数理解析研究所講究録
第1568巻 2007年 156-161

整数係数の近似因数分解はなぜ難しいか

長坂耕作
KOSAKU NAGASAKA
神戸大学発達科学部
FACULTY OF HUMAN DEVELOPMENT, KOBE UNIVERSITY*

1 整数係数の近似因数分解

本講演で取り上げた問題は，整数係数の既約多項式を整数の範囲で多少変化させて良いから，元の多項式に近い可約な多項式を求めるという，これまでに少なくない研究が行われている多変数多項式の近似因数分解の整数版のようなものである．きちんと書き下すと次のような問題となる．

問題 1
$\mathbb{Z}$ 上既約な多項式 $f(x) \in \mathbb{Z}[x]$ に対して，$\|f(x)-g(x)h(x)\|$ を最小化する定数でない多項式の組 $g(x), h(x) \in \mathbb{Z}[x]$ を求めよ．ただし，零点の数が変化しないように，$\deg(f) = \deg(g) + \deg(h)$ を満すものとする． ◁

例 1
$\mathbb{Z}$ 上の近似因数分解の例．

$$\begin{aligned} x^2 - 143 &\to (x-12)(x+12)+1, \\ x^2+4x-2 &\to (x+3)(2x-1)-x^2+1, \\ &\to (x+2)(2x-1)-x^2+x. \end{aligned}$$

◁

近似代数演算については，様々な研究が行われているが，こと近似因数分解に限ると，二変数ないしはそれ以上の変数を含む多項式の研究に限られている（[GKMYZ04, CG06, S01] など）．一方で，単変数多項式の場合について取り上げている研究はないが，これは二変数以上の場合のように係数体を実数や複素数の範囲で考えると，単変数多項式が必ず一次か複素共役の積に分解されてしまうため，数式処理というよりは単純な数値解析の分野になってしまうためだろう．そのため，単変数で近似因数分解を考える場合，自ずと係数の範囲を離散的な整数などの空間に制限する必要がでてくる．そして，そのような制限を加えた近似因数分解の成果はこれまでに発表されていない．

今回の講演では，多項式ノルムとして無限大ノルムを採用した場合について，既存の因数分解法などに若干の変更を加えることで，整数係数の近似因数分解が可能であるかの検討を行った結果を発表している．その結果を次の章から記述するが，その前に，いくつかの定義を行っておく．

定義 1
$\mathbb{Z}$ 上既約な多項式 $f(x) \in \mathbb{Z}[x]$ に対して，$\| f(x) - g(x)h(x) \|_\infty$ を最小化する定数でない多項式の組 $g(x), h(x) \in \mathbb{Z}[x]$ で，$\deg(f) = \deg(g) + \deg(h)$ を満すものを，$f(x)$ の整数上の近似因数分解と定義する．このとき，$g(x)$ と $h(x)$ を近似因子，$\varepsilon = \|f(x) - g(x)h(x)\|_\infty$ を許容度とよぶ． ◁

*nagasaka@main.h.kobe-u.ac.jp

2 全探索アルゴリズム

事前に許容度 ε が与えられた場合，定義に基づく近似因子は次の多項式集合 $P_\varepsilon(f)$ の既約因子に含まれているはずである．許容度に制限がなければ，必ず近似因子は存在するが，許容度が与えられている場合，係数の変動に上限があるため近似因子が存在しないこともある．

$$P_\varepsilon(f) = \{\tilde{f}(x) | \ \|f(x) - \tilde{f}(x)\|_\infty \leq \varepsilon\}.$$

従って，この多項式集合に含まれる全ての多項式の因数分解を行えば，近似因子を見つけることが可能となる．この全探索アルゴリズムを書き下したものが次のアルゴリズムである．

アルゴリズム 1
入力：$f(x) \in \mathbb{Z}[x]$ と許容度 $\varepsilon \in \mathbb{N}$
出力：$f(x)$ の近似因子のリスト

1. $f(x)$ が可約なら既約因子を返し終了
2. $\tilde{\varepsilon} \leftarrow 1$ とし，$\tilde{\varepsilon} \leq \varepsilon$ の間，以下を繰り返す
   (a) $S \leftarrow P_{\tilde{\varepsilon}}(f) \backslash P_{\tilde{\varepsilon}-1}(f)$
   (b) $S \neq \phi$ の間，以下を繰り返す
      i. $s(x) \in S$ に対して，$s(x)$ が可約ならば既約因子を返し終了
      ii. $S \leftarrow S \backslash \{s(x)\}$
   (c) $\tilde{\varepsilon} \leftarrow \tilde{\varepsilon} + 1$
3. 許容度 ε で既約と返し終了

