一次元アフィン変換の軌道分解

学習院大学理学部数学科４年
猪口　修史

2009 年 2 月 3 日

目 次

第1章　目的

分数を循環小数に１０進数で変換する時のことを思い浮かべて頂きたい。
例えば、**1/7=0.142857142857...** 等が挙げられるだろう。
これは具体的には、循環小数の１周期を求めるのに、

$$\begin{aligned}\mathbf{1} \times 10 &= 7 \times 1 + 3 \\ 3 \times 10 &= 7 \times 4 + 2 \\ 2 \times 10 &= 7 \times 2 + 6 \\ 6 \times 10 &= 7 \times 8 + 4 \\ 4 \times 10 &= 7 \times 5 + 5 \\ 5 \times 10 &= 7 \times 7 + \mathbf{1}\end{aligned}$$

という風に、**左上の初期値が再び同じ値に戻るまで**を計算をしている。
ここで、初期値１に注目すると、(1,3,2,6,4,5) と変化しているのがわかる。

では、これに平行移動を加える、即ち１０倍して mod 7 をした後に
＋１をしてみたらどうなるだろうかと考えた。

つまり、

$$\begin{gathered}\mathbf{1} \times 10 = 7 \times 1 + 3\ \mathbf{+1} = 4 \\ \mathbf{4} \times 10 = 7 \times 5 + 5\ \mathbf{+1} = 6 \\ \mathbf{6} \times 10 = 7 \times 8 + 4\ \mathbf{+1} = 5 \\ \mathbf{5} \times 10 = 7 \times 7 + 1\ \mathbf{+1} = 2 \\ \mathbf{2} \times 10 = 7 \times 2 + 6\ \mathbf{+1} = 7 \equiv 0 (\mathrm{mod}\ 7) \\ \mathbf{0} \times 10 = 7 \times 10 + 0\ \mathbf{+1} = 1\end{gathered}$$

という作業（これが１次元アフィン変換）をする訳である。

そこから今回は、ここでの１周期の軌道
（今の例ならば初期値の１の変化 (1,4,6,5,2,0)）
の性質を、調べる対象 (「*/*を*進数で」の部分）を色々変えて調べてみることにした。

その中で、

- 不動点（軌道が点になる場合）になる条件
- それ以外の点に関して、類方程式（後述）が特殊な形をする条件

に注目し、場合分けをしてこの軌道の性質を研究することを目的とした。

第2章　原理

2.1　アフィン変換について

2.1.1　アフィン空間の定義

　アフィン空間 (affine space) とは、ユークリッド空間から計量の概念を取り去ったものである。
集合 A と体 K 上の n 次元ベクトル空間 V の組 (A, V) が、

1. $\forall P, \forall Q \in A$ なる (P, Q) に対し、$a \in V$ がただ一つ決まる。
   (これを $\mathbf{a} = \overrightarrow{PQ}$ と表す。)
2. $\forall P \in A, \forall a \in V$ に対し、$\overrightarrow{PQ} = \mathbf{a}$ を満たす $Q \in A$ がただ一つ存在する。
   (これを $Q = fa(P)$ あるいは $Q = P + a$ と記す。)
3. $\forall a, \forall b \in V$ に対し、$f_{\mathbf{b}} \circ f_{\mathbf{a}} = f_{\mathbf{a+b}}$

を満たすとき、(A, V) は**アフィン空間**であるという。

2.1.2　アフィン写像の定義

　アフィン写像とは、平行移動と線形写像を組み合わせることのできる写像のことで、射影変換の特殊な場合である。
二つのアフィン空間 $(A, V(A))$ と、$(B, V(B))$ に対して、
写像 $f : A \to B$ と線形写像 $V(f) : V(A) \to V(B)$ の組 $(f, V(f))$ を考えたとき、
線形性を持ち、かつ平行移動可能、即ち

1. $\forall a \in V(A)$ に対し、$\mathbf{a} = \overrightarrow{PQ}(P, Q \in A)$ ならば、$V(f)(\mathbf{a}) = \overrightarrow{f(P)f(Q)}$
2. $\forall P \in A, \forall a \in V$ に対し、$f(P + a) = f(P) + V(f)(a)$

を満たすものを、**アフィン写像**という。

2.1.3　アフィン変換

アフィン写像の中で、始域と終域が同じものを**アフィン変換**という。

例えば、$\forall x_s, \forall g_1 \in \mathbf{Z}$ に対して、$V(\mathbf{Z})$ として、[x_s全体の集合] と [$g_1 x_s + 1$ 全体の集合] の２つをとってやると、これらは当然ベクトル空間で、生成される２つの $(\mathbf{Z}, V(\mathbf{Z}))$ はアフィン空間の定義を満たす。
さらに、

$$g \cdot x_s : x_s(\in \mathbf{Z}) \to g_1 x_s + 1(\in \mathbf{Z}) \tag{2.1}$$

として定義される $g : \mathbf{Z} \to \mathbf{Z}$ はアフィン写像かつ、アフィン変換である。
この場合は１次元ベクトル空間 (つまり整数) について考えているので、**１次元アフィン変換**という。

2.2　類方程式について

2.2.1　類方程式

n, g_1 は正の整数かつ互いに素、さらに $\forall x_s (s = 1, 2, 3, \ldots) \in \mathbf{Z}/(n)$ について、$g \cdot x_s : x_s \to g_1 x_s + 1 (g : 1$ 次元アフィン変換$)$ とする。

いま、任意の n に対して、g_1 を一つとり、x_s を動かして

$$g^{u_k} \cdot x_s = x_s \tag{2.2}$$

をみたす**軌道の長さ** $u_k (k = 1, 2, 3, \ldots)$ と、**初期値** x_s を考える。

ただし、$\forall \alpha, \forall \beta \in s(\alpha \neq \beta)$ と $\forall a, \forall b \in \mathbf{Z}$ について、

$$g^a \cdot x_\alpha = g^b \cdot x_\beta \tag{2.3}$$

が成り立つ場合、$min(x_\alpha, x_\beta)$ の方のみを考える。
(この条件によって、軌道を集合としてみたとき、互いに素なもの同士だけが残る。)

ここで、

$$n = u_1 + u_2 + u_3 + \cdots + u_k \tag{2.4}$$

の形で n を軌道の長さ u_k の和の形で分割したものを、**n の類方程式**と定義する。
（条件 (2.3) から当然、$\sum_{r=1}^{k} u_r = n$ である。）

2.2.2 不動点の定義

式 (2.2) で、$u_k = 1$ のときを考える。
即ち、

$$g \cdot x_s = x_s \tag{2.5}$$

となる $\exists x_s \in x$ を、**不動点**と定義する。

2.2.3 不動点の解法

式 (2.5) を変形すると、

$$g \cdot x_s = x_s \Leftrightarrow (1 - g_1)x_s = 1 \tag{2.6}$$

とできる。

$x_s \in \mathbf{Z}/(n)$ だから、これは、

$$gcd(1 - g_1, n) = 1 \tag{2.7}$$

と同値である。

このことから、$U_n = (\mathbf{Z}/n\mathbf{Z})^\star$ としたとき、任意の n に対して、
$1 - g_1 \in U_n$ なる g_1 を一つとり、

$$g \cdot x_s \equiv x_s (modn) \tag{2.8}$$

を x_s について解けば不動点が求まる。

第3章　方法

3.1　手順

g, n の順番に固定して、x を動かして、

1. その軌道の中に不動点が存在するかどうか調べた。
2. 存在しない場合、その g, n での軌道を分類して、類方程式の形にして軌道の種類、長さを比較して特徴を調べた。

