数理解析研究所講究録
第2054巻 2017年 55-67

疎な多変数多項式の拡張Hensel構成の効率化

Enhancing the Extened Hensel Construction of Sparse Multivariate Polynomials

佐々木 建昭 (Tateaki Sasaki) *
筑波大学 名誉教授
(University of Tsukuba)

稲葉 大樹 (Daiju Inaba) †
(公財) 日本数学検定協会
(Japan Assoc. Math. Certification)

Abstract

筆者らは2000年、拡張Hensel構成の初期因子を多項式とすることで、従変数の原点で主係数が0となり且つ従変数に関して疎な多変数多項式の因数分解法を提案した。一般Hensel構成に基づく算法では従変数の原点移動が必要で、項数が増大して計算効率が著しく低下するが、筆者らの算法では原点移動は必要なく、計算効率の低下は避けられる。しかし、主変数に関して高次かつ疎な場合は依然として未解決だった。本稿では未解決点も含め算法を抜本的に改善する。従来の拡張Hensel構成算法は、一般Hensel構成と同様、Moses-Yun補間式を用いて構成を行う。拡張Hensel構成では、Moses-Yun補間式は従変数に関して有理式となり、計算に非常に時間がかかる。Hensel因子の計算も、有理式を扱うので複雑でしかも重い。本稿で提案する算法はMoses-Yun補間式を全く用いず、初期因子を生成元とするグレブナー基底を用いる。ただし、初期因子が3個以上の場合でも2個づつの組に分割して計算する。旧算法では、分母因子の候補は初期因子の終結式で、分子多項式と簡約されて小さな分母因子になるが、新しい算法では分母因子の候補はグレブナー基底の最小順位の多項式で、終結式より遥かに小さいので以後の計算が軽い。さらに、分母多項式をシステム記号で置き換え有理式を多項式化するので、計算が一層効率化される。まだ小規模な例題でテストしただけだが、テスト結果は非常によい。

1 はじめに

Hensel構成は、$\mathbb{Z}$ 上の1変数多項式の素数 p を法とするもの、$\mathbb{Q}$ 上の多変数多項式 $F(x,\boldsymbol{u}) \stackrel{\text{def}}{=} F(x,u_1,\ldots,u_\ell)$ の $(u_1,\ldots,u_\ell)$ を法とする**一般Hensel構成** (generalized Hensel construction) が有名である [2, 3]。Hensel構成は多項式因数分解やGCD計算等で不可欠

*sasaki@math.tsukuba.ac
†d.inaba@su-gaku.net

で、計算代数の中で基本的演算の一つである。一方、**拡張 Hensel 構成** (extended Hensel construction) は日本で考案され [14, 15]、宣伝しないので知名度は低いが、近年、外国にも知られるようになった。拡張 Hensel 構成は、数学的には多変数代数関数の臨界点 (critical point, singular point を含むがより広い) での "級数展開" の算法として [6, 12]、計算代数では疎な多変数多項式の因数分解や GCD 計算等で重要な役割をはたす。

与えられた多変数多項式を $F(x, \boldsymbol{u})$, $\boldsymbol{u}$ は従変数全体, とする。$(\boldsymbol{u})$ の原点 $(\mathbf{0})$ で $F(x, \boldsymbol{u})$ の主係数 (主変数 x の最大次数項の係数) が 0 になるとき、F の**主係数は特異**、あるいは F は**主係数特異**であるという。$F(x, \boldsymbol{u})$ の Hensel 構成は通常、F が主係数非特異であれば $F(x, \mathbf{0})$ の互いに疎な因子を初期因子に設定し、その初期因子を "リフティング" して行う。F が主係数特異な場合には、従変数の原点移動 $(\boldsymbol{u}) \to (\boldsymbol{v}) + (\boldsymbol{s})$, $\boldsymbol{s} \in \mathbb{Q}^{\ell}$, を行い、主係数非特異な $F(x, \boldsymbol{v} + \boldsymbol{s})$ を Hensel 構成する。原点移動は平凡な操作だが、$F(x, \boldsymbol{u})$ が従変数に関して疎な場合、項数を (場合によるが爆発的に) 増加させ計算時間を著しく増大させる。これは**非零代入問題**と呼ばれている。非零代入問題を回避する方策として現在、最も有効なのは Zippel の方法である [19, 20]。L は大きな自然数とし、S は $-L$ から L までの整数の集合とする。非零の多変数多項式 $f(\boldsymbol{u}) \in \mathbb{Q}[\boldsymbol{u}]$ に対し、ランダムに $\boldsymbol{r} \in S^{\ell}$ を選び $f(\boldsymbol{u})$ に代入した場合、$f(\boldsymbol{r}) = 0$ となる確率は (L によるが) 非常に小さい (Schwartz-Zippel の補題 [16])。そこで Zippel は、未知多項式 $F(x, \boldsymbol{u}) = f_n(\boldsymbol{u})x^n + \cdots + f_0(\boldsymbol{u})$ の従変数 $\boldsymbol{u}$ に $\boldsymbol{r}$ を代入したとき、たとえば x^k-項, $n > k \geq 0$, が $F(x, \boldsymbol{r})$ に出現しなければ f_k は元々 0 であったとみなそうと提案した。Zippel の算法のように、疎な多項式を疎性を利用して効率よく決定する算法は**疎補間法 (sparse interpolation)** と呼ばれている。

上述の Zippel の方法は非零代入の回避策だが、一般 Hensel 構成を素直に拡張することで、主係数特異な $F(x, \boldsymbol{u})$ を非零代入なく Hensel 因子に分解する方法が日本で考案され、拡張 Hensel 構成と命名された [14, 15]。当初目的は 1 変数代数関数 (= 2 変数多項式の根) の Puiseux 級数展開の多変数化であったが、著者らは 2000 年、初期因子を多項式に設定することで計算代数での幅広い応用を目指した [11]。実際、著者の一人 (D.I) が実装し、疎な多変数多項式の因数分解に適用してその有用性を実証した [5]。昨年は Sanuki と共に疎な多変数多項式の GCD 計算にも応用した [13]。

拡張 Hensel 構成の従来の算法は、一般 Hensel 構成と同様、**Moses-Yun 補間式** (MY と略記)[10] を用いて Hensel 構成する。MY 補間式は拡張互除法で計算される。数学的には簡単で明快だが、時間がかかる上に $\boldsymbol{u}$ に関して有理式となる。また、Hensel 因子の計算も面倒で時間がかかる。そのため、従来の算法は主変数に関して高次で疎な多変数多項式には不向きで、算法の改善が求められていた。

第 2 章では、拡張 Hensel 構成の起点ともいえる Newton 多項式を定義し、MY 補間式の計算法および Hensel リフティングについて簡単に復習し、MY 補間式と Hensel 因子の例を用いつつ従来算法の欠点を指摘する。第 3 章では、初期因子のグレブナー基底を用いて如何に Hensel 構成を行うか、如何に分母因子を決定するかについて、定理を証明しつつ説明する。第 4 章では、トリビアルでない例で旧算法と新算法を比較し、計算時間の観点から新算法が非常に優れていることを実証する。なお、ページ数の制約から幾つかの事項の説明を割愛した。興味ある読者は英語論文を執筆中なのでそれを参照されたい。

