数理解析研究所講究録
1085巻 1999年 181-184



# 線形多項式方程式の解法プログラムの作成

電子技術総合研究所 元吉文男 (Fumio MOTOYOSHI)*

## 1 動機

数式処理で問題を解こうとしたときに、多項式を決定する問題に帰着させることができるものが多い。たとえば、多項式の剰余計算、拡張ユークリッド法、有理式の不定積分などの場合である。このうち、最初の二つには専用のアルゴリズムがあり、効率的に多項式を決定できるが、最後の例はいまのところ、多項式の各係数を未知として線形方程式を作成して解く方法に勝るものはない。

そこで、一般に、多項式を未知とする方程式を解くプログラムを作成しておことによって、以上のような問題を汎用に解くことができる。ただし、汎用プログラムでは効率の面では最良のものとはいえない場合が生じるが、1) 個別のアルゴリズム化が面倒である場合、2) 効率は二の次にして、ともかく答が欲しい場合、3) 効率的な解法アルゴリズムが知られていない場合などの状況において、有用であると考えられる。

一般的解法プログラムが理想であるが、ここではとりあえず、変数は一つで、その未知多項式に関して線形のものに限定してプログラムを作成したので紹介する。

## 2 構文と例

まず、作成したプログラムの記述法と使用例を紹介する。未知方程式を解かせるためには、以下の構文を使用する。

```
<多項式>, polysolve(<未知 1>[ , <次数 1>][,<未知 i>, <次数 i>]*);
```

未知多項式は何個あってもよく、未知多項式とそのとり得る最大の次数を対にしたものの連続を `polysolve` という関数名の引数にして、解くべき多項式の後に並べる。なお、最大次数が不明のときはそれを省略してもよいが、次数が不明の未知多項式は高々一個であるものとする。¡未知 i¿は変数名であり、上の式は

$$< \text{多項式} >= 0$$

*moto@etl.go.jp

という方程式を¡未知 i¿について解くことを指示している。

### 例

以下の例で (Gi) で始まる行が入力であり、(Hi) で始まる行が結果である。

最初は、$x^5$ を $x^2-1$ で割ったときの商 (Q) と余り (R) を求める例であり、余り (R) の次数が高々1であると指示している。

```
(G1) x^5 - Q(x)*(x^2+1) - R(x), polysolve(Q, R, 1);
(H1) {Q(x) = x^3 - x, R(x) = x}
```

次は、拡張ユークリッド法によって $x^3+1$ と $x^2+1$ の線形結合で 1 にする係数 (G, H) を求める問題である。

```
(G2) G(x)*(x^3+1) + H(x)*(x^2+1) - 1, polysolve(G, H, 2);
(H2) {G(x) = 1/2 x + 1/2, H(x) =  - 1/2 x^2 - 1/2 x + 1/2}
```

最後の例は、多項式の引数として $x$ 以外のものが入っている場合にも、この解法プログラムが適用できることを示すものである。

```
(G3) x^4 - 3*x + 4 - P(x^2) - P(x-1), polysolve(P);
(H3) {P(x) = x^2 - x + 1}
```

## 3 実現法

線形の多項式方程式の解法プログラムは、次の 2 つのステップからなっている。求める多項式を $P_i(1 \leq i \leq m)$ とする。このとき最大次数が未定のものがあればそれを $P_1$ とする。多項式の変数を $x$ とする。

1. $P_1$ の最大次数 $n_1$ が未知の場合にはそれを決定する。
2. 未知係数を決定する。

### 最大次数の決定

$n_1$ を変数として、それを決定するために以下の操作を行なう。元の方程式の $x$ の最高次の項の候補となる項を計算する。

$P_1$ 以外の未知多項式ではその次数が確定しているので、その項の次数は整数である。$P_1$ が関係する式から出る項の次数は、$kn_1+b_k$ である。ここで $k, b_k \in Z,\ k \geq 0$ である。ま

た、これらのうちで同じ $k$ を持つものでは、$b_k$ が最大のものが最高次の項に成り得るので、結局、最高次の候補となる式だけを集めた項は

$$\sum_{k=0}^{l}(c_k x^{kn_1+b_k})$$

となる。元の方程式が 0 に等しいことから、これらの項が消えなければならず、そのためには次のいずれかが成り立つ必要がある ($0 \leq k \leq l$ のいずれかについて)。

$$c_k = 0$$

$$kn_1 + b_k = jn_1 + b_j, \quad (k \neq j)$$

そこで、これらのそれぞれを満たす $n_1$ のうち最大のものを $P_1$ の最大次数だとすればよいことがわかる。

### 未知係数の決定

この段階では、未知多項式の最大次数がわかっているので、各未知多項式の係数を変数 $c_{ij}$、すなわち

$$P_i(x) = \sum_{j=0}^{n_j} c_{ij} x^j, \quad 1 \leq i \leq m$$

として、これを元の方程式に代入する。得られる式を $x$ について整理すると

$$\sum_{k=0}^{n} a_k x^k$$

という形になる。ここで、元の式は線形であるとしたので、各 $a_j$ は $c_{ij}$ の線形結合になっている。そこで

$$a_k = 0 \quad 0 \leq k \leq n$$

という $N(=\sum_{j=1}^{m}(n_j+1))$ 変数の線形方程式を解くことによって、すべての $c_{ij}$ が決定できる。このときに、$N$ と $n+1$ の大小関係によって、解が存在しない場合、ちょうど 1 組だけ存在する場合、解空間が 1 次元以上の場合があり得るが、答えの組の集合として全体の解を表現することによって、いずれの場合の結果も得られる (解空間が 1 次元以上の場合にはパラメータを使用して表現する)。

## 4 おわりに

一変数の多項式を未知とする線形方程式の解法プログラムについて述べたが、これは、Java で作成している数式処理システム上に実装されている。現状では、多項式の係数は整数

の場合を扱っているが、プログラムでは、係数の演算はそのクラス (上の場合では BigInteger class) のメソッドを利用するように作成してあるため、係数を拡張することは容易である。現在は係数のクラスを再設計中であり、係数クラスが環か体かのサブクラスによって、使用するアルゴリズムを変更するように予定している。

汎用の多項式解法プログラムは、とりあえず解を求めたい場合、効率的アルゴリズムが不明の場合などに有用であると思われるため、係数の拡張、あるいは非線形への拡張などが重要であると思われる。