計算機プログラミングI 講義ノート #10

担当：井上 純一 (情報科学研究科棟 8-13), 赤間 清 (情報基盤センター)
URL：http://chaosweb.complex.eng.hokudai.ac.jp/~j_inoue/PROG2007/PROG2007.html

平成 19 年 7 月 6 日

目 次

この講義: 計算機プログラミング I で学習する内容は, 「ポインタ」「構造体」が今年度から後期開講の計算機プログラミング II にまわされた関係で, 前回で無事に全て終了しました. 従って, 「ポインタ」「構造体」を多用するような, より C 言語らしいプログラミング技法は後期になってから学習していくことになります. しかし, 現在の知識だけでもある程度のプログラムは書けるようになっているはずですので, 今回と次回 (7/20: 最終回) の 2 回にわたり, 今までの知識を使った応用問題をやってもらうことにします. 題材は自然界に数多く現れる**フラクタル**に関してです. 前回その使い方を学んだ gnuplot を用いて, そのフラクタル図形を描いてもらうことにします.

22 フラクタル図形の計算機による作成

下の図 11 を見てください. この図形は「フラクタル」と呼ばれる構造をもっています. この種の図形の特徴は, 部分的な構造の積み重ねで全体が構成され, 描かれている点です. 言い方を変えれば, **スケールを変えて図形を眺めても, 常に同じ図形として見える**ということです.

図 11 は私がフリーハンドで描いたものではなく, また, 木の影の写真でもありません. **これは簡単なルールに基づき, コンピュータに描かせたもの**です. 今回の計算機プログラミング I の講義・演習では皆さんにこのようなフラクタル図形を実際に描いてもらうことにします.

22.1 複素数で記述される力学系：決定論的なフラクタル

まずは「決定論的」なルールに従って描かれるフラクタルについて見て行きましょう.

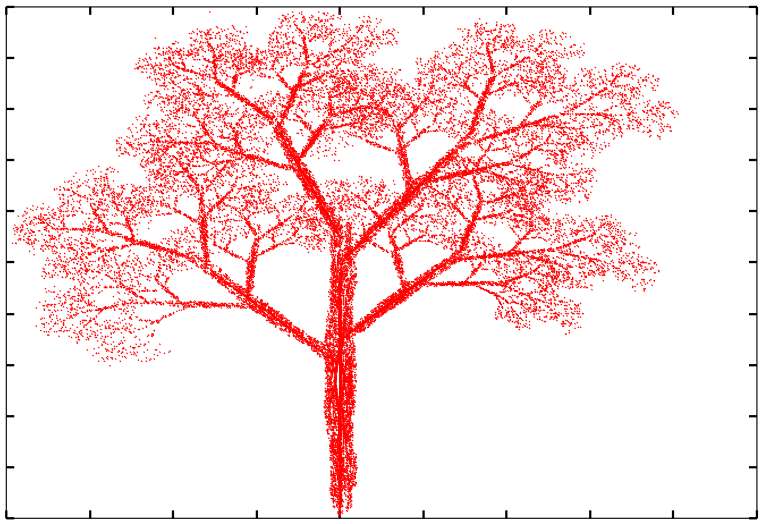

図 11: フラクタルな木.

22.1.1　ロジスティック写像の複素数への拡張

まずは前回の講義で提出して頂いた課題であった**ロジスティック写像**を思い出してみましょう.

$$x_{n+1} \quad = \quad ax_n(1-x_n) \tag{26}$$

写像 (漸化式)(26) に含まれるパラメータ a を変えていくと $x_0 \to x_1 \to x_2 \to \cdots \to x_n \cdots$ という時系列は「固定点」「周期軌道」「カオス」というような様々な振る舞いを見せましたが, a の変化に対してその周期軌道の分岐は図 12 のようになります. 今回のこの講義では写像の変数 x を複素数 z に拡張した場合に得ら

