拡大体 $\mathbf{GF}(p^n)$ 上の数体篩法における 3 次元 Lattice Sieve の構成

早坂 健一郎 * 青木 和麻呂 † 小林 鉄太郎 † 高木 剛 ‡

*九州大学大学院数理学府
819-0395 福岡市西区元岡 744

†NTT セキュアプラットフォーム研究所
180-8585 東京都武蔵野市緑町 3-9-11

‡ 九州大学 マス・フォア・インダストリ研究所
819-0395 福岡市西区元岡 744

あらまし ペアリング暗号は，拡大体 $\mathrm{GF}(p^n)$ 上の離散対数問題を安全性の基礎の一つとする．素体 $\mathrm{GF}(p)$ 上の離散対数問題に対する現在漸近的に最速の解法として数体篩法 (JL03-NFS) が知られている．一方 Joux らは CRYPTO 2006 において，拡大体 $\mathrm{GF}(p^n)$ 上へ拡張した数体篩法 (JLSV06-NFS) を考案した．JL03-NFS では，2 次元の篩処理 (2 次元 lattice sieve) を用いることで十分であったが，JLSV06-NFS では，3 次元以上の篩処理が必要となる．本稿では，JL03-NFS において用いられる 2 次元 lattice sieve を拡張した 3 次元 lattice sieve を提案する．

A Construction of 3-dimensional Lattice Sieve for Number Field Sieve over $\mathbf{GF}(p^n)$

Kenichiro HAYASAKA* Kazumaro AOKI† Tetsutaro KOBAYASHI†
Tsuyoshi TAKAGI‡

*Graduate School of Mathematics Kyushu University
744, Motooka, Nishi-ku, Fukuoka, 819-0395, Japan

†NTT Secure Platform Laboratories
3-9-11 Midori-cho, Musashino-shi, Tokyo, 180-8585, Japan

‡Institute of Mathematics for Industry
744, Motooka, Nishi-ku, Fukuoka, 819-0395, Japan

Abstract The security of pairing-based cryptography is based on the hardness of the discrete logarithm problem over $\mathrm{GF}(p^n)$. Joux et al. proposed the number field sieve over $\mathrm{GF}(p^n)$ at CRYPTO 2006 (JLSV06-NFS). JLSV06-NFS includes a sieving step of more than 2 dimensions. In this paper, we present 3-dimensional lattice sieve as extension of 2-dimensional lattice sieve used by the number field sieve over $\mathrm{GF}(p)$.

1 はじめに

次世代公開鍵暗号であるペアリング暗号は，従来の公開鍵暗号では実現が困難な ID ベース暗号やインテリジェント暗号などの利便性の高い暗号システムが構築可能であるため実用化が期待されている．これまでペアリング暗号の高速実装が多く報告されているが，現在最も高速な実装が提案されているペアリングの一つとして BN 曲線 [1] を用いた Optimal Ate ペアリング [14] が知られている．その安全性は 12 次拡大体 $\mathrm{GF}(p^{12})$ 上の離散対数問題の困難性によっ

て保たれており，位数が 3072 ビットの拡大体 $\mathrm{GF}(p^{12})$ 上の離散対数問題の困難性は，128 ビットセキュリティレベルを満たすと考えられている [1]．ここで，素因数分解問題や素体 $\mathrm{GF}(p)$ 上の離散対数問題の困難性は，500 ビットを超える大規模な計算機実験によって評価されているが，ペアリング暗号で用いられる $n = 6$ や $n = 12$ の拡大体 $\mathrm{GF}(p^n)$ 上の離散対数問題の計算機実験による困難性評価はまだ少ない．このため，ペアリング暗号の安全性評価の精度を高める上で，拡大体 $\mathrm{GF}(p^n)$ 上の離散対数問題の困難性を計算機実験により評価することは不可欠である．

素体 $\mathrm{GF}(p)$ 上の離散対数問題に対する漸近的に現在最速な解法として数体篩法 [12, 4](JL03-NFS) が知られているが，CRYPTO 2006 においてJoux らは，JL03-NFS を拡大体 $\mathrm{GF}(p^n)$ 上に拡張した数体篩法 (JLSV06-NFS) を提案した [5]．JL03-NFS と，その拡張である JLSV06-NFS では，関係式の探索を行う関係探索ステップが大きく異なる．関係探索ステップで用いられる手法として line sieve と lattice sieve[11] が知られているが，一般に lattice sieve がより効率的であるとされている．このため，JL03-NFS や素因数分解での数体篩法 [8] では，2 次元篩領域上の lattice sieve が主流である．この lattice sieve では，ある基底により生成される格子点を篩領域内から探索する処理を含むが，2 次元の lattice sieve においては Franke-Kleinjung 法 [2, 7] を用いることで効率的かつ網羅的な格子点列挙を実現している．一方，$n = 6$ や $n = 12$ の拡大体 $\mathrm{GF}(p^n)$ 上の数体篩法では，次元を拡張して 3 次元以上の多次元篩領域における lattice sieve を考慮しなければならない．しかし，一般次元の line sieve[15, 3] や Franke-Kleinjung 法を考慮しない lattice sieve[16, 3] が知られているものの，2 次元 Franke-Kleinjung 法の単純な一般次元への適用は難しく，3 次元以上の Franke-Kleinjung 法が実現可能かはまだ知られていない．

