文教大学教育研究所公開講座　1998.3.27

コンピュータを使って数学の実験をしよう

－カオスとフラクタル－

白石 和夫

1．BASIC

（1）BASIC の起動

BASIC のシステムが入ったフロッピーディスクを本体に挿入する。

ﾏｲ ｺﾝﾋﾟｭｰﾀ のアイコンをダブルクリック。（マウスの左ボタンを休まないで続けて 2 回押す）

3.5 ｲﾝﾁ FD(A:) をダブルクリック。

BASIC のアイコンをダブルクリック。

（2）簡単なプログラム

① 足し算

```
PRINT 2+3
END
```

END の上の行に

[P] [R] [I] [N] [T] [ ] [2] [+] [3]

と打ち込んで [F9] キーを押してみよう。

☆ PRINT は小文字で print と入力してもよい。

※ [＋] は [Shift] キーを押しながら [；れ] のキーを押す。

※ 入力を間違えたら，[Back space] キーを押して消去する。

※ [ ] は，最下段中央の何も刻印されていないキーを押す。（スペースキー）

② 引き算

```
PRINT 2-3
END
```

2+3 を 2-3 と書き換えて [F9] キーを押す。

＜操作＞矢印キーを操作して，＋の右端に点滅する｜を移動する。

[Back space] キーを押して＋を消す。

[－] は，0 の右隣り，[－ほ] のキーを押す。

※ [Delete] キーを押すと，右隣の文字が消去される。

③ 乗除算，べき乗

```
PRINT 2*3, 2/3, 2^3
END
```

☆ 乗算は * ，除算は / で表す。

2^3 は，2^3 を表す。

＜操作＞ [＊] は，[Shift] を押しながら，[＋] の右となりの [：け] キーを押す。

[／] は，[＋] [＊] の下にある [／め] のキーを押す。

[^] は，[－] の右隣り。[＾へ] のキーを押す。

④ 平方根

```
PRINT SQR(2),SQR(3)
END
```

（２）変数

計算結果をコンピュータ内部の記憶領域に記憶させておき，後で利用できる。

コンピュータ内部の記憶領域を**変数**という。

```
LET A=2^3
PRINT A+1,A*2,A/3
END
```

☆ LET 文は，変数に数値を代入する。

LET 変数名＝数値式

の形式で書く。

☆ 変数名は，大文字と小文字の違いを無視される。

すわなち，変数 A と変数 a は同じ変数を表す。

※ ＝は，shift キーをおしながら－キーを押す。

※ 改行は，Enter と書かれた大きなキーを押す。

（３）FOR～NEXT の繰り返し

①　（prog01.BAS）

```
10 FOR x=1 TO 10
20     PRINT x,SQR(x)
30 NEXT x
40 END
```

X の値を 0 から 10 まで順に変化させて 20 行の PRINT 文を繰り返し実行する。

行の先頭の番号を行番号という。行番号は省略してよい（入力しなくてもよい）。

プログラムの読込み

マウスで  をクリック（左ボタンを押す）。

公開講座 をダブルクリック。

prog01.bas をダブルクリック。

プログラムの実行

マウスで  をクリック。

②　(prog02.BAS)

```
10 FOR x=0 TO 1 STEP 0.1
20     PRINT x,SQR(x)
30 NEXT x
40 END
```

（４）関数の宣言 (prog03.BAS)

```
10  DEF f(x)=x^2 + 3*x +2
20 FOR x=0 TO 1 STEP 0.1
30     PRINT x,f(x)
40 NEXT x
50 END
```

