1 Mathematica と初等整数論

前回に引き続き，Mathematica を用いた数学的実験を行う。Mathematica を用いると，手計算ではかなりの時間がかかる計算を，一瞬で実行することができる。そうした結果から，数学的に新しい結果が見出される場合がある。

Mathematica で整数を扱うための命令をいくつか紹介しておこう。(前回紹介したものも含む。以下のもの以外にも様々な命令があるので，詳しくはヘルプを参照すること。)

[整数の割り算] (小学生割り算)

`Quotient[m, n]` は，`n` と `m` の整数商を与える。また，`Mod[m, n]` は，`m` を `n` で割った余りを与える。

例：$7 \div 3 = 2$ 余り 1

`Quotient[7,3]` は商 `2` を返し，`Mod[7,3]` は余り `1` を返す。

[整数のベキの剰余]

`PowerMod[a, b, n]` は，a^b を `n` で割った余りを与える。

(`PowerMod[a, b, n]` は，`Mod[a^b, n]` と同じ答を与えるが，効率がよくなる。)

[最大公約数] (Greatest Common Divisor)

`GCD[m, n,..。]` は，整数 `m, n, ...` の最大公約数を返す。

[最小公倍数] (Least Common Multiple)

`LCM[m, n,..。]` は，整数 `m, n, ...` の最大公約数を返す。

[素因数分解]

`FactorInteger[n]` は，整数 `n` の素因数をこれらの指数とともにリストとして返す。

例：$2434500 = 2^2 \cdot 3^2 \cdot 5^3 \cdot 541^1$ を `FactorInteger` を用いて計算すると，次のように出力される。

`FactorInteger[2434500]` ⇒ `{{2,2},{3,2},{5,3},{541,1}}`

[素数を呼び出す]

`Prime[n]` は，第 `n` 番目の素数を与える。

例： `Prime[1]` ⇒ `2`, `Prime[2]` ⇒ `3`, `Prime[3]` ⇒ `5`,

[素数判定]

`PrimeQ[n]` は，`n` が素数の場合には `True` を，そうでない場合には `False` を与える。

[n を法とする方程式]

`Solve` では，自然数 n に対して，mod n での方程式を解くこともできる。

例：$5x \equiv 6 \mod 7$ を満たす $x \in \{0,1,2,3,4,5,6\}$ を求める。

```
Solve[{5 x == 6, Modulus == 7}, x, Mode -> Modular]
    ⇒ {{Modulus -> 7, x -> 4}}
```

解がない場合には，“`x -> **`” という部分が出てこない。

2 合同式

前回扱った「mod m の掛け算」とは，要するに自然数 m で割った余りのみに注目した計算であった。ここでは，前節で紹介した Mathematica の命令を利用して，mod m の計算を行う例を紹介する。

2.1 m を法とした 1 次方程式

Mathematica には，方程式を解くための `Solve` 命令があった。前節に紹介したように，`Solve` 命令で「mod m」での解を求めることもできるが，それには限界がある。以下の例を見ていただきたい。

[例 1] $5x \equiv 6 \mod 7$ (基本的な例)

入力： `Solve[{5 x == 6, Modulus == 7}, x, Mode -> Modular]`

出力： `{{Modulus -> 7, x -> 4}}`

[例 2] $10x \equiv 12 \mod 14$ (解が複数ある例)

入力： `Solve[{10 x == 12, Modulus == 14}, x, Mode -> Modular]`

出力： `{{Modulus -> 14, x -> 4}, {Modulus -> 14, x -> 11}}`

[例 3] $12x \equiv 14 \mod 16$ (解がない場合)

入力： `Solve[{12 x == 14, Modulus == 16}, x, Mode -> Modular]`

出力： `{{Modulus -> 2}}`

[例 4] $x^2 \equiv 1 \mod 15$ (2 次方程式でも解ける)

入力： `Solve[{x^2 == 1, Modulus == 15}, x, Mode -> Modular]`

出力：
```
{{Modulus -> 15, x -> 1}, {Modulus -> 15, x -> 4},
 {Modulus -> 15, x -> 11}, {Modulus -> 15, x -> 14}}
```

上記のように小さい自然数に対してはうまく計算できるが，扱おうとする自然数が大きくなるにつれて，うまく計算できない場合がでてくる。

[例 5] $123456789x \equiv 234567890 \mod 345678901$

この方程式は数学的には解があるのだが，上の例と同様に入力しても，Mathematica Version 4 では答えは出力されない。(Version 4 ではダメでも，より新しい version のものなら答えが求められることもある。)