本稿では，2 次元の Franke-Kleinjung 法によって生成される基底が満たすべき条件を 3 次元篩領域上へ拡張し，それらの条件を満たすような基底を生成するアルゴリズムを考察した．具体的には，2 次元 Franke-Kleinjung 法の第 1 成分を，第 2 及び第 3 成分の 2 次元として拡張することで 3 次元篩領域への適用を行った．そして，得られた提案 3 次元 Franke-Kleinjung 法を実装し，篩領域のサイズを変化させてそれぞれ約 8000 個の基底の生成を行った．その結果，約 60%の基底に対して拡張した条件を満たすような基底を生成でき，網羅的に格子点を列挙できた．さらに，条件を満たさない約 40%の基底に関しても，約 70%程度の格子点を列挙できた．

2 拡大体 $\mathrm{GF}(p^n)$ 上の数体篩法 (JLSV06-NFS) の概要 [5]

本節では，Joux らが提案した $\mathrm{GF}(p^n)$ 上の数体篩法の概要について記述する．

素数 p，自然数 n に対して，有限体 $\mathrm{GF}(p^n)$ の乗法群を $\mathrm{GF}(p^n)^\times$ とする．ここで，JLSV06-NFS では $\gamma, \delta \in \mathrm{GF}(p^n)^\times$ に対して，$\gamma^x = \delta$ を満たすような $x = \log_\gamma \delta = \{0, 1, \ldots, p^n - 2\}$ を求める．

JLSV06-NFS の多項式選択では，代数体を定義する 2 つの多項式を $f_1, f_2 \in \mathbb{Z}[X] \setminus \{0\}$ で，$f_1 \neq f_2$，$\deg f_1 = n$，f_1 は $\mathrm{GF}(p)$ 上既約，$f_1 \mid f_2 \bmod p$ となるものを選択する．このとき条件から，$\mathrm{GF}(p^n)$ 上で $f_1(v) = f_2(v) = 0$ を満たすような $v \in \mathrm{GF}(p^n)$ が存在する．f_1，f_2 それぞれの根 α_1, $\alpha_2 \in \mathbb{C}$ に対して代数体を $\mathbb{Q}(\alpha_1)$, $\mathbb{Q}(\alpha_2)$，その整数環を $\mathcal{O}_1, \mathcal{O}_2$ とすると，準同形写像

$$\begin{aligned} \phi_1 &: \mathbb{Z}[\alpha_1] \to \mathrm{GF}(p^n),\ \alpha_1 \mapsto v \\ \phi_2 &: \mathbb{Z}[\alpha_2] \to \mathrm{GF}(p^n),\ \alpha_2 \mapsto v \end{aligned} \tag{1}$$

が存在する．

JLSV06-NFS の関係探索では，整数 $t \geq 1$ に対してある条件を満たす $t+1$ 次元ベクトル $\mathbf{a} = (a_0, a_1, \ldots, a_t)^{\mathrm{T}}$ を収集する．

smoothness bound $B_1, B_2 \in \mathbb{R}_{>0}$ と，2 つの多項式 f_1, f_2 に対して，factor base $\mathcal{B}_1, \mathcal{B}_2$ を

$$\begin{aligned} \mathcal{B}_i = \{(q, g) \mid\ & q\text{: prime}, q \leq B_i, \\ & g\text{: monic polynomial}, \deg g \leq t, \\ & \text{irreducible factor of } f_i \bmod q\} \end{aligned}$$

とする．また，ある $t+1$ 次元列ベクトルの有限集合 $\mathcal{H}_a \subset \mathbb{Z}^{t+1}$ を篩領域とよぶ．さらに，$i = 1, 2$ の両方について，$\mathbf{a}$ の f_i に対するノルムを

$$N(\mathbf{a}, f_i) = \left| \mathrm{Res}(\sum_{j=0}^{t} a_j X^j, f_i(X)) \right|$$

とする．ただし，$\mathrm{Res}(f(X), g(X))$ は $f(X), g(X)$ の終結式を表す．このとき，smoothness bound B_1, B_2 に対して，$N(\mathbf{a}, f_1)$，$N(\mathbf{a}, f_2)$ がそれぞれ B_1-smooth，B_2-smooth であるような $\mathbf{a} \in \mathcal{H}_a$(以下 hit tuple とよぶ) を次節の lattice sieve を用いて収集する．ここで B-smooth とは，最大の素因数が B 以下の整数である．収集した hit tuple の集合を S としたとき，$\sharp S \geq \sharp\mathcal{B}_1 + \sharp\mathcal{B}_2 + 2n$ を満たさない場合は $\mathcal{H}_a, B_1, B_2$ を選びなおす．hit tuple $\mathbf{a} = (a_0, a_1, \ldots, a_t)^{\mathrm{T}} \in S$ は $i = 1, 2$ の両方について，$(\sum_{j=0}^{t} a_j \alpha_i^j)\mathcal{O}_i = \prod_{\mathfrak{q} \in \mathcal{B}_i} \mathfrak{q}^{\varepsilon_{\mathbf{a},\mathfrak{q}}}$ と表すことができる．準同型写像 (1) を用いて，$\phi_1(\sum_{j=0}^{t} a_j \alpha_1^j) = \phi_2(\sum_{j=0}^{t} a_j \alpha_2^j)$ であることから，関係式 (relation)

$$\sum_{\mathfrak{q} \in \mathcal{B}_1} \varepsilon_{\mathbf{a},\mathfrak{q}} \log \phi_1(\mathfrak{q}) + \sum_{j=1}^{r_1} \lambda_{\mathbf{a},1,j} \log \Lambda_{1,j} \equiv \sum_{\mathfrak{q} \in \mathcal{B}_2} \varepsilon_{\mathbf{a},\mathfrak{q}} \log \phi_2(\mathfrak{q}) + \sum_{j=1}^{r_2} \lambda_{\mathbf{a},2,j} \log \Lambda_{2,j} \pmod{p^n - 1}$$