実際に計算するにあたっては、**SWI-Prolog** を計算機支援に使った。

3.2 SWI-Prologについて

Prologの正式名称は、”Programming In Logic”という。
非手続き型プログラミング言語の一つで、論理型言語の一種である。
1972年ごろにフランスのカルメラウアー氏とコワルスキー氏により考案された。
プログラムは一階述語論理に基づいてデータ間の関係を示す命題として記述され、処理系がそれらにパターンマッチング（ユニフィケーション）を施しながら、与えられた命題が成立するか再帰的手続きによって探索する。

Prologの最大の特徴は、複数解を持つことが出来る事である。その為、CやJava等とは構文形式が大きく異なるが、これらの性質を上手く利用すると、少ない記述量でより多くのアウトプットが得られることもある。人工知能におけるトップ・ダウン式の問題解決と相性が良いために、かつては人工知能研究とエキスパートシステムの実現のための主要言語として広く採用された。

厳密には、
「おおもとの知識データがすべて0と1にはっきり切り分けられていれば」
「その知識がカバーする範囲については」
「知識データに述べられていないことはすべて嘘（偽）であると仮定すれば」
という条件下であれば論理演算の答えを出してくれる（閉世界仮説）。
但しこれはつまり逆に言えば、実世界のように無限にひろがり、未知のものが多々あり、クリアな分類が難しい世界（フレーム問題）の役には立たないということであるとも言える。
今回使用したSWI-Prologは、オランダのUniversity of Amsterdamが開発したフリーソフトで、数あるProlog処理系の中でも非常に完成度が高いので、目的の為の計算機支援に最適であると考え、これを用いることにした。

