解　説　数式処理の最近の研究動向

2．方程式を解く

Solving Algebraic Equations by Masayuki NORO（High Performance Computing Research Center, FUJITSU LABORATORIES LTD）.

野呂　正行[1]

1（株）富士通研究所HPC研究センター

1．はじめに

方程式の求解はさまざまな問題の解決に現れる．現状では数値的方法による方程式の求解が広く用いられているが，解の存在，解の個数（有限個か否か）といった方程式の性質の検証，解の精度保証，あるいはパラメタを含んだままの計算を行うことは，数値的な方法によっては一般に困難である．しかし，制約条件が多項式で与えられる代数方程式に関しては，代数的方法の進歩により，ある程度の規模の問題に対し，このような要請に答えられるようになってきた．ここでは，代数的方法，特にグレブナ基底による方程式の求解について述べる．特に解が有限個の方程式を解く最近の成果について解説する．

2．線形方程式を解く

線形方程式を解くことは数学的には明らかなことであるが，計算機上で高効率，高精度に線形方程式を求解しようと思えば，周知のとおりさまざまなテクニックが必要となる．実際には，このようなテクニックは名の通った数値計算ライブラリにはすべて採り入れられているであろうから，マニュアルどおりにサブルーチンを呼びさえすれば，楽天的な人は，得られた結果が解だと信ずるかもしれない．あるいは少し用心深い人なら，結果を元の方程式に代入してみて“ほとんど”0になっていることを確認して安心するかもしれない．しかし，代入して真に0になることは解であることの必要十分条件であるが，そうでない場合，真の解と，代入した“解”が近いという保証は一般にはない．解の精度を直接保証する方法として，計算精度を上げることが考えられるが，そのような単純な方法では，方程式の規模が大きくなると，計算に必要な精度が大きくなってたちまち行き詰まる．計算機代数（数式処理）は，多倍長演算に代表される誤差なしの演算のみならず，モジュラ計算（整数を法とする演算）などの応用によりこのような問題に対しても効率よく正確な解を与えることができる．

次のアルゴリズムは，整数係数の線形方程式$MX=B$の解をモジュラ計算により求める．

$det(M)$を割らない素数pを選び，法pでの解から出発して，法p^nでの解（p-進近似）を構成し，適当な段階で有理数に“引き戻し”，元の方程式に代入する，という操作で解に到達できる．

$\mathbf{Z}$を整数全体の集合とし，

$$\phi_p : \mathbf{Z} \to GF(p) = \mathbf{Z}/(p) \qquad (1)$$

を，素数pで割った余りを対応させる写像とする（$GF(p)$は有限体）．また，ϕ_p^{-1}は$(-p/2, p/2]$から選んだ逆像とする．

アルゴリズム 2.1　p-進近似による線形方程式の求解[1), 2)]

入力：$n\times n$整数行列M，$n\times 1$整数行列B
　　$\phi_p(det(M))\neq 0$なる素数
出力：$MX=B$なる$n\times 1$行列X

```
INV ← φ_p(M)^{-1} ; c ← B ; x ← 0 ; q ← 1
count ← 0
do {
    t ← φ_p^{-1}(INV φ_p(c))
    x ← x+qt ; c ← (c−Mt)/p ; q ← qp
    count ← count+1 mod MaxCount
    ((c−Mt)/pは整除.)
    if count=0 then {
        X ← inttorat(x, q)
        if X≠nil and MX=B then return X
    }
}
```

ここで，$inttorat(x, q)$は，整数x, qに対し

$$bx \equiv a \bmod q \text{かつ} |a|, |b| \le \sqrt{\frac{q}{2}} \qquad (2)$$

なる有理数a/bが存在すればその値を返す関数で，ユークリッドの互除法類似の方法で計算できる．**MaxCount**回ループを回るごとに各成分に対しこの変

換を行い，その結果が方程式を満たすかどうか代入して調べる．

方程式の係数がすべて整数だから，解の各成分は有理数である．したがって，$\sqrt{\frac{D^n}{2}}$ が解の各成分の分母分子の絶対値を超えるようなnに対し，n回ループを回った後のxの各成分を有理数に引き戻せば元の方程式の解となる．

