存在すれば** $x = u/v$
4. **なければ，より大きい** M **に対して 1. が成り立つような** y **を探す.**

例 1: 線形方程式系の求解

$AX = B$ を解きたい (整数係数)

1. 素数 p_0 をとり，$AX = B \bmod p_0$ を解く

   一意的に解ければ，$AX = B$ の解も一意と分かる

2. $AX = B \bmod p_i$ が一意的に解けるような p_i をたくさん集める.

3. CRT により，$A\overline{X} = B \bmod M$ ($M = p_i$ の積) なる $\overline{X}$ を作る

4. $\overline{X}$ の各成分を整数-有理数変換

   結果が $AX = B$ を満たせばそれが解

p_i をどれくらい集めればいいか?

- **事実 1 — クラメルの公式**

  $x_i = \det(A_i)/\det(A)$

  (A_i : A の第 i 列を B に変えたもの)

- **事実 2 — アダマールの評価**

  $|\det(A)| < |a_1||a_2|\cdots|a_n|$

$\Rightarrow |\det(A)|, |\det(A_1)|, \ldots, |\det(A_n)| < m$

なる m に対し, $\Pi_i p_i > 2m^2$ となるように p_i をとればよい

(実際には incremental にやる)

例 2: 多項式因数分解

$x^2 + 3x + 2 = (x + 1)(x + 2)$

人間 : かけて 2, 足して 3 ...

$x^2 - 122500515x - 2267324677062 46 = ???$

こんなことは不可能

$\Rightarrow$ **有限体の出番**

Step 1 : 有限体上での因数分解

$f = x^2 - 122500515x - 226732467706246$

$f \equiv x^2 + 2 = (x+1)(x+2) \bmod 3$

二次式なら代入で分解できるが，一般には

Berlekamp アルゴリズム

有限体上の線形代数に帰着させる

p **が大きい場合には，確率的アルゴリズム(Cantor-Zassenhauss アルゴリズム) も有効**

Step 2 : 因子の係数の上限

f, g がモニックで, $g|f$ ならば, g の各係数の絶対値は $2^{\deg(f)}||f||_2$ を越えない.

($||f||_2$ は f の係数の絶対値の 2 乗和の平方根)

今の場合, $M = 906929870825117$ を越えない.

$3^{32} > M > 3^{31}$ である.

Step 3 : mod3^k に持ち上げ

Hensel の補題により,

$f \equiv g_1 h_1 \bmod 3$,

$\mathrm{GCD}(g_1, h_1) = 1$ **(**GF(3)$[x]$ **で)**

から

$f \equiv g_k h_k \bmod 3^k$

$g_k \equiv g_1 \bmod 3$, $h_k \equiv h_1 \bmod 3$

となる g_k**,** h_k **を一意に構成できる.**

Step 2 により $k = 33$ **まで行えばよい.**

($3^{33} > 2M$**)**

Step 4 : 整数への引き戻し

$\mathrm{mod} 3^{33}$ での各係数 $c \geq 0$ に対し,

$c < 3^{33}/2$ ならばそのまま,

$c > 3^{33}/2$ ならば $c - 3^{33}$

と変換する.

$g_{33} \to g = x + 1823719$

$h_{33} \to h = x - 124324234$

は確かに $f = gh$ を満たす（おわり）

一般にはもっと複雑

- モニックでない場合

  これはうまい方法あり

- mod$p$ での因子がたくさんある場合

  Hensel 構成の後，いくつかをかけて引き戻して試し割り

  $\Rightarrow$ ニセ因子が多い場合には組合せ的な困難

  $\Rightarrow$ 最近，この困難を克服するアルゴリズムが考案された（knapsack factorization）

多項式因数分解に関する他の話題

- **多変数多項式**

  変数に値を代入して，少ない変数の分解

  $\Rightarrow$ **Hensel 構成により変数を復活**

- **代数体上の多項式**

  $$x^6 + (\sqrt{2}+2)x^5 + (\sqrt{2}+1)x^4 + (2\sqrt{2}+4)x^3 - 2(\sqrt{2}+1)x^2 - (\sqrt{2}+2)x - 4\sqrt{2} - 6$$
  $$= (x + \sqrt{2} + 2)(x^2 + \sqrt{2} + 1)(x^3 - \sqrt{2})$$

  種々の方法あり

- $\overline{Q}$ **上の多項式（絶対既約分解）**

  代数体上の分解に帰着

方程式の求解

- 一変数の場合

  因数分解でなるべく低次に $\Rightarrow$ 2 分法，Newton 法 etc.