を得る．ここで，$\log \phi_i(\mathfrak{q})$ 及び $\log \Lambda_{i,j}$ は virtual logarithms とよばれる未知数 [4, 13]，$\lambda_{\mathbf{a},i,j}$ は単数による差を補正する Schirokauer 指標 [12]，r_i は $\mathcal{O}_i$ の torsion-free rank である．得られた relation から連立一次合同式を構成し解くことで，$\log \phi_i(\mathfrak{q}), \log \Lambda_{i,j} \pmod{p^n - 1}$ を得る．

2.1 多次元篩領域における lattice sieve[16, 3]

本節では，$t+1$ 次元篩領域における lattice sieve を用いた hit pair の探索法について記述する．

lattice sieve では，ある $\mathfrak{r} = (r, g) \in \mathcal{B}_i$ で割れる $\mathbf{a} \in \mathbb{Z}^{t+1}$ が格子になっていることを利用して，ある $\mathfrak{q} = (q, g) \in \mathcal{B}_i$ (special-$\mathfrak{q}$) に対して，既に $q \mid N(\mathbf{a}, f_i)$ であるような $\mathbf{a}$ の格子上において，さらに篩処理を行う方法である．lattice sieve の利点として，既に q で割れているため $N(\mathbf{a}, f_i)$ が smooth になりやすいことや，篩処理の対象を $\mathfrak{q}$ で割れている $\mathbf{a}$ と限定することで篩領域のサイズを縮小できることなどが挙げられる．

special-$\mathfrak{q}$ で割れる $\mathbf{a}$ は $t+1$ 個の基底により生成されるが，それらの基底を列ベクトルとしてもつ $t+1$ 次正方行列を $M_{\mathfrak{q}}$ とし，$t+1$ 個の基底の $\mathbb{Z}$ 係数の空間を $\mathbf{c}$ 空間とよぶ．また，lattice sieve では $\mathbf{a}$ 空間の篩領域 $\mathcal{H}_a$ は用いず，篩領域の閾値 I (整数かつ偶数値)，$J \in \mathbb{Z}$ に対する $\mathbf{c}$ 空間の篩領域

$$\mathcal{H}_c = \{(c_0, c_1, \ldots, c_t)^{\mathrm{T}} \in \mathbb{Z}^{t+1} \mid -I/2 \leq c_i < I/2 \ (0 \leq i \leq t-1), \ 0 \leq c_t < J\}$$

を用いる．lattice sieve では，ある $\mathfrak{r} = (r, g) \in \mathcal{B}_i$ で割れるような $\mathbf{c}$ 空間上の $\mathbf{c} \in \mathbb{Z}^{t+1}$ もまた格子となっていることを利用し，$\mathcal{H}_c$ 上で篩処理を行う．$\mathbf{c}$ 空間上において，$\mathfrak{r}$ で割れる $\mathbf{c}$ を生成する $t+1$ 個の基底を列ベクトルとしてもつ $t+1$ 次正方行列 $M_{\mathfrak{q},\mathfrak{r}}$ は，$M_{\mathfrak{q}}$ 及び $M_{\mathfrak{r}}$ の kernel の基底として得られる．

$\mathcal{H}_c$ と $M_{\mathfrak{q},\mathfrak{r}}$ に対して，$\mathcal{H}_c$ に含まれ，かつ $M_{\mathfrak{q},\mathfrak{r}}$ の格子点であるような $\mathbf{c}$ を効率的に列挙する方法は，$t = 1$ すなわち 2 次元の lattice sieve に関しては Franke-Kleinjung 法 [2] が知られている．しかし，$t > 1$ すなわち 3 次元以上の lattice sieve において Franke-Kleinjung 法が適用可能かどうかは知られていない．

3 2次元 Franke-Kleinjung 法 [2]

本節では，$t = 1$ すなわち 2 次元篩領域上での格子点の列挙法である Franke-Kleinjung 法について記述する．

$\mathbf{c}$ 空間上の格子点の基底を列ベクトルとする $M_{\mathfrak{q},\mathfrak{r}}$ と，篩領域 $\mathcal{H}_c$ とその閾値 I に対して，$\mathbf{c} \in \mathcal{H}_c$ かつ $M_{\mathfrak{q},\mathfrak{r}}$ の格子点であるような $\mathbf{c}$ を効率的に列挙する方法として，2 次元篩領域の場合は Franke-Kleinjung 法が知られている．

3.1 基底 $M_{\mathfrak{q},\mathfrak{r}}^{\mathbf{FK}}$ の生成

$M_{\mathfrak{q},\mathfrak{r}}$ を Hermite normal form (HNF) に変換した行列を $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{HNF}}$ とすると，$M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{HNF}}$ は以下の 3 通りのいずれかの形になる．

$$1: \begin{pmatrix} r & z \\ 0 & 1 \end{pmatrix}, \ 2: \begin{pmatrix} 1 & 0 \\ 0 & r \end{pmatrix}, \ 3: \begin{pmatrix} r & 0 \\ 0 & 1 \end{pmatrix}$$

ただし，$z \in \mathbb{Z}_{>0}, z < r$ である．形状 2，3 及び $r \leq I$ である場合の形状 1 では，$\mathbf{c}$ 空間上で line sieve を行うことで $L_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}_c$ を得る．よって以降では，$r > I$ であるときの形状 1 の場合を扱う．