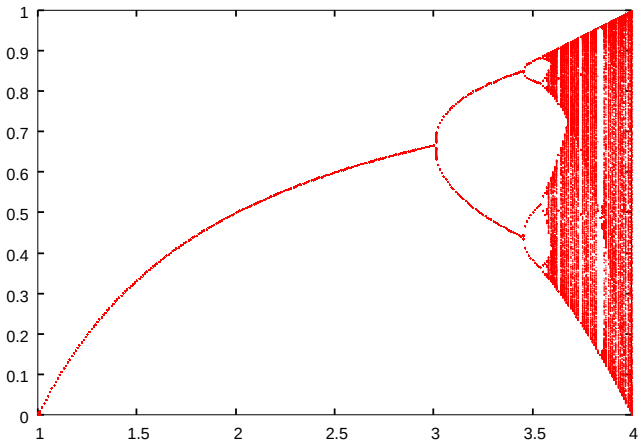

図 12: ロジスティック写像の分岐図

れる複素数版のロジスティック写像からは何がわかるのかを調べてみましょう. 複素数 z の実部を x, 虚部を y で表せば, z は $z = x + iy$ と書けるので, これをロジスティック写像の式： $z_{n+1} = az_n(1-z_n)$ に代入

すれば

$$\begin{aligned} x_{n+1} + iy_{n+1} &= a(x_n + iy_n)(1 - x_n - iy_n) \\ &= a(x_n - x_n^2 + y_n^2) + iay_n(1 - 2x_n) \end{aligned} \tag{27}$$

が得られます. ここでパラメータ a は $a = a_R + ia_I$ として複素数に選ぶこともできますが, まずは簡単のため実数であるとして話を進めましょう.

さて, (27) 式の両辺の実部, 虚部をそれぞれ等しいと置くことにより

$$x_{n+1} = a(x_n - x_n^2 + y_n^2) \tag{28}$$

$$y_{n+1} = ay_n(1 - 2x_n) \tag{29}$$

が得られます. この連立漸化式 (28)(29) をある初期条件 $\boldsymbol{x}_0 \equiv (x_0, y_0)$ からスタートさせると, この初期条件の選び方が良ければ連立漸化式は複素平面 x-y 内のある点へと収束するか, 有限範囲内での周期軌道へと収束していきます. しかし, 一方で初期条件の選び方がうまくなければこの連立漸化式は収束せず, $|\boldsymbol{x}_\infty| \equiv \sqrt{x_\infty^2 + y_\infty^2} \to \infty$ のように発散してしまいます. 従って, 全ての可能な初期条件はその初期条件から出発した軌道が [発散してしまうもの] と [有限に留まるもの] との 2 種類に分類されることがわかります. つまり, 初期条件の作る集合：$\{\boldsymbol{x}_0\}$ は

$$\begin{aligned} \{\boldsymbol{x}_0\}_\infty &\equiv \{\boldsymbol{x}_0 |\ |\boldsymbol{x}_\infty| \to \infty\} \\ \{\boldsymbol{x}_0\}_{\rm finite} &\equiv \{\boldsymbol{x}_0 |\ |\boldsymbol{x}_\infty| < \infty\} \end{aligned}$$

でそれぞれが定義される部分集合 $\{\boldsymbol{x}_0\}_\infty$ と $\{\boldsymbol{x}_0\}_{\rm finite}$ に分類されます.

