

多変数多項式の近似因数分解の効率化 —複数点での Taylor 級数根の利用—

森田 泰弘

筑波大学 理工学研究科 *

YASUHIRO MORITA

MASTER'S PROGRAM IN SCIENCE AND ENGINEERING, UNIVERSITY OF TSUKUBA

佐々木 建昭

筑波大学 数学系 †

TATEAKI SASAKI

INSTITUTE OF MATHEMATICS, UNIVERSITY OF TSUKUBA

1 はじめに

近似代数は計算代数の分野では世界的トピックの 1 つとなりつつあり, 将来, 理工学などの応用分野においては, 必須の計算法を提供すると予想される. 近似因数分解 [SSKS91] は厳密因数分解の概念を拡張したものとして考案された. 多変数多項式の近似因数分解の代表的算法の 1 つに和零関係を利用したもの [Sas01, SSKS91] があるが, 本研究はその算法の効率化と安定化を目標としている. 和零関係を求めるには, Hensel 構成で Taylor 級数根 φ_i を高次項まで計算しなければならない. しかし高次まで計算すると Hensel 構成には, a) 計算時間がかかる, b) 級数根の係数部に誤差が累積する, という 2 つの欠点がある. そこで, Hensel 構成を 1 つの展開点ではなく複数の展開点で行う [Sas01] ことにより, 高次項の計算を減らす方法について考察する.

2 記号と定義

本稿では x を主変数, \mathbf{u} を従変数とする, 次のような無平方な多変数多項式 $F(x, \mathbf{u})$ を考える:

$$F(x, \mathbf{u}) = f_n(\mathbf{u})x^n + f_{n-1}(\mathbf{u})x^{n-1} + \cdots + f_0(\mathbf{u}) \in \mathbf{C}[x, \mathbf{u}]. \quad (1)$$

ここで, \mathbf{u} は l 個の従変数 u_1, \dots, u_l を略記したもので, $\mathbf{u} = (u_1, \dots, u_l)$ である: $F(x, \mathbf{u}) = F(x, u_1, \dots, u_l)$, $\mathbf{C}[x, \mathbf{u}] = \mathbf{C}[x, u_1, \dots, u_l]$.

定義 1 (無限大ノルム)

F の無限大ノルム (infinity norm) を F の絶対値最大の数係数で定義し, $\|F\|$ と表す.

*yasu@math.tsukuba.ac.jp

†sasaki@math.tsukuba.ac.jp

定義 2 (展開点, 特異点)

Hensel 構成において, 従変数 $\mathbf{u} = (u_1, \dots, u_l)$ に代入する点 $\mathbf{s} = (s_1, \dots, s_l) \in \mathbf{C}^l$ を展開点 (expansion point) と呼び, 展開点 \mathbf{s} のうち $F(x, \mathbf{s})$ が無平方とならないものを特異点 (singular point) と呼ぶ.

定理 3 (一般 Hensel 構成 [SSKS91])

多項式イデアル $\langle u_1 - s_1, \dots, u_l - s_l \rangle$ を U と表す. 任意の正整数 k に対して, 次式を満たす $F_i^{(k)}$ が存在する.

$$F(x, \mathbf{u}) \equiv f_n(\mathbf{u}) \cdot F_1^{(k)}(x, \mathbf{u}) \cdots F_n^{(k)}(x, \mathbf{u}) \pmod{U^{k+1}}, \quad (2)$$

$$F_i^{(k)}(x, \mathbf{u}) \equiv F_i^{(0)}(x) \pmod{U}, \quad i = 1, \dots, n. \quad (3)$$

定義 4 (近似因数分解 [SSKS91])

$F, G, H, \Delta \in \mathbf{C}[x, \mathbf{u}]$, ε を微小正数とする. F が次式を満たすとき, F は許容度 (tolerance) ε で G, H に近似因数分解 (approximate factorization) されるという.

$$F = GH + \Delta, \quad \|\Delta\|/\|F\| = \varepsilon \ll 1. \quad (4)$$

これを $F = GH$ (tol ε) と表す.

定義 5

$[\varphi_i(\mathbf{u})]_a$ を全次数が a 以下の $\varphi_i(\mathbf{u})$ の単項式の和, $[\varphi_i(\mathbf{u})]_b$ を全次数が b 以上の $\varphi_i(\mathbf{u})$ の単項式の和と定義する. また, $[\varphi_i(\mathbf{u})]_b^{\text{def}} \equiv [[\varphi_i(\mathbf{u})]_b]_a$ とする.