解の各成分の分母分子は係数行列から作られる行列式として書けるから，nとしてどれだけの値をとれば十分かを具体的に与えることができる．**MaxCount**の値をその値にとれば1回目の変換結果が解となるようにできる．しかし，この値を小さい値にとっておくことで，解の各成分の分母分子が小さい場合に少ないステップ数で解に到達することができる．すなわち，計算の手間が表現の大きさに依存して決まるという利点を持つ．

3. 非線形方程式とグレブナ基底

一変数の高次方程式に関してはさまざまな解法が知られているが，多変数の高次連立代数方程式に関しては，ほとんどの数値的方法はニュートン法を基礎に置くと思われる．一般に，ニュートン法による求解は高速だが次のような短所を持つ．

- 解の精度が，解を代入した値でしか計れない
- 1つの初期値から解が1つしか求まらない
- 全解を求めるためには，それぞれに対応する初期値を与える必要がある
- 重解，近接解などの悪条件に弱い

連立代数方程式を代数的に解く場合の基本は消去法である．古典的には終結式の計算を繰り返すことにより，より変数の少ない方程式を生成していくことができる．そして，ある場合には一変数方程式が得られる．しかし，この方法も次のような短所を持つ．

- 消去後の式が大きくなる
- 消去で得られた式は必要条件に過ぎない
- 消去された変数の値を求めることは容易でない

より進んだ消去法として一般消去法とよばれる方法があるが，これに関しては本誌特集の記事[5]を参照されたい．

Buchbergerにより提案され近年盛んに研究されてきたグレブナ基底は，消去法をシステマティックに行うことを可能にする．グレブナ基底に関する詳細は教科書[3),4)]に譲り，ここでは最小限の解説に留める．他方面への応用も視野に入れたグレブナ基底に関する教科書[6]，解説[7]および本誌特集の記事[8]も入手しやすく有用であろう．

グレブナ基底を理解するために，まずイデアルという概念を理解する必要がある．以下，体K上の多項式環$R=K[x_1,\cdots,x_n]$（Kを係数とする$x_1,\cdots,x_n$の多項式全体）を固定して考える．

定義 3.1 $I\subset R$がイデアルであるとは，

1. Iは加法について閉じている．
2. $a\in R,\ f\in I\Rightarrow af\in I$

を満たす時をいう．

この定義では抽象的すぎるが，多項式環に関してはイデアルはすべてある方程式に対応するイデアルとして書ける．

定義 3.2 $f_1,\cdots,f_m\in R$に対し，
$E=\{f_1(x_1,\cdots,x_n)=0,\cdots,f_m(x_1,\cdots,x_n)=0\}$ なる方程式を考える．このとき，

$$I(E)=\{\sum_{i=1}^{m}g_if_i\mid g_i\in R\} \qquad (3)$$

をEのイデアルと呼ぶ．（これは一般的な呼称ではない．通常は，$f_1,\cdots,f_m$で生成されるイデアルと呼ばれる．）

このイデアルに属する多項式はすべて，元の方程式の解を零点に持つ．また，消去法の操作を考えれば，イデアルの中には，消去法で得られるあらゆる多項式が含まれている．グレブナ基底はイデアルから取り出した有限個の多項式の組であり，元のイデアルを生成する．すなわち，解を考える上では元の方程式と同値な方程式である．それがどのような性質を持つかを述べる前に，いくつか言葉の定義をしておく．0以上の整数全体の集合を$\mathbf{N}$と書く．

定義 3.3 $T=\{x_1^{i_1}\cdots x_n^{i_n}\mid i_1,\cdots,i_n\in\mathbf{N}\}$ とし，Tの元を**項（term）**と呼ぶ．このとき，Tにおける全順序$<$が*admissible*であるとは，$<$が次を満たすことをいう．