2 拡張Hensel構成の簡単な復習

本稿ではxは主変数を、$\boldsymbol{u}$は従変数全体を表すことは既に述べた。多変数多項式$F(x,\boldsymbol{u})$に対し、$\deg(F)$, $\mathrm{lc}(F)$, $\mathrm{cont}(F)$はそれぞれ、主変数xに関する**次数**、**主係数**、**係因数**(content, xに関する係数の**最大公約子**(GCD))をそれぞれ表す。項$T=cu_1^{e_1}\cdots u_\ell^{e_\ell}$, $c\in\mathbb{K}$, に対し、$e_1+\cdots+e_\ell$を$\boldsymbol{u}$に関する**全次数**(total degree)といい、$\mathrm{tdeg}(T)$と表す。$F(x,\boldsymbol{u})$と$G(x,\boldsymbol{u})$に対し、$\mathrm{res}(F,G)$はxに関する**終結式**(resultant)を表し、$\langle F,G\rangle$はFとGから生成される**イデアル**(ideal)を表す。

読者は多変数多項式に対する一般Hensel構成法を熟知しているとする。まず、拡張Hensel構成で最も重要な概念である**Newton多項式**を定義する。

定義 1 (Newton線とNewton多項式、正味Newton多項式)
$F(x,\boldsymbol{u})$の各項に$F(x,t\boldsymbol{u})$なる変換で従変数の全次数変数tを導入する。$F(x,t\boldsymbol{u})$の各項を$cx^it^ju_1^{j_1}\cdots u_\ell^{j_\ell}$とする；ここで、$c\in\mathbb{K}$, $j=j_1+\cdots+j_\ell$である。この項を(e_x,e_t)-面上の点(i,j)にプロットする。全てのプロット点を囲む凸包を$\mathcal{N}$と表す。$\mathcal{N}$の全底辺を時計周りに$\mathcal{N}_1,\ldots,\mathcal{N}_\rho$と表し、それぞれ**Newton線**と呼ぶ。各$i\in\{1,\ldots,\rho\}$に対し、$\mathcal{N}_i$上にプロットされた全ての項の和を**Newton多項式**と呼び、$\overline{F}_{\mathcal{N}_i}(x,\boldsymbol{u})$と表す。$\mathcal{N}_i$の左端の$x$座標を$n_i$とすれば$\overline{F}_{\mathcal{N}_i}$は$x^{n_i}$で割り切れる。$\overline{F}_{\mathcal{N}_i}/x^{n_i}$を$F_{\mathcal{N}_i}(x,\boldsymbol{u})$と表し**正味Newton多項式**と呼ぶ。 □

拡張Hensel構成は、ρ個のNewton線上で$\mathcal{N}_1\Rightarrow\mathcal{N}_2\Rightarrow\cdots\Rightarrow\mathcal{N}_\rho$の順に実行される。定義1でNewton多項式を定義するために全次数変数tを導入したが、拡張Hensel構成では各Newton線の傾きに依存してxと従変数$\boldsymbol{u}$の重み付けを行う(それにより式表現と計算が簡潔になる)。その重み付けにも変数tを用いる。$\mathcal{N}_1$の右端の座標点を(n_0,ν_0)、$\mathcal{N}_i$の左端の座標点を(n_i,ν_i)とすれば、$\mathcal{N}_i$の傾きは$\lambda_i=(\nu_{i-1}-\nu_i)/(n_{i-1}-n_i)$である。$\hat{n}_i$と$\hat{\nu}_i$は$\hat{n}_i>0$, $\hat{\nu}_i/\hat{n}_i=\lambda_i$, $\gcd(\hat{n}_i,\hat{\nu}_i)=1$を満たす整数とする。このとき、重み付きの多項式$\mathcal{F}(x,\boldsymbol{u},t)$, $\mathcal{F}_{\mathcal{N}_i}(x,\boldsymbol{u})$およびイデアル$\mathcal{I}_k$を次式で定義する。

$$\begin{cases} \mathcal{F}(x,\boldsymbol{u},t) & \stackrel{\mathrm{def}}{=} & t^{\hat{n}_i(\lambda_i n_i-\nu_i)}F(x/t^{\hat{\nu}_i},t^{\hat{n}_i}\boldsymbol{u}), \\ \mathcal{F}_{\mathcal{N}_i}(x,\boldsymbol{u}) & \stackrel{\mathrm{def}}{=} & t^{\hat{n}_i(\lambda_i n_i-\nu_i)}F_{\mathcal{N}_i}(x/t^{\hat{\nu}_i},t^{\hat{n}_i}\boldsymbol{u}), \\ \mathcal{I}_k & \stackrel{\mathrm{def}}{=} & \langle t^k\rangle, \quad k=1,2,3,\ldots. \end{cases} \tag{2.1}$$

上記で、$\mathcal{F}_{\mathcal{N}_i}(x,\boldsymbol{u})$が$t$を含まないのは誤植ではなく、そうなるように重み付けをしたのである。さらに、$\mathcal{I}_k$もiによらない。そして、拡張Hensel構成は、イデアル$\mathcal{I}_k$を$\mathcal{I}_1\Rightarrow\mathcal{I}_2\Rightarrow\mathcal{I}_3\Rightarrow\cdots$と上げて行われる(**Hensel リフティング**)。

拡張Hensel構成では、Newton多項式$\overline{F}_{\mathcal{N}}(x,\boldsymbol{u})$を$\mathbb{K}[x,\boldsymbol{u}]$内で因数分解し、その因子を初期因子としてHensel構成を行う。通常まず行うのが各Newton線にHensel因子1個が対応するように$F(x,\boldsymbol{u})$を分解することである(よって$\rho=1$の場合は不必要)。この分解を**Newton線上の最大因子の分離**という。最大因子分離の説明は本稿では割愛するので、興味ある読者は稿末の文献を参照されたい。。

各 Newton 線 $\mathcal{N}_i$ 上の最大 Hensel 因子 $F_i^{(k)}(x,\boldsymbol{u})$ を分離すると、次にはこの因子をさらに低次の Hensel 因子の積に分解する (以下、添字 i を省略する)。まず、$F_{\mathcal{N}}(x,\boldsymbol{u})$ を既約因子に因数分解する；次式で $\mathrm{cont}(F_{\mathcal{N}})$ は $F_{\mathcal{N}}$ の係因数 (係数の GCD) である。

$$\begin{cases} F_{\mathcal{N}}(x,\boldsymbol{u}) = \mathrm{cont}(F_{\mathcal{N}})\, G_1^{(0)}(x,\boldsymbol{u})\cdots G_r^{(0)}(x,\boldsymbol{u}), \\ \qquad \gcd(G_{j_1}^{(0)}, G_{j_2}^{(0)}) = 1 \quad (\forall j_1 \neq j_2). \end{cases} \tag{2.2}$$

