Magma で広がる数学の世界

Exploring Mathematics with Magma

GCOE レクチャーノート　報告集

木田 雅成
Masanari Kida

原田 昌晃
Masaaki Harada

横山 俊一
Shun'ichi Yokoyama

[編]

研究集会「Magma で広がる数学の世界」　GCOE レクチャーノート　報告集

Exploring Mathematics with Magma

はじめに

開催当日のプログラム

報告集原稿　※ プログラム順に掲載・講演タイトルはプログラムを参照

＜講演者リスト＞

木田雅成（入門講義担当）

原田昌晃（符号理論担当）

長尾孝一（暗号理論担当）

宗政昭弘（組合せ論担当）

藤田亮（可換環論担当）

横山俊一（保型形式論担当）

松野一夫（楕円曲線論担当）

千吉良直紀（有限群論担当）

大浦学（不変式論担当）

＊＊＊

本報告集のほぼ全ての原稿と, 一部の講演者の当日のスライド・デモプログラムは, 研究集会のホームページ

http://www2.math.kyushu-u.ac.jp/~s-yokoyama/magma/

から入手可能です. なお, 修正・変更等は随時 HP 上のデータに反映させる予定です.

はじめに

本報告集は 2010 年 10 月 9 日 (土) と 10 月 10(日) の 2 日間にわたって九州大学伊都キャンパスで開催された研究集会「Magma で広がる数学の世界」の記録である．この研究集会は九州大学大学院数理学研究院のグローバル COE プログラム「マス・フォア・インダストリ教育研究拠点」との共催で開かれた．連休中にも関わらず全国から 50 人以上の大学院生，研究者が集まり，充実した講演と，それをめぐる活発な討議がなされ，集会は成功裡に終了した．

Magma はオーストラリアのシドニー大学で開発された代数学の計算を行うためのソフトウェアであり，ヨーロッパを中心に信頼性の高い標準的なシステムとして多くの分野の研究者に使われている．さらに海外では Magma に関する研究集会も開かれるなど，Magma をめぐった活発な活動が行われている．これまで，そのような機会に恵まれなかった日本でも，Magma に関する情報交換・意見交換の場を提供としようというのが，この集会を中心となって企画した木田・原田のアイディアであった．Magma を中心に据えた初めての集会ということもあって，初心者へのチュートリアル講演から始め，様々な分野における Magma の様々な使われ方を知ることを目的とし，整数論，組合せ論，暗号理論を中心に 9 つの講演がそれぞれの分野の専門家によって行われた．

他分野の聴衆を意識した講演をすることに心を砕いてくださり，また時間的な制約の厳しい中，本報告集に原稿を寄せてくださった講演者の方々にここに感謝したい．また，多忙にもかかわらずグローバル COE プログラムとの共催をすすめてくださった九州大学大学院数理学研究院長の金子昌信さんに深く感謝する．

この研究集会を端緒として，より広い分野を包含し，より深化した Magma に関する研究集会が，再び開催されることを願う．

2010 年 11 月

木田雅成 (電気通信大学)
原田昌晃 (山形大学/JST さきがけ)
横山俊一 (九州大学)

研究集会「Magma で広がる数学の世界」

九州大学大学院数理学研究院グローバル COE プログラム「マス・フォア・インダストリ教育研究拠点」との共催で、下記のような研究集会を開催しますので、ご案内申し上げます。

世話人　木田 雅成（電気通信大学）
原田 昌晃（山形大学・ＪＳＴさきがけ）
横山 俊一（九州大学）

記

日程：　２０１０年１０月９日（土）～１０日（日）
場所：　九州大学大学院数理学研究院（伊都キャンパス）数理学研究教育棟

プログラム

１０月９日（土）

１０：００～１０：０５ 開会の挨拶
１０：０５～１１：００ 木田 雅成（電気通信大学大学院情報理工学研究科)
　初心者のための Magma 入門
１１：２０～１１：５０ 原田 昌晃（山形大学理学部・ＪＳＴさきがけ）
　符号理論における Magma
１１：５０～１２：１０ 諸連絡など

１４：００～１４：４５ 長尾 孝一（関東学院大学工学部）
　代数曲線暗号解読アルゴリズム研究での Magma の利用
１４：５５～１５：４０ 宗政 昭弘（東北大学大学院情報科学研究科）
　最小固有値 -2 を持つグラフとルート系
１６：００～１６：３０ 藤田 亮（中央大学研究開発機構）
　Magma のグレブナ基底計算を利用した暗号解析～多変数公開鍵暗号に対する代数攻撃～
１６：４０～１７：１０ 横山 俊一（九州大学大学院数理学府）
　Serre の保型性予想をめぐって：計算機的保型形式論入門

１０月１０日（日）

１０：００～１０：４５ 松野 一夫（津田塾大学学芸学部）
　代数体上の楕円曲線の計算と Magma
１０：５５～１１：４０ 千吉良 直紀（熊本大学大学院自然科学研究科）
　有限群論での Magma の利用
１１：５０～１２：２０ 大浦 学（高知大学理学部）
　モジュラー形式に関係ある不変式論
１２：２０～１３：０５ 交流会（兼昼食会）
１３：０５～１３：１０ 閉会の挨拶

最新情報については下記の研究集会のホームページをご覧下さい：
`http://www2.math.kyushu-u.ac.jp/~s-yokoyama/magma/`

初心者のための Magma 入門

木田雅成

1　この講演の目的

Magma は John Cannon をリーダーとするシドニー大学の計算代数グループで開発され, 頒布されている代数構造計算のためのソフトウェアである. Magma の簡単な紹介が [5] にある.

マニュアルには, Magma の特徴として,

設計思想　代数学と圏論に基づいて設計されている.
代数構造の明示　計算をおこなう代数構造を明示的に指定する必要がある.
統一性　どの対象においても一般的な構造は統一的に指定できる (商構造, 部分構造, 写像など).
構造の関連　例えば商構造を定義すると元の対象からの自然な射が自動的に定義される.
データベース　有限群, 楕円曲線などのデータベースをもっている.
効率　最速をめざして設計, 実装されている.

があげられている.

この講演では学部四年生程度までの基礎数学, 代数学を題材として Magma の使い方を解説する. その際に, 上記の特徴がなるべくはっきりとわかるように留意したい. ただしデータベースについてはこの講演ではふれることができないので, 松野さん, 千吉良さんにその紹介を譲ることにする.

2　Magma で計算してみる

実際に Magma を使って計算をやってみる. /*　*/ で囲まれた部分はコメントである. 一行だけのコメントには // も使う. 以下では, 行末にも必要に応じて日本語のコメントを付け加えてある.

Magma のコマンドはそのほとんどが省略形ではなく, 計算したいものの名詞形がそのまま用いられている. したがって以下では必要な箇所以外ではコマンド自体の説明を省略する. 長いコマンドは入力するのが大変と思われるかもしれないが, コマンドの途中で TAB キーを押すことにより, 入力補完, 候補の表示が行われるので, それほどの苦労はない.

2.1　関数電卓としての Magma

まずは四則演算を中心とした機能をみながら, Magma に慣れていくことにする.

```
> /*
> MAGMA for Beginners
>   at Kyushu University
>   Oct. 9, 2010
>   By Masanari Kida
> */
> 32123*1000;          // 文はセミコロン (;) で終わる
32123000
> 2^
> 20;                  // 途中で改行しても; までは計算されない
1048576
> 13/60+1/36;          // 有理数の計算は約分されて表示される
11/45
> q:=2^30 div 17;      // 2^30 を 17 で割った商を q に代入．結果は表示されない
> r:=2^30 mod 17;r;    // 結果を表示するためにこのような書き方も使う
13
> 2^30 eq 17*q+r;      // 検算．等号が成立することを調べるには eq を使う
true
```

関数電卓にグレードアップしよう.

```
> Log(2);                       // このような値はデフォルトの精度で計算される
0.693147180559945309417232121458
> Arccos(1/2);
1.047197551196597746154214461 09
> Pi(RealField())/3;            // Pi は精度つき実数体を引数にとる
1.047197551196597746154214461 09
> Pi(RealField(50));            // 精度を変えると
3.14159265358979323846264338327950288419716939937 51
> Sin($1/6);           // $1 には直前の結果が代入されている
0.500000000000000000000000000000000000000000000000
>  // この計算はあたえられた精度をもつ実数体で実行される
> ZetaFunction(3);     // このような数論に関係する関数もたくさんある
1.202056903159594285399738161 51
```

この項の最後にちょっと注意.

```
> a:=54534135111/2*4;a;  // 結果は整数に見えるが.....
109068270222
> Factorization(a);      // 因数分解ができない!

>> Factorization(a);
                 ^
Runtime error in 'Factorization': Bad argument types
Argument types given: FldRatElt

> Parent(a);            // a はどこの元かと調べると
Rational Field
> Factorization(IntegerRing()!a);   // 明示的に含まれる環を変えると因数分解できる
[ <2, 1>, <3, 1>, <18178045037, 1> ]
```

```
> RealField(50)!Log(2);            // 別の例
0.69314718055994530941723212145798186356626205936510
```

このように計算の対象の属する環や体を意識することが大切である.

2.2 集合, 数列, 写像など

Magma を使う際の基礎となる集合, 数列, 写像などの扱いかたをみる. これらがうまく使えるようになれば, Magma が飛躍的に役に立つ道具になる.

```
> S1:={1,2,3,2,3,3,5,7,7,9};        // 集合を作る
> S1;                                // 重複は無視される
{ 1, 2, 3, 5, 7, 9 }
> Random(S1);                        // ランダムな元をとる
5
> S2:={ x^3 : x in [1..20]};S2;      // 別の方法
{ 1, 8, 27, 64, 125, 216, 343, 512, 729, 1000, 1331, 1728, 2197, 2744, 3375, 4096, 4913,
5832, 6859, 8000 }
> S1 meet S2;                        // 集合の交わり
{ 1 }
> S3:={x : x in [1..100] | IsPrime(x) };S3;  // 条件を使った定義（素数の集合）
{ 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83,
89, 97 }
> TP:={[x,x+2] : x in [1..2000] | IsPrime(x) and IsPrime(x+2)}; // 少し複雑
> TP;   // これは双子素数の組
{
    [ 461, 463 ],
    [ 137, 139 ],
    [ 1049, 1051 ],
    [ 599, 601 ],
中略
    [ 239, 241 ],
    [ 347, 349 ],
    [ 59, 61 ],
    [ 1427, 1429 ],
    [ 1061, 1063 ]
}
> // 順序はバラバラ
> #TP;   // 集合の濃度
61
> L1:=[1,2,3,2,3,3,5,7,7,9];L1;    // 数列（リスト）
[ 1, 2, 3, 2, 3, 3, 5, 7, 7, 9 ]
> L1[7];
5
> L2:=[1..20];L2;                  // 連続する整数のリスト
[ 1 .. 20 ]
> LP:=[x : x in [1..100] | IsPrime(x)];LP;  // 素数のリスト
[ 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83,
89, 97 ]
> // 簡単な繰り返し文
```

```
// LP の元に対してオイラー関数を計算する
> for p in LP do
for>   EulerPhi(p);
for> end for;
1
2
4
6
中略
82
88
96
> // 簡単な条件文
> for p in LP do
for>    if IsPrime(p+2) then
for|if>      [p,p+2];
for|if>    end if;
for> end for;
[ 3, 5 ]
[ 5, 7 ]
[ 11, 13 ]
[ 17, 19 ]
[ 29, 31 ]
[ 41, 43 ]
[ 59, 61 ]
[ 71, 73 ]
```

Magma の集合・リストは通常の数学の書き方と非常に近い形で定義し使うことができる．リストをうまく使いこなすと, 複雑な計算を簡単に実行できることが多い．より複雑な集合や組構造の使い方が宗政さんの講演にあるので参考にしてほしい．

次に簡単な写像の使い方を見る．

```
> f:=map<Integers()->Integers() | x:->2*x >;   //  f(x) = x^2
> f(1);                                        //  f(1)
2
> 1@f;                                         //  別の書き方
2
> f([1,2,3]);                                  // f(2), f(4), f(6) のリスト
[ 2, 4, 6 ]
> g:=f^5;                                      // 写像の合成
> 1@g;
32
```

この項については, 良くまとまった解説が [1] にある．

2.3 素因数分解

Magma が内部でどのようなことをやっているのかのぞいてみよう. 素因数分解が題材である．

```
> SetVerbose("Factorization",1);  // 出力の冗長度を指定
> n:=155091189582461323967422273374060483191006034694003396144697710870373304529655232 1
1078730059982890210854424804938137751121970242869392787964301895870281115157593851998 41
02178816;
> Factorization(n);
Integer main factorization (primality of factors will be proved)
Effort: 3
Seed: 3706546146 66
    Number: 155091189582461323967422273374060483191006034694003396144697710870373304529 6
5523211078730059982890210854424804938137751121970242869392787964301895870281115157 59
38519984102178816

Pollard Rho
    Trials: 8191
    Number: 231661664844289682481058010180425143978317160529552433050689942523374022958 8
704158097520762177388796152 51
    (105 digits)
    Factor: 21727894856911 (14 digits)
    Cofactor: 106619470671179617891148702910743456307136150318626307032909495939009604 65
823539183339042941 (92 digits)
    Time: 0.010

Pollard Rho
    Trials: 8191
    Number: 21727894856911
    (14 digits)
    Factor: 3726911 (7 digits)
    Cofactor: 5830001 (7 digits)
    Time: 0.009

Pollard Rho
    Trials: 8191
    Number: 10661947067117961789114870291074345630713615031862630703290949593900960465 82
3539183339042941
    (92 digits)
    No factor found
    Time: 0.009

1 composite number remaining

ECM
    x: 10661947067117961789114870291074345630713615031862630703290949593900960465823539 1
83339042941
    (92 digits)
    Initial B1: 5000, limit: 858248
    Initial Pollard p - 1, B1: 45000
    Step 1; B1: 5000 [858248], digits: 92, elapsed time: 0.070
    Factor: 146234082633259 (15 digits)
    Cofactor: 7291013746677030493510653527522890463289832881050306657705546478817020159 1
799 (77 digits)
    Total ECM time: 0.190

ECM
```

```
    x: 72910137466770304935106535275228904632898328810503066577055464788170201591799
    (77 digits)
    Initial B1: 5070, limit: 176526
    Step 1; B1: 5070 [176526], digits: 77, elapsed time: 0.000
    Step 10; B1: 5725 [176526], digits: 77, elapsed time: 0.510
    Step 20; B1: 6500 [176526], digits: 77, elapsed time: 1.129
    Factor: 6722279202985811 (16 digits)
    Cofactor: 10846044215834722665042065816010288453465430807005605353874509 (62 digits)
    Total ECM time: 1.490

ECM
    x: 10846044215834722665042065816010288453465430807005605353874509
    (62 digits)
    Initial B1: 6824, limit: 17552
    Step 1; B1: 6824 [17552], digits: 62, elapsed time: 0.000
    Step 10; B1: 7582 [17552], digits: 62, elapsed time: 0.559
    Step 20; B1: 8472 [17552], digits: 62, elapsed time: 1.229
    Factor: 75045259055838983 (17 digits)
    Cofactor: 144526707646708212356380247022426585248301323 (45 digits)
    Total ECM time: 1.449

Total time: 3.220

[ <2, 206>, <67, 2>, <271, 1>, <5351, 1>, <3726911, 1>, <5830001, 1>, <146234082633259,
1>, <6722279202985811, 1>, <75045259055838983, 1>,
<144526707646708212356380247022426585248301323, 1> ]
> SetVerbose("Factorization",0);    // 冗長度をもとに戻す
```

小さな素数を取り除いたあと, ロー法, 楕円曲線法を繰り返し適用していく様子がわかる.

2.4 多項式環

有理整数環とともに代数学の基礎になる多項式環を扱ってみよう. $(x+1)^{20}$ を展開しようとして

```
> (x+1)^20;

>> (x+1)^20;
    ^
User error: Identifier 'x' has not been declared or assigned
```

となってエラーが出てしまう. Magma では計算の対象となる代数構造をあらかじめ定義しなくてはならない. いまの場合は,

```
> PZ<x>:=PolynomialAlgebra(Integers());
```

によってそれを行う. このコマンドにより有理整数環上の一変数多項式環 PZ($= \mathbb{Z}[x]$) を定義し, その変数を x とした. その名前 PZ は好きにつけてよいが, 何を表すかがある程度わかるように自分で名付けの規則を作るのがよい. あとで使う必要がない場合はアンダースコア (_) を使うと, 名前を付

けないでおくこともできる．これで x の数学的な意味がはっきりしたので, $(1+x)^{20}$ も問題なく計算することができる．

```
> (x+2)^20;
x^20 + 40*x^19 + 760*x^18 + 9120*x^17 + 77520*x^16 + 496128*x^15 + 2480640*x^14 +
    9922560*x^13 + 32248320*x^12 + 85995520*x^11 + 189190144*x^10 + 343982080*x^9 +
    515973120*x^8 + 635043840*x^7 + 635043840*x^6 + 508035072*x^5 + 317521920*x^4 +
    149422080*x^3 + 49807360*x^2 + 10485760*x + 1048576
> Factorization($1);              // 因数分解して確認
[
    <x + 2, 20>
]
```

以下では次のふたつの多項式 F1, F2 に対していろいろな計算を行う．

```
> F1:= x^3 - 210*x^2 + 1223743;
> F2:= 3*x^4 + 4*x^3 + x^2 - 9*x - 3;
> F1+F2;                                  // たし算
3*x^4 + 5*x^3 - 209*x^2 - 9*x + 1223740
> F1*F2;                                  // かけ算
3*x^7 - 626*x^6 - 839*x^5 + 3671010*x^4 + 4896859*x^3 + 1224373*x^2 - 11013687*x -   3671229
> Factorization(Discriminant(F1));        // 判別式の因数分解
[ <3, 6>, <17, 2>, <19, 1>, <103, 1>, <109, 2> ]
> F1/3;

>> F1/3;
     ^
Runtime error in '/': Argument 2 is not a unit
```

多項式の係数は $\mathbb{Z}$ なので判別式は $\mathbb{Z}$ の元, よって因数分解は問題なくできる．しかし 1/3 をかけることはこの環ではできない．

この問題を解決するためにはいろいろなやりかたがあるが，ここでは有理数体上の多項式環 $\mathbb{Q}[y]$ を定義し, $\mathbb{Z}[x]$ から自然な写像 v でこの環に F1, F2 を送る．

```
> PQ<y>:=PolynomialAlgebra(Rationals());
> v:=hom<PZ->PQ | y>;             //  x を y におくる準同型
> U1:=v(F1);U1;                    //  U1 = v(F1)
y^3 - 210*y^2 + 1223743
> U2:=v(F2);U2;                    //  U2 = v(F2)
3*y^4 + 4*y^3 + y^2 - 9*y - 3
> U1/3;                            // もちろん計算できる
1/3*y^3 - 70*y^2 + 1223743/3
> d,s,t:= ExtendedGreatestCommonDivisor(U1,U2);s;t;    // 返り値は 3 個
-14129458682629/20180107766364476035310370*y^3 +
    8488994468666693/605403232990934281059311 10*y^2 + 44521/610770131210897559*y +
    16490479102509401017/20180107766364476035310370
14129458682629/605403232990934281059311 10*y^2 -
    5815690424484493/605403232990934281059311 10*y +
    601977685893436261/605403232990934281059311 10
```

```
> ?ExtendedGreatestCommonDivisor        //  オンラインヘルプの参照
4 matches:
     1   I   /magma/ring-field-algebra/integer/gcd-lcm/ExtendedGreatestCommonDivisor
     2   I   /magma/ring-field-algebra/integer/gcd-lcm/ExtendedGreatestCommonDivisor
     3   I   /magma/ring-field-algebra/univariate-polynomial/common/gcd-lcm/ExtendedGreatestCommonDivisor
     4   I   /magma/ring-field-algebra/valuation/element/other/ExtendedGreatestCommonDivisor
To view an entry, type ? followed by the number next to it
```

この例のように Magma には 2 個以上の値を返す関数が多く存在する.

また, システムが適切に設定されていればウェブブラウザーからヘルプが参照できる.

```
> s*U1+t*U2;                      // 検算
1
> Evaluate(U1,34/21);             // 値の代入
11328025267/9261
```

多項式環のイデアルを定義し, その商環を作ってみる.

```
> I1:=ideal<PQ | U1>;                // イデアルの定義
> I2:=ideal<PQ | U2>;
> I1+I2;                                // イデアルの演算
Univariate Polynomial Ring in y over Rational Field
> I1 meet I2;
Ideal of Univariate Polynomial Ring in y over Rational Field generated by y^7 - 626/3*y^6
    - 839/3*y^5 + 1223670*y^4 + 4896859/3*y^3 + 1224373/3*y^2 - 3671229*y - 1223743
> IsPrime(I1);                          // 素イデアルか?
true
> IsIrreducible(U1);                     // ということは U1 は既約
true
> IsMaximal(I2);                        // 極大イデアルか?
false
> IsIrreducible(U2);
false
> Q1<a>:=quo< PQ | U1 >;        //  Q1 = ℚ [x]/(U1)
> a^5;                         // a には y の自然な像が代入される
8037257*a^2 - 256986030*a - 53967066300
> MinimalPolynomial(a);        // 検算
y^3 - 210*y^2 + 1223743
> IsIntegralDomain(Q1);       // 整域か?
true
> IsField(Q1);                  // 体か?
true
> Q2,q2:=quo< PQ | U2 >;       //  Q2 = ℚ [x]/(U2)
>  // q2 は ℚ [x] から Q2 への写像
> IsIntegralDomain(Q2);
false
> IsZeroDivisor((3*y+1)@q2);   //  (3y + 1) の q2 による像は零因子か?
true
```

構造を積み重ねてもっと複雑な環を作ってみる.

```
 > _<t>:=PowerSeriesRing(PQ); // ℚ [y] 上の形式的冪級数環
> 1/(1-y*t)^3;
1 + 3*y*t + 6*y^2*t^2 + 10*y^3*t^3 + 15*y^4*t^4 + 21*y^5*t^5 + 28*y^6*t^6 +
    36*y^7*t^7 + 45*y^8*t^8 + 55*y^9*t^9 + 66*y^10*t^10 + 78*y^11*t^11 +
    91*y^12*t^12 + 105*y^13*t^13 + 120*y^14*t^14 + 136*y^15*t^15
    + 153*y^16*t^16    + 171*y^17*t^17 + 190*y^18*t^18 + 210*y^19*t^19
    + O(t^20)
> gf:=t*Exp(y*t)/(Exp(t)-1);    // Bernoulli 多項式の母関数
> [PQ!(Factorial(n)*Coefficient(gf,n)): n in [0..10]];  // 係数を取り出す
[
    1,
    y - 1/2,
    y^2 - y + 1/6,
    y^3 - 3/2*y^2 + 1/2*y,
    y^4 - 2*y^3 + y^2 - 1/30,
    y^5 - 5/2*y^4 + 5/3*y^3 - 1/6*y,
    y^6 - 3*y^5 + 5/2*y^4 - 1/2*y^2 + 1/42,
    y^7 - 7/2*y^6 + 7/2*y^5 - 7/6*y^3 + 1/6*y,
    y^8 - 4*y^7 + 14/3*y^6 - 7/3*y^4 + 2/3*y^2 - 1/30,
    y^9 - 9/2*y^8 + 6*y^7 - 21/5*y^5 + 2*y^3 - 3/10*y,
    y^10 - 5*y^9 + 15/2*y^8 - 7*y^6 + 5*y^4 - 3/2*y^2 + 5/66
]
```

2.5 体とガロア理論

$U1 = y^3 - 210y^2 + 1223743$ に戻って, この多項式の根を $\mathbb{Q}$ に添加してできる体のことを調べることにする.

```
> K1<a1>:=NumberField(U1);     // a1 は U1 のひとつの根
> Degree(K1);                  // 次元
3
> a1^10;                        // 元の演算
1179802847765165400*a1^2 - 5131317430986797407*a1 - 8633019640022592196410
> S<s1>:=SplittingField(K1);S;    // S は K1 の分解体
Number Field with defining polynomial y^6 + 6*y^5 - 34*y^4 - 266*y^3 - 438*y^2 + 10*y +
    289 over the Rational Field
> _,u:=IsSubfield(K1,S);u;          // u は K1 から S への写像
Mapping from: FldNum: K1 to FldNum: S
> u(a1);                            // 原始元を送ってみる
1/10*(-5*s1^5 - s1^4 + 248*s1^3 + 93*s1^2 - 1409*s1 + 176)
> S!a1;                             // 別のやりかた
1/10*(-5*s1^5 - s1^4 + 248*s1^3 + 93*s1^2 - 1409*s1 + 176)
```

体 $K1$ の $\mathbb{Q}$ 上の分解体が S で 6 次の体であるから, $S/\mathbb{Q}$ のガロア群は S_3 に同型である.

```
> G,A,tau:=AutomorphismGroup(S);
```

G には S_6 の部分群として S_3 が代入され, A は S の $\mathbb{Q}$ 上の同型写像が入っている. τ は G から A への写像である.

```
> G;
Permutation group G acting on a set of cardinality 6
Order = 6 = 2 * 3
    (1, 2, 4)(3, 6, 5)
    (1, 3)(2, 5)(4, 6)
> G.1;                        // 生成元のひとつ
(1, 2, 4)(3, 6, 5)
> G.2;
(1, 3)(2, 5)(4, 6)
> Order(G.1);                     // 元の位数
3
> G.1*G.2;                       // かけ算
(1, 5)(2, 6)(3, 4)
> G.2*G.1;                         // 非可換
(1, 6)(2, 3)(4, 5)
> // A の各元に対して原始元の行き先を調べる
> for g in G do
for>   tau(g)(s1);
for> end for;
s1
1/60*(-s1^5 - s1^4 + 45*s1^3 + 53*s1^2 - 115*s1 - 101)
1/180*(13*s1^5 + 61*s1^4 - 519*s1^3 - 2753*s1^2 - 2213*s1 + 1775)
1/60*(s1^5 + s1^4 - 45*s1^3 - 53*s1^2 + 55*s1 - 79)
1/180*(7*s1^5 + 19*s1^4 - 321*s1^3 - 887*s1^2 + 733*s1 + 1205)
1/9*(-s1^5 - 4*s1^4 + 42*s1^3 + 182*s1^2 + 74*s1 - 176)
> FixedField(S,[tau(G.1)]);   // τ (G.1) で生成される群の固定体
Number Field with defining polynomial y^2 + 10*y - 7803 over the Rational Field
Mapping from: Number Field with defining polynomial y^2 + 10*y - 7803 over the Rational
Field to FldNum: S
> Subfields(S);      // 部分体を全部求める. tuple の列が帰ってくる
[
    <Number Field with defining polynomial y^2 - 137*y + 289 over the Rational Field,
    Mapping from: Number Field with defining polynomial y^2 - 137*y + 289 over the
    Rational Field to FldNum: S>,
    <Number Field with defining polynomial y^3 + 6*y^2 - 58*y - 289 over the Rational
    Field, Mapping from: Number Field with defining polynomial y^3 + 6*y^2 - 58*y - 289
    over the Rational Field to FldNum: S>,
    <Number Field with defining polynomial y^3 + 9*y^2 - 121*y - 289 over the Rational
    Field, Mapping from: Number Field with defining polynomial y^3 + 9*y^2 - 121*y - 289
    over the Rational Field to FldNum: S>,
    <Number Field with defining polynomial y^3 - 24*y^2 + 152*y - 289 over the Rational
    Field, Mapping from: Number Field with defining polynomial y^3 - 24*y^2 + 152*y - 289
    over the Rational Field to FldNum: S>,
    <Number Field with defining polynomial y^6 + 6*y^5 - 34*y^4 - 266*y^3 - 438*y^2 +
        10*y + 289 over the Rational Field, Mapping from: FldNum: S to FldNum: S>
]
> CS:=$1[2][1];   // ひとつの三次部分体をとる
> CS;
Number Field with defining polynomial y^3 + 6*y^2 - 58*y - 289 over the Rational Field
```

```
> FixedGroup(S,CS);  // CS を固定する群
Permutation group acting on a set of cardinality 6
    Id($)
    (1, 3)(2, 5)(4, 6)
Mapping from: GrpPerm: $, Degree 6 to GrpPerm: G
> #$1;           // その位数
2
```

このようにガロア対応が具体的に計算できる.

2.6 少しだけ整数論

整数論の講演がないので整数論について少しだけここでふれる. 代数的整数論の最初の大切な対象は $\mathbb{Q}$ の拡大体 K に含まれる代数的整数のなす環 O_K である. これは $K/\mathbb{Q}$ の拡大次数と同じ個数の基底をもつ自由 $\mathbb{Z}$ 加群で, その基底を K の整数底という. O_K の可逆元全体を慣用的に K の単数群という. この群は有限生成アーベル群にになる. O_K は一般に一意分解整域ではない. 一意分解整域からの遠さをはかる群がイデアル類群である. これは有限アーベル群である. これらを計算することが計算代数的整数論の第一目標になる. Magma では次のように計算することができる.

```
> CS;                    // 先ほどの S の三次部分体
Number Field with defining polynomial y^3 + 6*y^2 - 58*y - 289 over the Rational Field
> O:=MaximalOrder(CS);     // 整数環
> [S!O.k : k in [1..3]];    // 整数環の基底 を S の元として表示
[
    1,
    1/180*(-17*s1^5 - 77*s1^4 + 705*s1^3 + 3481*s1^2 + 1645*s1 - 3757),
    1/180*(-7*s1^5 - 31*s1^4 + 297*s1^3 + 1403*s1^2 + 419*s1 - 1253)
]
> U,fU:=UnitGroup(O);  // U は抽象的な単数群. fU は実際の単数群への射
> fU(U.1);             // 単数群の生成元. 整数底での座標表示
[-1, 0, 0]
> fU(U.2);               // 単数群の生成元
[1, 0, 1]
> C,fC:=ClassGroup(O);      // C は抽象的な群.
> I:=fC(C.1);               // イデアル類群の生成元
> IsPrincipal(I);           // 単項イデアルか?
false
> IsPrincipal(I^3);
false
> IsPrime(I);                 // 素イデアル
true
> rcf,pi:=ResidueClassField(I);   // O/I を作る
> rcf;                            // それは有限体になる
Finite field of size 2^2
> pi(O![1,2,0]);
1
```

2.7 Magma は本当に速いのか

以前 Magma と他のソフトウェアとの実行速度の比較をしたことがある ([4, 1.3 節]). それによれば, Magma は一般に高速であるといってよい. しかし次のような例もある. 2000 番目の Bernoulli 数を Magma の組込みの関数と, 金子さんに教えていただいた Zagier の公式で計算した結果の比較である.

Zagier の公式を Magma の関数として定義すると

```
BNByZagier:=function(k)
    h:=0;s:=1;c:=k+1;
    for n in [2..k+1] do
         c:=c*(n-k-2)/n;
         h:=h+c*s/n;
         s:=s+n^k;
    end for;
    return h;
end function;
```

となる. ここで引数 k は局所変数として扱われる. これを使って, 速さの比較をすると

```
> time  _:=BernoulliNumber(2000);
Time: 8.770
> time _:=BNByZagier(2000);
Time: 0.280
```

となって Zagier の公式の方が圧倒的に速いことがわかる.

Magma を終了するには

```
> quit;
```

とすればよい.

講演では十分にはふれられなかった整数論の計算については [3], [4] を参照してほしい. 後者には簡単なプログラムの例ものっている.

参考文献

[1] G. Bailey, *Appendix: the magma language*, Discovering mathematics with Magma (W. Bosma and J. Cannon, eds.), Algorithms and Computation in Mathematics, Springer-Verlag, 2006, pp. 331–356.

[2] W. Bosma, J. Cannon, and C. Playoust, *The Magma algebra system. I. The user language*, J. Symbolic Comput. **24** (1997), no. 3-4, 235–265, Computational algebra and number theory

(London, 1993).
[3] 木田雅成, 計算代数システム Magma による代数構造の計算, 数理研講究録別冊, vol. B19, 2010, pp. 107–116.
[4] _____, 数論研究者のための Magma 入門, 第 7 回北陸数論研究集会報告集, 2009, pp. 59–79.
[5] 原田昌晃, 木田雅成, *Magma*, 数学セミナー (2010) 9 月号, 44–47.

182-8585 調布市調布ヶ丘 1-5-1
電気通信大学数学教室
e-mail: kida@sugaku.e-one.uec.ac.jp

符号理論における Magma

山形大学理学部／ＪＳＴさきがけ
原田 昌晃

2010 年 10 月 9 日

1 はじめに

この原稿は, 九州大学で行なわれた研究集会「Magma で広がる数学の世界」での講演内容をまとめたものである．講演では, 著者の主な研究対象の一つである extremal doubly even self-dual code にターゲットを絞り, 聴講者の多くが非専門家であることを意識して, 比較的簡単に理解できるように（予備知識を比較的必要としない）幾つかのよく知られている結果を Magma での計算を交えながら説明した．つまり Magma を用いた extremal doubly even self-dual code に対する入門講義を行なった．なお, 講演では, 主催者であることもありこの研究集会が実施されるまでの経緯や, 著者自身の Magma との関わりについても述べたが, この原稿では数学的な内容についてのみ記述することにする．

では, この原稿で必要になる定義を述べることから始めよう．まず, 位数 2 の有限体を $\mathbb{F}_2$ で表す．(binary) $[n,k]$ code（符号）C とは $\mathbb{F}_2^n$ の k 次元部分空間のことである．ここで n を C の長さ, k を C の次元とよぶ．$x=(x_1,x_2,\ldots,x_n)\in\mathbb{F}_2^n$ の weight $\mathrm{wt}(x)$ を $|\{i \mid x_i\neq 0\}|$ とする．C の minimum weight を $\min\{\mathrm{wt}(x) \mid \mathbf{0}\neq x\in C\}$ で定義して $d(C)$ で表す, ただし $\mathbf{0}$ はゼロベクトルを表す．$[n,k,d]$ code は minimum weight が d である $[n,k]$ code を意味する．C の基底ベクトルを行に並べた $k\times n$ 行列を C の generator matrix とよぶ．C の weight enumerator とは $\mathbb{C}$ 上の 2 変数の多項式で $W_C(x,y)=\sum_{c\in C}x^{n-\mathrm{wt}(c)}y^{\mathrm{wt}(c)}$ である．C の dual code $C^\perp$ を $\{x\in\mathbb{F}_2^n \mid \ x\cdot y=0\ (\forall y\in C)\}$ で定義する, ただし $x\cdot y$ は標準的な内積を表す．$C=C^\perp$ であるとき C を *self-dual* とよぶ．全ての $x\in C$ に対して $\mathrm{wt}(x)\equiv 0 \pmod 4$ が成り立つとき *doubly even* とよぶ．C が self-dual code であれば全ての $x\in C$ に対して $\mathrm{wt}(x)\equiv 0 \pmod 2$ が成り立つが, doubly even となる self-dual code が存在し, この原稿で扱う code がこれである．

2 Doubly even self-dual code と Gleason の定理

まず doubly even self-dual code の例を 2 つ挙げる．

Example 1. 次の generator matrix

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

で定義される code は $[8,4,4]$ code になり extended Hamming $[8,4,4]$ code とよばれ, e_8 で表される. e_8 は self-dual code で, その weight enumerator は $W_{e_8}(x,y) = x^8 + 14x^4y^4 + y^8$ となる. したがって, 長さ 8 の doubly even self-dual code になることが分かる.

上の性質であれば手計算で確かめることが出来るが, Magma で code をどのように扱うかの紹介になるので Magma での確認の方法を載せる：

```
> e8:=LinearCode(KMatrixSpace(GF(2),4,8)![
> 1,1,0,1,0,0,0,1,
> 0,1,1,0,1,0,0,1,
> 0,0,1,1,0,1,0,1,
> 0,0,0,1,1,0,1,1
> ]);
> Length(e8);
8
> Dimension(e8);
4
> MinimumWeight(e8);
4
> IsSelfDual(e8);
true
> IsDoublyEven(e8);
true
> W<x,y>:=WeightEnumerator(e8);W;
x^8 + 14*x^4*y^4 + y^8
```

Example 2. 次の 12×24 行列を generator matrix とする code を extended Golay code G_{24} とよぶ：

$$\begin{pmatrix} \boxed{\quad v \quad} & & & & & 1 \\ & \boxed{\quad v \quad} & & & & 1 \\ & & \boxed{\quad v \quad} & & & 1 \\ & & & \ddots & & \vdots \\ & & & & \ddots & \vdots \\ & & & & \boxed{\quad v \quad} & 1 \end{pmatrix}$$

ただし $v = (1,1,0,0,0,1,1,1,0,1,0,1)$ $(\in \mathbb{F}_2^{12})$, 第 24 列目を除いて第 2 行目は第 1 行目を右に 1 つずらしたもので, 第 3 行目以降も同様に定められる（空白は 0 を表す）. G_{24}

は doubly even self-dual $[24, 12, 8]$ code で, その weight enumerator は

$$W_{G_{24}}(x, y) = x^{24} + 759x^{16}y^8 + 2576x^{12}y^{12} + 759x^8y^{16} + y^{24}$$

となることは, 例えば Magma を用いればすぐに確かめられる. なお, G_{24} の別の構成方法を後ほど紹介する.

次に doubly even self-dual code の weight enumerator についての考察を与える. 詳細は, 例えば [10, Chap. 19] を参照していただきたい.

Lemma 3. C を doubly even self-dual code とする. このとき次が成り立つ.

(1) $W_C(x, y) = W_C\Big(\dfrac{x+y}{\sqrt{2}}, \dfrac{x-y}{\sqrt{2}}\Big)$.

(2) $W_C(x, y) = W_C(x, iy)$ $(i = \sqrt{-1})$.

Proof. C の長さを n とする.
(1) MacWilliams 恒等式より $W_{C^\perp}(x, y) = \frac{1}{|C|}W_C(x+y, x-y)$ が成り立つ. $C = C^\perp$ より

$$W_C(x, y) = W_{C^\perp}(x, y) = \frac{1}{2^{n/2}}W_C(x+y, x-y) = W_C\Big(\frac{x+y}{\sqrt{2}}, \frac{x-y}{\sqrt{2}}\Big).$$

(2) 全ての $c \in C$ に対して $\mathrm{wt}(c) \equiv 0 \pmod 4$ なので

$$W_C(x, iy) = \sum_{c\in C} x^{n-\mathrm{wt}(c)}(iy)^{\mathrm{wt}(c)} = \sum_{c\in C} x^{n-\mathrm{wt}(c)}y^{\mathrm{wt}(c)} = W_C(x, y)$$

が成り立つ. □

Lemma 3 から次が成り立つ:

$$\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \circ W_C(x, y) = W_C(x, y),$$
$$\begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \circ W_C(x, y) = W_C(x, y),$$

ただし, 行列 $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ と多項式 $f(x, y)$ に対して $A \circ f(x, y) = f(ax + by, cx + dy)$ とする. この 2 つの行列で生成される群を考える:

$$G_{192} = \Big\langle \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \Big\rangle \ \ (\subset GL(2, \mathbb{C})).$$

この群は位数 192 の unitary reflection group で Shephard–Todd [12] のリストの No. 9 の群である. $W_C(x, y)$ は G_{192} によって不変な多項式全体

$$\mathbb{C}[x, y]^{G_{192}} = \{f(x, y) \in \mathbb{C}[x, y] \mid A \circ f(x, y) = f(x, y) \ (\forall A \in G_{192})\}$$

に含まれる. さらに, Gleason の定理としてよく知られている次の結果が成り立つ.

Theorem 4 (Gleason [4])**.** C を doubly even self-dual code とする．このとき

$$W_C(x,y) \in \mathbb{C}[x,y]^{G_{192}} = \mathbb{C}[W_{e_8}(x,y), W_{G_{24}}(x,y)]$$

が成り立つ.

Gleason の定理については MAGMA で次のような確認が出来る：

```
> G:=ShephardTodd(9);
> R:=InvariantRing(G);
> PI:=PrimaryInvariants(R);
> PI;
[
    x1^8 + 14*x1^4*x2^4 + x2^8,
    x1^24 + 10626/1025*x1^20*x2^4 + 735471/1025*x1^16*x2^8 +
        2704156/1025*x1^12*x2^12 + 735471/1025*x1^8*x2^16 +
        10626/1025*x1^4*x2^20 + x2^24
]
> SecondaryInvariants(R);
[
    1
]
> P<x,y>:=PolynomialRing(Rationals(),2);
> f1:=P!PI[1];f2:=P!PI[2];
> 1025/772*(f2-(10626/(1025*42))*f1^3);
x^24 + 759*x^16*y^8 + 2576*x^12*y^12 + 759*x^8*y^16 + y^24
```

今回の MAGMA による計算では 24 次の生成元は Theorem 4 に挙げられているものとは別のものを出力したが, 最後の行で $W_{G_{24}}(x,y)$ が得られることを確認した.

Gleason の定理より直ちに次が得られる：

Corollary 5. 長さ n の doubly even self-dual code が存在すれば $n \equiv 0 \pmod 8$.

さらに minimum weight に関する上限を得ることが出来る．証明については [11] を見ていただきたい.

Theorem 6 (Mallows–Sloane [11])**.** C を長さ n の doubly even self-dual code とすると $d(C) \leq 4\lfloor n/24 \rfloor + 4$ が成り立つ.

C を長さ n の doubly even self-dual code とする．$d(C) = 4\lfloor n/24 \rfloor + 4$ であるときに *extremal* とよぶ.

3 Extremal doubly even self-dual code の存在について

この節では extremal doubly even self-dual code の存在についての結果を紹介する．まず code の同値と自己同型群の定義を述べよう．C, C' を $[n,k]$ code とする．n 次対称群

S_n の元 σ に対して

$$C^\sigma = \{(x_{\sigma^{-1}(1)}, \ldots, x_{\sigma^{-1}(n)}) \mid (x_1, \ldots, x_n) \in C\}$$

とするとき, $C^\sigma = C'$ となる $\sigma \in S_n$ が存在する場合に C と C' を同値とよび, $C \cong C'$ で表す. また S_n の部分群 $\{\sigma \in S_n \mid C^\sigma = C\}$ を C の自己同型群とよび $\mathrm{Aut}(C)$ で表す.

表 1: Extremal doubly even self-dual code について

長さ n	$d(C)$	N_n	Reference
8	4	1	Pless [15]
16	4	2	Pless [15]
24	8	1	Pless [14]
32	8	5	Conway–Pless–Sloane [2]
40	8	≥ 12579	King [8]
48	12	1	Houghten–Lam–Thiel–Parker [9]
56	12	≥ 1151	Harada [5]
64	12	≥ 3270	Harada–Kimura [6]
72	16	?	

長さ n の非同値な extremal doubly even self-dual code の個数を N_n で表すことにし, N_n について現時点で知られている結果を紹介しよう. 表 1 の 1 列目には長さ n, 2 列目には extremal となる minimum weight の値 $d(C)$, 3 列目には N_n の値, 4 列目には文献が与えられている.

この表についての補足説明を行なう. 長さ 32 までは extremal に限らず全ての doubly even self-dual code の分類が完成している（[2] を参照）. すでに例として挙げた e_8 と G_{24} はそれぞれの長さにおいて同値を除いて一意的に存在する extremal doubly even self-dual code であることが分かる. 長さ 48 の場合にのみ extremal についての分類が完成している.

長さ $n \leq 64$ ($n \equiv 0 \pmod 8$) については存在が古くから分かっていたこともあり, 1973 年には Sloane [13] は長さ 72 での存在について気にしており, その存在を決定することを提案している.

Problem A (Sloane [13])**.** 長さ 72 の extremal doubly even self-dual code は存在するか？

この問題は 35 年以上経った現在でも解決されておらず, 代数的符号理論における有名な未解決問題の 1 つと言える.

さて, 講演中では extended quadratic residue code が幾つかの長さで extremal doubly even self-dual code になることを紹介した. extended quadratic residue code の定義やその性質については例えば [10, Chap. 16] を見ていただくことにして, 素数 $p \equiv 7 \pmod 8$ に対して定義される長さ p の quadratic residue code の extended code は doubly even self-dual code になることが分かっている. 例えば Example 1 と 2 で与えられている e_8

と G_{24} は extended quadratic residue code として構成されている．長さ $24, 32, 48$ では extended quadratic residue code は extremal になることが次のようにすれば確認出来る：

```
> isExtremalQR:=function(n)
function>   C:=ExtendCode(QRCode(GF(2),n-1));
function>   if MinimumWeight(C) eq 4*(n div 24)+4 then;
function|if>     return "extremal";
function|if>   else
function|if>     return "non-extremal";
function|if>   end if;
function> end function;
>
> [isExtremalQR(x): x in [24,32,48]];
[ extremal, extremal, extremal ]
```

次の長さである 72 では残念ながら extremal にならない．しかし, その次の長さ $80, 104$ では extremal になることが分かる．

```
> [isExtremalQR(x): x in [72,80,104]];
[ non-extremal, extremal, extremal ]
```

この辺りが長さ 72 の extremal doubly even self-dual code の存在をミステリアスにしている一つなのかもしれない．

4 $|a+x|b+x|a+b+x|$ 構成法で e_8 から G_{24} を作る

まず $|a+x|b+x|a+b+x|$ 構成法とよばれるものを紹介しよう [10, p. 587]. C_1 を $[n, k_1]$ code, C_2 を $[n, k_2]$ code とするとき

$$C_1 \star C_2 = \{(a+x, b+x, a+b+x) \in \mathbb{F}_2^{3n} \mid a, b \in C_1, x \in C_2\}$$

が $[3n, 2k_1+k_2]$ code となることが分かり, この構成法を $|a+x|b+x|a+b+x|$ 構成法とよぶ. M_1 と M_2 をそれぞれ C_1 と C_2 の generator matrix とするとき, その構成法から

$$\begin{pmatrix} M_1 & O & M_1 \\ O & M_1 & M_1 \\ M_2 & M_2 & M_2 \end{pmatrix}$$

は $C_1 \star C_2$ の generator matrix となることが直ぐに分かる, ここで O は M_1 と同じサイズの零行列を表す. 定義より直ちに次を示すことが出来る.

Lemma 7. C_1 と C_2 がともに長さ n の doubly even self-dual code であれば $C_1 \star C_2$ は長さ $3n$ の doubly even self-dual code である.

Turyn による $|a+x|b+x|a+b+x|$ 構成法を用いた G_{24} の構成が MacWilliams–Sloane の本 [10] の 588 ページに書かれてある. $\sigma = (1,7)(2,6)(3,5) \in S_8$ であれば $d(e_8 \star {e_8}^{\sigma}) = 8$, つまり, 長さ 24 の extremal doubly even self-dual code の一意性（表 1 を参照）から $e_8 \star {e_8}^{\sigma} \cong G_{24}$ となる. このことを MAGMA で確認をしてみよう.

```
> Turyn:=function(C,x)
function>   O:=ZeroMatrix(GF(2),Dimension(C),Length(C));
function>   M1:=GeneratorMatrix(C);
function>   N1:=HorizontalJoin(M1,HorizontalJoin(O,M1));
function>   N2:=HorizontalJoin(O,HorizontalJoin(M1,M1));
function>   M2:=GeneratorMatrix(C^x);
function>   N3:=HorizontalJoin(M2,HorizontalJoin(M2,M2));
function>   N4:=VerticalJoin(N1,VerticalJoin(N2,N3));
function>     return LinearCode(N4);
function> end function;
>
> MinimumWeight(Turyn(e8,Sym(8)!(1,7)(2,6)(3,5)));
8
```

次に, どの置換 $\sigma \in S_8$ の場合に $e_8 \star {e_8}^{\sigma}$ は G_{24} になるのかを考えよう. まず, 長さ 8 の doubly even self-dual code は同値を除けば e_8 のみであることから, $e_8 \star {e_8}^{\sigma}$ だけを考えれば良いことが理解出来るだろう. さらに, 長さ 8 の doubly even self-dual code の集合は $\{{e_8}^{\sigma} \mid \sigma \in T\}$ で与えられる, ただし, $T = \{\sigma_1, \sigma_2, \ldots, \sigma_{30}\}$ は $\mathrm{Aut}(e_8) \setminus S_8$ の完全代表系を表す. ここで $|\mathrm{Aut}(e_8)| = 1344$ なので $|\mathrm{Aut}(e_8) \setminus S_8| = 30$ であることに注意しておく.