（５）関数のグラフ

二次関数 $y = 2x(2-x)$ のグラフを描いてみよう。

まず x 座標，y 座標の範囲を決める。

たとえば，$-1 \leqq x \leqq 3$，$-1 \leqq y \leqq 3$ とする。

→　SET WINDOW -1,3,-1,3

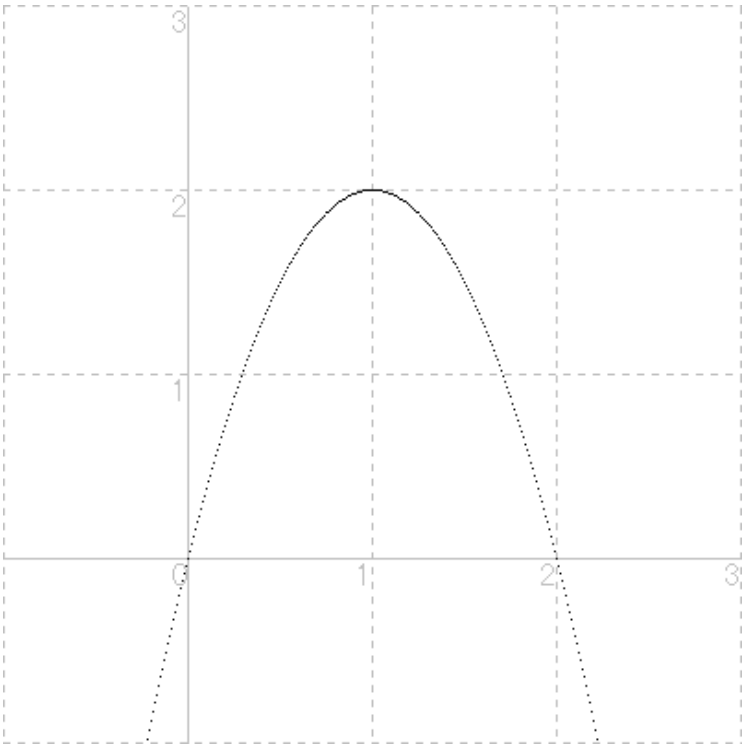
画面上の点(x, y)に点を描く

→　PLOT x,y

座標軸(格子)を描く

→　DRAW grid

以上をまとめて次のプログラムになる。



(prog1.BAS)

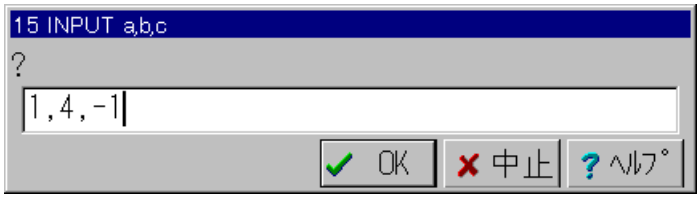
```
10 DEF f(x)=2*x*(2-x)
20 SET WINDOW -1,3,-1,3
30 DRAW grid
40 FOR x=-1 TO 3 STEP 0.01
50     PLOT x,f(x)
60 NEXT x
70 END
```

（６）二次関数 $y=ax^2+bx+c$ のグラフ

プログラムの実行時に係数 a，b，c を入力する。

(prog2.BAS)

```
10 DEF f(x)=a*x^2+b*x+c
15 INPUT a,b,c
20 SET WINDOW -1,3,-1,3
30 DRAW grid
40 FOR x=-1 TO 3 STEP 0.01
50     PLOT x,f(x)
60 NEXT x
70 END
```



実行時には，図のように，３数をコンマで区切って入力する。（最後に Enter キーを押す）

（７）ヘルプの使い方

プログラム中の機能語で意味のわからないものがあったら，

点滅する｜をその位置に移動して，F1 キーを押すと関連するヘルプが表示されます。

ヘルプを読み終わったら，右上の×をクリックして閉じてください。

２．カオス

（１）ウサギの島

ある島におけるウサギの頭数の年毎の変化を考える。ある年の頭数を x_0 として，n 年後の頭数を x_n で表す。ウサギの頭数が少ないときは，正常に繁殖するから，ある年のウサギの頭数はその前年の頭数に比例する。ウサギの頭数が多いと，食料事情等の理由から死ぬウサギが多くなる。これを単純化して $x_{n+1}=k\,x_n(N-x_n)$ と表せるものとしよう。ただし，k，N は定数。

（２）数列の挙動（カオス）

k，x_0 を定数,

$f(x) = k\,x\,(1-x)$

$x_n = f(x_{n-1})$ $(n = 1,2,3,\cdots)$

とする。

たとえば，$k = 3$，$x_0=0.5$ のとき,

$x_1 = 3\times0.5\times0.5 = 0.75$

$x_2 = 3\times0.75\times0.25 = 0.5625$

$x_3 = 3\times0.5625\times0.4375 = 0.73828125$

$x_0=0.5$ として，k の値を変えて数列 $x_0, x_1, x_2, x_3,\cdots$ の挙動を調べてみる。

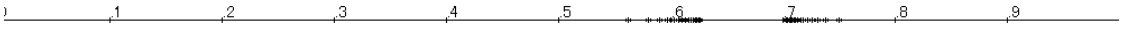
（prog2.bas）

```
10 INPUT k
20 LET x=0.5
30 FOR i=1 TO 30
40     LET x=k*x*(1-x)
50     PRINT x
60 NEXT i
70 END
```

k=3 のとき

```
 .75
 .5625
 .73828125
 .579666137695313
 .730959919514134
 .589972546734074
 .725714822502555
 .59715845670792
 .721680702870406
 .602572997924648
 .71843634029025
 .606856695721806
 .715744939738252
 .610362362932014
 .713460446544187
 .613303913283469
 .711486669703956
 .615820165612589
 .709757067712417
 .618005917634066
 .708223810210026
 .619928534584857
 .706851439776986
 .621637445586564
 .705612995493527
 .623169888252537
 .704487535883573
 .624554543004793
 .703458497450602
 .62581391944543
```