このアルゴリズムは全探索を行うので非常に効率が悪い．実際に，Pentium 4 3.2GHz，メモリ 2GB の Linux 上の Mathematica 5.2 に実装し実験を行ってみた．まず，係数が閉区間 $[-100, 100]$ にある 2 次と 3 次の多項式に許容度 $\varepsilon = 1$ で近似因数分解可能な多項式をランダムに 100 個生成し，近似因数分解を行ってみたところ，平均して 0.203 秒の計算時間が必要であった．与式の次数は 5 次なので，$P_1(f)$ の要素数は 729 となり，まだ待てる時間で終了している．次に，係数が閉区間 $[-100, 100]$ にある 3 次と 5 次の多項式に許容度 $\varepsilon = 1$ で近似因数分解可能な多項式をランダムに 100 個生成し，近似因数分解を行ってみたところ，平均して 44.3145 秒の計算時間が必要であった．与式の次数は 8 次なので，$P_1(f)$ の要素数は 19683 となり，計算時間が大きくなっている．従って，予想された通り，試みるべき多項式の数が指数関数的に増大するので，特殊なケースを除けば使用に耐えうるものでないことがわかる．

3 試し割りによる方法

整数上の通常の因数分解には，有限体上で分解したのちに Hensel 構成で係数を復元し，試し割りにより真の因子を見つける方法がある．この試し割りは，初期の多変数近似因数分解法でも取り上げられており ([SSKS91])，考えられる因子候補から真の因子を見つけるような場合に適用できるシンプルな方法と言える．そこで，整数上の近似因数分解においても，数値的に一次因子に分解してから，試し割りにより整数上の近似因子を見つけることを考える．この場合，許容度は後退誤差として計算されることになるため，最小の許容度での分解は難しい．この試し割りによるアルゴリズムを書き下すと次のようになる．

アルゴリズム 2
入力：$f(x) \in \mathbb{Z}[x]$ と許容度 $\varepsilon \in \mathbb{N}$
出力：$f(x)$ の近似因子の候補リスト

1. $f(x)$ の零点 $\omega_1, \ldots, \omega_{\deg(f)}$ を数値的方法で求める
2. $n \leftarrow 1$ とし，$n \leq \deg(f)/2$ の間，以下を繰り返す
   (a) $\omega_1, \ldots, \omega_{\deg(f)}$ から n 個を取り出す組みの集合 S を作成
   (b) $S \neq \phi$ の間，以下を繰り返す
      i. $s \in S$ に対して，$g(x), h(x) \in \mathbb{C}[x]$ を計算

$$g(x) = \prod_{i \in s}(x - \omega_i), \quad h(x) = \prod_{i \notin s}(x - \omega_i)$$

      ii. 係数を整数に丸めた多項式 $\tilde{g}(x), \tilde{h}(x) \in \mathbb{Z}[x]$ を計算
      iii. $\|f(x) - \tilde{g}(x)\tilde{h}(x)\|_\infty \leq \varepsilon$ ならば，$\tilde{g}(x), \tilde{h}(x)$ を返し終了
      iv. $S \leftarrow S \backslash \{s\}$
   (c) $n \leftarrow n + 1$
3. 近似因子候補を発見できずと返し終了

このアルゴリズムについて実験したところ，次のような場合にはうまく動作した．

$$\begin{aligned} f(x) &= x^2 - 143 = (x - 12)(x + 12) + 1 \\ &\to \approx (x - 11.9583)(x + 11.9583) \\ &\to (x - 12)(x + 12) \end{aligned}$$

しかしながら，次のような場合には全く機能しなかった．

$$\begin{aligned} f(x) &= x^3 + 181x^2 - 3900x + 20000 = (x + 200)(x^2 - 20x + 100) + x^2 \\ &\to (x + 200.907)(x^2 - 19.9074x + 99.5483) \\ &\to (x + 201)(x^2 - 20x + 100) \\ &= f(x) + x^2 - 20x + 100 \end{aligned}$$

結論から言えば，定数項の僅かな違いであれば期待する通りに機能することもあるが，基本的にどのような項の変動に対しても非常に脆い．これは，整数係数多項式として，$x^3 + 180x^2 - 3900x + 20000$ と $x^3 + 181x^2 - 3900x + 20000$ を比べれば，差は僅かではあるが，一般の多変数多項式の近似因数分解での変動が 1 未満の微小な量であることを考えれば，非常に巨大な変化なので当然の結果であろう．

Smith の誤差上界を用いると，整数における最小の変化である 1 の差が理論的に致命的な差を根に与えることがわかる（[TS00]）．$f(x)$ を次数 n のモニックかつ無平方な多項式とし，$\omega_1, \ldots, \omega_n$ を $f(x)$ の根候補としたとき，$f(x)$ の真の根は，複素平面上の中心が ω_i で半径 r_i の円の中に存在する．ここで，r_i は次式を満す Smith の誤差上界とする．

$$r_i = \left| \frac{n f(\omega_i)}{\prod_{j=1, \neq i}^{n} (\omega_i - \omega_j)} \right|.$$