定義 6 (次数上限 \bar{e})

$F(x, \mathbf{u})$ に対して $e_i \stackrel{\text{def}}{=} \text{tdeg}_{\mathbf{u}}(f_i(\mathbf{u}))$ ($i = 1, \dots, n$) と定義し, 次数上限 \bar{e} を

$$\bar{e} \stackrel{\text{def}}{=} e_n + \max\{(e_{n-j} - e_n)/j \mid j = 1, \dots, n\} \quad (5)$$

と定義する.

3 和零関係を用いた因数分解

実際の計算は近似的に行うが, ここでは分かり易さのため厳密な場合 ($\varepsilon = 0$) で示す. この算法は, 基本対称式系とベキ乗和系の同値性, および高次項の係数を利用する [Sas01, SSKS91] のがポイントである.

$F(x, \mathbf{u})$ は $F(x, \mathbf{u}) = G(x, \mathbf{u}) \cdot H(x, \mathbf{u})$ と因数分解できると仮定する.

$$\begin{cases} G(x, \mathbf{u}) = g_m(\mathbf{u})x^m + \cdots + g_0(\mathbf{u}), & m > 1, \\ H(x, \mathbf{u}) = h_{n-m}(\mathbf{u})x^{n-m} + \cdots + h_0(\mathbf{u}), & n - m > 1. \end{cases} \quad (6)$$

$F(x, \mathbf{u})$ が Hensel 構成により, 次のように表されたとする (各 $\varphi_i(\mathbf{u})$ は Taylor 級数根を表す).

$$F(x, \mathbf{u}) = f_n(\mathbf{u}) \cdot F_1(x, \mathbf{u}) \cdots F_n(x, \mathbf{u}) = f_n(\mathbf{u}) \cdot (x - \varphi_1(\mathbf{u})) \cdots (x - \varphi_n(\mathbf{u})). \quad (7)$$

このとき,

$$\begin{cases} [f_n \cdot (\varphi_{i_1}^1 + \cdots + \varphi_{i_m}^1)]_{\bar{e}+1}^{n\bar{e}} = 0, \\ [f_n^2 \cdot (\varphi_{i_1}^2 + \cdots + \varphi_{i_m}^2)]_{2\bar{e}+1}^{2n\bar{e}} = 0, \\ \vdots \\ [f_n^m \cdot (\varphi_{i_1}^m + \cdots + \varphi_{i_m}^m)]_{m\bar{e}+1}^{m\bar{e}} = 0, \end{cases} \quad (8)$$

を満たす $\varphi_{i_1}, \dots, \varphi_{i_m}$ ($i_1, \dots, i_m \in \{1, \dots, n\}$) の組を見つけることにより, $F(x, \mathbf{u})$ の既約因子 $G(x, \mathbf{u})$ を決定できる. 求め方は次の通り. $\mathbf{a}_i = (a_{i1} \ a_{i2} \ \dots \ a_{in'})$ を $\varphi_i, \varphi_i^2, \dots, \varphi_i^m$ の係数からなるベクトルとし, 行列 M を

$$M \stackrel{\text{def}}{=} \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n'} \\ a_{21} & a_{22} & \cdots & a_{2n'} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn'} \end{pmatrix} \begin{array}{l} \leftarrow \varphi_1, \varphi_1^2, \dots \\ \leftarrow \varphi_2, \varphi_2^2, \dots \\ \vdots \\ \leftarrow \varphi_n, \varphi_n^2, \dots \end{array} \quad (9)$$

で定義する. M に第 i 列による第 i 行消去を繰り返し, 和零関係 $\mathbf{a}_{i_1} + \mathbf{a}_{i_2} + \dots + \mathbf{a}_{i_m} = \mathbf{0}$ を満たす因子の組を求める. この組に対応する因子の積 $F_{i_1}(x, \mathbf{u}) \cdots F_{i_m}(x, \mathbf{u})$ が $F(x, \mathbf{u})$ の既約因子 $G(x, \mathbf{u})$ になっている [Sas01].

4 複数展開点の利用

4.1 Hensel 構成の欠点

浮動小数係数多項式の Hensel 構成では, 級数根を高次まで計算すると, a) 計算時間がかかる, b) 級数根の係数部に誤差が累積する, という 2 つの欠点があり, この 2 点が算法の効率化と安定化の妨げとなっている. 表 1 を見て頂きたい.