次のプログラムによって $|\{\sigma \in T \mid d(e_8 \star {e_8}^{\sigma}) = 8\}| = 8$ であることが確認出来る, つまり $e_8 \star {e_8}^{\sigma} \cong G_{24}$ となる置換 $\sigma \in T$ は 8 個であることが分かる：

```
> Aut:=AutomorphismGroup(e8);
> S:=Sym(Length(e8));
> [#Aut,Index(S,Aut)];
[ 1344, 30 ]
> T:=Transversal(S,Aut);
> perm:=[x: x in T|MinimumWeight(Turyn(e8,x)) eq 8];
> #perm;
8
```

$d(e_8 \star {e_8}^{\sigma}) = 8$ となる σ について調べてみよう.

```
> pc:=LinearCode<GF(2),Length(e8)|[1 : i in [1..Length(e8)]]>;
> {pc eq (e8 meet e8^x): x in perm};
{ true }
> #{x: x in T| pc eq (e8 meet e8^x)};
8
```

上の１つ目の計算から分かることは $d(e_8 \star {e_8}^\sigma) = 8$ となる 8 個の置換 σ に対して $e_8 \cap {e_8}^\sigma = \langle(1,1,1,1,1,1,1,1)\rangle$ が成り立つことである．次の計算から $e_8 \cap {e_8}^\sigma = \langle(1,1,1,1,1,1,1,1)\rangle$ となる T の置換 σ は 8 個であることが分かる．以上の計算から次が成り立つ．なお，この事実は理論的にも簡単に証明出来る（Proposition 9 の証明を参照）．

Proposition 8. $d(e_8 \star {e_8}^\sigma) = 8$ である必要十分条件は $e_8 \cap {e_8}^\sigma = \langle(1,1,1,1,1,1,1,1)\rangle$.

以上，どのような置換が $|a+x|b+x|a+b+x|$ 構成法で e_8 から G_{24} を作るかについて調べてみた．では，G_{24} からまだ存在の分かっていない長さ 72 の extremal doubly even self-dual code を構成することが出来ないだろうか．

Problem B. $G_{24} \star {G_{24}}^\sigma$ が長さ 72 の extremal doubly even self-dual code となる置換 $\sigma \in S_{24}$ が存在するか？

```
> G24:=Turyn(e8,Sym(8)!(1,7)(2,6)(3,5));
> Aut:=AutomorphismGroup(G24); S:=Sym(Length(G24));
> [#Aut,Index(S,Aut)];
[ 244823040, 2534272925184000 ]
> T:=Transversal(S,Aut);

>> T:=Transversal(S,Aut);
                  ^
Runtime error in 'Transversal': Index of subgroup is too large
```

上の計算では，まず $|\operatorname{Aut}(G_{24})| = 244823040$, $|\operatorname{Aut}(G_{24}) \setminus S_{24}| = 2534272925184000$ であることを確認している．最後のエラーはそのメッセージから分かる通り指数が大きいことで Magma では計算出来ないことを表示している．$\operatorname{Aut}(G_{24}) \setminus S_{24}$ の代表元は Dixon–Majeed [3] に与えられている方法で求めることが出来，Magma にも実装出来るので，このエラーの部分はクリア出来る．次に Proposition 8 に対応する結果を考えよう．

Proposition 9. $G_{24} \cap {G_{24}}^\sigma \neq \langle(1,1,\ldots,1)\rangle$ ならば $d(G_{24} \star {G_{24}}^\sigma) \leq 12$, つまり extremal にならない．

Proof. $G_{24} \cap {G_{24}}^\sigma \supset \{\mathbf{0}, \mathbf{1}\}$ であることに注意，ただし $\mathbf{1} = (1,1,\ldots,1)$ とする．$x \in G_{24} \cap {G_{24}}^\sigma$ とすると $(x,\mathbf{0},x), (x,x,x) \in G_{24} \star {G_{24}}^\sigma$ であることが構成法から直ちに分かる．したがって $(x,\mathbf{0},x) + (x,x,x) = (\mathbf{0},x,\mathbf{0}) \in G_{24} \star {G_{24}}^\sigma$ で $\operatorname{wt}(x) = \operatorname{wt}((\mathbf{0},x,\mathbf{0}))$ である．もし $\operatorname{wt}(x) = 8, 12$ であれば $G_{24} \star {G_{24}}^\sigma$ は extremal にならない．$\operatorname{wt}(x) = 16$ のときは $\operatorname{wt}(x+\mathbf{1}) = 8$ で $x + \mathbf{1} \in G_{24} \cap {G_{24}}^\sigma$ より上の場合に帰着される． □

定義は割愛するが，G_{24} の weight $8, 12$ の codeword の集合はそれぞれ 5-$(24,8,1)$ design, 5-$(24,12,48)$ design とよばれる組合せ構造になる．これらの design が交わらないようにどれだけ与えられるかということは古くから調べられている（例えば [1] を参照）．実際に $G_{24} \cap {G_{24}}^\sigma = \langle(1,1,\ldots,1)\rangle$ となる置換 $\sigma \in S_{24}$ は簡単に見付けることが出来る．実際に構成するところまでいかなくても，置換に関する条件を決定することが出来れば非常に興味深いと思う．

Problem C. $d(G_{24} \star {G_{24}}^{\sigma}) = 16$ となる σ に関する必要十分条件が得られるか？

講演中にも紹介した通り（この原稿を書いている時点でも）$d(G_{24} \star {G_{24}}^{\sigma}) = 16$ となる σ を探す計算を続けているが, 上から分かる通り, 全ての可能性を調べるのは難しいし, もちろん何らかの代数的な結果が必要である. また, extremal なものが存在するかどうかも全く見当が付かないし, もし存在したとしても, この方法で構成出来る見通しがある訳でもないことに注意しておく（少しだけこの方法で探す理由を講演では述べたが論理的な根拠があるわけではないので, この原稿で述べることは避ける）.

5 最後に

いくつかコメントを与えて, この原稿を終える. まず, 今回, $|a+x|b+x|a+b+x|$ 構成法で, 長さ 8 から長さ 24 の extremal doubly even self-dual code を構成し, また, 長さ 24 から長さ 72 の extremal doubly even self-dual code が構成出来ないかということを述べた. C_1, C_2 を長さ n の extremal doubly even self-dual code とすると, 構成法より $d(C_1 \star C_2)$ は高々$2d(C_1) = 2(4\lfloor n/24 \rfloor + 4)$ となる. したがって $C_1 \star C_2$ が extremal になるためには

$$2\left(4\left\lfloor \frac{n}{24} \right\rfloor + 4\right) \geq 4\left\lfloor \frac{3n}{24} \right\rfloor + 4$$

でなければいけない. この不等式を満たす n ($\equiv 0 \pmod 8$) は 8 と 24 だけである. つまり, 今回考えた場合以外では extremal を構成することは出来ないことを意味する.

この講演では $\mathbb{F}_2$ 上の code のみを扱った. その他の有限体 $\mathbb{F}_q$ や有限環 $\mathbb{Z}/k\mathbb{Z}$ 上の self-dual code の研究も活発に行なわれており, 特に, 分類結果については, Harada–Munemasa [7] によるデータベースを参考にしていただきたい. なお, このデータベースでは Magma のフォーマットで書かれた code のデータが与えられている.

Magma を用いて研究を行なっている際に最大のメリットだと思う点[1]として, Magma は代数系を中心として幅広い分野における計算を行なえる統合ソフトであり, 複数の分野にまたがる計算をする際に, それぞれ分野に特化したソフトを使用することなく, Magma だけで扱うことが出来ることが挙げられる. この講演では, 特に, extended Hamming $[8,4,4]$ code e_8 と extended Golay $[24,12,8]$ code G_{24} を, 他の分野の方が触れる可能性が高いと期待しつつ, 主人公にした話題を選んだ. 最後に, 符号理論以外の分野の方が Magma で code を扱う際に, この原稿が何らかの役に立つことを願っている.

参考文献

[1] M. Araya and M. Harada, Mutually disjoint Steiner systems $S(5,8,24)$ and 5-$(24,12,48)$ designs, *Elect. J. Combin.* **17** (2010), #N1 (6 pp.).

[2] J.H. Conway, V. Pless and N.J.A. Sloane, The binary self-dual codes of length up to 32: a revised enumeration, *J. Combin. Theory Ser. A* **60** (1992), 183–195.

[1] 最先端のアルゴリズムが実装されている場合が多いことも忘れてはいけないメリットである.

[3] J.D. Dixon and A. Majeed, Coset representatives for permutation groups, *Portugal. Math.* **45** (1988), 61–68.

[4] A.M. Gleason, Weight polynomials of self-dual codes and the MacWilliams identities, *Act. Congr. Int. Math.*, Vol. 3, pp. 211–215, Gauthier-Villars, Paris, 1971.

[5] M. Harada, Self-orthogonal 3-$(56, 12, 65)$ designs and extremal doubly-even self-dual codes of length 56, *Des. Codes Cryptogr.* **38** (2006), 5–16.

[6] M. Harada and H. Kimura, New extremal doubly-even $[64, 32, 12]$ codes, *Des. Codes Cryptogr.* **6** (1995), 91–96.

[7] M. Harada and A. Munemasa, Database of Self-Dual Codes, `http://www.math.is.tohoku.ac.jp/~munemasa/selfdualcodes.htm`.

[8] O.D. King, The mass of extremal doubly-even self-dual codes of length 40, *IEEE Trans. Inform. Theory* **47** (2001), 2558–2560.

[9] S.K. Houghten, C.W.H. Lam, L.H. Thiel and J.A. Parker, The extended quadratic residue code is the only $(48, 24, 12)$ self-dual doubly-even code, *IEEE Trans. Inform. Theory* **49** (2003), 53–59.

[10] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam 1977.

[11] C.L. Mallows and N.J.A. Sloane, An upper bound for self-dual codes, *Inform. Control* **22** (1973), 188–200.

[12] G.C. Shephard and J.A. Todd, Finite unitary reflection groups, *Canadian J. Math.* **6** (1954), 274–304.

[13] N.J.A. Sloane, Is there a $(72, 36)$ $d = 16$ self-dual code? *IEEE Trans. Inform. Theory* **19** (1973), 251.

[14] V. Pless, On the uniqueness of the Golay codes, *J. Combin. Theory* **5** (1968), 215–228.

[15] V. Pless, A classification of self-orthogonal codes over GF(2), *Discrete Math.* **3** (1972), 209–246.

代数曲線暗号解読アルゴリズム研究での Magma の利用

長尾孝一
nagao@kanto-gakuin.ac.jp

関東学院大学 工学部 基礎・教養教室

1 はじめに

この発表では、ANTS-IX. シンポジウム（Nancy 2010）[9] で話した研究に数式処理ソフト magma を使ったことをお話いたします。昨今の公開鍵暗号では、有限体上定義された楕円曲線の群や曲線の Jacobian 群を使う、楕円曲線 Elgamal 型暗号が使われています。これらの暗号は、その群の離散対数問題 (discrete logarithm problem 以下 DLP と略記する) を解く事が難しいということを安全性の根拠としています。[9] で筆者は、定義体が（拡大次数の小さい）拡大体である場合について、曲線の Jacobian 群の DLP を解く新しい方式を提案しています。論文では（楕円曲線を含む）超楕円曲線の Jacobian 群に関する結果を中心に述べていますが、これはこの場合には方程式系の導出が単純であり論じやすかった為であり、実際には論文前半に述べた一般的な方式で、一般の拡大体上定義された曲線の Jacobian 群の DLP についても、それを解く Algorithm を与えています。この方式では、ある種の多次多変数の方程式系を（たくさん）解くことが必要とされます。（具体的には、有限体上定義された代数曲線 $C/\mathbb{F}_{q^n}$ の Jacobian 群の DLP を解く問題については、拡大次数 n と $genus(C)$ が小さい時、2 次、$(n^2-n)\times genus(C)$ 個の変数、同じ個数の方程式を持つ方程式系を何度も解く事が必要とされます。）
実際にそのような多次多変数の方程式系の零点の次元が 0 で方程式系の解があることが必要である為、magma を使い実際の具体例で方程式を解き、方式に誤りのないことのチェックに使っています。この発表では、まず最初に、超楕円曲線の Jacobian 群の計算について述べ、次に、一般の有限群の DLP の Index Calculus による解法について述べ、それから本題である、Gaudry, Diem, Thériault や筆者とうによって研究がなされている、代数曲線の Jacobian 群に Index Calculus を適応する話題について述べ、最後に方程式系の計算に magma をどう使ったかについて述べます。

2 超楕円曲線の Jacobian 群

K を体（ここでは簡単のために、体 K の標数は 2 ではないとします）、C/K を体 K 上で奇数次式で定義された、種数 g の超楕円曲線

$$C/K : y^2 = x^{2g+1} + ... + a_0$$

とします。ここで、式が奇数次式で定義されるので、曲線は unique な無限遠点 ∞ を持ちます。曲線 C/K の Jacobian 群は通常以下で定義されます。ここで、$\sim$ は線形関係を表します。

$$Jac(C/K) := \{Div_C(K)\}/\sim .$$

この定義は、よくわからないので、厳密さを欠きますが、Jacoobian 群の定義を以下で与えます。

$$Jac(C/K) \doteqdot \{\{P_1, ..., P_g\} | P_i \in C(\bar{K}), Gal(\bar{K}/K) \text{ で } \{P_i\} \text{ は不変 }\}.$$

要するに、Jacobian 群の元は、曲線 C 上の g 個の点の組 $\{P_i\}$ であるということです。

曲線 C 上の点 $P = (x, y)$ について、$\bar{P} := (x, -y)$ とおきます。Jacobian 群の元 $D = \{P_i\}$ のマイナス倍をを $(-D) := \{\bar{P}_i\}$ と置きます。また、$D_1 := \{P_1, ..., P_g\}$, $D_2 := \{P_{g+1}, ..., P_{2g}\}$, $D_3 := \{P_{2g+1}, ..., P_{3g}\}$ とし、$P_1, ..., P_{3g}$ がある有理関数 $y = f(x)/g(x)$ と C の全ての交点と一致するとき、$D_1 + D_2 + D_3 = 0$ であると演算を入れます。この演算によって Jacobian 群は（トートロジーですが）群になります。以下の図は genus=3 の場合を表しています。

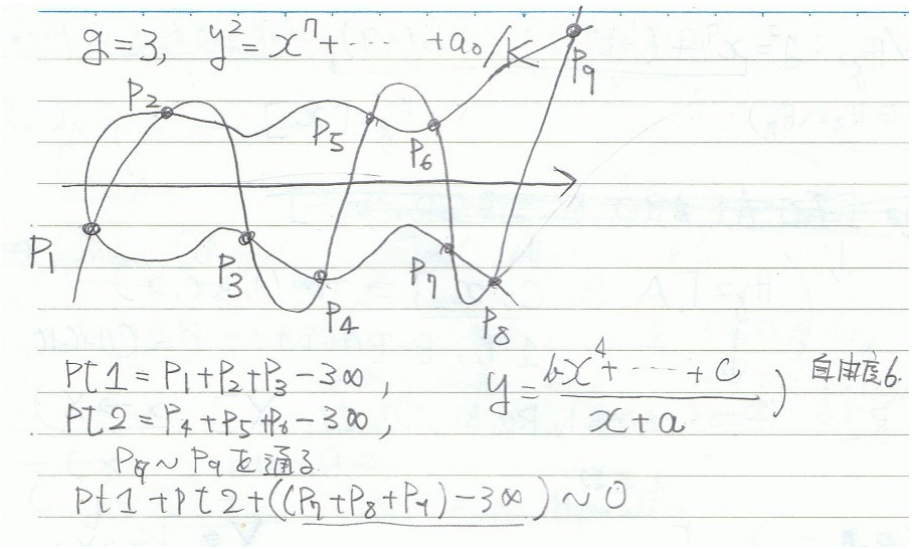

Fig. 1. 超楕円曲線の群演算

組 $\{P_i\}$ で書かれる Jacobian 上の点は通常 divisor というものを使って $D = P_1 + ... + P_g - g\infty \in Jac_C(K)$ とかかれます。この D について、以下では、通常 Munford 表現と呼ばれる、超楕円曲線の Jacobian 点の表現を説明します。

D に対して、$m_1(x), m_2(x)$ を以下を満たす多項式とします。

$$m_1(x) := \prod_{i=1}^{g} (x - x(P_i)) \in K[x],$$

$$m_2(x) \in K[x] \, such \, that \quad m_2(x(P_i)) = y(P_i), \deg(m_2) < \deg(m_1).$$

Munford 表現では $(m_1(x), m_2(x))$ を Jacobian の元と解釈します。

magma では、超楕円曲線の Jacobian 群の計算は munford 表現でなされています。以下のプログラムでは、$p = 10007$、$t \in \mathbb{F}_{p^2} \backslash \mathbb{F}_p$ とし、genus 3 の超楕円曲線 $C/\mathbb{F}_{q^2} : y^2 = x^7 + (-t^6 + t)x + (6t + 9)$ を定義し、その Jacobian 群を計算しています。基本的に Jacobiann 群の元は genus 個（今の場合 3 個）の曲線上の点から作られますが、genus 個より小さな点から作られることもあります。以下の例では、pt0 は 1 個の点、pt1 は 2 個の点から作られているので、magma の Munford 表現計算の最後に 1,2 という数字がついています。

```
p:=10007;
Fp:=GF(p);
Fq<t>:=GF(p,2);
R<T>:=PolynomialRing(Fp);
R0<x>:=PolynomialRing(Fq);
Cafq:=-t^6+t;
Cbfq:=6*t+9;
F0:=x^7+(R0!Cafq)*x+(R0!Cbfq);
Cu:=HyperellipticCurve(F0);
print "Cu=",Cu;
J:=Jacobian(Cu);
pt0:=Cu![t,t+3];
pt0:=J![pt0, Cu![1,0,0]];
print "pt0=",pt0;
pt1:=2*pt0; print "pt1=",pt1;
pt2:=2*pt0; print "pt2=",pt2;

> load "C:\\wk\\kyudai1.txt";
Loading "C:\wk\kyudai1.txt"
Cu= Hyperelliptic Curve defined by y^2 = x^7 + (568*t + 468)*x + 6*t
+ 9 over GF(10007^2)
pt0= (x + 10006*t, t + 3, 1)
pt1= (x^2 + 10005*t*x + 4867*t + 1955, (7929*t + 6086)*x + 471*t +
9658, 2)
pt2= (x^3 + (4073*t + 4609)*x^2 + (7997*t + 9717)*x + 8935*t + 5656,
(1731*t +    1603)*x^2 + (90*t + 8660)*x + 7361*t + 7710, 3)
> 1234*pt2;
(x^3 + (8344*t + 7876)*x^2 + (4978*t + 452)*x + 8235*t + 6517, (9600
*t + 3865)*x^2 + (2998*t + 2590)*x + 8633*t + 8004, 3)
```

3 離散対数問題と Index calculus

公開鍵暗号の世界では、有限体の乗法群 $\mathbb{F}_p^*$ を使った Elgamal 方式の暗号が提案され実際に使われていた。楕円曲線暗号を含む代位数曲線暗号は暗号に使う群を有限体の乗法群から、曲線の Jacobian 群にかえたものである。ここでは、まず有限体の乗法群について述べる。

Definition 1 (離散対数問題（DLP)) $a, b \in \mathbb{F}_p^* st. a^n = b => Find\ n$

しばらくの間、Index Calculus と呼ばれている離散対数問題を解く方式について説明する。Factor base と呼ばれる集合を下記のように -1 と小さな素数からなる集合とする。

$$B_0 = \{-1, 2, 3, 5, 7..., p_n\}.$$

このとき、$a^i b^j \in < B_0 >$ となる i, j を $|B_0| + 1$ 個以上見つけたとすると、役 $|B_0| \times |B_0|$ の大きさの $|\mathbb{F}_p^*|$ を法とする線形代数を解くことによって DLP は解くことができる。このことは、以下の例でみるとわかり易い。

例 $p = 179$, $a = 23$, $b = a^{23} = 111$, $B_0 = \{2, 3\}$

関係式を集める

$$\begin{cases} a^1 \cdot b^{20} = 96 = 2^5 3^1 \\ a^2 \cdot b^{16} = 12 = 2^2 3^1 \\ a^3 \cdot b^{17} = 27 = 2^0 3^3 \end{cases}$$

mod $p - 1$ での線形代数を解く

$$\left[\begin{array}{cc|cc} 1 & 20 & 5 & 1 \\ 2 & 16 & 2 & 1 \\ 3 & 17 & 0 & 3 \end{array}\right]$$

$$\left[\begin{array}{cc|cc} 3 & 60 & 15 & 3 \\ 6 & 48 & 6 & 3 \\ 3 & 17 & 0 & 3 \end{array}\right]$$

$$\left[\begin{array}{cc|cc} 0 & 43 & 15 & 0 \\ 3 & 31 & 6 & 0 \\ 3 & 17 & 0 & 3 \end{array}\right]$$

$$\left[\begin{array}{cc|cc} 0 & 86 & 30 & 0 \\ 15 & 155 & 30 & 0 \end{array}\right]$$

$$\left[\begin{array}{cc|cc} 15 & 69 & 0 & 0 \end{array}\right] \times -1/69 \bmod 178$$

$$\left[\begin{array}{cc|cc} 23 & -1 & 0 & 0 \end{array}\right]$$

DLP が解かれた：$a^{23} \cdot b^{-1} = 1 \bmod 178$.

ここで、関係式を集める計算では、実際に $a^i b^j \bmod p$ を計算し、それを素因数分解し、$< B_0 >$ の元であるか否かを判別している。本論文で筆者がやったことは、群が拡大体上定義された曲線の Jacobian 群であった時に $a^i b^j$ に相当する Jacobian 群の元が $< B_0 >$ の元であるか否かの判定方法であり、その判定はある種の方程式系を解くことによって与えられる。つまり、通常の Index calculus で素因数分解に相当する箇所がある種の方程式系の求解に取って代わられるということである。

Index Calculus では下記に述べる Large prime の利用によってその計算が高速化（主に線形代数のサイズを小さくするのに役立つ）される。

$$B = \{-1, 2, 3, 5, 7..., p_N\}, \quad (p_n \text{は } N \text{ 番目の素数、 })B_0 \subset B$$

とする。ここで、B_0 は先ほどと同じく Factor Base の元の集合であり、$B \backslash B_0$ の元は（作り方から大きな素数の集まりであるので）Large prime と呼ばれる。

$a^i b^j \in < B >$ となる i, j を潤沢にあつめ、まず Large prime$B \backslash B_0$ の項を消去しその後 役 $|B_0| \times |B_0|$ サイズの $|\mathbb{F}_p^*|$ を法とする線形代数を解くことによって、DLP を解くことにより高速化がなされる。

次に一般の Jacobian 群等の演算が加法で書かれている群についての DLP を定義しておく。

Definition 2 (離散対数問題（DLP)) *G (Additive) Group, Solve DLP i.e. $a, b \in G$ s.t. $n \cdot a = b$ => Find $n \in \mathbb{Z}/|G|\mathbb{Z}$*

B, B_0 を G の部分集合で $B_0 \subset B \subset G$ を満たすものとする。ここで、B_0 の元を Factor base、$B \backslash B_0$ の元を Large prime と呼ぶ。また、Index Calculus が上手く動く為に、以下の仮定が成り立つものとする。（$B \subset G$ が以下の仮定を満たすように取れれば、Index Calclus は上手く動く）

Assumption 1 (Assumption of Decomposition) 小さな整数 N が存在して、任意の $g \in G$ に対して、$g = g_1 + g_2 + ... + g_N$ $(g_i \in B)$ と分解する確率は $O(1)$ であり、g がこのような形で書けるか否かの判定と $\{g_i\}$ の計算コストも $O(1)$ で抑えられる。

実際、群 G が種数 g の拡大体 $\mathbb{F}_{p^n}$ 上定義された曲線 C の Jacobian 群であった時、$N = ng$, $B = \{(x, y) - \infty \,|(x, y) \in C(\mathbb{F}_{p^n}), x \in \mathbb{F}_p\}$ にとると、g が $g = g_1 + g_2 + ... + g_N$ $(g_i \in B)$ と書ける確率は $\frac{1}{(ng)!}$ であり、n, g を小さな定数としたとき、上の仮定が成り立っている。

群演算が加法群で書かれている Index calculus は下記のような手順で行われている。$i \cdot a + j \cdot b \in < B >$ となる i, j を充分たくさん見つける、その後、まず Large prime のパートの消去を行い、その後にサイズが約 $|B_0| \times |B_0|$ である $|G|$ を法とする線形代数を解く事により、DLP を解く。

なお、筆者 [8] と Gaudry ら [5] は独立に、B が上の仮定を満たす形の Index Calculus に関して、2 個の Large prime を消去する方式が、離散対数問題全体の計算の削減につながることを示している。

4 一般の有限体上定義された曲線の Jacobian 群の Index Calculus

ここでは、Gaudry [4] によって得られら一般の有限体上定義された曲線の Jacobian 群の Index Calculus について手短に解説する。$C/\mathbb{F}_q$ を有限体上定義された種数 g の曲線、 群 G をその Jacobian 群とする。このとき、Gaudry [4] は $B = B_0 = C(\mathbb{F}_q) = \{P - \infty | P \in C(\mathbb{F}_q)\}$, $N = g$ とれば、 群 G は前節の仮定をみたし、Index Calculus が上手く動く事を示した。この方式で DLP を解くのにかかるコストは $O(q^{2+\epsilon})$ である。

この解き方では、計算量のドミナントな部分は実は線形代数部分であるため、$B = B_0$ を上で取った集合の部分集合とし、計算量の rebalance をする改良方式が、Gaudry と Harley によってなされた。この計算量は $O(q^{(4g-2)/(2g+1)+\epsilon})$ である。

次に、Large Prime の消去が計算量的に意味のあることが、Thériault[11], 筆者 [8], Gaudry,Thomé, Diem[6] によってなされた。まず、Thériault によって、$i \cdot a + j \cdot b = g_1 + .. + g_N$, $\{g_i\}$ の内 1 つのみが Large Prime である i, j のみを集め、その後 Large prime の項を消去し、線形代数を解くことによって、DLP 計算の計算量が下がることがわかった。また、筆者と Gaudry らによって独立に、$i \cdot a + j \cdot b = g_1 + .. + g_N$, $\{g_i\}$ の内 2 つのみが Large Prime である i, j のみを集め, その後 Large prime の項を消去し、線形代数を解く方式によって計算量が下がることが示された。

5 拡大体上定義された曲線の Jacobian 群の Index Calculus

ここでは、$C/\mathbb{F}_{q^n}$ を拡大体上定義された種数 g の曲線としたとき、その Jacobian 群 $G = Jac_c(\mathbb{F}_{q^n})$ の DLP を解くアルゴリズムについて述べる。

Gaudry [6] は, G が楕円曲線の作る群（つまり、種数 $g = 1$ の曲線の Jacobian 群の場合）のとき、以下に述べる Semaev 公式を使って、上手く Index Calculus を実現する方式を発見した。$G = E(\mathbb{F}_{q^n})$ を拡大体上定義された楕円曲線 $E/\mathbb{F}_{p^n}$ のなす群とする。まず、Large prime と Factor base の合併集合 B は次のようにとる：$B = \{(x, y) \in E(\mathbb{F}_{q^n}) | x \in \mathbb{F}_q\}$ Semaev's formula は次のような公式である。

Lemma 1 (Semaev formula[10]). $(x, y) \in E(\overline{\mathbb{F}}_{q^n})$ とせよ。$x_1, ..., x_n \in \overline{\mathbb{F}}_{q^n}$ に対して、$\phi(X, X_1, .., X_n) \in \mathbb{F}_{q^n}[X, X_1, ..., X_n]$ で以下の性質を満たす多項式が存在する。
1. $\deg \phi = 2^{n-1}$,
2. $\phi(x, x_1, ...x_n) = 0$ となる必要充分条件は $(x, y) + (x_1, y_1) + ... + (x_n, y_n) = 0$ を満たす $(x_i, y_i) \in E(\overline{\mathbb{F}}_{q^n})$ が存在することである。

$(x, y) \in G$ とせよ。$(x_i, y_i) \in B (i = 1, .., n)$ で、$(x, y) + (x_1, y_1) + .. + (x_n, y_n) = 0$ を満たすものが存在する必要充分条件は上に述べた Semaev formula より方程式 $\phi(x, X_1, .., X_n) = 0$ が解 $(X_1, ..., X_n) = (x_1, ..., x_n) \in \mathbb{A}^n(\mathbb{F}_q)$ を持つことである。なお、$|G| \sim p^n$, $|B| \sim p$ より、一般には (x, y) がこのように n 個の B の元に分解する確率は $= 1/n!(= O(1))$ である。特殊な楕円曲線の場合には成り立たないが、x 座標の平行移動によって成り立つ曲線が得られる。

$x \in \mathbb{F}_{q^n}$ はある楕円曲線の点の x 座標であることに注意したい。$[\alpha_1, .., \alpha_n]$ を $\mathbb{F}_{q^n}/\mathbb{F}_q$ の基底とし、以下これを止めて議論する。$\phi(x, X_1, .., X_n) \in \mathbb{F}_{q^n}[X_1, ..., X_n]$ をこの基底で分解して考えると、
$\phi(x, X_1, .., X_n) = \sum_{i=1}^{n} \alpha_i \phi_i(X_1, ..., X_n)$, $\phi_{x,i}(X_1, ..., X_n) \in \mathbb{F}_q[X_1, ..., X_n]$ という、素体上の多項式の線形和で書くことができる。

$\phi(x, X_1, .., X_n) = 0$ が $(X_1, , , X_n) = (x_1, ..., x_n) \in \mathbb{A}^n(\mathbb{F}_q)$ という形の解を持つという条件は、$\phi_{x,i}(X_1, ..., X_n) = 0, /\mathbb{F}_q \qquad (i = 1, ..., n)$ という素体上の n 変数 n 本の方程式からなる方程式系が解を持つことと同値である。

よって、解 $\{x_i\}$ を得る為には、次数 2^{n-1}, n 変数,n 本の方程式を持つ $\mathbb{F}_q$ 上定義された方程式系を解くことが必要となる。n が小さい定数のとき, このコストは $O(1)$ とみなすことができ（通常の $\mathbb{F}_p^*$ の Index Calculus の因数分解パートにあたる部分)、 $q \to \infty$ のとき DLP を解く計算量は $O(q^{(2ng-2)/ng+\epsilon})$ であると見積もられる。

次に、筆者 [9] によってなされた、群 G を超楕円曲線とした場合の DLP 解法アルゴリズムの拡張アルゴリズムについて解説する。このアルゴリズムは標数が 2 で無い体上定義された超楕円曲線に関して定理を述べているが、実際には一般の曲線の Jacobian 群に関しても同様の方式が得られる。ただし、一般の場合は得られる方程式系がかなり複雑になる為、簡単の為に超楕円曲線に限って解説する。

$C : y^2 = x^{2g+1} + ... + a_0/\mathbb{F}_{q^n}$ を（奇数次式で定義される）種数 g の超楕円曲線、∞ を唯一の無限遠点とする。$G = Jac_C(\mathbb{F}_{q^n})$ とし、その DLP を解きたい。まず、Large prime と Factor base の合併集合 B は次のようにとる：$B = \{(x, y) - \infty | (x, y) \in C(\mathbb{F}_{q^n}), x \in \mathbb{F}_q\}$. ここで簡単の為に、$-\infty$ の項を取り除き $B = \{(x, y) | (x, y) \in C(\mathbb{F}_{q^n}), x \in \mathbb{F}_q\}$ と書くことにする。曲線が楕円曲線の場合の B の定義は Gaudry のそれと同じであり、この定義は Gaudry の方式の拡張と

なっていることに注意されたい。ここでの新方式のアイデアは Semaev's formula の替わりに関数体の元を使うことであり、これによって Index calculus が上手く動く。

$$D_0 = Q_1 + Q_2 + ... + Q_g - (g)\infty \in Jac_C(\mathbb{F}_{p^n})$$

を一つの divisor(Jacobian 群の元) とする。以下、D_0 は止めて議論するものとする。また、D_0 は munford 表現で $D_0 = (\phi_1(x), \phi_2(x))$ と書かれているものとする。

Definition 3 D_0 が以下の形で書かれるとき *decomposed* と呼ぶ：

$$D_0 + P_1 + P_2 + ... + P_{ng} - (ng)\infty \sim 0 \quad \exists P_i \in B.$$

また、$\{P_i\}$ を D_0 の *decomposed factor* と呼ぶ。

$|G| \sim q^{ng}$, $|B| \sim q$ より、D_0 が decomposed である確率は一般に $1/(ng)!$ である。

以下では、種数 $g = 3$, 拡大次数 $n = 2$ の場合について詳しく D_0 が decomposed であるかどうかの判定アルゴリズムと（decomposed である場合）decomposed factor の計算アルゴリズムを解説する。

種数が 3 より、曲線の方程式は

$$C : y^2 = f(x)/\mathbb{F}_{q^2}, \quad f(x) = x^7 + ... + a_0$$

で書かれている。止めた $D_0 \in Jac(C/\mathbb{F}_{q^2})$ に対して、

$D_0 = Q_1 + Q_2 + Q_3 - 3\infty$ を満たす曲線上の 3 点 $Q_1, Q_2, Q_3 \in C(\bar{\mathbb{F}}_q)$ が存在し、Mumford 表現 $D_0 = (\phi_1(x), \phi_2(x))$ では 性質 $\phi_1, \phi_2 \in \mathbb{F}_{q^2}[x], \phi_1$monic, $3 \geq \deg \phi_1 > \phi_2$, $\phi_2^2 - f(x) \equiv 0 \bmod \phi_1$ が成り立つことを注意しておく。

Definition 4 *D:divisor* に対して $L(D) := \{h \in C(\bar{\mathbb{F}}_{q^2}) | (h) + D \geq 0\}$ と置く。

$L(D)$ は $\mathbb{F}_{q^n}$-ベクトル空間である。

Lemma 2 (Riemann Roch). $\deg D \geq 2g-1$ のとき $\mathbf{dim} L(D) = \deg D - g + 1$.

$D = 6\infty - D_0 = 9\infty - (Q_1 + Q_2 + Q_3)$ と置く。すると、$\{\phi_1(x), \phi_1(x)x, (y - \phi_2(x)), (y - \phi_2(x))x\}$ は ベクトル空間 $L(D)$ の基底である。仮に、D_0 が decomposed であったとすると, 点 $\{P_i\}$ は式

$$D_0 + P_1 + ... + P_6 - 6\infty = Q_1 + ... + Q_3 + P_1 + ... + P_6 - 9\infty \sim 0$$

を満たすので、ある $L(D)$ 内の関数体の元のゼロ点になる。

また、$h \in L(D)$ に対して $\mathrm{ord}_\infty h = 9$ とすると、h は $(y - \phi_2(x))x$ の項を必ず持つことが判る。

$$h(x, y) := (A + Bx)\phi_1(x) + (C + 1)(y - \phi_2(x)).$$

と置く。ここで、 A, B, C は $\mathbb{F}_{q^2}$ を動くパラメータである。

以下 $h(x, y) = 0$ と曲線 C の交点を求める。$h(x, y) = 0$ は式

$$y = \frac{(A + Bx)\phi_1(x) - (C + 1)\phi_2(x)}{C + x}$$

と変形でき、これと C の定義式 $y^2 = x^{2g+1} + ... + a_0$ を連立し y を消去することにより、

$$p(x) := (x+C)^2(x^7 + ...) - ((A+Bx)\phi_1(x) - (C+1)\phi_2(x))^2$$

を得る。ここで、$p(x) = 0$ の 9 個の解は $Q_1, ..., Q_3, P_1, ..., P_6$ の x 座標たちと一致する。

$$g(x) := p(x)/\phi_1(x) = x^6 + Co_5 x^5 + ... + Co_0$$

と置く。すると、下記の Lemma が成り立つ。

Lemma 3. *1)*$g(x) = 0$ の *6* つの解は $P_1, ..., P_6$ の x 座標と一致する。
2) パラメータ A, B, C を文字と解釈したとき、$Co_0, .., Co_5 \in \mathbb{F}_{q^2}[A, B, C]$, $\deg Co_i = 2$ が成り立つ。
3)D_0 が *decomposed* とする。このとき、$a, b, c \in \mathbb{F}_{q^2}$ で $Co_i(a, b, c) \in \mathbb{F}_q$ を満たすものが存在する。

以下、$Co_i(a, b, c) \in \mathbb{F}_q\ (i = 0, ..., 5)$ という条件式を詳しく調べよう。

$[1, t]$ を 体の拡大 $\mathbb{F}_{q^2}/\mathbb{F}_q$ の基底とし、以下の議論で止めておく。$A_0, A_1, B_0, B_1, C_0, C_1$ を $\mathbb{F}_q$ を動く新しいパラメータとし、
$A = A_O + A_1 t$, $B = B_O + B_1 t$, $C = C_O + C_1 t$
と置く。すると、Co_i は $\mathbb{F}_{q^2}[A_0, A_1, B_0, B_1, C_0, C_1]$ の多項式とみなすことができる。

Co_i を基底 $[1, t]$ で下記のように分解する。

$$Co_i = Co_{i,0} + Co_{i,1} t, \quad (i = 0, 1, ..., 5, j = 0, 1)$$

$$(Co_{i,j} \in \mathbb{F}_q[A_0, A_1, B_0, B_1, C_0, C_1]).$$

すると、 $\deg Co_{i,0} = \deg Co_{i,1} = 2$ であり、$Co_i \in \mathbb{F}_q, i = 0, 1, .., 5$ という条件は $Co_{i,1} = 0$ for $i = 0, 1, .., 5$ という条件と同値となる。よって

$$\{Co_{i,1} = 0/\mathbb{F}_q | i = 0, 1, .., 5\}$$

という 2 次, 6 変数, 6 方程式 から成る $\mathbb{F}_q$ 上の方程式系をが、$\mathbb{F}_q$ 内で解を持つことが、この必要十分条件となる。

$\boldsymbol{v} = (a_0, a_1, b_0, b_1, c_0, c_1) \in \mathbb{A}^6(\mathbb{F}_q)$ を上の方程式系の解とせよ。$c_i := Co_{i,0}(\boldsymbol{v})$, $g(x) = x^6 + c_5 x^5 + ... + c_0$ と置くと以下がわかる。

Lemma 4. $x^6 + c_5 x^5 + ... + c_0$ が $\mathbb{F}_q[x]$ 内で完全分解することは、$x(P_1), ..., x(P_6) \in \mathbb{F}_q$ である必要十分条件である。

実際の計算では、因数分解のパートは簡単であるので、方程式系を解く部分の計算量がドミナントであることに注意されたい。

一般の超楕円曲線に関しては以下の結果が成り立つ：

Theorem 1. $C/\mathbb{F}_{q^n}$ を拡大体上定義された種数 g の超楕円曲線, $D_0 \in Jac_c(\mathbb{F}_{q^n})$ をその *divisor(Jacobian* 群の元*)* とする。$V_1, V_2, ..., V_{(n^2-n)g}$ を変数とする。このとき、2 次の多項式 $C_{i,j} \in \mathbb{F}_q[V_1, V_2, ..., V_{(n^2-n)g}] \quad (0 \le i \le ng-1,\ 0 \le j \le n-1)$ で以下を満たすものが存在する。

D_0 が *decomposed* である必要十分条件は下記 *1) 2)* が成り立つことである：
1) 方程式系 $S = \{C_{i,j} = 0 \,|\, 0 \le i \le ng-1,\ 1 \le j \le n-1\}$ がある解 $\boldsymbol{v} = (v_1, .., v_{(n^2-n)g}) \in \mathbb{A}^{(n^2-n)g}(\mathbb{F}_q)$ をもつ。
2) $c_i = C_{i,0}(v_1, .., v_{(n^2-n)g})$ $(0 \le i \le ng-1)$ と置いたとき、$G(x) = x^{ng} + c_{ng-1}x^{ng-1} + ... + c_0 \in \mathbb{F}_q[x]$ は $\mathbb{F}_q[x]$ 内で完全分解する。.
また、 D_0 が *decomposed* の時、*decomposed factor* の x 座標は $G(x) = 0$ の解と一致する。

この定理より、D_0 の decomposition の判定と decomposed factor の計算は、本質的に 2 次, $(n^2-n)g$ 変数, $(n^2-n)g$ 方程式の方程式系を解くことに帰着する。n, g が小さな定数のとき、この計算量は $O(1)$ と解釈できるので、Index Calculus の仮定（**Assumption 1**）が成り立ち、Index Calculus が上手く動く。また、2 個の Large prime の消去を使った方式での DLP を解くコストは $O(q^{(2ng-2)/ng+\epsilon})$ と見積もられる。ただし、計算機代数で方程式系を解くことの限界から考えて、1) $(g,n) = (1,3)$, 2) $(g,n) = (2,2)$, 3) $(g,n) = (3,2)$ という 3 つのケースでは実際に、与えられた D_0 の decomposed factor の計算が可能であったが、それ以外のケースでは、方程式系が複雑になるため現状では解くことができない。以下では、$(g,n) = (3,2)$ の実例を示す。

$q = 1073741789$(prime number),
$\mathbb{F}_{q^2} := \mathbb{F}_q[t]/(t^2 + 746495860 * t + 206240189)$,

$$C/\mathbb{F}_{q^2} : y^2 = x^7 + (111912375 * t + 1046743132) * x + 6 * t + 9$$

and

$$D_0 := (x^2 + 1073741787 * t * x + 327245929 * t + 867501600,$$

$$(473621736 * t + 256126568) * x + 145989647 * t + 687383736) \in Jac(C)$$

(Mumford 表現) とする。
ここでは magma のプログラムによって、$nD_0 : n = 1, 2, ..3000$ が decomposed であるか否かを調べ、そのうち 6 個が decomposed であることが判った。 また、それらの decomposed factor は下記のとおりである。

$$414D_0 \sim (1001437837, 752632260*t+700158497)+(747112084, 656073918*t+400137619)$$
$$+(620249588, 127943213 * t + 635474623) + (614180498, 206297635 * t + 445250468)$$
$$+(515769009, 607297126*t+554290493)+(488549466, 627952783*t+854182612)-6\infty$$
$$657D_0 \sim (939617127, 695261735*t+239531611)+(933351280, 935312661*t+961494096)$$
$$+(799612924, 341923983 * t + 677495100) + (294787599, 279723229 * t + 760003067)$$
$$+(273118782053704103*t+577497766)+(153381525, 983211238*t+517037777)-6\infty$$
$$921D_0 \sim (1034634787, 400751409*t+829801342)+(763888873, 757155774*t+829936954)$$
$$+(619620874, 800641683 * t + 200272230) + (603032615, 115219564 * t + 655011145)$$
$$+(436423191, 285214454*t+450812747)+(125198811, 884750621*t+123305741)-6\infty$$

$$1026D_0 \sim (1024020017, 267457905*t+41452942)+(794174628, 615676821*t+723336407)$$
$$+(738567269, 433647609*t+128304659)+(629287731, 465842490*t+789390318)$$
$$+(435082408, 878213106*t+603353206)+(79621979, 479459622*t+672937516)-6\infty$$
$$1121D_0 \sim (764081031, 812350603*t+347878564)+(673426715, 687737442*t+381588704)$$
$$+(6102522082007139*t+99219637)+(467560104, 619342780*t+228756808)$$
$$+(179787786, 333322906*t+75482151)+(59221667, 860686653*t+625301206)-6\infty$$
$$2289D_0 \sim (729358563, 482925408*t+170057124)+(529840657, 42328987*t+857983002)$$
$$+(514618236, 436901100*t+416530686)+(350106356, 183495333*t+950710579)$$
$$+(175898979, 411808870*t+427518366)+(96240558, 703780413*t+461022225)-6\infty$$

6 magma でのコーディング

ここでは、magma でのプログラミングでの工夫について幾つかのべる。

6.1 Eltseq

ここでの計算では、素体 $\mathbb{F}_p$, 拡大体 $\mathbb{F}_q = \mathbb{F}_p[t]/irr(t)$ に対して、$f(t) \in \mathbb{F}_q$ を $bF_p[T]$ の元とみなす計算やその一般化としての $f(t, X, Y, ...) \in \mathbb{F}_q[X, Y, ...]$ を $bF_p[T, X, Y, ...]$ とみなす計算が必要となる。これに利用できそうな関数として、下記にあげた Eltseq 関数がある。

```
p:=10007;
Fp:=GF(p); print Fp;
Fq<t>:=GF(p,5); print Fq;
R<T>:=PolynomialRing(Fp);
irr<T>:= MinimalPolynomial(t); print irr;
f:=2*t^7; print f;
lst:=Eltseq(f); print lst;

Loading "C:\wk\kyudai2.txt"
Finite field of size 10007
Finite field of size 10007^5
T^5 + 439*T^4 + 3439*T^3 + 4424*T^2 + 3469*T + 8668
5147*t^4 + 3708*t^3 + 709*t^2 + 5292*t + 2618
[ 2618, 5292, 709, 3708, 5147 ]
```

この Eltseq を使って、下記にある 2 つの自作関数を作った。

```
// A=a+bt+ct^2 in Fq ==> a+bT+cT^2 in R(....,T,.....)
MyEltseq:=function(A,T)
R:=Parent(T);
Lst:=Eltseq(A);
ans:=R!0;
```

```
for i in [1..#Lst] do
   ans:=ans+(R!Lst[i])*T^(i-1);
   end for;
return ans;
end function;
// MyEltseq2: Fq[x]=R0-->Fp[z,X,A0,....]=R1
// t->z, x->X
MyEltseq2:=function(A,x,X,t,z)
R1:=Parent(X);
Fq:=Parent(t);
Lst:=Coefficients(A);
// print Lst;
aaa:=R1!0;
for i in [1..#Lst] do
   bbb:=Fq!Lst[i];
//   print "bbb=",bbb,Parent(bbb);
   ccc:=MyEltseq(bbb,z);
   aaa:=aaa+ccc*X^(i-1);
   end for;
return aaa;
end function;
```

6.2 環の定義

$q = p^2$, $R_2 := \mathbb{F}_q[X, A0, A1, B0, B1, C0, C1]$ という環を定義し、この元 $f(X, A0, A1, B0, B1, C0, C1) + t * g(X, A0, A1, B0, B1, C0, C1)$ の $f(X, A0, ..., C1), g(X, A0, ..., C1) \in \mathbb{F}_p[X, A0, A1, B0, B1, C0, C1]$ を取り出したい。このために, テクニカルな方式を取る。まず、$R1 = \mathbb{F}_p[z, X, A0, ..., C1]$ とし、$irr2 \in R1$ を $\mathbb{F}_q/\mathbb{F}_p$ の最小多項式 $irr(T)$ に $T = z$ を代入した式（上の MyEltseq 関数で導出) とし、$R2 := quo < R1, irr2 >$ とする。$R1$ の元 $a = f(X, A0, .., C1) + z * g(X, A0, ..C1)$ を、$R2!a$ することにより、拡大体上の多項式と考え、また、a から $f(X, A0, .., C1), g(X, A0, ..C1)$ を取り出すときは一旦 $R1!a$ と $R1$ 上の元とみなし、その z での係数を取り出す。

```
// definition of base field
 p:=MyBeforePrime(2^30);
//p:=10007;
Fp:=GF(p);
Fq<t>:=GF(p,2);
R<T>:=PolynomialRing(Fp);
irr<T>:= MinimalPolynomial(t);
print "p=",p,"irr=",irr;
R0<x>:=PolynomialRing(Fq);
R1<z,X,A0,A1,B0,B1,C0,C1>:=PolynomialRing(Fp,8);
irr2:=MyEltseq(irr,z);
R2:=quo<R1 | irr2>;
```

```
R3<AA0,AA1,BB0,BB1,CC0,CC1>:=PolynomialRing(Fp,6);
phi1:=hom<R1->R3 | 0,0,AA0,AA1,BB0,BB1,CC0,CC1>;
```

$R1 = \mathbb{F}_p[z, X, A0, ..., C1]$
$R2 = \mathbb{F}_{p^2}[X, A0, ..., C1]$ ただし、$[1, z]$ は BASE