次に、$l = 0, 1, \ldots, n-1$ に対して、次式を満たす **MY 補間式** $A_1^{(l)}, \cdots, A_r^{(l)}$ を計算する。

$$\begin{cases} A_1^{(l)}(x,\boldsymbol{u})\dfrac{F_{\mathcal{N}}(x,\boldsymbol{u})}{G_1^{(0)}(x,\boldsymbol{u})} + \cdots + A_r^{(l)}(x,\boldsymbol{u})\dfrac{F_{\mathcal{N}}(x,\boldsymbol{u})}{G_r^{(0)}(x,\boldsymbol{u})} = x^l, \\ \qquad \deg(A_i^{(l)}) < \deg(G_i^{(0)}) \quad (1 \leq \forall i \leq r). \end{cases} \tag{2.3}$$

次に、F の主係数、それを $\mathrm{lc}(F)$ と表す、の各因子を $\mathrm{lc}(F) = \mathrm{lc}(G_1^{(0)}\cdots G_r^{(0)})$ が成立するように $F, G_1^{(0)}, \ldots, G_r^{(0)}$ に分配する。具体的には、$\mathrm{lc}(F)$ の各因子の Newton 線上の項は $F_{\mathcal{N}}$ に入っているので、Newton 線上の項が $\mathrm{lc}(G_1^{(0)}), \ldots, \mathrm{lc}(G_r^{(0)})$ にどのように分配されているかを見て、各因子を割り振る。詳細は [5] を参照されたい。$\mathrm{cont}(F_{\mathcal{N}})$ は Hensel 構成した後で同様に分配する。そして、(2.1) により t-order を導入しイデアル $\mathcal{I}_k$ を定義する。この過程で、$F, G_1^{(0)}, \ldots, G_r^{(0)}$ をそれぞれ $\mathcal{F}, \mathcal{G}_1^{(0)}, \ldots, \mathcal{G}_r^{(0)}$ に変換する。

最後に、$\mathcal{G}_1^{(0)}, \ldots, \mathcal{G}_r^{(0)}$ を初期因子として、よく知られた次の計算式により、$\mathcal{G}_1^{(k)}, \ldots, \mathcal{G}_r^{(k)}$ を $k = 0 \to 1 \to 2 \to \cdots$ と逐次的に構成する (次式で $n = \deg(F(x,\boldsymbol{u}))$ である)。

$$\begin{cases} \delta\mathcal{F}^{(k)} &\equiv \mathcal{F}(x,\boldsymbol{u},t) - \mathcal{G}_1^{(k-1)}(x,\boldsymbol{u},t)\cdots\mathcal{G}_r^{(k-1)}(x,\boldsymbol{u},t) \pmod{t^{k+1}} \\ &= t^{k+\hat{n}(\nu-\lambda n)}\left[\delta f_{n-1}(\boldsymbol{u})x^{n-1} + \cdots + \delta f_0(\boldsymbol{u})\right], \\ \mathcal{G}_i^{(k)} &= \mathcal{G}_i^{(k-1)} + t^{k+\gamma_i}\textstyle\sum_{l=0}^{n-1} A_i^{(l)}(x,\boldsymbol{u})\delta f_l(\boldsymbol{u}) \quad (i=1,\ldots,r), \\ &\quad \text{where } \gamma_i \text{ is the } t\text{-order of } G_i^{(0)}(x/t^{\hat{\nu}}, t^{\hat{n}}\boldsymbol{u}). \end{cases} \tag{2.4}$$

拡張 Hensel 構成の従来算法は下記のような弱点を持つ。

弱点 1： MY 補間式 $A_i^{(l)}, B_i^{(l)}$ $(1 \leq i \leq r;\ 0 \leq l < \deg(F))$ の計算が非常に重い。さらに下記の例で具体的に示すように、与式が主変数に関して疎な場合、その疎性を全く利用していない。疎性を利用すれば算法を効率化できるはずである。

弱点 2： 拡張互除法で決まる MY 補間式の分母項は (例で示すように) 非常に大きいのが普通で、分子多項式との間で巨大な共通因子がキャンセルするのが普通である。我々は巨大な分母因子候補を経由しないで簡単に分母因子を計算したい。

弱点 3： 有理式係数の多項式を扱うのはかなり面倒でしかも計算が重い。我々は可能な限り多項式演算を用いて Hensel 因子の計算を簡単に行いたい。

本章の最後に、簡単な例で MY 補間式と Hensel 因子を具体的に見よう。そして、上記の弱点を具体的に確認する。

例 1 $F = GH + 3x^{10}y^5z^5$。ここで G と H は下記で、x が主変数である。

$$G = (x^5y^3 + 2z)(x^5z^3 - 2y) + x^3z^4, \quad H = (x^5z^3 + 3y)(x^5y^3 - 3z) + x^7y^6. \tag{2.5}$$

F は従変数のみならず主変数に関しても疎となるように設定した。

F はただ一本の Newton 線を持ち、Newton 多項式は $(x^5y^3 - 3z)(x^5y^3 + 2z)(x^5z^3 + 3y)(x^5z^3 - 2y)$ である。初期多項式は次のように設定する：$G_0 := (x^5y^3 - 3z)(x^5y^3 + 2z)$、$H_0 := (x^5z^3 + 3y)(x^5z^3 - 2y)$。$G_0$ と H_0 には x^{10}-, x^5- x^0-項が現れるのみだが、F は x^{20}-, x^{17}-, x^{15}-, x^{13}-, x^{12}-, x^{10}-, x^8-, x^7-, x^5-, x^3-, x^0-項を持つ。ゆえに、F の拡張 Hensel 構成において、従来法では最少でも 10 個の異なる l の値に対して MY 補間式 $A^{(l)}, B^{(l)}$ を計算する必要がある。一方、新算法ではイデアル $\langle G_0, H_0 \rangle$ のグレブナー基底が用いられるが、基底の要素多項式は x^{10}-, x^5-, x^0-項のみから成っている。

参考のため、$l = 0$ に対する MY 補間式 $A^{(0)}, B^{(0)}$ と $G^{(14)}, H^{(14)}$ を下記に示す (指標 14 とは、(2.1) における指標 k のことである)。なお、G_0 と H_0 の終結式は $\mathrm{res}(G_0, H_0) = 7776\,(3y^4 - 2z^4)^5\,(2y^4 - 3z^4)^5\,(y^4 + z^4)^{10}$ となり、分母因子はこの終結式の約数である。

$$A^{(0)} = \frac{(y^7z^3 - y^3z^7)x^5 - 2y^8 + 5y^4z^4 - 2z^8}{5(3y^4 - 2z^4)(2y^4 - 3z^4)}, \quad B^{(0)} = \frac{(-y^7z^3 + y^3z^7)x^5 - 3y^8 + 5y^4z^4 - 3z^8}{5(3y^4 - 2z^4)(2y^4 - 3z^4)}$$

$$G^{(14)} = G_0 + x^3z^4 + \frac{18y^5z^5(y^4 - z^4)x^5 + 12y^6z^6}{5(3y^4 - 2z^4)(2y^4 - 3z^4)} + \frac{45y^8z^{12}(y^4 - z^4)x^8 + 9y^5z^9(2y^8 - y^4z^4 + 2z^8)x^3}{25(3y^4 - 2z^4)^2(2y^4 - 3z^4)^2},$$

$$H^{(14)} = H_0 + x^7y^6 - \frac{18y^5z^5(y^4 - z^4)x^5 + 27y^6z^6}{5(3y^4 - 2z^4)(2y^4 - 3z^4)} - \frac{45y^8z^{12}(y^4 - z^4)x^8 + 27y^5z^9(9y^8 - 17y^4z^4 + 9z^8)x^3}{25(3y^4 - 2z^4)^2(2y^4 - 3z^4)^2}.$$