- 多変数，線形の場合

  Gauss 消去で原理的には可

  厳密求解 : 有限体近似も有効

- 多変数，高次の場合

  解の個数（有限 or 無限），解空間（多様体）の次元，解空間の分解をしたい

  $\Rightarrow$ 終結式，グレブナー基底

消去法とイデアル

消去法

与えられた方程式に，適当な多項式をかけて和を作り，
変数の少ない多項式を作ること

$$f_1 = 0, \ldots, f_r = 0 \Rightarrow a_1 f_1 + \cdots + a_r f_r = 0$$

一般化

$$\langle f_1, \ldots, f_r \rangle = \{a_1 f_1 + \ldots + a_r f_r \mid a_1, \ldots, a_r \in K[X]\}$$

をまるごと考える ($f_1, \ldots, f_r$ で生成されるイデアル)

$\Rightarrow$ この中に「都合がよい」多項式が含まれている

(数学ではよくこういうことをする)

終結式

連立方程式

$$f(x,y) = a_n(y)x^n + \cdots + a_0(y), \quad a_n \neq 0$$
$$g(x,y) = b_m(y)x^m + \cdots + b_0(y), \quad b_m \neq 0$$

から x **を消去したい.**

方針：イデアル $I = \langle f, g \rangle$ **から** $r(y)$ **を見つける**

$\Rightarrow$ $r(y) = 0$ **を満たす** $y_1, \ldots, y_l$ **を求め,**

$f(x, y_i) = 0$, $g(x, y_i) = 0$ $(i = 1, \ldots, l)$

をそれぞれ解けばよい.

終結式

$$S(f,g)=\begin{pmatrix}
a_n & a_{n-1} & \cdots & a_1 & a_0 & & \\
 & a_n & a_{n-1} & \cdots & a_1 & a_0 & \\
 & & \ddots & \ddots & & \ddots & \ddots \\
 & & & a_n & a_{n-1} & \cdots & a_1 & a_0 \\
b_m & b_{m-1} & \cdots & b_1 & b_0 & & \\
 & b_m & g_{m-1} & \cdots & b_1 & b_0 & \\
 & & \ddots & \ddots & & \ddots & \ddots \\
 & & & b_m & b_{m-1} & \cdots & b_1 & b_0
\end{pmatrix}$$

とおくと， $\mathrm{Res}_x(f,g)=\det(S(f,g))=uf+vg$

なる多項式 u**，** v **が存在する．すなわち** $\mathrm{Res}_x(f,g)$

は f**，** g **から** x **を消去した多項式**

$\mathrm{Res}_x(f,g)$**：** f**，** g **の** x **に関する終結式**

3 変数以上の場合

2 変数の終結式計算の繰り返しも適用できるが，一度に消去する方法もある

$\Rightarrow$ 多重多項式終結式（定義も計算も大変）

Gröbner（グレブナー）基底

イデアルの「よい」生成系

- ある多項式 f がイデアル I に入るかどうかを機械的にチェックできる.
- イデアルの生成系の一つであるため，零点集合は保たれる
- 消去イデアルを直接求められる

計算機の能力向上，アルゴリズムの進歩により実用化

項順序

体 K 上の n 変数多項式環を固定して考える.

定義

T を**係数 1 の単項式全体**とする. T の元を項と呼ぶ.

T の全順序 $<$ で次を満たすものを**項順序**とよぶ.

- 任意 の $t \in T$ に対し $1 \leq t$. (**1 が最低**)
- $t, s \in T$ が $t \leq s$ を満たすなら, 任意の $u \in T$ に対し $tu \leq su$. (**平行移動不変性**)

多項式 $f \neq 0$ に現れる項のうち $<$ に関して最高順序のものを $\mathrm{HT}(f)$ と書く. (head term)

Gröbner 基底

定義

I を多項式イデアルとし，$<$ を項順序とする．有限集合 $G \subset I$ が I の$<$ に関する **Gröbner 基底**であるとは，任意の 0 でない $f \in I$ に対し，ある $g \in G$ が存在して $\mathrm{HT}(g) \mid \mathrm{HT}(f)$ が成り立つときをいう．
(先頭を割り切る G の元があるということ)

消去イデアルの Gröbner 基底

$X = \{x_1, \ldots, x_n\}$, $Y \subset X$ とする．イデアル $I \subset K[X]$ に対し，$I_Y = I \cap K[Y]$ を (X, Y) に関する I の**消去イデアル**と呼ぶ．

消去イデアルは，消去で得られる全ての多項式を含む．

定理

イデアル $I \subset K[X]$ のある順序（消去順序）に関する Gröbner 基底G に対し，I_Y の Gröbner 基底を G_Y とすれば $G_Y = G \cap K[Y]$.