6.3 $g(x)$ の計算

z は多項式環 `R1<z,X,A0,A1,B0,B1,C0,C1>:=PolynomialRing(Fp,8);` の元であり、関数 MyEltseq2 で、超楕円曲線の定義式 $y^2 = f(x)$ の $f(x) \in \mathbb{F}_q[x]$, $pt1$ の Munford 表現に出てくる多項式 $m1(x), m2(x) \in \mathbb{F}_q[x]$ は環 $R1$ の元 `FF0,mm1,mm2` に移される。この後、$g(x)$ が計算される。

```
pt1:=n*pt0;
m1:=pt1[1];
m2:=pt1[2];
// map to R1
FF0:=MyEltseq2(F0,x,X,t,z);
mm1:=MyEltseq2(m1,x,X,t,z);
mm2:=MyEltseq2(m2,x,X,t,z);

A:=A0+A1*z;
B:=B0+B1*z;
C:=C0+C1*z;

FFF:=-(mm1*(B*X+C)+(X+A)*mm2)^2+(X+A)^2*FF0;
FFF:=R1!(R2!FFF);
G,rr:=MyPolynomialDiv(FFF,mm1,X);
rr:=R1!(R2!rr);
G:=R1!(R2!G);

Lst1:=Coefficients(G,X);
C0:=Lst1[1];
中略
C5:=Lst1[6];

Lst1:=Coefficients(C0,z);
C00:=Lst1[1]; C01:=Lst1[2];
中略
Lst1:=Coefficients(C5,z);
C50:=Lst1[1];  C51:=Lst1[2];

CC00:=phi1(C00);
中略
CC51:=phi1(C51);
```

6.4 方程式系の解の計算

方程式系を作る方程式は上で、C01,C11,...,C51 ∈ R1<z,X,A0,A1,B0,B1,C0,C1>:=PolynomialRing(Fp,8); として定義された。方程式は変数 X の入っていない環上で定義する必要があるので、R3<AA0,AA1,BB0,BB1,CC0,CC1>:=PolynomialRing(Fp,6); と **R1** から **R3** への写像phi1:=hom<R1->R3 | 0,0,AA0,AA1,BB0,BB1,CC0,CC1>; を定義し、**phi1** で C01,C11,...,C51 を **R3** 上にうつし、そこでイデアルをつくり、そのグレブナ基底を計算し、**Variety** 関数でその零点を計算した。

```
I:=ideal<R3|CC01,CC11,CC21,CC31,CC41,CC51>;
Groebner(I);
V:=Variety(I);
```

実行結果： $n = 414$ の場合の Groebner(I); 後の Ideal I

```
Ideal of Polynomial ring of rank 6 over GF(1073741789)
Lexicographical Order
Variables: AA0, AA1, BB0, BB1, CC0, CC1
Basis:
[
131871035*AA0^2 + 78580125*AA0*AA1 + 892000224*AA0*CC0
+ 14165663*AA0*CC1 +316961574*AA1^2 + 14165663*AA1*CC0
+ 981254322*AA1*CC1 + 499583599*CC0^2 + 274744509*CC0*CC1
 + 532262852*CC1^2,
1021081575*AA0^2 + 950616333*AA0*AA1 + 892000224*AA0*BB0
+ 14165663*AA0*BB1+ 258846761*AA0*CC0 + 552415644*AA0*CC1
+ 263742070*AA0 +319953178*AA1^2 + 14165663*AA1*BB0
+ 981254322*AA1*BB1 +552415644*AA1*CC0 + 315930839*AA1*CC1
+ 78580125*AA1 + 999167198*BB0*CC0+ 274744509*BB0*CC1
+ 274744509*BB1*CC0 + 1064525704*BB1*CC1 +1017039927*CC0^2
+ 877085201*CC0*CC1 + 892000224*CC0 + 693136302*CC1^2
+14165663*CC1,
729524633*AA0^2 + 1054446311*AA0*AA1 + 258846761*AA0*BB0
+ 552415644*AA0*BB1+ 959696564*AA0*CC0 + 446479665*AA0*CC1
+ 968421361*AA0 +994360792*AA1^2 + 552415644*AA1*BB0 +
 315930839*AA1*BB1 +446479665*AA1*CC0 + 904553148*AA1*CC1
+ 950616333*AA1 + 499583599*BB0^2+ 274744509*BB0*BB1
+960338065*BB0*CC0 + 877085201*BB0*CC1 +892000224*BB0
+ 532262852*BB1^2 + 877085201*BB1*CC0 + 312530815*BB1*CC1
+ 14165663*BB1 + 588256351*CC0^2 + 738573150*CC0*CC1
+ 258846761*CC0 +762056845*CC1^2 + 552415644*CC1 + 131871035,
588256351*AA0^2 + 738573150*AA0*AA1 + 959696564*AA0*BB0
+ 446479665*AA0*BB1+ 385307477*AA0 + 762056845*AA1^2
+ 446479665*AA1*BB0 +904553148*AA1*BB1 + 1054446311*AA1
+ 1017039927*BB0^2 +877085201*BB0*BB1 + 102770913*BB0*CC0
+ 738573150*BB0*CC1 +258846761*BB0 + 693136302*BB1^2
```

```
+ 738573150*BB1*CC0 + 450371901*BB1*CC1+ 552415644*BB1
+ 1073741787*CC0*CC1 + 959696564*CC0 + 746495860*CC1^2
+446479665*CC1 + 1021081575,
2*AA0*AA1 + 102770913*AA0 + 327245929*AA1^2 + 738573150*AA1
+ 588256351*BB0^2 + 738573150*BB0*BB1 + 1073741787*BB0*CC1
+ 959696564*BB0+ 762056845*BB1^2 + 1073741787*BB1*CC0
+ 419249931*BB1*CC1 +446479665*BB1 + 729524633,
2*AA1 + 1073741787*BB0*BB1 + 746495860*BB1^2 + 588256351
]
```

実行結果： $n = 414$ の場合の V： Variety(I) 方程式系（6 変数 A0,...,C1）の体 $\mathbb{F}_p$ 上の解集合

```
[ <33112584, 648474771, 797125543, 564708046, 256786628, 182945850>,
 <107194206,657038295, 995044947, 634535146, 851891696, 694745215>,
<500233772, 438915565,137992372, 492712108, 926322346, 799679486>,
<704897670, 100937984, 929939619,521988388, 17748540, 836371569> ]
```

謝辞: このようにすばらしい研究集会を企画し、また講演の機会を与えてくださいました、オーガナイザーの電通大の木田雅成 先生、山形大の原田昌晃 先生、九州大学の金子昌信 数理学府長 並びに、会場運営に御尽力していただいた横山俊一さんをはじめとする九州大の院生の皆様に感謝いたします。

References

1. C. Diem, An Index Calculus Algorithm for Plane Curves of Small Degree, Algorithmic Number Theory - ANTS VII, LNCS 4076, 2006
2. C. Diem, On the discrete logarithm problem in class groups, preprint, 2009, (`http://www.math.uni-leipzig.de/~diem/preprints/small-genus.pdf`)
3. A. Enge, P. Gaudry, A general framework for subexponential discrete logarithm algorithms, *Acta Arith.,* **102**, no. 1, 2002, pp. 83–103.
4. P.Gaudry, An algorithm for solving the discrete log problem on hyperelliptic curves, *Eurocrypt 2000*, LNCS 1807, Springer-Verlag, 2000, pp. 19–34.
5. P. Gaudry, E. Thomé, Thériault, C. Diem, A double large prime variation for small genus hyperelliptic decomposed attack, Math. Comp. 76, 2007, pp.475–492. (Preprint version is available on `http://eprint.iacr.org/2004/153/`)
6. P. Gaudry, Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem, Journal of Symbolic Computation, Vol.44, **12**, 2009, pp. 1690–1702. (Preprint version is available on `http://eprint.iacr.org/2004/073`)
7. R. Granger, F. Vercauteren, On the Discrete Logarithm Problem on Algebraic Tori, Advances in Cryptology, CRYPTO 2005, LNCS 3621, Springer-Verlag, 2005, pp. 66-85.
8. K. Nagao, Index calculus for Jacobian of hyperelliptic curve of small genus using two large primes, Japan Journal of Industrial and Applied Mathematics, **24**, no.3, 2007. (Preprint version entitled by "Improvement of Thériault Algorithm of decomposed attack for Jacobian of Hyperelliptic Curves of Small Genus" is available on `http://eprint.iacr.org/2004/161`)

9. K. Nagao, Decomposition Attack for the Jacobian of a Hyperelliptic Curve over an Extension Field, 9th International Symposium,ANTS-IX., Nancy, France, July 2010, Proceedings LNCS 6197,Springer, pp.285–300, 2010.

10. I. Semaev. Summation polynomials and the discrete logarithm problem on elliptic curves. Preprint, 2004.
11. N. Thériault, Index calculus for hyperelliptic curves of small genus, ASIACRYPT2003, LNCS 2894, Springer-Verlag, 2003, pp. 75–92.

7 Appendix

ここでは、参考の為に $g = 3, n = 2$ の場合の decomposition 計算のプログラムを提示する。上手くかけていない箇所が目に付くが、研究当時のままのプログラムである。

```
// Let f be a mul var polynomial f in R<..x,..>,
// and g=g0+g1 x+...g_n-1 x^n-1+x^n be a monic polynomial of x,
// however considered as multivaiable polynomial.
// compute quotientand remainder of f,g assciated by x.

MyPolynomialDiv:=function(f,g,x)
// print f,g;
R:=Parent(f);
r:=f; q:=R!0;
N1:=#Coefficients(g,x);
LST:=Coefficients(r,x);
N2:=#LST;
while N2 ge N1 do
   q:=q+LST[N2]*x^(N2-N1);
   r:=r-g*LST[N2]*x^(N2-N1);
   LST:=Coefficients(r,x);
   N2:=#LST;
end while;
return q,r;
end function;

// Before prime
MyBeforePrime:=function(p)
pp:=p;
if pp le 3 then
   return 2;
   end if;
pp:=pp-1;
while not (IsPrime(pp)) do
  pp:=pp-1;
  end while;
return pp;
end function;
```

```
// A=a+bt+ct^2 in Fq ==> a+bT+cT^2 in R(....,T,.....)
MyEltseq:=function(A,T)
R:=Parent(T);
Lst:=Eltseq(A);
ans:=R!0;
for i in [1..#Lst] do
   ans:=ans+(R!Lst[i])*T^(i-1);
   end for;
return ans;
end function;

// Let [c0,c1,c2] in Fp, solve T^3+c2T^2+c1T+c0=0 in Fp
// must assume monic so degree 3 coeff is omitted!
MySolveEq:=function(Lst,Fp,Fq)
if #Lst eq 0 then
   return [];
   end if;
//print "Lst[1]=",Lst[1];
// K:=Parent(Lst[1]);
R<T>:=PolynomialRing(Fp,1);
//print "K=",K;
FF:=R!0+T^(#Lst);
for i in [1..#Lst] do
   FF:=FF+(R!Lst[i])*T^(i-1);
   end for;
//print "FF=",FF,Parent(FF);
Lst2:=Factorisation(FF);
// print Lst2;
Lst3:=[];
for i in [1..#Lst2] do
   aaa:=R!Lst2[i][1];
   bbb:=Coefficients(aaa,T);
   if Degree(aaa,T) eq 1 then
       x:=-bbb[1] / bbb[2] ;
       Append(~Lst3,Fq!x);
       end if;
   end for;
return Lst3;
end function;

//MySolveEqLst
MySolveEqLst:=function(Lst,Fp,Fq)
if #Lst eq 0 then
   return [];
   end if;
Lst3:=[];
for i in [1..#Lst] do
```

```
//print Lst[i];
       aaa:=MySolveEq(Lst[i][1],Fp,Fq);
//print aaa, Lst[i][2];
//bbb:=[aaa,  Lst[i][2] ];
//print bbb;
       Append(~Lst3,[aaa,Lst[i][2]]);
   end for;
return Lst3;
end function;



// t in Fq. [c0,c1,c2,c3...] be a list of Fp returns
// c0+c1*t+c2*t^2+... in Fq
MyFqnize:=function(Lst,t)
Fq:=Parent(t);
ans:=Fq!0;
for i in [1..#Lst] do
   ans:=ans+(Fq!Lst[i])*t^(i-1);
   end for;
return ans;
end function;

//List Evaluation
MyLstEvaluation:=function(Lst,v)
if #Lst eq 0 then
   return [];
   end if;
Lst3:=[];
for i in [1..#Lst] do
       aaa:=Evaluate(Lst[i],v);
//       print Parent(aaa);
       Append(~Lst3,aaa);
   end for;
return Lst3;
end function;

//List Evaluation
MyFldEvaluation:=function(v,t)
R:=Parent(t);
a0:=R!v[1];
a1:=R!v[2];
b0:=R!v[3];
b1:=R!v[4];
c0:=R!v[5];
c1:=R!v[6];
bbb:=[a0+a1*t,b0+b1*t,c0+c1*t];
return bbb;
end function;
```

```
// From list, choose the # of solution is n
MyChooseSolution:=function(Lst,n)
if #Lst eq 0 then
   return [];
   end if;
Lst3:=[];
for i in [1..#Lst] do
       aaa:=Lst[i][1];
       if #aaa eq n then
          Append(~Lst3,[aaa,Lst[i][2]]);
           end if;
   end for;
return Lst3;
end function;



// My Mkpt (technical)
MyMkpt:=function(C,Afq,Bfq,Cfq,m1,m2,Lst)
if #Lst eq 0 then
   return [];
   end if;
Fq:=Parent(Afq);
Lst3:=[];
for i in [1..#Lst] do
    x1:=Fq!Lst[i];
    y1:=(Evaluate(m1,x1)*(x1*Bfq+Cfq)+Evaluate(m2,x1)*(x1+Afq))/(x1+Afq);
    y1:=-y1;
    Append(~Lst3,C![x1,y1]);
   end for;
return Lst3;
end function;

// Mk pt 2 Jac
// P point of Curve C, J C's JAcobian,
// returns Jacobian of P-infty
// It's curious but infty=C![1,0,0]
MyMkpt2jac:=function(P,C,J)
return J![C!P,C![1,0,0]];
end function;

MyMkpt2jacLst:=function(Lst,C,J)
return [ J![C!P,C![1,0,0]] : P in Lst];
end function;



// MyEltseq2: Fq[x]=R0-->Fp[z,X,A0,....]=R1
// t->z, x->X
```

```
MyEltseq2:=function(A,x,X,t,z)
R1:=Parent(X);
Fq:=Parent(t);
Lst:=Coefficients(A);
// print Lst;
aaa:=R1!0;
for i in [1..#Lst] do
   bbb:=Fq!Lst[i];
//   print "bbb=",bbb,Parent(bbb);
   ccc:=MyEltseq(bbb,z);
   aaa:=aaa+ccc*X^(i-1);
   end for;
return aaa;
end function;

// definition of base field
 p:=MyBeforePrime(2^30);
//p:=10007;
Fp:=GF(p);
Fq<t>:=GF(p,2);
R<T>:=PolynomialRing(Fp);
irr<T>:= MinimalPolynomial(t);
print "p=",p,"irr=",irr;
R0<x>:=PolynomialRing(Fq);
R1<z,X,A0,A1,B0,B1,C0,C1>:=PolynomialRing(Fp,8);
irr2:=MyEltseq(irr,z);
R2:=quo<R1 | irr2>;
R3<AA0,AA1,BB0,BB1,CC0,CC1>:=PolynomialRing(Fp,6);
phi1:=hom<R1->R3 | 0,0,AA0,AA1,BB0,BB1,CC0,CC1>;

Cafq:=-t^6+t;
Cbfq:=6*t+9;
F0:=x^7+(R0!Cafq)*x+(R0!Cbfq);
Cu:=HyperellipticCurve(F0);

print "Cu=",Cu;

J:=Jacobian(Cu);
pt0:=Cu![t,t+3];
pt0:=J![pt0, Cu![1,0,0]];
// pt1:=Cu![t,-t-2];
// pt1:=J![pt1, Cu![1,0,0]];
pt0:=2*pt0;
print "pt0=",pt0;

MAINMAIN:=function(n,p,Fp,Fq,t,R,T,irr,R0,x,R1,z,X,A0,A1,B0,B1,C0,C1,R2,
AA0,AA1,BB0,BB1,CC0,CC1,Cafq,Cbfq,F0,Cu,J,pt0)
   pt1:=n*pt0;
   m1:=pt1[1];
   m2:=pt1[2];
```

```
   // map to R1
   FF0:=MyEltseq2(F0,x,X,t,z);
   mm1:=MyEltseq2(m1,x,X,t,z);
   mm2:=MyEltseq2(m2,x,X,t,z);

   A:=A0+A1*z;
   B:=B0+B1*z;
   C:=C0+C1*z;

   FFF:=-(mm1*(B*X+C)+(X+A)*mm2)^2+(X+A)^2*FF0;
   FFF:=R1!(R2!FFF);
   G,rr:=MyPolynomialDiv(FFF,mm1,X);
   rr:=R1!(R2!rr);
   G:=R1!(R2!G);
// print "rr=",rr;
   Lst1:=Coefficients(G,X);
   C0:=Lst1[1];
   C1:=Lst1[2];
   C2:=Lst1[3];
   C3:=Lst1[4];
   C4:=Lst1[5];
   C5:=Lst1[6];


   Lst1:=Coefficients(C0,z);
   C00:=Lst1[1];
   C01:=Lst1[2];

   Lst1:=Coefficients(C1,z);
   C10:=Lst1[1];
   C11:=Lst1[2];

   Lst1:=Coefficients(C2,z);
   C20:=Lst1[1];
   C21:=Lst1[2];

   Lst1:=Coefficients(C3,z);
   C30:=Lst1[1];
   C31:=Lst1[2];

   Lst1:=Coefficients(C4,z);
   C40:=Lst1[1];
   C41:=Lst1[2];

   Lst1:=Coefficients(C5,z);
   C50:=Lst1[1];
   C51:=Lst1[2];


   CC00:=phi1(C00);
```

```
   CC01:=phi1(C01);
   CC10:=phi1(C10);
   CC11:=phi1(C11);
   CC20:=phi1(C20);
   CC21:=phi1(C21);
   CC30:=phi1(C30);
   CC31:=phi1(C31);
   CC40:=phi1(C40);
   CC41:=phi1(C41);
   CC50:=phi1(C50);
   CC51:=phi1(C51);


   I:=ideal<R3|CC01,CC11,CC21,CC31,CC41,CC51>;
   Groebner(I);
   V:=Variety(I);
i:=[];
// print "n=",n,"V=",V;
Lst2:=[];
Lst3:=[];
Lst4:=[];
Lst5:=[];
Lst6:=[];
Lst7:=[];
//   print [MyLstEvaluation([CC01,CC11,CC21,CC31],v):v in V];
   Lst2:=[ [MyLstEvaluation([CC00,CC10,CC20,CC30,CC40,CC50],v),
MyFldEvaluation(v,t)]:v in V];
//   Lst3:=MySolveEqLst(Lst2,Fp,Fq);
  Lst3:=MySolveEqLst(Lst2,Fp,Fq);
   Lst4:=MyChooseSolution(Lst3,6);
//   print Lst3,Lst4;
   Lst5:=[MyMkpt(Cu,v[2][1],v[2][2],v[2][3],m1,m2,v[1]) : v in Lst4];
   if Lst5 ne [] then
        print "n=",n;
        print Lst5;
        end if;
   Lst6:=[MyMkpt2jacLst(v,Cu,J) : v in Lst5];
   Lst7:=[&+v eq pt1: v in Lst6];
   if Lst7 ne [] then
        print Lst7;
        end if;
return [n];
 end function;

for n in [1..3000] do
MAINMAIN(n,p,Fp,Fq,t,R,T,irr,R0,x,R1,z,X,A0,A1,B0,B1,C0,C1,R2,AA0,AA1,
BB0,BB1,CC0,CC1,Cafq,Cbfq,F0,Cu,J,pt0);
end for;
```

最小固有値 -2 を持つグラフと
ルート系

宗政昭弘
東北大学大学院情報科学研究科

1 MAGMA におけるグラフの取扱い

ここでは簡単のため、向きのない単純グラフのみを扱う。

定義. グラフとは有限集合 V と V の 2 点部分集合いくつかからなる族 E の組 (V, E) のことである。すなわち

$$|V| < \infty, \quad E \subset \{\{x, y\} \mid x, y \in V,\ x \neq y\}.$$

V の要素は「点」とよび、E の要素は「辺」とよぶ。

例. $V = \{1, 2, 3\}$, $E = \{\{1, 2\}, \{2, 3\}\}$. MAGMA では

```
Graph<{1,2,3}|{{1,2},{2,3}}>;
```

で定義できる。または、もっと簡単に

```
PathGraph(3);
```

でもよい。

代表的なグラフの例として

(i). 完全グラフ

```
G:=CompleteGraph(4);
```

(ii). サイクル

```
G:=PolygonGraph(4);
```

などがある。また、グラフ G の補グラフは

```
Complement(G);
```

のようにして作れる。ただし、MAGMA にはグラフを描く機能はありません。

定義. グラフ G の ライングラフ（line graph） とは、G の辺を新たに点と考えて、それらが共有点をもつときに新たに辺で結んだものである。

MAGMA では

```
LineGraph(G);
```

のようにして作れる。

2 グラフと線形代数

MAGMA のマニュアルでは Basic Rings and Linear Algebra → Matrices の節を参照。

定義. グラフ G の 隣接行列（adjacency matrix） とは、G の頂点集合 V で添字付けられている $n \times n$ 行列（$n = |V|$）で、成分が

$$A_{x,y} = \begin{cases} 1 & x \text{ と } y \text{ は辺で結ばれているとき}, \\ 0 & \text{そうでないとき} \end{cases}$$

で定義されるものである。V の並べ方によって行列は一意的に決まらないが、行と列の同時置換を除いて定まる。この行列 A の固有値を単にグラフ G の固有値とよぶ。

隣接行列は MAGMA では

```
A:=AdjacencyMatrix(G);
```

のようにして作れる。例えば

```
> AdjacencyMatrix(CompleteGraph(5));
[0 1 1 1 1]
[1 0 1 1 1]
[1 1 0 1 1]
[1 1 1 0 1]
[1 1 1 1 0]
> AdjacencyMatrix(PathGraph(5));
[0 1 0 0 0]
[1 0 1 0 0]
[0 1 0 1 0]
[0 0 1 0 1]
[0 0 0 1 0]
```

一般に行列の固有値を MAGMA で求めることができるが、少し注意が必要である。

- 固有値は一般には整数とは限らない。

- 固有値の計算は行列成分の属している環を指定すると、その環に属す固有値のみが計算される（多項式の因数分解も、係数環を指定しておくのでそこで分解するのと同じ）。
- 係数環として実数体を指定すれば近似計算される。
- 係数環として分解体を指定しても分解体の実数体への埋め込みまで指定したことにはならないので注意が必要。

```
> G:=PathGraph(3);
> A:=AdjacencyMatrix(G);
> Eigenvalues(A);
{ <0, 1> }
```

これは、固有値 0 が重複度 1 で現れることを意味している。このように、3 次実対称行列である A の固有値が 1 個しか見つからない。Magma では隣接行列は整数成分の行列環の元として作られ、したがってその固有値も整数の範囲でしか計算されないからである。すべての固有値を求めるには、固有方程式の分解体または実数体での近似計算を利用する必要がある。代数体の取り扱いについては Magma のマニュアルでは Global Arithmetic Fields → Quadratic Fields を参照。ここでは最も基本的な 2 次体を作ってみる。まず、s で $\sqrt{2}$ を表す 2 次体を作って、そこで固有値を求めると

```
> K<s>:=QuadraticField(2);
> AA:=ChangeRing(A,K);
> Eigenvalues(AA);
{ <0, 1>, <-s, 1>, <s, 1> }
```

となって確かに $0, \pm\sqrt{2}$ が固有値であることがわかる。ところで、s で $\sqrt{2}$ を表す、と書いたが、本当は $\sqrt{2}$ と $-\sqrt{2}$ のどちらを表すかの区別はない（抽象的な代数拡大を作っただけだから）ので、体 K には本来大小関係は定義されていないはずである。K での大小関係の比較は予想外の答えが出てくる。

```
> eigen:={x[1]:x in Eigenvalues(AA)};
> Minimum(eigen);
0
> s gt 0;
true
> -s gt 0;
true
```

もちろん実数体に埋め込めば大小関係はきちんと処理される。埋め込み方は簡単で、R| とすることで K の元を R の元とみなすことができる。

```
> R:=RealField(10);
> eigen:={R|x[1]:x in Eigenvalues(AA)};
> eigen;
{ 0.0000000000, -1.414213562, 1.414213562 }
> Minimum(eigen);
-1.414213562
```

定義. グラフ G のノルム m 表現とは、G の頂点集合から $\mathbb{R}^n$ への単射 φ で、ある正整数 m が存在して

$$(\varphi(x), \varphi(y)) = \begin{cases} m & x = y \text{ のとき}, \\ 1 & x \text{ と } y \text{ は辺で結ばれているとき}, \\ 0 & \text{それ以外}. \end{cases}$$

G の隣接行列を A とすると、G がノルム m 表現をもつ $\iff$ $A + mI$ が非負値 $\iff$ G の最小固有値が m 以上、となる。これは Magma では次のようにして確かめられる。

```
IsPositiveSemiDefinite(A+m*I);
```

例えば

```
> G:=PathGraph(3);
> A:=AdjacencyMatrix(G);
> I:=MatrixAlgebra(Integers(),3)!1;
> IsPositiveSemiDefinite(A+I);
false
> IsPositiveSemiDefinite(A+2*I);
true
```

3 ルート系

最小固有値がある値以上であるかを問題にするので、最小固有値をはっきり求める必要はない。また、これから主に扱うグラフは最小固有値がちょうど -2 である。ノルム 2 のベクトルからなるユークリッド空間の部分集合で互いの内積が整数となるものは、

- 鏡映による閉包をとる（またはこれと同値な）、
- 整数係数一次結合で表されるノルム 2 のベクトルをすべて付加

という操作で、鏡映による閉包で閉じている「ルート系」という分類されたもののどれかになる。グラフ G の隣接行列を A とするとき、

$$
\begin{aligned}
&G \text{ があるグラフのラインググラフ}\\
&\Longrightarrow G \text{ の最小固有値} \geq -2\\
&\Longleftrightarrow A+2I \text{ は非負値行列,}\\
&\Longrightarrow \text{グラフ } G \text{ はノルム 2 の表現をもつ}\\
&\Longrightarrow \text{グラフ } G \text{ はルート系 } A_n, D_n, E_n \text{ の部分集合で表現される}\\
&\qquad \text{（ただし } G \text{ が連結なら）}
\end{aligned}
$$

となるので、このようなグラフを調べることはルート系を調べることに帰着される。Magma においてルート系は

Lie Theory → Root Systems　または

Lattices and Quadratic Forms → Lattices

で利用できる。

最小固有値 -2 をもつグラフの表現は、その隣接行列 A を用いて、$A+2I$ をグラム行列とするユークリッド空間の部分集合のことである。この部分集合で生成された lattice が何であるか Magma で計算できる。

例. 5 点の完全グラフのラインググラフの補グラフ $\overline{L(K_5)}$ は最小固有値 -2 であり、このグラフをノルム 2 で表現した集合は E_6 root lattice を生成する。

```
> G:=Complement(LineGraph(CompleteGraph(5)));
> A:=AdjacencyMatrix(G);
> I:=Parent(A)!1;
> F:=LLLGram(A+2*I);
> F;
[ 2 -1 -1  1 -1 -1  0  0  0  0]
[-1  2  1 -1  1  1  0  0  0  0]
[-1  1  2  0  0  1  0  0  0  0]
[ 1 -1  0  2 -1  0  0  0  0  0]
[-1  1  0 -1  2  0  0  0  0  0]
[-1  1  1  0  0  2  0  0  0  0]
[ 0  0  0  0  0  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0  0  0]
> F6:=Submatrix(F,1,1,6,6);
> L:=LatticeWithGram(F6);
```

```
> t:=IsIsomorphic(L,Lattice("E",6));
> t;
true
```

逆に、$\overline{L(K_5)}$ をルート系 E_6 の中に作れる。E_6 ルート系 72 点からなるグラフをつくる（内積が 1 のとき辺で結ぶが、E_6 全体では負の内積があるので、これは 1 つのグラフの表現の像ではない）。

```
E6short:=ShortestVectors(Lattice("E",6));
#E6short eq 36;
E6:=&join{{x,-x}:x in E6short};
#E6 eq 72;
E6graph:=Graph<E6|{{x,y}:x,y in E6|(x,y) eq 1}>;
V:=Vertices(E6graph);
```

ここで、ShortestVectors は与えられた lattice の元で最小のノルムを持つものを列挙する関数だが、符号を除いて列挙するので、そのままでは E_6 の場合 36 個の元しか得られない。これらにマイナスの符号をつけて最小のノルムの元 72 個すべて作るには、上のようにする必要がある。

1 つの頂点 a を任意の近傍 N は $|N| = 20$ であり、

$$N = \{b \in E_6 \mid (a,b) = 1\} = \bigcup_{b \in N} \{b, a-b\} \quad (10\text{ 個の和集合})$$

となる。

```
a:=Random(E6);
N:={b:b in E6|(a,b) eq 1};
#N eq 20;
NN:={{b,a-b}:b in N};
#NN eq 10;
```

これら 10 個の組のうち、うまく片方だけ選んで 10 点の部分グラフを作ると $\overline{L(K_5)}$ ができる。そのために、10 点のとり方 2^{10} 通りすべてについて、10 点部分グラフを作り、それらを集めた集合を Gs としよう。

```
C:=CartesianProduct(Setseq(NN));
#C eq 2^10;
Gs:={sub<E6graph|{V|c[i]:i in {1..10}}>:c in C};
```

G は $\overline{L(K_5)}$ と定義してあったから、集合 Gs の中に G と同型なものが存在するかは、以下のようにして調べられる（実際には true が返ってくる）。

```
&or{IsIsomorphic(G,H):H in Gs};
```

さて、最小固有値が -2 であるグラフはルート系で表現されることはすでに述べた。

ノルム 2 の表現が生成する root lattice が A_n $(n \geq 2)$ であるようなグラフ、D_n $(n \geq 4)$ であるようなグラフは無限にあるが、それらの構造的な特徴付けがある（ライングラフとその一般化 [2]）。E_n $(n = 6, 7, 8)$ についてはグラフ自体有限個しかなく、それらの構造的な特徴付けは知られていない。有限個しかないのだからとりあえず分類して、それらの性質を調べてみればよい。極大なものを分類すれば、他はそれらの部分グラフになっている。

4 Maximal exceptional graphs

定義. グラフ G が maximal exceptional graph とは、G がノルム 2 の表現 φ を E_8 ルート系にもち、その像 $\varphi(G)$ が内積の非負性に関して E_8 内で極大であるときをいう。すなわち、

$$\forall a \in E_8,\ a \notin \varphi(G) \text{ に対して}, \exists x \in \varphi(G),\ (a, x) < 0.$$

Theorem (Kitazume–Munemasa [3])**.** G を maximal exceptional graph とし、$X = \varphi(G) \subset E_8$ を G のノルム 2 の表現 φ による像とすると、次のいずれかが成り立つ。

(i). $\exists a_1, \ldots, a_6 \in X$, $(a_i, a_j) = 0$ $(i \neq j)$,

(ii). $\exists a \in X$, $|\{b \in X \mid (a, b) = 1\}| = 28$.

(ii) の意味は

$$N = \{b \in E_8 \mid (a, b) = 1\}$$

とおくと $|N| = 56$ で

$$N = \bigcup_{b \in N} \{b, a - b\} \quad (28 \text{ 個の和集合})$$

となるので、$\{b, a - b\}$ のうち 1 つだけしか $\varphi(G)$ の像には入り得ないが、28 個すべての組に対して必ず 1 つが $\varphi(G)$ の像に入っていることを意味する。

- (i) をみたすグラフの分類は簡単で、Magma を使わず理論的にできる。
- (ii) の分類は、2^{28} 個のグラフを分類しなければならない。経験的な目安として Magma 内で size $2^{23} \sim$ 1 千万個の（例えばベクトル）の集合を作るのに memory 1GB ぐらい必要。2^{28} だと 32GB になる。もちろん、使用するメモリはその集合の要素がどのようなものかに大きく依存するので、これはあくまでも目安である。

上の (ii) に関して、例えば

```
> #Set(VectorSpace(GF(2),20));
1048576
> quit;
Total time: 0.760 seconds, Total memory usage: 37.19MB
```

となるので、`#Set(VectorSpace(GF(2),28));` だと

$$37.19\mathrm{MB} \times 2^8 \sim 10\mathrm{GB}$$

が必要であることがわかる。

E_8 ルート系の鏡映群 $W(E_8)$ における a の安定部分群（すなわち $W(E_7)$）が作用しているので、その作用で 2^{28} 個のグラフ（実際には E_8 ルート系の 28 点部分集合）を軌道分解する。軌道分解の対象となっている集合がメモリ内に作れて、その集合上での置換群の生成元を Magma で作れれば、`Orbits` という関数により軌道分解を実行することができる。メモリ内に作れない場合は工夫してプログラムを書く必要が生じる。これを実行した 2000 年当時は、今よりはるかに使えるメモリが少なかったので、工夫をせざるを得なかった。そのとき用いた方法は以下の通り。

- 直接はできないので、$W(E_7)$ の部分群である $W(D_6)$ を使う。
- $W(D_6)$ は 2^{28} 点上には $2^{16} \times 2^{12}$ という直積への作用をしているので、それぞれを軌道分解する（これならメモリ不足にはならない）。
- $W(D_6)$ による軌道の代表元から、$W(E_7)$ で同値なものを捨てる

このようにして 467 個の代表元が得られる。(i) と合わせて 473 個となり、Cvetković–Lepović–Rowlinson–Simić [1] による、全く別の計算方法による分類結果と一致する。これが分類できる範囲だとメドが立った理由は、軌道の個数を概算で求めることができたからである。

$$\left\lceil \frac{2^{28}}{|W(E_7)|} \right\rceil = 93$$

によって軌道の個数の下界が得られる。Magma では

```
> E7:=RootSystem("E7");
> Ceiling(2^28/#ReflectionGroup(E7));
93
```

通常、下界と大きくことなるということは考えにくく、実際には 467 個であることが判明したというわけである。

なお、この講演の後、この 467 個を求める高速な方法を野崎寛氏から教えてもらったのでそれを参考に改良を加えたプログラムを次節に掲載する。というのも、野崎氏 [4] によれば、これら 467 個がちょうど、7 次元ユークリッド空間内の単位球面上の、サイズが最大の 2 距離集合（その集合の異なる 2 点間の距離が 2 通り）になっていることから、分類を独立に実行したとのことである。

5　467 個のグラフの生成プログラム

まず、上記 E_6 の場合と同様にして、E_8 ルート系 240 頂点からなるグラフを作り、1 点の近傍として 56 点からなる部分グラフを作る。これが 28 個の組になっていることを確認する。

```
LE8:=Lattice("E",8);
E8short:=ShortestVectors(LE8);
#E8short eq 120;
E8:=&join{{x,-x}:x in E8short};
#E8 eq 240;
V:={@x:x in E8@};
E8graph:=Graph<V|{{x,y}:x,y in E8|(x,y) eq 1}>;
VG:=Vertices(E8graph);
a:=LE8!V[1];
N:={b:b in E8|(a,b) eq 1};
#N eq 56;
NN:={{Position(V,b),Position(V,a-b)}:b in N};
#NN eq 28;
autE8:=AutomorphismGroup(E8graph);
G:=Stabilizer(autE8,1);
```

ここで、

$$N = \bigcup_{i=1}^{28} \{b_i, a - b_i\}$$

とおけば、問題はここで定義した群 G の作用によって、N の部分集合のうち {b, a-b} という形の集合とつねに 1 点で交わるような 2^{28} 通りの集合

$$\{R \mid \forall i \in \{1, \ldots, 28\},\ |R \cap \{b_i, a - b_i\}| = 1 \text{ かつ } |R| = 28\} \tag{1}$$

を軌道に分解しその代表元を求めることである。群 G の 2^{28} 次の置換表現を構成するのはメモリを大量に消費するので、その部分群 $W(D_6)$ を H とおいてまず H の作用で細かく軌道分解した代表元を求め、後でそれらの中から G の作用で同値なものだけを取り出す、という手法を採用する。ではまず群 H を構成しよう。a と直交するルート a1 を任意にとり、集合 {a1, -a1} を固定する部分群が H である。

```
a1:=LE8!Random({x:x in E8|(a,x) eq 0});
i1:=Position(V,a1);
i1m:=Position(V,-a1);
H:=Stabilizer(G,{i1,i1m});
```

群 H は集合 N 上に 2 つの軌道をもち、番号を適当に付け替えることでそれらは

$$\bigcup_{i=1}^{12} \{b_i, a - b_i\}, \quad \bigcup_{i=13}^{28} \{b_i, a - b_i\}$$

と表される。H を集合

$$\{S \mid \forall i \in \{13,\ldots,28\},\ |S \cap \{b_i, a-b_i\}| = 1 \text{ かつ } |S| = 16\} \tag{2}$$

に作用させると 37 個の軌道に分解される。この計算が瞬時にできるのは、この集合のサイズが 2^{16} であって、メモリ容量ぎりぎりまたはオーバーしかねない 2^{28} よりもはるかに小さいからである。

```
orbs:=Orbits(H,GSet(H,NN));
[#o:o in orbs] eq [12,16];
C1:=CartesianProduct(Setseq(orbs[1]));
C2:=CartesianProduct(Setseq(orbs[2]));
N1:={{r[i]:i in [1..12]}:r in C1};
N2:={{r[i]:i in [1..16]}:r in C2};
o2:=Orbits(H,GSet(H,N2));
#o2 eq 37;
```

次に、37 個の軌道の代表元を固定し、代表元の固定部分群によって集合

$$\{T \mid \forall i \in \{1,\ldots,12\},\ |T \cap \{b_i, a-b_i\}| = 1 \text{ かつ } |S| = 12\} \tag{3}$$

を軌道分解する。得られた軌道の代表元と、もとの (2) の代表元の和集合を作ることで、28 点部分集合ができる。このようにして、H による集合 (1) の軌道分解が 11197 個となることがわかる。

```
r2s:=[Random(o):o in o2];
stabs:=[Stabilizer(H,r2):r2 in r2s];
reps:=Setseq(&join{{Random(o) join r2s[j]
 :o in Orbits(stabs[j],GSet(stabs[j],N1))}
 :j in [1..#r2s]});
#reps eq 11197;
```

さて、G による軌道分解はもっと粗く、したがってこれら 11197 個を G の作用で同値類に分解しなければならない。ルート系の性質から、G の作用での同値性と抽象的なグラフとしての同型が同値であることがわかる。すなわち

```
IsConjugate(G,r,s);
```

と

```
IsIsomorphic(sub<E8graph|{VG|V[i]:i in r}>,
             sub<E8graph|{VG|V[i]:i in s}>);
```

は同じ結果を返すが、後者の方が判定が 2 倍くらい速い。そこで後者を利用する。これが以下の関数である。

```
subg:=func<r|sub<E8graph|{VG|V[i]:i in r}> >;
isiso:=func<r,s|IsIsomorphic(subg(r),subg(s)) >;
```

以上で準備が完了したので、これより 11197 個から同値類の代表元を抜き出す作業に入る。ここでも計算時間短縮のために次のような工夫をする。

- 見つかった代表元に対して、その自己同型群（すなわち G における固定部分群）の位数を計算し、それで |G| を割ることで軌道の長さがわかる。軌道の長さの総和は (1) のサイズである 2^{28} であるから、11197 個すべて調べなくても見つかった軌道の長さの和が 2^{28} に達したら終了してよい。ここで、軌道本体を保存するとメモリを消費してしまうので、軌道の長さの和だけを保存してそれに新たに見つかった軌道の長さを加えていく。

- たくさんの軌道が見つかっていくと、新しい軌道の候補者と思われたものがすでに見つかったものと同値になることが多くなり、新たな軌道が見つかりにくくなる。これを防ぐために、未発見の軌道の長さの総和と、新しいかどうか調べたい軌道の長さを比べて、後者の方が長い場合は同値判定をせずに捨てる、という近道をする。

これらを実際に書いたのが次のプログラムである。

```
remain:=2^28;
// 軌道の長さの総和。
// 見つかった軌道の長さをこれから引いて行って 0 になれば終了。
result:=[];
// G による同値類の代表元として見つかったものを入れておく場所。
g:=#G;
// G の位数。
for i in [1..#reps] do
// 11197 個の H による軌道の代表元に関するループ。
 grph:=subg(reps[i]);
// i 番目の代表元から作られる２８点の部分グラフ。
 stab:=#AutomorphismGroup(grph);
// その部分グラフの自己同型群、すなわち
// G における i 番目の代表元の固定部分群、の位数。
 if g/stab le remain and
// その代表元を含む G での軌道の長さが残っている
// 軌道の長さの総和を超えないとき、かつ
  &and{not (stab eq p[2] and IsIsomorphic(grph,p[1]))
   :p in result} then
// その部分グラフがすでに見つかっている G による代表元の
// 表す部分グラフのいずれとも同型でないとき、
  Append(~result,<grph,stab>);
// その部分グラフと固定部分群の位数の組を result に加える。
  remain -:=g/stab;
// 新しく見つかった軌道の長さを remain から引く。
```

```
 end if;
 if remain eq 0 then break; end if;
// 未発見の軌道の長さの総和 remain が 0 になったらループを中断。
end for;
#result eq 467;
```

これで G による (1) の軌道分解が 467 個となることが確かめられた。この節に掲載したプログラムの実行時間は Intel® Xeon® 2.66GHz Linux で

```
Magma V2.16-13
Total time: 13.650 seconds, Total memory usage: 56.84MB
```

であった。

参考文献

[1] D. Cvetković, M. Lepović, P. Rowlinson and S.K. Simić, The maximal exceptional graphs, J. Combin. Theory, Ser. B 86 (2002), 347–363.

[2] P. J. Cameron, J. M. Goethals, J. J. Seidel, and E. E. Shult, Line graphs, root systems, and elliptic geometry, J. Algebra, 43 (1976), 305–327.

[3] M. Kitazume and A. Munemasa, Classification of exceptional graphs of smallest eigenvalue -2 — Maximal subgraphs embedded in E_6, E_7, E_8, unpublished.

[4] H. Nozaki, Classification of maximum two-distance sets on a sphere, in preparation.

Magma のグレブナ基底計算を利用した暗号解析
～多変数公開鍵暗号に対する代数攻撃～

藤田　亮
中央大学研究開発機構

2010 年 11 月 6 日

謝辞

研究集会「Magma で広がる数学の世界」における講演の機会を与えてくださった，本研究集会のオーガナイザーを務められた木田雅成教授（電気通信大学），ならびに，松尾和人教授（情報セキュリティ大学院大学）に感謝いたします．

本研究における成果の多くは，辻井重男教授，只木孝太郎准教授，五太子政史専任研究員（中央大学研究開発機構）との共同研究によるものです．また，本研究における計算機実験環境においては，松尾教授，只木准教授，五太子研究員によるご協力と，中央大学 21 世紀 COE プログラム「電子社会の信頼性向上と情報セキュリティ」ならびに，平成 20 年度総務省戦略的情報通信研究開発推進制度（SCOPE）ICT イノベーション創出型研究開発 ICT 安心・安全技術（量子コンピュータの出現に対抗し得る公開鍵暗号の研究）の援助を受けました．この場をお借りして，御礼申し上げます．

1　はじめに

グレブナ基底は，1960 年代，多項式環のイデアルに関する問題を解くために導入された概念である．グレブナ基底の理論および応用は，非常に多岐にわたっており，特に，暗号の分野において，多変数非線形連立方程式の解を求めるために，しばしば，グレブナ基底計算が用いられる．とりわけ，多変数公開鍵暗号と呼ばれる公開鍵暗号に対しては，暗号解析や安全性解析において，グレブナ基底計算を行うために，Magma が利用されることが多い．

本文では，多変数公開鍵暗号に対する暗号解析のための一手法として広く知られている，代数攻撃について取り上げ，そのために提案されている，さまざまな方法について紹介する．また，代数攻撃を利用した，多変数公開鍵暗号の安全性解析において，これまでに行われている研究成果について述べる．

本文の構成は以下の通りである．

まず 2 節では，有限体上の多変数非線形連立方程式の求解について，具体的に Magma を利用して解く例を提示し，グレブナ基底計算が実際にどのように行われるかについて述べる．

続く 3 節では，多変数非線形連立方程式の求解困難性に，その安全性を求める，多変数公開鍵暗号について説明する．多変数公開鍵暗号は，1980 年代に，日本が発祥の公開鍵暗号であり，多くの文献 [Kob98, Kob99, DGS06b, TK08, BBD09] に取り上げられているように，近年，注目を集めている，量子コンピュータの出現に対抗し得る公開鍵暗号の候補の一つとして考えられている暗号方式である[*1]．

4 節では，多変数公開鍵暗号に対する，代表的な攻撃手法として知られている，代数攻撃について述べる．Magma のグレブナ基底計算は，多変数公開鍵暗号に対する代数攻撃のための強力なツールとして利用可能である．一方で，Magma を利用する以外に，代数攻撃のために数多くの手法が，これまでに提案されている．本文では，これらの手法について簡単に紹介する．

5 節では，代数攻撃の手法を利用した，多変数公開鍵暗号の安全性解析において，これまでに得られている結果について述べる．多変数公開鍵暗号の安全性を強化するための一手法として提案されている持駒方式について，代数攻撃に対する安全性を評価する際に，

[*1] その他の候補としては，線形符号の復号問題に基づく暗号方式，格子に関する問題に基づく暗号方式，ハッシュ関数の安全性に基づく暗号方式などが考えられている [BBD09].