Algorithm 1 2 次元 Franke-Kleinjung 法における $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ の生成方法

Input: $\mathcal{H}_c$ の閾値 I，基底 $M^{\mathrm{HNF}}_{\mathfrak{q},\mathfrak{r}} = (\mathbf{u},\mathbf{v}) = ((r,0)^{\mathrm{T}},(z,1)^{\mathrm{T}}), r > I, 0 < z < r$
Output: 変換された行列 $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$

1: $\mathbf{u} \leftarrow -\mathbf{u}$
2: **while** $|v_0| \geq I$ **do**
3: $\quad \mathbf{u} \leftarrow \mathbf{u} + a\mathbf{v},\ a = \lfloor -u_0/v_0 \rfloor$
4: $\quad$ SWAP$(\mathbf{u},\mathbf{v})$
5: **end while**
6: $\mathbf{u} \leftarrow \mathbf{u} + a\mathbf{v}$, a は $|u_0| < I$ を満たす最小の正の整数
7: **if** $u_0 < 0$ **then** SWAP$(\mathbf{u},\mathbf{v})$
8: **return** $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}} = (\mathbf{u},\mathbf{v})$

Algorithm 2 NEXTFK2$(I,\mathbf{p},M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}})$

Input: $\mathcal{H}_c$ の閾値 I，格子点 $\mathbf{p} = (p_0,p_1)^{\mathrm{T}} \in L_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}_c$，基底 $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}} = (\mathbf{u},\mathbf{v}) = ((u_0,u_1)^{\mathrm{T}},(v_0,v_1)^{\mathrm{T}})$
Output: 格子点 $\mathbf{p}' = (p'_0,p'_1)$ s.t. $\mathbf{p}' \in L_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}_c$ かつ $p'_1 > p_1$ かつ $p'_1 - p_1$ が最小

1: **if** $-I/2 \leq p_0 + u_0$ **then return** $\mathbf{p} + \mathbf{u}$
2: **if** $p_0 + v_0 < I/2$ **then return** $\mathbf{p} + \mathbf{v}$
3: **return** $\mathbf{p} + \mathbf{u} + \mathbf{v}$

Franke-Kleinjung 法では，$\mathcal{H}_c$ を考慮した $M_{\mathfrak{q},\mathfrak{r}}$ の基底変換を行い，成分が $\mathbb{Z}$ である 2 次正方行列

$$M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}} = (\mathbf{u},\mathbf{v}) = \begin{pmatrix} u_0 & v_0 \\ u_1 & v_1 \end{pmatrix},$$

を生成する．ただし，次の条件を満たすとする．

- 長さ条件: $|v_0| < I$ かつ $|u_0| < I$
- 距離条件: $|u_0 - v_0| \geq I$
- 単調条件: $u_1 > 0$ かつ $v_1 > 0$
- 条件 a: $\mathrm{sign}(u_0) \neq \mathrm{sign}(v_0)$
- 条件 b: $u_0 > 0$

ここで，条件 a は長さ条件及び距離条件から成り立ち，条件 b は $\mathbf{u},\mathbf{v}$ の交換を行うことで満たすことができる．上記のような $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ の生成方法を Algorithm 1 に示す．

3.2 格子点列挙方法

本節では，$L_{\mathfrak{q},\mathfrak{r}}$ を $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ で生成される格子点全体とする．はじめに $\mathbf{p} \leftarrow (0,0)$ として初期化を行い，$M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ 及び篩領域の閾値 I,J に対して，Algorithm 2 の NEXTFK2 を用いて $\mathbf{p} \notin \mathcal{H}_c$，すなわち $\mathbf{p}$ の第 2 成分 p_1 が J 以上となるまで $\mathbf{p} \leftarrow \mathrm{NEXTFK2}(I,\mathbf{p},M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}})$ として繰り返すことで格子点を列挙する．

Franke-Kleinjung 法の効率的である特徴として，以下が考えられる．

- Algorithm 2 の分岐が 3 通り，
- $L_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}_c$ における格子点列挙が網羅的，
- Algorithm 2 の p_1 が単調増加．

次節では，3 次元の $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ がこれらの特徴を保持するよう，Franke-Kleinjung 法の拡張を行った．

4 提案 Franke-Kleinjung 法

本節では，$t = 2$ すなわち 3 次元篩領域上に拡張した Franke-Kleinjung 法の基底生成及び格子点列挙について記述する．

4.1 基底 $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ の生成

Algorithm 2 を用いた格子点列挙において，列挙される格子点 $\mathbf{p}$ の第 2 成分 p_1 は単調増加であり，基底 $\mathbf{u},\mathbf{v}$ の第 2 成分は共に正であった．このことから，u_1 及び v_1 は可能な限り 0 に近い値を保ちつつ基底変換を行うべきである．

