数式処理 *Bulletin of JSSAC(2012)*
Vol. 18, No. 2, pp. 105 - 116

特集: 数式処理システム ～いま使うならこれだ～

計算機代数のためのソフトウェア Magma

原田昌晃* 木田雅成†

山形大学理学部／ＪＳＴさきがけ 電気通信大学大学院情報理工学研究科

1 Magma とは何ではないのか/何なのか

Magma はオーストラリアのシドニー大学 (The University of Sydney) で，John Cannon をリーダーとする計算機代数グループによって開発された代数構造計算のためのソフトウェアである．マイクロソフト Windows をはじめ，Mac OS X，Linux など通常使われているほとんどの OS 上で動作する．

この記事で Magma を紹介するにあたって，代数構造計算を専門とする Magma と Maple に代表される一般的な数式処理システムとの違いから始めることにしよう．実際，Maple などの数式処理システムにできて Magma にできないことはたくさんある．たとえば，

- 三角関数や指数関数などの関数を含んだ式の簡単化
- 導関数や不定積分を求めること

などを行うことは不可能である．逆に何ができるかというと，大まかにいえば代数的な構造に埋め込まれるような，数学的対象の計算である．具体例はあとであげるが，群，環，体などの代数系がその代表になる[1)]．つまり一般的な数式処理システムと重なる部分もあるが，代数学の領域に特化して，その領域を深くカバーしたシステムなのである．歴史的に言えば，Magma は群論のソフトウェアである Cayley を前身としているが，その後，例えば整数論の分野の KANT グループの成果を統合するなど，現在では代数学の計算にかかわる非常に広汎な分野をカバーする大きなシステムに成長している．また，計算代数の最新の成果を貪欲に取り込むことで，それぞれの分野において，現在知られている最先端のアルゴリズムまでが実装されている．したがって，よほど複雑な計算をしない限り，通常計算できると知られているものの多くはあらかじめコマンドとして用意されていると考えてよい．

*mharada@sci.kj.yamagata-u.ac.jp

†kida@sugaku.e-one.uec.ac.jp

1) もともと magma という言葉は Bourbaki の数学原論の中の「代数」の巻で一番はじめで定義されている概念で，演算をもつような集合をよぶ普通名詞である．

2 誰が Magma を使うのか

Magma が扱うことのできる分野を 5000 ページを越えるマニュアルの目次から抜き出すと，線形代数，加群，群論，可換環論（グレブナー基底を含む），環論，代数的整数論，表現論，代数幾何，数論幾何，保形形式，組合せ論，符号理論，暗号理論などがある．こうして実際に列挙してみると，一般的な数式処理システムとの守備範囲の違いが際だってくるのではないだろうか．その一方，ここで注目して欲しいのは，整数論の PARI/GP やグレブナー基底計算の Singular のようなある分野に特化したシステムと異なり，Magma は代数学の多くの分野をカバーしている統合ソフトウェアであるという点である．代数学の基礎をなすコアな構造を共通化することによって，例えば部分構造や商構造の統一した扱いを可能にしている．このことから想像できるように，Magma は計算対象を計算機の中で数学的な対象として実現している．したがって，その数学的な対象に対して，ある程度の理解がないと Magma を使いこなすことは難しい．その意味では，Maple などの大学の新入生でもある程度は使えると思われるシステムとは異なり，代数学の知識をある程度もった，あるいはそれを学びつつある人が研究や実験を行なうために使うシステムといってよいかもしれない．

3 最初の一歩

まず簡単な例で Magma の基本的な文法を確認する[2)]．

```
> 32123*1000;           // 文はセミコロンで終わる
32123000
> 2^100;
1267650600228229401496703205376
> q:=2^30 div 17;       // 代入文．結果は表示されない
> r:=2^30 mod 17; r;    // 表示するにはもう一文を付け加える
13
> 2^30 eq 17*q+r;       // 等号のテスト
true
```

ここで確認したいのは，

- 文はセミコロンで終わる
- 一行だけのコメントは //で指定する
- 代入文には := を使う
- 等価演算子は = ではなく eq である．同様に比較演算子は lt や gt である