Magma のグレブナ基底計算を用いている．

6 節では，まとめと今後の研究課題について述べる．

1.1 記法について

本文では，ベクトルについて，たとえば $\boldsymbol{p}$, $\boldsymbol{X}$ といったようにボールド体を用いる．

- $\circ$: 合成写像.
- $\mathbf{F}_q$: GF(q)，すなわち元の数が q の有限体.
- $(\mathbf{F}_q)^n = \left\{(\alpha_1, \ldots, \alpha_n)^T \mid \alpha_i \in \mathbf{F}_q, i = 1, \ldots, n\right\}$: n 次元 $\mathbf{F}_q$ ベクトル空間.
- ^{T} : ベクトルの転置.
- $\mathbf{F}_q[x_1, \ldots, x_n]$: 係数を $\mathbf{F}_q$ に持つ，$x_1, \ldots, x_n$ を変数とする多項式全体の集合（多項式環）.
- ϕ: $\mathbf{F}_{q^l} \to (\mathbf{F}_q)^l$: $\mathbf{F}_{q^l}$ の多項式基底 $\omega_1, \ldots, \omega_l$ を用いた以下の写像：

$$\begin{aligned} \phi(x_1\omega_1 + x_2\omega_2 + \cdots + x_l\omega_l) = \ \phi(X) \ &= \quad (x_1, x_2, \ldots, x_l)^T \quad = \boldsymbol{x}, \\ \phi^{-1}((x_1, x_2, \ldots, x_l)^T) \quad &= \phi^{-1}(\boldsymbol{x}) = x_1\omega_1 + x_2\omega_2 + \cdots + x_l\omega_l = X. \end{aligned}$$

  例えば，$q = 2$, $l = 3$ の場合に，多項式基底を $\omega_1 = 1$, $\omega_2 = \omega$, $\omega_3 = \omega^2$ として，$\omega^3 + \omega + 1$ などのような，ω に関する l 次の $\mathbf{F}_q$ 上既約多項式 $f(\omega)$ に対し，$\mathbf{F}_{q^l} = \mathbf{F}_q[\omega]/f(\omega)$ を考えると，ベクトル空間 $(\mathbf{F}_q)^l$ と拡大体 $\mathbf{F}_{q^l}$ は ϕ を通じて同一と見ることができる．
- $x \in_U A$: 要素 x は，集合 A から一様な確率でランダムに選んだものである．

1.2 計算機実験環境

本文における計算機実験において，下記の環境 A，または，環境 B を使用した．

計算機実験環境 A

- コンピュータ：PROSIDE edAEW416R2 workstation
- プロセッサ：2.80GHz AMD Opteron Model 854 プロセッサ
- メモリ：64GB RAM
- 数式処理ソフトウェア：Magma V2.12-21
- グレブナ基底計算アルゴリズム：F_4
- 項順序（単項式順序）：全次数逆辞書式順序（DRL; grevlex）

計算機実験環境 B

- コンピュータ：日本コンピューティングシステム（JCS）VC98220WSA-4U/T workstation
- プロセッサ：2.80GHz AMD Opteron 8220 クアッドコアプロセッサ
- メモリ：128GB RAM
- 数式処理ソフトウェア：Magma V2.15-5
- グレブナ基底計算アルゴリズム：F_4
- 項順序（単項式順序）：全次数逆辞書式順序（DRL; grevlex）

2 有限体上の多変数非線形連立方程式の求解

例えば，係数体を $\mathbf{F}_7$ とする，下記の方程式 (2.1) の解を求めることを考える．

$$\begin{cases} 3x^2 + xy + 2xz + 6y^2 + 4yz + 5z^2 + 6x + 4y + 5z = 3 \\ 6x^2 + 6xy + xz + 3y^2 + yz + 3z^2 + 2x + y + 4z = 5 \\ 5x^2 + 2xy + 3xz + 3y^2 + 4yz + 5z^2 + 6x + 5y + z = 3 \end{cases} \tag{2.1}$$

方程式 (2.1) は，一般に，$\mathbf{F}_q[x_1, \ldots, x_n]$ の要素である m 個の多項式 $e_1(x_1, \ldots, x_n), \ldots, e_m(x_1, \ldots, x_n)$ と，m 次元ベクトル $\boldsymbol{c} = (c_1, c_2, \ldots, c_m)^T \in (\mathbf{F}_q)^m$ を用いて，以下の式 (2.2) のように表すことができる．

$$\begin{cases} e_1(x_1, \ldots, x_n) = c_1 \\ e_2(x_1, \ldots, x_n) = c_2 \\ \vdots \\ e_m(x_1, \ldots, x_n) = c_m \end{cases} \tag{2.2}$$

このような方程式を解くための準備として，まず，方程式 (2.2) を以下のような形に変形する．

$$\begin{cases} f_1 = e_1(x_1, \ldots, x_n) - c_1 = 0 \\ f_2 = e_2(x_1, \ldots, x_n) - c_2 = 0 \\ \vdots \\ f_m = e_m(x_1, \ldots, x_n) - c_m = 0 \end{cases} \tag{2.3}$$

方程式 (2.1) について，ここまでを，Magma を用いて計算する[*2]．

[*2] 方程式 (2.1) に関する一連の計算に使用した Magma のバージョンは V2.12-17 である．

```
> q := 7; // 有限体の位数
> n := 3; // 変数の個数
> m := 3; // 式の数
> G := GF(q); // 有限体 G を定義する
> R<x,y,z> := PolynomialRing(G,n); // G 上の n 変数多項式環 R を定義する
> E := [R |
> 3*x^2 +   x*y + 2*x*z + 6*y^2 + 4*y*z + 5*z^2 + 6*x + 4*y + 5*z,
> 6*x^2 + 6*x*y +   x*z + 3*y^2 +   y*z + 3*z^2 + 2*x +   y + 4*z,
> 5*x^2 + 2*x*y + 3*x*z + 3*y^2 + 4*y*z + 5*z^2 + 6*x + 5*y +   z
> ];
> c := [G | 3, 5, 3];
> F := [E[i] - c[i] : i in [1..m]];
> F;
[
    3*x^2 + x*y + 2*x*z + 6*x + 6*y^2 + 4*y*z + 4*y + 5*z^2 + 5*z + 4,
    6*x^2 + 6*x*y + x*z + 2*x + 3*y^2 + y*z + y + 3*z^2 + 4*z + 2,
    5*x^2 + 2*x*y + 3*x*z + 6*x + 3*y^2 + 4*y*z + 5*y + 5*z^2 + z + 4
]
```

なお，方程式に用いられる変数 x, y, z を，例えば u[1], u[2], u[3] といったように表示するには，下記のように，多項式環を定義すればよい．

```
> Ru<[u]> := PolynomialRing(G,n);
> Ru;
Polynomial ring of rank 3 over GF(7)
Order: Lexicographical
Variables: u[1], u[2], u[3]
> Ru ! F[1]; // F[1] を Ru の元として表す
3*u[1]^2 + u[1]*u[2] + 2*u[1]*u[3] + 6*u[1] + 6*u[2]^2 + 4*u[2]*u[3] + 4*u[2]
         + 5*u[3]^2 + 5*u[3] + 4
```

多項式環における項順序については後述するが，Magma において，項順序を何も指定しない場合，辞書式順序（lexicographic order）として定義される．

2.1 XL アルゴリズム

方程式 (2.3) を解く方法として，いくつか考えられるが，ここでは，まず，XL（eXtended Linearization）と呼ばれる，以下に示すアルゴリズム [CKPS00] を用いて解くことを考える．

1. ある次数 d 以下のすべての単項式を各多項式 $f_1, \ldots, f_m$ に乗じ，その結果からなる多項式集合（ここでは LargeF と表す）を生成する．
2. 1. において生成された LargeF の各要素である，多項式のなかに現れる単項式を，それぞれ別個の変数とみなし，各多項式における係数からなる行列（ここでは CoefMat と表す）を掃き出す．ただし，項順序は，ある変数（ここでは x_i とする）が，最後に消去されるような順序とする．
3. 2. において，x_i に関する一変数多項式が得られたものとする．この多項式の零点を求めることにより，その変数 x_i の値を得る．
4. 3. において求めた変数を消去することにより，方程式を簡単化し，他のすべての変数の値が得られるまで 1. 2. 3. を繰り返す．

Magma を利用し，方程式 (2.1) を XL アルゴリズムによって解いてみる．

```
> // *********************
> // *****  Step 1.  *****
> // *********************
> d := 5;
> LinPoly := &+[R.i : i in [1..Rank(R)]] + 1;
> dPoly := (LinPoly)^d;
> MonosSeq := Monomials(dPoly); // d 次以下のすべての単項式からなる Sequence
> LargeF := [];
> for i in [1..m] do
for>    for j in [1..#MonosSeq] do
for|for>                Include(~LargeF, F[i] * MonosSeq[j]);
for|for>        end for;
for> end for;
> // *********************
> // *****  Step 2.  *****
> // *********************
> PickUpMonos := [];
> for i in [1..#LargeF] do
for> // MonosLargeF は LargeF[i] に含まれる単項式からなる Sequence
for>    MonosLargeF := Monomials(LargeF[i]);
for>    for j in [1..#MonosLargeF] do
for|for>                Include(~PickUpMonos, MonosLargeF[j]);
for|for>        end for;
for> end for;
> MonosSet := SequenceToSet(PickUpMonos); // Set にして，重複する要素（単項式）を消す
> MonosSeq0 := SetToSequence(MonosSet); // Set から Sequence に戻す
> // MonosSeq0 の各要素の順序を並べ替える
> Apoly := &+[MonosSeq0[i] : i in [1..#MonosSeq0]];
> MonosSeq := Monomials(Apoly); // 項順序に従って，Sequence の要素の順番を並べ替える
> CoefSeq := [[BaseRing(Parent(LargeF[1])) | 0 : i in [1..#MonosSeq]] : i in [1..#LargeF]];
```

```
> for i in [1..#LargeF] do
for>    for j in [1..#MonosSeq] do
for|for>                CoefSeq[i][j] := MonomialCoefficient(LargeF[i],MonosSeq[j]);
for|for>        end for;
for> end for;
> CoefMat := Matrix(CoefSeq); // LargeF[i] の各多項式における，単項式の係数からなる行列
> Sweep := EchelonForm(CoefMat); // 行列 CoefMat を掃き出す
> TailPoly := &+[MonosSeq[i] * Sweep[Rank(Sweep)][i] : i in [1..#MonosSeq]];
> TailPoly;
z^7 + 3*z^6 + 5*z^5 + 6*z^3 + 3
```

以上，方程式 (2.1) に対し，XL アルゴリズムにおける 1. 2. を行うことにより，z に関する一変数多項式 (2.4) を得ることができた．

$$z^7 + 3z^6 + 5z^5 + 6z^3 + 3 \tag{2.4}$$

Magma を用いて得られた式 (2.4) は，多変数多項式環の元となっており，例えば，`Roots` コマンドなどを使って，直接，零点を求めようとするとエラーとなる．そこで，`UnivariatePolynomial` コマンドを用いて一変数多項式環上の多項式に移し変えてから解いてみる．

```
> // *********************
> // *****  Step 3.  *****
> // *********************
> UniTailPoly := UnivariatePolynomial(TailPoly);
> Factorization(UniTailPoly);
[
    <$.1 + 4, 1>,
    <$.1^2 + 2, 1>,
    <$.1^4 + 6*$.1^3 + $.1 + 3, 1>
]
> Roots(UniTailPoly); // TailPoly の零点を求める.
[ <3, 1> ]
```

以上により，変数 z の値が 3 と求めることができた．以下，方程式から z を消去し，残りの変数 x, y を求めてゆくが，ここでは省略する．

なお，XL アルゴリズムに関する理論的な解析や計算量については，以下に述べるグレブナ基底計算と関連して，文献 [CP03, YC04a, YCC04, YC04b] などに述べられている．また，本文において取り上げる，多変数公開鍵暗号に対する代数攻撃以外に，共通鍵暗号に対する攻撃手法としての XL アルゴリズムについては，文献 [CP02, Cou02] などに記述されている．

2.2 グレブナ基底計算

XL アルゴリズムにより，方程式 (2.3) における多項式 $f_1, \ldots, f_m$ から，一変数多項式を得ようとする操作は，実は，多項式環 $\mathbf{F}_q[x_1, \ldots, x_n]$ における，$f_1, \ldots, f_m$ を基底とするイデアル

$$I = \langle f_1, \ldots, f_m \rangle = \{g_1 f_1 + \cdots + g_m f_m \mid g_1, \ldots, g_m \in \mathbf{F}_q[x_1, \ldots, x_n]\} \tag{2.5}$$

を考え，この I における要素を，くまなく探していることになる．つまり，各 g_i として，まず，d 次以下の単項式を考え，これらを各 f_i に乗じたもの同士の和を，順次，取ってゆくことにより，I の要素を探す操作を行っている．

式 (2.5) のようなイデアル I の基底は一意でないが，それらのうち，性質のよいものの一つとして，グレブナ基底と呼ばれる基底が知られている．なお，ここでは，グレブナ基底の概念や理論などについて詳述しない．これらについては，これまでに数多くの良書が出版されているので，それらを参照されたい（例えば [CLO00, KR00, NY03, KR05, Hib06, CLO07] など）．

さて，グレブナ基底を計算するアルゴリズムとして，いくつかのアルゴリズムが，これまでに提案されているが，Magma には，Buchberger によるアルゴリズム [Buc65]，Faugère による F_4 アルゴリズム [Fau99] が実装されている．特に F_4 アルゴリズムは，XL アルゴリズムよりも，グレブナ基底の計算において，効率的となり得ることが示されている [AFIKS04]．

方程式 (2.1) の解空間をなす，式 (2.5) のようなイデアル I について，Magma を利用して，グレブナ基底を計算する．

```
> I := ideal<R | [F[i] : i in [1..m]]>; // 各 F[i] を基底とするイデアル
> GroebnerBasis(I); // I のグレブナ基底を求める
[
    x + 3*y + 2*z^6 + 2*z^5 + 5*z^4 + 3*z^3 + 5*z^2 + 4*z,
    y^2 + 5*z^5 + 3*z^3 + z^2 + 2*z + 3,
    y*z + 4*y + 5*z^6 + 2*z^5 + 3*z^4 + 4*z^3 + 2*z^2 + 4*z + 3,
    z^7 + 3*z^6 + 5*z^5 + 6*z^3 + 3
]
```

方程式 (2.1) に関する，ここまでの計算では，項順序を特に設定していなかったため，辞書式順序としてグレブナ基底の計算を行っている．辞書式順序のグレブナ基底は三角形式，すなわち，一変数ずつ解いてゆくことができるような形となることが知られている．

グレブナ基底の計算過程を考えず，Magma を利用して，単に，イデアルの零点を求めるだけであれば，以下のように計算することも可能となっている．

```
> I := ideal<R | [F[i] : i in [1..m]]>;
> Variety(I); // I の零点を求める
[ <0, 4, 3>, <3, 3, 3> ]
>
> VarietySequence(I); // Sequence として解を出力する
[
    [ 0, 4, 3 ],
    [ 3, 3, 3 ]
]
```

2.2.1 項順序

グレブナ基底を計算する際，計算効率に大きく影響を及ぼすのが，項順序である．

Magma では，代表的な項順序として，辞書式順序（lexicographic order），全次数辞書式順序（degree（graded）lexicographic order），全次数逆辞書式順序（Degree（Graded）Reverse Lexicographic order: DRL（grevlex））を利用することができる[*3]．一般に，全次数逆辞書式順序によるグレブナ基底計算が高速であり，辞書式順序による計算は，さほど速くないと考えられている．

```
> R1<x,y,z> := PolynomialRing(GF(7),3,"lex"); // 辞書式順序
> R1;
Polynomial ring of rank 3 over GF(7)
Lexicographical Order
Variables: x, y, z
> R2<x,y,z> := PolynomialRing(GF(7),3,"glex"); // 全次数辞書式順序
> R2;
Polynomial ring of rank 3 over GF(7)
Graded Lexicographical Order
Variables: x, y, z
> R3<x,y,z> := PolynomialRing(GF(7),3,"grevlex"); // 全次数「逆」辞書式順序
> R3;
Polynomial ring of rank 3 over GF(7)
Graded Reverse Lexicographical Order
Variables: x, y, z
```

[*3] これらの項順序以外では，消去順序，ブロック順序，重み付き順序などの項順序が利用できる．

例えば，$f(x,y,z)=(x+y+z+1)^3$ は，これらの項順序に基づいて，下記のように表される．

```
> f := (x+y+z+1)^3;
> R1 ! f; // 辞書式順序
x^3 + 3*x^2*y + 3*x^2*z + 3*x^2 + 3*x*y^2 + 6*x*y*z + 6*x*y + 3*x*z^2
    + 6*x*z + 3*x + y^3 + 3*y^2*z + 3*y^2 + 3*y*z^2 + 6*y*z + 3*y
    + z^3 + 3*z^2 + 3*z + 1
> R2 ! f; // 全次数辞書式順序
x^3 + 3*x^2*y + 3*x^2*z + 3*x*y^2 + 6*x*y*z + 3*x*z^2 + y^3 + 3*y^2*z
    + 3*y*z^2 + z^3 + 3*x^2 + 6*x*y + 6*x*z + 3*y^2 + 6*y*z + 3*z^2
    + 3*x + 3*y + 3*z + 1
> R3 ! f; // 全次数「逆」辞書式順序
x^3 + 3*x^2*y + 3*x*y^2 + y^3 + 3*x^2*z + 6*x*y*z + 3*y^2*z + 3*x*z^2
    + 3*y*z^2 + z^3 + 3*x^2 + 6*x*y + 3*y^2 + 6*x*z + 6*y*z + 3*z^2
    + 3*x + 3*y + 3*z + 1
```

2.2.2 表示オプション

Magma のコマンド `SetVerbose` によって値を設定することにより，グレブナ基底の計算過程を表示させることができる．設定する数値が大きくなるにつれて，計算過程のより詳細を表示させることができるようになっている．

```
> // グレブナ基底の計算過程を表示する（0: 表示しない; 1, 2, 3, 4: 表示する）
> SetVerbose("Groebner",4);
> GetVerbose("Groebner"); // 設定されている値を出力する
4
```

その他，グレブナ基底計算に関しては `Buchberger`, `Faugere` と，グレブナ基底の項順序を変換するアルゴリズムである `FGLM` [FGLM93], `GroebnerWalk` [CKM97] にも表示オプションがあり，それぞれのアルゴリズムにおいて，計算過程の表示について設定することができる．

2.2.3 計算アルゴリズム

Magma のデフォルトでは，実装されている F_4 アルゴリズム [Fau99] を利用して，グレブナ基底計算を行っている．

例えば，方程式 (2.1) を解くために行ったグレブナ基底計算について，Magma の F_4 アルゴリズムによる計算過程を表示すると，下記のようになる．

```
> I := ideal<R | [F[i] : i in [1..m]]>;
> SetVerbose("Groebner",1); // グレブナ基底の計算過程を表示する
> GroebnerBasis(I);
Homogeneous weights search
Number of variables: 3, nullity: 0
********************
FAUGERE F4 ALGORITHM
********************
Coefficient ring: GF(7)
Rank: 3
Order: Graded Reverse Lexicographical
NEW hash table
Matrix kind: Modular FP
Datum size: 4
Initial length: 3
Inhomogeneous

*******
STEP 1
Basis length: 3, queue length: 2, step degree: 2, num pairs: 2
Basis total mons: 30, average length: 10.000
Number of pair polynomials: 2, at 10 column(s), 0.000
Average length for reductees: 10.00 [2], reductors: 10.00 [1]
Symbolic reduction time: 0.000, column sort time: 0.000
2 + 1 = 3 rows / 10 columns, 100% / 100% (10/r)
Before ech memory: 3.5MB
Row sort time: 0.000
0.000 + 0.000 = 0.000 [2]
After ech memory: 3.5MB
Queue insertion time: 0.000
Step 1 time: 0.000, [0.010], mat/total: 0.000/0.000 [0.010], mem: 3.5MB

*******
STEP 2
Basis length: 5, queue length: 2, step degree: 3, num pairs: 2
Basis total mons: 46, average length: 9.200
Number of pair polynomials: 2, at 15 column(s), 0.000
Average length for reductees: 8.00 [2], reductors: 8.67 [9]
Symbolic reduction time: 0.000, column sort time: 0.000
2 + 9 = 11 rows / 19 columns, 44.976% / 62.959% (8.5455/r)
Before ech memory: 3.5MB
Row sort time: 0.000
0.000 + 0.000 = 0.000 [2]
After ech memory: 3.5MB
Queue insertion time: 0.000
Step 2 time: 0.000, [0.000], mat/total: 0.000/0.000 [0.010], mem: 3.5MB
```

```
*******
STEP 3
Basis length: 7, queue length: 4, step degree: 4, num pairs: 4
Basis total mons: 64, average length: 9.143
Number of pair polynomials: 4, at 20 column(s), 0.000
Average length for reductees: 9.00 [4], reductors: 8.77 [13]
Symbolic reduction time: 0.000, column sort time: 0.000
4 + 13 = 17 rows / 22 columns, 40.107% / 58.743% (8.8235/r)
Before ech memory: 3.5MB
Row sort time: 0.000
0.000 + 0.000 = 0.000 [1]
After ech memory: 3.5MB
Queue insertion time: 0.000
Step 3 time: 0.000, [0.000], mat/total: 0.000/0.000 [0.020], mem: 3.5MB

*******
STEP 4
Basis length: 8, queue length: 2, step degree: 5, num pairs: 2
Basis total mons: 71, average length: 8.875
Number of pair polynomials: 2, at 18 column(s), 0.000
Average length for reductees: 7.00 [2], reductors: 8.57 [14]
Symbolic reduction time: 0.000, column sort time: 0.000
2 + 14 = 16 rows / 22 columns, 38.068% / 58.035% (8.375/r)
Before ech memory: 3.5MB
Row sort time: 0.000
0.000 + 0.000 = 0.000 [0]
After ech memory: 3.5MB
Queue insertion time: 0.000
Step 4 time: 0.000, [0.010], mat/total: 0.000/0.000 [0.030], mem: 3.5MB

Reduce 8 final polynomial(s) by 8
0 redundant polynomial(s) removed; time: 0.000
Interreduce 6 (out of 8) polynomial(s)
Symbolic reduction time: 0.000
6 + 0 = 6 rows / 14 columns, 60.714% / 80.52% (8.5/r)
Row sort time: 0.000
0.000 + 0.000 = 0.000 [6]
Interreduction time: 0.000
Final number of polynomials: 8

Number of pairs: 10
Total pair setup time: 0.000
Max num entries matrix: 17 by 22
Max num rows matrix: 17 by 22
Total symbolic reduction time: 0.000
Total column sort time: 0.000
Total row sort time: 0.000
```

```
Total matrix time: 0.000
Total new polys time: 0.000
Total queue update time: 0.000
Total Faugere F4 time: 0.000, real time: 0.040
*****************
FGLM ORDER CHANGE
*****************

Coefficient ring: GF(7)
Rank: 3
Initial order: Graded Reverse Lexicographical
Final order: Lexicographical
Basis length: 6
Dimension: 8
New polynomial 0, leading monomial: z^7
New polynomial 1, leading monomial: y*z
New polynomial 2, leading monomial: y^2
New polynomial 3, leading monomial: x
Total FGLM time: 0.000
[
    x + 3*y + 2*z^6 + 2*z^5 + 5*z^4 + 3*z^3 + 5*z^2 + 4*z,
    y^2 + 5*z^5 + 3*z^3 + z^2 + 2*z + 3,
    y*z + 4*y + 5*z^6 + 2*z^5 + 3*z^4 + 4*z^3 + 2*z^2 + 4*z + 3,
    z^7 + 3*z^6 + 5*z^5 + 6*z^3 + 3
]
```

F_4 アルゴリズムによるグレブナ基底計算は，V2.11 以降の Magma に導入され，開発者のホームページにおいて，その性能について記載されている [Ste04]．当時，開発されていた他のソフトウェアとの性能比較などが行われており，非常に高速にグレブナ基底計算を行うことが報告されている．その後，Magma におけるグレブナ基底計算アルゴリズムの改良が行われている一方で，最近の結果では，F_4 アルゴリズムの改良版である F_5 アルゴリズム [Fau02] と，行列演算部分を改良したパッケージを利用して，グレブナ基底計算のベンチマークとして，しばしば用いられる Katsura-n 方程式 [KFSM83, KFIFG87] の求解において，$\mathbf{F}_{65521}$ 上の Katsura-16 方程式を 1 時間半程度かけて解いたという報告がなされている [FL10] [*4]．このアルゴリズムが，本文執筆時（2010 年 11 月）における，世界最速のグレブナ基底計算アルゴリズムと見られる．

F_4 アルゴリズムではなく，従来のアルゴリズムである，Buchberger アルゴリズムを用いて計算する場合には，以下のように入力すればよい．

[*4] 複数の CPU を使用した並列計算を，アルゴリズムに組み込むことによって，高速化を図っているものと思われるが，Magma V2.16-1 を利用した場合と比較して，数十倍から数百倍，高速であるという結果が示されている．

```
> I := ideal<R | [F[i] : i in [1..m]]>;
> SetVerbose("Groebner",0);
> SetVerbose("Buchberger",4); // Buchberger アルゴリズムだけを詳しく表示させる
> GroebnerBasis(I: Faugere := false); // F4 アルゴリズムを使わずに計算する
********************
BUCHBERGER ALGORITHM
********************

Coefficient ring: GF(7)
Rank: 3
Order: Graded Reverse Lexicographical
Initial length: 3
Using monomial division list
Initial Reduce: true
Remove Redundant: true
New Reduce: false
Final Reduce: true
Homogenization: false
Initial basis:
[
    3*x^2 + x*y + 6*y^2 + 2*x*z + 4*y*z + 5*z^2 + 6*x + 4*y + 5*z + 4,
    6*x^2 + 6*x*y + 3*y^2 + x*z + y*z + 3*z^2 + 2*x + y + 4*z + 2,
    5*x^2 + 2*x*y + 3*y^2 + 3*x*z + 4*y*z + 5*z^2 + 6*x + 5*y + z + 4
]
Reduce initial basis

Insert 0:
    x*y + 6*x*z + 6*y*z + 4*z^2 + 2*x + 2*y + 6*z + 6
    Total degree: 2

Insert 1:
    y^2 + 3*x*z + 5*y*z + z^2 + 2*x + 4*y + z + 1
    Total degree: 2

Insert 2:
    x^2 + 2*x*z + y*z + 3*z^2 + 2*x + 2*y + 2
    Total degree: 2

Reduce pairs

1 [0, 1] (0 done): Degree: 3, lcm = x*y^2, 0.000

Insert 3:
    x*z^2 + 3*y*z^2 + 3*z^3 + 4*x*z + 2*y*z + 5*x + 3*y + 2*z
    Total degree: 3

2 [0, 2] (1 done): Degree: 3, lcm = x^2*y, 0.000
```

```
Insert 4:
    y*z^2 + 4*z^3 + 5*x*z + 5*y*z + 5*z^2 + 6*x + 4*y + 6*z + 4
    Total degree: 3

3 [1, 4] (2 done): Degree: 4, lcm = y^2*z^2, 0.000

Insert 5:
    z^4 + 6*z^3 + x*z + 5*y*z + 4*x + 6*y + 3*z
    Total degree: 4

4 [0, 3] (3 done): Degree: 4, lcm = x*y*z^2, 0.000
3 [0, 4] (4 done): Degree: 4, lcm = x*y*z^2, 0.000
2 [2, 3] (5 done): Degree: 4, lcm = x^2*z^2, 0.000
1 [4, 5] (6 done): Degree: 5, lcm = y*z^4, 0.000
0 [3, 5] (7 done): Degree: 5, lcm = x*z^4, 0.000
Reduce final polynomials
Reduce 0
Reduce 1
Reduce 2
Reduce 3
Reduce 4
Reduce 5
Reduction time: 0.000
Number of pairs reduced: 8
Total Buchberger time: 0.000
[
    x + 3*y + 2*z^6 + 2*z^5 + 5*z^4 + 3*z^3 + 5*z^2 + 4*z,
    y^2 + 5*z^5 + 3*z^3 + z^2 + 2*z + 3,
    y*z + 4*y + 5*z^6 + 2*z^5 + 3*z^4 + 4*z^3 + 2*z^2 + 4*z + 3,
    z^7 + 3*z^6 + 5*z^5 + 6*z^3 + 3
]
```

グレブナ基底の計算過程に関する出力から，Magma では，グレブナ基底計算において，項順序として，全次数逆辞書式順序のみを用いて計算しているものと見られる．一般に，この順序による計算が高速と考えられているが，文献 [Saw02] などにおいて報告されているように，全次数逆辞書式順序による計算が，必ずしも高速であると限らない．解く問題によっては，非常に非効率な項順序となってしまうことも考えられ得る．また，F_4 アルゴリズムの提案者によって，その後，提示された F_5 アルゴリズム [Fau02] などのように，グレブナ基底の計算過程における無駄な計算をなくすことにより，Magma の F_4 アルゴリズムを利用するよりも，効率的に計算を行うことができたという報告がなされている [GT08b, MCDBB09].

Magma はオープンソースでないため，グレブナ基底計算について，どのような実装が

なされているか，窺い知ることができない．しかしながら，ここにいくつか挙げた改善点のように，Magma のグレブナ基底計算が効率化される可能性は，まだ残されていると考えられる．

2.3 問題の困難性

方程式 (2.2) を解くような，有限体上の多変数非線形連立方程式の求解は，NP 困難であることが知られている [GJ79, p.251]．実際に，計算機実験環境 A（1.2 節）のもとで，方程式 (2.2) について，$q = 2, n = m$ とし，方程式の次数を 2 とした場合に，グレブナ基底を計算するために要した時間を表 1 に示す．

表 1　連立 2 次方程式を解くためのグレブナ基底計算時間（係数体 $\mathbf{F}_2$, 変数の数 = 式の数）

n：変数の数	17	18	19	20	21	22	23	24	25	26	27	28	29
計算時間（秒）	1	2	4	8	16	54	85	577	1185	2516	5274	11521	22236
メモリ使用量（MB）	26	40	62	93	136	337	472	3065	5341	8733	13591	20423	30026

表 1 からわかるように，変数の数が増えるにつれて，計算時間，および，計算に必要なメモリは，指数的に増大する．ただし，この結果は，変数の数 n と，式の数 m が等しい場合であり，そうでない場合について，グレブナ基底計算以外の方法もまた，考えられ得る．上に述べた XL アルゴリズムは，本来，特に，式の数 m の方が，変数の数 n よりも非常に多い，すなわち $n \ll m$ である場合に，有効となるものと考えられたアルゴリズムである．また，これとは逆に $m \ll n$ である場合，有効となるアルゴリズムが，いくつか提案されている [CGMT02, Has09]．

さて，こうした NP 困難な問題は，現用のコンピュータのみならず，量子状態を利用して，高度な並列計算を行うことができる量子コンピュータを用いたとしても，解くことが困難であると強く信じられている．一方で，素因数分解や離散対数問題などといった，現用の公開鍵暗号である，RSA 暗号や楕円曲線暗号の安全性の根拠となっている問題は，量子コンピュータを用いて，容易に解かれてしまう [Sho94, BL95]．このような背景から，量子コンピュータの出現に対抗し得る公開鍵暗号の研究が盛んに行われており，その候補の一つとして，多変数非線形連立方程式の求解困難性に，その安全性を求める，多変数公開鍵暗号がある．

表 2　主要な多変数 2 次公開鍵暗号

	暗号系	Modifier を適用した暗号系	Modifier
MI 型	MI [MI88a]	C^{*--} [PGC98] SFLASH [CGP03]	–
		PMI [Din04]	i
		PMI+ [DG06]	i, +
	HFE [Pat96a]	HFE$^-$ [Pat96a]	–
		QUARTZ [PCG01]	v, –
		IPHFE [DS05a]	i
順序解法型	順序解法 [TKIFM86]	Birational Permutation Scheme [Sha93]	–
	R(S)SE [KS04, KS05a]	RSSE$^-$ [KS05b]	–
UOV 型	UOV [KPG99]		

–: Minus method, +: Plus method, v: Vinegar 変数, i: Internal Perturbation

3　多変数公開鍵暗号

多変数公開鍵暗号（Multivariate Public Key Cryptosystem: MPKC）とは，MI 暗号 [MI88a] や，順序解法に基づく公開鍵暗号 [TKIFM86] などのように，公開鍵が，平文変数，および，暗号文変数を変数とする多項式のタプルとして表される公開鍵暗号である．

多変数公開鍵暗号のうち，公開鍵多項式 $e_i \in \mathbf{F}_q[x_1, ..., x_n]$ の次数が 2，すなわち，平文変数 $x_1, \ldots, x_n$ を変数とする 2 次多項式タプルによって表され，暗号文変数 y_i について $y_i = e_i(x_1, ..., x_n)$ といった形をなすものを，ここでは，多変数 2 次公開鍵暗号と呼ぶ．

これまでに提案されている，主要な多変数 2 次公開鍵暗号について，表 2 に要約する．

3.1　多変数 2 次公開鍵暗号

多変数 2 次公開鍵暗号とは，n 変数からなるベクトルを平文変数ベクトル $\boldsymbol{x} = (x_1, \ldots, x_n)^T$，$m$ 変数からなるベクトルを暗号文変数ベクトル $\boldsymbol{y} = (y_1, \ldots, y_m)^T$ とし，線形変換 L_1, L_2 と非線形変換 G を秘密鍵，

$$\boldsymbol{y} = E(\boldsymbol{x}) = (L_1 \circ G \circ L_2)(\boldsymbol{x})$$

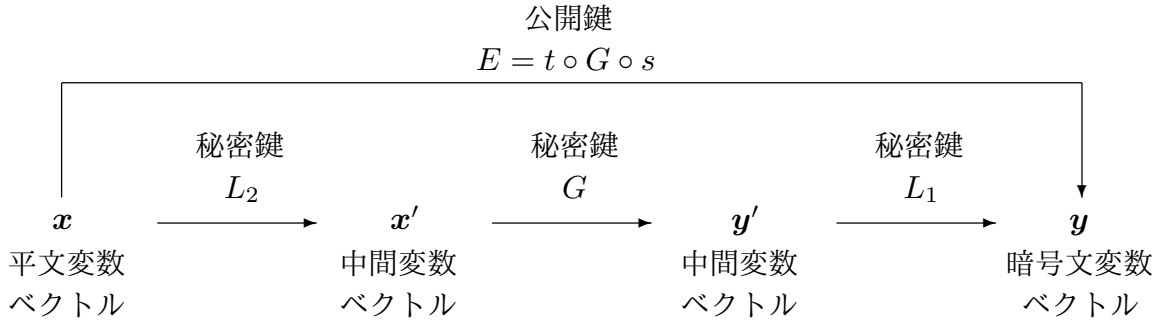

図 1　多変数 2 次公開鍵暗号

なる暗号化変換 E を公開鍵とする公開鍵暗号である（図 1）．$\boldsymbol{y}$ から $\boldsymbol{x}$ への復号変換は

$$\boldsymbol{x} = (L_2^{-1} \circ G^{-1} \circ L_1^{-1})(\boldsymbol{y})$$

となる．秘密鍵 G の違いにより，これまでにさまざまな多変数 2 次公開鍵暗号が提案されている．

多変数 2 次公開鍵暗号において，各公開鍵多項式は以下のような形をなす：

$$y_i = \sum_{1 \le j,l \le n} \alpha_{i,j,l} x_j x_l + \sum_{1 \le j \le n} \beta_{i,j} x_j + \gamma_i.$$

これらの方程式の各係数について，$\alpha_{i,j,l}$，$\beta_{i,j}$，$\gamma_i \in \mathbf{F}_q$ である．

一方，以下のように，公開鍵多項式の全次数が 3 となる，Dragon 型多変数公開鍵暗号が提案されている [Pat96b].

$$\sum_{\substack{1 \le j_1, j_2 \le n \\ 1 \le l \le m}} \alpha_{i,j_1,j_2,l}\ x_{j_1} x_{j_2} y_l + \sum_{1 \le j,l \le n} \beta_{i,j,l} x_j x_l$$
$$+ \sum_{\substack{1 \le j \le n \\ 1 \le l \le m}} \gamma_{i,j,l}\ x_j y_l + \sum_{1 \le j \le n} \delta_{i,j} x_j + \sum_{1 \le l \le m} \varepsilon_{i,l} y_l + \xi_i = 0.$$

これらの方程式の各係数について，$\alpha_{j_1,j_2,l}$，$\beta_{i,j,l}$，$\gamma_{i,j,l}$，$\delta_{i,j}$，$\varepsilon_{i,l}$，$\xi_i \in \mathbf{F}_q$ である．

3.1.1　暗号系

- パラメータ：
  - q : 暗号系を構成する有限体の位数.
  - n : 平文（ベクトル）の次元，すなわち，平文変数 x_i の個数.
  - m : 暗号文（ベクトル）の次元，すなわち，公開鍵多項式 $e_i(\boldsymbol{x})$ の数.

- 平文：$\boldsymbol{p} = (p_1, p_2, \ldots, p_n)^T \in (\mathbf{F}_q)^n$.
- 暗号文：$\boldsymbol{c} = (c_1, c_2, \ldots, c_m)^T \in (\mathbf{F}_q)^m$.
- 秘密鍵：
  1. $L_1 : (\mathbf{F}_q)^m \to (\mathbf{F}_q)^m$：正則な線形変換.
  2. $L_2 : (\mathbf{F}_q)^n \to (\mathbf{F}_q)^n$：正則な線形変換.
  3. $G : (\mathbf{F}_q)^n \to (\mathbf{F}_q)^m$：2 次多項式タプルによって表される，逆変換が容易な非線形変換.

  ここでの線形変換の代わりとして，一般にはアフィン変換を用いる.
- 公開鍵：$\boldsymbol{y} = E(\boldsymbol{x}) = (e_1(\boldsymbol{x}), \ldots, e_m(\boldsymbol{x}))^T$

  $\Leftrightarrow \boldsymbol{y} = (L_1 \circ G \circ L_2)(\boldsymbol{x})$

  ここに，$e_i(\boldsymbol{x}) \in \mathbf{F}_q[x_1, \ldots, x_n]$ $(i = 1, \ldots, m)$.
- 暗号化：公開鍵の $\boldsymbol{x}$ に平文 $\boldsymbol{p}$ を代入し，$\boldsymbol{y}$ の値となる暗号文 $\boldsymbol{c}$ を得る.

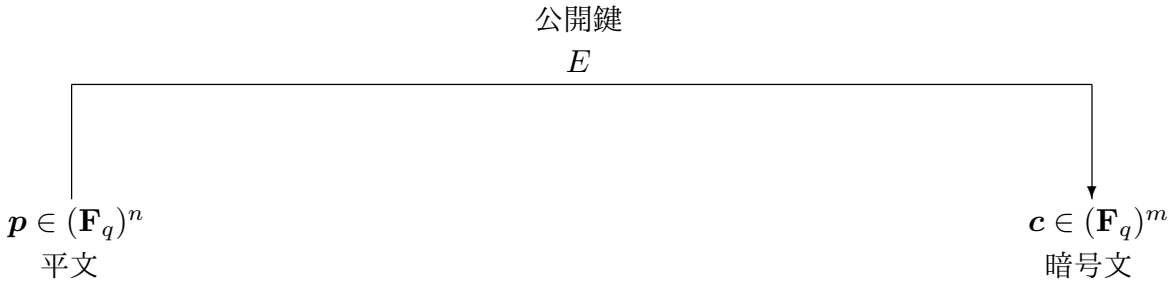

図 2　多変数 2 次公開鍵暗号系における暗号化

- 復号：
  1. $\boldsymbol{w} = L_1^{-1}(\boldsymbol{c})$.
  2. $\boldsymbol{v} = G^{-1}(\boldsymbol{w})$.
  3. $\boldsymbol{p} = L_2^{-1}(\boldsymbol{v})$.

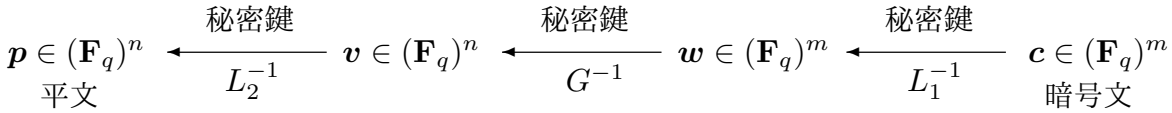

図 3　多変数 2 次公開鍵暗号系における復号

多変数 2 次公開鍵暗号系を用いた署名については，上記における平文を署名，暗号文を文書（ハッシュ値）とし，秘密鍵を署名鍵，公開鍵を検証鍵とする．また，秘密鍵を用いた復号が，署名者による署名鍵を用いた署名生成となり，署名検証は，文書 $\boldsymbol{c}$ と署名 $\boldsymbol{p}$ に

対し，検証鍵 E を用いて $\boldsymbol{c} = E(\boldsymbol{p})$ かどうか検査することとなる．

なお，図 1 に示した暗号系は秘密鍵 G について 1 段構成であるが，このような構成を拡張したものとして，非線形変換 G_1, G_2, 線形変換 L_1, L_2, L_3 について，公開鍵 $E = L_1 \circ G_1 \circ L_2 \circ G_2 \circ L_3$ といった 2 段構成をなし，公開鍵多項式の次数が 4 以上となる暗号系がいくつか提案されている [TFH89, PG97a, PG97b]．以下では，図 1 に示されるような，1 段構成の暗号系のみを取り上げる．

3.2 多変数 2 次公開鍵暗号の分類

多変数 2 次公開鍵暗号系の秘密鍵 G について，本文では，MI 型，順序解法型，UOV 型の 3 つに分類する．この分類法については，これまでにいくつか提案されているものの [WP05, DGS06b]，新たに提案されている暗号系について，こうした分類に含まれないものがある．そこで，本文では，上記のような，新しい分類を提案する．

以下では，表 2 に分類された，多変数 2 次公開鍵暗号の方式について説明する．これらの暗号方式のうち，いくつかの方式に関する Magma のプログラムが，多変数公開鍵暗号の解説書 [DGS06b] の著者らによる下記ホームページにて公開されている．

`http://math.uc.edu/~aac/MPKC/software.html`

3.2.1 MI 型

MI 型とは，中間変数ベクトル $\boldsymbol{v}$, $\boldsymbol{w}$ を拡大体の元 V, W として表現し，この拡大体上の単項式写像，あるいは多項式写像を非線形変換 G に用い，かつ，G^{-1} において，順序解法型に用いられるような計算を行わないものである（図 4）．

$$
\begin{tikzcd}
\boldsymbol{v} \in (\mathbf{F}_q)^n \arrow[r, shift left, "G"] \arrow[d, shift left, "\phi^{-1}"] & \boldsymbol{w} \in (\mathbf{F}_q)^m \arrow[l, shift left, "G^{-1}"] \arrow[d, shift left, "\phi^{-1}"] \\
V \in \mathbf{F}_{q^n} \arrow[r, shift left, "F"] \arrow[u, shift left, "\phi"] & W \in \mathbf{F}_{q^m} \arrow[l, shift left, "F^{-1}"] \arrow[u, shift left, "\phi"]
\end{tikzcd}
$$

図 4　MI 型の非線形変換 G

■MI (Matsumoto-Imai; MIA, C^*) [MIHM83, IM85, MI88a, MI88b]　中間変数ベクトル $\boldsymbol{v}$, $\boldsymbol{w}$ のそれぞれの次元 n, m について $n = m$ とする．$V = \phi^{-1}(\boldsymbol{v}) \in \mathbf{F}_{q^n}$ に関する単項式写像 F を $F(V) = V^{q^\theta+1} = V^\iota$ とする．ただし g.c.d.$(q^\theta + 1, q^n - 1) = 1$ とする．この条件は，F が全単射であることと同値である．

MI における G は以下のように表される：

$$G(\boldsymbol{v}) = (\phi \circ F \circ \phi^{-1})(\boldsymbol{v}).$$

法 $q^n - 1$ における ι の乗法の逆元を ι' とすると，$W = \phi^{-1}(\boldsymbol{w}) \in \mathbf{F}_{q^n}$ に対する F の逆変換 F^{-1} は以下のように表される：

$$F^{-1}(W) = W^{\iota'}.$$

F^{-1} を用いて，G の逆変換 G^{-1} は以下のように表される：

$$G^{-1}(\boldsymbol{w}) = (\phi \circ F^{-1} \circ \phi^{-1})(\boldsymbol{w}).$$

■HFE (Hidden Field Equations) [Pat96a]　中間変数ベクトル $\boldsymbol{v}$, $\boldsymbol{w}$ のそれぞれの次元 n, m について $n = m$ とする．$V = \phi^{-1}(\boldsymbol{v}) \in \mathbf{F}_{q^n}$ に関する d 次多項式写像 F が以下のような形をなすものとする：

$$F(V) = \sum_{\substack{0 \le i,j \le d \\ q^i + q^j \le d}} \beta_{i,j} V^{q^i + q^j} + \sum_{\substack{0 \le l \le d \\ q^l \le d}} \alpha_l V^{q^l} + \mu_0.$$

ここで，$\beta_{i,j}, \alpha_l, \mu_0 \in_U \mathbf{F}_{q^n}$ である．

HFE における G は以下のように表される：

$$G(\boldsymbol{v}) = (\phi \circ F \circ \phi^{-1})(\boldsymbol{v}).$$

HFE における G^{-1} は下記のように計算される：

1. $W = \phi^{-1}(\boldsymbol{w})$.
2. 1 において得られた W と未知変数 V に関する方程式 $F(V) = W$ を V について解く．
3. 2 において得られた V それぞれについて $\boldsymbol{v} = \phi(V)$ を得る．

3.2.2　順序解法型

順序解法型とは，G^{-1} を計算する際に，中間変数 $v_1, \ldots, v_n$ のうち，1 変数，あるいはいくつかの変数について順序的に解いてゆくものである．

■順序解法 [Tsu85, TKIFM86, Sha93]　順序解法における $G = (g_1, \ldots, g_n)$ は以下のように表される：

$$
\begin{aligned}
w_1 &= g_1(v_1, \ldots, v_n) = v_1 \\
w_2 &= g_2(v_1, \ldots, v_n) = v_2 \cdot l_2(v_1) + q_2(v_1) \\
w_3 &= g_3(v_1, \ldots, v_n) = v_3 \cdot l_3(v_1, v_2) + q_3(v_1, v_2) \\
&\vdots \\
w_n &= g_n(v_1, \ldots, v_n) = v_n \cdot l_n(v_1, \ldots, v_{n-1}) + q_n(v_1, \ldots, v_{n-1})
\end{aligned}
$$

ここに l_i は $v_1, \ldots, v_{i-1}$ に関する線形変換，q_i は $v_1, \ldots, v_{i-1}$ に関する非線形変換である．

順序解法 における G の逆変換 G^{-1} は以下のように計算される：

1. $v_1 = w_1$.
2. $v_2 = \dfrac{w_2 - q_2(v_1)}{l_2(v_1)}, v_3 = \dfrac{w_3 - q_3(v_1, v_2)}{l_3(v_1, v_2)}, \ldots, v_n = \dfrac{w_n - q_n(v_1, \ldots, v_{n-1})}{l_n(v_1, \ldots, v_{n-1})}$.

■R(S)SE (Random (Singular) Simultaneous Equations) [KS04, KS05a]　中間変数ベクトル $\boldsymbol{v}$, $\boldsymbol{w}$ のそれぞれの次元 n, m について $n = m$ とする．また，整数 $t \geq 2$, $N \geq 2$ に対し，$n = Nt$ をみたすものとする．

$\boldsymbol{v}$, $\boldsymbol{w}$ をそれぞれ N 個の t 次元ベクトルに分割したものを，それぞれ $\boldsymbol{v}_i$, $\boldsymbol{w}_i$ と表す．すなわち $\boldsymbol{v}_i = (v_{(i-1)t+1}, \ldots, v_{it})^T$, $\boldsymbol{w}_i = (w_{(i-1)t+1}, \ldots, w_{it})^T$ である．また $1 \leq i < j \leq N$ に対し $\boldsymbol{v}_{i,\ldots,j} = (v_{(i-1)t+1}, \ldots, v_{jt})^T$, $\boldsymbol{w}_{i,\ldots,j} = (w_{(i-1)t+1}, \ldots, w_{jt})^T$ とする．

F_i を逆変換が一意な（非特異な）非線形変換とし，Ψ_i を（必ずしも逆変換が一意とは限らない）非線形変換とする．

RSE における $G = (G_1, \ldots, G_N) : \boldsymbol{v} \mapsto \boldsymbol{w}$ は以下のように表される：

$$
\begin{aligned}
\boldsymbol{w}_1 &= G_1(v_1, \ldots, v_n) = F_1(\boldsymbol{v}_1) \\
\boldsymbol{w}_2 &= G_2(v_1, \ldots, v_n) = F_2(\boldsymbol{v}_2) + \Psi_2(\boldsymbol{v}_1) \\
\boldsymbol{w}_3 &= G_3(v_1, \ldots, v_n) = F_3(\boldsymbol{v}_3) + \Psi_3(\boldsymbol{v}_{1,2}) \\
&\vdots \\
\boldsymbol{w}_N &= G_N(v_1, \ldots, v_n) = F_N(\boldsymbol{v}_N) + \Psi_N(\boldsymbol{v}_{1,\ldots,N-1})
\end{aligned}
$$

RSSE における $G = (G_1, \ldots, G_N)$ は以下のように表される：

$$\begin{aligned}
\boldsymbol{w}_1 &= G_1(v_1, \ldots, v_n) = \Psi_1(\boldsymbol{v}_1) \\
\boldsymbol{w}_2 &= G_2(v_1, \ldots, v_n) = \Psi_2(\boldsymbol{v}_2) \\
\boldsymbol{w}_3 &= G_3(v_1, \ldots, v_n) = \Psi_3(\boldsymbol{v}_3) \\
&\vdots \\
\boldsymbol{w}_N &= G_N(v_1, \ldots, v_n) = \Psi_N(\boldsymbol{v}_N)
\end{aligned}$$

RSE における G^{-1} は以下のように計算される：

1. $\boldsymbol{v}_1 = F_1^{-1}(\boldsymbol{w}_1)$.
2. $\boldsymbol{v}_2 = F_2^{-1}(\boldsymbol{w}_2 - \Psi_2(\boldsymbol{v}_1)), \ldots, \boldsymbol{v}_N = F_N^{-1}(\boldsymbol{w}_N - \Psi_N(\boldsymbol{v}_{1,\ldots,N-1}))$.