上記の理由と 3.1 節に習い，3 次元の場合でも $M^{\mathrm{HNF}}_{\mathfrak{q},\mathfrak{r}}$ から $M^{\mathrm{FK}}_{\mathfrak{q},\mathfrak{r}}$ の生成を開始する．$M_{\mathfrak{q},\mathfrak{r}}$ を HNF に変換した行列を $M^{\mathrm{HNF}}_{\mathfrak{q},\mathfrak{r}}$ とすると，$M^{\mathrm{HNF}}_{\mathfrak{q},\mathfrak{r}}$ は以下の 14 通りのいずれかと一致する．

$$\mathbf{1}: \begin{pmatrix} r & z_1 & z_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},\ 2: \begin{pmatrix} r & z_1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$3: \begin{pmatrix} r & 0 & z_2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},\ 4: \begin{pmatrix} r & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$5: \begin{pmatrix} 1 & 0 & 0 \\ 0 & r & z_2 \\ 0 & 0 & 1 \end{pmatrix},\ 6: \begin{pmatrix} 1 & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$7: \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{pmatrix},$$

$$\mathbf{8}: \begin{pmatrix} r & 0 & z_1 \\ 0 & r & z_2 \\ 0 & 0 & 1 \end{pmatrix},\ 9: \begin{pmatrix} r & 0 & 0 \\ 0 & r & z_2 \\ 0 & 0 & 1 \end{pmatrix},$$

$$10: \begin{pmatrix} r & 0 & z_1 \\ 0 & r & 0 \\ 0 & 0 & 1 \end{pmatrix},\ 11: \begin{pmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$12: \begin{pmatrix} r & z_1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{pmatrix},\ 13: \begin{pmatrix} r & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{pmatrix},$$

$$14: \begin{pmatrix} 1 & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{pmatrix}.$$

Algorithm 3 3 次元 Franke-Kleinjung 法の $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ の生成

Input: $\mathcal{H}_c$ の閾値 I，基底 $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{HNF}} = (\mathbf{u},\mathbf{v},\mathbf{w}) = ((r,0,0)^{\mathrm{T}},(z_1,1,0)^{\mathrm{T}},(z_2,0,1)^{\mathrm{T}})$, $r > I, 0 < z_1 < r, 0 < z_2 < r$,
Output: 基底 $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$

$v_0 < 0, w_1 < 0, w_0 = 0$ となるように基底変換を行う
$I, \mathbf{u}, \mathbf{v}$ に対して Algorithm 1 を行う
if $|u_1| \geq |w_1|$ **then do** $\mathbf{w}$ により $\mathbf{u}$ を reduce
if $|v_1| \geq |w_1|$ **then do** $\mathbf{w}$ により $\mathbf{v}$ を reduce
while $|w_1| \geq I$ **do**
  if $\mathrm{sign}(u_1) = \mathrm{sign}(v_1)$ **then do**
    REDUCE1($\mathbf{w},\mathbf{u},\mathbf{v}$)
  else do
    REDUCE2($\mathbf{w},\mathbf{u},\mathbf{v}$)
  RADIATE($\mathbf{u},\mathbf{v},\mathbf{w}$)
  if $|u_1| > |v_1|$ **then do**
    SWAP($\mathbf{w},\mathbf{u}$)
  else do
    SWAP($\mathbf{w},\mathbf{v}$)
end while
ADJUST($\mathbf{u},\mathbf{v},\mathbf{w}$)
return $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}} = (\mathbf{u},\mathbf{v},\mathbf{w})$

Algorithm 4 NEXTFK3($\mathcal{H}_c, \mathbf{p}, M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$)

Input: $\mathbf{p} \in L_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}_c$, $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}} = (\mathbf{u},\mathbf{v},\mathbf{w})$
Output: $\mathbf{p}' \in L_{\mathfrak{q},\mathfrak{r}} \cap \mathcal{H}_c$

if $\mathbf{p}+\mathbf{u} \in \mathcal{H}_c$ **then return** $\mathbf{p}+\mathbf{u}$
if $\mathbf{p}+\mathbf{v} \in \mathcal{H}_c$ **then return** $\mathbf{p}+\mathbf{v}$
if $\mathbf{p}+\mathbf{w} \in \mathcal{H}_c$ **then return** $\mathbf{p}+\mathbf{w}$
if $\mathbf{p}+i\mathbf{u}+j\mathbf{v} \in \mathcal{H}_c$ となる $i,j \in \mathbb{Z}_{>0}$ が存在する **then return** $\mathbf{p}+i\mathbf{u}+j\mathbf{v}$
$\mathbf{p} \leftarrow \mathbf{p}+\mathbf{w}$
if $\mathbf{p}+i\mathbf{u}+j\mathbf{v} \in \mathcal{H}_c$ となる $i,j \in \mathbb{Z}_{\geq 0}$ が存在する **then return** $\mathbf{p}+i\mathbf{u}+j\mathbf{v}$

ただし，$z_1, z_2 \in \mathbb{Z}_{>0}, z_1, z_2 < r$ である．形状 4, 6, 7, 11, 13, 14 の場合や $r \leq I$ の場合は line sieve を行うことで格子点を列挙する．$r > I$ で形状 2, 3, 5, 9, 10, 12 の場合は，要素に色がついた部分に対して 2 次元 Franke-Kleinjung 法 (Algorithm 1) を用いて基底を生成する．よって以降では，$r > I$ であるときの形状 1, 8 の場合について扱う．

3 節の長さ，距離及び単調条件を 3 次元の lattice sieve に拡張し，2 次元 Franke-Kleinjung 法の 3 つの特徴を満たすような新たな条件を次のように定めた．成分が $\mathbb{Z}$ である 3 次正方行列

$$M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}} = (\mathbf{u},\mathbf{v},\mathbf{w}) = \begin{pmatrix} u_0 & v_0 & w_0 \\ u_1 & v_1 & w_1 \\ u_2 & v_2 & w_2 \end{pmatrix}$$