これらの点を数直線上に図示してみる。

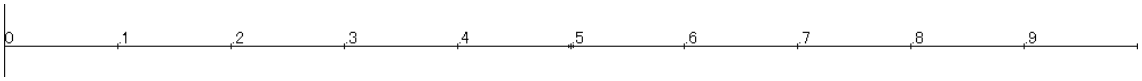


さらに n を大きくしていくと，x_n は，0.682581086920433 付近の値と 0.64999244009686 付近の値とを交互に取るようになる。

$k = 2$ のとき，

$x_2 = 2\times0.5\times0.5 = 0.5\ (=x_1)$

だから，$x_1 = x_2 = x_3 = \cdots = 0.5$

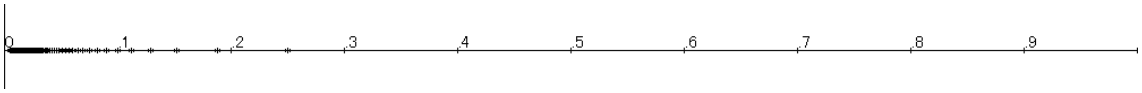


k=1 のとき，

$x_2 = 0.5\times0.5 = 0.25$

$x_3 = 0.25\times0.75 = 0.1875$

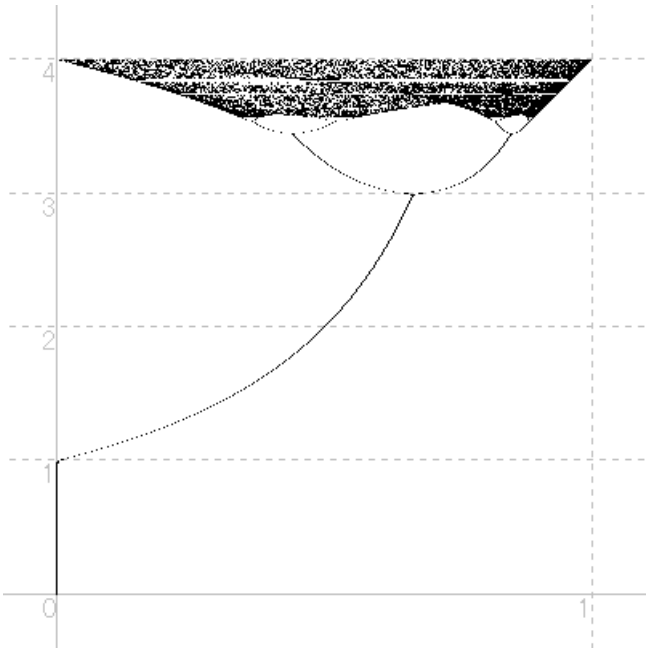
$x_4 = 0.1875\times0.8175 = 0.15234375$



n が大きい値を取るときの x_n の挙動を調べるために，数列のはじめのほうを無視し，x_{201}，x_{202}，x_{203}，…，x_{500} を図示する。

（prog3.bas）

```
100 DEF f(x)=k*x*(1-x)
110 SET WINDOW -0.1, 1.1, -0.4, 4.4
120 DRAW grid
130 FOR k=0 TO 4 STEP 4.8/400
140     LET x=0.5
150     FOR i=1 TO 200
160         LET x=f(x)
170     NEXT i
180     FOR i=201 TO 500
190         LET x=f(x)
200         PLOT x,k
210     NEXT i
220 NEXT k
230 END
```



(注) 130 行の数値 400 は描画領域のドット数によって次のように変更する。

321×321	320
401×401	400
501×501	500

このサイズは，メニューから オプションーグラフィックスーサイズ で確認できる。

ここで，$3 \leqq k \leqq 4$ の範囲を詳細に調べてみる（prog4.bas）。

k の値が 4 に近いときには，n を十分に大きくとっても，$\{ x_n\}$は，一見，不規則な挙動を示す。この状態をカオスという。

3. 複素数

（１）複素数 (complex numbers)