表 1: Hensel 構成の展開次数による変化

次数 (k 次)	計算時間 (秒)	係数部 (coefficient of u_1^k -term)	誤差部 (error of coefficient)
10	0.03	$-0.00090 - 0.02439i$	1.72877×10^{-14}
20	0.14	$0.01348 + 0.00533i$	1.70781×10^{-10}
30	0.44	$-0.01871 + 0.00968i$	1.79381×10^{-6}
40	0.91	$-0.03125i$	2.83210×10^{-2}
50	1.75	$352.0 + 60.0i$	5.76133×10^2

表 1 は, 10 次の 2 変数多項式 $F(x, u_1)$ の Taylor 級数根 $\varphi_i(u_1)$ ($i = 1, \dots, 10$) を, Hensel 構成で k 次まで計算した結果である. 計算時間は $\varphi_1, \dots, \varphi_{10}$ を k 次まで計算するのに要した時間であり, 係数部と誤差部は, φ_{10} の u_1^k の係数と係数に含まれる桁落ち誤差をそれぞれ表している. 40 次や 50 次まで展開すると, u_1^k の係数は誤差に埋もれてしまい, 全く意味を持たない値になっている. 展開次数が高くなるにつれて, 計算時間が必要になり, 精度が落ちていく様子が分かって頂けると思う. そこで, 展開点を複数個選び, 高次項の計算を減らすことを考える [Sas01].

4.2 複数点での展開

本稿では, 異なる P 個の展開点を $\mathbf{s}^{(p)} \stackrel{\text{def}}{=} (s_1^{(p)}, \dots, s_n^{(p)})$ ($p = 1, \dots, P$) とし,

$$\begin{aligned} F(x, \mathbf{s}^{(p)}) &= f_n(\mathbf{s}^{(p)}) \cdot (x - \alpha_1^{(p)}) \cdots (x - \alpha_n^{(p)}), & \alpha_i^{(p)} &\text{は } \mathbf{s}^{(p)} \text{ での数値根,} \\ F(x, \mathbf{u}) &= f_n(\mathbf{u}) \cdot (x - \varphi_1^{(p)}) \cdots (x - \varphi_n^{(p)}), & \varphi_i^{(p)} &\text{は } \mathbf{s}^{(p)} \text{ での Taylor 級数根,} \end{aligned} \quad (10)$$

とする.

展開点が異なっても、各級数根が解析的に繋がっていれば同じ和零関係が成り立つ。しかし、どの根とどの根が対応するかは自明でない。本稿では級数根の対応づけに、Smith の定理を用いる方法 ([方法 S]), 最小根の上界定理を用いる方法 ([方法 B]), Taylor 級数を用いる方法 ([方法 T]) の 3 通りの方法を用いる。級数根の対応づけに関する詳細は、参考文献 [SI02a, SI02b] を参照されたい。

本章では、各展開点での Taylor 級数根に

$$\varphi_i^{(1)} \longleftrightarrow \varphi_i^{(2)}, \dots, \varphi_i^{(1)} \longleftrightarrow \varphi_i^{(P)}, \quad i = 1, \dots, n, \quad (11)$$

と対応づけが行われたとして、前章の M と同様に行列 M' を作ることを考える。

a'_i を $\varphi_i^{(1)}, \varphi_i^{(2)}, \dots, \varphi_i^{(1)^2}, \varphi_i^{(2)^2}, \dots$ の係数からなるベクトルとし、行列 M' を

$$M' \stackrel{\text{def}}{=} \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n'} \\ a_{21} & a_{22} & \cdots & a_{2n'} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn'} \end{pmatrix} \leftarrow \begin{array}{l} \varphi_1^{(1)}, \varphi_1^{(2)}, \dots, \varphi_1^{(1)^2}, \varphi_1^{(2)^2}, \dots \\ \varphi_2^{(1)}, \varphi_2^{(2)}, \dots, \varphi_2^{(1)^2}, \varphi_2^{(2)^2}, \dots \\ \vdots \\ \varphi_n^{(1)}, \varphi_n^{(2)}, \dots, \varphi_n^{(1)^2}, \varphi_n^{(2)^2}, \dots \end{array} \quad (12)$$