は次を満たすとする．

- 長さ条件: $\forall \mathbf{x} = (x_0,x_1,x_2)^{\mathrm{T}} \in \{\mathbf{u},\mathbf{v},\mathbf{w}\}$, $|x_0| < I$ かつ $|x_1| < I$
- 距離条件 1: 任意の $\mathbf{x},\mathbf{y} \in \{\mathbf{u},\mathbf{v},\mathbf{w}\}, \mathbf{x} \neq \mathbf{y}$ に対して $|x_0 - y_0| \geq I$ または $|x_1 - y_1| \geq I$
- 距離条件 2: $\mathbf{x} = i\mathbf{u} + j\mathbf{v} - \mathbf{w}$ $(i,j \in \mathbb{Z}_{>0})$ としたとき，$|x_0| < I$ かつ $|x_1| < I$ を満たすような i,j が存在しない
- 単調条件: $u_2 > 0$ かつ $v_2 > 0$ かつ $w_2 > 0$

上記のような $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ の生成方法を Algorithm 3 に示す．ただし，step 7, 8, 10, 16 は Sub-algorithm として付録 A に示す．Algorithm 3 の step 1-2 によって u_0, v_0, w_0 は長さ条件を満たす．そして，step 7, 9 により c_0 成分の長さ条件を保ちながら，u_1, v_1 を用いて w_1 のサイズを減少させる．また，step 10 では $\mathbf{u},\mathbf{v},\mathbf{w}$ の向きを調節し，step 16 によって距離条件 1 を満たすように調整を行う．

付録 B に基底生成例を示す．付録 B に示した基底は，長さ条件，距離条件 1 及び 2，単調条件を満たすことがわかる．

4.2 格子点列挙方法

本節では，$L_{\mathfrak{q},\mathfrak{r}}$ を $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ で生成される格子点全体とする．はじめに $\mathbf{p} = (0,0,0)$ として初期化を行い，$M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ 及び篩領域の閾値 I, J に対して，Algorithm 4 の NEXTFK3 を用いて $\mathbf{p} \notin \mathcal{H}_c$，すなわち $\mathbf{p}$ の第 3 成分 p_2 が J 以下となるまで $\mathbf{p} \leftarrow \mathrm{NEXTFK3}(I,\mathbf{p},M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}})$ として繰り返すことで格子点を列挙する．

5 実装実験

本節では，4 節で提案した基底 $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ の生成に関する実験と，生成された $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ を用いた格子点列挙実験について述べる．計算機は，CPU を Intel Core i7-3770 3.40GHz，RAM を 8GB 搭載した PC1 台を用いた．OS は Linux(64 ビット) を，言語は C++によって行い，コンパイラは gcc-4.7.2 を用いた．また，多倍長演算には gmp-5.0.5 を用いた．

実験で用いたパラメータは，拡大体は $p = 38486027$，$n = 12$，p^n: 303 ビット，代数体を定義する多項式は $f_1(X) = X^{12} + X^2 - 1$，$f_2(X) = f_1(X) + p$，special-$\mathfrak{q}$ として $99871 \leq q \leq 99989$ であるような $\mathfrak{q} = (q,h) \in \mathcal{B}_2$ を 10 個とった．

表 1: (長さ，距離 1，距離 2，単調) 条件を満たす $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ の個数と割合

I	$M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ の総数	(1,1,1,1)	(1,1,1,0)	(1,1,0,1)	(1,1,0,0)	平均実行時間 (msec)
4	84265	60372(71%)	15056(17%)	10871(12%)	2072(2%)	14.8
16	84245	55195(65%)	15899(18%)	16040(19%)	2925(3%)	12.1
32	84205	56188(66%)	14029(16%)	16845(20%)	2890(3%)	11.2
64	84145	58464(69%)	10341(12%)	17477(20%)	2169(2%)	10.4
128	84005	62395(74%)	3724(4%)	18765(22%)	901(1%)	11.0
256	83865	61717(73%)	322(0%)	21918(26%)	110(0%)	12.1
512	83445	55746(66%)	2(0%)	27681(33%)	2(0%)	16.4
1024	82645	54204(65%)	0(0%)	28425(34%)	0(0%)	27.0
2048	81185	52446(64%)	0(0%)	28725(35%)	0(0%)	47.2

表 2: 全ての条件を満たす場合とその他の場合での格子点数と見積もられた格子点数とその割合

I	全ての条件を満たす場合	満たされない条件が存在する場合
4	232(64%)	26(24%)
16	15101(93%)	6612(68%)
32	112776(98%)	44345(73%)
64	804707(99%)	363001(83%)
128	5458255(99%)	2622610(90%)
256	38002237(99%)	21571710(94%)
512	244756676(99%)	155292427(96%)
1024	1562391487(99%)	1086778163(98%)
2048	9814496267(99%)	6964332990(99%)

5.1 基底生成実験

10 個の special-$\mathfrak{q}$ と篩領域の閾値 I に対して，$I < r \leq 85386$ であるような $\mathfrak{r} = (r, h) \in \mathcal{B}_2, \deg(h) = 1$，合計約 80000 個の $\mathfrak{r}$ に対して Algorithm 3 を用いて $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ の生成を行った．そして，生成した基底が 4 節の条件，長さ条件，距離条件 1，距離条件 2，単調条件を満たすかを確かめた．

この結果を表 1 に示す．列の項目は，4 節の条件 (長さ条件，距離条件 1，距離条件 2，単調条件) が満たされているかを，満たしていれば 1，そうでなければ 0 で表している．そして，special-$\mathfrak{q}$ 1 個あたりの平均の基底生成時間を最後の列に示している．全ての条件を満たすような基底は，どの篩領域の場合でも 60%から 70%ほどの確率で得られた．また，篩領域が大きくなるにつれて，単調条件を満たさない基底が減り，距離条件 2 を満たさない基底が増加した．