ことである．

さて Magma で多項式を計算してみる．$(x+2)^{20}$ を計算してみると，

```
> (x+2)^20;
```

[2)] あとで紹介する Magma の web site には，オンラインで計算が実行できる「Magma Calculator」が用意されているので，実際にこれらのコマンドを試してみることができる．

```
>> (x+2)^20;
    ^
User error: Identifier 'x' has not been declared or assigned
```

となってエラーが出てしまう．ここで Magma のひとつの設計方針に出会うことになる．

- **計算を行う代数構造をあらかじめ明示的に指定する必要がある．**

今の場合だと

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

とすればよい．これは有理整数環上の多項式環 $\mathbb{Z}[x]$ を PZ と名前をつけて，その変数を x とするということである．以後 x を含む多項式を入力するとこの環の元として認識される．

```
> (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);
[
    <x + 2, 20>
]
```

このような計算ができるようになる[3)]．ここで $1 は直前の結果の参照を表す．

```
> (x+1)^4/3;

>> (x+1)^4/3;
          ^
Runtime error in '/': Argument 2 is not a unit

> (x+1)^4/(-1);
-x^4 - 4*x^3 - 6*x^2 - 4*x - 1
```

この例では $(x+1)^4/3$ の計算ではエラーが出てしまう．なぜなら 3 は $\mathbb{Z}[x]$ の単数ではないので，結果がこの環からはみ出てしまうのである．一方 -1 は単数だから $(x+1)^4/(-1)$ は問題なく計算できる．このように**どこの環で計算をしているのかを意識する**ことが大切である．これは一見面倒に感じるかもしれないが，複雑な代数構造を考えているときには逆に対象が明確にとらえられるという利点も実感できる．

さて，上の計算を可能にするには，いくつかやり方はあるが例えば次のようにすればよい．

```
> PQ<y>,u:=ChangeRing(PZ,Rationals()); PQ;
Univariate Polynomial Ring in y over Rational Field
```

[3)] 紙面のサイズの都合で出力を改行している場合があり，実際の出力と異なる場合がある．

新しく $\mathbb{Q}[y]$ という環が作られ，$u : \mathbb{Z}[x] \to \mathbb{Q}[y]$ という写像が作られる．

```
> u((x+1)^4);
y^4 + 4*y^3 + 6*y^2 + 4*y + 1
> u((x+1)^4)/3;
1/3*y^4 + 4/3*y^3 + 2*y^2 + 4/3*y + 1/3
```

となる．さらに $\mathbb{Q}[y]$ で計算を進めると

```
> g:=((2*y^2+4*y)^6+2)/2; g;
32*y^12 + 384*y^11 + 1920*y^10 + 5120*y^9 + 7680*y^8 + 6144*y^7
+ 2048*y^6 + 1
> Discriminant(g);
106008003172799993633779105136 64
> Factorization($1);

>> Factorization($1);
                ^
Runtime error in 'Factorization': Bad argument types
Argument types given: FldRatElt
```

g の判別式を因数分解しようとするとエラーが出てしまう．これは判別式が見かけは整数だが有理係数の多項式環の中で計算されているので，有理数として扱われることで問題が生じている．これを避けるには次のように判別式を整数環 Integers() の中に明示的にいれてやればよい．

```
> Factorization(Integers()!$1);
[ <2, 79>, <3, 13>, <11, 1> ]
```

以上で分かるように，Magma のコマンドはほとんどが名詞の形で省略がなく（例えば判別式は discrim とかではなくて Discriminant），いくつかの基礎となるコマンドを除いては大文字ではじまるのが特徴である．したがって，以降では個々のコマンドの説明は省略する．

この節では Magma の初心者の起こしやすいミスを通じて，Magma のいくつかの特徴をみた．次節以降，より本質的な Magma の特性に踏み込んでいく．

4 なぜ Magma を使うのか

前節の例をみると，Magma は敷居が高く使いにくいシステムと思われるかもしれない．しかしながら，Magma は 1994 年頃に開発が始まって以来，約 4,000 本の論文に引用された実績をもつ[4]．つまりそれだけのプロのユーザーが本格的に Magma を使っているということである．同時にこれは Magma の高い信頼性を物語っている．なぜ，Magma はこれだけ多くの数学者に使われているのだろうか．それは次にあげる Magma の設計思想によるところが大きいと考える．