□

上記の計算は、著者の一人 (D.I) が Mathematica 上にインプリメントしたパッケージを用いて実行した。上記の数式は簡潔にまとまっているが、それは Mathematica の GCD や因数分解ルーチンを縦横に用いて簡潔にしたからである。そのため、計算はかなり面倒で非常に遅い (タイミングデータは 4 章で示す)。

3 グレブナー基底を利用した拡張 Hensel 構成

式 (2.3) と (2.4) を見ると、Hensel 構成のみならず MY 補間式も r 個の初期因子を一緒に扱えば効率よく計算できると思える。しかし、実際には、MY 補間式は各 $i \in \{1, \ldots, r\}$ について別々に 2 因子 $G_i^{(0)}, F_{\mathcal{N}}/G_i^{(0)}$ から計算している。式 (2.4) でも各 $G_i^{(k)}$ は指標 i 毎に独立に計算する。故に、初期因子が 2 個の場合を調べておけば十分であり、本章では初期因子は互いに疎な多項式 G_0, H_0 であるとする。この場合、(2.4) は下記のようになる。

$$\begin{cases} \mathcal{F}(x, \boldsymbol{u}, t) \equiv \mathcal{G}^{(k)}(x, \boldsymbol{u}, t)\mathcal{H}^{(k)}(x, \boldsymbol{u}, t) \pmod{t^{k+1}}, \\ t^k \delta F^{(k)} = \delta\mathcal{F}^{(k)} \equiv \mathcal{F} - \mathcal{G}^{(k-1)}\mathcal{H}^{(k-1)} \pmod{t^{k+1}}. \end{cases} \tag{3.1}$$

$$\mathcal{G}^{(k)} = \mathcal{G}^{(k-1)} + t^k \delta G^{(k)}, \quad \mathcal{H}^{(k)} = \mathcal{H}^{(k-1)} + t^k \delta H^{(k)}, \quad \mathcal{G}^{(0)} = G_0, \ \mathcal{H}^{(0)} = H_0. \tag{3.2}$$

$$\delta H^{(k)} G_0 + \delta G^{(k)} H_0 = \delta F^{(k)}, \quad \deg(\delta G^{(k)}) < \deg(G_0), \ \deg(\delta H^{(k)}) < \deg(H_0). \tag{3.3}$$

なお、前章では主係数の分配問題を記述したが、本章では既に分配済みと仮定する。

3.1 イデアル $\langle G_0, H_0\rangle$ のグレブナー基底

まず、グレブナー基底に関する術語を定める。$x^d u_1^{e_1}\cdots u_\ell^{e_\ell}$ を**べき積**という。$\mathbb{K}[x,\boldsymbol{u}]$ の全てのべき積を一意的に順序づける整列順序 (最低順序項は 0) を $\succ$ で表し**項順序**という。あとで具体的に定めるが、当面、$\succ$ は $x \succ u_1,\ldots,u_\ell$ を満たすとする。$F(x,\boldsymbol{u})$ が $\mathbb{K}[x,\boldsymbol{u}]$ の多項式のとき、F を単項式 c_iT_i (T_i はべき積) の和として次のように表す。

$$F(x,\boldsymbol{u}) = c_1T_1 + c_2T_2 + \cdots + c_mT_m, \quad c_i \in \mathbb{K}, \quad T_1 \succ T_2 \succ \cdots \succ T_m. \tag{3.4}$$

このとき、$c_1T_1, c_1, T_1, F - c_1T_1$ をそれぞれ**頭項**, **頭係数**, **頭べき積**, **残余** と言い、$\mathrm{htm}(F)$, $\mathrm{hc}(F)$, $\mathrm{hpp}(F)$, $\mathrm{rest}(F)$ と表す。F が他の多項式 G に対して G-**既約**とは、$\mathrm{hpp}(G)$ が $\mathrm{hpp}(F)$ を割り切らないことをいう。F と G の **S 多項式** $\mathrm{Spol}(F,G)$ とは $\mathrm{Spol}(F,G) = [L/\mathrm{htm}(F)]\,F - [L/\mathrm{htm}(G)]\,G$、$L = \mathrm{lcm}(\mathrm{hpp}(F),\mathrm{hpp}(G))$ で定義される多項式である (lcm は最小公倍子演算)。F の G による **M 簡約** $\mathrm{Mred}(F,G)$ とは $\mathrm{Mred}(F,G) = F - [\mathrm{htm}(F)/\mathrm{htm}(G)]\,G$ で定義される；$\mathrm{Mred}(F,G)$ は $\mathrm{hpp}(G)$ が $\mathrm{hpp}(F)$ を割り切るときにのみ定義できる。F に対して $F := \mathrm{Mred}(F,G)$ を繰り返すと F が G-既約になるが、このときの簡約結果を $\mathrm{rem}(F,G)$ と表す。

イデアル $\langle G_0, H_0\rangle$ の、項順序 $\succ$ に関するグレブナー基底を Γ とする；基底の各要素を $\widehat{G}_i$ とすれば、$\widehat{G}_i$ は G_0 と H_0 の線形結合で下記のように表される。

$$\begin{aligned} \Gamma &= \{\widehat{G}_1,\ldots,\widehat{G}_s\} \subset \mathbb{K}[x,\boldsymbol{u}], \quad \widehat{G}_1 \prec \cdots \prec \widehat{G}_s, \\ \widehat{G}_i &= A_iG_0 + B_iH_0, \quad A_i, B_i \in \mathbb{K}[x,\boldsymbol{u}] \quad (i = 1,\ldots,s). \end{aligned} \tag{3.5}$$

(A_i, B_i) を $\widehat{G}_i$ **に対するシジジー**と呼び、多項式 $S_i(\gamma,\eta) \in \mathbb{K}[\gamma,\eta]$ で表す：$S_i(G_0,H_0) = \widehat{G}_i$. なお、拡張 Hensel 構成においては変数 t を用いて (2.1) のように x と $\boldsymbol{u}$ を重み付けするが、グレブナー基底の計算ではこの重み付けは不必要なことに注意されたい。

補題 1 ($\widehat{G}_1$ **が持つ重要な性質**)
1) $\widehat{G}_1 \in \mathbb{K}[\boldsymbol{u}]$. *2)* $\widehat{G}_{i\geq 2} \notin \mathbb{K}[\boldsymbol{u}]$ ならば $\widehat{G}_1 \mid \mathrm{res}(G_0,H_0)$.