有理数と無理数をあわせて実数という。

実数の体系に，$i^2=-1$ となる新たな '数 'i を付け加え，加減乗除の演算について通常用いる計算法則（加法の交換法則と結合法則，乗法の交換法則と結合法則，分配法則）が成立する体系を作ることができる。

（その詳細は省く）

その結果できる体系を**複素数**という。

複素数は，$a+bi$ (a, b は実数)の形に表せる。

例

$(2+3i)(4-i) = 8+10i-3i^2 = 8+10i-3\times(-1) = 11+10i$

また，複素数を $a+bi$(a, b は実数)の形に表す方法は一通りに限ることも証明できる。

そこで，複素数 $a+bi$ に平面上の点(a,b)を対応させると，複素数はすべて平面上の点で表せることになる。しかも，異なる点は異なる複素数を表す。

用語　　i を**虚数単位**という

（２）複素数の計算

（仮称）十進 BASIC で複素数の計算ができる。

マウスで  をクリックすると複素数モードになる。

複素数 $a+bi$ をプログラム中に書くときは COMPLEX(a,b)と書く。

例　(prog5.BAS)

```
10 PRINT COMPLEX(2,3)*COMPLEX(4,-1)
20 END
```

結果は (11　10)　のように表示される。

$i=\sqrt{-1}$ なので，次のようにプログラムを書いてもよい。(prog6.BAS)

```
10 LET i=SQR(-1)
20 PRINT (2+3*i)*(4-i)
30 END
```

４ マンデルブローのλ集合

$0 \leqq k \leqq 4$ のときには,

$f(x) = k\,x\,(1-x)$

$x_0 = 0.5$,

$x_n = f(x_{n-1})$ $(n = 1,2,3,\cdots)$

によって定められる数列の値は,

$0 \leqq x_n \leqq 1$

の範囲から出ることはない。

しかし，k の値が 4 より大きいと，その制限から外れ，大きな変化を示すようになる。

例 k=4.001 のとき

1.00025
-.0010005000625
-4.00700575256313E-3
-1.60962704525045E-2
-6.54377968603137E-2
-.2789493283751
-1.42740498676025
-13.8630248219361
-824.391974624246
-2722466.52571775
-29654718651186.9
-3.51848875546261E27
-4.95314322523896E55

このように，$k>4$ のときには ，$|x_n| \to \infty$ （$n \to \infty$）となる。

（注 ‖ は絶対値を表わす。）

ここで，k の取る値を複素数に拡張し，k の代わりにλという文字を用いて

$\lambda = a + b\,i$ (a,b は実数，$i^2 = -1$)

とする。

λはギリシャ文字でラムダと読む。（英語では lambda）

複素数 $z = x + y\,i$ に対して，その絶対値 $\|z\|$を $\|z\| = \sqrt{x^2 + y^2}$ によって定義する。

$z_0 = 0.5$,

$z_n = \lambda z_{n-1}(1 - z_{n-1})$ $(n = 1,2,3,\cdots)$

で定義される数列 $\{z_n\}$ に対し,

すべての自然数 n について$|z_n| < M$

となるような正数 M の存在する複素数λの集合をマンデルブローのλ集合という。

実験

2 数 a,b を入力すると，$\lambda = a+bi$ とし，

$z_0=0.5$,

$z_n = \lambda z_{n-1}(1- z_{n-1})$　(n =1,2,3,…)

で定義される数列の第 1 項から 250 項までを計算するプログラムを作ってみる(prog7.bas)。

```
10 INPUT a,b
20 LET lambda=COMPLEX(a,b)
30 LET z=0.5
40 FOR n = 1 TO 250
50    LET z=lambda*z*(1-z)
60    PRINT z
70 NEXT n
80 END
```

入力する数値によっては，250 項まで計算することができずに桁あふれのエラーになる。

そこで，エラーになるような場合は，

すべての自然数 n について$|z_n|<M$

となるような M はないとみなし，そうでないときは，上の条件を満たす M があるものとみなすことにする。

エラーになるかどうかを調べるのに例外状態処理の機能を利用する。

それを平面上の各点に対してテストする。

(prog8.bas)

```
100 LET left = -2
110 LET right = 4
120 LET height = (right - left)
130 SET WINDOW left, right, -height/2, height/2
140 ASK PIXEL SIZE px,py
150 DRAW grid
160 FOR a= left TO right step (right-left)/px
170    FOR b = -height/2 TO height/2 STEP height/py
180       LET lambda=COMPLEX(a,b)
190       LET z=0.5              ! 初期値
200       WHEN EXCEPTION IN
210          FOR n = 1 TO 250
220             LET z=lambda*z*(1-z)
230          NEXT n
240          PLOT a,b
250       USE
260       END WHEN
270    NEXT b
280 NEXT a
290 END
```