RSSE における G^{-1} の計算は，各 i について $\boldsymbol{w}_i = \Psi_i(\boldsymbol{v}_i)$ となるような $\boldsymbol{v}_i$ の候補をそれぞれ求めることにより行う．

3.2.3 UOV 型

UOV 型とは，oil 変数と vinegar 変数による双線形形式を非線形変換 G として用いるものである．

■UOV (Unbalanced Oil and Vinegar) [KPG99]　中間変数ベクトル $\boldsymbol{v}$, $\boldsymbol{w}$ のそれぞれの次元 n, m について，任意の整数 $k \geq 1$ に対し $n = m + k$ であるものとする．

UOV における $G = (g_1, \ldots, g_m) : \boldsymbol{v} \mapsto \boldsymbol{w}$ は以下のように表される：

$$g_i = \sum_{\substack{1 \leq j \leq m \\ m+1 \leq l \leq m+k}} \gamma_{i,j,l} v_j v_l + \sum_{m+1 \leq j,l \leq m+k} \lambda_{i,j,l} v_j v_l + \sum_{1 \leq j \leq m} \xi_{i,j} v_j + \sum_{m+1 \leq j \leq m+k} \eta_{i,j} v_j + \delta_i.$$

これらの方程式の係数について，$\gamma_{i,j,l}$, $\lambda_{i,j,l}$, $\xi_{i,j}$, $\eta_{i,j}$, $\delta_i \in_U \mathbf{F}_q$ である．ここで $v_1, \ldots, v_m$ を oil 変数，$v_{m+1}, \ldots, v_{m+k}$ を vinegar 変数と呼ぶ．g_i には oil 変数同士の積，すなわち $1 \leq j, l \leq m$ に対する $v_j v_l$ の項が含まれていない．

UOV における G の逆変換 G^{-1} を計算するためには，g_i における vinegar 変数 $v_{m+1}, \ldots, v_{m+k}$ に値を与えたもの $\overline{g}_i(v_1, \ldots, v_m)$ について以下の線形連立方程式を $v_1, \ldots, v_m$ について解く：

$$\begin{cases}
\overline{g}_1(v_1, \ldots, v_m) &= w_1 \\
& \vdots \\
\overline{g}_m(v_1, \ldots, v_m) &= w_m
\end{cases}$$

UOV において，非線形変換 G はランダムに生成される．このため，秘密鍵 L_1 は非線形変換 G に吸収される．それゆえ，UOV において L_1 を考えないのが一般的である．

UOV のパラメータとしては，以下が提案されている：

- $q = 2$, $n = 192$, $m = 64$, $k = 128$.
- $q = 16$, $n = 48$, $m = 16$, $k = 32$.

3.3 多変数 2 次公開鍵暗号に対する Modifier

多変数 2 次公開鍵暗号に用いられる非線形変換 G は，その構造を攻撃に利用されやすい．そこで，こうした構造を破壊するように，多変数 2 次公開鍵暗号に対する Modifier が提案されている．

以下では，多変数 2 次公開鍵暗号に対する Modifier として提案されている，主要な手法について説明する．Modifier を適用した，非線形変換 G や公開鍵多項式による変換 E などを，チルダを付けて，それぞれ $\widetilde{G}$, $\widetilde{E}$ といったように表記する．

■Minus method [Sha93, Pat96a, PGC98, PCG01, CGP03]　Minus method とは，m 個の公開鍵多項式のうち, r 個 $(1 \leq r < m)$ を公開せず，残りの $(m-r)$ 個のみを公開する Modifier である（図 5）．

$$\left.\begin{array}{rcl} \widetilde{e}_1(x_1,\ldots,x_n) & = & e_1(x_1,\ldots,x_n) \\ \vdots & & \\ \widetilde{e}_{m-r}(x_1,\ldots,x_n) & = & e_{m-r}(x_1,\ldots,x_n) \end{array}\right\} \text{公開}$$
$$\left.\begin{array}{c} e_{m-r+1}(x_1,\ldots,x_n) \\ \vdots \\ e_m(x_1,\ldots,x_n) \end{array}\right\} \text{秘密}$$

図 5　Minus method

この r について，秘匿目的には小さな r しか使えない．なぜなら，暗号文 $\boldsymbol{c} = (c_1,\ldots,c_{m-r})^T$ から平文 $\boldsymbol{p} = (p_1,\ldots,p_n)^T$ を復号するためには, 欠落した r 個の要素 $c_{m-r+1},\ldots,c_m$ について, q^r 通りの計算を行わなければならないためである．

一方で, 署名目的なら, この r について十分大きな値でも差し支えない. 特に, 攻撃者が文書 $\boldsymbol{c}$ から署名 $\boldsymbol{p}$ を偽造するのを手こずらせるためには, $r = 1, 2$ あるいは $r = m/2$ 程度が効果的とされている．

Minus method を適用した多変数 2 次公開鍵暗号としては，MI を原方式とする SFLASH などが提案されている．これらの方式のパラメータは以下の通りである．

- C^{*--} (MI minus): $q = 128 = 2^7$, $n = 67$, $r = 11$, $\theta = 33$ (SFLASH$^{\mathrm{v3}}$)
- HFE$^-$: $q = 16$, $n = 36$, $d = 4352$, $r = 4$ (HFE Challenge 2)

■Plus method [Pat96a, PGC98, DG06]　Plus method とは，m 個の公開鍵多項式と h 個のランダムな多項式について，これらの $(m+h)$ 個の多項式を線形変換したものを新たに公開鍵多項式とする Modifier である（図 6）．

$$\widetilde{E}(\boldsymbol{x}) = \widetilde{L}_1(e_1, \ldots, e_m, \varepsilon_1, \ldots, \varepsilon_h) \longleftarrow \left\{ \begin{array}{c} e_1(x_1, \ldots, x_n) \\ \vdots \\ e_m(x_1, \ldots, x_n) \\ \left.\begin{array}{c} \varepsilon_1(x_1, \ldots, x_n) \\ \vdots \\ \varepsilon_h(x_1, \ldots, x_n) \end{array}\right\} \text{ランダム多項式} \end{array} \right.$$

図 6　Plus method

この h について，署名目的には小さな h しか使えない．なぜなら，文書 $\boldsymbol{c} = (c_1, \ldots, c_{m+h})^T$ から生成された署名 $\boldsymbol{p} = (p_1, \ldots, p_n)^T$ に対して，$\boldsymbol{c} = \widetilde{E}(\boldsymbol{p})$ をみたす確率は $1/q^h$ であるためである．

■Vinegar 変数 [Pat96a, KPG99, PCG01]　Vinegar 変数とは，非線形変換 $G : \boldsymbol{v} \mapsto \boldsymbol{w}$ に導入する，中間変数 $\boldsymbol{v}$, $\boldsymbol{w}$ と独立な変数である．

MI 型の MPKC における非線形変換 $F : V \mapsto W$ が以下のような形をなすものとする：

$$F(V) = \sum_{0 \le i,j \le n} \beta_{i,j} V^{q^i + q^j} + \sum_{0 \le l \le n} \alpha_l V^{q^l} + \mu_0.$$

F に k 個の Vinegar 変数 $\boldsymbol{z} = (z_1, \ldots, z_k)^T$ を導入した非線形変換 $\widetilde{F}(V)$ は以下のよう

に表される：

$$\widetilde{F}(V) = \sum_{0\le i,j\le n} \beta_{i,j} V^{q^i+q^j} + \sum_{0\le l\le n} \eta_l(\boldsymbol{z}) V^{q^l} + \tau_0(\boldsymbol{z}),$$
$$\eta_l(\boldsymbol{z}) = \sum_{1\le i\le k} z_i \lambda_{i,l} + \alpha_l,$$
$$\tau_0(\boldsymbol{z}) = \sum_{1\le i\le j\le k} z_i z_j \varphi_{i,j} + \sum_{1\le i\le k} z_i \sigma_i + \mu_0.$$

ここで $\lambda_{i,l}, \varphi_{i,j}, \sigma_i \in_U \mathbf{F}_{q^n}$ である.

一方，MPKC における非線形変換 $G = (g_1, \ldots, g_m) : \boldsymbol{v} \mapsto \boldsymbol{w}$ が以下のような形をなすものとする：

$$g_1(\boldsymbol{v}) = \sum_{1\le j\le l\le n} \alpha_{1,j,l} v_j v_l + \sum_{1\le j\le n} \beta_{1,j} v_j + \gamma_1,$$
$$\vdots$$
$$g_i(\boldsymbol{v}) = \sum_{1\le j\le l\le n} \alpha_{i,j,l} v_j v_l + \sum_{1\le j\le n} \beta_{i,j} v_j + \gamma_i,$$
$$\vdots$$
$$g_m(\boldsymbol{v}) = \sum_{1\le j\le l\le n} \alpha_{m,j,l} v_j v_l + \sum_{1\le j\le n} \beta_{m,j} v_j + \gamma_m.$$

ここに $\alpha_{i,j,l}, \beta_{i,j}, \gamma_i \in_U \mathbf{F}_q$ である.

G に k 個の Vinegar 変数 $\boldsymbol{z} = (z_1, \ldots, z_k)^T$ を導入した非線形変換 $\widetilde{G}(\boldsymbol{v}) = (\widetilde{g}_1, \ldots, \widetilde{g}_m)$ は以下のように表される：

$$\widetilde{g}_1(\boldsymbol{v}) = \sum_{1\le j\le l\le n} \alpha_{1,j,l} v_j v_l + \sum_{1\le j\le n} \eta_{1,j}(\boldsymbol{z}) v_j + \tau_1(\boldsymbol{z}),$$
$$\vdots$$
$$\widetilde{g}_i(\boldsymbol{v}) = \sum_{1\le j\le l\le n} \alpha_{i,j,l} v_j v_l + \sum_{1\le j\le n} \eta_{i,j}(\boldsymbol{z}) v_j + \tau_i(\boldsymbol{z}),$$
$$\vdots$$
$$\widetilde{g}_m(\boldsymbol{v}) = \sum_{1\le j\le l\le n} \alpha_{m,j,l} v_j v_l + \sum_{1\le j\le n} \eta_{m,j}(\boldsymbol{z}) v_j + \tau_m(\boldsymbol{z}),$$
$$\eta_{i,j}(\boldsymbol{z}) = \sum_{1\le l\le k} \lambda_{i,j,l} z_l + \beta_{i,j}, \quad \tau_i(\boldsymbol{z}) = \sum_{1\le j\le l\le k} \varphi_{i,j,l} z_j z_l + \sum_{1\le j\le k} \sigma_{i,j} z_j + \gamma_i.$$

ここで $\lambda_{i,l}, \varphi_{i,j}, \sigma_i \in_U \mathbf{F}_q$ である．

$\widetilde{G}$ の逆変換は，$\boldsymbol{z}$ に値を与えたものに対し，G^{-1} を計算することにより行う．

Vinegar 変数を導入した多変数 2 次公開鍵暗号としては，HFE を原方式とする QUARTZ が提案されている [PCG01]．QUARTZ には Vinegar 変数に加えて，Minus method が Modifier として適用されている．QUARTZ のパラメータは以下の通りである．

- $q = 2$, $n = 103$, $k = 4$, $r = 3$, $d = 129$.

■Internal Perturbation [Din04, DS05a, DG06]　Internal Perturbation とは，非線形変換 G に中間変数 $\boldsymbol{v}$ に対するランダムな変換（摂動多項式）を加える Modifier である（図 7）．

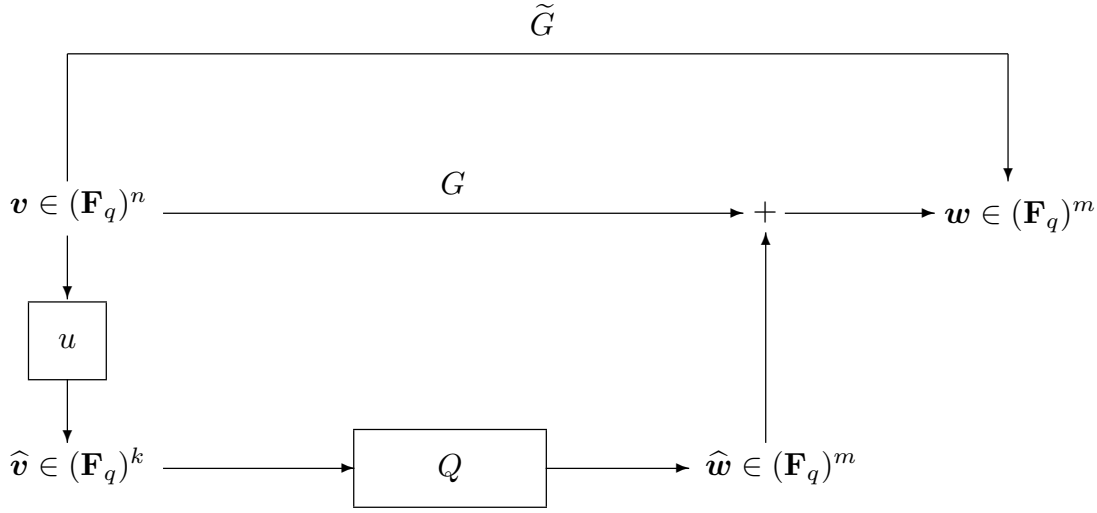

図 7　Internal Perturbation

線形変換 $u : (\mathbf{F}_q)^n \to (\mathbf{F}_q)^k$，非線形変換 $Q = (q_1, \ldots, q_m) : (\mathbf{F}_q)^k \to (\mathbf{F}_q)^m$ に対し，非線形変換 G に Internal Perturbation を適用した変換 $\widetilde{G}$ は $\widetilde{G}(\boldsymbol{v}) = (G + (Q \circ u))(\boldsymbol{v})$ と表される．

$\widetilde{G}^{-1}$ の計算は，すべての $\widehat{\boldsymbol{v}} = u(\boldsymbol{v})$ に対し，$G^{-1}(\boldsymbol{w} - Q(\widehat{\boldsymbol{v}})) = \boldsymbol{v}$ であるかどうか検査するため G^{-1} を q^k 回繰り返す必要がある．

Internal Perturbation を適用した多変数 2 次公開鍵暗号について，以下のパラメータが提案されている．なお，PMI+ とは，MI に Internal Perturbation と Plus method を適用したものである．

- PMI: $q = 2$, $n = 96$, $k = 5$, $h = 0$.
- PMI: $q = 2$, $n = 136$, $\theta = 40$, $k = 6$, $h = 0$.
- PMI+: $q = 2$, $n = 84$, $\theta = 4$, $k = 6$, $h = 14$.
- PMI+: $q = 2$, $n = 136$, $\theta = 8$, $k = 6$, $h = 18$.
- IPHFE: $q = 2$, $n = 89$, $d = 9$, $k = 2$.

4 代数攻撃

多変数 2 次公開鍵暗号の公開鍵

$$E(\boldsymbol{x}) = (e_1(\boldsymbol{x}), \ldots, e_m(\boldsymbol{x}))^T \in (\mathbf{F}_q[x_1, \ldots, x_n])^m$$

と暗号文 $\boldsymbol{c} = (c_1, c_2, \ldots, c_m)^T \in (\mathbf{F}_q)^m$ を用いて，以下の非線形連立方程式

$$\begin{cases} e_1(x_1, \ldots, x_n) = c_1 \\ e_2(x_1, \ldots, x_n) = c_2 \\ \vdots \\ e_m(x_1, \ldots, x_n) = c_m \end{cases} \tag{4.1}$$

を $x_1, \ldots, x_n$ について解くことにより，攻撃者が暗号文に対応する平文を得る攻撃を，多変数 2 次公開鍵暗号に対する代数攻撃と呼ぶ.

4.1 代入攻撃

$n > m$ の場合，一般に，方程式 (4.1) の解の個数は q^{n-m} であり，特に $n \gg m$ の場合，これらの解をすべて求めるのは困難である.

方程式 (4.1) について，少なくとも 1 つの解を求めればよい場合，e_i における n 個の変数のうち $(n - m + u)$ 個の変数に値を代入したもの $\overline{e}_i$ について，以下の非線形連立方程式

$$\begin{cases} \overline{e}_1(x_1, \ldots, x_{m-u}) = c_1 \\ \overline{e}_2(x_1, \ldots, x_{m-u}) = c_2 \\ \vdots \\ \overline{e}_m(x_1, \ldots, x_{m-u}) = c_m \end{cases} \tag{4.2}$$

を解く方がより効率的である．このような計算によって行う代数攻撃を代入攻撃と呼ぶ.

$u = 0$ の代入攻撃の場合，方程式 (4.2) の解を必ず得られると限らないものの，解を得られる確率は $(1 - 1/e)$（e は自然対数）であり，平均して 1.6 回に 1 回の割合で解を得

ることができる．一方，$u > 0$ の代入攻撃の場合，$u = 0$ の場合よりも余分に代入した q^u 個の値に相当する回数分，方程式 (4.2) を解く必要がある．

4.2 グレブナ基底攻撃

方程式 (4.1) をグレブナ基底計算を用いて解く代数攻撃は，グレブナ基底攻撃と呼ばれている．

方程式 (4.1) に対するグレブナ基底計算のための計算量は，計算の際の中間基底の最大次数 d_{reg} に対し，$\mathcal{O}\left(\left(m\binom{n+d_{\mathrm{reg}}-1}{d_{\mathrm{reg}}}\right)^w\right)$ である [BFS04, BFSY05]．ここに $2 \leq w \leq 3$ は線形代数の計算コストである．$m = 16, 20$ に対する d_{reg} の理論値および実測値を以下に示す [FP08, BFP08]．

m	n	d_{reg}（理論値）	d_{reg}（実測値）
16	15	9	9
16	14	7	7
16	13	6	6

m	n	d_{reg}（理論値）	d_{reg}（実測値）
20	19	11	
20	18	9	9
20	17	8	8
20	16	7	7
20	15	6	6

4.2.1 HFE に対するグレブナ基底攻撃 [Pat96a, CDF03, FJ03, GJS06]

HFE の提案時，グレブナ基底攻撃のための計算量は理論的に $\mathcal{O}(D^{3n})$ あるいは，実験的に $\mathcal{O}(D^{2.7n})$ と考えられていた．ここに D は公開鍵多項式の次数であり，HFE の場合 $D = 2$ である．それゆえ，この攻撃計算量が 2^{64} 以上となるためには，$n \geq 23$ が必要と考えられていた．

その後の研究により，$d \leq 512$ の HFE に対し，グレブナ基底攻撃によって，以下の計算量により暗号解読可能とされている．

- $4 \leq d \leq 16$ の場合 $\mathcal{O}(n^6)$
- $17 \leq d \leq 128$ の場合 $\mathcal{O}(n^8)$
- $129 \leq d \leq 512$ の場合 $\mathcal{O}(n^{10})$

実際に，$n = 80, d = 96$ の HFE について，約 2 日かけてグレブナ基底攻撃に成功したという結果が示されている[*5]．

[*5] 文献 [FJ03] では，1GHz の CPU を搭載した Sunfire v880 と，F_5 アルゴリズムを用いて，およそ 52.2 時間かけて解読に成功したことが報告されている．一方 [Ste04] では，750 MHz の CPU を搭載したコンピュータを用いて，Magma V2.11-9 に実装された F_4 アルゴリズムと，HFE 解読のためのオプションを利用することにより，およそ 22.1 時間かけて解いたという結果が示されている．

この計算量は，ヒューリスティックな議論により $\mathcal{O}(n^{\mathcal{O}(\log d)})$ と見積もられている．特に，d がある定数 α に対して $d = \mathcal{O}(n^{\alpha})$ である場合，この計算量は $2^{\mathcal{O}(\log n)^2}$ となり，n に対する準指数時間計算量となる．しかしながら，この計算量は，いわゆる準指数時間計算量よりもずっと小さいため，準多項式時間 (quasipolynomial time) 計算量と言われている．

一方，Modifier（Vinegar 変数，Minus method）を適用した，$q = 2$ の HFE については，平文変数の数が n であり，k 個の Vinegar 変数を導入し，r 個の公開鍵多項式を公開しない Minus method を適用した場合，$u = 0$ の代入グレブナ基底攻撃のための計算量は $q^{k+r} \cdot (n-r)^{10}/4$ と推測されている．たとえば $n = 103$, $k = 4$, $r = 3$ の場合，この計算量は 2^{71} 程度と見積もられていた．しかしながら，その後の研究により，この計算量は 2^{62} と推測されている．

4.3 代数攻撃のためのアルゴリズム

XL アルゴリズムや，Magma のグレブナ基底計算アルゴリズム以外にも，代数攻撃のために，さまざまなアルゴリズムが提案されている．

■GeometricXL [MP08] XL アルゴリズムにおける計算過程を，代数幾何学の観点から解釈した，GeometricXL と呼ばれるアルゴリズムが提案されている．

■HXL（“Heuristic and Hybrid” XL）アルゴリズム [GT08a, GT08b] XL アルゴリズムにおいて生成される多項式集合について，互いに線形従属な多項式が生成されにくくなるように，アルゴリズムの改善を行っている．Magma のスクリプト言語を用いて実装した HXL アルゴリズムと，Magma V2.13 の F_4 アルゴリズムを用いて計算した場合との比較が行われており，HXL の方が，計算に使用するメモリ容量を，より少なくすることができるという結果が報告されている．

■MXL（Mutant XL）アルゴリズム [DBMMW08, MMDB08, MCDBB09] Mutant と呼ばれる，グレブナ基底の計算過程において，突然変異的に得られる多項式に着目して，XL アルゴリズムを改善したものである．MXL アルゴリズムの新しいバージョン（MXL_3）においては，Magma の F_4 アルゴリズムよりも，効率的に計算されうるという結果が示されている．

■Zhuang-Zi（荘子）アルゴリズム MI 型の多変数 2 次公開鍵暗号の構成に用いられているように，有限体上の多変数多項式は，一変数多項式として見ることができる．この性

質を利用し，主に，一変数多項式の因数分解を行うことにより，求解を行ってゆくアルゴリズムである [DGS06a]．いくつかの例においては，Magma の F_4 アルゴリズムを用いて解くのが困難であっても，Zhuang-Zi アルゴリズムを用いて解けることが確認されている．最近，Mutant の概念を導入した Zhuang-Zi アルゴリズムが提案されている [DS10].

■CS method　特に，幾何の定理の自動証明などの分野において，多変数非線形連立方程式を解くために，古くから用いられている手法として，CS（Characteristic Set）method（特性集合法）と呼ばれる方法がある [Rit50, Wu78] [*6]．この方法を，共通鍵ストリーム暗号の解読に利用するといった研究が行われており，グレブナ基底計算よりも効率的に計算され得ることが示されている [GH09].

■F_5 アルゴリズム [Fau02]　F_4 アルゴリズムにおいては，必ずしも，掃き出す行列がフルランクにならないという問題点がある．F_5 アルゴリズムでは，この行列が，必ず，フルランクとなるように，グレブナ基底の計算過程に関する履歴を利用して，冗長な部分を取り除いている．

■PET SNAKE [GMS09]　共通鍵ブロック暗号の解読を目的として，開発が進められている，代数攻撃用のハードウェアである．PET SNAKE は，非線形連立方程式を解くために，新たに提案された，MRHS（Multiple Right Hand Sides）と呼ばれる手法 [RS08] をベースとしている．ちなみに PET SNAKE とは，Parallel Elimination Technique Supporting Nice Algebraic Key Elimination を略した名称である．

■PolyBoRi [BD07]　特に，係数体，および，解の値が，いずれも $\mathbf{F}_2$ 上にあるような，非線形連立方程式の求解は，暗号分野のみならず，論理回路の設計など，さまざまな応用が考えられる．このような，ブール多項式環上の多項式（Polynomials over Boolean Rings）に関するグレブナ基底計算のためのパッケージを提供するプロジェクトが進められている．

5　多変数公開鍵暗号の安全性解析

多変数公開鍵暗号に対する代数攻撃の手法は，単に，暗号方式を攻撃するだけでなく，その安全性を評価するために用いられている．以下では，代数攻撃を利用して，多変数公開鍵暗号の安全性を解析した，いくつかの結果について述べる．

[*6] Wu の方法，Ritt-Wu の分解アルゴリズムなどと呼ばれることがある．

5.1 UOV に対する安全性解析 [BWP05, FP08]

Magma の F_4 アルゴリズムを用いて，UOV（3.2.3 節）に対する代入攻撃を行った結果から，$n = 3m, 4m$ の UOV に対し，$u = 0$ の代入グレブナ基底攻撃のための計算量が 2^{64} 以上となるためには，$q = 2$ の場合 $m \geq 38$，$q = 3$ の場合 $m \geq 24$ である必要があると推測されている [BWP05].

方程式 (4.1) において，$m = 16, n = 14$ の場合，$d_{\mathrm{reg}} = 7$ であり，グレブナ基底計算の計算量は高々 $2^{52.7}$ となる．一方で，$q = 16, m = 16, n = 32, 48$ の UOV に対し，F_5 アルゴリズムを用いた $u = 2$ の代入グレブナ基底攻撃のための計算量は，この値よりもずっと小さく，$2^{32.3}$ となった実験結果が報告されている [FP08].

5.2 PMI に対する安全性解析

Magma の F_4 アルゴリズムを用いて，PMI（3.3 節）に対するグレブナ基底攻撃を行った結果から，$q = 2$ の PMI について，提案者によって，十分安全と考えられていたパラメータである，$k = 5, n = 96$ の場合であっても，グレブナ基底攻撃のための計算量が 2^{80} を大きく下回ると推測されている．一方，$k = 6$ の場合，$n \geq 83$ であれば，グレブナ基底攻撃の計算量が 2^{80} を上回ると推測されている [DGSWY05].

$k = 6$ の場合，PMI に対するグレブナ基底攻撃の計算量が指数時間となる可能性が示唆されている．しかしながら，計算機実験環境 A（1.2 節）を使用して，実験を行ってみたところ，$k \leq 10, 24 \leq n \leq 26$ の場合，PMI に対するグレブナ基底攻撃と，グレブナ基底計算によるランダムな連立 2 次方程式の求解のための計算量が同等とならないという結果が得られている（表 3）.

表 3　PMI に対するグレブナ基底攻撃のための計算時間（秒）

		$n = 22$	$n = 23$	$n = 24$	$n = 25$	$n = 26$
PMI	$k = 9$	56	102	160	303	501
	$k = 10$	57	90	156	403	515
	$k = 11$	57	89	596	1225	2597
	$k = 12$	57	89	604	1231	2592
random		54	85	577	1185	2516

5.3 持駒方式の安全性解析

任意の多変数公開鍵暗号系の安全性を強化する概念として，持駒概念 [Tsu03] が提案されている．この概念を具現化したものである，持駒方式と呼ばれる安全性強化手法が，これまでに，いくつか提案されている．持駒方式の具体的な構成方法については，文献 [TTF07a, TTF08, FTT08c] などを参照されたい．

以下では，これらの持駒方式の安全性について，グレブナ基底計算を用いた代数攻撃に対する安全性に関する結果について述べる．下記の計算機実験においては，計算機実験環境 A（1.2 節）を使用した．なお，HFE の解読に使われた HFE オプションなどのような，グレブナ基底計算に関する Magma のオプションについては一切使用していない．

5.3.1 線形持駒行列方式 [TTF07a]

HFE（3.2.1 節）に乱数変数を付加した線形持駒行列方式を適用することにより，グレブナ基底攻撃に対する安全性が強化されることが，計算機実験により示されている（表 4）．特に平文変数の数を一定（10）とした場合，原方式である HFE と比較して，持駒方式において，乱数変数を導入することにより，変数の総数 $z = 39$ とした方が，計算時間が約 10^6 倍になることが計算機実験から明らかとなった．また，復号時間が一定となるように，n を一定（20）とした場合，原方式である HFE と比較して，持駒方式において変数の総数 $z = 39$ の方が，計算時間が 740 倍以上になることが計算機実験から明らかとなった．

表 4　グレブナ基底攻撃のための計算時間の比較（線形持駒行列方式）

方式	パラメータ				計算時間
	p	n	z	g	(sec.)
HFE（$q = 2$, $128 < d < 513$）		10			$< 10^{-3}$
		20			8
		25			184
		28			959

方式	パラメータ				計算時間
	p	n	z	g	(sec.)
線形持駒行列方式（原方式：HFE（$q = 2$, $128 < d < 513$））	10	20	35	25	1000
	10	20	37	25	2424
	10	20	38	25	5288
	10	20	32	28	665
	10	20	36	28	2290
	10	20	38	28	4460
	10	20	39	28	5963

p : 持駒方式における平文変数の数，n : 原方式における平文変数の数

z : 持駒方式における変数（平文変数，乱数変数）の総数，g : 持駒方式における公開鍵多項式の数

5.3.2　非線形持駒行列方式 [TTF08]

MI（3.2.1 節），RSE（3.2.2 節）に，非線形持駒行列方式を適用することにより，線形持駒行列方式よりもグレブナ基底攻撃に対する安全性が強化されることが，計算機実験により示されている（表 5）．表 5 から，非線形持駒行列方式の方が，線形持駒行列方式と比較して，計算時間が約 10 倍から 100 倍大きくなっていることがわかる．また，平文変数の数を一定（25）とした場合，原方式である MI，RSE と比較して持駒方式において変数の総数 $z = 52$ の方が，計算時間が約 10^4 倍になることが計算機実験から明らかとなった．

表 5　グレブナ基底攻撃のための計算時間の比較（持駒行列方式）

方式	パラメータ				計算時間 (sec.)
	p	n	z	g	
MI（$q = 2$）		15			$< 10^{-2}$
		20			0.01
		25			0.03
		30			0.07
		35			0.2
		40			0.4
		45			0.7
		50			1
		55			2
		60			4

方式	パラメータ				計算時間 (sec.)	
	p	n	z	g	線形	非線形
持駒行列方式（原方式：MI（$q = 2$））	25	35	50	47	3	52
	25	35	51	47	6	260
	25	35	52	47	22	1307
	25	35	54	47	58	n/a
	25	35	56	47	829	n/a
	30	40	54	50	3	59
	30	40	55	50	5	263
	30	40	56	50	7	1281
	30	40	58	50	47	n/a
	30	40	60	50	1016	n/a

方式	パラメータ				計算時間 (sec.)
	p	n	z	g	
RSE（$q = 2$）		15			0.01
		20			0.03
		25			0.08
		30			0.2
		35			0.5
		40			1
		45			2
		50			5
		55			9
		60			16

方式	パラメータ				計算時間 (sec.)	
	p	n	z	g	線形	非線形
持駒行列方式（原方式：RSE（$q = 2$））	25	35	50	47	6	50
	25	35	51	47	13	250
	25	35	52	47	19	1309
	25	35	54	47	131	n/a
	25	35	56	47	1622	n/a
	30	40	54	50	8	58
	30	40	55	50	11	264
	30	40	56	50	28	1285
	30	40	58	50	158	n/a
	30	40	60	50	1770	n/a

p : 持駒方式における平文変数の数，n : 原方式における平文変数の数
z : 持駒方式における変数（平文変数，乱数変数）の総数，g : 持駒方式における公開鍵多項式の数
n/a は計算不可を示す．

5.3.3　非線形持駒摂動ベクトル方式 [FTT08c]

MI に非線形持駒摂動ベクトル方式を適用することにより，グレブナ基底攻撃に対する安全性が Internal Perturbation（3.3 節）と同等に強化されることが計算機実験により示されている（表 6，表 7）.

乱数変数を付加しない非線形持駒摂動ベクトル方式について，特に，平文変数の数を一定 ($n = 30$) とした場合，原方式である MI と比較して，持駒方式は PMI+ と同様に計算時間が約 10^4 倍になることが計算機実験から明らかとなった.

一方，乱数変数を付加した非線形持駒摂動ベクトル方式について，平文変数の数を一定（15）とした場合，原方式である MI，RSE と比較して，持駒方式において変数の総数 z と公開鍵多項式の数 g がそれぞれ $z = 47$, $g = 35$（原方式：MI）$z = 44$, $g = 35$（原方式：RSE）の方が，計算時間が約 10^5 倍になることが計算機実験から明らかとなった（表 8）.

表 6　PMI+（$q = 2$）に対するグレブナ基底攻撃のための計算時間

パラメータ			計算時間
n	k	h	(sec.)
28	6	0	845
28	6	5	733
28	6	10	563
28	6	15	436
29	6	15	747
30	6	15	1305

n: 平文変数の数
k: perturbation dimension
h: Plus 多項式の数

表 7　非線形持駒摂動ベクトル方式を適用した MI（$q = 2$）に対するグレブナ基底攻撃のための計算時間

パラメータ			計算時間
n	l	h	(sec.)
28	17	3	290
28	17	4	289
28	17	5	263
29	17	3	537
29	17	8	402
29	17	10	349
30	17	3	936
30	17	8	701
30	17	13	513

n: 平文変数の数
l: 補助方式における変数の数
h: ランダム項ベクトルの次元

なお，非線形持駒摂動ベクトル方式の計算機実験において，補助方式として，HFE の公開鍵多項式による非線形変換を用いたが，補助方式として，どのような方式が最適であるかについては未解決問題である.

表 8　グレブナ基底攻撃のための計算時間の比較（非線形持駒摂動ベクトル方式）

方式	パラメータ				計算時間
	p	n	z	g	(sec.)
MI（$q=2$）		15			$< 10^{-2}$
		20			0.01
		25			0.03
		30			0.07
		35			0.2
		40			0.4
		45			0.7
		50			1
		55			2
		60			4

方式	パラメータ				計算時間
	p	n	z	g	(sec.)
非線形持駒摂動ベクトル方式（原方式：MI（$q=2$））	15	20	40	35	75
	15	20	43	35	129
	15	20	45	35	260
	15	20	46	35	320
	15	20	47	35	1029
	15	20	40	40	97
	15	20	43	40	161
	15	20	47	40	284
	15	20	48	40	495
	15	20	49	40	1077

方式	パラメータ				計算時間
	p	n	z	g	(sec.)
RSE（$q=2$）		15			0.01
		20			0.03
		25			0.08
		30			0.2
		35			0.5
		40			1
		45			2
		50			5
		55			9
		60			16

方式	パラメータ				計算時間
	p	n	z	g	(sec.)
非線形持駒摂動ベクトル方式（原方式：RSE（$q=2$））	15	20	40	35	40
	15	20	41	35	71
	15	20	42	35	179
	15	20	43	35	713
	15	20	44	35	2791
	15	20	40	40	51
	15	20	42	40	82
	15	20	44	40	231
	15	20	45	40	877
	15	20	46	40	2327

p : 持駒方式における平文変数の数，n : 原方式における平文変数の数

z : 持駒方式における変数（平文変数，乱数変数）の総数，g : 持駒方式における公開鍵多項式の数

5.4　rSTS 型多変数公開鍵暗号の安全性解析

5.4.1　rSTS 型多変数公開鍵暗号 [WBP04, WBP06]

多変数 2 次公開鍵暗号の秘密鍵 $G(\boldsymbol{v}) = (g_1(\boldsymbol{v}), \ldots, g_n(\boldsymbol{v}))$ が図 8 のような形（gSTS: general STS あるいは単に STS）をなすものを，STS 型多変数公開鍵暗号と呼ぶ（以下では，STS 型 MPKC と略記する）．ここに，$n = r_1 + \cdots + r_L$, $m = m_1 + \cdots + m_L$ である．

STS 型多変数公開鍵暗号のうち，特に $r_1 = r_2 = \cdots = r_L = m_1 = m_2 = \cdots = m_L (= r)$ であるものを rSTS 型多変数公開鍵暗号と呼ぶ（以下では，rSTS 型 MPKC と略記

$$
\begin{array}{l}
\text{Step } 1 \left\{ \begin{array}{c} g_1(v_1, \ldots, v_{r_1}) \\ \vdots \\ g_{m_1}(v_1, \ldots, v_{r_1}) \end{array} \right. \\
\qquad\qquad \vdots \\
\text{Step } l \left\{ \begin{array}{l} g_{m_1+\cdots+m_{l-1}+1}(v_1, \ldots, v_{r_1}, \ldots, v_{r_1+\cdots+r_{l-1}+1}, \ldots, v_{r_1+\cdots+r_l}) \\ \vdots \\ g_{m_1+\cdots+m_l}(v_1, \ldots, v_{r_1}, \ldots, v_{r_1+\cdots+r_{l-1}+1}, \ldots, v_{r_1+\cdots+r_l}) \end{array} \right. \\
\qquad\qquad \vdots \\
\text{Step } L \left\{ \begin{array}{c} g_{m_1+\cdots+m_{L-1}+1}(v_1, \ldots, v_n) \\ \vdots \\ g_m(v_1, \ldots, v_n) \end{array} \right.
\end{array}
$$

図 8　STS 型多変数公開鍵暗号における秘密鍵 G

する）．

rSTS 型 MPKC としては，これまでに，$r = 1$ の場合である，順序解法を用いた方式，r が任意の場合である，R(S)SE（いずれの方式も 3.2.2 節）が提案されている．

5.4.2　rSTS 型多変数公開鍵暗号の安全性

$r = 1, 4, 10$ として，$q = 2$, $n = m = 40$ である rSTS 型多変数公開鍵暗号に対し，計算機実験環境 B（1.2 節）を使用して，グレブナ基底攻撃の計算機実験を行った．

実験方法としては，まず，各 r ごとに，10 個の公開鍵を生成した．次に，これらの公開鍵を用いて，暗号文をそれぞれ 100 対ずつ生成し，これらの暗号文に対応する平文をグレブナ基底攻撃によって求め，計算時間を計測した．求めたグレブナ基底における線形式の数と，計算時間の関係を図 9 に示す．図 9 の横軸である，グレブナ基底における線形式の数は，そのイデアルの零点における非線形性，すなわち，方程式 (4.1) の解空間の複雑さに影響する．一般に，この複雑さが，グレブナ基底計算の困難性に影響を及ぼすといわれている．また，図 9 においては，計算時間を表す縦軸を対数目盛としている．これは，グレブナ基底計算途中に生成される中間多項式の次数の増加に対し，計算時間が指数的に増加するためである．

図 9 より，rSTS 型 MPKC に対するグレブナ基底攻撃において，グレブナ基底における線形式の数と，計算時間との間に，強い相関がみられない．このため，この線形式の数が，計算時間の長短に及ぼす影響は大きくないと考えられる．また，rSTS 型 MPKC のパラメータを一定としても，公開鍵や暗号文によって，計算時間に大きなばらつきがあ

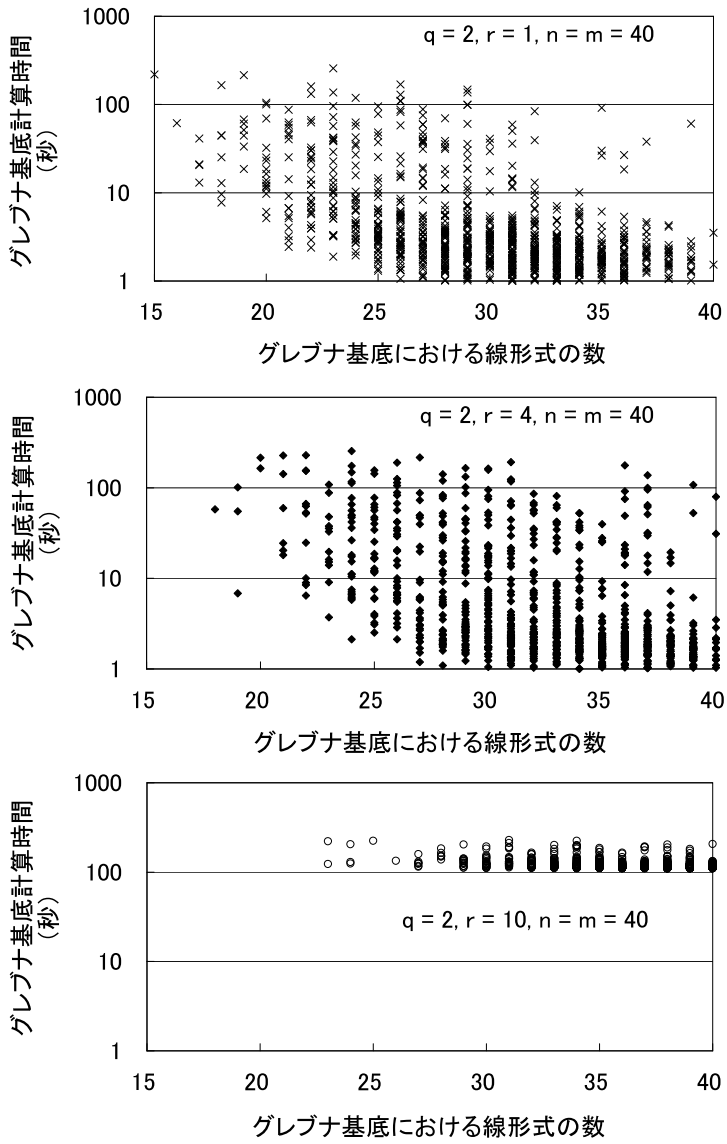

図 9　rSTS 型 MPKC に対するグレブナ基底攻撃の計算時間（鍵数 10 × 平文数 100）

り，特に，$r = 1, 4$ の場合，その差が数百倍程度に及んでいる．一方，$r = 10$ の場合，$r = 1, 4$ の場合と比較して，グレブナ基底攻撃の計算時間が全体的に増大するとともに，そのばらつきが小さくなることが，図 9 から明らかとなった．

次に，グレブナ基底攻撃における計算途中の中間多項式の次数の遷移を図 10 に示す．

図 10 より，r が小さくなるにつれて，グレブナ基底計算のステップ数が大きくなり，途中の中間多項式の次数の変動回数が多くなるという結果が得られた．こうした次数の変動が，グレブナ基底攻撃の計算時間のばらつきなどに影響を及ぼすと見られるが，この次

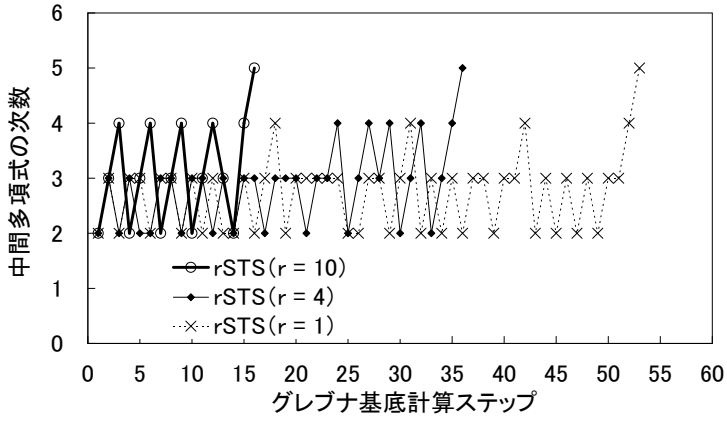

図 10　グレブナ基底計算途中の中間多項式の次数の遷移（rSTS 型 MPKC : $q = 2$, $n = m = 40$）

数の遷移に関する理論的な解析は，今後の研究課題である．

5.4.3　R(S)SE の安全性

$q = 2, n = m = 40$ とし，t あるいは r を $4, 10$ とした場合の RSE, RSSE（3.2.2 節）と，rSTS 型 MPKC に対するグレブナ基底攻撃の計算時間の比較を表 9 に示す．

表 9 では，各方式およびパラメータにおいて生成した 9 個の鍵に対し，それぞれ 100 通りの暗号文に対応する平文を，グレブナ基底攻撃によって求める計算時間の平均値について，それぞれの 9 個の鍵の間での最大値，最小値，中央値を示している．また，標準偏差についても，各鍵ごとに統計量を算出し，それらのうちの中央値を表 9 に示す．

表 9　グレブナ基底攻撃の計算時間の比較

$q = 2$ $n = m = 40$		計算時間（秒）			
	方式	最小	中央値	最大	標準偏差（中央値）
$t = 4$	RSE	0.049	0.426	0.541	0.031
	RSSE	0.306	0.336	0.459	0.039
$r = 4$	rSTS	6	14	27	25.508
$t = 10$	RSE	0.586	0.652	0.738	0.006
	RSSE	81	82	91	0.477
$r = 10$	rSTS	119	121	124	15.588

表 9 から，RSE, RSSE は，いずれも，rSTS 型 MPKC に分類されるものの，グレブナ基底攻撃の計算時間が同等になると限らないことが明らかとなった．

次に，RSE, RSSE に対するグレブナ基底攻撃における，計算途中の中間多項式の次数の遷移を図 11 に示す．

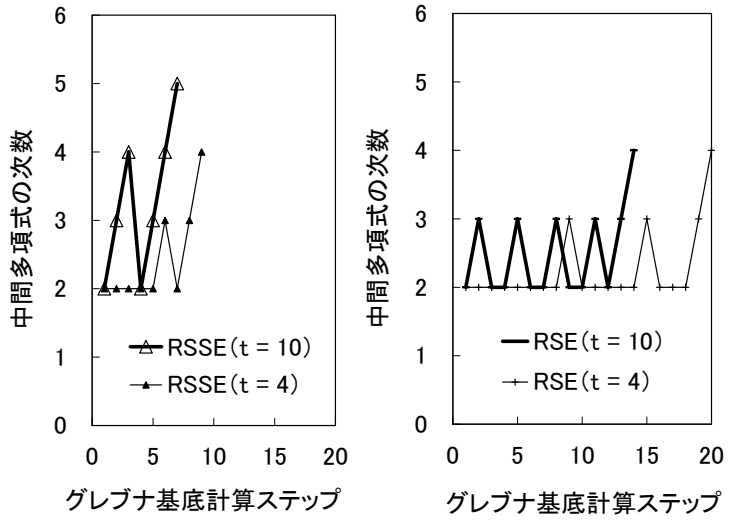

図 11　グレブナ基底計算途中の中間多項式の次数の遷移（RSE, RSSE : $q = 2$, $n = m = 40$）

図 10，図 11 より，表 9 における計算時間が小さいものは，いずれも，中間多項式の最大次数が小さいという結果が得られた．

グレブナ基底攻撃の際，計算途中の中間多項式について，その最大次数の理論的な見積もりについては，5.4 節に述べた，次数の遷移と同様に，今後の研究課題として残されている．

6　おわりに

多変数公開鍵暗号は，その数学的構造や構成方法，安全性に関して，まだ解明されていない部分が非常に多い．このため，既存の攻撃手法に対して，十分に安全であると考えられて，多変数公開鍵暗号が提案されても，その後まもなく，実用的な攻撃法が提案されるケースが多々ある．このような現状を特徴付けるように，[DFSS07] の結言において，Dubois らは以下のように述べている．

> Multivariate cryptographic schemes are very efficient but have a lot of exploitable mathematical structure. Their security is not fully understood, and

new attacks against them are found on a regular basis. It would thus be prudent not to use them in any security-critical applications.

多変数公開鍵暗号が耐量子コンピュータ公開鍵暗号として実用に供されるためには，今後多くの研究を積み重ねてゆく必要があると考える．

一方，グレブナ基底計算において，例えば，計算途中の中間多項式の次数の遷移など，アルゴリズムの動作について，いまだ十分に解明されていない部分が多い．また，計算アルゴリズムについても，どのような問題に対して，どのようなアルゴリズムを用いれば，より効率的に解くことができるのかといったことなど，いまだ明らかとなっていない点が多い．

以上，多変数公開鍵暗号と，その周辺分野における数学について，解くべき問題が山積している．これらの分野における，理論および応用面における研究の発展が，新しい暗号を生み出す礎となり，来るべき未来に備えるべく，情報セキュリティの基盤技術としての暗号の分野を切り開き，多大なる貢献をもたらすものと考えている．

参考文献

[AFIKS04] G. Ars, J. C. Faugère, H. Imai, M. Kawazoe, and M. Sugita, "Comparison between XL and Gröbner basis algorithms," *Proc. ASIACRYPT 2004*, Lecture Notes in Computer Science, vol.3329, pp.338–353, Springer, 2004.

[BFS04] M. Bardet, J. C. Faugère, and B. Salvy, "On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations," Proceedings of International Conference on Polynomial System Solving (ICPSS 2004), pp.71–75, Nov. 2004.