消去イデアルの例

$$
\begin{aligned}
f(x,y,z) &= xyz - 1 = 0 \\
g(x,y,z) &= xy^2 + y^2z + z^2x - 1 = 0 \\
h(x,y,z) &= x^3 + (3y+3z)x^2 + (3y^2+6zy+3z^2)x + y^3 + 3zy^2 \\
&\quad +3z^2y + z^3 - 1 = 0
\end{aligned}
$$

イデアル $\langle f, g, h\rangle$ の，$x > y > z$ なる辞書式順序に関する Gröbner 基底 G を計算すると，

消去イデアルの例

$$G = \{g_1(x,z), g_2(y,z), g_3(z)\}$$

$$\begin{aligned}
g_1(x,z) &= 1186553908x + 336578207z^{25} - 6268329005z^{22} \\
&\quad +16915340910z^{19} - 51793068785z^{16} + 7685641640z^{13} \\
&\quad -33454174976z^{10} + 1506632709z^{7} + 5249450545z^{4} \\
&\quad +4459728625z \\
g_2(y,z) &= 1186553908y + 437152001z^{25} - 7967132531z^{22} \\
&\quad +18848765946z^{19} - 60751368703z^{16} - 11889405544z^{13} \\
&\quad -55877240468z^{10} - 21725267873z^{7} - 9360059953z^{4} \\
&\quad -2126807385z \\
g_3(z) &= z^{27} - 18z^{24} + 39z^{21} - 129z^{18} - 59z^{15} - 132z^{12} - 77z^{9} \\
&\quad -30z^{6} - 6z^{3} - 1
\end{aligned}$$

消去イデアル： $I_{\{y,z\}} = \langle g_2, g_3 \rangle$, $I_{\{z\}} = \langle g_3 \rangle$

解の表示

$$g_1(x,z) = 1186553908(x - u_1(z))$$
$$g_2(y,z) = 1186553908(y - u_2(z))$$

とおくと，I の零点 $V(I)$ は

$$V(I) = \{(u_1(\alpha), u_2(\alpha), \alpha) \mid g_3(\alpha) = 0\}$$

と書ける．この場合の Gröbner 基底は，変数の消去だけではなく，解そのものも与えていることになる．これは特殊なことではなく，より一般な状況のもとで成り立つ．

Shape Lemma

定理(Shape Lemma)

イデアル I が根基イデアル，すなわち $\sqrt{I} = I$ のとき，適当な変数変換のもとで，I の辞書式順序による Gröbner 基底 G は

$$G = \{z_1 - g_1(z_n), \ldots, z_{n-1} - g_{n-1}(z_n), g_n(z_n)\}$$

となる．(Shape Base)

Shape Base は，代数方程式系の解を(ほぼ)与える

(あとは，一変数方程式 $g_n(x_n) = 0$ を解けばよい.)

計算法 I：Buchberger アルゴリズム

Input: 多項式集合 F，項順序 $<$

Output: $\langle F \rangle$ の $<$ に関するグレブナー基底 G

$D \leftarrow \{\{u, v\} \mid u, v \in F, u \neq v\}$; $G \leftarrow F$

While $D \neq \emptyset$

$\{u, v\} \leftarrow D$ の要素; $D \leftarrow D \setminus \{\{u, v\}\}$

$\mathrm{Spoly}(u, v) \xrightarrow[G]{*} r$ (r は G に関して正規形)

if $r \neq 0$ **then** $D \leftarrow D \cup \{\{h, r\} \mid h \in G\}$; $G \leftarrow G \cup \{r\}$

end while

return G

—— 一種のガウス消去である

Spoly と $\xrightarrow[G]{*}$

- **Spoly**(f, g)

  なるべく次数の低い単項式 a**,** b **により** $af - bg$ **を作り，** af**,** bg **の先頭が打ち消しあうようにする.**

- $h \xrightarrow[G]{*} r$

  $h \to (h - a_1 g_1) \to (h - a_1 g_1 - a_2 g_2) \to \cdots$

  ($g_i \in G$**,** a_i **は単項式) を繰り返して，** h **の項を消す剰余算── 引けなくなったら終了 (正規形とよぶ)**

計算法 II : 項順序変換

例 : FGLM アルゴリズム

I : 0 次元イデアル (方程式の解が有限個)

G_0 : I の, $<_0$ に関するグレブナー基底

$\Rightarrow$ 項順序 $<_1$ に関するグレブナー基底 G_1 を計算

原理

G_0 がある $\Rightarrow$ 多項式 h が $h \in I$ かどうか判定可

順序が下の単項式から順に, それらの線形和で I に入るものがあるかどうか調べる (未定係数法)