証明 G_0 と H_0 は互いに素ゆえ、x に関する終結式, $R \stackrel{\mathrm{def}}{=} \mathrm{res}(G_0,H_0)$, は 0 でなく。$R \in \langle G_0.H_0\rangle$ ゆえ、R は Γ の要素で 0 に M 簡約される。そのためには Γ は $\mathbb{K}[\boldsymbol{u}]$ の元を含む必要があり、1) が導かれる。2) は M 簡約で 0 になることを言い換えただけである。 □

3.2 $\delta G^{(k)}$ と $\delta H^{(k)}$ の多項式部分の計算法

簡単のため、$\delta F^{(k)} \in \mathbb{K}[x,\boldsymbol{u}]$ であるとする。$\delta F^{(k)}$ を Γ の要素で M 簡約した結果を

$$\delta F^{(k)} = \delta\widehat{F}^{(k)} + \delta R^{(k)}, \quad \delta\widehat{F}^{(k)} = q_1\widehat{G}_1 + \cdots + q_s\widehat{G}_s, \quad q_i \in \mathbb{K}[x,\boldsymbol{u}] \ \text{ for } \ \forall i, \tag{3.6}$$

とする。ここで、$\delta R^{(k)} \in \mathbb{K}[x,\boldsymbol{u}]$ は Γ-既約である。$\delta\widehat{F}^{(k)} \in \langle G_0,H_0\rangle$ だが $\delta R^{(k)} \notin \langle G_0,H_0\rangle$ であり、$\delta\widehat{F}^{(k)}$ と $\delta R^{(k)}$ は一意的に定まる ($q_1,\ldots,q_s$ は一意的でない)。$\delta\widehat{F}^{(k)} = \widehat{h}G_0 + \widehat{g}H_0$ を満たす $\mathbb{K}[x,\boldsymbol{u}]$ の多項式 $\widehat{h}$ と $\widehat{g}$ は次のように計算できる。

$\delta\widehat{F}^{(k)} = \widehat{h}G_0 + \widehat{g}H_0$ を満たす多項式 $\widehat{h}$ と $\widehat{g}$ の計算法：

$\delta\widehat{F}^{(k)}$ の中の各 $\widehat{G}_i$ $(1 \leq i \leq s)$ を $S_i(\gamma, \eta)$ で置き換えると、$\delta\widehat{F}^{(k)}$ は $\mathbb{K}[\gamma, \eta, x, \boldsymbol{u}]$ 内の多項式となる。この $\delta\widehat{F}^{(k)}$ の変数 γ と η の係数がそれぞれ $\widehat{h}$ と $\widehat{g}$ である。 □

次数条件 $\deg(\widehat{g}) < \deg(G_0)$ と $\deg(\widehat{h}) < \deg(H_0)$ が満たされれば、$\delta G^{(k)} = \widehat{g}$、$\delta H^{(k)} = \widehat{h}$ であるが、そうでない場合も頻繁にある (こちらの方が多いくらい)。

補題 2 ($\widehat{h}$ と $\widehat{g}$ の次数低減法)
上記の方法で計算した $\widehat{h}$ と $\widehat{g}$ が $\deg(\widehat{h}) \geq \deg(H_0)$, $\deg(\widehat{g}) \geq \deg(G_0)$ であるとする。もし $\delta H^{(k)}, \delta G^{(k)} \in \mathbb{K}[x, \boldsymbol{u}]$ が存在し、$\delta\widehat{F}^{(k)} = \delta H^{(k)}G_0 + \delta G^{(k)}H_0$ を満たし、かつ $\deg(\delta H^{(k)}) < \deg(H_0)$, $\deg(\delta G^{(k)}) < \deg(G_0)$ ならば、$\delta H^{(k)} = \mathrm{rem}(\widehat{h}, H_0)$, $\delta G^{(k)} = \mathrm{rem}(\widehat{g}, G_0)$ である。ここで、rem は M 簡約で計算しても除算で計算してもよい。

証明 $\delta\widehat{F}^{(k)}$ が二通りに表されることから $(\widehat{h} - \delta H^{(k)})G_0 = (\delta G^{(k)} - \widehat{g})H_0$ を得る。G_0 と H_0 は互いに素ゆえ、この等式から $G_0 \mid (\widehat{g} - \delta G^{(k)})$, $H_0 \mid (\widehat{h} - \delta H^{(k)})$ を得る。次数条件よりこれらはそれぞれ $\delta G^{(k)} = \mathrm{rem}(\widehat{g}, G_0)$, $\delta H^{(k)} = \mathrm{rem}(\widehat{h}, H_0)$ を意味する。 □

定理 1 (主定理 1：$\delta G^{(k)}, \delta H^{(k)}$ の多項式部分に関する定理)
$\delta R^{(k)}$ と $\widehat{h}, \widehat{g}$ が上述のように決まったとする。もしも $\delta R^{(k)} = 0$ かつ $\widehat{h}, \widehat{g}$ が補題 2 により次数低減できれば、$\delta G^{(k)}$ と $\delta H^{(k)}$ は $\mathbb{K}[x, \boldsymbol{u}]$ の要素として計算できる。もしも $\delta R^{(k)} \neq 0$ あるいは $\widehat{h}, \widehat{g}$ が次数低減できなければ、$\delta G^{(k)}, \delta H^{(k)}$ には $\boldsymbol{u}$ の有理式が現れる。

証明 多項式 $\widehat{h}, \widehat{g}$ の計算法と次数低減法は上に述べた。よって、定理の前半部は正しい。次に、$\delta R^{(k)} \neq 0$ であるが $\delta G^{(k)}, \delta H^{(k)} \in \mathbb{K}[x, \boldsymbol{u}]$ と仮定してみる。すると、$\delta R^{(k)} = \delta F^{(k)} - \delta\widehat{F}^{(k)} = (\delta H^{(k)} - \widehat{h})G_0 + (\delta G^{(k)} - \widehat{g})H_0$ なので $\delta R^{(k)} \in \langle G^{(0)}, H^{(0)} \rangle$ となり、$\delta R^{(k)}$ が Γ-既約との仮定に反する。$\widehat{h}, \widehat{g}$ の次数低減ができない場合は、後述の "強制次数低減法" で示すように、$\delta G^{(k)}, \delta H^{(k)}$ には $\mathrm{lc}(G_0), \mathrm{lc}(H_0)$ を分母因子とする有理式が現れる。 □

3.3 有理式係数を如何に扱うか？

本節では、$\widehat{h}, \widehat{g}$ の次数低減ができない場合は $\delta\widehat{F}^{(k)}$ を $\delta R^{(k)}$ に繰り込み、次数低減が可能か否かに拘わらず $\delta R^{(k)} \neq 0$ と仮定する。なお、以下では分母項は D と表す。

有理式係数に対する著者らの方針：

分母因子 $D \in \mathbb{K}[\boldsymbol{u}]$ は積 $D\,\delta R^{(k)}$ がイデアル $\langle G_0, H_0 \rangle$ の要素となるように、即ち $D\,\delta R^{(k)} = hG_0 + gH_0$ を満たす $h, g \in \mathbb{K}[x, \boldsymbol{u}]$ が存在するように決める。ついで、$\delta G^{(k)}, \delta H^{(k)}$ において D をシステム変数 $\%D$ で置き換え、$\delta G^{(k)}$ と $\delta H^{(k)}$ を $\mathbb{K}[\%D, x, \boldsymbol{u}]$ の多項式に変換する。D は可能な限り低順序の多項式になるように定め、変数 $\%D$ は $\%D \succ x \succ u_1, \ldots, u_\ell$ と順序づける。 □