連立漸化式 (28)(29) を数値的に解き, 複素平面 x-y 内にこの $\{\boldsymbol{x}_0\}_{\rm finite}$ をプロットしてできる図は, その作り方が上述のように極めて単純なのにも関わらず非常に複雑な形状を持ちます. それを図 13 に示すことにします. この集合 $\{\boldsymbol{x}_0\}_{\rm finite}$ のことを**ジュリア集合**と呼びます. 図 13 では $a = 3.3$ に選びましたが, この

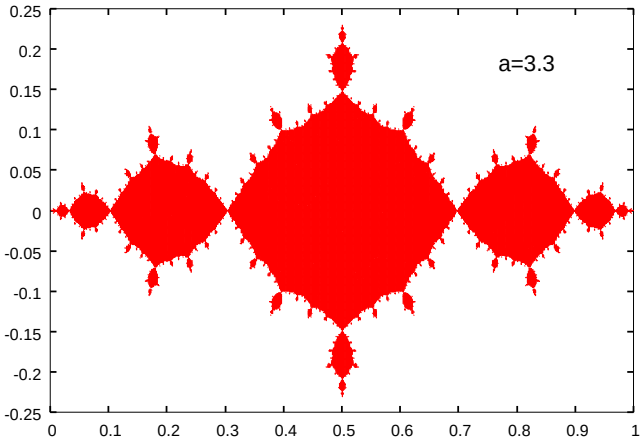

図 13: ジュリア集合. $a = 3.3$

場合に得られるジュリア集合はフラクタルであり, 前述の自己相似性を有します. a の値をいろいろ変えええれば様々なジュリア集合が得られます. 図 14 にその中から数例を載せましょう.

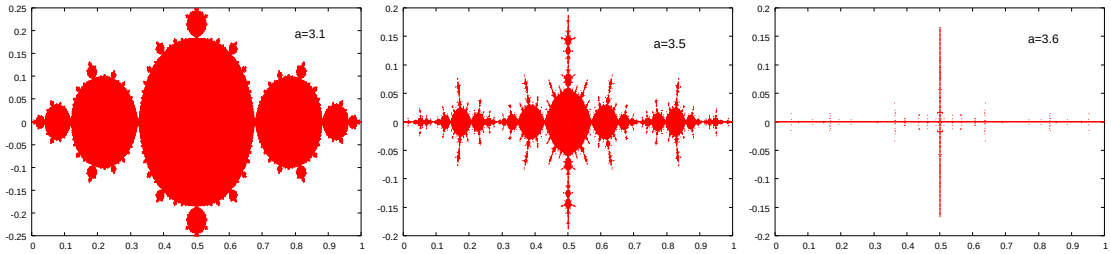

図 14: 左から $a = 3.1, 3.5$, 及び, $a = 3.6$ に対するジュリア集合.

22.1.2　複素力学系の数値解法：ニュートン法の復習

具体的に $z_{n+1} = f(z_n)$ のタイプの漸化式の解は $z_{n+1} = z_n = z$ としたときの方程式 $f(z) - z \equiv g(z) = 0$ の解です. 従って, ジュリア集合を求める際にはこの方程式 $g(z) = 0$ の解 z が有限であるかどうかを判定すればよいことになります. このとき, $g(z) = 0$ の解法としては既に**ニュートン法**を学んでいますので, それがここで使えるわけです.

ニュートン法を簡単に復習しておきますと, 方程式 $g(z) = 0$ の解の候補 $z = z_0$ で曲線 $y = g(z)$ に接する接線の方程式：

$$y \quad = \quad g(z_0) + g^{'}(z_0)(z - z_0)$$

の z 軸との交点：

$$z \equiv z_1 \quad = \quad z_0 - \frac{g(z_0)}{g^{'}(z_0)}$$

を解候補 z_0 の改良版とするもので, これをおし進めて n-番目の解候補と $n+1$ 番目の候補の間に成り立つ関係式：

$$z_{n+1} \quad = \quad z_n - \frac{g(z_n)}{g^{'}(z_n)} \tag{30}$$

を z_n に関する漸化式とみて, この収束点をもって解きたい方程式 $g(z) = 0$ の解とするものでした.