- **代数学と圏論に基づいて設計されている**

[4] あとで紹介する Magma の web site では Magma を使った論文は [2]（またはマニュアル）を引用することを奨励している．

このことによって，計算機の中にあたかも数学的な対象が実在するかのような扱いが可能になっている．つまり，数学的対象はそれが孤立して存在するだけでなく，商構造や部分構造と関連し，それらは射によって結ばれている．まさにこれが Magma の中で実現されているのである．

このことを群論の簡単な例でみてみよう．

```
> G<a,b>:=Group<a,b | a^5, b^2, a^b = a^-1 >; G;
Finitely presented group G on 2 generators
Relations
    a^5 = Id(G)
    b^2 = Id(G)
    a^b = a^-1
> #G;
10
```

G は正五角形のシンメトリーの群（位数 10 の二面体群）である：

$$G = \langle a, b \mid a^5 = e, b^2 = e, b^{-1}ab = a^{-1} \rangle.$$

部分群と商群は次のようにして作られる．

```
> H,i:=sub<G|a>; H;
Finitely presented group H on 1 generator
Generators as words in group G
    H.1 = a
> i(H.1^2);
a^2
```

H は a で生成される巡回群である．H.1 にはその生成元が入っている．i は自然な単射 $H \to G$ である．H が正規部分群であることを確かめて，商群を作ると，

```
> IsNormal(G,H);
true
> Q, p:=quo<G|H>; Q;
Finitely presented group Q on 2 generators
Relations
    Q.1^5 = Id(Q)
    Q.2^2 = Id(Q)
    Q.1^Q.2 = Q.1^-1
    Q.1 = Id(Q)
> #Q;
2
```

商群 $Q = G/H$ が作られると同時に自然な全射 $p : G \to Q = G/H$ が作られる．

```
> p(a);
Q.1
```

```
> Kernel(p);
Finitely presented group on 2 generators
Generators as words in group G
    $.1 = a
    $.2 = Id(G)
> p(i(H.1));
Q.1
```

群論の初歩的な例ではあるが「計算機の中にあたかも数学的な対象が実在する」ような感じをつかめていただけたであろうか.

5 Magma を使った計算例

Magma でどのような計算ができるのかを知るには，一昨年開催の研究集会「Magma で広がる数学の世界」の報告集 [5] が役立つ．この研究集会では Magma を使っている研究者が集まり，初心者への紹介・チュートリアルから始め，群論，数論，組合せ論，暗号理論など様々な専門分野での Magma の使用例が紹介された．また，数論に関するプログラムや例が [3]， [4] にある．

ここでは，筆者の専門分野であるガロア理論と組合せ構造においての計算例を挙げる．これらを通じて，Magma が代数の様々な分野を扱える統合ソフトウェアであるという利点も明らかになるであろう．

5.1 ガロア理論

ガロア理論は方程式の根の置換の研究から始まったが，現代的にいえばガロア拡大とよばれる体の拡大とその自己同型群であるガロア群が主役であり，ガロア群の部分群と，部分体が 1 対 1 に対応するという「ガロアの基本定理」が重要である．Magma を使うと，この対応が非常に具体的に書ける．

ここでは $f = x^6 + x^4 - 2x^2 - 1$ のガロア群を計算しその分解体 S の部分体との対応を見る．

```
> _<x>:=PolynomialRing(Rationals());
> f:=x^6 + x^4 - 2*x^2 - 1;
> Factorization(Integers()!Discriminant(f));
[ <2, 6>, <7, 4> ]
```