$\delta R^{(k)} = hG_0 + gH_0$ を満たす多項式 h と g の計算法：

$D = \widehat{G}_1/\gcd(\widehat{G}_1, \delta R^{(k)})$ と定め、$D\,\delta R^{(k)}$ の因子 $\widehat{G}_1$ を $A_1\gamma + B_1\eta$ で置き換えると、h, g は置き換えられた式のそれぞれ γ, η の係数である。得られた h と g は補題 2 により次数低減を試みるが、失敗すれば下記のように強制次数低減する。 □

次数低減に失敗した h, g の強制次数低減法：

たとえば g を G_0 で次数低減する場合、擬除算のように $D' = \mathrm{lc}(G_0)/\gcd(\mathrm{lc}(g), \mathrm{lc}(G_0))$ を g に掛ければ $\mathrm{ltm}(g)$ を消去できる。このように、除算が可能なように最低順位の多項式 ($\in \mathbb{K}[\boldsymbol{u}]$) を掛けて剰余を計算する；掛けられる多項式の積も D' と表す。 □

定理 2 (主定理 2：$\delta G^{(k)}, \delta H^{(k)}$ の有理式部分に関する定理)

D' は強制次数低減で導入される分母因子とする (強制次数低減をしなければ $D' = 1$)。まず、分母項 D を $D = \widehat{G}_1/\gcd(\widehat{G}_1, \delta R^{(k)})$ と定め、$D'D\,\delta R^{(k)} = \delta H^{(k)}G_0 + \delta G^{(0)}H_0$, $\deg(\delta H^{(k)}) < \deg(H_0)$, $\deg(\delta G^{(k)}) < \deg(G_0)$ を満たす多項式 $\delta H^{(k)}, \delta G^{(k)}$ を計算する。最後に、$C = \gcd(D'D, \mathrm{cont}(\delta G^{(k)}), \mathrm{cont}(\delta H^{(k)})) \neq 1$ であれば、$D'D := D'D/C$ とする。このとき、$\delta G^{(k)}$ と $\delta H^{(k)}$ のあらゆる有理式係数は $N_j/D'D$, $N_j \in \mathbb{K}[\boldsymbol{u}]$, と表すことができる。

証明 $D\,\delta R^{(k)}$ は $\mathbb{K}[x, \boldsymbol{u}]$ 内の多項式で、$\widehat{G}_1$ を因子として持つ。よって、$D\,\delta R^{(k)} = hG_0 + gH_0$ を満たす多項式 $h, g \in \mathbb{K}[x, \boldsymbol{u}]$ を計算できる。さらに、(必要なら) 強制次数低減により、$D'D\,\delta R^{(k)} = \delta H^{(k)}G_0 + \delta G^{(0)}H_0$ を満たす $\delta G^{(k)}, \delta H^{(k)} \in \mathbb{K}[x, \boldsymbol{u}]$ が計算できる。この式の両辺を $D'D/C$ で割れば定理が得られる。 □

3.4 項順序とシジジー計算について

3.1 節では従変数に関しては項順序を定めなかった。もしも項順序として**辞書式順序** (lexicographic order) を選ぶなら、計算は遅いことが多い。そこで著者らは**従変数全次数順序** (sub-variable total-degree order; stdeg と略記) と呼ぶ項順序を採用する。stdeg 順序は次式で定義される。

$$x^d u_1^{e_1} \cdots u_\ell^{e_\ell} \longleftrightarrow (d, \textstyle\sum_{i=1}^{\ell} e_i, e_1, \ldots, e_\ell). \tag{3.7}$$

実際に計算してみて、stdeg 順序は我々の計算には非常に適していることがわかった。

上述したように、本稿の算法ではグレブナー基底のみならずシジジーも不可欠である。シジジー算法は簡単でよく知られているが [1]、従来算法はグレブナー基底の計算よりも遥かに時間を要することが多い。そこで、シジジーの効率的算法を呈示する。

呈示する方法は一般の場合にも適用できるが、本稿では $\langle G_0, H_0 \rangle$ について説明する。$P_1 = G_0$, $P_2 = H_0$ から出発し、グレブナー基底算法で $P_3 \to P_4 \to \cdots \to P_k$ と順に非零多項式が生成されるとする。従来のシジジー算法は、$(A_1, B_1) = (1, 0)$, $(A_2, B_2) = (0, 1)$ から出発し、多項式生成と並行して $(A_3, B_3) \to (A_4, B_4) \to \cdots \to (A_k, B_k)$ と計算する。この方法は、i) 計算される P_i はその後の M 簡約で消去されることが多く、その場合にはシジジー計算は無駄になる、ii) シジジーは基底計算の進行とともに大きくなるが、基底

計算の多く、特に終盤の計算の大部分は $\mathrm{Spol}(P_i, P_j)$ が 0 に M 簡約されることの確認で、シジジーには無関係である、の 2 点で非常に無駄な計算をしている。

我々は、基底計算の最中は『シジジーの計算手順を定める』だけとし、基底計算終了後に『"手順的シジジー" を実際のシジジーに変換する』ことにする。この方法では手順的シジジーの計算が非常に軽いことが必要だが、次のようにすれば非常に軽くなる。

グレブナー基底計算は M 簡約、S 多項式生成、多項式の規格化の反復である；規格化に関しては、本稿では頭係数を 1 にすることとする。下記では、多項式 F と G に対し、$\mathrm{htm}(F) = f_1 S_1$, $\mathrm{htm}(G) = g_1 T_1$ とし、$\#_F$ と $\#_G$ はそれぞれ F と G に付けられた式番号とする；P が i 番目に生成された非零多項式なら、式番号として i を P に付与する。なお、ある多項式がそのときの中間基底で既約となればその時点で式番号を付与し、それが後に別の基底で簡約された場合には別の式番号の多項式として扱う。そうすれば、番号の若いものから順にシジジー計算を実行できる (下記プログラムはそうなっている)。

さて、手順的シジジーであるが、それは下記のように非常に簡単なもので、計算が進み実際のシジジーがいかに巨大化しようと、全く巨大化することはない。

次の "手順的シジジー" を配列 %Syzygy に収納する：

On $\mathrm{Mred}(F, G)$: (Mred ($\#_F$, (0, .., 0), 1) ($\#_G$, S_1/T_1, $-f_1/g_1$)),
On $\mathrm{Spol}(F, G)$: (Spol ($\#_F$, L/S_1, g_1) ($\#_G$, L/T_1, $-f_1$)), where $L := \mathrm{lcm}(S_1, T_1)$,
On Normalization of F : (Nmlz $1/f_1$).