例 5 については，数学的に等価な問題に書き換えることで，Mathematica Version 4 でも解を求めることができる。これについては，後ほど課題の形で扱うことにする。やり方はいくつか考えられるが，今回の課題では，時節で紹介する `PowerMod` という命令を利用した方法を紹介する。

2.2 m を法としたベキ乗

次節で説明する RSA 暗号の計算を Mathematica で実行する際には，`PowerMod` という命令が有用である。使用例として，次の定理が成立することを確認してみよう。

定理 (フェルマーの小定理)
p を素数, a を p と互いに素な整数とすると，次が成立する。

$$a^{p-1} \equiv 1 \bmod p$$

例えば $p = 5$ に対して定理が成り立つことを確認してみよう。

```
In[1]:= p=5
Out[1]= 5
In[2]:= Table[PowerMod[a,p-1,p],{a,1,20}]
Out[2]= {1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0}
```

a が $p = 5$ と互いに素でないところ ($a = 5, 10, 15, 20$) では 0 で，他は全て 1 になっている。

3 暗号化技術と整数論

3.1 現代社会と暗号 – 情報セキュリティ技術としての暗号

軍事・外交用の特殊技術 $\longrightarrow$ 電子的情報伝達におけるセキュリティ確保の技術

- 電子的情報の偽造防止
- コンピュータ上の商取引の安全確保
  1. 取引内容の偽造防止
  2. 取引内容の機密保持
  3. コンピュータ上での本人の確認（電子印鑑・ディジタル署名）
- 電子マネー
- 個人情報・プライバシー保護

以上のようなことが問題となる理由として, 電子的情報に特有の次のような特徴がある：

- コピーが容易であり, しかも, コピーとオリジナルとの差がない
- 改変されても証拠が残らない
- 情報の作成者・発信人が特定できない。

電子的情報に見られるこれらの弱点を克服する技術の基礎を学ぶのが, この章の目的である. [1]

辻井重男：『暗号』, 講談社選書メチエ 73, 講談社, 1996。
では, 現代におけるこのような情報セキュリティの必要性をふまえて, 暗号の歴史から最近の公開鍵暗号・電子署名などまで平易に解説されている（初期の版では, p.217 の図と p.222 の図が入れ違っているので注意)。

3.2 公開鍵暗号のアイディア

さて, 3.1 で述べたような要請に応えるための技術の核心になるアイディアとして「公開鍵暗号」, 「ゼロ知識証明」という 2 つのアイディアがある。ここでは「公開鍵暗号」についてその考え方を説明する。

一例として, 電子メールをやりとりしながらメールの内容が他人に漏れず信書の秘密が保たれるようにするには, どうしたらよいか, ということを考えよう。コンピュータネットワークにはハッカーがいて, あなたに来たメールを盗み読みするかも知れない。これは, 十分あり得る危険である。ハッカーに盗み読みされても秘密が保たれるためには, あなたに来るメールを全て暗号化しておく必要がある。

ここで問題なのは, あなたにメールを送ってくる人は一人だけではない。たとえば, 商売をやっていて注文をメールで受けていれば, 見ず知らずの人からも注文がやってくる（やってきてくれなければ困る）のだ。しかし, 見ず知らずの人間ともあらかじめ暗号を取り決めておくことなどは, 全くできない相談である。そもそも暗号にならないであろう。だが,

1. 暗号化鍵（= 暗号化ルール）が電話番号のように公開されたデータベースに登録されていて, それを用いれば誰でもあなたに暗号化されたメールを送れるが,

2. その暗号を解くことは秘密の復号化鍵（= 原文に戻すルール）を知るあなたにしかできない,

このような暗号システムがあったらどうだろう。このような都合の良い暗号を **「公開鍵暗号」** といい, このアイディアは 1976 年に Deffi と Hellman によって提唱された。

しかし, 暗号化のルールを逆にたどれば元の文が復元できるはずで, 暗号化鍵を他人に知らせながらしかも復号化させない, このようなことが可能なのであろうか？そこが, 数学の出番なのである。