判別式が平方数なので f のガロア群は 6 次交代群の部分群に同型であることがわかる．実際に計算してみると

```
> Gf:=GaloisGroup(f); Gf;
Permutation group Gf acting on a set of cardinality 6
Order = 12 = 2^2 * 3
    (1, 3, 2)(4, 6, 5)
    (2, 5)(3, 6)
> IsAbelian(Gf);
false
```

```
> [g‘subgroup : g in Subgroups(Gf) | g‘order eq 3];
[
    Permutation group acting on a set of cardinality 6
    Order = 3
        (1, 3, 2)(4, 6, 5)
]
> IsNormal(Gf,$1[1]);
false
```

となる．先にも述べたように，Magma の前身は計算群論のソフトウェアである Cayley であり，置換群の計算に非常に強い．小さい次数の多項式のガロア群を計算するソフトウェアは多くあるが，高い次数でもソフトウェア的な制限なしに計算できるのは Magma しかない．

ここでは，Gf には 6 次対称群の部分群として計算された f のガロア群が代入される．その群は位数 12 でアーベル群でなく，さらにその位数 3 の部分群は正規部分群でない．位数 12 の群の分類によれば，このような群は 4 次交代群に同型であり，同型写像も Magma が作ってくれる．

```
> IsIsomorphic(Gf,AlternatingGroup(4));
true Homomorphism of GrpPerm: Gf, Degree 6, Order 2^2 * 3 into
GrpPerm: $, Degree 4, Order 2^2 * 3 induced by
    (1, 3, 2)(4, 6, 5) |--> (2, 3, 4)
    (2, 5)(3, 6) |--> (1, 4)(2, 3)
```

ガロア群がわかったので，f の分解体を作り，その部分体を計算しよう．

```
> S<b>:=SplittingField(f); S;
Number Field with defining polynomial x^12 + 4*x^10 + 10*x^8
+ 34*x^6 - 7*x^4 + 98*x^2 + 49 over the Rational Field
> Factorization(PolynomialRing(S)!f);
```

最後の出力は省略したが，f が S で 1 次因子に分解される様子がわかる．これで 12 次の分解体 S が Magma の中に作られた．この体の自己同型群が方程式のガロア群と同型になる．

```
> G,A,tau:=AutomorphismGroup(S);
> G;
Permutation group G acting on a set of cardinality 12
Order = 12 = 2^2 * 3
    (1, 2, 4)(3, 8, 9)(5, 11, 6)(7, 12, 10)
    (1, 3, 5)(2, 6, 7)(4, 10, 8)(9, 12, 11)
> A;
Set of all automorphisms of S
```

G には置換群としてのガロア群が代入されていて，実際の体の自己同型群 A へは写像 tau が定義されている．このような一見複雑な定義にも理由があって，置換群の世界では部分群の計算が簡単だが，A のままでやるのは難しい．しかし一方で G の元での原始元 b の像を求めるのはその

ままでは困難で，それは G の元に A の元を対応させることで実現できるのである．

```
> G.1;
(1, 2, 4)(3, 8, 9)(5, 11, 6)(7, 12, 10)
> tau(G.1)(b);
1/22806*(-284*b^11 - 191*b^10 - 1138*b^9 - 295*b^8 - 3261*b^7
- 111*b^6 - 11090*b^5 - 908*b^4 + 2277*b^3 + 19089*b^2 - 42259*b
- 16576)
```

ガロアの基本定理の対応を見るために G の共役なものを除いた全ての部分群を求める．

```
> SG:=Subgroups(G); SG;
Conjugacy classes of subgroups
------------------------------

[1]     Order 1            Length 1
        Permutation group acting on a set of cardinality 12
        Order = 1
[2]     Order 2            Length 3
        Permutation group acting on a set of cardinality 12
        Order = 2
            (1, 12)(2, 9)(3, 7)(4, 11)(5, 10)(6, 8)
[3]     Order 3            Length 4
        Permutation group acting on a set of cardinality 12
        Order = 3
            (1, 2, 4)(3, 8, 9)(5, 11, 6)(7, 12, 10)
[4]     Order 4            Length 1
        Permutation group acting on a set of cardinality 12
        Order = 4 = 2^2
            (1, 12)(2, 9)(3, 7)(4, 11)(5, 10)(6, 8)
            (1, 6)(2, 10)(3, 4)(5, 9)(7, 11)(8, 12)
[5]     Order 12           Length 1
        Permutation group acting on a set of cardinality 12
        Order = 12 = 2^2 * 3
            (1, 2, 4)(3, 8, 9)(5, 11, 6)(7, 12, 10)
            (1, 12)(2, 9)(3, 7)(4, 11)(5, 10)(6, 8)
            (1, 6)(2, 10)(3, 4)(5, 9)(7, 11)(8, 12)
```