3.3 プログラム

使用した具体的なプログラムを以下に示す。

```
/**動的変数領域の確保**/

:-dynamic s/1.
:-dynamic l/1.
:-dynamic m/1.
:-dynamic n/4.
:-dynamic cnt/2.
:-dynamic scnt/3.
:-dynamic chk/4.


/** retract1\1:最後に入れたものを取り出して、中身を消す**/

retract1(P):-retract(P),!.


/**ctr_\2...:カウンター述語の定義
           set(K,T):K 番目のカウンターを T にセットする。
           inc(K,T):カウンターを１増やす
           dec(K,T):カウンターを１減らす
           is(K,T) :カウンターの中身を第２変数に入れる
           banish(K,T):is の機能+別解を捨てる**/

ctr_set(K,T):-asserta(cnt(K,T)),!.
ctr_is(K,T):-cnt(K,T),!.
ctr_banish(K,T):-retract1(cnt(K,T)).
ctr_inc(K,T):-retract1(cnt(K,T)),T1 is T+1,asserta(cnt(K,T1)),!.
ctr_dec(K,T):-retract1(cnt(K,T)),T1 is T-1,asserta(cnt(K,T1)),!.
ctr_x(K,T,X):-retract1(cnt(K,T)),T1 is T+X,asserta(cnt(K,T1)),!.
```

```
/**sctr_\3...:基本は上のものと同じで、パターンマッチの機能を強化する為に
              番号を２変数にしてみた。
              上のと両方混在している。**/


sctr_set(K1,K2,T):-asserta(scnt(K1,K2,T)),!.

sctr_is(K1,K2,T):-scnt(K1,K2,T),!.

sctr_banish(K1,K2,T):-retract1(scnt(K1,K2,T)).

sctr_inc(K1,K2,T):-retract1(scnt(K1,K2,T)),T1 is T+1,
                   asserta(scnt(K1,K2,T1)),!.

sctr_dec(K1,K2,T):-retract1(scnt(K1,K2,T)),T1 is T-1,
                   asserta(scnt(K1,K2,T1)),!.

sctr_x(K1,K2,T,X):-retract1(scnt(K1,K2,T)),T1 is T+X,
                   asserta(scnt(K1,K2,T1)),!.



/**while_do\2:命題「Ｐ⇒Ｑ」が真になるまで必ず失敗する**/

while_do(P,Q):-P,!,Q.
while_do(P,Q).

/**retrieve\3:リストのＮ番目の要素をＸへ抽出する**/
retrieve(1,  [X | Ls], X):-!.
retrieve(N, [Y | Ls], X) :-
    N > 1, N1 is N - 1, retrieve(N1, Ls, X).

/**my_reverse\2:リストの反転**/

my_reverse([], []).
my_reverse([X | Xs], Ys) :-
    my_reverse(Xs, Zs), append(Zs, [X], Ys).
```

```
/**for\2:I から J になるまで繰り返す。重要。**/

for(I =<J,I):- I =<J.
for(I =<J,K):- I =<J,
      I1 is I + 1, for(I1 =<J,K).


/**append0\3:リストの加法。 両方リストでないと機能しないので、
             初期段階で予め asserta 等で空リストを入れておく必要がある**/

append0(Z= [] + Z).
append0([A|Z] = [A|X] + Y):- append0(Z = X + Y).


/**余りを求める**/

res_q(A = B*Q + R):-Q is floor(A/B),R is A - B*Q.


/**2 数の最大公約数を求める**/

gcd(A=A*1+0*0):-!.
gcd(D=A*X+B*Y):-
           res_q(A=B*Q+R),
           (A1,B1)=(B, R),
           gcd(D=A1*X1+B1*Y1),
           T is X1-Y1*Q,(X,Y)=(Y1,T).


/**false:必ず失敗する**/

false:-fail,!.
```

```
/**factor(P/I):I は P の最小因数**/

factor(P/2):-Q is P//2,P =:=2*Q,!.
factor(P/I):-P1 is floor(sqrt(P)),
     for_step(1 =< P1,J,1),
     J1 is 2*J+1,
     Q is P//J1,
     P =:=J1*Q,I = J1,!.
factor(P/P):-!.

/**member\2:組み込み述語 member\2 だと、別解が出てきて困るので捨てる
            ようにした**/

memberd(X,List):-member(X,List),!.



/**junkan\4:1 次元アフィン変換して、それをリスト化する**/

junkan(K,P,N,G):-

                   P1 is G*P mod N,
                   S1 is (P1+1) mod N,

                   asserta(s(S1)),
                   retract1(l(L)),/** Input l/1 in kidou2\3**/
                   append0(LS1=L +[S1]),
                   asserta(l(LS1)),

  ctr_inc(0,T),
 (memberd(S1,L))->(retract1(l(LS1)),asserta(m(L)));
                  (retract1(s(S2)),junkan(K,S2,N,G)).
  /**If this program is succsseed,asserta(m/1)**/
```

```
/** kidou(P,Q,R):・junkan\4 でアフィン変換に用いる初期値を l に入れて
                  計算、結果のリストを m に入れる
                ・この関数を使って計算した回数を 0 番目のカウンター
                  に記録**/
kidou3(P,Q,R,S,G):-
K is P,
asserta(l([K])),/**l/1 is used in junkan\4**/
ctr_set(0,0),
junkan(K,P,Q,R),
ctr_is(0,G),
retract1(m(S)).


/**kidou3_1\5:jikken4\2,5\2 で使用
              上のものを改良して、g,n を固定した状態で
              不動点が何個あるか調べる check_id3 を追加した**/
kidou3_1(P,Q,R,S,G):-
K is P,
asserta(l([K])),/**l/1 is used in junkan\4**/
ctr_set(0,0),
junkan(K,P,Q,R),
ctr_is(0,G),
retract1(m(S)) ,
check_id3(G).

/**check_id3\1:軌道の長さが 1 ならば 4 番目のカウンターを増やす**/
check_id3(G):-(G=:=1)->(ctr_inc(4,G3),true);(true).
```

```
/** check_id\5: ・軌道を集合としてみたとき、互いに素なものだけ
                  残るようにする。
                ・2番目のカウンターはjikkenx\2でg,nを固定した
                  1パターンのループ回数をカウントする為のもの**/
/**for jikken1,2,3**/

check_id(N1,M,K,L,N1):-!.
check_id(N1,M,K,L,N2):-
n(N2,N,K,L),!,
(memberd(M,N))->(N4 is N1 - 1,asserta(cnt(2,N4)),M=:=K-1,
                     ctr_set(3,0),
                     write('G-Orbit:;;'),write(K),
             write(';'),write(L),write(';;'),
             write('G('),write(K),write(') = '),
             check_id2(N1,N4,K,L,1),!,false);

                 (N3 is N2 + 1,!, check_id(N1,M,K,L,N3)).

/**for jikken4,5**/

check_id_2(N1,M,K,L,N1):-!.
check_id_2(N1,M,K,L,N2):-
                     n(N2,N,K,L),!,
                     (memberd(M,N))->(N4 is N1 - 1,
                                      asserta(cnt(2,N4)),
                                      M=:=K-1,

                                      ctr_set(3,0),

                               write('G-Orbit:;;'),
                               write(K),write(';'),
                               write(L),write(';;'),
                               write('G('),write(K),
                               write(') = '),

                                           asserta(l([])),
                               check_id2_2(N1,N4,K,L,1),!,false);

                           (N3 is N2 + 1,!, check_id_2(N1,M,K,L,N3)).
```

```
/**check_id2\5:軌道の長さを３番目のカウンターに記録**/
/**for jikken1,jikken2,jikken3**/

check_id2(N1,N11,K,L,N1):-!,tab(3),ctr_is(3,G2),!,
                          write(G2),write(';'),nl,nl,
                          write('Rules;x;n;g;Orbit;Count'),nl.

check_id2(N1,N11,K,L,N11):-
                           chk(N2,G,K,L),!,write(G),write(';'),
               ctr_inc(3,G2),
               N3 is N2 + 1,!,
               check_id2(N1,N11,K,L,N3).

check_id2(N1,N11,K,L,N2):-
                      chk(N2,G,K,L),!,
                    write(G),tab(1),write('+'),tab(1),
                    N3 is N2 + 1,!,
                    ctr_inc(3,G2),

                      check_id2(N1,N11,K,L,N3).
```

```
/**list_bunkai\2:・check_id2_2で使用
                ・リストが長くて困ったので、類方程式を足し算
                  から掛け算の形へ変換**/

list_bunkai(LG2,1,1):-
                      retrieve(LG21,LG2,LG22),!,
                  write(LG22),write('*'),write('1;').

list_bunkai(LG2,LG21,LG21):-
                            retrieve(LG21,LG2,LG22),!,
                        write(LG22),sctr_set(1,LG22,1),
                        N2 is LG21 - 1,

                             list_bunkai(LG2,N2,LG21).

list_bunkai(LG2,0,LG21):-!.

list_bunkai(LG2,1,LG21):-
                     retrieve(1,LG2,LG22),!,
                       check_id4_2(LG22),

                              list_bunkai(LG2,0,LG21).

list_bunkai(LG2,N,LG21):-
                     retrieve(N,LG2,LG22),!,
             check_id4_1(LG22),
                     N2 is N - 1,

                             list_bunkai(LG2,N2,LG21).
```

```
/**check_id4_x:・list_bunkai\3で使用
                ・リストのどの部分に当たるか判定して、実際に
                  書き出す部分**/

check_id4_1(LG22):-sctr_is(1,LG23,N11),
           asserta(l(LG23)),asserta(s(LG22)),

                   (LG22=:=LG23)->

                    /**true**/
                     (retract1(s(LG24)),retract1(l(_)),
                     sctr_inc(1,LG24,N11));

                    /**false**/
             (retract1(s(LG25)),retract1(l(LG24)),
             write('*'),
                     sctr_banish(1,LG24,N11),!,write(N11),
                     write(' + '),!,write(LG25),
                     sctr_set(1,LG25,1),!).
check_id4_1(LG22).

check_id4_2(LG22):-sctr_is(1,LG23,N11),asserta(l(LG23)),
                   asserta(s(LG22)),

                   (LG22=:=LG23)->

                    /**true**/
            (retract1(s(LG24)),retract1(l(_)),
                    sctr_inc(1,LG24,N11),write('*'),
                    sctr_banish(1,LG24,N12),write(N12),
                    write(';'));

                    /**false**/
                (retract1(s(LG25)),retract1(l(LG24)),
                write('*'),sctr_banish(1,LG24,N11),!,
                write(N11),write(' + '),!,
                    write(LG25),write('*1;'),!).