文章が数値化されて整数値の列として表現されているとしよう。暗号化とはこのようにして得られた整数に新しい整数を対応させるルールである, 数学的に言うと, 写像である：

$$\text{暗号化鍵} = \text{暗号化写像 } f : \mathbb{Z} \longrightarrow \mathbb{Z}.$$

ただし, 暗号化写像 $f : \mathbb{Z} \to \mathbb{Z}$ は整数全てに対して定義されている必要はなく, 符号化に利用される一定個数の整数に対して定義されていればよい。そこで, 符号化に利用される整数の（有限）集合を A と記すと, 暗号化写像 f は,

$$f : A \to \mathbb{Z}, \quad m \in A \text{ が平文のとき, } f(m) \in \mathbb{Z} \text{ が暗号文}$$

[1] 電子的情報の伝達過程で紛れ込むエラーに対する対策の問題はここでは扱わない。それは, 「符号理論」, とくに「誤り訂正符号の理論」と呼ばれ, 有限体の代数学・代数幾何学の応用分野である。

と A 上で定義されているとして良い。

暗号化写像 $f : A \to B$ の逆写像 $g : B \to A$ が, すなわち, 暗号文 $f(m)$ から平文 m を復元するルール（= 復号化鍵）に他ならない。したがって, 公開鍵暗号を実現させるには, 暗号化写像として, それ自体は容易なものだがその逆写像は（あらかじめ答を知っていない限り）決して求められない（ないしは, 実用にならないほど莫大な時間を掛けないと求められない）ような写像を用いればよいことになる。このように, それ自体は易しいのにその逆写像は難しいような写像を **一方向写像 (one way map)** という：

一方向写像

$A \underset{g}{\overset{f}{\rightleftarrows}} B$	やさしい 難しい

現在のところ, 整数論から現れる次の 2 つの困難な問題が有用な一方向写像の構成に利用されている。

- 巨大な整数の素因数分解
- 離散対数の計算

次節では, このうち, 巨大な整数の素因数分解の困難さを利用した RSA 暗号を紹介する。

3.3 RSA 暗号

この暗号は リヴェスト (Rivest), シャミア (Shamir), アドレマン (Adleman) により考案され, その頭文字をとって **RSA 暗号** といわれる。それは, 次のように実に簡単なものである。

RSA 暗号の仕組み

下準備 ：

1. 2 つの素数 p, q をとり, $m = pq$ とおく。
2. $p-1$ と $q-1$ の最小公倍数 $L = LCM(p-1, q-1)$ を計算しておく。
3. $L = LCM(p-1, q-1)$ と互いに素な自然数 k_1 (すなわち，L との最大公約数が 1 となるような k_1) を定め,

$$k_1 k_2 \equiv 1 \bmod L$$

となる自然数 k_2 を計算しておく。

暗号化鍵（公開） m, k_1： 暗号化関数 $f : i \longmapsto i^{k_1} \bmod m.$

復号化鍵（秘匿） k_2： 復号化関数 $g : c \longmapsto c^{k_2} \bmod m.$

この暗号システムでは, 符号化に利用される整数の集合 A として, $\{i \in \mathbb{Z} \,|\, 0 \le i \le m-1\}$ (もっと理論的な表現をすれば, $\mathbb{Z}/m\mathbb{Z}$) をとって考えている。この f によって暗号化された暗号文を g によって復号化できることは，次の定理による。

定理　p, q が異なる素数ならば、任意の整数 a に対して次が成り立つ。

$$a^{kL+1} \equiv a \bmod pq \quad (L \text{ は } p-1 \text{ と } q-1 \text{ の最小公倍数であり，} k \text{ は任意の整数})$$

この定理の証明には「Fermat の小定理」が用いられるのであるが，ここではこれ以上は述べないことにする。

以上の考え方にしたがって，公開鍵，秘密鍵を作ってみよう。まず用意するのは2つの素数 p, q であるが，ここでは簡単のため 80 番目の素数，90 番目の素数を使ってみよう。

```
In[1]:= p = Prime[80]
Out[1]= 409
In[2]:= q = Prime[90]
Out[2]= 463
```

これを元に，積 $m = pq$, および $p-1, q-1$ の最小公倍数 $L = LCM(p-1, q-1)$ を計算する。

```
In[3]:= m = p*q
Out[3]= 189367
In[4]:= L = LCM[p-1,q-1]
Out[4]= 31416
```

次に，今求めた L と互いに素である自然数 k_1 を，何でもいいから 1 つ決める。ただし，k_1 はあまり小さくてはまずいので，ここでは 1000 番目の素数を使ってみよう。この選び方だと互いに素でなくなるかもしれないので，確認が必要である。(もし互いに素でないなら，別の素数で試してみればよい。)

```
In[5]:= k1 = Prime[1000]
Out[5]= 7919
In[6]:= GCD[L,k1]
Out[6]= 1
```