そのため，$f(x)$ の項の僅か 1 の変化でも，半径 r_i も同程度の変化をする可能性があり，$f(x)$ の根と近似因子の根が 1 程度変化する可能性が大きい．上記の試し割りのアルゴリズムでは，$f(x)$ の根を使って，近似

因子を求めようとしているため，$f(x)$ と近似因子の根が大きく異なる場合は，近似因子を発見することは非常に困難となる．

また，試し割りは，最悪のケースでは組み合わせの個数分の回数を行う必要があるため，ナップザック法([H02]) などにより効率的に真の因子を見つける方法が提案されている．ナップザック法を使う場合，より一層，$f(x)$ と近似因子の根の違いによる影響を大きく受ける．基本的には，根の線形和が整数となる組を格子算法により発見するため，根の大きさに比べて僅かな変化であれば対応できるが，上記の Smith の誤差上界からもわかるように，一般に整数としては最小の 1 の変化であっても難しい．例えば，$1.321 + 0.679$ という関係が近似因子の根にあったとして，それぞれが $f(x)$ の根では，$1.346 + 0.611$ となっただけでも，和は整数になることはない．

なお，目的が既約因子の近似因数分解ではなく，可約な多項式を数値的に整数上で因数分解することであれば，実験でもある程度可能であった．これについては，論文 [H02] 中にも次のように記述されている．

> We could also consider computing the factors f_j in $\mathbb{R}[x]$ or $\mathbb{C}[x]$ instead of $\mathbb{Z}_p[x]$. Then compute the $Tr_i(f_j)$, cut away the integer part, and constrcut a knapsack problem in a similar way. Perhaps this is the algorithm one was looking for in Section 6 by Sasaki. et. al (1993).

4 その他の新しい方法

近似因数分解すべき多項式の使える性質としては，零点と係数が主なものと考えられる．零点については，前章にあるように，整数上での僅かな変動であっても，零点の変動は非常に大きく近似因数分解での利用は困難であった．とすれば，変動が生じている係数をそのまま利用することが，変動の影響を小さくすることに繋がると考えられる．

各係数にパラメータを付加することで，許容度の範囲で変化可能な多項式の全てを表現し，この多項式が可約となるパラメータを求めることができれば，整数上の近似因数分解が行えたことになる．そこで，有限体上の因数分解法の中で係数をそのまま利用可能な Niederreiter による方法 [N93] を使って，有限体上で可約となるパラメータの条件を計算することが考えられる．この方針で実装したものを，10 次以下の小さい多項式で実験してみたが，パラメータの条件を求めるグレブナー基底計算で破綻してしまう．また，この方法で求まる条件は必要条件であり，有限体上の因数分解法から求めていることもあり，全探索の候補が半分に減るくらいにしかならない．

次に，有限体でなく真に係数をそのまま使った因数分解法があれば良いのだが，通常は問題をリニアにするために有限体上の因数分解に帰着しており，そのまま使った効果的な方法は知られていない．可能性としては，近似 GCD や最近特異多項式などで利用される，因子次数を固定して最適化により係数決定を行うものもあるが，Mathematica に実装されているものを使った限りでは，全探索アルゴリズムの方が著しく効率的であった．この場合の最適化手法は，Integer Polynomial Programming であり，専用のソルバーを使えば若干の速度向上は望まれると思われるが，どちらにしても本質的な解決ではない．

5 多変数の近似因数分解との比較

整数上の多項式は，整数の p 進展開を行うことで，二変数多項式に似た形となるが，整数には桁下がりと桁上がりが存在するため，整数上の多項式と二変数多項式とを同一視することはできない．そのため，これから説明する多変数多項式の近似因数分解法を整数上の近似因数分解に適用することは残念ながらできない．この章では，主変数を除く，係数多項式の高次項にのみ誤差を含んでいる二変数多項式を扱うが，多変数多項式であっても基本的に同じ方法で分解できる．

次の二変数多項式の近似因数分解を考える.

$$f(x,y) = g(x,y)h(x,y) + \tilde{f}(x,y),\ \deg(f) \approx \mathrm{ord}(\tilde{f})$$

ここで, $g(x,y)h(x,y)$ は $f(x,y)$ に最も近い可約な多項式とする. 誤差は係数部の高次部にのみ存在するという仮定から $f(x,0) = g(x,0)h(x,0)$ が成り立つ. 従って, $f(x,0)$ が無平方であれば任意の方法で因数分解したのちに, ある程度の次数まで Hensel 構成することで, 誤差が含まれる次数にもよるが, 可約な多項式である $g(x,y)$ と $h(x,y)$ という因子を求めることが出来る. 最終的に最適化等の方法を用いなくとも因子が計算できるかは, 誤差の含まれる次数, 因子毎の次数などに依存する.