140 行の ASK PIXEL SIZE を実行すると，px,py には，それぞれ，横方向，縦方向のドット数から 1 を減じた数が代入される。

例外状態処理は，

WHEN EXCEPTION IN

　　エラーを起こす可能性のある文（複数でもよい）

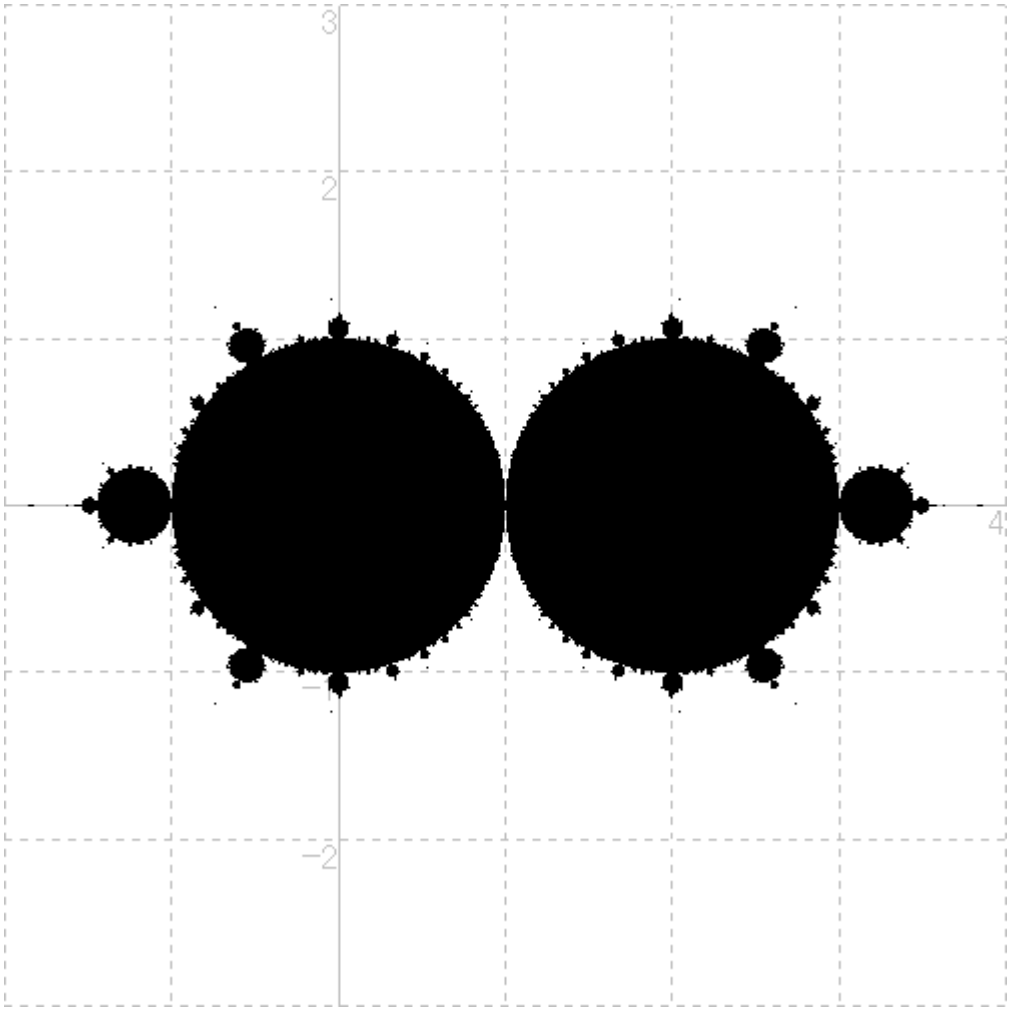
USE

　　エラーのとき実行する文（複数でもよい）

END WHEN

の形に書く。

WHEN EXCEPTION IN 行と USE 行の間の文でエラーが起こると，USE 行と END WHEN 行の間の文が実行されて，END WHEN の次の行に進む。



５．マンデルブローのμ集合

μを複素数の定数とするとき，

$f(x) = x^2 + \mu$

$x_0=0$,

$x_n = f(x_{n-1})$ $(n = 1,2,3,\cdots)$

で定義される数列について，

　すべての n に対して$|x_n|<M$

となるような M が存在するμの集合をマンデルブローのμ集合という。

（prog9.bas）