上記の "($\#_G$, S_1/T_1, $-f_1/g_1$)" 等を "IPC 三つ組" と呼ぶ。F は大抵複数回 M 簡約されるが、各 M 簡約毎に対応する IPC 三つ組を F のシジジーの尾部に付加する。同じことは $\mathrm{Spol}(F, G)$ が M 簡約される場合にも行う。規格化は F を可能なかぎり M 簡約した結果が非零の場合に行うので、"(Nmlz $1/f_1$)" は F の手順的シジジーの最後尾の要素である。

つぎに、手順的シジジーの実際シジジーへの変換について説明する。手順的シジジーの収納場所として配列 %Syzygy を準備し、F の手順的シジジーを %Syzygy[$\#_F$] に収める。グレブナー基底の計算で生成された多項式の最大式番号を #mx とする。手順的シジジーを実際シジジーに変換する主プロシジャ `convPsyz2Asyz` は、$i = 3$ to #mx に対して順にサブプロシジャ `evalPsyz` を呼び、`evalPsyz` で計算された実際シジジーを %Syzygy[i] に収納する。`evalPsyz` の働きについては下記プロシジャを見られたい。最後にプロシジャ `IPC2Asyz` は与えられた IPC 三つ組 (#, PowP, Coef) を実際シジジーに変換する。具体的には、単項式 Coef×PowP を (既に実際シジジーが収納された) %Syzygy[#] に掛けるだけである。下記プロシジャにおいて `first`(l) と `rest`(l) は、IPC 三つ組を要素とするリスト l のそれぞれ第一要素と第一要素を除いたリストを答とするプロシジャである。

```
Procedure  convPsyz2Asyz(%Syzygy, #mx) ==
    for  i = 3 to #mx do
        { Asyz := evalPsyz(%Syzygy[i]);
          store Asyz to %Syzygy[i] }.
```

```
Procedure  evalPsyz(Psyz) ==
    begin  Asyz := IPC2Asyz(first(Psyz));  goto next;
    loop:  Asyz := Asyz + IPC2Asyz(first(Psyz));
    next:  if (Psyz := rest(Psyz)) ≠ ( ) then goto loop;
           return Asyz;      end.
```

実は上記のプロシジャはまだ無駄を含む。#mx個の多項式のうちグレブナー基底に残るのは一部である。そこで、Γ の要素に対してのみ式番号の小さいものから順に evalPsyz を適用するならば、Γ の要素のシジジーに不必要なシジジーを計算しなくて済む。ただし、この場合には、番号 k の多項式のシジジー計算で Γ の要素でない番号 j の多項式, $j<k$, のシジジーが必要になることもある。この場合には evalPsyz を再帰的に適用する。

実験では上記のシジジー算法は十分に効率的である。実際、計算時間はグレブナー基底の計算時間のせいぜい数割である。

4 タイミングデータと新算法に対する注釈

前章に記述した算法は全て数式処理システム GAL(佐々木研を中心に開発した数式処理システム) に実装され、簡単ながらいくつかの例題でテストされた。実験に用いた例題は、2 章の例 1 とその類題である。

実験に用いた多項式 $F\in\mathbb{Q}[x,y,z]$ は下記とする。

$$F=\{(x^5y^3+2z)(x^5z^3-2y)+x^3z^4\}\times\{(x^5z^3+3y)(x^5y^3-3z)+x^7y^6\}+3x^{10}y^5z^5. \quad (4.1)$$

F はただ1本のニュートン線を持ち、ニュートン多項式は $(x^5y^3-3z)(x^5y^3+2z)(x^5z^3+3y)(x^5z^3-2y)$ である。ニュートン線の傾きが 2/5 なので、x と y,z にはそれぞれ重み (-2) と (+5) が付与される (t を Hensel 構成の位数とするとき、$x\to x/t^2$、$(y,z)\to(t^5y,t^5z)$ と重み付ける)。x に関する次数を 10 とする初期因子の選び方は次の 3 通りある。

$$\begin{aligned}
&\text{選択 }\mathbf{A}: \ G_0=(x^5y^3+2z)(x^5z^3-2y),\quad H_0=(x^5y^3-3z)(x^5z^3+3y),\\
&\text{選択 }\mathbf{B}: \ G_0=(x^5y^3-3z)(x^5z^3-2y),\quad H_0=(x^5y^3+2z)(x^5z^3+3y),\\
&\text{選択 }\mathbf{C}: \ G_0=(x^5y^3-3z)(x^5y^3+2z),\quad H_0=(x^5z^3+3y)(x^5z^3-2y),
\end{aligned} \quad (4.2)$$

選択 A が 2 章で用いた例題で、(4.1) が示唆するように $\delta G^{(4)}$ に単項式 x^3z^4 が現れ、$\delta H^{(6)}$ に単項式 x^7y^6 が現れ、t^{10} から有理式係数項が現れる。選択 B と選択 C ではいずれも t^4 から有理式係数項が現れる。あとでみるように、選択 C が最も複雑な振舞いをする。

選択 A, 選択 B, 選択 C におけるグレブナー基底をそれぞれ Γ_A, Γ_B, Γ_C とする。

$$\Gamma_A:\begin{cases}
\widehat{G}_{11} &= 1y^9z-13/6y^5z^5+1yz^9,\\
\widehat{G}_3 &= 1x^5y^4-1x^5z^4-1yz,\\
\widehat{G}_{12} &= 1x^5z^8-6y^5z+7yz^5,\\
\widehat{G}_2 &= 1x^{10}y^3z^3+3x^5y^4-3x^5z^4-9yz,\\
\widehat{G}_{12} &= 1x^{10}z^7+3x^5y^5-2x^5yz^4-9y^2z.
\end{cases} \quad (4.3)$$

$$\Gamma_B : \begin{cases} \widehat{G}_6 = 1y^5z + 1yz^5, \\ \widehat{G}_3 = 1x^5y^4 + 1x^5z^4, \\ \widehat{G}_2 = 1x^{10}y^3z^3 + 3x^5y^4 + 2x^5z^4 + 6yz, \\ \widehat{G}_4 = 1x^{10}z^7 - 3x^5y^5 - 2x^5yz^4 - 6y^2z. \end{cases} \tag{4.4}$$

$$\Gamma_C : \begin{cases} \widehat{G}_5 = 1y^{12} - 7/6y^8z^4 - 7/6y^4z^8 + 1z^{12}, \\ \widehat{G}_7 = 1x^5y^8 - 1x^5z^8 - 1y^5z - 1yz^5, \\ \widehat{G}_3 = 1x^5y^7z^3 + 1x^5y^3z^7 - 6y^8 + 6z^8, \\ \widehat{G}_4 = 1x^5y^4z^7 + 1x^5z^11 - 6y^9 + 1y^5z^4 + 7yz^8, \\ \widehat{G}_2 = 1x^{10}z^6 + 1x^5yz^3 - 6y^2, \\ \widehat{G}_1 = 1x^{10}y^6 - 1x^5y^3z - 6z^2. \end{cases} \tag{4.5}$$

まず、本稿で与えた新算法と旧算法を計算時間の観点から比較する。ただし、両算法の共通部分はニュートン多項式の決定と因数分解しかなく、旧算法の特徴は MY 補間式で新算法の特徴はグレブナー基底なので、両者のタイミングデータは別の表として与える。新算法は前記 GAL に実装され、計算には Intel(R), U2300, 1.20GHz を搭載したパソコン (OS は Linux 3.4.100) を用いた。旧算法は Mathematica に実装され、計算には Intel(R), B800, 1.50GHz を搭載したパソコン (OS は MS Windows 7) を用いた。また、計算時間は各ユニット演算を 100 回 繰り返したときの平均時間である。

旧算法 (MY 補間式を用いる) のタイミングデータ (msec)

	選択 A	選択 B	選択 C
F_N 計算と因数分解	同右	133.38	同左
MY 補間式の計算	567.07	318.24	472.22
4 次因子の計算	34.630	33.080	35.720
6 次因子の計算	59.280	63.340	67.710
8 次因子の計算		110.14	115.75
10 次因子の計算	99.920	158.65	170.36