今の複素数版ロジスティック写像の場合には $g(z) = az(1-z) - z$ ですから, 反復式 (30) は

$$z_{n+1} \quad = \quad z_n - \frac{az_n(1-z_n) - z_n}{a - 2az_n - 1}$$

となりますが, 例によって $z_{n+1} = x_{n+1} + iy_{n+1}, z_n = x_n + iy_n$ をこれに代入し, x, y に関する漸化式に直してみると, やや煩雑ですけれど

$$x_{n+1} \quad = \quad x_n - \frac{(a - 2ax_n - 1)[(a-1)x_n - a(x_n^2 + y_n^2)]}{(a - 2ax_n - 1)^2 + 4a^2y_n^2} \tag{31}$$

$$y_{n+1} \quad = \quad y_n - \frac{y_n(a - 2ax_n - 1)^2 + 2ay_n\{a(x_n - x_n^2 + y_n^2) - x_n\}}{(a - 2ax_n - 1)^2 + 4a^2y_n^2} \tag{32}$$

が得られます. これら (31)(32) 式を様々な初期条件からスタートし, その解が有限範囲に収まるか否かをチェックして行けばよいことになります.

練習問題 22.1

gnuplot を用いてジュリア集合 ($a = 3.3$) を描くプログラムを作成せよ.

22.2　マンデルブロ集合

前節ではロジスティック写像をコントロールするパラメータ a を実数として扱いましたが, 先に述べたようにこれをも複素数に拡張することができます. $a = a_R + ia_I$ として (27) 式に代入し, 実部と虚部にわけた反復式を書き出してみると

$$x_{n+1} = a_R(x_n - x_n^2 + y_n^2) - a_I y_n(1 - 2x_n) \tag{33}$$

$$y_{n+1} = a_R y_n(1 - 2x_n) + a_I(x_n - x_n^2 + y_n^2) \tag{34}$$

が得られます. ジュリア集合を求めたときには, a の値を固定し, 反復式 (33)(34) が有限の値に収まるような初期条件の集合を複素平面内にプロットしました. ここでは逆に, 初期条件 $\{x_0, y_0\}$ が一つ与えられた場合, a_R, a_I の値を様々変えたときに, 反復式 (33)(34) の解が有限の値を持つような集合 $\{a_R, a_I\}$ を x-y 平面にプロットしたならばどのような図形が得られるのかを考えてみましょう. 図 15 に $x_0 = 0.5, y = 0$ を

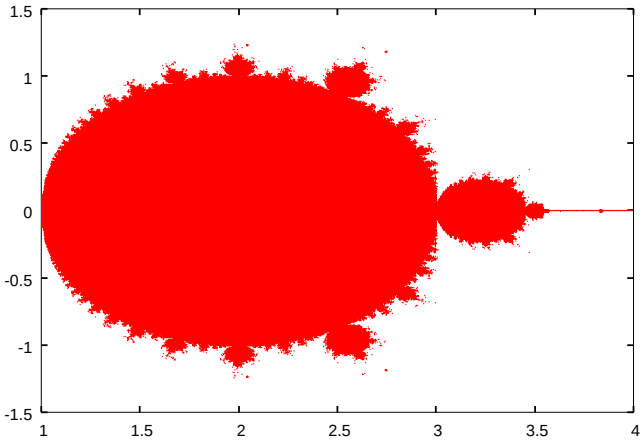

図 15: マンデルブロ集合. $x_0 = 0.5, y_0 = 0$ と初期条件を選んである.

初期条件に選んだ場合の結果を載せます. このような図形のことを**マンデルブロ集合**と呼びます.

練習問題 22.2

$x_0 = 0.5, y_0 = 0$ を初期条件に選んだ場合のマンデルブロ集合を gnuplot を用いて描くプログラムを作成せよ.

22.3　確率的フラクタル

ここからは確率を用いて描かれるフラクタルについて見ていきます. コンピュータ上で確率を扱うわけであるから, 擬似乱数を用いなければなりません. そこでまずはこの実験で用いる一様乱数を生成する関数を見ていきます.