check_id4_2(LG22).
```

```
/**check_id2_2:jikken4,5で使用
    ・全ての軌道を計算し終えた後に、類方程式とその長さ、
      分類を書き出す。
    ・最初は類方程式をリスト化していなかったので、length\2と
      カウンターを使うやり方が混在。
**/

check_id2_2(N1,N11,K,L,N1):-!,ctr_is(3,G2),!,ctr_is(4,G3),!,
                            ctr_set(4,0),

                            retract1(l(LG)),!,msort(LG,LG2),
                            my_reverse(LG2,LG22),
                            length(LG22,LG21),
                            sort(LG,LG3),length(LG3,LG31),
                        list_bunkai(LG22,LG21,LG21),

                            write(G2),
                write(';'),write(LG31),write(';'),
                            write(G3),write(';'),

                            nl,nl,write('Rules;x;n;g;Orbit;Count;
                            Orbit Count;kind')
                            ,nl.

check_id2_2(N1,N11,K,L,N11):-
                         chk(N2,G,K,L),!,

                         retract1(l(L2)),
                             append0(LG=L2 +[G]),
                             asserta(l(LG)),

                             ctr_inc(3,G2),
                 N3 is N2 + 1,!,

                         check_id2_2(N1,N11,K,L,N3).
```

```
check_id2_2(N1,N11,K,L,N2):-
                chk(N2,G,K,L),!,

                retract1(l(L2)),
                            append0(LG=L2 +[G]),
                            asserta(l(LG)),

                        N3 is N2 + 1,!,
                            ctr_inc(3,G2),

                          check_id2_2(N1,N11,K,L,N3).
```

```
/**メインプログラム**/

/**jikken1:・(nの値,gの値) と指定して、実際に類方程式を求める。
           ・条件はgcd(Q,R)=1,P<Q,1=<P,2=<Q,2=<R**/
jikken1(Q,R):-asserta(cnt(2,0)),ctr_set(5,0),

          tab(3),write('Rules;x;n;g;Orbit;Count'),nl,
          for(2=<Q,K),

          for(2=<R,L),

           retract1(cnt(2,_)),
           asserta(cnt(2,0)),

           gcd(D = K*_ + L*_),D=:=1,

           for(1=<K-1,M),

        retract1(cnt(2,N1)),
        N2 is N1+1,
        check_id(N2,M,K,L,1),
                kidou3(M,K,L,S,G),

        write('[n = '),write(K),write(', g = '),write(L),
                write(' , x = '),write(M),write('];'),
        tab(3),write(M),write(';'),tab(2),
        write(K),write(';'),
        tab(2),write(L),write(';'),tab(2),write(S),
        write(';'),tab(2),
        write(G),write(';'),nl,
        ctr_inc(5,G11),

        asserta(chk(N2,G,K,L)),
        asserta(n(N2,S,K,L)),
           asserta(cnt(2,N2)),

        M=:=K-1,
        N3 is N2 + 1,
        write('G-Orbit:;;'),write(K),write(';'),
```

```
        write(L),write(';;'),
        write('G('),write(K),write(') = '),

                ctr_set(3,0),

    check_id2(N3,N2,K,L,1),
    fail.
jikken1(Q,R):-ctr_is(5,G11),write(G11).
```

```
/**jikken2:・基本はjikken1と同じ。nが素数の場合のみを
            計算する。
          ・結局あまり違いが見られなかったので
            結果には反映せず。 **/

jikken2(Q,R):-asserta(cnt(2,0)),

     tab(3),write('Rules;x;n;g;Orbit;Count'),nl,

      for(2=<Q,K),

       for(2=<R,L),

        retract1(cnt(2,_)),
        asserta(cnt(2,0)),
        gcd(D = K*_ + L*_),D=:=1,
        factor(K/K1),K=:=K1,

         for(1=<K-1,M),

      retract1(cnt(2,N1)),
      N2 is N1+1,
      check_id(N2,M,K,L,1),
              kidou3(M,K,L,S,G),

      write('[n = '),write(K),write(', g = '),write(L),
              write(' , x = '),write(M),write('];'),
      tab(3),write(M),write(';'),tab(2),
      write(K),write(';'),
      tab(2),write(L),write(';'),tab(2),write(S),
      write(';'),tab(2),write(G),write(';'),nl,

      asserta(chk(N2,G,K,L)),
      asserta(n(N2,S,K,L)),
      asserta(cnt(2,N2)),

      M=:=K-1,
      N3 is N2 + 1,
      write('G-Orbit:;;;;;'),
```

```
      write('G('),write(K),write(') = '),
      check_id2(N3,N2,K,L,1),

      fail.
jikken2(Q,R).

/**jikken3:・今度は反対に、n が素数ではない場合。
           ・jikken2 と同じく結果には反映していない。**/

jikken3(Q,R):-asserta(cnt(2,0)),

     tab(3),write('Rules;x;n;g;Orbit;Count'),nl,

      for(2=<R,L),

       for(2=<Q,K),

        retract1(cnt(2,_)),
        asserta(cnt(2,0)),
        gcd(D = K*_ + L*_),D=:=1,
        factor(K/K1),K=:=K1,

        for(1=<K-1,M),

     retract1(cnt(2,N1)),
     N2 is N1+1,
     check_id(N2,M,K,L,1),
             kidou3(M,K,L,S,G),

     write('[n = '),write(K),write(', g = '),write(L),
             write(' , x = '),write(M),write('];'),
     tab(3),write(M),write(';'),tab(2),
     write(K),write(';'),
     tab(2),write(L),write(';'),tab(2),
     write(S),write(';'),tab(2),write(G),write(';'),nl,

     asserta(chk(N2,G,K,L)),
     asserta(n(N2,S,K,L)),
     asserta(cnt(2,N2)),
```

```
    M=:=K-1,
    N3 is N2 + 1,
    write('G-Orbit:;;;;;;'),
    write('G('),write(K),write(') = '),
    check_id2(N3,N2,K,L,1),
        fail.
jikken3(Q,R).
```

```
/**jikken4\2:/・jikken1を改良して、不動点のみを抜き出す
              ようにしたもの。
              gcd(n,1-g)=1⇔(1-g)x=1(但しx=Z/(n))を利用した。
             ・条件を少しいじると、不動点を持つn,gの
              パターンが計算できる。
             ・読もうとすると殆ど他の述語全部と
              つながりがある。**/

jikken4(Q,R):-asserta(cnt(2,0)),ctr_set(4,0),ctr_set(5,0),

      tab(3),write('Rules;x;n;g;1-g;Orbit;Count;Orbits
      Count;kind'),nl,

      for(2=<R,L),

       for(2=<Q,K),

        retract1(cnt(2,_)),
        asserta(cnt(2,0)),

        gcd(D = K*_ + L*_),D=:=1,

        M1 is (1-L),
        M1<K,

        gcd(E = K*_ + M1*_),memberd(E,[1,-1]),

        for(1=<K-1,M),

     M2 is M*M1 mod K,/**solve (1-g)x=1 mod n**/
         M21 is 1-K,

     memberd(M2,[1,M21]), /**enable IT if you would like to
                              cheese ONLY unmoving point**/

          retract1(cnt(2,N1)),
     N2 is N1+1,

     check_id_2(N2,M,K,L,1),
```

```
        kidou3_1(M,K,L,S,G),

    write('[n = '),write(K),write(', g = '),write(L),
        write(' , x = '),write(M),write(',1-g = '),
    write(M1),write('];'),
    tab(3),write(M),write(';'),tab(2),write(K),write(';'),
    tab(2),write(L),write(';'),tab(2),write(M1),write(';'),
    tab(2),write(S),write(';'),tab(2) ,write(G),write(';'),nl,

        ctr_inc(5,G11),

    asserta(chk(N2,G,K,L)),
    asserta(n(N2,S,K,L)),
    asserta(cnt(2,N2)),
    /**M=:=K-1,**/
    fail,/**if you choose ONLY unmoving point,
                    ENABLE IT instead of M=:=K-1**/
    N3 is N2 + 1,
    write('G-Orbit :;;'),write(K),write(';'),
    write(L),write(';;;;;'),
    write('G('),write(K),write(') = '),
    ctr_set(3,0),asserta(l([])),

    check_id2_2(N3,N2,K,L,1),
    fail.

jikken4(Q,R):-write('Total: '),ctr_is(5,G11),write(G11),
              write(' objects').
```

```
/**jikken5\2:反対に、不動点を持たない場合の g,n に対する x の値
             を求める。**/


jikken5(Q,R):-asserta(cnt(2,0)),ctr_set(4,0),ctr_set(5,0),

      tab(3),write('Rules;x;n;g;1-g;Orbit;Count;Orbits
              Count;kind'),nl,

      for(2=<R,L),

       for(2=<Q,K),

        retract1(cnt(2,_)),
        asserta(cnt(2,0)),

        gcd(D = K*_ + L*_),D=:=1,

        M1 is (1-L),

    gcd(E = K*_ + M1*_),not(memberd(E,[1,-1])),

    for(1=<K-1,M),

         retract1(cnt(2,N1)),
     N2 is N1+1,
     check_id_2(N2,M,K,L,1),
             kidou3_1(M,K,L,S,G),

     write('[n = '),write(K),write(', g = '),write(L),
             write(' , x = '),write(M),write(',1-g = '),
             write(M1),write('];'),
     tab(3),write(M),write(';'),tab(2),write(K),write(';'),
     tab(2),write(L),write(';'),tab(2),write(M1),write(';'),
     tab(2),write(S),write(';'),tab(2) ,write(G),write(';'),nl,

     ctr_inc(5,G11),

     asserta(chk(N2,G,K,L)),
```

```
    asserta(n(N2,S,K,L)),
    asserta(cnt(2,N2)),

    M=:=K-1,/** specific means**/

    N3 is N2 + 1,

    write('G-Orbit:;;'),write(K),write(';'),
            write(L),write(';;'),
    write('G('),write(K),write(') = '),

    ctr_set(3,0),

    asserta(l([])),

    check_id2_2(N3,N2,K,L,1),
    fail.

jikken5(Q,R):-ctr_is(5,G11),write('Total: '),write(G11),
              write(' Objects').
```

