数理解析研究所講究録 1295 巻 2002 年 203-208



# 安定化した Wu's method のロボット制御への応用

白石 啓一　　甲斐 博　　野田 松太郎
詫間電波高専*　愛媛大学 工学部†　愛媛大学 工学部‡

## 1 はじめに

制御・計算機援用設計 (CAD) 等多くの分野において連立代数方程式の解を求める問題が現れる。本稿では Wu's method[1, 2] を用いて解く方法を検討する。

工学上の問題では浮動小数係数への対応が重要な課題である。そこで、浮動小数係数の多変数連立代数方程式にも適用可能な Wu's method が野竹ら [3] により提案された。論文 [3] では、近似 GCD を利用した方法と安定化理論 [4] を利用した方法が提案され、安定化理論を用いた方法がより安定な解を与えるとされている。しかし安定化理論を用いると計算効率に問題があり並列化などの高速化が課題である。数式処理システムによる計算は一般に計算時間がかかるので、この点からも高速化は重要な課題である。Wu's method の並列計算は Wang[5] によりすでに提案されており、また、その手法を数式処理システム Risa/Asir にインプリメントし、クラスタ計算機上で実験した報告もある [6]。本稿では並列計算機上での安定化理論を用いた Wu's method の効率的な実装方法について検討し実験を行なった。

以下、2 節では Wu's method について、3 節では Wu's method の安定化について述べる。4 節では Wu's method の並列化について述べ、最後に 5 節で結果と今後の課題をまとめる。

## 2 Wu's method

Wu's method とは、多項式集合 $PS$ の characteristic set $CS$ を計算する方法である。$CS$ は、$L$-zero(·) で多項式集合の零点を表すと、

$$L\text{-zero}(CS/J) \subseteq L\text{-zero}(PS) \subseteq L\text{-zero}(CS)$$

を満たす。ここで、$J = \prod_{f \in CS} \mathrm{ini}(f)$ であり、$\mathrm{ini}(f)$ は多項式 $f$ の主係数を表す。

Wu's method のアルゴリズムを示す前に、用語を簡単に説明する。Wu's method では、多変数多項式集合を扱う。この多変数多項式に含まれる変数には順序が決められている。以後、

$$x_1 \prec x_2 \prec \cdots \prec x_n$$

と考える。ある多項式に含まれる変数のうち、順序最大 (添数最大) の変数を主変数と呼び、$\mathrm{lvar}(f)$ で表す。

非零多項式の有限個の列 $P_1, P_2, \ldots, P_r$ が、次の 2 つの条件のうち、どちらか一方を満たしているとき、この多項式集合を ascending set と呼ぶ。

*siraisi@dc.takuma-ct.ac.jp
†kai@cs.ehime-u.ac.jp
‡noda@cs.ehime-u.ac.jp

1. $r = 1$ かつ $P_1$ は定数

2. $\mathrm{lvar}(P_1) \prec \mathrm{lvar}(P_2) \prec \cdots \prec \mathrm{lvar}(P_r)$ 、かつ、整数の対 $(i,j)(0 < i < j \leq r)$ それぞれについて、多項式 $P_i$,$P_j$ が次の条件を満たす
   $P_i$ の主変数を $x_i$ とすると、
   $$\deg_{x_i} P_i > \deg_{x_i} P_j$$
   ここで、 $\deg_x f$ は多項式 $f$ の変数 $x$ に関する最大次数を表す。

環 $k[x_1, x_2, \ldots, x_n, y]$ 上の 2 つの多項式

$$\begin{aligned} f &= c_p y^p + \cdots + c_1 y + c_0 \\ g &= d_m y^m + \cdots + d_1 y + d_0 \end{aligned}$$

を考える。多項式 $f$ の $y$ に関する多項式 $g$ による擬剰余 $r$ は、

$$d_m^s \cdot f = q \cdot g + r$$

を満たすように決められる。$r = \mathrm{prem}(f,g,y)$ と表し、$q$ は擬商と呼ばれる。$d_m^s$ は除算が $k[x_1, x_2, \ldots, x_n, y]$ の範囲で計算できるように決められ、実際のアルゴリズムでは係数多項式が大きくならないよう $f$ の係数と $d_m$ の最大公約数で割ったものが使われる。擬除算を行うアルゴリズムを次に示す。