それぞれの群に対して，対応する S の部分体を計算してみる．

```
> FixedField(S,SG[2]‘subgroup);
Number Field with defining polynomial x^6 + 8*x^5 + 40*x^4 + 272*x^3
- 112*x^2 + 3136*x + 3136 over the Rational Field
> FixedField(S,SG[3]‘subgroup);
Number Field with defining polynomial x^4 + 8*x^3 + 40*x^2 + 160*x
+ 336 over the Rational Field
> FixedField(S,SG[4]‘subgroup);
Number Field with defining polynomial x^3 + 8*x^2 - 16*x - 64 over
the Rational Field
> FixedField(S,SG[5]‘subgroup);
Rational Field
```

逆の対応も次のように計算することができる．

```
> FixedGroup(S,FixedField(S,SG[4]‘subgroup));
Permutation group acting on a set of cardinality 12
    Id($)
    (1, 6)(2, 10)(3, 4)(5, 9)(7, 11)(8, 12)
    (1, 8)(2, 5)(3, 11)(4, 7)(6, 12)(9, 10)
    (1, 12)(2, 9)(3, 7)(4, 11)(5, 10)(6, 8)
Mapping from: GrpPerm: $, Degree 12 to GrpPerm: G
> IsIsomorphic($1,SG[4]‘subgroup);
true Homomorphism of GrpPerm: $, Degree 12, Order 2^2 into
GrpPerm: $, Degree 12, Order 2^2 induced by
    Id($) |--> Id($)
    (1, 6)(2, 10)(3, 4)(5, 9)(7, 11)(8, 12) |--> (1, 12)(2, 9)(3, 7)
    (4, 11)(5, 10)(6, 8)
    (1, 8)(2, 5)(3, 11)(4, 7)(6, 12)(9, 10) |--> (1, 6)(2, 10)(3, 4)
    (5, 9)(7, 11)(8, 12)
    (1, 12)(2, 9)(3, 7)(4, 11)(5, 10)(6, 8) |--> (1, 8)(2, 5)(3, 11)
    (4, 7)(6, 12)(9, 10)
```

代数学の教科書で学ぶ抽象的な対応がこれだけ具体的に書けることに新鮮な驚きを感じられる方もおられるのではないだろうか．Magma が計算機の中に数学的対象を実在させていることはこのような例に端的に現れている．

5.2 組合せ構造

上で Magma が計算機の中に数学的対象を実在させている例をみたが，組合せ構造などの有限な対象であれば，それがさらに強力に行なえることを実感できる．（組合せ）デザイン，グラフなどの組合せ構造を Magma で実現させるのは原始的な方法で簡単に行なえるが，計算できることは非常に幅広い．また，ある構造をグラフに関連付けて調べることは少なくないと思われるので，Magma でグラフを扱えるのは統合ソフトウェアである魅力を増すことになる．筆者の個人的な感触かもしれないが，Magma でデザインやグラフなどの組合せ構造を扱っているユーザーは多くないように感じている．ユーザーが増えることを願い，ここではデザインやグラフなどの組合せ構造においての Magma の使用例を少しだけ挙げたい．新たなコマンド（関数）を定義する方法も紹介する．

デザインはブロックの集合を与えることで構成できる．2-(7, 3, 1) デザインを例として挙げる．

```
> D1:=Design<2, 7|{1,2,3}, {1,4,5}, {1,6,7}, {2,4,7},
> {2,5,6}, {3,5,7}, {3,4,6} >; D1;
2-(7, 3, 1) Design with 7 blocks
> Design<2,7|IncidenceMatrix(D1)> eq D1;
true
```