1. 任意のtに対し，$1\le t$
2. 任意のt,s,uに対し，$t\le s$ならば$tu\le su$

多項式fに現れる項のうち，$<$で順序最大の項を**頭項（head term）**と呼び，$HT_<(f)$と書く．

このような順序として，**辞書式順序**（冪指数を順に比較する），**全次数辞書式順序**（まず全次数で比較し，等しい場合に辞書式で比較），**全次数逆辞書式順序**（全次数で等しい場合に逆辞書式で比較）などがある．admissibleな全順序$<$を用いて，イデアルIのグレブナ基底は次のように定義される．

定義 3.4 GをイデアルIの有限部分集合とする．任意の$f\in I$に対し$HT_<(g)\mid HT_<(f)$なる$g\in G$が存在するとき，Gは$<$に関する**グレブナ基底（Gröbner basis）**であるという．

次に定義する演算は，多項式による頭項消去である．これは単項式に注目した除算である．

定義 3.5 f, gを多項式とし，fのある項tを$HT_<(g)$が割り切るとき，ある単項式mがあって，$h=f-mg$に

tが現れないようにできる．この操作を**頭項消去**（**head term reduction**）と呼び$f \underset{g}{\rightarrow} h$と書く．集合$G$のいずれかの元による頭項消去を0回以上繰り返すとき$\overset{*}{\underset{G}{\rightarrow}}$なる記号を用いる．

定義およびadmissibleな項順序の性質から，グレブナ基底に関する次の重要な性質が導かれる．

定理 3.6 GをイデアルIのグレブナ基底とする．このとき，多項式fに対し，

$$f \in I \Leftrightarrow f \overset{*}{\underset{G}{\rightarrow}} 0 \qquad (4)$$

右辺は頭項消去の仕方によらない．

すなわち，ある多項式が元の方程式の左辺の多項式係数の積和で得られるか否かをグレブナ基底による頭項消去のみで判定することができる．

解が有限個の方程式に対応するイデアルを0次元と呼ぶが，イデアルの0次元性は，グレブナ基底により次のように簡明に記述される．

定理 3.7 GをイデアルIのグレブナ基底とする．このとき，

Iが0次元，任意のi（$1 \leq i \leq n$）に対し，$g \in G$，$m \in \mathbf{N}$が存在して$HT_<(g) = x_i^m$

次のアルゴリズムは，定理3.6の応用としてBuchberger自身により与えられているもので，n変数の連立代数方程式から$n-1$変数を消去した1変数の方程式を求める方法を示している．

アルゴリズム 3.8 最小多項式の計算

入力：イデアルIのグレブナ基底G；変数x

出力：xの満たす方程式$m(x)=0$

$n \leftarrow 1$

do {

if $\exists\, a_0, \cdots, \exists\, a_{n-1}$ *such that*

$m(x) = x^n + a_{n-1}x^{n-1} + \cdots + a_0 \in I$ (*)

then return $m(x)$

$n \leftarrow n+1$

}

このアルゴリズムが停止するためには，あるnに対して，(*)を満たす一変数多項式が存在することが必要だが，方程式の解が有限個ならば存在が保証される．このアルゴリズムの出力$m(x)$を，xの**最小多項式**（**minimal polynomial**）と呼ぶ．

注目すべきは，定理 3.6，3.7，アルゴリズム 3.8が，グレブナ基底Gがどの順序に関するものかに無関係に成立する点である．アルゴリズム 3.8においては$m(x) \in I$なる条件は，$m(x) \overset{*}{\underset{G}{\rightarrow}} 0$と同値である．これより，$m(x) \in I$となるための条件として，未定係数$\{a_i\}$に対する線形方程式が得られる．これが解を持てば，$m(x) \in I$なる$m(x)$が得られることになる．

$m(x)=0$を解けば解におけるxの値の候補が求まる．各変数に対してこのような一変数多項式が得られれば，それらを解いて，全体として元の方程式を満たす組を集めればそれが解全体となる．あるいは，$m(x)=0$の各解に対し，その値を元の方程式に代入して，変数が1つ少ない方程式を作って解くという方法も考えられる．しかし，グレブナ基底を応用して連立代数方程式を解く方法としては，これらではいまだ不十分である．次章でより進んだ方法を紹介する．