**アルゴリズム 1 (擬除算)**
入力：多項式 $f,g$ 、主変数 $y$
出力：擬剰余 $r = prem(f,g,y)$

1. $r \leftarrow f$, $m \leftarrow \deg_y r$, $n \leftarrow \deg_y g$

2. $m < n$ ならば、終了。

3. 以下の計算を行い、 2. へ。
   $$r \leftarrow \frac{d_m}{\gcd(d_m, \mathrm{ini}(r))} r - \frac{\mathrm{ini}(r)}{\gcd(d_m, \mathrm{ini}(r))} g \cdot y^{m-n}$$
   $$m \leftarrow \deg_y r$$

多項式集合 $PS = \{P_1, P_2, \ldots, P_n\}$ が ascending set ならば、多項式 $f$ の多項式集合 $PS$ による擬剰余 $r_0$ は以下の計算で求められる。

$$\begin{aligned} r_{n-1} &= \mathrm{prem}(f, P_n, \mathrm{lvar}(P_n)) \\ r_{n-2} &= \mathrm{prem}(r_{n-1}, P_{n-1}, \mathrm{lvar}(P_{n-1})) \\ &\vdots \\ r_1 &= \mathrm{prem}(r_2, P_2, \mathrm{lvar}(P_2)) \\ r_0 &= \mathrm{prem}(r_1, P_1, \mathrm{lvar}(P_1)) \end{aligned}$$

$r_0 = \mathrm{premas}(f, PS)$ と表す。
$CS$ を求める Wu's method のアルゴリズムは以下のように記述される。

**アルゴリズム 2 (Wu's method)**
入力：多項式集合 $PS = \{PS_1, PS_2, \ldots, PS_n\}$
出力：*characteristic set* $CS$

1. $PS$ より *ascending set* の条件を満たすように多項式集合 ($PS$ の部分集合) を選択し、 $CS$ とする。
2. $RS = \emptyset$
   $i = 1, 2, \ldots, n$ について、
   $$RS = RS \cup premas(PS_i, CS)$$
   を計算する。
3. $RS = \emptyset$ ならば、終了。
   その他の場合、 $PS = PS \cup RS$ とし、 1. へ。

# 3 Wu's method の安定化

浮動小数係数の多項式集合に対する Wu's method を考える。浮動小数係数の多項式集合に対し Wu's method を実行すると、擬剰余演算において微小項が残り、計算が終了しない。擬剰余演算における GCD 計算を浮動小数係数の多項式に対し実行できるものに置き換える必要がある。論文 [3] では、近似 GCD を利用した方法と安定化理論 [4] を利用した方法が提案され、安定化理論を用いた方法がより安定な解を与えるとされている。本稿でも、安定化理論を利用し、計算が終了しない問題を解決する。具体的には、係数に浮動小数点数を上限・下限に持つ矩形型区間数を用い、アルゴリズムの実行中に 0 を含む区間となった区間数を 0 へ書き換える "Zero-Rewriting" を行う。また、有効桁数を上げてアルゴリズムを実行し、計算結果が同じか否か判定することで解の安定性を調べている。ただし、理論的には安定性の議論をさらに詰める必要がある。浮動小数点数には、PARI ライブラリの bigfloat を用いた。GCD には、ユークリッドアルゴリズムを拡張したアルゴリズム [7, pp.420-439] を使用する。安定化した Wu's method のアルゴリズムは以下のように記述される。

**アルゴリズム 3 (安定化した Wu's method)**
入力：多項式集合 $PS = \{PS_1, PS_2, \ldots, PS_n\}$
出力：*characteristic set* $CS$

1. $PS$ より *ascending set* の条件を満たすように多項式集合 ($PS$ の部分集合) を選択し、 $CS$ とする。
2. $RS = \emptyset$
   $i = 1, 2, \ldots, n$ について、
   $$RS = RS \cup interval\text{-}premas(PS_i, CS)$$
   を計算する。
3. $RS \neq \emptyset$ ならば、$PS = PS \cup RS$ とし、 1. へ。
4. $CS$ が安定ならば、終了。
   その他の場合、有効桁数を上げて 1. へ。