[BFSY05] M. Bardet, J. C. Faugère, B. Salvy, and B. Y. Yang, "Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems," Proceedings of MEGA 2005, May 2005.

[BBD09] D. J. Bernstein, J. Buchmann, and E. Dahmen (editors), Post-Quantum Cryptography, Springer, 2009.

[BFP08] L. Bettale, J. C. Faugère, and L. Perret, "Cryptanalysis of the TRMS signature scheme of PKC'05," *Proc. AFRICACRYPT 2008*, Lecture Notes in Computer Science, vol.5023, pp.143–155, Springer, 2008.

[BL95] D. Boneh and R. J. Lipton, "Quantum cryptanalysis of hidden linear func-

tions," *Proc. CRYPTO '95*, Lecture Notes in Computer Science, vol.963, pp.424–437, Springer, 1995.

[BWP05] A. Braeken, C. Wolf, and B. Preneel, "A study of the security of unbalanced oil and vinegar signature schemes," *Proc. CT-RSA 2005*, Lecture Notes in Computer Science, vol.3376, pp.29–43, Springer, 2005.

[BD07] M. Brickenstein and A. Dreyer, "POLYBORI: A Gröbner basis framework for Boolean polynomials," Reports of Fraunhofer ITWM, no.122, 2007.

[Buc65] B. Buchberger, "Ein Algorithmus zum auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal," PhD thesis, Innsbruck, 1965.

[CKM97] S. Collart, M. Kalkbrener, and D. Mall, "Converting bases with the Gröbner walk," Journal of Symbolic Computation, vol.24, no.3, pp.465–469, Sept. 1997.

[CKPS00] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations," *Proc. EUROCRYPT 2000*, Lecture Notes in Computer Science, vol.1807, pp.392–407, Springer, 2000.

[Cou01] N. Courtois, "The security of Hidden Field Equations (HFE)," *Proc. CT-RSA 2001*, Lecture Notes in Computer Science, vol.2020, pp.266–281, Springer, 2001.

[CGMT02] N. Courtois, L. Goubin, W. Meier, and J. D. Tacier, "Solving underdefined systems of multivariate quadratic equations," *Proc. PKC 2002*, Lecture Notes in Computer Science, vol.2274, pp.211–227, Springer, 2002.

[CP02] N. Courtois and J. Pieprzyk, "Cryptanalysis of block ciphers with overdefined systems of equations," *Proc. ASIACRYPT 2002*, Lecture Notes in Computer Science, vol.2501, pp.267–287, Springer, 2002.

[Cou02] N. Courtois, "Higher order correlation attacks, XL algorithm and cryptanalysis of Toyocrypt," *Proc. ICISC 2002*, Lecture Notes in Computer Science, vol.2587, pp.182–199, Springer, 2003.

[CDF03] N. Courtois, M. Daum, and P. Felke, "On the security of HFE, HFEv- and Quartz," *Proc. PKC 2003*, Lecture Notes in Computer Science, vol.2567, pp.337–350, Springer, 2003.

[CP03] N. Courtois and J. Patarin, "About the XL algorithm over $GF(2)$," *Proc. CT-RSA 2003*, Lecture Notes in Computer Science, vol.2612, pp.141–157, Springer,

2003.

[CGP03] N. Courtois, L. Goubin, and J. Patarin, "SFLASHv3, a fast asymmetric signature scheme," Cryptology ePrint Archive, Report 2003/211, 2003. `http://eprint.iacr.org/`

[CLO00] D. コックス，J. リトル，D. オシー著，落合啓之，示野信一，西山享，室政和，山本敦子訳，グレブナ基底と代数多様体入門（上・下），シュプリンガー・フェアラーク東京, 2000.

[CLO07] D. Cox, J. Little, and D. O'Shea, Ideals, Varieties, and Algorithms, third edition, Springer, 2007.

[Din04] J. Ding, "A new variant of the Matsumoto-Imai cryptosystem through perturbation," *Proc. PKC 2004*, Lecture Notes in Computer Science, vol.2947, pp.305–318, Springer, 2004.

[DS05a] J. Ding and D. Schmidt, "Cryptanalysis of HFEv and internal perturbation of HFE," *Proc. PKC 2005*, Lecture Notes in Computer Science, vol.3386, pp.288–301, Springer, 2005.

[DGSWY05] J. Ding, J. E. Gower, D. Schmidt, C. Wolf, and Z. Yin, "Complexity estimates for the F_4 attack on the perturbed Matsumoto-Imai cryptosystem," *Proc. IMA Int. Conf. 2005*, Lecture Notes in Computer Science, vol.3796, pp.262–277, Springer, 2005.

[DG06] J. Ding and J. E. Gower, "Inoculating multivariate schemes against differential attacks," *Proc. PKC 2006*, Lecture Notes in Computer Science, vol.3958, pp.290–301, Springer, 2006.

[DGS06a] J. Ding, J. E. Gower, and D. Schmidt, "Zhuang-Zi: a new algorithm for solving multivariate polynomial equations over a finite field," Workshop Record of the International Workshop on Post-Quantum Cryptography (PQCrypto 2006), pp.227–240, May 2006.

[DGS06b] J. Ding, J. E. Gower, and D. Schmidt, Multivariate Public Key Cryptosystems, Springer, 2006.

[DBMMW08] J. Ding, J. Buchmann, M. S. E. Mohamed, W. S. A. E. Mohamed, and R. P. Weinmann, "MutantXL," Proceedings of the First International Conference on Symbolic Computation and Cryptography (SCC 2008), pp.16–22, Apr. 2008.

[DS10] J. Ding and D. Schmidt, "Mutant Zhuang-Zi algorithm," *Proc. PQCrypto 2010*, Lecture Notes in Computer Science, vol.6061, pp.28–40, Springer, 2010.

[DFSS07] V. Dubois, P. A. Fouque, A. Shamir, and J. Stern, "Practical cryptanalysis of SFLASH," *Proc. CRYPTO 2007*, Lecture Notes in Computer Science, vol.4622, pp.1–12, Springer, 2007.

[FGLM93] J. C. Faugère, P. Gianni, D. Lazard, and T. Mora, "Efficient computation of zero-dimensional Gröbner bases by change of ordering," Journal of Symbolic Computation, vol.16, no.4, pp.329–344, 1993.

[Fau99] J. C. Faugère, "A new efficient algorithm for computing Gröbner bases (F_4)," Journal of Pure and Applied Algebra, vol.139, issues 1-3, pp.61–88, June 1999.

[Fau02] J. C. Faugère, "A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5)," *Proc. ISSAC 2002*, pp.75–83, ACM Press, 2002.

[FJ03] J. C. Faugère and A. Joux, "Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using Gröbner bases," *Proc. CRYPTO 2003*, Lecture Notes in Computer Science, vol.2729, pp.44–60, Springer, 2003.

[FP08] J. C. Faugère and L. Perret, "On the security of UOV," Proceedings of the First International Conference on Symbolic Computation and Cryptography (SCC 2008), pp.103–109, Apr. 2008.

[FL10] J. C. Faugère and S. Lachartre, "Parallel Gaussian elimination for Gröbner bases computations in finite fields," Proceedings of the 4th International Workshop on Parallel and Symbolic Computation (PASCO 2010), pp.89–97, July 2010.

[FTT08a] 藤田亮，只木孝太郎，辻井重男，"多様な多変数公開鍵暗号を汎用的に強化する非線形持駒摂動ベクトル方式，" *Proc. SCIS2008*, 1F1-1, Jan. 2008.

[FTT08b] R. Fujita, K. Tadaki, and S. Tsujii, "Nonlinear piece in hand perturbation vector method for enhancing security of multivariate public key cryptosystems," Cryptology ePrint Archive, Report 2008/298, July 2008. `http://eprint.iacr.org/`

[FTT08c] R. Fujita, K. Tadaki, and S. Tsujii, "Nonlinear piece in hand perturbation vector method for enhancing security of multivariate public key cryptosystems," *Proc. PQCrypto 2008*, Lecture Notes in Computer Science, vol.5299, pp.148–164, Springer, 2008.

[Fuj10a] 藤田亮，"rSTS 型多変数公開鍵暗号のグレブナ基底計算を用いた代数攻撃に対する安全性解析，" *Proc. SCIS2010*, 3A3-3, Jan. 2010.

[Fuj10b] R. Fujita, "Security analysis of rSTS type multivariate public key cryptosystems against algebraic attack using Gröbner bases," Recent Results Session at

the third international workshop on Post-Quantum Cryptography (PQCrypto 2010), May 25-28, 2010, Darmstadt, Germany.

[GH09] X. S. Gao and Z. Huang, "Efficient characteristic set algorithms for equation solving in finite fields and application in analysis of stream ciphers," Cryptology ePrint Archive, Report 2009/637, 2009. `http://eprint.iacr.org/`

[GJ79] M. Garey and D. Johnson, Computers and Intractability, A Guide to the Theory of NP-Completeness, Freeman, 1979.

[GMS09] W. Geiselmann, K. Matheis, and R. Steinwandt, "PET SNAKE: A special purpose architecture to implement an algebraic attack in hardware," Cryptology ePrint Archive, Report 2009/222, 2009. `http://eprint.iacr.org/`

[GT08a] 五太子政史，辻井重男，"有限体上の多変数連立二次方程式に関する新しい求解法の提案，" *Proc. SCIS2008*, 3B1-3, Jan. 2008.

[GT08b] M. Gotaishi and S. Tsujii, "HXL — a variant of XL algorithm computing Gröbner bases," Proceedings of Inscrypt 2008 Special Track on Symbolic Computation and Cryptology, pp.2–21, December 2008.

[GJS06] L. Granboulan, A. Joux, and J. Stern, "Inverting HFE is quasipolynomial," *Proc. CRYPTO 2006*, Lecture Notes in Computer Science, vol.4117, pp.345–356, Springer, 2006.

[Has09] Y. Hashimoto "Algorithms to solve massively under-defined systems of multivariate quadratic equations," Cryptology ePrint Archive, Report 2009/154. `http://eprint.iacr.org/`

[Hib06] 日比孝之編，グレブナー基底の現在，数学書房，2006.

[IM85] H. Imai and T. Matsumoto, "Algebraic methods for constructing asymmetric cryptosystems," *Proc. AAECC-3*, Lecture Notes in Computer Science, vol.229, pp.108–119, Springer, 1985.

[KS04] M. Kasahara and R. Sakai, "A construction of public key cryptosystem for realizing ciphertext of size 100 bit and digital signature scheme," *IEICE Transactions on Fundamentals*, vol.E87-A, no.1, pp.102–109, Jan. 2004.

[KS05a] M. Kasahara and R. Sakai, "A construction of public-key cryptosystem based on singular simultaneous equations," *IEICE Transactions on Fundamentals*, vol.E88-A, no.1, pp.74–80, Jan. 2005.

[KS05b] M. Kasahara and R. Sakai, "A construction of public-key cryptosystem based on singular simultaneous equations and its variants," IEICE Technical Report,

ISEC2005-7 (2005-05), May 2005.

[KFSM83] 桂重俊，藤木澄義，末永敏幸，松野明，“ランダムスピン系の統計力学における積分方程式，” 京都大学数理解析研究所講究録，no.486, pp.166–175, Apr. 1983.

[KFIFG87] S. Katsura, W. Fukuda, S. Inawashiro, N. M. Fujiki, and R. Gebauer, “Distribution of effective field in the ising spin glass of the $\pm J$ model at $T = 0$,” Cell Biochemistry and Biophysics, vol.11, no.1, pp.309–319, 1987.

[KPG99] A. Kipnis, J. Patarin, and L. Goubin, “Unbalanced oil and vinegar signature schemes,” *Proc. EUROCRYPT '99*, Lecture Notes in Computer Science, vol.1592, pp.206–222, Springer, 1999.

[KS99] A. Kipnis and A. Shamir, “Cryptanalysis of the HFE public key cryptosystem by relinearization,” *Proc. CRYPTO '99*, Lecture Notes in Computer Science, vol.1666, pp.19–30, Springer, 1999.

[Kob98] N. Koblitz, Algebraic Aspects of Cryptography, Springer, 1998.

[Kob99] N. コブリッツ著，林 彬訳，暗号の代数理論，シュプリンガー・フェアラーク東京，1999.

[KR00] M. Kreuzer and L. Robbiano, Computational Commutative Algebra 1, Springer, 2000.

[KR05] M. Kreuzer and L. Robbiano, Computational Commutative Algebra 2, Springer, 2005.

[MIHM83] 松本勉，今井秀樹，原島博，宮川洋，“暗号化変換の自明でない表現を用いる非対称暗号系，” 昭和 58 年度電子通信学会情報・システム部門全国大会講演論文集，S8-5, Sept. 1983.

[MI88a] T. Matsumoto and H. Imai, “Public quadratic polynomial-tuples for efficient signature-verification and message-encryption,” *Proc. EUROCRYPT '88*, Lecture Notes in Computer Science, vol.330, pp.419–453, Springer, 1988.

[MI88b] 松本勉，今井秀樹，“署名機能と機密保持機能を効率よく実現する多変数多項式タプル非対称暗号系の構成，” 電子情報通信学会論文誌（A）, vol.J71-A，no.7，pp.1441–1452，July 1988.

[MMDB08] M. S. E. Mohamed, W. S. A. E. Mohamed, J. Ding, and J. Buchmann, “$MXL2$: Solving polynomial equations over GF(2) using an improved mutant strategy,” *Proc. PQCrypto 2008*, Lecture Notes in Computer Science, vol.5299, pp.203–215, Springer, 2008.

[MCDBB09] M. S. E. Mohamed, D. Cabarcas, J. Ding, J. Buchmann, and S. Bulygin,

"MXL$_3$: An efficient algorithm for computing Gröbner bases of zero-dimensional ideals," *Proc. ICISC 2009*, Lecture Notes in Computer Science, vol.5984, pp.87–100, Springer, 2009.

[MP08] S. Murphy and M. B. Paterson, "A geometric view of cryptographic equation solving," Journal of Mathematical Cryptology, vol.2, no.1, pp.63–107, Apr. 2008.

[NY03] 野呂正行，横山和弘，グレブナー基底の計算　基礎篇　計算代数入門，東京大学出版会，2003.

[Pat96a] J. Patarin, "Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms," *Proc. EUROCRYPT '96*, Lecture Notes in Computer Science, vol.1070, pp.33–48, Springer, 1996.

[Pat96b] J. Patarin, "Asymmetric cryptography with a hidden monomial," *Proc. CRYPTO '96*, Lecture Notes in Computer Science, vol.1109, pp.45–60, Springer, 1996.

[PG97a] J. Patarin and L. Goubin, "Trapdoor one-way permutations and multivariate polynomials," *Proc. ICICS '97*, Lecture Notes in Computer Science, vol.1334, pp.356–368, Springer, 1997.

[PG97b] J. Patarin and L. Goubin, "Asymmetric cryptography with S-boxes," *Proc. ICICS '97*, Lecture Notes in Computer Science, vol.1334, pp.369–380, Springer, 1997.

[PGC98] J. Patarin, L. Goubin, and N. Courtois, "C^*_{-+} and HM: variations around two schemes of T. Matsumoto and H. Imai," *Proc. ASIACRYPT '98*, Lecture Notes in Computer Science, vol.1514, pp.35–49, Springer, 1998.

[PCG01] J. Patarin, N. Courtois, and L. Goubin, "QUARTZ, 128-bit long digital signatures," *Proc. CT-RSA 2001*, Lecture Notes in Computer Science, vol.2020, pp.282–297, Springer, 2001

[RS08] H. Raddum and I. Semaev, "Solving multiple right hand sides linear equations," Designs, Codes and Cryptography, vol.49, no.1-3, pp.147–160, Dec. 2008.

[Rit50] J. F. Ritt, Differential Algebra, American Mathematical Society, Colloquium Publications, vol.33, 1950.

[Saw02] 沢田浩之，"グレブナ基底計算を効率的に行うための項順序自動設定法，" 数式処理，vol.9, no.2, pp.56–77, 2002.

[Sha93] A. Shamir, "Efficient signature schemes based on birational permutations," *Proc. CRYPTO '93*, Lecture Notes in Computer Science, vol.773, pp.1–12,

Springer, 1993.

[Sho94] P. W. Shor, "Algorithms for quantum computation: Discrete logarithms and factoring," *Proc. FOCS '94*, pp.124–134, Nov. 1994.

[Ste04] A. Steel, Allan Steel's Gröbner basis timings page, `http://magma.maths.usyd.edu.au/users/allan/gb/` (last updated 2004/10/22)

[Tsu85] 辻井重男，"非線形連立方程式の順序解法を利用する公開鍵暗号方式，" 情報理論とその応用研究会，第 8 回シンポジウム資料，pp.156–157，Dec. 1985.

[TKIFM86] 辻井重男，黒澤馨，伊東利哉，藤岡淳，松本勉，"非線形連立方程式の順序解法による公開鍵暗号方式，" 電子通信学会論文誌（D），vol.J69-D，no.12，pp.1963–1970，Dec. 1986.

[TFH89] 辻井重男，藤岡淳，平山裕介，"順序解法の一般化による公開鍵暗号系，" 電子情報通信学会論文誌（A），vol.J72-A，no.2，pp.390–397，Feb. 1989.

[Tsu03] S. Tsujii, A new structure of primitive public key cryptosystem based on soldiers in hand matrix. Technical Report TRISE 02-03, Chuo University, July 2003.

[TFT04] S. Tsujii, R. Fujita, and K. Tadaki, "Proposal of MOCHIGOMA (piece in hand) concept for multivariate type public key cryptosystem," Technical Report of IEICE, ISEC2004-74 (2004-09), Sept. 2004.

[TTF04] S. Tsujii, K. Tadaki, and R. Fujita, "Piece in hand concept for enhancing the security of multivariate type public key cryptosystems: public key without containing all the information of secret key," Cryptology ePrint Archive, Report 2004/366, Dec. 2004. `http://eprint.iacr.org/`

[TTF05] S. Tsujii, K. Tadaki, and R. Fujita, "Piece in hand concept for enhancing the security of multivariate type public key cryptosystems: public key without containing all the information of secret key," *Proc. SCIS2005*, 2E1-3, pp.487–492, Jan. 2005.

[TTF06a] 辻井重男，只木孝太郎，藤田亮，"持駒行列の提案 その 2 —多変数多項式型公開鍵暗号の安全性強化のための汎用的手法—，" *Proc. SCIS2006*, 2A4-1, Jan. 2006.

[TTF06b] S. Tsujii, K. Tadaki, and R. Fujita, "Proposal for piece in hand matrix ver.2: general concept for enhancing security of multivariate public key cryptosystems," Cryptology ePrint Archive, Report 2006/051, Feb. 2006. `http://eprint.iacr.org/`

[TTF06c] S. Tsujii, K. Tadaki, and R. Fujita, "Proposal for piece in hand matrix ver.2: general concept for enhancing security of multivariate public key cryptosystems," Workshop Record of the International Workshop on Post-Quantum Cryptography (PQCrypto 2006), pp.103–117, May 2006.

[TTF07a] S. Tsujii, K. Tadaki, and R. Fujita, "Proposal for piece in hand matrix: general concept for enhancing security of multivariate public key cryptosystems," *IEICE Transactions on Fundamentals*, vol.E90-A, no.5, pp.992–999, May 2007.

[TTF07b] 辻井重男，只木孝太郎，藤田亮，"多様な多変数公開鍵暗号を汎用的に強化する非線形持駒行列の構成法，" 電子情報通信学会技術研究報告, ISEC2007-56 (2007-07), July 2007.

[TTF08] S. Tsujii, K. Tadaki, and R. Fujita, "Nonlinear piece in hand matrix method for enhancing security of multivariate public key cryptosystems," Proceedings of the First International Conference on Symbolic Computation and Cryptography (SCC 2008), pp.124–144, 2008.

[TK08] 辻井重男，笠原正雄編著，"暗号理論と楕円曲線，" 森北出版，2008.

[WBP04] C. Wolf, A. Braeken, and B. Preneel, "Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC," *Proc. SCN 2004*, Lecture Notes in Computer Science, vol.3352, pp.294–309, Springer, 2004.

[WP05] C. Wolf and B. Preneel, "Taxonomy of public key schemes based on the problem of Multivariate Quadratic equations," Cryptology ePrint Archive, Report 2005/077, 2005. `http://eprint.iacr.org/`

[Wol05] C. Wolf, "Multivariate Quadratic polynomials in public key cryptography," Ph.D. thesis, Katholieke Universiteit Leuven, Cryptology ePrint Archive, Report 2005/393, 2005. `http://eprint.iacr.org/`

[WBP06] C. Wolf, A. Braeken, and B. Preneel, "On the security of stepwise triangular systems," Designs, Codes and Cryptography, vol.40, no.3, pp.285–302, Sept. 2006.

[Wu78] W. T. Wu, "On the decision problem and the mechanization of theorem-proving in elementary geometry," Science in China Series A: Mathematics, vol.21, no.2, pp.159–172, 1978.

[YC04a] B. Y. Yang and J. M. Chen, "Theoretical analysis of XL over small fields," *Proc. ACISP 2004*, Lecture Notes in Computer Science, vol.3108, pp.277–288, Springer, 2004.

[YC04b] B. Y. Yang and J. M. Chen, "All in the XL family: theory and practice," *Proc. ICISC 2004*, Lecture Notes in Computer Science, vol.3506, pp.67–86, Springer, 2004.

[YCC04] B. Y. Yang, J. M. Chen, and N. Courtois, "On asymptotic security estimates in XL and Gröbner bases-related algebraic cryptanalysis," *Proc. ICICS 2004*, Lecture Notes in Computer Science, vol.3269, pp.401–413, Springer, 2004.

Ryo Fujita
Research and Development Initiative, Chuo University
1-13-27 Kasuga, Bunkyo-ku, Tokyo, 112-8551 Japan
E-mail Address: `rfujita@tamacc.chuo-u.ac.jp`

Serre の保型性予想をめぐって
- 計算機的保型形式論入門 -

横山 俊一（九大数理）

on October 25th, 2010

1 概要

本稿は, 2010 年 10 月 9 日（土）, 10 日（日）に九大数理にて行われた研究集会「Magma で広がる数学の世界（Exploring Mathematics with Magma）」にて筆者が行った講演のノート, 及び当日使用したスライド等に加筆・修正を行ったものである. 本稿のタイトルと当日の講演タイトルは同一であり, 保型形式論に纏わる Magma の使用法を非専門家向けに解説することを主題としている.

当初は講演時間全てを保型形式の計算デモに充てることも考えたが, 計算機を使って研究を行うという事の目的意識を明確にするため, **Serre の保型性予想**（cf. [12], 以下「Serre 予想」と略記する）という代数的整数論乃至数論幾何の分野における非常に trendy な話題を内容に組み込むことにした（因みに筆者が修士時代から研究テーマとして扱っているものである）.

この予想に関する詳細は後述するが, 簡潔に言えば代数的対象物である「Galois 表現」に対し, 解析的対象物である「保型形式」が一対一に対応しているであろう, という予想であり, 最も基本的な有理数体上の場合が 2007 年に Khare-Wintenberger [8] によって解決されたばかりである. 本予想は様々な形で一般化が期待されており, 各々研究が進められているものの, 未だ一般化された予想の定式化すら不十分といったケースも少なくない. そのため, 計算機による数値実験を重ねて両者の対応を考察することが定式化・精密化への第一歩となると期待されている. 本稿ではそのような背景を踏まえ, 予想の数値実験に纏わる計算例をいくつか紹介したい.

以下, 本稿は大きく分けて 2 つのパートから成る. まず前半部分（2 章）は古典的保型形式を題材にデモンストレーションを行う. 予備知識としては必要最小限の古典的保型形式論（例えば Serre [11] の 7 章）があれば十分であるが, より進んだ内容に興味を持たれた方には Diamond-Shurman [4], Miyake [9] または Shimura [13] などをお薦めする.

続く後半部分（3 章）では, Serre 予想に纏わる計算実験を行う. まず 3.1 節では有理数体上の Serre 予想を述べ, 簡単な対応例をご覧に入れる. そして 3.2 節で前節の一般化として基礎体が虚二次体の場合を取り上げ, 対応する保型形式として有力視されている Bianchi 保型形式を紹介すると共に, Magma による計算例を紹介する. 最後に補足として, 上記 2 種類の保型形式以外の重要な例として **Hilbert 保型形式**の簡単な計算例を 3.3 節で紹介する. Hilbert 保型形式は総実代数体上の Serre 予想において Galois 表現に対応する保型形式として大変注目されている.

なお, 本稿の目的上, Galois 表現や楕円曲線に纏わる基礎知識については大部分を割愛せざるを得なかった. これらについては適宜紹介している参考文献で補って頂きたい.

締めとなるが, 本研究集会のメインオーガナイザーである木田雅成氏（電気通信大学）, 並びに原田昌晃氏（山形大学）のお二方には, 講演の機会を頂いただけではなく, 九大数理側の運営・お手伝い等貴重な体験をさせて頂いた. この場を借りて特に感謝御礼申し上げたい. また本研究集会の開催にあたり, 事前の準備から運営に至るまでご助力頂いた金子昌信先生, 出版前の草稿に目を通したくさんの有益なコメントを下さった田口雄一郎先生, そして田坂浩二, 高田芽味, 武田枝梨加, 櫻ひろみの 4 名の学生スタッフにもお礼を述べたい.

※ The author also wants to thank Haluk Şengün for conversations on the subject, especially two Magma prototype programs to compute Bianchi modular forms.

目 次

2 デモンストレーション

まずは最も基本的な古典的保型形式を題材として, Magma による計算に慣れ親しんでいこう. 最初に保型形式の空間を構成する. 例えば $M_{12}(\Gamma_0(1)) = M_{12}(\mathrm{SL}_2(\mathbb{Z}))$ は

```
> M := ModularForms(Gamma0(1),12);
> M;
Space of modular forms on Gamma_0(1) of weight 12 and dimension 2
over Integer Ring.
```

となる. 空間の基底は q-展開の形で計算される.

```
> Basis(M);
[
   1 + 196560*q^2 + 16773120*q^3 + 398034000*q^4 + 4629381120*q^5 +
   34417656000*q^6 + 187489935360*q^7 + O(q^8),
   q - 24*q^2 + 252*q^3 - 1472*q^4 + 4830*q^5 - 6048*q^6 - 16744*q^7
   + O(q^8)
]
```

基底の q-展開の係数だけを取り出すことも容易である.

```
> f := Basis(M)[2];  // 基底の２つ目の要素
> Coefficient(f,5);
4830
```

更に高い次数まで求めたいときは, 次数を次のように指定すればよい.

```
> PowerSeries(f,11);
[
   q - 24*q^2 + 252*q^3 - 1472*q^4 + 4830*q^5 - 6048*q^6 - 16744*q^7
   + 84480*q^8 - 113643*q^9 - 115920*q^10 + O(q^11)
]
```

ところで, 上の基底は $M_{12}(\mathrm{SL}_2(\mathbb{Z}))$ の cuspidal subspace $S_{12}(\mathrm{SL}_2(\mathbb{Z}))$ の基底である. 尖点形式の空間は Magma では

```
> S := CuspidalSubspace(M);
> S;
Space of modular forms on Gamma_0(1) of weight 12 and dimension 1
over Integer Ring.
```

として実現出来る. もしくは単に `CuspForms(Gamma0(1),12)` としても良い. 因みに上の出力で保型形式の空間の次元が含まれているが, Magma の保型形式のパッケージでは, 保型形式の次元の計算は dimension formula を用いて比較的高速に求める事が出来る. 勿論, 計算時間はレベルや重さに依存する. 計算時間を比較したい場合は, 実行するコマンドの前に `time` と書けば計算時間を出力してくれる.

```
> M1 := ModularForms(Gamma1(500),500);
> M2 := ModularForms(Gamma1(1000),500);
> M3 := ModularForms(Gamma1(5000),500);
> time Dimension(M1);
3742950
Time: 0.094
> time Dimension(M2);
14971080
Time: 0.641
> time Dimension(M3);
374256600
Time: 15.938
```

このようなデータは, ループとリストの機能を使うことによって効率的に整理することが出来る. 必要なデータを書き出したり読み出したりするのが非常に容易であることも Magma のメリットと言えよう.

Magma にはグラフィカル・アウトプットの機能も備わっている[1]. 例えば合同部分群として $\Gamma_0(N)$ をとったときの**基本領域**[2]（fundamental domain）を出力するには `DisplayPolygons` コマンドを用いる[3]. デフォルトでは次のように指定されている.

```
Colours: SeqEnum                Default: [1,1,0]
Outline: BoolElt                Default: true
Fill: BoolElt                   Default: true
Show: BoolElt                   Default: false
Labels: SeqEnum                 Default: {[0,1]}
Fontsize: RngIntElt             Default: 2
Size: SeqEnum                   Default: [ ]
Pixels: RngIntElt               Default: 300
Overwrite: BoolElt              Default: false
Radius: FldReElt                Default: 0.5
PenColours: SeEnum              Default: [0,0,0]
```

実際のプログラムを見てみよう. 以下は $\Gamma_0(39)$ の例である.

```
> G := Gamma0(39);
> H<i,rho> := UpperHalfPlaneWithCusps();
> tri := [H|Infinity(),i,rho];
> tri1 := [H|0,i,rho];
> C11 := CosetRepresentatives(G);
> Colours := [[0.6,0.2,0.9] : i in [1..#C11]] cat [[1,1,0] : i in [1..#C11]];
```

[1]但し機能的には Mathematica などの様に充実しているとは言い難いので, 実用面では他のソフトウェアを併用することをすすめる.

[2]$\mathcal{H}^*$ 上で Γ の作用で移りあう点を同値とみなすことで得られる代表系のこと. 詳細は [11] Chapter. VII を参照せよ.

[3]基本領域のビジュアライゼーションについては Verrill による Java 製のソフトウェアが手頃. http://www.math.lsu.edu/~verrill/（本人のウェブページ）で試せる.

```
> triangles := [g*tri : g in C11] cat [g*tri1 : g in C11];
> DisplayPolygons(triangles,"C:/test/Gamma_0_39.ps":
> Colours := Colours, Show := false);
```

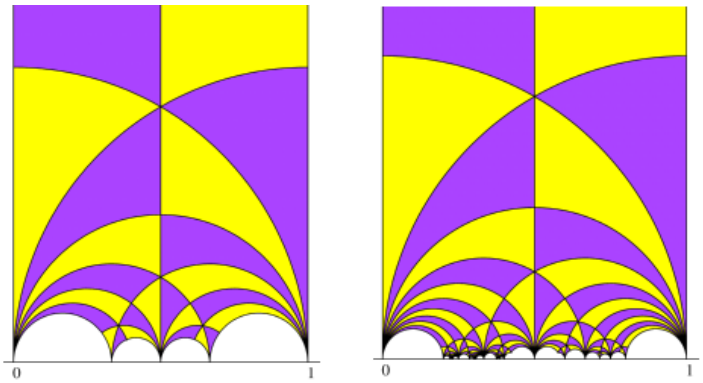

Figure 1. $\Gamma_0(11)$ および $\Gamma_0(39)$ の作用による $\mathcal{H}$ 上の基本領域.

なお Magma のマニュアルでは `FareySymbol` を用いたプログラム例も掲載されている. これは上と同じく, 与えられた合同部分群に対する基本領域を求める道具として使われているものである（cf. Kulkarni [7]）. 更に進んだ内容を知りたい場合は, 先程の脚注にて紹介した Verrill のホームページに Magma によるいくつかのプログラム例が掲載されているので, こちらも参照されたい.

続いて Hecke 作用素を計算してみる. 例として $M_2(\Gamma_0(41))$ を考え, T_2 を計算してみよう.

```
> M41 := ModularForms(Gamma0(41),2); M41;
Space of modular forms on Gamma_0(41) of weight 2 and dimension 4
over Integer Ring.
> T2 := HeckeOperator(M41,2);
> T2;
[ 3  0 12  6]
[ 0  0  3 -2]
[ 0  1 -2  0]
[ 0  0 -2  1]
> Parent(T2);
Full Matrix Algebra of degree 4 over Integer Ring
> Ch2 := CharacteristicPolynomial(T2);
> Ch2;
$.1^4 - 2*$.1^3 - 8*$.1^2 + 14*$.1 +3
> Factorization(Ch2);
[
   <$.1 - 3, 1>,
   <$.1^3 + $.1^2 - 5*$.1 - 1, 1>
]
```

なお, 上の $.1 は変数を表しており, 普段使う未知数 x と何ら変わりはない. 見栄えを良くしたいならば, あらかじめ最初に

```
> R<x> := PolynomialRing(Integers());
```

と定義しておけば $.1 は全て x に置き換わる.

次に指標を与えてみる. 例として導手 8 の指標を (何でも良いから 1 つ) 選んで $M_3(\Gamma_1(40), \varepsilon)$ を構成する.

```
> G := DirichletGroup(40, CyclotomicField(EulerPhi(40))); G;
Group of Dirichlet characters of modulus 40 over Cyclotomic
Fleld of order 16 and degree 8
> chs := Elements(G);
> #chs;
16
> [Conductor(eps) : eps in chs];
[ 1, 4, 8, 8, 5, 20, 40, 40, 5, 20, 40, 40, 5, 20, 40, 40]
> eps := chs[4]; eps;
$.1*$.2  // $.1 および $.2 は Dirichlet 指標の群の生成元
> M3 := ModularForms([eps],3); M3;
Space of modular forms on Gamma_1(40) with character $.1*$.2,
weight 3, and dimension 14 over Integer Ring.
> Dimension(CuspidalSubspace(M3));
10
```

続いてニューフォームの空間を計算してみよう. 例えば $S_2(\Gamma_0(45))$ を考えてみる.

```
> S := CuspForms(Gamma0(45),2); S;
Space of modular forms on Gamma_0(45) of weight 2 and dimension 3
over Integer Ring.
> N := Newforms(S); N;
[* [*
    q + q^2 - q^4 - q^5 - 3*q^8 - q^10 + 4*q^11 + O(q^12)
*] *]
> #N;
1
```

従って $\dim(S_2^{new}(\Gamma_0(45))) = 1$ である. ということは $\dim(S_2^{old}(\Gamma_0(45))) = 2$ となるはずである. これを確認してみよう.

まず $M \mid 45,\ M \neq 1, 45$ の候補は $M = 3, 5, 9, 15$ であり, このうち $S_2(M) \neq 0$ となるものは $M = 15$ の時のみである. 実際

```
> S15 := CuspForms(Gamma0(15),2); S15;
Space of modular forms on Gamma_0(15) of weight 2 and dimension 1
over Integer Ring.
> [PowerSeries(f,10) : f in Basis(S15)];
```

```
[
  q - q^2 - q^3 - q^4 + q^5 + q^6 + 3*q^8 + q^9 + O(q^10)
]
```

である（このただ一つの基底を f_{15} とおく）. そこでオールドフォームは

$$\alpha_d : \ S_2(15) \to S_2(45)$$

で $f(q) \in S_2(15)$ を $f(q^d) \in S_2(45)$ へ移す **degeneracy map** なる写像を用いて

$$S_2^{old}(45) = \mathrm{Im}(\alpha_1) \cup \mathrm{Im}(\alpha_3)$$

と表現されるので, その次元は 2 であることが確認出来る. ちなみに $S_2(45)$ の基底の q-展開を見ると

```
> [PowerSeries(f,20) : f in Basis(S)];
[
  q - q^4 - q^10 - 2*q^13 - q^16 + 4*q^19 + O(q^20),
  q^2 - q^5 - 3*q^8 + 4*q^11 - 2*q^17 + O(q^20),
  q^3 - q^6 - q^9 - q^12 + q^15 + q^18 + O(q^20)
]
```

とあるが, 一番下の基底は $\alpha_3(f_{15})$ に他ならない.

なお, ニューフォームは Magma ではラベル付けされており, Cremona [3] によるデータベース[4]から以下のようなラベルを用いて引き出す事が出来る.

```
[GON or G1N][Level]k[Weight][Isogeny Class]
```

例えば $S_2^{new}(\Gamma_0(11))$ は

```
> S11 := ModularForms(Gamma0(11),2);
> Newforms(S11);
[* [*
    q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10
    + q^11 + O(q^12)
*], [*
    5/12 + q + 3*q^2 + 4*q^3 + 7*q^4 + 6*q^5 + 12*q^6 + 8*q^7 + 15*q^8
    + 13*q^9 + 18*q^10 + q^11 + O(q^12)
*] *]
```

と求められるが, ラベルを用いると

```
> Newforms("GON11k2A");
[* [*
    q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10
    + q^11 + O(q^12)
```

[4]本人のウェブページから参照可能。 URL: http://www.warwick.ac.uk/~masgaj/

```
*] *]
> Newforms("G0N11k2B");
[* [*
    5/12 + q + 3*q^2 + 4*q^3 + 7*q^4 + 6*q^5 + 12*q^6 + 8*q^7 + 15*q^8
    + 13*q^9 + 18*q^10 + q^11 + O(q^12)
*] *]
```

とたった 1 行で済む．ラベルの最初の `G0N` は $\Gamma_0(N)$ のこと，$N = 11$, $k = 2$ と続き，最後の `A,B` が isogeny 類を表している．

次に楕円曲線に付随する保型形式について調べる．以下は導手 11 の楕円曲線 E を与え，それに付随する保型形式が $S_2(\Gamma_0(11))$ から得られることを確認したものである．

```
> E := EllipticCurve([0,-1,1,-10,-20]); E;
Elliptic Curve defined by y^2 + y = x^3 - x^2 - 10*x - 20 over
Rational Field
> Conductor(E);
11
> f1 := ModularForm(E); f1;
q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11
+ O(q^12)
> S11 := CuspForms(Gamma0(11),2);
> f2 := Basis(S11)[1]; f2;
q - 2*q^2 - q^3 + 2*q^4 + q^5 + 2*q^6 - 2*q^7 - 2*q^9 - 2*q^10 + q^11
+ O(q^12)
> [n : n in [1..100] | Coefficient(f1,n) ne Coefficient(f2,n)];
[]  \\ empty set
```

保型形式の合同判別定理として知られている Sturm の定理 [16] を用いると，$S_2(\Gamma_0(11))$ の Sturm bound は 2 であるから，明らかに `f1` と `f2` は一致していることが分かる．

最後に，**modular symbol** と呼ばれる重要な対象物を取りあげておく．上半平面にカスプを合わせたものをレベル N の合同部分群で割って compact Riemann 面 $X_0(N)$ を得たとき，modular symbol とはカスプ間をつなぐ $X_0(N)$ 上の path の事とイメージして頂いて差し支えない．これらは保型形式の空間を効率よく計算する為に有用な道具として用いられる．

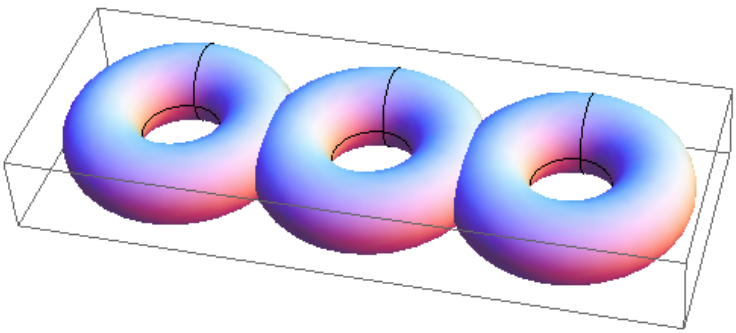

Figure 2. Compact Riemann 面 $X_0(39)$.

例を一つ挙げよう．$X_0(39)$ の 1 次のホモロジー群 $H_1(X_0(39), \mathbb{Z})$ は $\mathbb{Z}^{\oplus 6}$ に同型であり，幾何的には上図のような種数 3 の compact Riemann 面を与える．これは $S_2(\Gamma_0(39))$ の modular symbol の空間の次元が 6 であり, 1 つのトーラスは 2 つの path で張られ[5], それらが 3 つ連結して出来ていると解釈出来る．計算してみると, 確かに modular symbol の空間の基底は 6 つ得られる．

```
> S39 := CuspForms(Gamma0(39),2);
> MS := ModularSymbols(S39); MS;
Modular symbols space for Gamma_0(39) of weight 2 and dimension 6
over Rational Field
> Basis(MS);
[
   {-1/9, 0} + -1*{-1/36, 0},
   {-1/18, 0} + -1*{-1/36, 0},
   {-1/24, 0} + -1*{-1/36, 0},
   {-1/27, 0} + -1*{-1/36, 0},
   {-1/30, 0} + -1*{-1/36, 0},
   {-1/33, 0} + -1*{-1/36, 0}
]
```

勿論 modular symbol の空間にも Hecke 作用素の作用を考えることが出来る．

```
> T2 := HeckeOperator(MS,2); T2;
[ 0  0  1  1 -1 -1]
[ 1 -2  1  1  1 -2]
[ 1 -1  0  0  1  0]
[ 0  0  0  1  0  0]
[-1  0  1  1  0 -1]
[ 0 -1  1  1  0 -1]
> Ch2 := CharacteristicPolynomial(T2);
> Factorization(Ch2);
[
   <x - 1, 2>,
   <x^2 + 2*x - 1, 2>
]
```

なお modular symbol を用いず計算したものと比較すると, 両者の固有値は一致していることが分かる．

```
> T2 := HeckeOperator(S39,2); T2;
[ 0  2 -1]
[ 1 -2  1]
[ 0 -1  1]
> Ch2 := CharacteristicPolynomial(T2);
```

[5] トポロジーなどの分野では, この 2 つの path は**メリディアン**（経線：meridian）と**ロンジチュード**（緯線：longitude）などと呼ばれている．

```
> Factorization(Ch2);
[
    <$.1 - 1, 1>,
    <$.1^2 + 2*$.1 - 1, 1>  // $.1 は特性多項式の変数
]
```

再び modular symbol の計算に戻り, 固有形式の q-展開を計算しておこう.

```
> Dec2 := NewformDecomposition(MS); Dec2;
[
    Modular symbols space for Gamma_0(39) of weight 2 and dimension 2
    over Rational Field,
    Modular symbols space for Gamma_0(39) of weight 2 and dimension 4
    over Rational Field
]
> qEigenform(Dec2[1],10);
q + q^2 - q^3 - q^4 + 2*q^5 - q^6 - 4*q^7 - 3*q^8 + q^9 + O(q^10)
> qEigenform(Dec2[2],10);
q + a*q^2 + q^3 + (-2*a - 1)*q^4 + (-2*a - 2)*q^5 + a*q^6 +
 (2*a + 2)*q^7 + (a - 2)*q^8 + q^9 + O(q^10)
```

modular symbol について更に詳しく知りたいという場合は [14] を参照願いたい. この文献では一般化した modular symbol についても詳しく取り扱われている.

3 Serre の保型性予想とその検証

3.1 有理数体上の Serre 予想

まずは Galois 表現（正確には **mod l Galois 表現**）の定義から始めよう. l を素数とする.

定義 3.1. ρ が $\mathbb{Q}$ 上の **2 次元 mod l Galois 表現**であるとは, 連続準同型

$$\rho:\ G_{\mathbb{Q}} \to \mathrm{GL}_2(\overline{\mathbb{F}}_l)$$

のことである. 但し $G_{\mathbb{Q}} = \mathrm{Gal}(\overline{\mathbb{Q}}/\mathbb{Q})$（$\mathbb{Q}$ の絶対 Galois 群）とし, $G_{\mathbb{Q}}$ には Krull topology を, $\mathrm{GL}_2(\overline{\mathbb{F}}_l)$ には discrete topology を入れて考える.

さて, 保型形式には性質を特徴付けるタイプ (N, k, ε) と言う概念があったが, Galois 表現にも同様にタイプ $(N(\rho), k(\rho), \varepsilon(\rho))$ を定義することが出来る. しかしその定義はやや複雑であるので, ここではレベルだけを紹介する. 残る重さと指標の定義については, Serre の原論文 [12] を参照願いたい.

定義 3.2. V を 2 次元 $\overline{\mathbb{F}}_l$-ベクトル空間とし, $L/\mathbb{Q}$ を有限次 Galois 拡大とする. ρ : $\mathrm{Gal}(L/\mathbb{Q}) \to \mathrm{GL}(V)$ のレベル $N(\rho)$ は

$$N(\rho) = \prod_{p \neq l} p^{n(p,\rho)}$$

で定義される. ここで $n(p, \rho)$ は

$$n(p, \rho) = \sum_{i \geq 0} \frac{1}{(G_0 : G_i)} \dim(V/V_i)$$

で与えられる. なお $G_0 \supset G_1 \supset \cdots \supset G_i \supset \cdots$ は有限次 Galois 拡大 $L/\mathbb{Q}$ の Galois 群 $\mathrm{Gal}(L/\mathbb{Q})$ の高次下付き分岐群, V_i は V の G_i-固定部分空間を表す.

上から分かるように, ρ のレベルは ρ の l 以外の素数における分岐の様子を反映している. 逆に ρ の l での分岐の様子を反映しているのが重さ（**Serre weight** と呼ばれる）である.

mod l Galois 表現 ρ で $\det \rho(c) = -1$ を満たすとき, ρ は**奇**（odd）であるという（但し c は複素共役）. この ρ に対して

$$a_p = \mathrm{Tr}\,(\rho(\mathrm{Frob}_p))$$

と定義する.

それでは Serre 予想のステートメントを述べよう. 本来ならばこれは現在「予想」ではないので「Khare-Wintenberger の定理」と書くべきであろうが, ここでは「予想」のまま記すことにしたい.