この方法は, 誤差が低次に含まれる場合は次数を反転した多項式を分解すれば良いので, 誤差が係数多項式の低次部分にのみ含まれる場合も機能する. 更に言えば, 拡張 Hensel 構成 [S00, I05] や解析的因数分解 [I05] を使うことで, ニュートンポリゴンの特定の周辺部にのみ誤差が含まれる場合にも, 厳密演算のみで近似因数分解を行うことが出来る.

アルゴリズム 3
入力: $f(x,y) \in \mathbb{C}[x,y]$ と誤差の場所
出力: $f(x,y)$ の近似因子のリスト

1. $f(x,y)$ が可約なら既約因子を返し終了
2. 誤差の部分に応じて変数変換 (誤差を高次項へ)
3. 単変数多項式に落として因数分解を行い変数復元
   - 原点で特異でない多項式の場合は *Hensel* 構成
   - 原点で特異な多項式の場合は拡張 *Hensel* 構成または解析的因数分解法
4. 因子による試し割りにより近似因子を決定 (係数部の高次項を無視)
5. 必要に応じて最適化で残差を小さくした近似因子を返し終了

整数上の近似因数分解を考える場合, やはり p 進展開が基本と思われる. 実際, 近似因数分解に必要な変動の大きさが p の倍数の場合, 有限体 Z_p 上ならば, 簡単に近似因子候補を普通の方法で求めることが出来る.

$$\begin{aligned} f(x) &= g(x)h(x) + p\tilde{f}(x) \\ &\equiv g(x)h(x) \pmod{p} \end{aligned}$$

従って, 古典的な因数分解法を複数の小さな素数に対して行うことで近似因子候補が求められる可能性は高い. しかしながら, そこから Hensel 構成などの p 進展開で整数を復元しようとすると, 無視できたはずの $p\tilde{f}(x)$ の影響を受けうまくいかない. 結局のところ p 進展開を使って整数上の近似因数分解を行おうとすると, 低次に誤差があれば Hensel 構成などの展開がうまく行かず, 誤差を高次側に変換しようとすると, 桁上がりと桁下がりの問題に阻まれてうまくいかない.

6 まとめ

本講演では, 整数係数多項式の整数上の近似因数分解を取り上げた. この問題は非常に簡単だが既知の方法の安易な応用だけでは困難であることがわかった. 現時点で近似因子を求める場合は, 全探索アルゴリズムのような発見的方法か, 整数多項式計画問題などの最適化を用いるしかないだろう. 問題を難しくしてい

る原因としては，僅かな変動であっても離散的な変動は零点にとっての大きな変動であること，多変数多項式の絶対既約分解と異なり，整数には桁上がりと桁下がりが存在するために簡単な問題への変換が出来ないことがあげられる．

解決に向けての方策としては，変動を許容する係数をそのまま使える因数分解法を考えること，有限体に落とさずに構成的に因数分解を行うことを考えることなどがあるが，現実的なところとして，まずは整数係数多項式の整数上の近似 GCD について取り組んでいきたい．

参 考 文 献

[CG06] G. Chèze and A. Galligo. From an approximate to an exact absolute polynomial factorization. *J. Symbolic Comput.*, 41(6):682–696, 2006.

[GKMYZ04] S. Gao, E. Kaltofen, J. May, Z. Yang and L. Zhi. Approximate factorization of multivariate polynomials via differential equations. *ISSAC 2004 Proc.*, pages 167–174, 2004.

[H02] M. Hoeij. Factoring Polynomials and the Knapsack Problem. *J. Number Theory*, 95(2):167–189, 2002.

[I05] T. Inaba. Factorization of multivariate polynomials by extended Hensel construction. *SIGSAM Bull.*, 39(1):2–14, 2005.

[I05] M. Iwami. Extension of expansion base algorithm for multivariate analytic factorization including the case of singular leading coefficient. *SIGSAM Bull.*, 39(4):122–126, 2005.

[N93] H. Niederreiter. A new efficient factorization algorithm for polynomials over small finite fields. *Appl. Algebra Engrg. Comm. Comput.*, 4(2):81–87, 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.*, 8:357–375, 1991.

[S00] T. Sasaki. Properties of Extended Hensel Factors and Application to Approximate Factorization. *Preprint (unpublished), Univ. Tsukuba*, 2000.

[S01] T. Sasaki. Approximate multivariate polynomial factorization based on zero-sum relations. *ISSAC 2001 Proc.*, pages 284–291, 2001.

[TS00] A. Terui and T. Sasaki. "Approximate Zero-points" of Real Univariate Polynomial with Large Error Terms. *IPSJ J.*, 41:974–989, 2000.