# 4 Wu's methodの並列化

Wu's methodの並列化による高速化を考える。安定化したWu's methodの並列化を考える場合、次の2種類の並列化が考えられる。並列計算のモデルとしてマスタ–ワーカ・モデルを用い、マスタでは擬剰余演算をしないこととした。

**Prem** 安定化したWu's methodアルゴリズム(アルゴリズム3)のstep2の擬剰余計算を並列化する方法

**Prec** ワーカそれぞれへ異なる有効桁数の安定化したWu's method計算を割り当てる方法

Wu's methodのアルゴリズムのstep2では、$PS_i$ のCSに関する擬剰余演算interval-premas$(PS_i, CS)$を行っている。これらはそれぞれ独立に行うことができる。また、Wu's methodのアルゴリズムの中で、step2に最も計算時間がかかる。従って、step2を並列に計算させると効率良く高速化できる[6]。この並列化方法が、Premである。

一方で、安定化したWu's methodを並列化する場合、Premの並列化を行うことも可能であるが、ワーカそれぞれへ異なる有効桁数の計算を割り当てることも考えられる。例えば、有効桁数10桁のWu's methodの計算をワーカ1へ、有効桁数20桁のWu's methodの計算をワーカ2へ等と割り当てる。有効桁数の大きいところで安定になる問題であれば、Precの並列化を行うことにより通信回数が減り、高速な計算が期待できる。

以下では、並列版安定化したWu's methodを用いて、実際に問題を解いた結果について評価する。計算は、並列計算機AP3000(ノード: UltraSPARCII 360MHz×2, メモリ640MB)上で行ない、計算時間はRisa/Asirが持つtime()関数を用いて実時間(秒)により比較した。なお、表中の並列版でのプロセッサ数はワーカの数を表しており、マスタは含まれていない。実際に計算した有効桁数は19, 28, 38, 48, 57, 67, 77, 86, 96, 105, 115桁である。多倍長浮動小数点数演算にPARIライブラリを利用してるため、このような有効桁数になっている。用いた例題を以下に示す。

例題1

変数順序 $s_1 \prec c_1 \prec s_2 \prec c_2 \prec s_3 \prec c_3$

$$\begin{cases} f_1 = [1.4, 1.4]\,(c_1c_2c_3 - c_1s_2s_3 - s_1c_2s_3 - s_1s_2c_3) + [1.2, 1.2]\,(c_1c_2 - s_1s_2) + [1.0, 1.0]\,c_1 - [3.0, 3.0] \\ f_2 = [1.4, 1.4]\,(c_1c_2s_3 + c_1s_2c_3 + s_1c_2c_3 - s_1s_2s_3) + [1.2, 1.2]\,(c_1s_2 + c_2s_1) + [1.0, 1.0]\,s_1 - [1.3, 1.3] \\ f_3 = [1.0, 1.0]\,(c_1^2 + s_1^2 - 1.0) \\ f_4 = [1.0, 1.0]\,(c_2^2 + s_2^2 - 1.0) \\ f_5 = [1.0, 1.0]\,(c_3^2 + s_3^2 - 1.0) \end{cases}$$

例題2

変数順序 $s_1 \prec c_1 \prec s_2 \prec c_2 \prec d_3 \prec s_4 \prec c_4 \prec s_5 \prec c_5 \prec s_6 \prec c_6$