で定義する。後は前章と同様にして和零関係を満たす因子の組を見つければよい。展開点を p 個選ぶと Hensel 構成を p 回行う必要があるが、その分、各展開点での級数根の展開次数を $\lceil \frac{(n+p-1)\bar{e}}{p} \rceil$ 程度に抑えて行列を構成できる ($\lceil \cdot \rceil$ はガウス記号を表す)。

5 比較実験

モニックかつ無平方な n 次の 2 変数多項式 $F(x, u_1)$ を、各 n に対して 50 個ずつ生成し、実験を行った。ここで、 $n = \text{tdeg}_x(F)$ であり、式 (5) の \bar{e} は $\bar{e} = 1$ を選んだ。行列演算には有効浮動小数 [KS97] のゼロ書き換えを利用し、それ以外の演算は倍精度の浮動小数で行っている。実験には PC(OS : Vine Linux 2.6, CPU : AthlonXP 1.60GHz, Memory : 1GB) 上で数式処理言語 GAL を用いた。各アルゴリズムを次のように名づけておく。

- **Algorithm-1** : 展開点が 1 つのアルゴリズム (展開点 : $s = 0$).
- **Algorithm-2** : 展開点が 2 つのアルゴリズム (展開点 : $s_1^{(1)} = 0, s_1^{(2)} = 0.02$).
 - Algorithm-2S** : 級数根の対応づけに [方法 S] を用いたアルゴリズム.
 - Algorithm-2B** : 級数根の対応づけに [方法 B] を用いたアルゴリズム.
 - Algorithm-2T** : 級数根の対応づけに [方法 T] を用いたアルゴリズム (級数根は 3 次まで計算).

5.1 計算時間の比較

図 1(左) は、各アルゴリズムの計算時間を比較したものである。10 次以下では、展開点を 2 個とった方が時間がかかっている。これは、級数根の展開次数は $F(x, u_1)$ の次数 n に比例するため、低次では Hensel 構成を 2 回行うという 2 度手間により計算時間が余計にかかってしまうからである。展開点を 2 個とったことによる利点は 12 次を越えてから表れており、次数が上がるにつれて計算時間の差が広がっていく。

図 1(右) は対応づけに要した時間の比較である。**Algorithm-2** の 3 つのアルゴリズムは、級数根の対応づけ以外は同じアルゴリズムを用いているので、対応づけに要する時間の差が因数分解に要する計算時間の差となって表れる。**Algorithm-2S** は 1 回の接続に要する時間が短い。しかし、Smith の定理を用いるため、その性質上、1 回で接続できる展開点間の距離は限られてしまい [SI02a, SI02b], 何度も接続を繰り返す

返すことになる。一方、Algorithm-2Bと-2Tは、1回の接続に要する時間は長いですが、ある程度の距離を離して行える。反復回数が少ないので全体の計算時間が短くて済む。

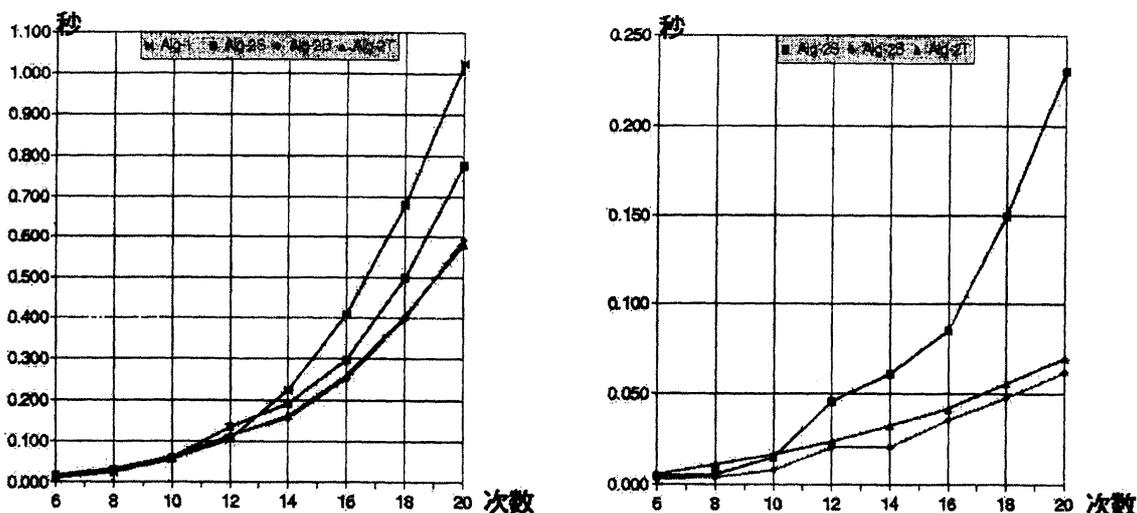


図 1: (左) 因数分解に要した時間, (右) 級数根の対応づけに要した時間

例 1 (2点での展開により計算時間が短縮される場合)