第4章　結果

4.1　不動点になるパターン(g,n,xと不動点の関係)

条件は$n =< 100, g_1 =< 10$

x	n	g	$1-g$	類方程式
2	3	2	-1	[2]
4	5	2	-1	[4]
6	7	2	-1	[6]
8	9	2	-1	[8]
10	11	2	-1	[10]
12	13	2	-1	[12]
14	15	2	-1	[14]
16	17	2	-1	[16]
18	19	2	-1	[18]
20	21	2	-1	[20]
22	23	2	-1	[22]
24	25	2	-1	[24]
26	27	2	-1	[26]
28	29	2	-1	[28]
30	31	2	-1	[30]
32	33	2	-1	[32]
34	35	2	-1	[34]
36	37	2	-1	[36]
38	39	2	-1	[38]
40	41	2	-1	[40]
42	43	2	-1	[42]
44	45	2	-1	[44]
46	47	2	-1	[46]
48	49	2	-1	[48]
50	51	2	-1	[50]
52	53	2	-1	[52]
54	55	2	-1	[54]
56	57	2	-1	[56]
58	59	2	-1	[58]
60	61	2	-1	[60]
62	63	2	-1	[62]
64	65	2	-1	[64]
66	67	2	-1	[66]
68	69	2	-1	[68]
70	71	2	-1	[70]
72	73	2	-1	[72]
74	75	2	-1	[74]
76	77	2	-1	[76]
78	79	2	-1	[78]
80	81	2	-1	[80]
82	83	2	-1	[82]
84	85	2	-1	[84]
86	87	2	-1	[86]
88	89	2	-1	[88]
90	91	2	-1	[90]
92	93	2	-1	[92]
94	95	2	-1	[94]
96	97	2	-1	[96]
98	99	2	-1	[98]

・g=3 のとき

x	n	g	$1-g$	類方程式
2	5	3	-2	[2]
3	7	3	-2	[3]
5	11	3	-2	[5]
6	13	3	-2	[6]
8	17	3	-2	[8]
9	19	3	-2	[9]
11	23	3	-2	[11]
12	25	3	-2	[12]
14	29	3	-2	[14]
15	31	3	-2	[15]
17	35	3	-2	[17]
18	37	3	-2	[18]
20	41	3	-2	[20]
21	43	3	-2	[21]
23	47	3	-2	[23]
24	49	3	-2	[24]
26	53	3	-2	[26]
27	55	3	-2	[27]
29	59	3	-2	[29]
30	61	3	-2	[30]
32	65	3	-2	[32]
33	67	3	-2	[33]
35	71	3	-2	[35]
36	73	3	-2	[36]
38	77	3	-2	[38]
39	79	3	-2	[39]
41	83	3	-2	[41]
42	85	3	-2	[42]
44	89	3	-2	[44]
45	91	3	-2	[45]
47	95	3	-2	[47]
48	97	3	-2	[48]

・g=4 のとき

x	n	g	$1-g$	類方程式
3	5	4	-3	[3]
2	7	4	-3	[2]
7	11	4	-3	[7]
4	13	4	-3	[4]
11	17	4	-3	[11]
6	19	4	-3	[6]
15	23	4	-3	[15]
8	25	4	-3	[8]
19	29	4	-3	[19]
10	31	4	-3	[10]
23	35	4	-3	[23]
12	37	4	-3	[12]
27	41	4	-3	[27]
14	43	4	-3	[14]
31	47	4	-3	[31]
16	49	4	-3	[16]
35	53	4	-3	[35]
18	55	4	-3	[18]
39	59	4	-3	[39]
20	61	4	-3	[20]
43	65	4	-3	[43]
22	67	4	-3	[22]
47	71	4	-3	[47]
24	73	4	-3	[24]
51	77	4	-3	[51]
26	79	4	-3	[26]
55	83	4	-3	[55]
28	85	4	-3	[28]
59	89	4	-3	[59]
30	91	4	-3	[30]
63	95	4	-3	[63]
32	97	4	-3	[32]

・n=5 のとき

x	n	g	$1-g$	類方程式
2	3	5	-4	[2]
5	7	5	-4	[5]
2	9	5	-4	[2]
8	11	5	-4	[8]
3	13	5	-4	[3]
4	17	5	-4	[4]
14	19	5	-4	[14]
5	21	5	-4	[5]
17	23	5	-4	[17]
20	27	5	-4	[20]
7	29	5	-4	[7]
23	31	5	-4	[23]
8	33	5	-4	[8]
9	37	5	-4	[9]
29	39	5	-4	[29]
10	41	5	-4	[10]
32	43	5	-4	[32]
35	47	5	-4	[35]
12	49	5	-4	[12]
38	51	5	-4	[38]
13	53	5	-4	[13]
14	57	5	-4	[14]
44	59	5	-4	[44]
15	61	5	-4	[15]
47	63	5	-4	[47]
50	67	5	-4	[50]
17	69	5	-4	[17]
53	71	5	-4	[53]
18	73	5	-4	[18]
19	77	5	-4	[19]
59	79	5	-4	[59]
20	81	5	-4	[20]
62	83	5	-4	[62]
65	87	5	-4	[65]
22	89	5	-4	[22]

x	n	g	$1-g$	類方程式
68	91	5	-4	[68]
23	93	5	-4	[23]
24	97	5	-4	[24]
74	99	5	-4	[74]

・g=6 のとき

x	n	g	$1-g$	類方程式
4	7	6	-5	[4]
2	11	6	-5	[2]
5	13	6	-5	[5]
10	17	6	-5	[10]
15	19	6	-5	[15]
9	23	6	-5	[9]
23	29	6	-5	[23]
6	31	6	-5	[6]
22	37	6	-5	[22]
8	41	6	-5	[8]
17	43	6	-5	[17]
28	47	6	-5	[28]
39	49	6	-5	[39]
21	53	6	-5	[21]
47	59	6	-5	[47]
12	61	6	-5	[12]
40	67	6	-5	[40]
14	71	6	-5	[14]
29	73	6	-5	[29]
46	77	6	-5	[46]
63	79	6	-5	[63]
33	83	6	-5	[33]
71	89	6	-5	[71]
18	91	6	-5	[18]
58	97	6	-5	[58]

・g=7 のとき

x	n	g	$1-g$	類方程式
4	5	7	-6	[4]
9	11	7	-6	[9]
2	13	7	-6	[2]
14	17	7	-6	[14]
3	19	7	-6	[3]
19	23	7	-6	[19]
4	25	7	-6	[4]
24	29	7	-6	[24]
5	31	7	-6	[5]
6	37	7	-6	[6]
34	41	7	-6	[34]
7	43	7	-6	[7]
39	47	7	-6	[39]
44	53	7	-6	[44]
9	55	7	-6	[9]
49	59	7	-6	[49]
10	61	7	-6	[10]
54	65	7	-6	[54]
11	67	7	-6	[11]
59	71	7	-6	[59]
12	73	7	-6	[12]
13	79	7	-6	[13]
69	83	7	-6	[69]
14	85	7	-6	[14]
74	89	7	-6	[74]
79	95	7	-6	[79]
16	97	7	-6	[16]

・n=8 のとき

x	n	g	$1-g$	類方程式
2	3	8	-7	[2]
2	5	8	-7	[2]
5	9	8	-7	[5]
3	11	8	-7	[3]
11	13	8	-7	[11]
2	15	8	-7	[2]
12	17	8	-7	[12]
8	19	8	-7	[8]
13	23	8	-7	[13]
7	25	8	-7	[7]
23	27	8	-7	[23]
4	29	8	-7	[4]
22	31	8	-7	[22]
14	33	8	-7	[14]
21	37	8	-7	[21]
11	39	8	-7	[11]
35	41	8	-7	[35]
6	43	8	-7	[6]
32	45	8	-7	[32]
20	47	8	-7	[20]
29	51	8	-7	[29]
15	53	8	-7	[15]
47	55	8	-7	[47]
8	57	8	-7	[8]
42	59	8	-7	[42]
26	61	8	-7	[26]
37	65	8	-7	[37]
19	67	8	-7	[19]
59	69	8	-7	[59]
10	71	8	-7	[10]
52	73	8	-7	[52]
32	75	8	-7	[32]
45	79	8	-7	[45]
23	81	8	-7	[23]
71	83	8	-7	[71]

x	n	g	$1-g$	類方程式
12	85	8	-7	[12]
62	87	8	-7	[62]
38	89	8	-7	[38]
53	93	8	-7	[53]
27	95	8	-7	[27]
83	97	8	-7	[83]
14	99	8	-7	[14]