$$
\begin{cases}
f_1 = ((-s_6s_4 + c_6c_5c_4)c_2 - c_6s_5s_2)c_1 + (-c_6c_5s_4 - s_6c_4)s_1 - 1 \\
f_2 = ((-c_6s_4 - s_6c_5c_4)c_2 + s_6s_5s_2)c_1 + (s_6c_5s_4 - c_6c_4)s_1 \\
f_3 = (s_5c_4c_2 + c_5s_2)c_1 - s_5s_4s_1 \\
f_4 = d_3s_2c_1 - d_2s_1 - d_2 \\
f_5 = (c_6c_5s_4 + s_6c_4)c_1 + ((-s_6s_4 + c_6c_5c_4)c_2 - c_6s_5s_2)s_1 \\
f_6 = (-s_6c_5s_4 + c_6c_4)c_1 + ((-c_6s_4 - s_6c_5c_4)c_2 + s_6s_5s_2)s_1 - 1 \\
f_7 = s_5s_4c_1 + (s_5c_4c_2 + c_5s_2)s_1 \\
f_8 = d_2c_1 + d_3s_2s_1 \\
f_9 = -c_6s_5c_2 + (s_6s_4 - c_6c_5c_4)s_2 \\
f_{10} = s_6s_5c_2 + (c_6s_4 + s_6c_5c_4)s_2 \\
f_{11} = c_5c_2 - s_5c_4s_2 - 1 \\
f_{12} = d_3c_2 - 2d_2 \\
f_{13} = c_1^2 + s_1^2 - 1 \\
f_{14} = c_2^2 + s_2^2 - 1 \\
f_{15} = c_4^2 + s_4^2 - 1 \\
f_{16} = c_5^2 + s_5^2 - 1 \\
f_{17} = c_6^2 + s_6^2 - 1
\end{cases}
$$

これらの例題は、ロボットの逆問題と呼ばれる問題で、例題2はスタンフォード型マニピュレータに由来する [8]。例題2について、係数を区間数で記述していないが、実験では区間数を用いている。

計算時間を表1に示す。これより安定化した Wu's method を並列計算する場合、それぞれのワーカへ異なる有効桁数をの計算を割り当てる Prec が有利であると言える。これは、Prem では、通信時間等が Prec よりも増えるためと考えられる。

表 1: 安定化した Wu's method の並列計算における計算時間 (sec.)

| | 逐次処理 | Prem(プロセッサ数) | | | Prec(プロセッサ数) | | |
|---|---|---|---|---|---|---|---|
| | | 4 | 8 | 12 | 4 | 8 | 12 |
| 例題1 | 216 | 89.4 | 64.6 | 59.8 | 64.2 | 43.8 | 29.2 |
| 例題2 | 192 | 106 | 102 | 107 | 61.9 | 41.4 | 27.7 |

## 5 おわりに

安定化理論を用いた Wu's method の安定化と、その並列化を行い、分散メモリ型スカラ並列計算機 AP3000 上で動作する数式処理システム Risa/Asir 上に実装した。並列化には、擬剰余演算を分散させる Prem と異なる有効桁数の Wu's method 計算を分散させる Prec を用いて、実装した。ロボットの逆問題を例題に、Prec がより高速に計算できることを示した。

今後の課題として、解の安定判定をつめることや、他の問題へ応用することが挙げられる。

# 参 考 文 献

[1] Wu, W.: A Mechanization Method of Equations-solving and Theorem-proving, *Advances in Computing Research*, **6**, 1992, 103–138.

[2] Wu, W.: Mechanical Theorem Proving in Geometries, Texts and Monographs in Symbolic Computation, Springer-Verlag, Wien, 1994.

[3] Notake, Y., Kai, H. and Noda, M. T.: Symbolic-numeric computations of Wu's method: Comparison of the cut-off method and the stabilization techniques, *Computer Mathematics Proceedings of the Fifth Asian Symposium (ASCM2001)* (Shirayanagi, K. and Yokoyama, K., eds.), World Scientific, 2001, 122–130.

[4] Shirayanagi, K. and Sweedler, M.: A Theory of Stabilizing Algebraic Algorithms, *Tech.Rep.95-28*, Cornell Univ., pp.1–92, 1995.

[5] Wang, D.: On the Parallelization of Characteristic-Set-Based Algorithms, Proceedings of the 1st International ACPC Conference (Salzburg, Austria, September 30 - October 2, 1991), Springer's LNCS 591, 338–349.

[6] 白石 啓一, 那須 英正, 甲斐 博, 野田 松太郎: Wu の方法の並列化における負荷分散について, 数理解析研究所講究録 1199 数式処理における理論と応用の研究, 京都大学数理解析研究所, 2001, 10–19.

[7] Knuth, D. E.: The Art of Computer Programming, **2** Seminumerical Algorithms, Addison-Wesley, USA, 1997

[8] 吉川 恒夫: ロボット制御基礎論, コロナ, 1988