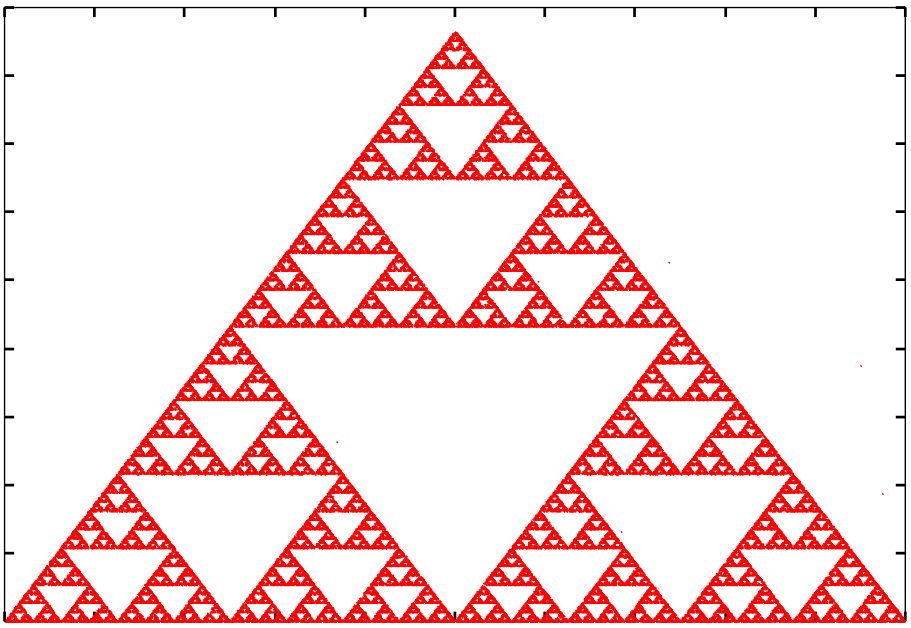

図 16: シェルピンスキー・ガスケット.

22.3.1　一様乱数生成関数

ここからは $[0,1]$ 間の一様乱数を用いますがここでは予めそのような乱数を発生させる関数を配布し, それを使ってもらうことにします. 次のアドレスを netscape で開いてください.

```
http://chaosweb.complex.eng.hokudai.ac.jp/~j_inoue/PROG2007/PROG2007.html
```

次にここにある randomnumber.c というプログラムファイルをダウンロードします. うまくダウンロードができたなら, このプログラムファイルを Xemacs を用いて開き, この中の乱数の「種」SEED の値を幾つか変えて乱数の系列を確認してみてください. ここで, 生成された乱数は rand.dat というファイルに格納されるようになっています. つまり, コマンドラインから

```
% gcc randnumber.c -lm
% a.out
% more read.dat
```

とすれば, 実際に $[0,1]$ の乱数が生成されたかどうかがわかるはずです. まずは各自がこれを確認してから次に進んでください.

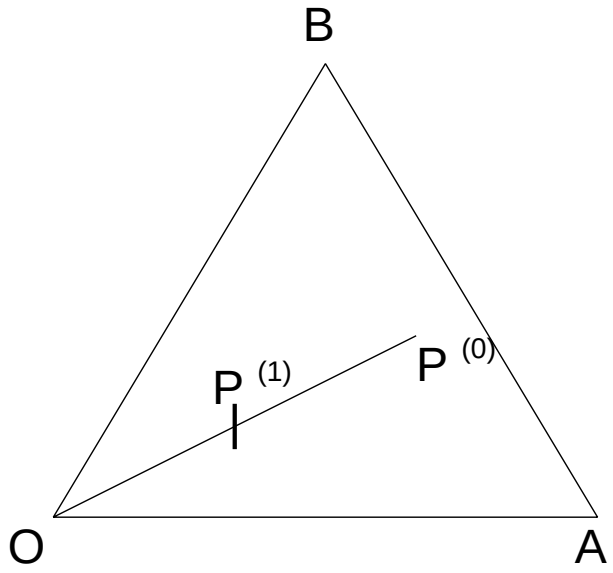

図 17: O が選ばれた場合には $\mathrm{P}^{(0)}$ と O の中点に $\mathrm{P}^{(1)}$ をおく.