・g=9 のとき

x	n	g	$1-g$	類方程式
3	5	9	-8	[3]
6	7	9	-8	[6]
4	11	9	-8	[4]
8	13	9	-8	[8]
2	17	9	-8	[2]
7	19	9	-8	[7]
20	23	9	-8	[20]
3	25	9	-8	[3]
18	29	9	-8	[18]
27	31	9	-8	[27]
13	35	9	-8	[13]
23	37	9	-8	[23]
5	41	9	-8	[5]
16	43	9	-8	[16]
41	47	9	-8	[41]
6	49	9	-8	[6]
33	53	9	-8	[33]
48	55	9	-8	[48]
22	59	9	-8	[22]
38	61	9	-8	[38]
8	65	9	-8	[8]
25	67	9	-8	[25]
62	71	9	-8	[62]
9	73	9	-8	[9]
48	77	9	-8	[48]
69	79	9	-8	[69]

x	n	g	$1-g$	類方程式
31	83	9	-8	[31]
53	85	9	-8	[53]
11	89	9	-8	[11]
34	91	9	-8	[34]
83	95	9	-8	[83]
12	97	9	-8	[12]

・g=10 のとき

x	n	g	$1-g$	類方程式
3	7	10	-9	[3]
6	11	10	-9	[6]
10	13	10	-9	[10]
15	17	10	-9	[15]
2	19	10	-9	[2]
5	23	10	-9	[5]
16	29	10	-9	[16]
24	31	10	-9	[24]
4	37	10	-9	[4]
9	41	10	-9	[9]
19	43	10	-9	[19]
26	47	10	-9	[26]
38	49	10	-9	[38]
47	53	10	-9	[47]
13	59	10	-9	[13]
27	61	10	-9	[27]
52	67	10	-9	[52]
63	71	10	-9	[63]
8	73	10	-9	[8]
17	77	10	-9	[17]
35	79	10	-9	[35]
46	83	10	-9	[46]
79	89	10	-9	[79]
10	91	10	-9	[10]
43	97	10	-9	[43]

以上 303 パターン。

4.2 不動点を持たないパターンの軌道

4.2.1 のみ $n =< 300, g_1 =< 20$ について調べた。あとは不動点のときと同じ。

4.2.1 軌道が一つのみのもの (以降各軌道の個数を*を使って表記)

n	g	$1-g$	類方程式
2	3	-2	G(2) = 2*1
3	4	-3	G(3) = 3*1
9	4	-3	G(9) = 9*1
27	4	-3	G(27) = 27*1
81	4	-3	G(81) = 81*1
243	4	-3	G(243) = 243*1
2	5	-4	G(2) = 2*1
4	5	-4	G(4) = 4*1
8	5	-4	G(8) = 8*1
16	5	-4	G(16) = 16*1
32	5	-4	G(32) = 32*1
64	5	-4	G(64) = 64*1
128	5	-4	G(128) = 128*1
256	5	-4	G(256) = 256*1
5	6	-5	G(5) = 5*1
25	6	-5	G(25) = 25*1
125	6	-5	G(125) = 125*1
2	7	-6	G(2) = 2*1
3	7	-6	G(3) = 3*1
6	7	-6	G(6) = 6*1
9	7	-6	G(9) = 9*1
18	7	-6	G(18) = 18*1
27	7	-6	G(27) = 27*1
54	7	-6	G(54) = 54*1
81	7	-6	G(81) = 81*1

n	g	$1-g$	類方程式
162	7	-6	G(162) = 162*1
243	7	-6	G(243) = 243*1
7	8	-7	G(7) = 7*1
49	8	-7	G(49) = 49*1
2	9	-8	G(2) = 2*1
4	9	-8	G(4) = 4*1
8	9	-8	G(8) = 8*1
16	9	-8	G(16) = 16*1
32	9	-8	G(32) = 32*1
64	9	-8	G(64) = 64*1
128	9	-8	G(128) = 128*1
256	9	-8	G(256) = 256*1
3	10	-9	G(3) = 3*1
9	10	-9	G(9) = 9*1
27	10	-9	G(27) = 27*1
81	10	-9	G(81) = 81*1
243	10	-9	G(243) = 243*1
2	11	-10	G(2) = 2*1
5	11	-10	G(5) = 5*1
10	11	-10	G(10) = 10*1
25	11	-10	G(25) = 25*1
50	11	-10	G(50) = 50*1
125	11	-10	G(125) = 125*1
250	11	-10	G(250) = 250*1
11	12	-11	G(11) = 11*1
121	12	-11	G(121) = 121*1
2	13	-12	G(2) = 2*1
3	13	-12	G(3) = 3*1
4	13	-12	G(4) = 4*1
6	13	-12	G(6) = 6*1
8	13	-12	G(8) = 8*1
9	13	-12	G(9) = 9*1
12	13	-12	G(12) = 12*1
16	13	-12	G(16) = 16*1
18	13	-12	G(18) = 18*1

n	g	$1-g$	類方程式
24	13	-12	G(24) = 24*1
27	13	-12	G(27) = 27*1
32	13	-12	G(32) = 32*1
36	13	-12	G(36) = 36*1
48	13	-12	G(48) = 48*1
54	13	-12	G(54) = 54*1
64	13	-12	G(64) = 64*1
72	13	-12	G(72) = 72*1
81	13	-12	G(81) = 81*1
96	13	-12	G(96) = 96*1
108	13	-12	G(108) = 108*1
128	13	-12	G(128) = 128*1
144	13	-12	G(144) = 144*1
162	13	-12	G(162) = 162*1
192	13	-12	G(192) = 192*1
216	13	-12	G(216) = 216*1
243	13	-12	G(243) = 243*1
256	13	-12	G(256) = 256*1
288	13	-12	G(288) = 288*1
13	14	-13	G(13) = 13*1
169	14	-13	G(169) = 169*1
2	15	-14	G(2) = 2*1
7	15	-14	G(7) = 7*1
14	15	-14	G(14) = 14*1
49	15	-14	G(49) = 49*1
98	15	-14	G(98) = 98*1
3	16	-15	G(3) = 3*1
5	16	-15	G(5) = 5*1
9	16	-15	G(9) = 9*1
15	16	-15	G(15) = 15*1
25	16	-15	G(25) = 25*1
27	16	-15	G(27) = 27*1
45	16	-15	G(45) = 45*1
75	16	-15	G(75) = 75*1
81	16	-15	G(81) = 81*1

n	g	$1-g$	類方程式
125	16	-15	G(125) = 125*1
135	16	-15	G(135) = 135*1
225	16	-15	G(225) = 225*1
243	16	-15	G(243) = 243*1
2	17	-16	G(2) = 2*1
4	17	-16	G(4) = 4*1
8	17	-16	G(8) = 8*1
16	17	-16	G(16) = 16*1
32	17	-16	G(32) = 32*1
64	17	-16	G(64) = 64*1
128	17	-16	G(128) = 128*1
256	17	-16	G(256) = 256*1
17	18	-17	G(17) = 17*1
289	18	-17	G(289) = 289*1
2	19	-18	G(2) = 2*1
3	19	-18	G(3) = 3*1
6	19	-18	G(6) = 6*1
9	19	-18	G(9) = 9*1
18	19	-18	G(18) = 18*1
27	19	-18	G(27) = 27*1
54	19	-18	G(54) = 54*1
81	19	-18	G(81) = 81*1
162	19	-18	G(162) = 162*1
243	19	-18	G(243) = 243*1
19	20	-19	G(19) = 19*1