5.2 格子点列挙実験

10 個の special-$\mathfrak{q}$ と篩領域の閾値 I に対して生成した $M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}}$ を用いて Algorithm 4 を用いて格子点の列挙を行った．c_2 の閾値は $J = I/2$ とした．$\mathcal{H}_c$ 内にあると期待される格子点の個数の見積もり値を $I^2 J/r$ とし，列挙実験で得られた格子点の個数とその見積もりの個数との比較を行った．ただし，単調条件を満たさない基底は，実際には lattice sieve において使用できないため，単調条件を満たさない基底によって得られた格子点は 0 個としている．

この結果を表 2 に示す．全ての条件を満たす場合はほぼ 100%の格子点を列挙しているが，満たされない条件が存在する場合でも，篩領域が大きいほど得られた格子点の数が大きく，$I \geq 128$ の結果では 90%以上の格子点を列挙できた．$I \geq 128$ では単調条件を満たさない基底の割合が非常に少ないことが，90%以上の格子点を得られた理由の一つであると考えられる．

6 まとめ

本稿では，3 次元篩領域上の lattice sieve において用いる 3 次元 Franke-Kleinjung 法を提案した．具体的には，3 次元篩領域上の格子点を効率的かつ網羅的に列挙できる基底の条件を既知の 2 次元 Franke-Kleinjung 法を基に拡張し，それらの条件を満たすような基底を生成するアルゴリズムを考案した．さらに，この 3 次元へ拡張した提案 Franke-Kleinjung 法を実装し，約 80000 個の基底を生成した結果，約 60%の基底に対して効率的かつ網羅的な条件を満たすような基底を生成でき，実際に格子点数の見積もり値とほぼ同数の格子点を列挙できた．また条件を満たさない基底に関しても，少ない場合では約 70%程度の格子点が列挙できたが，篩領域サイズの増加に対して得られる格子点数も増加した．smoothness bound に対して篩領域が十分大きいときは，ほとんどの格子点を列挙できると考えられるため，そのようなパラメータを用いる場合は 3 次元 lattice sieve を高速にできる可能性がある．

今後は，本稿で提案を行った 3 次元 Franke-Kleinjung 法を用いて 3 次元 lattice sieve を行った場合，網羅的でなかった基底を用いることで得られなかった relation 数について実験し検証する．また，任意の基底に対して網羅的に列挙が可能な基底の条件を考察し，基底生成アルゴリズムの改良を行う．さらに，4 次元以上篩領域において Franke-Kleinjung 法の適用が可能か考察する．

参考文献

[1] P.S.L.M. Barreto and M. Naehrig, “Pairing-friendly elliptic curves of prime order,” SAC 2005, Lecture Notes in Comput. Sci., vol.3897, pp.319-331, Springer, 2006.

[2] J. Franke and T. Kleinjung, “Continued fractions and lattice sieve,” Workshop record of SHARCS, 2005, http://www.ruhr-uni-bochum.de/itsc/tanja/SHARCS/talks/FrankeKleinjung.pdf.

[3] 早坂健一郎, 青木和麻呂, 小林鉄太郎, 高木剛, “GF(p^{12}) 上の離散対数問題に対する数体篩法の計算機実験,” 2013 年暗号と情報セキュリティシンポジウム, SCIS2013, 4A1-3, 2013.

[4] A. Joux and R. Lercier, “Improvements to the general number field sieve for discrete logarithms in prime fields. A comparison with the Gaussian integer method,” Math. Comp., vol.72, pp.953-967, 2003.

[5] A. Joux, R. Lercier, N.P. Smart and F. Vercauteren, “The number field sieve in the medium prime case,” CRYPTO ’06, Lecture Notes in Comput. Sci., vol.4117, pp.326-344, Springer-Verlag, 2006.

[6] T. Kleinjung et al., “Discrete logarithms in GF(p) - 160 digits,” email to the NMBRTHRY mailing list, 2007. http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0702&L=nmbrthry&T=0&P=194.

[7] T. Kleinjung, K. Aoki, J. Franke, A.K. Lenstra, E. Thomé, J.W. Bos, P. Gaudry, A. Kruppa, P.L. Montgomery, D.A. Osvik, H.J.J. te Riele, A. Timofeev and P. Zimmermann, “Factorization of a 768-bit RSA modulus,” CRYPTO ’10, Lecture Notes in Comput. Sci., vol.6223, pp.333-350, Springer-Verlag, 2010.

[8] A.K. Lenstra and H.W. Lenstra, *The Development of the Number Field Sieve,* Lecture Notes in Math., vol.1554, Springer-Verlag, 1993.

[9] A.K. Lenstra, H.W. Lenstra and L. Lovász, “Factoring polynomials with rational coefficients,” Math. Ann., vol.261, pp.515-534, 1982.

[10] PARI/GP, version 2.5.3, Bordeaux, 2012. http://pari.math.u-bordeaux.fr/.

[11] J.M. Pollard, “The lattice sieve,” pp.43-49, in [8].

[12] O. Schirokauer, “Discrete logarithms and local units,” Philos. Trans. Roy. Soc. London Ser. A, vol.345, pp.409-424, 1993.

[13] O. Schirokauer, “Virtual logarithms,” J. Algorithms, vol.57, pp.140-147, 2005.

[14] F. Vercauteren, “Optimal pairings,” IEEE Transactions on Information Theory, vol.56, pp.455-461, 2010.