次式の因数分解を考える。

$$\begin{aligned}
 F(x, u_1) = & x^{20} - (2u_1 + 1)x^{19} + (2u_1 + 3)x^{18} - (4u_1^2 + 7u_1)x^{17} + (2u_1^2 - u_1 + 1)x^{16} - (2u_1^2 + 1)x^{15} \\
 & - (6u_1^2 - 5u_1 - 3)x^{14} + (u_1^2 + 4u_1 + 3)x^{13} + (5u_1^2 + 15u_1)x^{12} - (6u_1^2 + u_1 - 5)x^{11} \\
 & + (6u_1^2 - 3u_1 + 4)x^{10} + (6u_1^2 - 9u_1 - 12)x^9 - (2u_1^2 + u_1 + 8)x^8 + (3u_1^2 - 12u_1 + 6)x^7 \\
 & + (5u_1^2 - 3u_1 + 15)x^6 - (12u_1^2 - 24u_1 - 2)x^5 + (2u_1^2 + 9u_1 - 5)x^4 \\
 & + (2u_1^2 - 15u_1 - 2)x^3 - (8u_1^2 + 4)x^2 - (3u_1 + 3)x + 2u_1^2 - 5u_1 + 2.
 \end{aligned}
 \tag{13}$$

展開点を2つとり、級数根の展開次数を半分にする事で、級数根の計算時間が2分の1以下に抑えられている。全体としても、従来の6割程度の計算時間で済む(表2)。

表 2: 式(13)の計算結果

	級数根の 展開次数	行列の サイズ	計算時間(秒)				因数分解 の結果
			対応づけ	級数根の計算	その他	合計	
Alg-1	20	20 × 37	—	0.54	0.33	0.87	成功
Alg-2S	10	20 × 38	0.08	0.23	0.23	0.54	成功
Alg-2B			0.05	0.25	0.23	0.53	成功
Alg-2T			0.06	0.22	0.26	0.54	成功

5.2 成功率の比較

図2は、Algorithm-1と-2Tで因数分解に成功した多項式の個数を比較したものである。6次から10次まではどちらのアルゴリズムでも成功率は同じであり、違いが表れたのは12次を越えてからである。Algorithm-2Tでは成功率に改善が見られたが、高次の多項式についてはまだ十分なものではない。

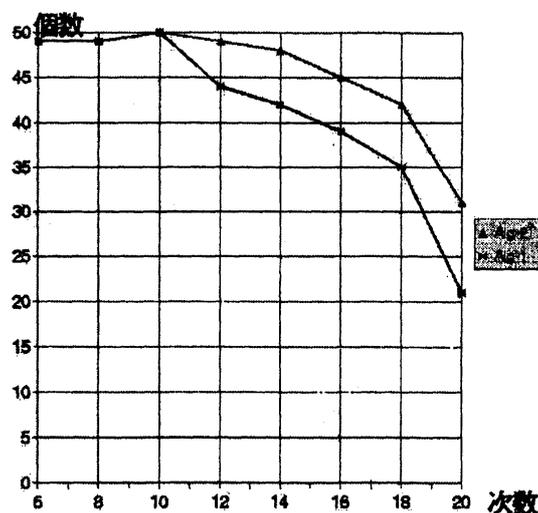


図2: 因数分解に成功した個数

実験サンプルの中には、行列の成分に含まれる誤差の最大値が、行列を構成した段階では約 2.55×10^{-14} であったが、消去が終わった段階では約 4.73×10^{-3} にまで膨れあがってしまうものがあった。つまり消去の計算過程で有効桁が11桁も失われたことになり、倍精度の演算では全く意味をもたない係数が含まれていることになる。この場合は、より高い精度での計算が必要になる。