2

4.2.2 軌道の種類が１種類しかないもの (前項のものも含む)

n	g	$1-g$	類方程式	長さ
2	3	-2	G(2) = 2*1	1
4	3	-2	G(4) = 2*2	2
8	3	-2	G(8) = 4*2	2
16	3	-2	G(16) = 8*2	2
32	3	-2	G(32) = 16*2	2
40	3	-2	G(40) = 4*10	10
64	3	-2	G(64) = 32*2	2
80	3	-2	G(80) = 8*10	10
3	4	-3	G(3) = 3*1	1
9	4	-3	G(9) = 9*1	1
21	4	-3	G(21) = 3*7	7
27	4	-3	G(27) = 27*1	1
63	4	-3	G(63) = 9*7	7
81	4	-3	G(81) = 81*1	1
2	5	-4	G(2) = 2*1	1
4	5	-4	G(4) = 4*1	1
6	5	-4	G(6) = 2*3	3
8	5	-4	G(8) = 8*1	1
12	5	-4	G(12) = 4*3	3
16	5	-4	G(16) = 16*1	1
24	5	-4	G(24) = 8*3	3
32	5	-4	G(32) = 32*1	1
48	5	-4	G(48) = 16*3	3
52	5	-4	G(52) = 4*13	13
64	5	-4	G(64) = 64*1	1
96	5	-4	G(96) = 32*3	3
5	6	-5	G(5) = 5*1	1
25	6	-5	G(25) = 25*1	1
2	7	-6	G(2) = 2*1	1
3	7	-6	G(3) = 3*1	1
4	7	-6	G(4) = 2*2	2

n	g	$1-g$	類方程式	長さ
6	7	-6	G(6) = 6*1	1
8	7	-6	G(8) = 2*4	4
9	7	-6	G(9) = 9*1	1
12	7	-6	G(12) = 6*2	2
16	7	-6	G(16) = 4*4	4
18	7	-6	G(18) = 18*1	1
24	7	-6	G(24) = 6*4	4
27	7	-6	G(27) = 27*1	1
32	7	-6	G(32) = 8*4	4
36	7	-6	G(36) = 18*2	2
48	7	-6	G(48) = 12*4	4
54	7	-6	G(54) = 54*1	1
57	7	-6	G(57) = 3*19	19
64	7	-6	G(64) = 16*4	4
72	7	-6	G(72) = 18*4	4
80	7	-6	G(80) = 4*20	20
81	7	-6	G(81) = 81*1	1
96	7	-6	G(96) = 24*4	4
7	8	-7	G(7) = 7*1	1
49	8	-7	G(49) = 49*1	1
2	9	-8	G(2) = 2*1	1
4	9	-8	G(4) = 4*1	1
8	9	-8	G(8) = 8*1	1
10	9	-8	G(10) = 2*5	5
16	9	-8	G(16) = 16*1	1
20	9	-8	G(20) = 4*5	5
32	9	-8	G(32) = 32*1	1
40	9	-8	G(40) = 8*5	5
64	9	-8	G(64) = 64*1	1
80	9	-8	G(80) = 16*5	5
3	10	-9	G(3) = 3*1	1
9	10	-9	G(9) = 9*1	1
27	10	-9	G(27) = 27*1	1
81	10	-9	G(81) = 81*1	1

4.2.3 軌道の種類が２種類のもの

n	g	$1-g$	類方程式	長さ
10	3	-2	G(10) = 2*1 + 4*2	3
14	3	-2	G(14) = 2*1 + 6*2	3
20	3	-2	G(20) = 2*2 + 4*4	6
22	3	-2	G(22) = 2*1 + 10*2	3
26	3	-2	G(26) = 2*1 + 6*4	5
28	3	-2	G(28) = 2*2 + 6*4	6
34	3	-2	G(34) = 2*1 + 16*2	3
38	3	-2	G(38) = 2*1 + 18*2	3
44	3	-2	G(44) = 2*2 + 10*4	6
46	3	-2	G(46) = 2*1 + 22*2	3
52	3	-2	G(52) = 2*2 + 6*8	10
56	3	-2	G(56) = 4*2 + 12*4	6
58	3	-2	G(58) = 2*1 + 28*2	3
62	3	-2	G(62) = 2*1 + 30*2	3
68	3	-2	G(68) = 2*2 + 16*4	6
74	3	-2	G(74) = 2*1 + 18*4	5
76	3	-2	G(76) = 2*2 + 18*4	6
82	3	-2	G(82) = 2*1 + 8*10	11
86	3	-2	G(86) = 2*1 + 42*2	3
88	3	-2	G(88) = 4*2 + 20*4	6
92	3	-2	G(92) = 2*2 + 22*4	6
94	3	-2	G(94) = 2*1 + 46*2	3
15	4	-3	G(15) = 3*1 + 6*2	3
33	4	-3	G(33) = 3*1 + 15*2	3
39	4	-3	G(39) = 3*1 + 6*6	7
45	4	-3	G(45) = 9*1 + 18*2	3
51	4	-3	G(51) = 3*1 + 12*4	5
57	4	-3	G(57) = 3*1 + 9*6	7
69	4	-3	G(69) = 3*1 + 33*2	3
87	4	-3	G(87) = 3*1 + 42*2	3
93	4	-3	G(93) = 3*1 + 15*6	7
99	4	-3	G(99) = 9*1 + 45*2	3
14	5	-4	G(14) = 2*1 + 6*2	3
18	5	-4	G(18) = 2*3 + 6*2	5

n	g	$1-g$	類方程式	長さ
22	5	-4	G(22) = 2*1 + 10*2	3
26	5	-4	G(26) = 2*1 + 4*6	7
28	5	-4	G(28) = 4*1 + 12*2	3
34	5	-4	G(34) = 2*1 + 16*2	3
36	5	-4	G(36) = 4*3 + 12*2	5
38	5	-4	G(38) = 2*1 + 18*2	3
42	5	-4	G(42) = 2*3 + 6*6	9
44	5	-4	G(44) = 4*1 + 20*2	3
46	5	-4	G(46) = 2*1 + 22*2	3
56	5	-4	G(56) = 8*1 + 24*2	3
58	5	-4	G(58) = 2*1 + 14*4	5
62	5	-4	G(62) = 2*1 + 6*10	11
66	5	-4	G(66) = 2*3 + 10*6	9
68	5	-4	G(68) = 4*1 + 16*4	5
72	5	-4	G(72) = 8*3 + 24*2	5
74	5	-4	G(74) = 2*1 + 36*2	3
76	5	-4	G(76) = 4*1 + 36*2	3
78	5	-4	G(78) = 2*3 + 4*18	21
82	5	-4	G(82) = 2*1 + 20*4	5
84	5	-4	G(84) = 4*3 + 12*6	9
86	5	-4	G(86) = 2*1 + 42*2	3
88	5	-4	G(88) = 8*1 + 40*2	3
92	5	-4	G(92) = 4*1 + 44*2	3
94	5	-4	G(94) = 2*1 + 46*2	3
35	6	-5	G(35) = 5*1 + 10*3	4
55	6	-5	G(55) = 5*1 + 10*5	6
65	6	-5	G(65) = 5*1 + 60*1	2
85	6	-5	G(85) = 5*1 + 80*1	2
95	6	-5	G(95) = 5*1 + 45*2	3
10	7	-6	G(10) = 2*1 + 4*2	3
15	7	-6	G(15) = 3*1 + 12*1	2
20	7	-6	G(20) = 2*2 + 4*4	6
22	7	-6	G(22) = 2*1 + 10*2	3
26	7	-6	G(26) = 2*1 + 12*2	3
30	7	-6	G(30) = 6*1 + 12*2	3