予想 3.3. 任意の既約かつ奇な 2 次元 mod l Galois 表現

$$\rho\colon\ \mathrm{Gal}(\overline{\mathbb{Q}}/\mathbb{Q}) \to \mathrm{GL}_2(\overline{\mathbb{F}}_l)$$

はタイプ $(N(\rho), k(\rho), \varepsilon(\rho))$ の（古典的）尖点形式から来る．即ちある尖点形式

$$f = \sum_{n\geq 1} b_n q^n \quad \left(q = e^{2\pi i z/N}\right) \ \in S_{k(\rho)}(\Gamma, \varepsilon(\rho))$$

が存在し（ここで Γ はレベル $N(\rho)$ の合同部分群）素数 $p \nmid lN(\rho)$ に対して次が成り立つ

$$\boxed{a_p \equiv b_p}\ \ ,\ \det\left(\rho(\mathrm{Frob}_p)\right) \equiv \varepsilon(p)p^{k-1} \pmod{l}.$$

上の予想は 1987 年に発表された “refined version” 乃至 “precise form” である．Serre 予想自体はそれより十数年ほど前に Serre 自身の別の論文に登場しており, そこでは上の「タイプ」を指定しないものが述べられている（これは “weak version” 乃至 “vague form” と呼ばれている）．さらに refined version の発表後, 上の「タイプ」のうち指標に関する若干の補正が必要であることが, これまた Serre 自身によって指摘された．因みに Khare-Wintenberger によって証明されたのもこの補正を仮定したものである．また, これとは異なる Serre 予想の定式化を Edixhoven [5] も行っており, こちらの流儀を採用するとこのような補正は必要無くなる．但し Edixhoven 版の Serre 予想はその一部が未解決のままである．

Serre 予想の証明には, 帰納法のアイデアを用いた現代数論の最新のテクニックが使われている．例えば報告集 [17] には Serre 予想に纏わる（関係する）非常に詳細な解説が収められており, 非専門家向けの分かりやすい記事もあるので, こちらを参照されたい．

それでは計算例に入る．まずは最も良く知られた簡単な例から始めよう．$f(x) = x^3 - x + 1 \in \mathbb{Q}[x]$ とし, $K/\mathbb{Q}$ を f を最小多項式に持つ最小分解体とする．このとき $\mathrm{Gal}(K/\mathbb{Q}) \simeq S_3$ が成り立つ．

```
> P<x> := PolynomialAlgebra(Rationals());
> f := x^3-x+1;
> GaloisGroup(f);
Symmetric group acting on a set of cardinality 3
Order = 6 = 2 * 3
    (1, 2, 3)
    (1, 2)
[ 85633*$.1^2 - 76473*$.1 - 9587 + O(13^5), 62674*$.1^2 + 56577*$.1 -
40199 + O(13^5), -148307*$.1^2 + 19896*$.1 + 49786 + O(13^5) ]
GaloisData over Z_13
```

そこで埋め込み $\rho\colon\ S_3 \to \mathrm{GL}_2(\mathbb{F}_{23})$ を与え[6], 自然な射影 $\pi\colon\ G_{\mathbb{Q}} \to \mathrm{Gal}(K/\mathbb{Q})$ と合成して Galois 表現

$$\rho\colon\ G_{\mathbb{Q}} \to \mathrm{Gal}(K/\mathbb{Q}) \simeq S_3 \to \mathrm{GL}_2(\mathbb{F}_{23})$$

[6]詳しくは $(1\ 2) \in S_3$ を $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ へ, $(1\ 2\ 3) \in S_3$ を $\begin{pmatrix} 0 & -1 \\ 1 & -1 \end{pmatrix}$ へ移す．

を構成する．この Galois 表現 ρ に関しては, $\mathrm{Tr}(\rho(\mathrm{Frob}_p))$ の値は f の mod p での分解の型によって定まることが分かっており, 計算すると以下のようになる．

分解の型	$\mathrm{ord}(\mathrm{Frob}_p)$	$\mathrm{Tr}(\rho(\mathrm{Frob}_p))$
3 つの 1 次式の積	1	**2**
1 次式と 2 次式の積	2	**0**
既約	3	$\mathbf{-1}$

そこで f の mod p での分解の型を調べる．

```
> FCT := function(p)
function> F<w> := GF(p);
function> Q<x> := PolynomialRing(F);
function> f := x^3-x+1;
function> Factorization(f);
function> IsIrreducible(f);
function> end function;
>
> FCT(3);
[
    <x^3 + 2*x + 1, 1>
]
true
> FCT(11);
[
    <x + 6, 1>,
    <x^2 + 5*x + 2, 1>
]
false
```

一点だけ注意しておこう．ここで $p = 23$ とすると

```
> FCT(23);
[
    <x + 3, 1>,
    <x + 10, 2>
]
false
```

と計算結果が出力されるが, これは $\mathrm{Tr}(\rho(\mathrm{Frob}_p))$ のリストからは除外しなければならない[7]．事実, 対応する保型形式の T_{23} の固有値とは一致しない（詳細はすぐ後）．

[7]仮定 $p \nmid lN(\rho) = 23N(\rho)$ に反するため．実際は $N(\rho) = 1$ であり, 除外するのはこの 1 個だけである．

さて, この Galois 表現に付随する尖点形式は **Ramanujan のデルタ**

$$\Delta = q\prod_{n=1}^{\infty}(1-q^n)^{24} = \sum_{n=1}^{\infty}\tau(n)q^n$$

と呼ばれる重さ 12（かつレベル 1）の尖点形式である．これは 2 章の序盤に計算した尖点形式の空間 $S_{12}(\Gamma_0(1)) = S_{12}(\mathrm{SL}_2(\mathbb{Z}))$ の基底であって, 少し計算範囲を伸ばせば

```
q - 24*q^2 + 252*q^3 - 1472*q^4 + 4830*q^5 - 6048*q^6 - 16744*q^7 + 84480*q^8 -
    113643*q^9 - 115920*q^10 + 534612*q^11 - 370944*q^12 - 577738*q^13 +
    401856*q^14 + 1217160*q^15 + 987136*q^16 - 6905934*q^17 + 2727432*q^18 +
    10661420*q^19 - 7109760*q^20 - 4219488*q^21 - 12830688*q^22 + 18643272*q^23
    + 21288960*q^24 - 25499225*q^25 + 13865712*q^26 - 73279080*q^27 +
    24647168*q^28 + 128406630*q^29 - 29211840*q^30 - 52843168*q^31 -
    196706304*q^32 + 134722224*q^33 + 165742416*q^34 - 80873520*q^35 +
    167282496*q^36 - 182213314*q^37 - 255874080*q^38 - 145589976*q^39 +
    408038400*q^40 + O(q^41)
```

と展開される．この係数を mod 23 すれば

```
q - q^2 - q^3 + q^6 + q^8 - q^13 - q^16 + q^23 - q^24 + q^25 +
    q^26 + q^27 - q^29 - q^31 + q^39 + O(q^41)
```

となる．以上より, 両者の係数を比較すれば一致している様子が観察出来る．

p	2	3	5	7	11	13	17	19	23	29	31	37	$\cdots$
ρ	-1	-1	0	0	0	-1	0	0	*	-1	-1	0	$\cdots$
Δ	-1	-1	0	0	0	-1	0	0	(1)	-1	-1	0	$\cdots$

続いてもう少し複雑な例に移ろう．まず $\mathbb{Q}$ 上の半安定[8]（semi-stable）楕円曲線を用意し, これに付随する Galois 表現を構成する．例として

$$E:\ y^2+xy+y=x^3+1$$

を考え, E の極小判別式（minimal discriminant）を Δ_E とおく．このとき, E に付随する 2 次元 mod l Galois 表現 $\rho_{E,l}:\ \mathrm{Gal}(\overline{\mathbb{Q}}/\mathbb{Q}) \to \mathrm{Aut}(E[l]) \simeq \mathrm{GL}_2(\mathbb{F}_l)$ のタイプは

$$N(\rho_{E,l}) = \prod_{p\neq l, l\mid \mathrm{ord}_p(\Delta_E)} p\ ,\quad k(\rho_{E,l}) = \begin{cases} 2 & (l \mid \mathrm{ord}_l(\Delta_E)) \\ l+1 & (\text{otherwise}) \end{cases}\ ,\quad \varepsilon(\rho_{E,l}) = 1$$

で与えられる（cf. [12], Prop.5）．ここまでを計算してみると

```
> E := EllipticCurve([1,0,1,0,1]);
Elliptic Curve defined by y^2 + x*y + y = x^3 + 1 over
Rational Field
> D := Discriminant(E); D;
-639
> Factorization(D);
[ <3, 2>, <71, 1> ]
```

[8]至る所良い還元（good reduction）または乗法的還元（multiplicative reduction）を持つ曲線のこと．

となり $\Delta_E = -3^2 \cdot 71$ が求まる．このとき E は $l = 3$ で分裂乗法的還元（split multiplicative reduction）, $l = 71$ で非分裂乗法的還元（non-split multiplicative reduction）を持つ．今 $E[3]$ 上の Galois 表現 $\rho_3 = \rho_{E,3}$ を考えると, 先程述べた事実から $(N(\rho_3), k(\rho_3), \varepsilon(\rho_3)) = (71, 4, 1)$ が分かる．ここで $\mathrm{Tr}(\mathrm{Frob}_p(\rho_3))$ を $p \leq 50$ の範囲で計算しておく（これは E の情報から得られる）．

```
> TracesOfFrobenius(E,50);
[ 1, 1, 2, 2, 0, -2, 0, 0, 0, -2, -10, -6, 0, -4, 12 ]
```

一方, タイプ $(71, 4, 1)$ を持つような尖点固有形式を求める．これは $S_4^{new}(\Gamma_0(71))$ から得られる．

```
> S71 := CuspForms(Gamma0(71),4);
> f := Newforms(S71,1);
> [Coefficient(f,p) : p in [1..50] | IsPrime(p)];
[ 1, 1, -16, -1, 24, 7, 72, -153, -213, 232, 149, -204, -432, 71, 273 ]
```

以上 2 つの計算結果を並べてみると, 各 p について両者の値は mod 3 で一致していることが分かる．

p	2	3	5	7	11	13	17	19	23	29	31	37	$\cdots$
ρ	1	(1)	2	2	0	-2	0	0	0	-2	-10	-6	$\cdots$
f	1	(1)	-16	-1	24	7	72	-153	-213	232	149	-204	$\cdots$

$\downarrow$（mod 3）$\downarrow$

p	2	3	5	7	11	13	17	19	23	29	31	37	$\cdots$
ρ	1	(1)	-1	-1	0	1	0	0	0	1	-1	0	$\cdots$
f	1	(1)	-1	-1	0	1	0	0	0	1	-1	0	$\cdots$

3.2 虚二次体上の Serre 予想

古典的保型形式は, 2 次元上半平面 $\mathcal{H}$ とカスプ $\mathbb{P}^1(\mathbb{Q})$ の合併空間 $\mathcal{H}^* = \mathcal{H} \cup \mathbb{P}^1(\mathbb{Q})$ 上の正則関数として定義された．この一般化として, 3 次元上半空間上の実解析的関数を考えることが出来るが, これから定義する Bianchi 保型形式はこのような関数のことである．

まず, **3 次元双曲空間**（hyperbolic 3-space）

$$\mathcal{H}_3 = \mathbb{C} \times \mathbb{R}^+ = \{(z, r) \in \mathbb{C} \times \mathbb{R} \ \mid \ r > 0\} = \left\{(x, y, r) \in \mathbb{R}^3 \ \mid \ r > 0\right\}$$

には次のような計量（= hyperbolic metric）が入る．

$$ds^2 = \frac{dx^2 + dy^2 + dr^2}{r^2}$$

さらに $\mathcal{H}_3$ には $\mathrm{SL}_2(\mathbb{C})$ が次のように作用する．

$$g = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{SL}_2(\mathbb{C}) \ , \quad g(z, r) = \left(\frac{(az + b)(\overline{cz} + \overline{d}) + a\overline{c}r^2}{|cz + d|^2 + |c|^2 r^2}, \ \frac{r}{|cz + d|^2 + |c|^2 r^2} \right)$$

この作用はカスプ $\mathbb{P}^1(\mathbb{C}) = \mathbb{C} \cup \{\infty\}$ に伸び, $\mathcal{H}_3^* = \mathcal{H}_3 \cup \mathbb{P}^1(\mathbb{C})$ 上の作用となることに注意しよう[9].

さて, 古典的保型形式論における modular 群に相当するものとして, 基礎体 $K = \mathbb{Q}(\sqrt{-d})$ の整数環 $\mathcal{O}_K$ を行列の成分に持つ $\mathrm{PSL}_2(\mathbb{C})$ の部分群 $\mathrm{PSL}_2(\mathcal{O}_K)$ を考えるのは自然であろう. この群を **Bianchi 群**と呼ぶ. それでは, $\mathrm{PSL}_2(\mathcal{O}_K)$ の合同部分群を定義しよう.

定義 3.4. $\mathcal{I}$ を $\mathcal{O}_K$ のイデアルとする. Γ は $\mathrm{PSL}_2(\mathcal{O}_K)$ の有限指数の部分群であって

$$\Gamma(\mathcal{I}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \mid \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \bmod \mathcal{I} \right\}$$

を含むものとする. このとき Γ は **レベル $\mathcal{I}$ の合同部分群**（congruence subgroup）であるという. 特に次の 2 つが重要である.

- $\Gamma_0(\mathcal{I}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \mid \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} * & * \\ 0 & * \end{pmatrix} \bmod \mathcal{I} \right\}$.
- $\Gamma_1(\mathcal{I}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \mid \begin{pmatrix} a & b \\ c & d \end{pmatrix} \equiv \begin{pmatrix} 1 & * \\ 0 & 1 \end{pmatrix} \bmod \mathcal{I} \right\}$.

続いて**係数加群**（coefficient module）を導入する. R を 1 を含む可換環とし, $E_k(R)$ を k 次の 2 変数 R-係数斉次多項式のなす空間とする. この空間の R-基底は

$$\left\{ X^{k-i}Y^i \ \mid \ 0 \le i \le k \right\}$$

で与えられる. ここで Steinberg の仕事 [15] によって, $\mathbb{C}$ 上の $\mathrm{SL}_2(K)$ の既約表現は次の形で与えられることが知られている.

$$E_{k_1,k_2}(\mathbb{C}) = E_{k_1}(\mathbb{C}) \otimes \overline{E}_{k_2}(\mathbb{C})$$

ここに 2 つ目の overline は複素共役を表す. また, l を $\mathcal{O}_K$ で分解する素数とする（$l = \lambda\overline{\lambda}$ と分解しているとする）と, 先程の [15] と Brauer-Nesbitt の仕事 [1] によって, $\mathbb{F}_l$ 上の $\mathrm{SL}_2(\mathcal{O}_K/(l))$ の既約表現は次の形で与えられることが知られている.

$$E_{k_1,k_2}(\mathbb{F}_l) = E_{k_1}(\mathbb{F}_l) \otimes \overline{E}_{k_2}(\mathbb{F}_l)$$

ここに 2 つ目の overline は mod $\overline{\lambda}$ reduction の意味, つまり $\mathrm{SL}_2(\mathcal{O}/(\overline{\lambda}))$ が作用していることを表している.

それでは Bianchi 保型形式を定義する.

[9] この作用は一見複雑に見えるが, 実は modular 群の作用（一次分数変換）の自然な一般化になっている. まず $\mathbb{H}$ をハミルトンの四元数体（Hamilton's quaternion）とし, その標準 $\mathbb{R}$-基底を $\{1, i, j, k\}$ とすると, 写像 $h:\ \mathcal{H}_3 \to \mathbb{H}$ を $h((z,r)) = p = z + rj$ と定義することによって, $\mathcal{H}_3$ を $\mathbb{H}$ の部分集合とみなすことが出来る. これによって p に上の g を作用させると $g(p) = (ap+b)/(cp+d)$ が成り立つ. しかも $-I \in \mathrm{SL}_2(\mathbb{C})$ は $\mathcal{H}_3$ に自明に作用することから, $\mathrm{PSL}_2(\mathbb{C})$ が $\mathcal{H}_3$ 上に一次分数変換で作用していることが分かる.

定義 3.5. f が**レベル** $\mathcal{I}$**, 重さ** (k_1, k_2) **の Bianchi 保型形式**（Bianchi modular form）であるとは, f が 1 次元コホモロジー $H^1(\Gamma(\mathcal{I}), E_{k_1,k_2}(\mathbb{C}))$ に属するコホモロジー類であることをいう.

同様に mod l Bianchi 保型形式も定義される.

定義 3.6. f が**レベル** $\mathcal{I}$**, 重さ** (k_1, k_2) **の mod** l **Bianchi 保型形式**であるとは, f が 1 次元コホモロジー $H^1(\Gamma(\mathcal{I}), E_{k_1,k_2}(\overline{\mathbb{F}}_l))$ に属するコホモロジー類であることをいう.

なお, 上述のコホモロジーの cuspidal part を **Bianchi 尖点形式**（Bianchi cuspform）と呼ぶ.

それでは虚二次体 $K = \mathbb{Q}(\sqrt{-d})$ 上の Serre 予想のステートメントを述べる. 都合上 $\mathrm{PSL}_2(\mathcal{O}_K)$ の代わりに $\mathrm{SL}_2(\mathcal{O}_K)$ を用いることにし, 記号は以下の通りとする.

- R: 1 を含む可換環.
- $\alpha = \begin{pmatrix} \pi & 0 \\ 0 & 1 \end{pmatrix}$, 但し π は $\mathcal{O}_K$ の素元.
- $\Gamma \subset \mathrm{SL}_2(\mathcal{O}_K)$: 合同部分群.
- $\Gamma_\alpha = \Gamma \cap \alpha^{-1}\Gamma\alpha, \quad \Gamma^\alpha = \Gamma \cap \alpha\Gamma\alpha^{-1}$.
- V: 右 $R[\mathrm{Mat}_2(\mathcal{O}_K)_{\det\neq 0}]$-加群.

この時, コホモロジー $H^m(\Gamma, V)$ 上の Hecke 作用素は, 次のように Γ_α, Γ^α のコホモロジーを経由して定義される.

$$H^m(\Gamma, V) \xrightarrow{\text{restriction}} H^m(\Gamma_\alpha, V) \xrightarrow{\tilde{\alpha}} H^m(\Gamma^\alpha, V) \xrightarrow{\text{transfer}} H^m(\Gamma, V)$$

ここで真ん中の写像 $\tilde{\alpha}$ は, コサイクル $c \in H^m(\Gamma_\alpha, V)$ を用いて

$$c \mapsto (g \mapsto c(\alpha^{-1}g\alpha)\det(\alpha)\alpha^{-1})$$

で定義される. もう少し explicit に書き出してみると

$$(T_\pi c)(g) = \sum_{1\leq i\leq m} c(\gamma_{j(i)}^{-1} g\gamma_i)\det(\gamma_i)\gamma_i^{-1}$$

となる. ここで γ_i は

$$\Gamma\alpha\Gamma = \bigsqcup_{1\leq i\leq m} \gamma_i\Gamma$$

で定まり, 添え字 $j(i)$ は任意の $g, \gamma_i \in \Gamma$ に対して唯一つ定まる.

また, 以上の操作を α の代わりに $\beta = \begin{pmatrix} \pi & 0 \\ 0 & \pi \end{pmatrix}$ で行って定義される Hecke 作用素を S_π とおく.

定義 3.7. $\mathbb{F}$ を標数 l の体とする（l は 0 または素数）．$f \in H^1(\Gamma(\mathcal{I}), E(\mathbb{F}))$ が**固有形式**（eigenform）であるとは, 任意の $n \geq 1$ に対してある $c_n \in \mathbb{C}$ が存在して

$$T_n(f) = c_n f$$

と書けることをいう.

予想 3.8. 絶対既約な mod l Galois 表現 $\rho\colon\ G_K \to \mathrm{GL}_2(\mathbb{F})$ は mod l Bianchi 尖点形式から来る. 即ち, ある $f \in H^1_{cusp}(\Gamma(\mathcal{I}), E(\mathbb{F}))$ が存在し, ρ が不分岐な任意の素点 $\lambda \nmid l\mathcal{I}$ に対して次が成り立つ.

$$\boxed{\mathrm{Tr}\,(\rho(\mathrm{Frob}_\lambda)) = a_\lambda}\ ,\ \det\,(\rho(\mathrm{Frob}_\lambda)) = b_\lambda N(\lambda)\ \ .$$

ここで a_λ, b_λ はそれぞれ T_λ, S_λ の固有値

$$T_\lambda f = a_\lambda f\ ,\ \ S_\lambda f = b_\lambda f$$

を表し, $N(\lambda)$ は $\mathbb{Q}$ 上の λ のノルムである. また $G_K = \mathrm{Gal}(\overline{\mathbb{Q}}/K)$ とする.

補足 3.9. 上の形の予想は, 有理数体上の予想における “weak version” にあたる. では同様にして “refined version” を提唱出来ないか？ と考えるのは自然であるが, 対応が示唆されているのは今の所レベルだけであり, 重さについては定式化がなされていない（有理数体上の場合とは違い, 重さは 2 つの整数のペア (k, l) であるから, Galois 表現の Serre weight とは（このままでは）比較出来ない）. そのためレベルだけを指定した “intermediate version” として予想の定式化が考えられている.

それでは計算例に移ろう. Bianchi 保型形式の計算プログラムは Grunewald や Şengün らによってプロトタイプが公開され, 現在でも研究され続けているが, Magma の最新バージョン（Ver.2.16）から標準パッケージとして搭載され, 容易に扱えるようになった. とはいえこちらも完全版ではなく, 多くの伸びしろを残した状態で機能拡充の余地がある. 現時点では尖点形式（重さの指定は不可）に関するものとその Hecke 作用素絡みの計算が可能となっている.

まずは Bianchi 尖点形式 $H^1_{cusp}(\Gamma_0(\mathcal{O}_{-14}), E_2(\mathbb{C}))$ を構成する. Magma 標準搭載のパッケージでは重さ 2 のものしか扱えないため, 引数は基礎体 $K = \mathbb{Q}(\sqrt{-d})$ とレベルの情報を持つイデアル $\mathcal{I}$ の 2 つである. なお $\mathcal{O}_{-d}$ は K の整数環を表す.

```
> P<x> := PolynomialRing(Rationals());
> K := NumberField(x^2+14);
> OK := Integers(K);
> level := 1*OK;
> M := BianchiCuspForms(K, level);
> M;
```

```
Cuspidal space of Bianchi modular forms over Number Field
    with defining polynomial x^2 + 14 over the Rational Field
    Level = Ideal of norm 1 generated by ( [1, 0] )
    Weight = 2
> time Dimension(M);
0
Time: 4.980
```

なお Verbose Output の設定を最初に変更しておくと, `BianchiCuspForms` コマンドを実行した際に何が計算されたかのログが追加出力される.

```
> SetVerbose("Bianchi",2);
Found perfect form.
Finding 3-dimensional cells.
  Found 9 3-dimensional cells.
Finding 2-dimensional cells.
  Found 23 2-dimensional cells.
Finding 1-dimensional cells.
  Found 14 1-dimensional cells.
```

Magma のパッケージでは perfect form や Voronoi 多面体といった組み合わせ論的アイデアを用いて Bianchi 保型形式の計算を行っている. これは 2 章の最後で少し紹介した modular symbol の理論の一般化のアイデアによるもので, A. Ash や P. Gunnells などによって行われたコホモロジー類の計算と Hecke action の考察に基づいて設計されている.

続いて素イデアル $\mathfrak{p}|(3)$ を考え, レベル $\mathcal{I} = \mathfrak{p}^2$ を持つ Bianchi 保型形式を与えてみる. 基礎体は上と同じものを使う.

```
> level := (Factorization(3*OK)[1][1])^2;
> Norm(level);
9
> time M9 := BianchiCuspForms(K, level);
Time: 7.300
> M9;
Cuspidal space of Bianchi modular forms over Number Field
    with defining polynomial z^2 + 14 over the Rational Field
    Level = Ideal of norm 9 generated by ( [9, 0], [5, 2] )
    Weight = 2
> Dimension(M9);
1
```

ここで素イデアル $\mathfrak{q}|(23)$ を考え, 上の空間 `M9` に Hecke 作用素 $T_{\mathfrak{q}}$ を作用させると次のようになる.

```
> Q:=Factorization(23*OK); Q;
[
    <Prime Ideal of OK
```

```
    Two element generators:
        [23, 0]
        [3, 1], 1>,
    <Prime Ideal of OK
    Two element generators:
        [23, 0]
        [20, 1], 1>
]
> HeckeOperator(M9, Q[1,1]);
[8]
```

$\mathfrak{q}$ を別の素イデアル $\mathfrak{q}'|(23)$ に取りかえてみよう.

```
> Q[2,1];
Prime Ideal of OK
Two element generators:
    [23, 0]
    [20, 1]
> HeckeOperator(M9, Q[2,1]);
[-8]
```

さて, 先程も述べたが, Magma 搭載のパッケージでは重さの指定が出来ない. 重さ 2（正確には重さ $(2,2)$ のこと）以外の尖点形式を扱うには Grunewald-Şengün によるアルゴリズムを用いる. 行列計算がメインとなるため計算時間を要するが, アイデアは本節で説明した内容そのものであり, コードを追うのは容易であろう. 但しソースコードは非常に長く, 掲載するのは得策ではないと思われるので, 計算結果の例を挙げるにとどめておく. 例として基礎体を $K = \mathbb{Q}(\sqrt{-2})$ として $T(\mathcal{P}) \curvearrowright H^1(\Gamma_0(\mathcal{I}), E_{3,3}(\mathbb{C}))$, $N(\mathcal{I}) = 3$, $N(\mathcal{P}) < 100$ を計算すると以下のようになる.

$\mathcal{P}$	eigenvalue	
$1 \pm \sqrt{-2}$	-14	6
$3 \pm \sqrt{-2}$	-46	-26
$3 \pm 2\sqrt{-2}$	-574	226
$1 \pm 3\sqrt{-2}$	434	134
$3 \pm 4\sqrt{-2}$	-1246	994
$5 \pm 3\sqrt{-2}$	-3502	-1882
$3 \pm 5\sqrt{-2}$	-238	-5018
$7 \pm 3\sqrt{-2}$	-5134	8006
$1 \pm 6\sqrt{-2}$	9506	386
$9 \pm \sqrt{-2}$	11186	-2234
$9 \pm 2\sqrt{-2}$	5474	-10046
$5 \pm 6\sqrt{-2}$	-9982	8738

このうち, 例えば一番上の $T(1 \pm \sqrt{-2})$ の Hecke 作用素は

```
[              -14             0             0             0]
[         16*w + 1             6             0             0]
[1/2*(-97*w + 512)             0             6            16]
[                0             0             0           -14]
```

と計算されるので, この特性多項式を求めて

```
[
    <$.1 - 6, 2>,
    <$.1 + 14, 2>
]
```

となり, 固有値のリスト $[6,\ -14]$ が得られる. なお, 上の `w` という文字は

```
> K<w> := QuadraticField(-2);
```

から来ている.

この方面の研究で現在最も進んでいる例としては A_5-拡大の場合の計算例が [10] にあり, ここでも実際に Galois 表現との対応例（と予想されるもの）を考察している. ここでは

- Galois 表現の連続性から, L が K の有限次拡大体の時ある a が存在して, 埋め込み

$$\rho:\ \mathrm{Gal}(L/K) \hookrightarrow \mathrm{GL}_2(\mathbb{F}_{p^a})$$

  が構成出来る.

- $A_5 \simeq \mathrm{SL}_2(\mathbb{F}_4)$ である.
- 更に $\mathrm{GL}_2(\mathbb{F}_4) \hookrightarrow \mathrm{GL}_2(\overline{\mathbb{F}}_2)$ と埋め込める.

ことを使って, mod 2 Galois 表現

$$\rho:\ \mathrm{Gal}(\overline{K}/K) \to \mathrm{GL}_2(\overline{\mathbb{F}}_2)$$

を構成出来るので, 従って mod 2 コホモロジーを計算することで対応する Bianchi 保型形式を探すことが出来るだろう, というアイデアをとっている.

なお, 上の [10] で対応が示唆されているのは $K = \mathbb{Q}(\sqrt{-1})$ の場合で

$$f(x) = x^5 + (-1+2\sqrt{-1})x^4 + (-6+2\sqrt{-1})x^2 + (-4-7\sqrt{-1})x - 3\sqrt{-1} \in K[x]$$

で K 上定義される Galois 表現（A_5-Galois 拡大, $N(\rho) = (2-\sqrt{-1})^2(2+\sqrt{-1})^3$）と, mod 2 Bianchi 保型形式

$$H^1(\Gamma_0((2-\sqrt{-1})^2(2+\sqrt{-1})^3), \mathbb{F}_2)$$

であり, 次項に示すように 1 つめの eigenvalue system と一致している. ここで w は $x^2 + x + 1 \in \mathbb{F}_4[x]$ の根とする.

$\mathcal{P}$	$N(\mathcal{P})$	$\mathrm{Tr}(\rho(\mathrm{Frob}_{\mathcal{P}}))$	eigenvalue	
$3+2\sqrt{-1}$	13	1	1	1
$3-2\sqrt{-1}$	13	w^2	w^2	w
$4+\sqrt{-1}$	17	w	w	w^2
$4-\sqrt{-1}$	17	0	0	0
$6+\sqrt{-1}$	37	1	1	1
$6-\sqrt{-1}$	37	w	w	w^2
$5+4\sqrt{-1}$	41	w^2	w^2	w
$5-4\sqrt{-1}$	41	w	w	w^2
$7+2\sqrt{-1}$	53	1	1	1
$7-2\sqrt{-1}$	53	1	1	1
$6+5\sqrt{-1}$	61	w	w	w^2
$6-5\sqrt{-1}$	61	w	w	w^2
$8+3\sqrt{-1}$	73	w^2	w^2	w
$8-3\sqrt{-1}$	73	w	w	w^2
$8+5\sqrt{-1}$	89	0	0	0
$8-5\sqrt{-1}$	89	w^2	w^2	w
$9+4\sqrt{-1}$	97	1	1	1
$9-4\sqrt{-1}$	97	w^2	w^2	w

一方, Bianchi 保型形式を古典的保型形式に倣って modular symbol の形で表現出来ないかという方面の研究も考えられる（先ほども少し触れた generalized modular symbol の計算）. これについては Cremona [2] によって, 双曲空間の mosaic 細工（hyperbolic tessellation）を用いた先行研究があり, その後 Cremona の学生数人[10]によって, 特別な基礎体の場合に対する考察がなされている.

[10]E. Whitley (1990), J. Bygott (1999), M. Lingham (2005) の 3 名. Whitley と Bygott は Exeter で, Lingham は Nottingham にて Ph.D. を取得している. Whitley は類数 1 の虚二次体を, Bygott は $K=\mathbb{Q}(\sqrt{-5})$ を, Lingham は $K=\mathbb{Q}(\sqrt{-23})$, $\mathbb{Q}(\sqrt{-31})$ をそれぞれ考察している.

3.3 付録「Hilbert 保型形式とその計算」

この章の最後に付録として, 古典的保型形式からの一般化として最も良く知られている **Hilbert 保型形式**についてその計算例と共に紹介する. なお, ここでは簡単な定義やその性質について触れるのみとし, 一切の詳細は省いている. 必要に応じて適宜文献を参照されたい.

Hilbert 保型形式（Hilbert modular form）とは, 簡潔には上半平面に属する $n = \deg(F/\mathbb{Q})$ 個の複素変数を持ち, **総実代数体**（totally real field）F の元を成分に持つ 2×2 行列に対して, 古典的保型形式とほぼ同様の関係式を満足するものである. 多変数化によって難しくなったように感じるかもしれないが, 要するに基礎体を $\mathbb{Q}$ から総実代数体 F へ, 群作用を $\mathrm{PSL}_2(\mathbb{Z})$ から $\mathrm{GL}_2^+(\mathcal{O}_F)$ へ一般化したものである.

なお, Hilbert 保型形式は総実代数体上の Serre 予想において Galois 表現の対応物として有力視されている. 今現在 Serre 予想の一般化の中では最も解決に近いと思われており, 代数幾何的手法を用いて研究が進んでいる.

それでは実際に Hilbert 保型形式を計算してみよう. まず総実代数体を $F = \mathbb{Q}(\sqrt{43})$ と定義し, F 上の Hilbert 尖点形式を構成する.

```
> P<x> := PolynomialRing(Rationals());
> F := NumberField(x^2-43);
> level := 1*Integers(F);
> H := HilbertCuspForms(F, level);
> H;
Cuspidal space of Hilbert modular forms over Number Field with defining
polynomial x^2 - 43 over the Rational Field
   Level = Ideal of norm 1 generated by ( [1, 0] )
   Weight = [ 2, 2 ]
```

この尖点形式は上半平面に属する 2 個の複素変数を持っていることが読み取れる. なお空間の次元は $\mathbb{C}$ 上 10 次元である.

```
> Dimension(H);
10
```

続いて基礎体を $F = \mathbb{Q}(\sqrt{2})$ に, レベルを $\mathcal{I}|(19) \subset \mathcal{O}_F$ に取りかえてから, 空間の次元の計算にかかる時間を計測してみよう.

```
> F := NumberField(x^2-2);
> OF := RingOfIntegers(F);
> level := Factorization(19*OF)[1][1];
> H19 := HilbertCuspForms(F, level);
> time Dimension(H19);
16
Time: 0.820
```

ところで Hilbert 保型形式の計算には “quaternion order” と呼ばれるものが internal に使われている．計算に際して時間を要するのはこの部分であり, Hilbert 保型形式のレベルによるものではない．そのため, 一度基礎体を固定して異なるレベルで同様の計算を行う場合は, `QuaternionOrder` コマンドを用いてセーブしておき, 必要な時に呼び出すようにしておけば計算時間を大幅に短縮出来る．例えば直前の次元の計算を行う場合は次のようにすれば良い．

```
> QO := QuaternionOrder(H19); QO;
Order of Quaternion Algebra with base ring F
with coefficient ring Maximal Order of Equation Order with defining
polynomial x^2 - 2 over its ground order
> H19Q := HilbertCuspForms(F, level : QuaternionOrder:=QO );
> time Dimension(H19Q);
16
Time: 0.100
```

計算時間が 1/8.2 と格段に抑えられていることが読み取れる．

続いて Hecke 作用素の計算に入ろう．直前で計算した `H19`（乃至 `H19Q`）に対して Hecke 作用素 T_2 を作用させてみる．

```
> T2 := Factorization(2*OF)[1][1];
> HeckeOperator(H19, T2);
[ 0  0  0  0  1  0  1  0  0  0  1  0  0  0  0  0]
[ 0  0  0  0  0  1  0  1  0  0  1  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0  1  0  1  0  0  0  3  0]
[ 0  0  0  0  0  1  1  0  0  1  0  0  0  0  0  0]
[ 1  0  0  0  2  0  0  0  0  0  0  0  0  0  0  0]
[-1  0 -1  0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]
[ 0 -1 -1  0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]
[ 0  1  0  0  0  0  0  2  0  0  0  0  0  0  0  0]
[ 0  0  1  0  0  0  0  0  0  0  0  2  0  0  0  0]
[-1 -1 -1  0 -1 -1 -1 -1 -1 -1 -1 -1  0 -1 -1 -1]
[ 1  1  1  0  0  0  0  0  0  0  0  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0  2  0  0  0  1  0  0  0]
[ 0  0  0  0  0  0  0  0  0  1  0  1  0  2  0  0]
[ 0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  2]
[ 0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0]
[ 0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1]
```

最後は固有形式の計算を行う．ここでは例として Hilbert 保型形式 `H19` のニューフォームが $\mathbb{Q}(\sqrt{2})$ 上の楕円曲線で導手が 19 のものから来ており, それが $\mathbb{Q}$ 上の楕円曲線から得られていることを見てみよう．まず `H19` の分割を行う．

```
> decomp := NewformDecomposition(NewSubspace(H19)); decomp;
[*
```

```
    New cuspidal space of Hilbert modular forms of dimension 1
        over Number Field with defining polynomial x^2 - 2
        over the Rational Field
        Level = Ideal of norm 361 generated by ( [19, 0] )
        New at Ideal of norm 361 generated by ( [19, 0] )
        Weight = [ 2, 2 ],
    New cuspidal space of Hilbert modular forms of dimension 1
        over Number Field with defining polynomial x^2 - 2
        over the Rational Field
        Level = Ideal of norm 361 generated by ( [19, 0] )
        New at Ideal of norm 361 generated by ( [19, 0] )
        Weight = [ 2, 2 ],
    New cuspidal space of Hilbert modular forms of dimension 6
        over Number Field with defining polynomial x^2 - 2
        over the Rational Field
        Level = Ideal of norm 361 generated by ( [19, 0] )
        New at Ideal of norm 361 generated by ( [19, 0] )
        Weight = [ 2, 2 ],
    New cuspidal space of Hilbert modular forms of dimension 8
        over Number Field with defining polynomial x^2 - 2
        over the Rational Field
        Level = Ideal of norm 361 generated by ( [19, 0] )
        New at Ideal of norm 361 generated by ( [19, 0] )
        Weight = [ 2, 2 ]
*]
```

このうち 2 番目の 1 次元空間に注目する.

```
> f := Eigenform(decomp[2]);
> primes := [P : P in PrimesUpTo(50,F) | IsOdd(Norm(P)) and IsPrime(Norm(P))];
> for P in primes do
>    Norm(P), HeckeEigenvalue(f,P);
> end for;
7 -1
7 -1
17 -3
17 -3
23 0
23 0
31 -4
31 -4
41 -6
41 -6
47 -3
47 -3
```

一方, レベル 19 の古典的保型形式（つまり導手 19 の $\mathbb{Q}$ 上の楕円曲線から来る）からも全く同じ固有値のリストが得られる.

```
> fQ := Newforms(CuspForms(19))[1][1];
> for P in primes do
```

```
>    p := Norm(P);
> p, Coefficient(fQ, p);
> end for;
7 -1
7 -1
17 -3
17 -3
23 0
23 0
31 -4
31 -4
41 -6
41 -6
47 -3
47 -3
```

参考文献

[1] R. Brauer and C. Nesbitt, *On the modular characters of groups*, Ann. of Math. (2), **42** (1941), 556-590.

[2] J. E. Cremona, *Hyperbolic tessellations, modular symbols, and elliptic curves over complex quadratic fields*, Compositio Math., **51** (1984), 275-324.

[3] J. Cremona, *Algorithms for modular elliptic curves*, Cambridge University Press, Cambridge, Second Edition (1997).

[4] F. Diamond and J. Shurman, *A First Course in Modular Forms*, Graduate Texts in Mathematics 228, Springer-Verlag (2005).

[5] B. Edixhoven, *The weight in Serre's conjectures on modular forms*, Invent. Math. **109** (1992), 563-594.

[6] J. Franke, *Harmonic analysis in weighted l_2-spaces*, Ann. Sci. École Norm. Sup. (4), **31** (1998), 181-279.

[7] R. S. Kulkarni, *An arithmetic-geometric method in the study of the subgroups of the modular group*, Amer. J. Math., **113** (6) (1991), 1053-1133.

[8] C. Khare, J.-P. Wintenberger, *Serre's modularity conjecture I,II*, 2007, preprint.

[9] T. Miyake, *Modular forms*, Springer-Verlag, Berlin (1989), Translated from the Japanese by Yoshitaka Maeda.

[10] M. H. Şengün, *A numerical study of mod p Bianchi modular forms and Galois representations*, preprint.

[11] J.-P. Serre, *A course in arithmetic*, Graduate Texts in Mathematics 7, Springer-Verlag, New York (1973).

[12] J.-P. Serre, *Sur les représentations modulaires de degré 2 de* $\mathrm{Gal}(\overline{\mathbb{Q}}/\mathbb{Q})$, Duke Mathematical Journal **54** (1987), 179-230.

[13] G. Shimura, *Introduction to the Arithmetic Theory of Automorphic Functions*, Publ. Math. Soc. Japan, **11**, Tokyo (1971).

[14] W. Stein, *Modular Forms: A Computational Approach*, with an appendix by P. Gunnells, AMS Graduate Studies in Mathematics, Vol. 79 (2007).

[15] R. Steinberg, *Tensor Product theorems*, in The Arcata Conference on Representations of Finite Groups (Arcata, Calif., 1986), vol. 47 of Proc. Sympos. Pure Math., Amer. Math. Soc., Providence, RI, 1987, 331-338.

[16] J. Sturm, *On the congruence of modular forms*, Springer Lect. Notes in Math. **1240** (1984), 275-280.

[17] T. Saito, G. Yamashita and S. Yasuda, *Recent progressions on R=T*, volume 1 and 2, Proceedings available at: http://www.kurims.kyoto-u.ac.jp/~gokun/R=T.html.

Shun'ichi Yokoyama
Graduate School of Mathematics, Kyushu University
744 Motooka, Nishi-ku, Fukuoka, 819-0395, Japan
E-mail Address: `s-yokoyama@math.kyushu-u.ac.jp`

代数体上の楕円曲線の計算と Magma[1]

松野 一夫

本稿では, 代数体上の楕円曲線に関連する計算に Magma がどのように利用できるかについて, Mordell-Weil 群の計算を中心に, 具体的な利用法[2]とともに (あまり脈絡もなく) 述べていく.

1 Magma での楕円曲線

楕円曲線は多くの場合,

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \tag{1}$$

という形の Weierstrass 方程式によって定義される. Magma でもこの形が標準的で,

```
> EllipticCurve([1,2,3,4,6]);
Elliptic Curve defined by y^2 + x*y + 3*y = x^3 + 2*x^2 + 4*x + 6 over Rational
Field
```

と係数を並べて定義する. この例では整数を並べたので有理数体上の楕円曲線として定義されたが, 他の体の元を並べれば, 当然その体上の楕円曲線が定義される. 次のような書き方もできる.

```
> EllipticCurve([9,1]);
Elliptic Curve defined by y^2 = x^3 + 9*x + 1 over Rational Field
>
> R<x>:=PolynomialRing(Rationals());
> EllipticCurve(x^3-3*x^2+1);
Elliptic Curve defined by y^2 = x^3 - 3*x^2 + 1 over Rational Field
```

方程式 (1) で定義される曲線が実際に楕円曲線になるためには, 曲線が特異点を持たないことが必要であるが, うっかり特異点を持つような係数を与えてしまった場合は,

```
> EllipticCurve([0,1,0,0,0]);

>> EllipticCurve([0,1,0,0,0]);
               ^
Runtime error in 'EllipticCurve': Curve is singular
```

とエラーが出力される.

有理数体上の楕円曲線については, 導手 (conductor) が 130000 以下の全ての楕円曲線を導手の小さい順に列挙した Cremona によるテーブル [Cr] が広く利用されているが, Magma にもこのデータベースが組み込まれており, 簡単に利用できるようになっている[3].

[1] 2010 年 10 月 10 日に研究集会「Magma で広がる数学の世界」で行った講演に基づいて作成したものであるが, 話の流れや実演部分などは講演時とは異なっているところもある.

[2] 本稿作成にあたって利用した Magma の version は V.2.16-13 であり, それ以外の version では挙動が異なるかもしれない.

[3] オプションで Stein-Watkins の (網羅的ではないが) より大きなデータベースを利用することもできる.

```
> EllipticCurve(CremonaDatabase(),"11A1");
Elliptic Curve defined by y^2 + y = x^3 - x^2 - 10*x - 20 over Rational Field
> EllipticCurve(CremonaDatabase(),37,1,1);
Elliptic Curve defined by y^2 + y = x^3 - x over Rational Field
> CremonaReference(EllipticCurve([2,1]));
472a1
```

楕円曲線の有理点全体にはアーベル群の構造が入る．この演算も以下のように自然に計算することができる．

```
> E:=EllipticCurve([-5,0]); E;
Elliptic Curve defined by y^2 = x^3 - 5*x over Rational Field
> P:=E![0,0]; P;              // P=(0,0) を E 上の点と思う
(0 : 0 : 1)
> Q:=Points(E,-1)[1]; Q;      // Q は x 座標が -1 であるような E 上の点の一方
(-1 : 2 : 1)
> P + Q;
(5 : 10 : 1)
> 2*Q;
(9/4 : -3/8 : 1)
> P - 3*Q;
(845/121 : 23270/1331 : 1)
> 2*P;
(0 : 1 : 0)
> Identity(E);                // これは単位元なので P は位数 2 の点
(0 : 1 : 0)
> Order(P);
2
> Order(Q);                   // Q の方は位数無限である
0
```

代数体上の楕円曲線の場合, その有理点全体のなす群は有限生成なアーベル群となることが知られており (Mordell-Weil の定理), その構造を調べることは重要な問題である．Magma で行える計算については, §4 で説明する．

2 Weierstrass 型でない楕円曲線

前節で見たように, 楕円曲線は Weierstrass 型の方程式 (1) で定義されるものを考察することが多いのだが, 本来は種数 1 の非特異射影曲線で定義体上に少なくとも 1 つは有理点を持つものとして定義され, Weierstrass 型以外の方程式で定められた楕円曲線を扱わなくてはならない場合もある．Magma ではそのような Weierstrass 型以外の楕円曲線もかなり幅広く扱えるよう準備されている．例えば, 平面 3 次曲線 $x^3 + y^3 = 1$ (の射影化) は特異点を持たないので種数 1 であり, $(x, y) = (1, 0)$ などの有理点を持つので楕円曲線と見なせるが, Magma では以下のように扱うことができる．

```
> P2<X,Y,Z>:=ProjectiveSpace(Rationals(),2);  // Q 上の射影平面
> C:=Curve(P2,X^3+Y^3-Z^3);                   // x^3 + y^3 = 1 の射影化
> pt:=C![1,0,1];                              // pt=(1,0)
> E,m:=EllipticCurve(C,pt);
> E;                                          // C と同型な Weierstrass model
Elliptic Curve defined by y^2 - 9*y = x^3 - 27 over Rational Field
```

```
> m;                                          // m : C -> E
Mapping from: CrvPln: C to CrvEll: E
with equations :
3*Y
-9*X
-X + Z
and inverse
$.2
-3*$.1
$.2 - 9*$.3
> m(pt);                                      // (1,0) は E の単位元に
(0 : 1 : 0)
> m(C![0,1,1]);                               // (0,1) は (x,y)=(3,0) に
(3 : 0 : 1)
> IsIsomorphic(E,EllipticCurve([0,-432]));    // この形の方程式をよく見る
true
```

このように, 実際には Weierstrass 型の楕円曲線に変換して扱うのであるが, 変換する写像 (と必要なら逆写像) もあわせて得られるので, 一方で計算したものを他方にうつすといったことを自由に行えるのである[4]. 扱える種数 1 の曲線は平面曲線に限らず, 次のような例も考えられる. これは 3 辺の長さが全て有理数であり, 面積が 6 であるような直角三角形を探すものである.

```
> P3<X,Y,Z,W>:=ProjectiveSpace(Rationals(),3);  // 射影空間
> C:=Curve(P3,[X^2+Y^2-Z^2,X*Y/2-6*W^2]); C;    // 2 つの 2 次曲面の交わり
Curve over Rational Field defined by
X^2 + Y^2 - Z^2,
1/2*X*Y - 6*W^2
> //  |X/W|, |Y/W|, |Z/W| が面積 6 の直角三角形の 3 辺
> A,f:=EllipticCurve(C,C![0,1,1,0]);
> E,g:=MinimalModel(A); h:=f*g;
> E; h;                                       // h : C -> E
Elliptic Curve defined by y^2 = x^3 - 36*x over Rational Field
Mapping from: Crv: C to CrvEll: E
with equations :
6*X
-72*W
-Y + Z
and inverse
192*$.1*$.3
16*$.1^2 - 576*$.3^2
16*$.1^2 + 576*$.3^2
-16*$.2*$.3
> P:=C![3,4,5,1]; P;                          // 有名な直角三角形
(3 : 4 : 5 : 1)
> h(P);
(18 : -72 : 1)
> _,hinv:=IsInvertible(h);                    // hinv は h の逆写像
> hinv(2*h(P));                               // これらも面積 6
(120/7 : 7/10 : 1201/70 : 1)
> hinv(3*h(P));
(-4653/851 : -3404/1551 : -7776485/1319901 : 1)
> hinv(4*h(P));
(-2017680/1437599 : -1437599/168140 : -2094350404801/241717895860 : 1)
```

[4] しかし, 別の model 上で演算等を直接考えられる訳ではないので, 用途によっては意味がないかもしれない.