新算法 (グレブナー基底を用いる) のタイミングデータ (msec)

	選択 A	選択 B	選択 C
F_N 計算と因数分解	同右	23.19	同左
Γ とシジジー計算	0.670	0.250	0.330
4 次因子の計算	0.018	0.190	0.270
6 次因子の計算	0.175	0.360	0.830
8 次因子の計算		0.400	0.830
10 次因子の計算	0.554	1.160	3.070

旧算法に比較して新算法の高速さは驚嘆に値する。もっと大きな例でどうなるか、気になるところである。新算法での計算時間の大部分をニュートン多項式の因数分解が占めることに驚かれるだろうが、その理由は用いた算法が一般 Hensel 構成に基づく算法だからであろう。上例ではニュートン多項式は主変数と従変数の両者に関して疎で、従来算法には悪条件であるが、GAL には拡張 Hensel 構成に基づく算法はまだ装備されてない。近い将来、因数分解は疎多項式用の新算法で一気に効率化する予定なので、タイミングデータでは因数分解時間は特別視して欲しい。

つぎに、初期因子の三つの選択に対して、分母因子がどう定まったかを簡単に述べる。

選択 A $6\widehat{G}_{11} = yz\,(6y^8-13y^4z^4+6z^8)$, $\delta R^{(10)} = 18y^3z^3$ ゆえ、$\widehat{G}_{11}/\gcd(\widehat{G}_{11}, \delta R^{(10)}) = (6y^8-13y^4z^4+6z^8) = (3y^4-2z^2)\,(2y^4-3z^4)$ と分母因子が定まる。

選択 B $\widehat{G}_6 = yz(y^4+z^4)$。$k=4$ では $\delta R^{(4)} = -5z^8z^8-15x^3yz^5$ より $\widehat{G}_6/\gcd(\widehat{G}_6, \delta R^{(4)}) = y^5+yz^4$。そのあと $\delta H^{4)}, \delta G^{(4)}$ を計算すれば $\gcd(\mathrm{cont}(\delta H^{4)}), \mathrm{cont}(\delta G^{(4)})) = y$ となるので、分母因子が y^4+z^4 と定まる。$k=6,8,10$ でも同様である。

選択 C $6\widehat{G}_5 = 6y^{12}-7y^8z^4-7y^4z^8+6z^{12} = (y^4+z^4)\,(3y^4-2z^4)\,(2y^4-3z^4)$。
$k=4,6,8$ では $\gcd(\widehat{G}_5, \delta R^{(k)}) = 1$ だが、$\gcd(\mathrm{cont}(\delta H^{k)}), \mathrm{cont}(\delta G^{(k)})) = 6y^8-13y^4z^4+6z^8$ なので、分母因子が y^4+z^4 と定まる。一方、$k=10$ では、$\gcd(\mathrm{cont}(\delta H^{10)}), \mathrm{cont}(\delta G^{(10)})) = 1$ なので、分母因子は $\widehat{G}_5$ に変化する。

上記より、分母因子は理論通り紆余曲折しつつ定まることがわかる。なお、三つの選択のいずれでも、$\widehat{h}, \widehat{g}, h, g$ の次数低減には補題 2 で十分で、強制次数低減は不必要だった。

当初、グレブナー基底の計算に時間がかかること、特にシジジー計算の不経済性を危惧していたが、シジジー計算に関しては本稿の効率化で十分だろう。グレブナー基底計算に関しては、今後の理論展開にも依るが、まだ効率化の余地があると思っている。

謝辞 本研究は科研費 (課題番号 15K00005) の援助で遂行された。

参 考 文 献

[1] B. Buchberger: Gröbner bases: an algorithmic methods in polynomial ideal theory. in *Multidimensional Systems Theory*, Chap. 6. Reidel Publishing, 1985.

[2] K.O. Geddes, S.R. Czapor and G. Labahn: *Algorithms for computer algebra.* Kluwer Academic Publishers, 1992.

[3] J. von zur Gathen and J. Gerhard: *Modern Computer Algebra.* Cambridge Univ. Press, 1999.

[4] J. von zur Gathen and E. Kaltofen: Factoring sparse multivariate polynomials. J. Comp. System Sci. **31**, 265-287 (1985).

[5] D. Inaba: Factorization of multivariate polynomials by extended Hensel construction. *ACM SIGSAM Bulletin*, **39**(1), 2-14 (2005).

[6] D. Inaba and T. Sasaki: A numerical study of extended Hensel series. *Proc. SNC'07*, J. Verschede and S.T. Watt (Eds.), ACM Press, 103-109 (2007).

[7] E. Kaltofen: Sparse Hensel lifting. *Proc. EUROCAL'85*, Springer-Verlag LNCS **2**, 4-17 (1985).

[8] J. de Kleine, M. Monagan and A. Wittkopf: Algorithms for the non-monic case of the sparse modular GCD algorithm, *Proc. ISSAC 2005*, 124-131 (2005).

[9] T.-C. Kuo: Generalized Newton-Puiseux theory and Hensel's lemma in $\mathbf{C}[[x, y]]$. *Canad. J. Math.*, **XLI**, 1101-1116 (1989).

[10] J. Moses and D.Y.Y. Yun: The EZGCD algorithm. *Proc. 1973 ACM National Conference*, ACM, 159-166 (1973).

[11] T. Sasaki and D. Inaba: Hensel construction of $F(x, u_1, \ldots, u_\ell)$, $\ell \geq 2$, at a singular point and its applications. *ACM SIGSAM Bulletin*, **34**(1), 9-17 (2000).

[12] T. Sasaki and D. Inaba: A study of Hensel series in general case. *Proceedings of SNC'11*, M. Moreno Maza ((Ed.), ACM Press, 34-43 (2011).

[13] M. Sanuki, D. Inaba and T. Sasaki: Computation of GCD of Sparse Multivariate Polynomial by Extended Hensel Construction. *Proceedings of SYNASC2015 (Symbolic and Numeric Algorithms for Scientific Computing)*, IEEE Computer Society (in printing).

[14] T. Sasaki and F. Kako: Solving multivariate algebraic equation by Hensel construction. Preprint of Univ. Tsukuba, March, 1993.

[15] T. Sasaki and F. Kako: Solving multivariate algebraic equation by Hensel construction. Japan J. Indust. Appl. Math., **16**(2), 257-285 (1999). (This is almost the same as [14]: the delay of publication is due to very slow reviewing process.)

[16] J.T. Schwarz: Fast probabilistic algorithms for verification of polynomial identities. J. ACM **27**, 701-717 (1980).

[17] P.S. Wang and L. P. Rothschild: Factoring multivariate polynomials over the integers. Math. Comp. **29**, 935-950 (1975).

[18] P.S. Wang: An improved multivariate factoring algorithm. Math. Comp. **32**, 1215-1231 (1978).

[19] R. Zippel: Probabilistic algorithm for sparse polynomials. *Proc. EUROSAM'79*, Springer-Verlag LNCS **72**, 216-226 (1979).

[20] R. Zippel: Newton's iteration and the sparse Hensel lifting (extended abstract), *Proc. SYMSAC'81*, 68-72 (1981).