n	g	$1-g$	類方程式	長さ
33	7	-6	G(33) = 3*1 + 30*1	2
34	7	-6	G(34) = 2*1 + 16*2	3
38	7	-6	G(38) = 2*1 + 6*6	7
39	7	-6	G(39) = 3*1 + 12*3	4
40	7	-6	G(40) = 2*4 + 4*8	12
44	7	-6	G(44) = 2*2 + 10*4	6
45	7	-6	G(45) = 9*1 + 36*1	2
46	7	-6	G(46) = 2*1 + 22*2	3
50	7	-6	G(50) = 2*1 + 4*12	13
51	7	-6	G(51) = 3*1 + 48*1	2
52	7	-6	G(52) = 2*2 + 12*4	6
58	7	-6	G(58) = 2*1 + 14*4	5
60	7	-6	G(60) = 6*2 + 12*4	6
62	7	-6	G(62) = 2*1 + 30*2	3
66	7	-6	G(66) = 6*1 + 30*2	3
68	7	-6	G(68) = 2*2 + 16*4	6
69	7	-6	G(69) = 3*1 + 66*1	2
74	7	-6	G(74) = 2*1 + 18*4	5
75	7	-6	G(75) = 3*1 + 12*6	7
76	7	-6	G(76) = 2*2 + 6*12	14
78	7	-6	G(78) = 6*1 + 12*6	7
82	7	-6	G(82) = 2*1 + 40*2	3
86	7	-6	G(86) = 2*1 + 6*14	15
87	7	-6	G(87) = 3*1 + 21*4	5
88	7	-6	G(88) = 2*4 + 10*8	12
90	7	-6	G(90) = 18*1 + 36*2	3
92	7	-6	G(92) = 2*2 + 22*4	6
93	7	-6	G(93) = 3*1 + 15*6	7
94	7	-6	G(94) = 2*1 + 46*2	3
99	7	-6	G(99) = 9*1 + 90*1	2
100	7	-6	G(100) = 2*2 + 4*24	26
21	8	-7	G(21) = 7*1 + 14*1	2
35	8	-7	G(35) = 7*1 + 28*1	2
63	8	-7	G(63) = 7*1 + 14*4	5
77	8	-7	G(77) = 7*1 + 70*1	2

n	g	$1-g$	類方程式	長さ
91	8	-7	G(91) = 7*1 + 28*3	4
14	9	-8	G(14) = 2*1 + 6*2	3
22	9	-8	G(22) = 2*1 + 10*2	3
26	9	-8	G(26) = 2*1 + 6*4	5
28	9	-8	G(28) = 4*1 + 12*2	3
34	9	-8	G(34) = 2*1 + 8*4	5
38	9	-8	G(38) = 2*1 + 18*2	3
44	9	-8	G(44) = 4*1 + 20*2	3
46	9	-8	G(46) = 2*1 + 22*2	3
50	9	-8	G(50) = 2*5 + 10*4	9
52	9	-8	G(52) = 4*1 + 12*4	5
56	9	-8	G(56) = 8*1 + 24*2	3
58	9	-8	G(58) = 2*1 + 14*4	5
62	9	-8	G(62) = 2*1 + 30*2	3
68	9	-8	G(68) = 4*1 + 8*8	9
70	9	-8	G(70) = 2*5 + 6*10	15
74	9	-8	G(74) = 2*1 + 18*4	5
76	9	-8	G(76) = 4*1 + 36*2	3
82	9	-8	G(82) = 2*1 + 4*20	21
86	9	-8	G(86) = 2*1 + 42*2	3
88	9	-8	G(88) = 8*1 + 40*2	3
92	9	-8	G(92) = 4*1 + 44*2	3
94	9	-8	G(94) = 2*1 + 46*2	3
100	9	-8	G(100) = 4*5 + 20*4	9

n	g	$1-g$	類方程式	長さ
21	10	-9	G(21) = 3*1 + 6*3	4
33	10	-9	G(33) = 3*1 + 6*5	6
39	10	-9	G(39) = 3*1 + 6*6	7
51	10	-9	G(51) = 3*1 + 48*1	2
57	10	-9	G(57) = 3*1 + 18*3	4
63	10	-9	G(63) = 9*1 + 18*3	4
69	10	-9	G(69) = 3*1 + 66*1	2
87	10	-9	G(87) = 3*1 + 84*1	2
93	10	-9	G(93) = 3*1 + 15*6	7
99	10	-9	G(99) = 9*1 + 18*5	6

4.2.4 軌道の種類が３種類のもの

n	g	$1-g$	類方程式	長さ
50	3	-2	G(50) = 2*1 + 4*2 + 20*2	5
98	3	-2	G(98) = 2*1 + 6*2 + 42*2	5
100	3	-2	G(100) = 2*2 + 4*4 + 20*4	10
75	4	-3	G(75) = 3*1 + 6*2 + 30*2	5
54	5	-4	G(54) = 2*3 + 6*2 + 18*2	7
98	5	-4	G(98) = 2*1 + 6*2 + 42*2	5
98	9	-8	G(98) = 2*1 + 6*2 + 42*2	5

4.2.5 軌道の種類が４種類のもの

n	g	1-g	類方程式	長さ
70	3	-2	G(70) = 2*1 + 4*2 + 6*2 + 12*4	9

以上、967 パターン。

第5章　考察

5.1　命題

$n =< 100, g =< 10(4.2.1$ は $n =< 300.g =< 20)$ としたときの結果から、次のような命題が立てられた。

1. 1 は不動点にならない。
2. $1 - g_1 \in U_n$ ならば、不動点 x_s が**ただ一つ**存在する。(不動点の一意性)
3. $p_t > 2$(t は番号) を任意の素数、l_t を任意の正数とし、p_t が $\mid 1 - g_1 \mid$ の約数であるとき、
$n = \prod p_t^{l_t} \Rightarrow n = n$(軌道が 1 つになる)
(ただし、$l_t >= 2$ のときは $p_t = 2$ のときも成り立つ)
4. また、命題 3 で 2 を約数に持ち、2 のべき乗を約数に持たないとき、
$n = p_t^{l_t}$ または $2p_t^{l_t} \Rightarrow n = n$

5.2　証明 (命題 1 のみ)

命題 1 (1 は不動点にならない)

・(背理法による証明)
1 が不動点であると仮定する。
原理から、式 (2.8) に $x_s = 1$ を代入すると、

$$(1 - g_1) \equiv 1 \pmod n \tag{5.1}$$

だから、

$$g_1 \equiv 0 \pmod n \tag{5.2}$$

となるが、不動点の定義から、$\gcd(n, 1-g) = 1$ なので、矛盾。故に題意が示せた。

第6章　所感

今後の課題

今回は、1次元アフィン変換について扱ったが、この議論をさらに発展させて、**n 次元に拡張するとどうなるか**、つまり n × m 行列にするとどうなるかという部分に踏み込んでいくと、新しい結果が得られると考えられたので、これを今後の課題として次の代に引き継いでいきたい。

感想

4年間数学をやってきてわかったことは、「意外にまだわからないことが多い」ということだった。例えば、フィボナッチ数列やメルセンヌ素数が無限に存在するかどうか、ということはまだ証明されていない。

しかしながら、数学には、そういう「わからないもの」について色々な手段を用いて固定観念を廃して解析・研究してきた先人達のノウハウが詰まっているし、そういう考え方を知ることが出来たのは、大きな経験となったと思う。

ゼミでは、Prolog も難しかったけれども、飯高先生に丁寧に教えて頂いたおかげで、最初のころとは比べ物にならないくらい理解を深めることが出来た。

最後に、飯高先生、飯高ゼミの皆、1年間一緒に勉強をさせて頂けて楽しかったです。

本当にありがとうございました。

以上を以って感想とさせて頂きます。