4. 解が有限個の方程式

前章で，解が有限個の方程式に対し，変数消去により一変数方程式を得る方法を示した．余分な解が現れることを許せば，これは終結式の計算を繰り返すことによっても可能であるが，終結式の計算にかかるコストおよび，他の変数の値をどう求めるかが問題となる．実は，解が有限個の方程式に関しては，次のような事実が成り立つ．

事実 4.1 解が有限個の方程式に対するイデアルIの辞書式順序グレブナ基底は，変数に対する適当な線形変換により“ほとんどの場合”

$\{m(x_n), x_1 - g_1(x_n), \cdots, x_{n-1} - g_{n-1}(x_n)\}$ ･･･（SB）

という形になる．

（SB）の形のグレブナ基底はshape basisと呼ばれる．“ほとんど”の意味はここでは説明しないが，グレブナ基底がこの形になれば，解が

$$\{(g_1(x_n), \cdots, g_{n-1}(x_n), x_n) \mid m(x_n)=0\} \qquad (5)$$

で与えられることは明らかである．すなわち，一変数方程式$m(x_n)=0$の解さえ求まれば，その他の変数の値は単なる代入により求まる．よって，原理的には任意精度（この場合の精度は，真の解との距離を表す）で近似解が求められる．

しかし，ここに落し穴が1つある．実際にさまざまな方程式に対し，上の形でグレブナ基底を計算してみると，mの係数の大きさに比較してg_iの係数がきわめて大きくなる場合が多い．アルゴリズムにもよるが，この事実は，グレブナ基底の計算時間の増大という結果につながる場合が多い．この困難を克服できる1つの方法として，**解の有理式表現**（**Rational Univariate Representation; RUR**）が考案された[9]．この方法では，解の表現として，

$\{m, m'x_1 - h_1, \cdots, m'x_{n-1} - h_{n-1}\}$ ･･･（RUR）（$m, h_1, \cdots, h_{n-1} \in K[x_n]$，$m'$は$m$の$x_n$に関する微分）

を用いる．この表現は，$GCD(m, m')=1$すなわちmが無平方の場合，（SB）と同等の解表現となる．注目すべき点は，

(RUR) におけるh_iの係数が，g_iの係数と比べて非常に小さい場合が多いということである．実験によれば，h_iの係数の大きさはmの係数の大きさと同程度である場合が多い．具体的な例でその違いを見てみよう．次の方程式はランダムに生成したもので，解は有限個である．

$$\begin{cases} (u_2+1)u_0^3 - u_0 - u_2u_1^2 - 3u_1 = 0 & (6) \\ (u_1^2-1)u_0^2 + u_1u_0 + u_2^2 + u_2 + 2 = 0 & (7) \\ -2u_0^4 - 2u_1^2u_0 + (u_2^2+1)u_1u_0 + 2u_1 + 2u_2 = 0 & (8) \end{cases}$$

この方程式に対し，u_2の最小多項式の係数はたかだか41ビットで，(RUR) における分子の係数のもたかだか47ビットだが，(SB) においてはu_0, u_1を表現する多項式の係数が最大615ビットに達する．

5. 計算機上での実現

グレブナ基底に代表される代数的方法を計算機上で実現しようとする場合，時間空間計算量が最大の問題となる．グレブナ基底について言えば，Buchbergerによる最初のアルゴリズムの提示の後，30年近くの間改良，効率化の努力が続けられてきた．それらをまとめると，およそ次のように分類できる．

- 不必要な計算を省く
- 計算順序の最適化
- 基底変換 (change of ordering)
- モジュラ計算の応用
- 並列化

これらを採り入れることにより，現実的に計算可能な対象が大きく広がる．これらのうち基底変換以降のものはここ数年のうちに大きく発展したものでまだ実験システム上にしか実現されていない．ここでは，これらの改良を用いてどの程度の問題を実際に解くことができるかについて，筆者らが開発中の計算機代数システムRisa/Asir[11]での実験結果を示す．方程式は，ベンチマークとしてよく用いられるK_n, C_nである．

K_nは$n+1$変数，C_nはn変数の方程式で次のように定義される．

$$K_n: \{u_l - \sum_{i=-n}^{n} u_iu_{l-i} = 0 \ (l=0,\cdots n-1), \ \sum_{i=-n}^{n} u_l - 1 = 0\}$$