3 代数体上の楕円曲線の reduction

代数体上の楕円曲線のことを調べる際, その体の素イデアルを法とする reduction の様子を知ることは非常に重要である. 例えば bad reduction となる素イデアルを決定し, reduction type などを調べたり, good reduction となる場合に剰余体上での楕円曲線の位数や群構造等を調べるといったことが必要となるが, Magma では有理数体上に限らない代数体上でそのような計算をすぐに行えるよう, 様々なコマンドが用意されている.

```
> E:=EllipticCurve([-33,-107]); E;
Elliptic Curve defined by y^2 = x^3 - 33*x - 107 over Rational Field
> Conductor(E);
3780
> Factorization(Conductor(E));        // conductor を割る素数で bad reduction
[ <2, 2>, <3, 3>, <5, 1>, <7, 1> ]
> BadPrimes(E);
[ 2, 3, 5, 7 ]
> KodairaSymbols(E);
[ IV, II, I3, I2 ]
> TamagawaNumbers(E);
[ 1, 1, 1, 2 ]
> [ReductionType(E,p) : p in [2,3,5,7,11]];
[ Additive, Additive, Nonsplit multiplicative, Split multiplicative, Good ]
> E11:=ChangeRing(E,FiniteField(11)); E11;          // mod 11 reduction
Elliptic Curve defined by y^2 = x^3 + 3 over GF(11)
> #E11;
12
> AbelianGroup(E11);
Abelian Group isomorphic to Z/12
Defined on 1 generator
Relations:
    12*$.1 = 0
> FrobeniusTraceDirect(E,11);         // 1+p-#(E mod p) が Frobenius の trace
0
> FrobeniusTraceDirect(E,NextPrime(10^100));
-1725688897428606483079072189241951756802510121190 02
>
> F<a>:=QuadraticField(2);            // F=Q(sqrt2)
> E:=EllipticCurve([a,1]); E;
Elliptic Curve defined by y^2 = x^3 + a*x + 1 over F
> Factorization(Conductor(E));        // E は 2 と 601 上の素イデアルで bad
[
    <Prime Ideal
    Two element generators:
        2
        $.2, 7>,
    <Prime Ideal
    Two element generators:
        601
        $.2 + 379, 1>
]
> KodairaSymbol(E,Factorization(Conductor(E))[1][1]);
III
> L:=Factorization(601*IntegerRing(F)); L;  // 601 は F で分解
[
    <Prime Ideal
```

```
    Two element generators:
        601
        $.2 + 222, 1>,
    <Prime Ideal
    Two element generators:
        601
        $.2 + 379, 1>
]
> EE,f:=Reduction(E,L[1][1]); EE; f;        // good reduction の方
Elliptic Curve defined by y^2 = x^3 + 379*x + 1 over GF(601)
Mapping from: CrvEll: E to Elliptic Curve defined by y^2 = x^3 + 379*x + 1 over
GF(601) given by a rule [no inverse]
> Reduction(E,L[2][1]);                     // bad reduction の方

>> Reduction(E,L[2][1]);
             ^
Runtime error in 'Reduction': model should be integral and of good reduction at
the prime
```

$\mathbb{Q}$ 上の楕円曲線は必ずある素数を法として bad reduction となることが知られているが, 一般の代数体では全ての素イデアルで good reduction (everywhere good reduction) となる楕円曲線が存在し得る. 次の例は [Se, p. 320] で扱われている有名な曲線である.

```
> K<b>:=QuadraticField(29); e:=FundamentalUnit(K);  // e は基本単数
> A:=EllipticCurve([1,0,e^2,0,0]); A;
Elliptic Curve defined by y^2 + x*y + 1/2*(-5*b + 27)*y = x^3 over K
> Discriminant(A);
1/2*(2646275*b - 14250627)
> -e^10;                                  // A の判別式は -e^10 なので単数
1/2*(2646275*b - 14250627)
> Conductor(A);
Principal Ideal
Generator:
    1
> BadPlaces(A);                           // 確かに everywhere good reduction
[]
> [#Reduction(A,Ideal(Decomposition(K,v)[1][1])) : v in [2,5,7]];
[ 6, 9, 6 ]
```

4 Mordell-Weil 群の計算

代数体 K 上の楕円曲線 E に対し, E の K-有理点全体のなすアーベル群 $E(K)$ は有限生成アーベル群となる, つまり, ある非負整数 r と有限アーベル群 T が存在し, $E(K) \cong \mathbb{Z}^r \oplus T$ と表せる, というのが有名な Mordell-Weil の定理である ([Si, Chapter VIII] など参照). この定理に由来して, $E(K)$ は E の K 上の Mordell-Weil 群と呼ばれるが, その階数 $r = \mathrm{rank} E(K)$ やねじれ部分 T を考察することは, 代数体上の楕円曲線の中心的な問題の 1 つとなっている. 一般に, 与えられた楕円曲線の Mordell-Weil 群を計算するのは容易なことではないのであるが, Magma にはその計算をサポートするコマンドが多数用意されている. まずは $\mathbb{Q}$ 上の楕円曲線の場合に, 基本的な計算法を見てみることにする.

```
> E1:=EllipticCurve(CremonaDatabase(),"11a1"); E1;
Elliptic Curve defined by y^2 + y = x^3 - x^2 - 10*x - 20 over Rational Field
> TorsionSubgroup(E1);                  // torsion part は位数 5 の巡回群
Abelian Group isomorphic to Z/5
Defined on 1 generator
Relations:
    5*$.1 = 0
> Rank(E1);                             // rank は 0
0
> Generators(E1);                       // Mordell-Weil 群の生成元
[ (16 : 60 : 1) ]
>
> E2:=EllipticCurve([-6^2,0]); E2;
Elliptic Curve defined by y^2 = x^3 - 36*x over Rational Field
> M,m:=MordellWeilGroup(E2); M;         // Mordell-Weil 群を計算するコマンド
Abelian Group isomorphic to Z/2 + Z/2 + Z
Defined on 3 generators
Relations:
    2*$.1 = 0
    2*$.2 = 0
> [m(M.i): i in [1,2,3]];               // Generators(E2) と同じこと
[ (6 : 0 : 1), (0 : 0 : 1), (18 : 72 : 1) ]
```

このように, うまく行く場合には, 単純にいくつかのコマンドを使うだけで Mordell-Weil 群を簡単に計算できる. 特に, ねじれ部分の計算は (具体的に与えられた楕円曲線に対しては) 難しくなく, もっと大きな代数体上の場合でも大抵すぐに求めることができる. しかしながら, rank の計算はそう簡単にはいかない.

```
> P3<X,Y,Z,W>:=ProjectiveSpace(Rationals(),3);
> C:=Curve(P3,[X^2+Y^2-Z^2,X*Y/2-157*W^2]);   // 面積 157 の直角三角形
> A,f:=EllipticCurve(C,C![0,1,1,0]);
> E,g:=MinimalModel(A); E;
Elliptic Curve defined by y^2 = x^3 - 24649*x over Rational Field
> TorsionSubgroup(E);                         // torsion part は問題なし
Abelian Group isomorphic to Z/2 + Z/2
Defined on 2 generators
Relations:
    2*Tor.1 = 0
    2*Tor.2 = 0
> Rank(E);                                    // rank は決められなかった
Warning: rank computed (0) is only a lower bound
(It may still be correct, though)
0
```

最後の警告は, 位数無限の点は見つからなかったのだが, ひょっとすると rank は正かもしれない, というものである. 正かもしれないと考えている 1 つの理由として, 裏で (おそらく) 次の計算を行っていることが挙げられる.

```
> Sel2:=TwoSelmerGroup(E); Sel2;
Abelian Group isomorphic to Z/2 + Z/2 + Z/2
Defined on 3 generators
Relations:
    2*Sel2.1 = 0
    2*Sel2.2 = 0
    2*Sel2.3 = 0
```

これは楕円曲線の 2-Selmer 群と呼ばれるものを計算している．代数体 K 上の楕円曲線 E の n-Selmer 群 $\mathrm{Sel}_n(E/K)$ とは, 楕円曲線の n 倍写像を使って定義され, Mordell-Weil 群の情報と, もう 1 つの重要な研究対象である Tate-Shafarevich 群 $Ш(E/K)$ の情報を併せ持つものであり,

$$0 \longrightarrow E(K)/nE(K) \longrightarrow \mathrm{Sel}_n(E/K) \longrightarrow Ш(E/K)[n] \longrightarrow 0 \tag{2}$$

という完全列を満たすものである (定義等は [Si, Chapter X] など参照)．上の例の場合には, $\mathrm{Sel}_2(E/\mathbb{Q}) \cong (\mathbb{Z}/2\mathbb{Z})^3$ となっているのであるが, $E(\mathbb{Q})$ は $(\mathbb{Z}/2\mathbb{Z})^2$ と同型な部分群を持っていたので, 完全列 (2) より, 残る 1 つの $\mathbb{Z}/2\mathbb{Z}$ が $E(\mathbb{Q})$ の位数無限の点か $Ш(E/\mathbb{Q})$ の位数 2 の元から来ることになる．つまり, この 2-Selmer 群の計算により $\mathrm{rank}E(\mathbb{Q}) \leq 1$ であることはわかるが, 位数無限の点を見つけられなかったので, 0 なのか 1 なのかわからない, ということである．

実はこの曲線は $\mathrm{rank}E(\mathbb{Q}) = 1$ なのであるが, 位数無限の点は座標の分母分子がかなり大きいため, 有理点を簡単には見つけられなかったのである．有理点の探索範囲を広げるという解決法もあり得るが, Magma にはそのような大きな有理点を探す際に使うと有効[5]と思われる, 以下のようなコマンドが用意されている．

```
> T:=TwoDescent(E:RemoveTorsion:=true); T;
[
    Hyperelliptic Curve defined by y^2 = 157*x^4 + 628 over Rational Field
]
> F:=FourDescent(T[1]:RemoveTorsion:=true); F;
[
    Curve over Rational Field defined by
    $.1^2 + 2*$.1*$.2 - $.2^2 + 2*$.2*$.3 + 2*$.3^2 + 2*$.2*$.4 + 16*$.3*$.4 -
        3*$.4^2,
    3*$.1^2 + 2*$.1*$.2 - 3*$.2^2 + 2*$.1*$.3 + 4*$.2*$.3 + 8*$.3^2 + 2*$.1*$.4
        + 4*$.2*$.4 - 6*$.3*$.4 - 9*$.4^2
]
```

この `T[1]` は, $\mathrm{Sel}_2(E/\mathbb{Q})$ の (先程 1 つ残った) 位数 2 の元に対応する, E の 2-covering と呼ばれる種数 1 の曲線であり, `T[1]` から E への $\mathbb{Q}$ 上定義された次数 4 の写像を持っている．更に, `F[1]` は $\mathrm{Sel}_4(E/\mathbb{Q})$ のある元に対応する `T[1]` の 2-covering で, `F[1]` から E への次数 16 の写像が存在する．もし E の rank が正であれば, この `F[1]` も有理点を持つことが示せるが, E への写像の次数が大きいことより, `F[1]` 上には比較的小さな分母分子の有理数を座標とする有理点が存在すると期待される．実際に探してみると, 次のようにあっさりと有理点が見つかってしまう．

```
> L:=PointsQI(F[1],1000); L;            // F[1] 上での有理点探索
[ (-983/417 : 118/139 : 11/417 : 1), (185/139 : 74/417 : 11/417 : 1), (343/11 :
566/11 : -417/11 : 1), (63/11 : -972/11 : -417/11 : 1) ]
> B,j:=AssociatedEllipticCurve(F[1]); _,k:=IsIsomorphic(B,E);
> P:=k(j(L[1])); P;                     // 見つけた点の 1 つを E に送る
(-16613623166818526754080 4/28256306942511458580 25 :
167661624456834335404812111469782006/150201095200135518108761470235125 : 1)
> Order(P);                             // 確かに位数無限
0
```

[5]今のような rank 1 の場合には, Heegner 点を求める方が良いかもしれないが, 以下の議論は rank が 2 以上の場合にも適用できる．

```
> _,hinv:=IsInvertible(f*g); hinv(P);   // 直角三角形の 3 辺の長さ
(68032984878264350512175 40/411340519227716149383203 :
411340519227716149383203/2166655569371476130961 0 :
-224403517704336969924557513090674863160948472041/8912332268928859588025535178 9
67163570016480830 : 1)
```

このような Selmer 群の計算とその周辺は現在でも開発が盛んに行われているようであり, 今後ますます便利になるものと思われる[6].

有理数体上の楕円曲線の Mordell-Weil 群に関係する最も重要な問題の 1 つとして, Birch, Swinnerton-Dyer 予想がある ([Si, Appendix C] など参照). それは, $\mathbb{Q}$ 上の楕円曲線 E の Mordell-Weil 群の rank は, E の Hasse-Weil L 関数 $L(E,s)$ の $s=1$ での零点の位数に等しい, というものであり, 零点の位数が 1 以下であれば正しいことが知られている. 更に, $L(E,s)$ の $s=1$ の周りでの Taylor 展開の先頭係数は, $Ш(E/\mathbb{Q})$ の位数や楕円曲線の実周期などの量で書ける, という精密化された予想もあり, そちらについても部分的な結果が知られている. Magma でも関連するいくつかの値を計算することが可能であり, Mordell-Weil 群の計算に活用することができる.

```
> E1:=EllipticCurve(CremonaDatabase(),"11a1");
> AnalyticRank(E1);        // L 関数の s=1 での零点の位数
0 0.25384
> _,L:=AnalyticRank(E1);   // 2 番目の戻り値は Taylor 展開の先頭係数
> L/RealPeriod(E1);        // それを実周期で割った値
0.20000
> L*(#TorsionSubgroup(E1))^2/(RealPeriod(E1)*(&*TamagawaNumbers(E1)));
0.99999
```

最後の値が曲線 E1 の Tate-Shafarevich 群の位数になるはずなのであるが, 近似計算のため, ちょうど 1 になっていないのが少々残念である. しかし, Magma では modular symbol と呼ばれるものの計算が可能であり, それを利用すると analytic rank が 0 の場合にはこの値を正確に (有理数として) 求めることができる[7].

```
> MS:=ModularSymbols(E1); MS;
Modular symbols space for Gamma_0(11) of weight 2 and dimension 2 over Rational
Field
> LRatio(MS,1);
1/5
> LRatio(MS,1)*(#TorsionSubgroup(E1))^2/(&*TamagawaNumbers(E1));
1
```

$\mathbb{Q}$ 上の楕円曲線ばかりを扱ってきたが, TwoSelmerGroup などは一般の代数体上でも使えるようになっており, 次のような計算を (何とか) 行うことが可能である (次節の例も見よ).

```
> K<b>:=QuadraticField(29); e:=FundamentalUnit(K);
> A:=EllipticCurve([1,0,e^2,0,0]);  // everywhere good red. の例
> Tor:=TorsionSubgroup(A); Tor;     // torsion part は易しい
Abelian Group isomorphic to Z/3
Defined on 1 generator
```

[6]そのかわり, バグに出くわすことも少なくない. 今回の講演の準備をしていた時にも, TwoSelmerGroup の計算でバグを発見し, 修正してもらったことがあった. また, 筆者が使っている Windows 版の Magma では, FourDescent がうまく動かないように思えるのだが, これはまだ直してもらえていない.

[7]ただし, 残念ながら, 導手がかなり小さい楕円曲線しか扱えない.

```
Relations:
    3*Tor.1 = 0
> Sel2:=TwoSelmerGroup(A); Sel2;    // この計算より rank=0 がわかる
Abelian Group of order 1
> MordellWeilSubgroup(A);           // M-W 群の奇数指数部分群を見つけるコマンド
true
Abelian Group isomorphic to Z/3
Defined on 1 generator
Relations:
    3*$.1 = 0
Mapping from: Abelian Group isomorphic to Z/3
Defined on 1 generator
Relations:
    3*$.1 = 0 to CrvEll: A given by a rule [no inverse]
>
> F<a>:=QuadraticField(2); E:=EllipticCurve([a,1]);  // 601 上の素点で bad の例
> T:=TorsionSubgroup(E); T;             // torsion part は自明
Abelian Group of order 1
> b,M,m:=MordellWeilSubgroup(E); b; M;  // rank は 2
true
Abelian Group isomorphic to Z + Z
Defined on 2 generators (free)
> m(M.1);
(0 : 1 : 1)
> m(M.2);
(-a + 1 : a - 2 : 1)
```

5 利用例

最後に, 自身の研究に関連する話題での Magma による計算例を少しだけ紹介する. 筆者が最初に Magma に興味を持ったのは, 前節でも紹介した modular symbol の計算を行えると聞いたからであった. 筆者は楕円曲線の岩澤理論の周辺を主に研究しているが, その中心的な研究対象の 1 つである ($\mathbb{Q}$ 上の) 楕円曲線の p 進 L 関数は, 適当な Dirichlet 指標で twist した L 関数の特殊値を p 進的に補間して作られるものであるため, その構成や計算には modular symbol の計算が必要になる. 筆者は大学院生時代に, 八森祥隆氏と共同で楕円曲線の p 進 L 関数の計算を行ったことがあるが ([HM] 参照), 当時作成したプログラム[8]は, modular symbol の計算の一部を手計算 (または数表利用) に頼っていたため, 計算できる楕円曲線の種類が少なかった. その後, しばらくはそのような計算から離れていたのであるが, Magma を使えば苦労せずに modular symbol が計算できることを知り, 更に, Magma 上での p 進 L 関数の計算プログラムを Robert Pollack 氏から頂いたことをきっかけに, Magma の世界に足を踏み入れることとなった[9]. 現在は, 前節で紹介したような Mordell-Weil 群や Selmer 群などの具体例の計算機としての利用が主となっているが, 新たに自ら書いた[10] p 進 L 関数の計算プログラムなどを使って, これから紹介するような計算も行っているところである.

[8]PARI の C ライブラリを利用して書いていた.
[9]当初は, 当時筆者が勤めていた都立大の中村憲先生の Magma を使わせていただいた.
[10]正確には書きかけ. まだ公開できるようなものではない.

さて, E が $y^2+y=x^3+x^2-258x-2981$ という方程式で定義される楕円曲線[11]の 3 進 L 関数を計算してみると, それに付随する $\mathbb{Z}_3$ 上の多項式[12]は次数が 4 の monic であり, 更に, $((1+T)^3-1)/T$ でちょうど 1 回割り切れることがわかる (この例は [HM] で既に計算済み). このとき, 楕円曲線の (円分的) 岩澤主予想を仮定すると, E の Mordell-Weil rank について次のことが成り立つと予想される:

$$\mathrm{rank}E(\mathbb{Q})=0,\quad \mathrm{rank}E(K_n)=2\quad (n\geq 1) \tag{3}$$

ここで, K_n は $\mathbb{Q}$ の円分 $\mathbb{Z}_3$ 拡大の n 番目の中間体であり, $K_n=\mathbb{Q}(\cos\frac{2\pi}{3^{n+1}})$ と表される ($\mathrm{Gal}(K_n/\mathbb{Q})$ は位数 3^n の巡回群). このうち, $\mathrm{rank}E(\mathbb{Q})=0$ と $\mathrm{rank}E(K_n)=\mathrm{rank}E(K_1)$ $(n\geq 1)$ であることは, 岩澤主予想に関する加藤の結果から従うので, $\mathrm{rank}E(K_1)=2$ を確認すれば (3) が示される. それを Magma で直接確認してみる.

```
> E:=EllipticCurve([0,1,1,-258,-2981]); E;
Elliptic Curve defined by y^2 + y = x^3 + x^2 - 258*x - 2981 over Rational Field
> R<x>:=PolynomialRing(Rationals());
> K<a>:=NumberField(x^3-6*x^2+9*x-3);  // 円分 Z_3 拡大の 1st layer
> EK:=ChangeRing(E,K);                 // E を K 上に楕円曲線と見なす
> b,M,f:=MordellWeilSubgroup(EK:SearchBound:=200);
> b; M;
false
Abelian Group isomorphic to Z
Defined on 1 generator (free)
> //  Mordell-Weil rank は確定していないが, 1 以上であることはわかったということ
> P1:=f(M.1); P1;                      // 見つかった位数無限の点
(1/3*(20*a^2 + 25*a + 54) : 1/3*(-300*a^2 - 375*a + 436) : 1)
> s:=Automorphisms(K)[2];              // Gal(K/Q) の自明でない元
> P2:=Points(EK,s(P1[1]))[1]; P2;      // P1 の共役
(1/3*(-125*a^2 + 520*a - 66) : 1/3*(-1875*a^2 + 7800*a - 2239) : 1)
> Regulator([P1,P2]);
8.50124934997152543284172382835
```

最後の値が 0 でないので, 2 点はともに位数無限で独立であることがわかり, $\mathrm{rank}E(K_1)=2$ が示されたことになる. ($\mathrm{rank}E(K_1)\leq 2$ であることは, 加藤の結果からもわかるし, `TwoSelmerGroup(EK)` を実行してもわかる.) 同様に, [Gr] に挙げられている, 導手 195 の楕円曲線の円分 $\mathbb{Z}_2$ 拡大での振る舞いの例 (2 番目の中間体まで rank が増え続ける) なども計算で確認することができる.

Magma 上で楕円曲線の p 進 L 関数を計算するプログラムは, (比較的) 最近行った次の実例検証でも利用した.

定理. E を導手が 5000 未満の楕円曲線で, 2 で good ordinary reduction を持つようなものとする. もし, E と同種な楕円曲線 A で, $\mathbb{Q}$ の円分 $\mathbb{Z}_2$ 拡大体 F_∞ 上で $\mathrm{Sel}_2(A/F_\infty)$ が有限になるものが存在すれば, E に対する 2 進岩澤主予想は正しい.

楕円曲線の岩澤主予想はかなり一般的な仮定の下で解決されたとアナウンスされており[13], このような検証はもはやあまり意味を持たない気もするが[14], $p=2$ の場合は "μ 不変量" の

[11] モジュラー曲線 $X_0(11)$ の $\sqrt{5}$-twist

[12] p で ordinary reduction を持つ楕円曲線の p 進 L 関数は $\mathbb{Z}_p$ 上のある冪級数によって表せるが, 更にその冪級数は適当な unit で割ると多項式になる.

[13] Skinner and Urban, The Iwasawa main conjecture for GL_2, preprint.

[14] しかし, 上の preprint では $p=2$ の場合は扱われていないと思われる.

扱いなど微妙な点もあるので, 全く無意味という訳でもないと考えている. 証明は, 2 進 L 関数の計算と $\mathbb{Q}$ 上での 2-Selmer 群などの計算に, 2 次 twist での "λ 不変量" の変化公式という, $p = 2$ 特有の現象を組み合わせることで与えられる. なお, 定理の主張中の, 有限な $\mathrm{Sel}_2(A/F_\infty)$ を持つ A の存在は, Selmer 群側の "μ 不変量" に関する仮定で, 常に成り立つと予想されているものである. その予想についても Magma を使った計算を行っており, 導手が 1000 未満の楕円曲線のかなり多くについては正しいことを確認している. これらの結果はいずれまとめて発表したいと考えているが, そのためにはもう少し計算例を増やす必要があり, まだしばらくは Magma のお世話になる予定である.

参考文献

[Mag] W. Bosma, J. Cannon and C. Playoust, *The Magma algebra system. I. The user language*, J. Symbolic Comput. **24** (1997), 235–265.

[Cr] J. E. Cremona, "Elliptic Curve Database", `http://www.warwick.ac.uk/staff/J.E.Cremona/ftp/data/INDEX.html`

[Gr] R. Greenberg, *Iwasawa theory for elliptic curves*, in "Arithmetic Theory of Elliptic Curves", Lecture Notes in Math., vol. 1716, Springer-Verlag, 1999, pp. 51–144.

[HM] 八森祥隆, 松野一夫, 楕円曲線の p 進 L 関数の計算, 第 3 回津田塾大学整数論シンポジウム報告集, 1998 年.

[Se] J. P. Serre, *Propriétés galoisiennes des points d'ordre fini des courbes elliptiques*, Invent. math. **15** (1972), 259–331.

[Si] J. H. Silverman, "The Arithmetic of Elliptic Curves", Graduate Texts in Math., vol. 106, Springer-Verlag, 1986.

津田塾大学学芸学部数学科
〒 187-8577 東京都小平市津田町 2-1-1
email: `matsuno@tsuda.ac.jp`

有限群論での MAGMA の利用

千吉良直紀
熊本大学大学院自然科学研究科

1 群論の復習

MAGMA では、さまざまな有限群を構成し、その性質を ”計算 ”することが出来る。有限群についてまずは少しだけ復習をしておく。

G を群とする。Jordan-Hölder の定理により、

$$G = N_0 \triangleright N_1 \triangleright \cdots \triangleright N_r = \{1\}, \quad N_i/N_{i+1} : \text{単純群}$$

という正規列 (composition series) があって、

$$N_i/N_{i+1} \qquad i = 0, \cdots, r-1$$

は（重複度も含めて）G によって定まり、正規列のとり方によらない。したがって有限群は単純群の ”積み重ね ”で構成される。単純群は以下のように分類されている。

定理 (有限単純群の分類定理). 有限単純群は次のいずれかと同型。

- 素数位数の巡回群　Z_p (p : 素数)
- 5 次以上の交代群　A_n ($n \geq 5$)
- Lie 型の単純群[1]（q : 素数べき）

  $A_\ell(q)$ ($q \geq 4$), $B_\ell(q)$ ($\ell \geq 3$), $C_\ell(q)$ ($\ell \geq 2$, $q \geq 3$), $D_\ell(q)$ ($\ell \geq 4$),

  $E_6(q)$, $E_7(q)$, $E_8(q)$, $F_4(q)$, $G_2(q)$ ($q \geq 3$),

  ${}^2A_\ell(q)$ ($\ell \geq 3$, $q \geq 3$), ${}^2D_\ell(q)$ ($\ell \geq 4$), ${}^3D_4(q)$, ${}^2E_6(q)$,

  ${}^2B_2(2^{2m+1})$ ($m \geq 1$), ${}^2F_4(2^{2m+1})$ ($m \geq 1$), ${}^2G_2(3^{2m+1})$ ($m \geq 1$),

  ${}^2F_4(2)'$
- ２６個の散在型単純群[2]

[1] ℓ, q によってはいくつか同型になるものもある。
[2] 右には名前を記している。

M_{11}, M_{12}, M_{22}, M_{23}, M_{24},	Mathieu
J_1, J_2, J_3, J_4,	Janko
Co_3, Co_2, Co_1,	Conway
Fi_{22}, Fi_{23}, Fi_{24}',	Fischer
He, $O'N$, Ly,	Held, O'Nan, Lyons
HS, McL, Ru, Suz	Higman-Sims, McLaughlin, Rudvalis, Suzuki
HN, Th,	Harada-Norton, Thompson
B, $\mathbb{M}$	Baby monster, Monster

アトラス [3] には単純群の構成法、極大部分群、指標表などの情報が書かれている。またアトラスのホームページ http://brauer.maths.qmul.ac.uk/Atlas/v3/ には MAGMA や GAP 用に書かれた群の定義などがあり、そのままダウンロードして使用することも出来る。

MAGMA には群に関するいくつかのデータベースがある。マニュアルの FINITE GROUPS の中にある DATABASES OF GROUPS を見ると

- Database of Small Groups
- Database of Perfect Groups
- Database of Almost-Simple Groups
- Database of Transitive Groups
- Database of Primitive Groups
- Database of ATLAS Groups

などがある。例えば、散在型単純群 J_1 は

```
> A:=ATLASGroup("J1");
> PermRepKeys(A);
[
Perm rep of degree 266,
Perm rep of degree 1045,
Perm rep of degree 1463,
Perm rep of degree 1540,
Perm rep of degree 1596,
Perm rep of degree 2926,
Perm rep of degree 4180
]
> G:=PermutationGroup(PermRepKeys(A)[1]);
> G;
Permutation group G acting on a set of cardinality 266
> ChiefFactors(G);
G
| J1
1
>
```

などとすればよい。

また、マニュアルの Overview の System Features の Databases of Structure Definitions の

- Database of Some Permutation Groups

も用いることが出来る。例えば

```
> load m24;
Loading "C:\Program Files\Magma\libs\pergps\m24"
M24 - Mathieu group on 24 letters - degree 24
Order 244 823 040 = 2^10 * 3^3 * 5 * 7 * 11 * 23; Base 1,2,3,4,5,6,7
Group: G
>
```

などとすればよい。

有限群の研究として、

- 予想の検証
- 個々の群の性質の研究
- 全ての群に成立するような性質を見出す

などがあり、さまざまな種類の群についての詳細な計算をすることが出来るので MAGMA は有用である。最近出版された [5] にも

"The range of examples in this area is rather limited if one restricts oneself to paper and pencil work, but can be greatly enhanced by using a computer algebra system GAP or MAGMA. $\cdots$"

と書かれている。計算機を利用することによって多くの情報を得ることが出来る。また、簡単に比較的位数の大きい群をいじったりすることも出来るので、群論の勉強にもなる。

しかしながら、具体的な群についての性質を調べる場合ではもちろんであるが、実際には詳しく理論的な考察をし、MAGMA や GAP などの計算を用いずに説明できることが必要である。

2 具体的な例

ここでは、具体的な例として交換子に関する Ore 予想（後述）を考えることにする。

群 G の交換子群 G' は交換子 $[x,y]=x^{-1}y^{-1}xy$ $(x,y\in G)$ で "生成された "部分群として定義される。交換子群は正規部分群である。実際に 1 つの交換子で表される元はどのくらいあるのであろうか？

例 . $G=A_5$ とする。各共役類の代表元 z が交換子で表せるかどうかを計算すれば良いので、

$$\sharp\{(x,y)\in G\times G \mid z=[x,y]\}$$

を求めればよい。手計算でもそれほど大変ではないが、計算機では一瞬である。MAGMA では交換子 $x^{-1}y^{-1}xy$ は (x,y) で表されることに注意する。

```
> G:=Alt(5);
> S:=Classes(G);
> S;
Conjugacy Classes of group G
-------------------
 [1]   Order 1          Length 1
       Rep Id(G)
 [2]   Order 2          Length 15
       Rep (1, 2)(3, 4)
 [3]   Order 3          Length 20
       Rep (1, 2, 3)
 [4]   Order 5          Length 12
       Rep (1, 2, 3, 4, 5)
 [5]   Order 5          Length 12
       Rep (1, 3, 4, 5, 2)
> [#{<x,y> : x in G, y in G | z[3] eq (x,y)} : z in S];
[ 300, 32, 63, 65, 65 ]
```

したがって A_5 のどの元も交換子で書けることが分かる。

上の例のように全ての元を動かして計算するには群の位数が大きくなると時間がかかってしまう。次のような定理がある。

定理 (Frobenius). $Irr(G)$ を規約指標全体の集合とする。このとき

$$\sharp\{(x,y) \in G \times G \mid z = [x,y]\} = |G| \times \sum_{\chi \in Irr(G)} \frac{\chi(z)}{\chi(1)}$$

例 . $G = W(D_5)'$（D_5 型の Weyl 群の交換子群）$\cong 2^4 : A_5$ とすると、$G = G'$ である。指標表を計算すれば１つの交換子では表現出来ない元があることがわかる。

```
> G:=DerivedSubgroup(CoxeterGroup("D5"));
> ChiefFactors(G);
G
|   Alternating(5)
*
|   Cyclic(2) (4 copies)
1
> t:=CharacterTable(G);
> [#G*&+[x[i]/x[1]:x in t]: i in [1..#t]];
[ 11520, 3072, 4992, 1024, 1152, 1024, 0, 1040, 1040, 864, 864, 1152 ]
```

最後の計算で 0 になるものがあるので、この 0 に対応する元が交換子で表すことが出来ない。したがって交換子群の交換子で ”生成された ”という部分は本質的に必要なものである。

Ore 予想とは次の予想である。

予想 . G を非可換有限単純群とする。G の任意の元は交換子で表される。

最近この予想が解決された。

定理 (Liebeck-O'Brien-Shalev-Tiep[4]). Ore 予想は成立する。

この証明でも MAGMA は小さい位数の群の場合の検証などに用いられている。

3 方程式の解集合

群上の方程式の解集合も具体的に計算することが出来る。ここでは

$$\sharp\{(x,y) \in G \times G \mid y = [x,y]^{[x^2,y]}\}$$

という方程式の解集合を考える。前節のように指標を使ってうまくこの個数を表すことが出来ないので、全ての元を動かして計算せざるを得ない。具体的な群で計算することによって予想を立て、実際に次のことが成り立つことが分かる。

命題 1. $\varphi(y) = \sharp\{(x,y) \in G \times G \mid y = [x,y]^{[x^2,y]}\}$ とおく。

- G が可解群であれば、φ は指標である。
- $G = S_n$ のとき、φ は一般指標である。
- $G = A_n$ のとき、φ は一般指標である。

注 . $G = L_2(13)$ では φ は一般指標にもならない。

4 群と組み合わせ構造

M_{24} の 24 点集合 Ω 上の置換表現を考えると、位数 2 の元は $2^8 1^8$ 型と、2^{12} 型の 2 つの型の置換になる。$2^8 1^8$ 型の位数 2 の元は全部で 11385 個あり、この元たちの固定点の集合（8 点集合）全体の集合 $\mathfrak{B}$ は、同じものが 15 個ずつ出てくるので、全部で 759 個になる。incidence structure $(\Omega, \mathfrak{B})$ が Steiner system $S(5,8,24)$ で自己同型群は M_{24}、$\mathfrak{B}$ に対称差で和を定義して 2 元体上の符号を考えると extended Golay code になる。extended Golay code は自己双対符号 (self-dual code) である。

```
> G:=PrimitiveGroup(24,3);
> ChiefFactors(G);
G
|   M24
1
> S:=Classes(G);
> S[2];
<2, 11385, (1, 24)(3, 15)(4, 19)(6, 22)(7, 13)(8, 21)(14, 17)(18, 20)>
> b:={i:i in [1..24]|i^S[2][3] eq i};
> b;
{ 2, 5, 9, 10, 11, 12, 16, 23 }
> B:={x:x in b^G};
> #B;
759
> I:=IncidenceStructure<24|B>;
> ChiefFactors(AutomorphismGroup(I));
G
|   M24
1
> C:=LinearCode(I,GF(2));
> C:Minimal;
[24, 12, 8] Linear Code over GF(2)
>
```

extended Golay code を用いて M_{24} の性質を詳しく調べることが出来る。その際符号の自己双対性も重要な役割をしている。他の群でもこのような符号が存在するかという問題がある。同様のことを群や共役類を代えてさまざまなものを計算することが出来る。

散在型単純群 J_2 は 100 点上の置換表現を考えると、J_2 の作用する自己相対符号が存在する。100 点への J_2 の作用の 1 点の固定部分群は $U_3(3)$ であるが、$U_3(3)$ の性質を使ってこの符号を実際に構成することが出来る。

定理 (C-Harada-Kitazume [2]). 自己同型群が $J_2:2$ となる $[100, 50, 10]$ 自己双対符号が存在する。

この問題に関連して一般に群が作用する自己直交符号について次のことがいえる。

定理 (C-Harada-Kitazume [1]). 群 G が Ω に作用するとする。$x \in G$ に対して $Fix(x) = \{i \in \Omega \mid i^x = i\}$ とおく。G が作用する自己直交符号 C が存在するならば、

$$C \subseteq \langle Fix(x) \mid 1 \neq x \in G, x^2 = 1 \rangle_{\mathbb{F}_2}^{\perp}$$

が成り立つ。

参考文献

[1] N. Chigira, M. Harada and M. Kitazume, Some self-dual codes invariant under the Hall-Janko group, J. Algebra **316** (2007) 578–590.

[2] N. Chigira, M. Harada and M. Kitazume, Permutation groups and binary self-orthogonal codes, **309** (2007) 610–621.

[3] J. H. Conway, R. T. Curtis, S. P. Norton, R. A. Parker and R. A. Wilson, "ATLAS of Finite Groups", Clarendon Press, Oxford, 1985.

[4] M. Liebeck, E. OBrien, A. Shalev, P. H. Tiep, The Ore Conjecture, J. European Math. Soc. **12** (2010) 939–1008.

[5] K. Lux and H. Pahlings: "Representations of Groups, A Computational Approach" Cambridge studies in advanced mathematics **124**, Cambridge, 2010.

モジュラー形式に関係ある不変式論[1]

大浦　学（高知大学理学部）

１９世紀の数学では不変式論は大きな部分を占めていたと思われます。中心問題の一つは様々なタイプの不変式環の有限生成性で、当時、不変式論の神様と呼ばれていた Paul Gordon が活躍していました。１９世紀も終わりに近づく頃、David Hilbert はあるタイプの不変式環が有限生成であることを鮮やかに証明します。その結果は Gordon の得た結果を含んでおり、Gordon はそれを見て、次のように述べたと言われています[2]。

”Das ist nicht Mathematik. Das ist Theologie”.

これは数学ではない、神学だ、という訳です。時は経って１９８４年の Kung-Rota の論文”Invariant Theory of Binary forms”, Bull.Amer.Math.Soc. は次の印象的な文章から始まっています。

”Like the Arabian phoenix rising out of its ashes, the theory of invariants, pronounced dead at the turn of the century, is once again at the forefront of mathematics”.

これらの文章は不変式論の持つ一面を表していると思うのであげておきました。

さて、今日お話しする内容の基本文献としては、僕自身が勉強した

Bernd Sturmfels
Algorithms in invariant theory.
Springer-Verlag, Vienna, 1993.

をあげておきます。この本の出版後、Gregor Kemper らにより発展され、そのアルゴリズムは Magma にも実装されています。

この講演で考えるのは、有限群の不変式論です。詳しく述べると、$GL_n(\mathbf{C})$ の有限部分群 G をとります。このとき、G は n 変数多項式環上に

$$A \circ f(\boldsymbol{x}) = f(A\boldsymbol{x})$$

で作用します。ただし、変数を $\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ とみて、$A\boldsymbol{x}$ は普通の行列の積です。次で定義される集合を G の不変式環と言います：

$$\mathbf{C}[\boldsymbol{x}]^G := \{f \in \mathbf{C}[\boldsymbol{x}] : A \circ f = f,\ \forall A \in G\}$$

まず、不変式の構成方法ですが、それは Reynolds Operator が知られています:

$$\begin{aligned} \mathbf{C}[\boldsymbol{x}] &\longrightarrow \mathbf{C}[\boldsymbol{x}]^G \\ f &\longmapsto f^* = \frac{1}{|G|} \sum_{A \in G} A \circ f \end{aligned}$$

[1]2010 年 10 月 10 日 ”Magma で広がる数学の世界”における講演のほぼ忠実な記録です。適当に改変されている部分もあります。

[2]C.Reid, ”Hilbert”.

ただし、Reynolds operator の結果が 0 となることもあります。ある特別な群に対しては、符号理論が応用される場合があります。これについてはこの研究集会の原田昌晃先生の講演を参照してください。この講演でも少し触れます。

上に述べた Hilbert の不変式環に関する論文は、1890 年, 1893 年に相次いで発表されています。Gordon の学生であった Emmy Noether は、Hilbert の影響も受けつつ次のような定理を証明します (1916)：$\mathbf{C}[\boldsymbol{x}]^G$ は有限生成で、

$$\mathbf{C}[\boldsymbol{x}]^G = \mathbf{C}[f^* : \ f \text{ は単項式で次数} \leq |G|]$$

である。Hilbert の第１４問題として知られている「群の不変式環が有限生成か否か」は、永田雅宜による否定的な結果が知られています (1958)。もっともその後もこの分野の研究は進んでいるようです。

上の Noether の定理を使って不変式環を見てみます。

$$G = \left\{ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \right\}$$

をとります。Noether の定理より $\mathbf{C}[x, y] = \mathbf{C}[x^*, y^*, (x^2)^*, (xy)^*, (y^2)^*]$
です。ただし

$$\begin{aligned} x^* &= 1/2\{x + (-x)\} = 0 \\ y^* &= 1/2\{y + (-y)\} = 0 \\ (x^2)^* &= 1/2\{x^2 + (-x)^2\} = x^2 \\ (xy)^* &= 1/2\{xy + (-x)(-y)\} = xy \\ (y^2)^* &= 1/2\{y^2 + (-y)^2\} = y^2 \end{aligned}$$

なので

$$\mathbf{C}[x, y]^G = \mathbf{C}[x^2, xy, y^2]$$

が得られます。ただし、このような計算は G の位数が大きいと破綻します。

有限群の不変式環の持つ著しい性質として Cohen-Macaulay 性

$$\exists \theta_1, \ldots, \theta_n \in \mathbf{C}[\boldsymbol{x}]^G \text{ s.t. } \mathbf{C}[\boldsymbol{x}]^G \text{ は free } \mathbf{C}[\theta_1, \ldots, \theta_n] \text{ 加群}$$

があげられます。特に

$$\mathbf{C}[\boldsymbol{x}]^G = \oplus_{i=1}^t \eta_i \mathbf{C}[\theta_1, \ldots, \theta_n]$$

を広中分解と呼びます。先の例でみると

$$\mathbf{C}[x, y]^G = \mathbf{C}[x^2, xy, y^2] = \mathbf{C}[x^2, y^2] \oplus xy\mathbf{C}[x^2, y^2]$$

です。ところで

$$\mathbf{C}[x^2, y^2] \oplus xy\mathbf{C}[x^2, y^2] = \mathbf{C} \oplus \underbrace{\mathbf{C}x^2 \oplus \mathbf{C}xy \oplus \mathbf{C}y^2}_{2\text{ 次}} \oplus \cdots$$

から分かるように、固定された次数のところでは有限次元部分空間となっております。次は Molien(1897) の定理です：$R = \mathbf{C}[\boldsymbol{x}]^G$ を次数ごとの $\mathbf{C}$ 上のベクトル空間として

$$R = R_0 \oplus R_1 \oplus R_2 \oplus \cdots$$

とみたとき、

$$\sum_d (\dim_{\mathbf{C}} R_d)\, t^d = \frac{1}{|G|} \sum_{A \in G} \frac{1}{\det(1 - tA)}$$

が成り立つ。

実際の Magma の計算を見る前に言葉をまとめておくと

$$\mathbf{C}[\boldsymbol{x}]^G = \mathbf{C}[x^2, xy, y^2] = \mathbf{C}[x^2, y^2] \oplus xy\mathbf{C}[x^2, y^2]$$

に対して

$$\begin{aligned}
\text{Fundamantal Invariants} &= x^2, xy, y^2 \\
\text{Primary Invariants} &= x^2, y^2 \\
\text{Secondary Invariants} &= 1, xy \\
\text{Molien Series} &= \frac{1+t^2}{(1-t^2)} \\
&= 1 + 3t^2 + 5t^4 + 7t^6 + 9t^8 + \cdots
\end{aligned}$$

です。この例を Magma で実行すると次のようになります。

```
> G:=MatrixGroup<2,Rationals()|[[-1,0],[0,-1]]>;
> R:=InvariantRing(G);
> PrimaryInvariants(R);
[
    x1^2,
    x2^2
]
> SecondaryInvariants(R);
[
    1,
    x1*x2
]
> M:=MolienSeries(G);
> M;
(t^2 + 1)/(t^4 - 2*t^2 + 1)
> S<t>:=PowerSeriesRing(IntegerRing());
> S!M;
1 + 3*t^2 + 5*t^4 + 7*t^6 + 9*t^8 + 11*t^10 + 13*t^12 +
  15*t^14 + 17*t^16 + 19*t^18 + O(t^20)
```

次にジーゲルモジュラー形式と不変式論の関係についてお話しします。よく知られた例から始めます。上半空間 $\mathbf{H}_1$ とモジュラー群 Γ_1

$$\begin{aligned}
\mathbf{H}_1 &= \{\tau = x + iy \in \mathbf{C} : \ y > 0\}, \\
\Gamma_1 &= SL(2, \mathbf{Z})
\end{aligned}$$

に対して、正則関数 $f:\ \mathbf{H}_1 \longrightarrow \mathbf{C}$ で

$$f(\tau) = a_0 + a_1 q + a_2 q^2 + \cdots, \quad q = \exp 2\pi\sqrt{-1},$$
$$f\left(\frac{a\tau+b}{c\tau+e}\right) = (c\tau+d)^k f(\tau), \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_1$$

を満たすものを重さ k のモジュラー形式と呼びます。ここでターゲットにするのは重さ $k = 0, 1, 2, \cdots$ のモジュラー形式が $\mathbf{C}$ 上生成する次数付き環

$$A(\Gamma_1) = A_0 \oplus A_1 \oplus A_2 \oplus \cdots$$

です。古典的な結果として次が知られています。

$$A(\Gamma_1) = \mathbf{C}[E_4, E_6] \cong \text{重さ}4,\ \ 6\text{の}2\text{変数多項式環}$$
$$\sum_d (\dim A_d)\, t^d = \frac{1}{(1-t^4)(1-t^6)}$$

ところで重さ 4 の Eisenstein 級数 E_4 はテータ関数を使って

$$E_4 = \theta_{00}(2\tau)^8 + 14\theta_{00}(2\tau)^4\theta_{10}(2\tau)^4 + \theta_{10}^8(2\tau)$$

という表示が可能です。原田先生の講演に現れた符号の重み多項式

$$W_{e_8} = x^8 + 14x^4y^4 + y^8$$

の式がここに現れます。このことは後で触れます。重み多項式 W_C にも種数 (genus g) の概念あって、一般種数では 2^g 変数の多項式となります。

ジーゲルは今述べた古典的なモジュラー形式の拡張をします (1939)。

$$\mathbf{H}_g = \{\tau = x + iy \in \mathrm{Mat}_{g\times g}(\mathbf{C}) :\ \tau = {}^t\tau, y > 0\},$$
$$\Gamma_g = Sp_g(\mathbf{Z})$$

に対して、$\mathbf{H}_g$ 上の正則関数 f で

$$f((a\tau+b)(c\tau+d)^{-1}) = \det(c\tau+d)^k f(\tau),\ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_g$$

を満たすものを重さ k のジーゲルモジュラー形式と言います。ここでも問題としたいのは

ジーゲルモジュラー形式が生成する環 $A(\Gamma_g)$ を定めよ

というものです。$A(\Gamma_g)$ の構造に関しては井草準一 (g=2, 1962), 露峰茂明 (g=3, 1986) の結果が知られています。ここでは Bernhard Runge (1993–1996) によるアプローチを見てみます。テータ関数

$$\theta_{ab}(\tau) = \sum_{n\in\mathbf{Z}^g} \exp 2\pi\sqrt{-1}\left\{\frac{1}{2}{}^t(n+\frac{a}{2})\tau(n+\frac{a}{2}) + {}^t(\tau+\frac{a}{2})\frac{b}{2}\right\}, \tau \in \mathbf{H}_g, a, b \in \mathbf{F}_2^g$$

の変換公式を $\theta_{a0}(2\tau)$ の場合に見ると $GL_{2^g}(\mathbf{C})$ の有限部分群 H_g が出てきます。元の群との関係は、ある合同部分群 $\Gamma_g^*(2,4)$ があって

$$\Gamma_g/\Gamma_g^*(2,4) \cong H_g/\pm 1$$

となります。この多項式の変数にテータ関数を代入するという写像は

$$\begin{aligned}\mathbf{C}[x_1,\ldots,x_{2g}]^{H_g} &\longrightarrow A(\Gamma_g)^{(2)}\\ f(\boldsymbol{x}) &\longmapsto f(\theta_{a0}(2\tau))\end{aligned}$$

を与えます。右側の (2) は重さが偶数のもののみ考えるという意味です。実は

$$A(\Gamma_g)^{(2)} \cong \left(\mathbf{C}[\boldsymbol{x}]^{H_g}/(\text{theta relations})\right)^N$$

が成り立ちます。N は商体内での正規化を表します。符号理論との関係は $d \equiv 0 \pmod 8$ のとき

$$\mathbf{C}[\boldsymbol{x}]_d^{H_g} = \langle W_C^{(g)} : C = \text{self-dual doubly-even of length } d\rangle$$

で与えられます。この事実により原田先生の講演に出てきた重み多項式 W_{e_8} と、E_4 のテータ関数を用いた表示の関連が説明されます。以上が Runge の結果でした。我々は Runge のアプローチに従い、$g \geq 4$ のとき、小さい重さのベクトル空間の構造、テータ関係式など得てきました。(Freitag-Oura 2001, Oura-Poor-Yuen 2008, Oura-Salvati Manni 2008)

さて、Γ_g と H_g が対応していると考えてみると、次のような対応表ができます。

$$\Gamma_g \longleftrightarrow H_g$$

$$\Delta = \{\begin{pmatrix} A & B \\ O & D\end{pmatrix} \in \Gamma_g\} \longleftrightarrow \Lambda = \left\{ \left(\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline & & & \\ * & & * & \\ & & & \end{array}\right) \in H_g \right\}$$

$$\sum_{\Delta\backslash\Gamma\ni\begin{pmatrix} A & B \\ C & D\end{pmatrix}} \det(C\tau+D)^{-r} \longleftrightarrow \sum_{\Lambda\backslash H_g\ni A} A\circ(x_1^r)$$

$$\text{Eisenstein 級数} \longleftrightarrow \text{E-多項式}$$

結局のところ、E-多項式は Reynolds operator を特別な多項式 x_1^r に施したもの、にたどり着きました。以上の状況を unitary reflection group（不変式環が重み付き多項式環となる。Shephard-Todd により No.1 から No.37 に分類されている）の場合に計算してみました。まだ結果は不完全ですが、次のようになります。矢野茂雄さん（高知大）と共同で、No.1 から No.37 のそれぞれの群 G に対して、

$$\mathbf{C}[(x^r)^* : \ r = 0,1,\ldots] \subset \mathbf{C}[x,y,\ldots]^G$$

を調べ、もとの不変式環と一致するか否かを調べたものです。

No.	コメント	$\mathbf{C}[\boldsymbol{x}]^{G_i}$ と一致？
1	S_n	Yes
2	$G(m,n,p)$	?
3	$\mathbf{Z}/m\mathbf{Z}$	Yes
4		No
5		No
6		Yes
7		No
8	H_1	Yes
9	$\langle H_1, \eta_8 \rangle$	Yes
10		Yes
11		No
12		No
13		No
14		No
15		No
16		No
17		Yes
18		No
19		No

No.	コメント	$\mathbf{C}[\boldsymbol{x}]^{G_i}$ と一致？
20		No
21		Yes
22		Yes
23		Yes
24		Yes
25		No
26		Yes
27		Yes
28	$W(F_4)$	Yes
29		Yes
30	$W(H_4)$	Yes
31	H_2	Yes
32		?
33		No
34		?
35	$W(E_6)$	No
36	$W(E_7)$	?
37	$W(E_8)$	?