[15] P. Zajac, “Discrete logarithm problem in degree six finite fields,” PhD thesis, Slovak University of Technology, 2008. `http://www.kaivt.elf.stuba.sk/kaivt/Vyskum/XTRDL`.

[16] P. Zajac, “On the use of the lattice sieve in the 3D NFS,” Tatra Mt. Math. Publ. vol.45, pp.161-172, 2010.

付録 B

$p = 38486027, n = 12$, p^n: 303 ビット. $f_1(X) = X^{12} + X^2 - 1$, $f_2(X) = f_1 + p$. $\mathfrak{q} = (q, g) = (99989, X + 8368), \mathfrak{r} = (r, h) = (89107, X + 54851)$ $I = 64$,

$$M_{\mathfrak{q}} = \begin{pmatrix} 99989 & 8368 & 0 \\ 0 & 1 & 8368 \\ 0 & 0 & 1 \end{pmatrix},$$

$$M_{\mathfrak{q}}^{\mathrm{LLL}} = \begin{pmatrix} 44 & -41 & -43 \\ 77 & -132 & 47 \\ -5 & -13 & 3 \end{pmatrix},$$

$$M_{\mathfrak{r}} = \begin{pmatrix} 89107 & 54851 & 0 \\ 0 & 1 & 54851 \\ 0 & 0 & 1 \end{pmatrix},$$

$$M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{HNF}} = \begin{pmatrix} 89107 & 27083 & -50795 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$M_{\mathfrak{q},\mathfrak{r}}^{\mathrm{FK}} = \begin{pmatrix} 23 & -57 & 35 \\ 23 & -10 & -48 \\ 7 & 28 & 13 \end{pmatrix}$$

付録 A

Algorithm 5 REDUCE1($\mathbf{w}, \mathbf{u}, \mathbf{v}$)

Input: $\mathcal{H}_c$ の閾値 I, ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$ s.t. $\forall \mathbf{x} \in \{\mathbf{u}, \mathbf{v}, \mathbf{w}\}, |x_0| < I$ かつ $(w_1 > u_1$ かつ $w_1 > v_1)$ かつ $\mathrm{sign}(u_1) = \mathrm{sign}(v_1)$
Output: ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$ s.t. $\forall \mathbf{x} \in \{\mathbf{u}, \mathbf{v}, \mathbf{w}\}, |x_0| < I$ かつ $(w_1 < u_1$ あるいは $w_1 < v_1)$

```
(x, y) ← (u, v)
if x_2 > y_2 then do SWAP(x, y)
else if (x_2 = y_2) ∧ (x_1 > y_1) then do SWAP(x, y)
while true do
    while |w_0 + x_0| < I do
        if (|w_1| < |u_1|) ∨ (|w_1| < I) then do
            return u, v, w
        w ← w + x
    end while
    if (|w_1| < |v_1|) ∨ (|w_1| < I) then do
        return u, v, w
    w ← w + y
end while
```

Algorithm 6 REDUCE2($\mathbf{w}, \mathbf{u}, \mathbf{v}$)

Input: $\mathcal{H}_c$ の閾値 I, ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$ s.t. $\forall \mathbf{x} \in \{\mathbf{u}, \mathbf{v}, \mathbf{w}\}, |x_0| < I$ かつ $(w_1 > u_1$ かつ $w_1 > v_1)$ かつ $\mathrm{sign}(u_1) \neq \mathrm{sign}(v_1)$
Output: ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$ s.t. $\forall \mathbf{x} \in \{\mathbf{u}, \mathbf{v}, \mathbf{w}\}, |x_0| < I$ かつ $(w_1 < u_1$ あるいは $w_1 < v_1)$

```
if sign(u_1) ≠ sign(w_1) then x ← u, y ← v else x ← v, y ← u
while |w_1| < I do
    w ← w + x
    while |w_0| ≥ I do
        w ← w + y
    end while
    if (|w_1| < |u_1|) ∨ (|w_1| < |v_1|) then break
end while
return u, v, w
```

Algorithm 7 IS_OPPOSITE($\mathbf{u}, \mathbf{v}, \mathbf{w}$)

Input: ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$
Output: $\mathbf{u}$ に対して $\mathbf{v}, \mathbf{w}$ とのなす角が 180 度未満かどうか

```
if u_0 = 0 then
    if sign(v_0) ≠ sign(w_0) then return true else return false
g = u_1/u_0
y = g v_0 − v_1, z = g w_0 − w_1
if sign(y) ≠ sign(z) then return true else return false
```

Algorithm 8 RADIATE($\mathbf{u}, \mathbf{v}, \mathbf{w}$)

Input: ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$
Output: ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$ s.t. $\mathbf{u}, \mathbf{v}, \mathbf{w}$ のなす角が 180 度

```
if IS_OPPOSITE(u, v, w) is true then
    if IS_OPPOSITE(v, w, u) is false then u ← −u
else
    if IS_OPPOSITE(v, w, u) is true then v ← −v else w ← −w
end if
return u, v, w
```

Algorithm 9 ADJUST($\mathbf{u}, \mathbf{v}, \mathbf{w}$)

Input: ベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$
Output: 距離条件 1 を満たすベクトル $\mathbf{u}, \mathbf{v}, \mathbf{w}$

```
for 全ての組み合わせ (x, y) do
    z = x − y
    if (|z_0| < I) ∧ (|z_1| < I) then
        if |x_2| ≥ |y_2| then x = z else y = z
        RADIATE(u, v, w)
        go to step 1
end for
return u, v, w
```