$u_l = u_{-l}$, $u_l = 0$ ($|l| > n$)，変数はu_i ($0 \le i \le n$)．(9)

$$C_n: \{f_1 = 0, \cdots, f_n = 0\}$$

$$f_k = \sum_{i=1}^{n}\prod_{j=i}^{k+j-1} c_{j \bmod n} - \delta_{k,n} \ (\delta_{i,i}=1, \ \delta_{i,j}=0 \ (i \neq j)).$$ (10)

$$F_6 = z - (c_1 + 2c_2 - 3c_3 - 2c_4 - 4c_5 - c_6)$$

表-1 基底変換によるグレブナ基底，RURの計算

		k_5	k_6	k_7	k_8	C_6	C_7
解の個数		32	64	128	256	156	924
計算時間 (秒)	*SB*	5.8	263	10160	—	56	—
	RUR	1.1	12	211	5554	3.3	2188
係数の最大ビット長	*SB*	1118	6001	34738	—	3933	—
	RUR	111	262	637	1256	415	4094

$$F_7 = z - (c_1 + 2c_2 - 6c_3 + 4c_4 - 10c_5 + 6c_6 - 9c_7)$$

C_n ($n=6, 7$) に関しては，それぞれF_nを追加して，zに関してshape basisとなるようにしてある．用いた計算機はPentiumII-300MHz PC (メモリ512MB) である．

これらの計算はRisa/Asir上では基底変換により実行されている．表には出発点となる全次数逆辞書式順序でのグレブナ基底の計算時間は含まれない．基底変換は，アルゴリズム3.8で用いた方法を，グレブナ基底の他の元に対しても同様に用いるもので，Buchbergerによりすでに提案され，Faugèreら[10]による改良版が知られている．この方法を正直にインプリメントすると，単項式を1つずつ追加するごとに有理数体上でのガウス消去に相当する計算を行うことになるが，実際に大きな行列に対して同様の操作を行ってみればわかるように，途中で現れる行列の成分が，解の成分に比べてしばしば大きくなり，計算時間も増大する．

Risa/Asir上では有限体上でのグレブナ基底計算によりグレブナ基底の形を推測し，その形に対する未定係数を直接求めるというアルゴリズム[2]が用いられている．その推測が正しければ，グレブナ基底の各元に対し1個の線形方程式を解くだけでよい．さらに，線形方程式をアルゴリズム2.1で解くことにより，解となるグレブナ基底の元の係数の大きさに応じた計算の手間で済むことになる．このことが，RURの計算がグレブナ基底の計算に比べて高速な理由である．

6. 最近の状況

基底変換を用いる場合，全次数逆辞書式順序でのグレブナ基底をいかに効率よく計算できるかが問題となる．あとで触れる新方法によらなければ，一般の入力に対しては，Buchbergerアルゴリズムにより計算せざるを得ない．一般にはこのアルゴリズムは組合せ的な困難を生じるが，解が有限個，あるいはそれに近い方程式では，中間係数膨張が問題になる場合も多い．筆者らは最近，途中生ずる多項式の係数の整数共通因子を効率よく取り除く方法を提案し，数学上のある問題から生ずる方程式の求解に応用した[12]．この方程式

McKay[12] は4変数だが方程式は18ある過剰決定系で，項数が最大1183の方程式を含む．グレブナ基底の計算時間はP6-200MHz PCで約5日であった．

ごく最近，FaugèreによりアルゴリズムF_4が提案された[13]．その詳細はまだ明らかにされていないが，一般の入力からのグレブナ基底の構成を，巨大だが疎な連立一次方程式の求解に帰着させるもので，疎な連立一次方程式の高速求解法を用いて，Buchbergerアルゴリズムに比較して数十倍から数万倍高速にグレブナ基底が計算できることが報告されている．線形方程式への帰着自体は古くから知られており，いかに高速に解くかがカギとなる．報告によれば，有理数係数の場合には“multi-modular”による方法で高速化したとのことであるが，これが中国剰余定理を用いるという意味なら，得られた結果が実際に入力イデアルのグレブナ基底になっていることを示す困難が残ると考えられるため，アルゴリズムの詳細の公表が待たれている．ちなみに，Faugère自身の計算によれば，前記*McKay*のグレブナ基底計算に要した時間は同じCPU上で54秒とのことである．