2 つ目の例のように結合行列を与えて構成することもできる．この場合 D1 の結合行列から構成しているので 2 つのデザインは D1 と同じである．同様に，（無向）グラフは辺集合を与えることで構成ができる．ピーターセングラフ（Petersen graph）を例として見ていこう．

```
> G:=Graph<10|{1,2},{1,5},{1,6},{2,3},{2,7},{3,4},{3,8},{4,5},{4,9},
> {5,10},{6,8},{6,9},{7,9},{7,10},{8,10}>;
> Graph<10|AdjacencyMatrix(G)> eq G;
true
```

2 つ目の例のように隣接行列を与えて構成することもできる．デザインの結合行列やグラフの隣接行列から符号を構成することも簡単に出来る．特定の計算に特化したソフトウェアは色々と開発されているが，このような計算が出来るのも統合ソフトウェアである Magma の魅力の一つであるといえよう．

```
> C1:=LinearCode(D1,GF(2));
> C2:=LinearCode(ChangeRing(AdjacencyMatrix(G),GF(2)));
```

2-$(4n-1, 2n-1, n-1)$ デザインの存在と位数 $4n$ のアダマール行列の存在が同値であることが知られている．このことについて調べてみよう．

```
> destoHmat:=function(D)
function> M1:=IncidenceMatrix(D);
function> n:=Ncols(M1);
function> J:=Matrix(Integers(),n,n,[1 : i in [1..n^2]]);
function> M2:=VerticalJoin(
function>     Matrix(Integers(),1,n,[1 : i in [1..n]]),2*M1-J);
function> M3:=HorizontalJoin(Transpose(
function>     Matrix(Integers(),1,n+1,[1 : i in [1..n+1]])),M2);
function> return M3;
function> end function;
```

ここでは新しいコマンド（関数） destoHmat を定義している．デザイン D を入力するとその結合行列から構成されるアダマール行列を出力する．次に新しいコマンド Hmattodes を定義する．位数 $4n$ のアダマール行列 H と i, j を入力すると（第 i 行に関する）3-$(4n, 2n, n-1)$ デザインを構成しその（ポイント j に関する）導来デザインとして 2-$(4n-1, 2n-1, n-1)$ デザインを出力する．

```
> Hmattodes:=function(H,i,j)
function> HD1:=HadamardRowDesign(H,i);
function> HD2:=Contraction(HD1,Point(HD1,j));
function> return HD2;
function> end function;
```

2-$(4n-1, 2n-1, n-1)$ デザインの存在と位数 $4n$ のアダマール行列の存在が同値であることを，実際に $n = 2$ の場合に確かめてみよう．

```
> H1:=destoHmat(D1); [IsHadamard(H1), Nrows(H1) eq 8];
[ true, true ]
> Hmattodes(H1,1,1);
2-(7, 3, 1) Design with 7 blocks
```

D1 から位数 8 のアダマール行列 H1 を構成し，逆に H1 から 2-(7, 3, 1) デザインが構成されていることが確認できた．H1 から得られる 2-(7, 3, 1) デザインは全て D1 に同型になることを確認しよう（このパラメータのデザインは同型を除いて一意的に存在することが知られている）．

```
> {IsIsomorphic(D1,Hmattodes(H1,i,j)):i,j in [1..8]};
{ true }
```

Magma はアダマール行列，デザイン，グラフなどの組合せ構造の同型判定や自己同型群の計算に威力を発揮し，組合せ構造の計算において Magma を使うメリットの一つと言える．

```
> AutH:=HadamardAutomorphismGroup(H1);
> AutD:=AutomorphismGroup(D1);
> AutG:=AutomorphismGroup(G);
> [#AutH,#AutD,#AutG];
[ 21504, 168, 120 ]
```