5.3 特異点への対策

図2では高次の多項式に対しての成功率が低いが、これは展開点が特異点の近傍であることに起因している場合が多い。例2がその例である。

例2 (特異点による影響)

$$\begin{aligned}
 F(x, u_1) = & x^{12} - u_1 x^{11} - 3u_1 x^{10} + (2u_1^2 + 3u_1 + 1)x^9 + (u_1^2 - 2u_1 - 1)x^8 \\
 & - (5u_1^2 + 3u_1 + 3)x^7 + (2u_1^2 + 7u_1 + 2)x^6 + (4u_1^2 + 3u_1 - 5)x^5 \\
 & - (2u_1^2 + 14u_1 + 5)x^4 + (4u_1 + 2)x^3 + 2u_1 x^2 - (4u_1 + 6)x + 4.
 \end{aligned} \tag{14}$$

式(14)は、原点付近に特異点を持つ。式(14)をAlgorithm-2で因数分解する場合、原点を展開点として級数根 $\varphi_i(u_1)$ を u_1^7 まで計算することになる。各 $\varphi_i(u_1)$ の係数は次のようになる。

$$\begin{cases} \|\varphi_i\|^7 \lesssim 2.4 \times 10^7, & i = 2, 3, 9, 11, \\ \|\varphi_j\|^7 \lesssim 1.4, & j \neq i, 1 \leq j \leq 12. \end{cases} \tag{15}$$

和零関係を求めるのに用いる行列は級数根の係数から構成されるので、今の場合、行列成分の大きさには著しい差がでてしまう。このような状況で数値計算を行っても、望むべき和零関係を得ることはできない。

この例では **Algorithm-1, -2** 共に因数分解に失敗する。しかし、展開点を特異点から離すことにより係数の大きさを抑えることができる (表 3)。このような補正を行うと、**Algorithm-1, -2** のどちらでも式 (14) の因数分解が可能になる。

表 3: 展開点の位置による係数の変化 ($1 \leq i, j \leq 12, i \neq j$)

	展開点			
	0	0.05	0.1	1.0
$\ \varphi_i\ ^7$ の最大値	2.4×10^7	9.7×10^4	4.3×10^3	19.0
$\ \varphi_j\ ^7$ の最大値	1.4	1.4	1.4	1.8

図 2 の実験で用いたサンプルに、次の補正を加えて実験を行った。

(補正) **Algorithm-1** の展開点を $s = \eta$, **Algorithm-2** の展開点を $s_1^{(1)} = \eta, s_1^{(2)} = 0.02 + \eta$ とし, $\eta = 0$ での Hensel 構成で因数分解に失敗した場合, η を $0 < \eta \leq 3$ の範囲で動かし, Hensel 構成をやり直す。

補正前と補正後の成功率を比較したのが図 3 である。この補正により、成功率が改善されている。特異点近傍での展開が、如何に算法を不安定にさせるかが分かって頂けると思う。展開点を特異点から離して選ぶ方法を確立する必要がある。