$\Rightarrow$ あったらそれが G_1 の要素

これを繰り返せば, G_1 が計算できる

FGLM の動作

$<_1$ で，単項式が $1 = t_0 <_1 t_1 <_1 \ldots$ と並んでいる
とする．

$$g = t_k + a_{k-1}t_{k-1} + \cdots + a_0 t_0 \in I$$

となる係数 $a_{k-1}, \ldots, a_0$ があれば，g が G_1 の要素

$\Rightarrow$ 線形代数に帰着

繰り返せばいずれ終る（I が 0 次元だから）

計算上の困難

- 終結式

  多項式要素の行列の行列式 $\Rightarrow$ 難しい

  互除法類似の方法で計算できるが，それでも大変

- グレブナー基底

  互除法類似なので，互除法と中間膨張

  アルゴリズムの不定性による中間膨張

- 項順序変換

  線形方程式系を数多く解く $\Rightarrow$ やはり係数膨張

$\Rightarrow$ **有限体近似で緩和可能!!**

(他にも種々の効率化が考案されている)

どれぐらい差があるか試して見る

Risa/Asir vs Maple

- Maple

  今年から，サイトライセンス取得

  汎用数学ソフトウェア

  整数演算: gmp（自前のを捨てて，フリーソフトを採用)

- Risa/Asir

  神戸大で開発，配布している数学ソフトウェア

  整数演算: 自前（gmp より遅い)

(相手が売り物だから，失礼には当たらないでしょう)

ランダム生成した方程式

$$
\begin{aligned}
f_1 &= (u_1^3 - 3u_1^2)u_0 - 3u_1^3 + 4u_1 - 5 \\
f_2 &= (3u_1^2 + 1)u_0^3 + u_0 + 4u_4u_1 + 3u_2 + 2 \\
f_3 &= (-3u_3u_1 - 4)u_0^2 + (-2u_1 + 1)u_0 + 4u_2^2 + 2 \\
f_4 &= -3u_0^2 + ((-2u_2 - 2u_3^2 - 2)u_1^2 - u_3u_2u_1)u_0 + u_2u_1^2 + u_1 \\
f_5 &= u_0^3 + (-2u_1 - 7)u_0 + 3u_2u_1^2 - u_2u_1 - 2
\end{aligned}
$$

(5 変数，方程式が 5 個)

DRL 順序での Gröbner 基底計算

- Maple

  1000 秒, 360MB で中断

- Risa/Asir

  4 秒 (Buchberger + trace lifting)

  2 秒 (F_4 アルゴリズム)

- Singular (有名な数学ソフトウェア; free)

  6 秒

結論 : Maple の Gröbner 基底計算は使えない

少しやさしくする

$$
\begin{aligned}
f_1 &= (u_1^2 - 3u_1^2)u_0 - 3u_1^3 + 4u_1 - 5 \\
f_2 &= (3u_1^2 + 1)u_0^2 + u_0 + 4u_4u_1 + 3u_2 + 2 \\
f_3 &= (-3u_3u_1 - 4)u_0^2 + (-2u_1 + 1)u_0 + 4u_2^2 + 2 \\
f_4 &= -3u_0^2 + ((-2u_2 - 2u_3^2 - 2)u_1^2 - u_3u_2u_1)u_0 + u_2u_1^2 + u_1 \\
f_5 &= u_0^2 + (-2u_1 - 7)u_0 + 3u_2u_1^2 - u_2u_1 - 2
\end{aligned}
$$

DRL 順序 Gröbner 基底 $\rightarrow$ FGLM

- Maple

  7 秒 (DRL) + 30 秒 (FGLM)

  FGLM を実行するために，マニュアルからプログラムを抜き出して改変する必要があった．(あまりにユーザをバカにしているのでは?)

- Risa/Asir

  0.5 秒 + 1.1 秒 (FGLM)

比べてみた結果

- Maple について

  比較以前の問題であった (この問題の場合, solve()
  で解けるが, もう少し複雑な問題になるとだめ)

- 有限体近似の効果

  Singular と比較してみる

  $\Rightarrow$ 結果の係数が小さいうちは, 有限体を用いなくても OK だが, 結果の係数が大きくなると, 有限体を使うほうが高速化する.

  有限体近似 : 結果の大きさに応じた計算の手間

おわりに

- 研究の現状

  そうとう高度な数学がアルゴリズム化されつつある

  (本職の)数学者が参入
- ソフトウェア

  商用，フリーとも，数多くのソフトウェアが開発されている

  特定分野，特定機能なら，それほど敷居が高くなく，参入可能
- OpenXM もよろしく

  http://www.math.kobe-u.ac.jp/OpenXM