Magma の中では，カテゴリーが自然な形で定義されていて，その対象を扱うことはもちろん，ふたつの対象の間の射も扱うことができる．

```
> g1:=Sym(7)!(2,5,7)(3,4,6); g2:=Sym(7)!(1,3,2)(5,6,7);
> PSL:=ProjectiveSpecialLinearGroup(2,GF(7));
> f:=hom<PSL -> AutD | PSL.1 -> g1, PSL.2 -> g2>;
> [f(PSL) eq AutD, #Kernel(f) eq 1];
[ true, true ]
```

PSL から AutD への写像 f を定義し，これが同型写像になることを確認している．つまり，デザイン D1 の自己同型群は射影特殊線形群 PSL(2, 7) になることを確認している．

最後にグラフの計算を少し行なっておこう．次は，ピーターセングラフ G は連結な正則グラフであり，その直径が 2 で内周が 5 であることを確認している．

```
> [IsConnected(G), IsRegular(G), Diameter(G) eq 2, Girth(G) eq 5];
[ true, true, true, true ]
```

紙面の関係で詳しく紹介することは無理だが，グラフ理論の機能は充実をしていて（少なくともデザインやアダマール行列よりも），数多くの標準的なグラフが定義されていたり，非常に豊富なコマンドも用意されていて魅力的である．

6 Magma に関する情報源

Magma に関するより詳しい情報を得るために，まず第一に見るべきは Magma の web site

http://magma.maths.usyd.edu.au/magma/

である．Magma の購入・ダウンロードの仕方から，アップデートの情報などに加えてオンラインでマニュアルが参照できたり，後にのべる研究集会の情報などが豊富に掲載されている．

6.1 マニュアル

Magma の具体的な使い方を詳しく知るためには，やはり Magma のマニュアルを読む必要がある．ただし，マニュアルは PDF にすると全部で 5,000 ページを超える分量があり，すべてを読み通すことはなかなか難しい．しかし，概説である [1] などを参考に最初の “The magma language”，“Sets, sequences, and mappings”，“Basic rings and linear algebra” の 3 章はどのような分野でも必要と思われるので一読をおすすめしたい．

個々のコマンドを調べる場合には，Magma のコマンドラインから使えるオンラインのマニュアルがある．最近のバージョンの Magma をインストールしてあるのであれば，例えば，次のように入力すると web ブラウザが立ち上がり，rank を含むコマンドの一覧を表示してくれる[5]．

```
> ?rank
```

また，Magma のインストール前（購入前）でも，HTML 形式のマニュアルの “Index” の章を利用すれば，個々のコマンドを調べることができ，どのような計算ができるのかを調べられる．例えば，「`Rank(A) : Mtrx -> RngIntElt`」を選べばそこには “Given an m x n matrix A over a ring R, return the rank of A.” と説明があるので，行列の階数の計算ができることが分かる．

6.2 Magma 関係の研究集会

Magma Conference とよばれる研究集会が海外では 2005 年以降に限っても 7 件開催されている．一方，国内では，2010 年 10 月に九州大学において国内最初の Magma を主題にした研究集会「Magma で広がる数学の世界」が数学セミナーの記事 [6] をきっかけに開催された．そこでは巨大なシステムである Magma の全体像に近づくことを目標に Magma に関する情報交換が行われた．その報告集が [5] である．今年も同様な研究集会の開催を 7 月に開催する．今後も継続的に開催していくことで国内での Magma ユーザーが増え，Magma を使った数学が益々発展，深化することを願っている．

参 考 文 献

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

[2] Bosma, W., Cannon, J., and Playoust, C.: The Magma algebra system. I. The user language, *J. Symbolic Comput.* **24**, No. 3-4, pp. 235–265, 1997.

[3] 木田雅成: 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』, pp. 59–79, 2009.

[4] ———: 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, pp. 107–116, 2010.

[5] 木田雅成・原田昌晃・横山俊一 (編): Magma で広がる数学の世界,『COE Lecture Note Vol. 29』, 九州大学, 2010.

[6] 原田昌晃・木田雅成: Magma,『数学セミナー』, 2010 年 9 月号, pp. 44–47.

[5] ブラウザが立ち上がらない場合は，マニュアルの `SetHelpUseExternalBrowser` の項を参考に設定をして欲しい．