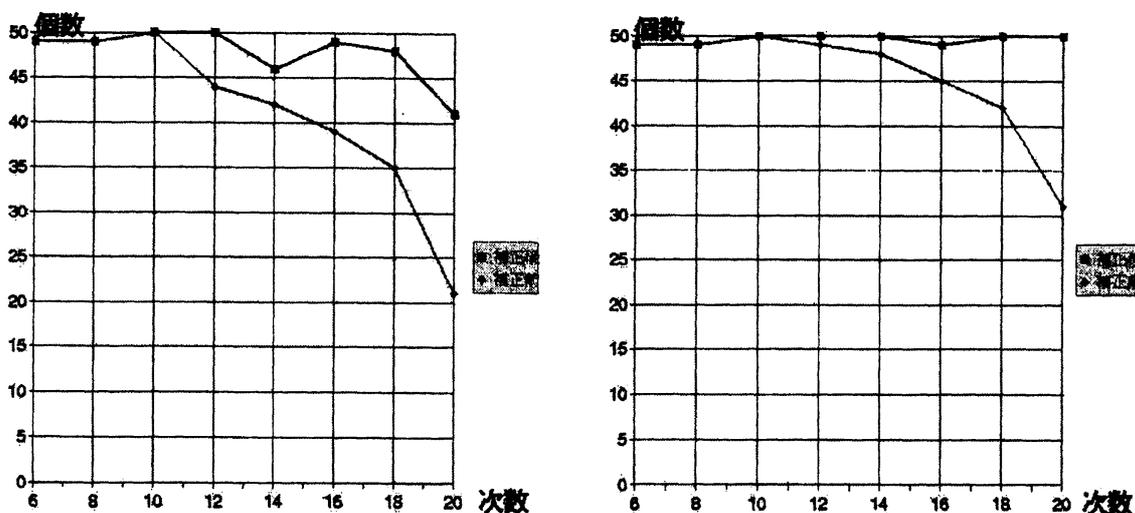


図 3: (左)Alg-1 の成功率比較, (右)Alg-2T の成功率比較

6 まとめ

本研究では和零関係を用いた近似因数分解算法の効率化と安定化を図るため、複数点での Taylor 級数根を利用する方法を検討した。原算法にとって、高次項の計算は効率と安定性の両面でネックとなっていた。

複数点での Taylor 級数根を用いたのは、この 2 つの欠点を除く可能性があるからである。実験結果を見る限り、本稿で用いたアイデアは一定の成果が表れていると言える。しかし、算法には改善すべき点があり、効率化の余地は残されている。今回のアイデアについて言えば、より効率的な対応づけの方法や、展開点を特異点から離して選ぶ方法、あるいは特異点近傍で展開した場合への対策が必要である。また、高次多項式の因数分解ではサイズの大きな行列に対して数値演算を行うことになる。浮動小数演算により生じる誤差の累積を如何に抑えるかといった工夫も必要であろう。

参 考 文 献

- [KS97] F. Kako and T. Sasaki. Proposal of “Effective Floating-point Number” for pp Approximate Algebraic Computation. Preprint, Nara Women’s Univ. and Univ. Tsukuba, 12 pages, 1997.
- [NS98] K. Nagasaka and T. Sasaki. Approximate Multivariate Polynomial Factorization and Its Time Complexity. Preprint, Univ. Tsukuba, 16 pages, 1998.
- [Na01] K. Nagasaka. Estimation of Cancellation Errors in Multivariate Hensel Construction with Floating-Point Numbers. In *Proc. ATCM 2001*, pp. 408–415, 2001.
- [Na02a] K. Nagasaka. *A Study on the Approximate Multivariate Factorization*. PhD Thesis, Univ. Tsukuba, January 2002.
- [Na02b] 長坂耕作. 多変数 Hensel 構成における展開点の特異点からの距離. 村尾裕一 (編), *Computer Algebra – Algorithms, Implementations and Applications 2001*, 数理解析研究所講究録, 第 1295 巻, pp. 1–12, 2002.
- [Sas01] T. Sasaki. Approximate Multivariate Polynomial Factorization Based on Zero-Sum Relations. In *Proc. ISSAC 2001*, pp. 284–291, 2001.
- [Smi70] B. T. Smith. Error Bounds for Zeros of a Polynomial Based Upon Gerschgorin’s Theorems. *J. ACM*, Vol. 17, No. 4, pp. 661–674, 1970.
- [SI02a] T. Sasaki and D. Inaba. On Analytic Continuation of Algebraic Function. Preprint, Univ. Tsukuba, 14 pages, January 2002.
- [SI02b] 佐々木建昭, 稲葉大樹. 代数関数の解析接続について. 村尾裕一 (編), *Computer Algebra – Algorithms, Implementations and Applications 2001*, 数理解析研究所講究録, 第 1295 巻, pp. 123–129, 2002.
- [SKK93] T. Sasaki, T. Kitamoto, and F. Kako. On Cancellation Error in Newton’s Method for Power Series Roots of Multivariate Polynomial. Preprint, Univ. Tsukuba, 32 pages, 1993.
- [SSKS91] T. Sasaki, M. Suzuki, M. Kolář, and M. Sasaki. Approximate Factorization of Multivariate Polynomials and Absolute Irreducibility Testing. *Japan J. Indust. Appl. Math.*, Vol. 8, No. 3, pp. 357–375, 1991.
- [SY98] T. Sasaki and S. Yamaguchi. An Analysis of Cancellation Error in Multivariate Hensel Construction with Floating-point Number Arithmetic. In *Proc. ISSAC 98*, pp. 1–8. ACM, 1998.
- [SSH92] T. Sasaki, T. Saito, and T. Hilano. Analysis of Approximate Factorization Algorithm I. *Japan J. Indust. Appl. Math.*, Vol. 9, No. 3, pp. 351–368, 1992.