7. おわりに

本稿では詳しく述べなかったが，グレブナ基底の理論は体を係数とする多項式環に対して構成されているため，係数体がいくつかの変数を含む有理関数体の場合にも成り立つ．これはパラメタを含んだまま方程式を解くことができることを意味する．実際には，有理数係数の場合には生じなかった問題（多変数多項式のGCDなど）が生じるため計算がより困難となるが，パラメタを含む解を求め，パラメタの値を最適化するなどの，数値的解法では困難な操作を行うこともできる．

最後に触れたF_4アルゴリズムによってもなお，計算機上で代数的方法により解かれ得る方程式の規模は数値的方法に比較してまだまだ小さい．しかし，たとえばC_n，*McKay*のように数学的な背景を持つ代数方程式は，解が何らかの意味で小さい表現を持つ場合が多く，現在知られている方法でもかなりの規模の方程式を解くことができる．また工学における問題においても，計算効率だけでは計れない解の品質を保証するために，代数的方法がもっと利用されるようになることを筆者は期待している．

参考文献

1) Gregory, R. T. and Krishnamurthy, E. V.: Methods and Applications of Error-Free Computation, Springer-Verlag, New York (1984).
2) Noro, M. and Yokoyama, K.: New Methods for the Change of Ordering in Gröbner Basis Computation, FUJITSU ISIS Research Report ISIS-RR-95-8E (1995). PostScriptファイルは文献11) に示されるディレクトリのpaper/changeoforder.ps.gz.
3) Becker, T. and Weispfenning, V.: Gröbner Bases, Graduate Texts in Math. 141, Springer-Verlag (1993).
4) 大阿久俊則: グレブナ基底と線型偏微分方程式系（計算代数解析入門），上智大学数学講究録, No.38 (1994).
5) 小林英恒: 多変数連立代数方程式の解法, 大特集: 数式処理, 情報処理, Vol.27, No.4, pp.414-421 (Apr. 1986).
6) 佐々木建昭他: 計算代数と計算幾何, 岩波講座応用数学［方法9］(1993).
7) 白柳　潔: グレブナ基底入門, システム/制御/情報, Vol.39, No. 5, pp.225-232 (1995).
8) 佐々木建昭, 古川昭夫: コンピュータ環論, 大特集: 数式処理, 情報処理, Vol.27, No.4, pp.404-413 (Apr. 1986).
9) Alonso, M. et al.: Zeros, Multiplicities and Idempotents for Zero Dimensional Systems, Algorithms in Algebraic Geometry and Applications, Birkhäuser, pp.1-16 (1996).
10) Faugère, et al.: Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering, J. Symb. Comp. 16/4, pp.329-344 (1993).
11) Noro, M. and Takeshima, T.: Risa/Asir－A Computer Algebra System. Proc. ISSAC '92, pp.387-396 (1992). さまざまな機種/OS用のバイナリがftp://endeavor.fujitsu.co.jp/pub/isis/asirから入手可能.
12) Noro, M. and McKay, J.: Computation of Replicable Functions by Risa/Asir, Proc. PASCO '97, pp.130-138 (1997). 方程式は文献11) に示されるディレクトリのexamples/odd_order.
13) Faugère, J. C.: A New Efficient Algorithm for Computing Gröbner Basis (F_4), Preprint (1997).

(平成9年11月19日受付)

野呂　正行（正会員）

1983年東京大学理学部数学科卒業．1985年同大学院理学系研究科修士課程（数学専攻）修了．同年富士通（株）入社．国際情報社会科学研究所勤務．現在，（株）富士通研究所HPC研究センター主任研究員．入社以来，計算機代数アルゴリズム，システムの研究開発に従事．数式処理学会会員．

e-mail:noro@para.flab.fujitsu.co.jp.