22.3.2　シェルピンスキー・ガスケット

図 16 は**シェルピンスキー・ガスケット (Sierpinsky's gasket)** と呼ばれる典型的なフラクタル図形の一つです[1]. この図形は次に示すような非常に簡単なルールを用いて生成させることができます.

シェルピンスキー・ガスケット作成のアルゴリズム

(1) 図 17 のように頂点を $(X_1, Y_1) = (0, 0)$ (点 O), $(X_2, Y_2) = (2l, 0)$ (点 A), $(X_3, Y_3) = (l, \sqrt{3}l)$ (点 B) とする正三角形を考える.
(2) この正三角形内部の任意の点 $\mathrm{P}^{(0)} = (x_0, y_0)$ を選ぶ.
(3) 数 0,1,2 を**ランダム**に選び
- $0 \Rightarrow$ O と $\mathrm{P}^{(0)}$ を結ぶ線分の中点に点 $\mathrm{P}^{(1)} = (x_1, y_1)$ を置く.
- $1 \Rightarrow$ A と $\mathrm{P}^{(0)}$ を結ぶ線分の中点に点 $\mathrm{P}^{(1)} = (x_1, y_1)$ を置く.
- $2 \Rightarrow$ B と $\mathrm{P}^{(0)}$ を結ぶ線分の中点に点 $\mathrm{P}^{(1)} = (x_1, y_1)$ を置く.

(4) プロセス (3) を十分多数回繰り返す.

上記のアルゴリズムにより, 点列

$$\mathrm{P}^{(0)}(x_0, y_0) \to \mathrm{P}^{(1)}(x_1, y_1) \to \cdots \mathrm{P}^{(n)}(x_n, y_n) \to \cdots$$

を 2 次元平面内にプロットする. 具体的には

x_0 y_0
x_1 y_1
x_2 y_2

[1] ガスケット (gasket) とは「詰め物」の意味である.

x_3 y_3
.....

という形式でデータファイル (例えば gasket.dat) に格納する.

練習問題 22.3

シェルピンスキー・ガスケットを gnuplot により描くプログラムを作成せよ. (※ 慣れないうちは, **シェルピンスキー・ガスケット作成のアルゴリズム**の手順を既に学んだ PAD 図に書き出してみると良い)

[プログラム作成上のヒント]
(3) の条件分岐の部分は r←[0, 1] 間の一様乱数 として

$$r \in [0, 1/3] \rightarrow \text{点 O を選ぶ.}$$
$$r \in [1/3, 2/3] \rightarrow \text{点 A を選ぶ.}$$
$$r \in [2/3, 1] \rightarrow \text{点 B を選ぶ.}$$

を if 文で表現すればよい. また, if 文を用いなくても.

```
rr = (int)(r*3.0);
```

とすると rr には $0, 1, 2$ がそれぞれ $1/3$ の確率で現れることから, switch 文を用いて

```
switch(rr){
case 0: (点 O を選ぶ);
break;
case 1: (点 A を選ぶ);
break;
case 2: (点 B を選ぶ);
break;
}
```

としてもよい.

(注) 今回のレポート提出に関して

今回の 3 つの練習問題は全てレポートにて提出して頂きます. 提出期限はこの講義の**最終回 (7/20) に回収します**ので, それまでに提出してください. なお, 提出すべきものは各問題に対し, プログラムソース・コードと描いたフラクタル図形をプリントアウトしたものの 2 点です.

次回 (7/20) 講義は最終回であり, この講義全体を通じての簡単な復習・確認をし, **期末試験に関する重要な伝達事項**がありますので, 必ず出席してください.