この k1 の mod m での逆数を求め，それを k2 とおく。

```
In[7]:= k2 = PowerMod[k1, -1, L]
Out[7]= 24287
```

これで，公開鍵 $(m, k_1) = (189367, 7919)$，秘密鍵 $k_2 = 24287$ が得られた。

次に，これらを使うと暗号化・複合化がちゃんとうまくいくことを，入力 $i = 1, 2, \ldots, 10$ に対して確認しておこう。まずは，入力データを，次のように作っておく。

```
In[8]:= tbl1 = Table[i, {i, 3001, 3010}]
Out[8]= {3001, 3002, 3003, 3004, 3005, 3006, 3007, 3008, 3009, 3010}
```

これを公開鍵 $(m, k_1) = (189367, 7919)$ で暗号化してみる。

```
In[9]:= tbl2 = PowerMod[tbl1, k1, m]
Out[9]= {8198, 120109, 54064, 32358, ..(中略).., 82069, 5381, 4065}
```

一見まったく関係ない数に変換されている。これを秘密鍵 $k_2 = 24287$ で複合化してみよう。

```
In[10]:= PowerMod[tbl2, k2, m]
Out[10]= {3001, 3002, 3003, 3004, 3005, 3006, 3007, 3008, 3009, 3010}
```

ちゃんと元のデータに戻ることが分かる。

4 今日の課題

【課題 1】(合同式)

合同式 $123456789x \equiv 234567890 \mod 345678901$ の解を，次の手順により求めよ。

(1) 123456789 の mod 345678901 での逆数 ($123456789a \equiv 1 \mod 345678901$ を満たす a) を求める。そのためには，「$a = (123456789)^{-1} \mod 345678901$」と書き直して，`PowerMod` を用いる。

(2) 上で求めた a と，合同式の右辺の値 234567890 との積を mod 345678901 で計算する。こうして得られる値が，求める解 x である。

(3) 念のため，求めた x がちゃんと解になっていることを確認しておこう。そのためには，求めた x と 123456789 との積を，mod 345678901 で計算すればよい。

【課題 2】(簡単な RSA 暗号)

各自の学籍番号に応じて，簡単な RSA 暗号を作成してみよう。

(1) 学籍番号の下 2 桁を `ab` とするとき，`1ab` 番目の素数を p, その次の素数を q として，積 $m = pq$, および $p-1$ と $q-1$ の最小公倍数 L を求める。 レポートには，p, q, m, L の値をそれぞれ記せ。

(2) 1000 番目あたりの素数から，L と互いに素であるものを選び k_1 とおく。k_1 の L を法とする逆数を k_2 とする。レポートには，選んだ k_1 の値と，対応する k_2 の値を記せ。また，k_2 を求める際の Mathematica の命令も記せ。

(3) 4 桁の自然数を 5 個並べたデータを作り，それを上で求めた公開鍵 (m, k_1) によって暗号化し，秘密鍵 k_2 で複合化されることを確認せよ。レポートには，元のデータ，暗号化の命令・結果，複合化の命令・結果の順に記せ。

これらの課題に対して，次ページのような出力が得られるように LaTeX で記述し，PDF ファイルを作成して CHORUS で提出せよ。ただし，枠で囲まれた部分については，該当する Mathematica の命令，計算結果を記入する。

計算機１・２ 第9回 課題

学籍番号: ******** 氏名: ********

平成19年6月19日

課題１ 合同式

(1) 123456789 の mod 345678901 での逆数を求める命令

`a =` ここには Mathematica に入力する命令を記す

出力結果 ここには Mathematica の出力

(2) (1) の `a` と 234567890 の mod 345678901 での積を求める命令

`x =` ここには Mathematica に入力する命令を記す

出力結果 ここには Mathematica の出力

(3) (2) の `x` と 123456789 の mod 345678901 での積を求める命令

ここには Mathematica に入力する命令を記す

出力結果 ここには Mathematica の出力

課題２ 簡単な RSA 暗号

(1) $p =$ 結果, $q =$ 結果, $m =$ 結果, $L =$ 結果

(2) 選んだ $k_1 =$ 結果, 対応する $k_2 =$ 結果

k_2 を求める命令： Mathematica に入力する命令

(3) 元のデータ： `tbl =` 元のデータを記す

暗号化の命令： Mathematica に入力する命令

暗号化の結果： 上の命令の出力

複合化の命令： Mathematica に入力する命令

複合化の結果： 上の命令の出力