今回の実験は, いろいろな図形が出てきます. よろしければ印刷してお持ち帰り下さい. 「KSEG」では「file」の「print」を選択して, 「Print in color if available」にチェックを入れることにより, カラー印刷を行うことが可能です. 「XaoS」では, 「File」の「Save image」によって画像を保存してから印刷して下さい.

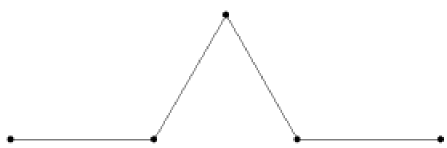
1 フラクタル

今から 30 年程前, ベノア・マンデルブローというポーランドの数学者によってフラクタル (fractal) という概念が生み出されました. これは, ラテン語の fracuts—不規則に壊れた状態—から派生した造語です. フラクタルは「自己相似性」という面白い特徴を持っています. この性質を我々が認識するには再帰的な操作を何度も行う必要がありますが, それはまさにコンピュータが得意とする仕事です. 今回は, コンピュータを利用することでフラクタルが持つ不思議な性質を体験することにします.

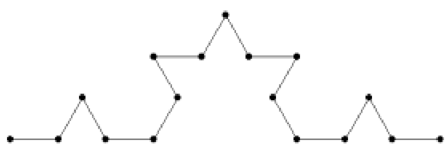
1.1 いろいろなフラクタル

平面上に線分を描いて, それを 3 等分します.

真ん中の線分を一辺とする正三角形を描き, その線分は取り除きます. すると, 4 本の線分によって山の形が描かれます.



今度は, 4 本の線分それぞれについて, 先程と同じ操作を考えます. すると, $4 \times 4 = 16$ 本の線分で山が 4 つ描かれることになります.



この操作を「無限に」続けます. こうして得られた図形を, **コッホ曲線**と呼びます.

コンピュータでは, 無限の操作を行うことはできないのですが, 概形を描くことは可能です. KSEG を用いて描いてみましょう.

実験 1. KSEG を起動して, 以下の操作を行なって下さい.

(1) 右クリックをすることで点を 2 つ作って下さい.
(2) Shift キーを押しながら左クリックをすることで 2 点を選択します.
(3)「play」の 「Quick play」から, koch.seg を選択します.
(4) Enter recursion depth という表示が出て来ます. これは, 「山を作る操作」を繰り返し行う回数を表しています. そこに数字を入れて, OK をクリックします. 数字は 5 ぐらいを入れれば十分概形が分かると思います. あまり大きくしすぎると, 表示までに時間がかかってしまいます.

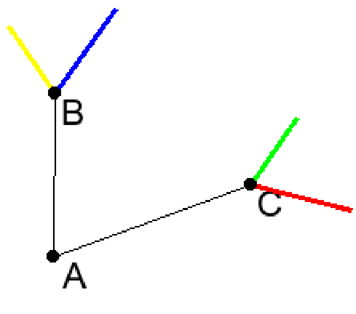
最初の 2 点動かすとそれに従って図形も動きます.

このように, コッホ曲線は, 図の一部を拡大しても同じ形が再び表れる, という面白い性質を持っています. このような性質を持つものを**フラクタル**と呼びます.

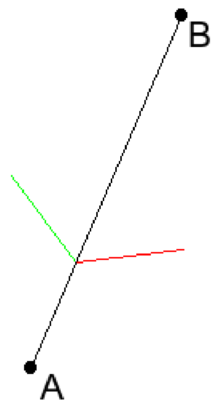
最初の条件を変えることによって, 様々なフラクタルを得ることができます. 実際にいくつか「作って」みましょう.

実験 2. KSEG の「File」から「Open」を選択して, ファイルを開きます.

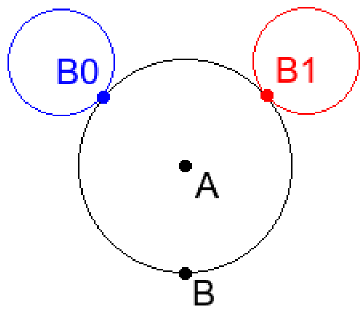
(1)「tree1.seg」 を開き, 点 A, B, C と右上の等式「Ratio $s_2/s_1 = 0.5$」 を順に選択して (Shift キーを押しながらクリック), メニューの「play」から「Quick play」にある, 「tree1.sec」を選んで下さい. これは, 「線分の一方の端から 2 本の線分を描く」という操作を繰り返すものです. 右にある点 P をマウスで動かすと, 次に描く線分の長さの比が変わります.



(2)「tree2.seg」 を開き, 点 A, B, C と右上の等式「Ratio $s_2/s_1 = 0.5$」と「Ratio $s_3/s_1 = 0.6$」を順に選択して, メニューの「play」の「Quick play」にある,「tree2.sec」を選んで下さい. 「Enter recursion depth」は 8 ぐらいが適当でしょう. これは, 線分上に一定比の内分点をとって, そこから 2 本の線分を描いています. 右にある点 P, Q をマウスで動かすと, 内分比と次に描く線分の長さが変わります.



(3)「circle.seg」 を開き, 4 点 A, B, B0, B1 と右上の等式「Ratio $s_0/s = 0.5$」,「Ratio $s_1/s = 0.5$」を順に選択して, メニューの「play」の「Quick play」にある,「circle.sec」を選んで下さい. これは, 円周上に 2 つの円を作るという操作を繰り返すものです. 円周上の 2 点 B1, B2 が次に描く円の位置, 右上にある点 R1, R2 が円の半径の比を決めています.



「samples」というフォルダにいろいろなフラクタルをまとめておきました. 時間があればご覧下さい.

以後, フラクタルについてもう少し厳密に考えて見ることにしましょう.

1.2 フラクタル次元

ある図形の面積や体積を求めさせる問題は, 小学生から大学生までやらされます. 長さ, 面積, 体積—これらを日常では使い分けて用いていますが, 数学では, 全て「体積」と定義されます[*1]. 特に 1 次元集合の体積を「長さ」と, 2 次元集合の体積を, 「面積」と呼んでいるのです.

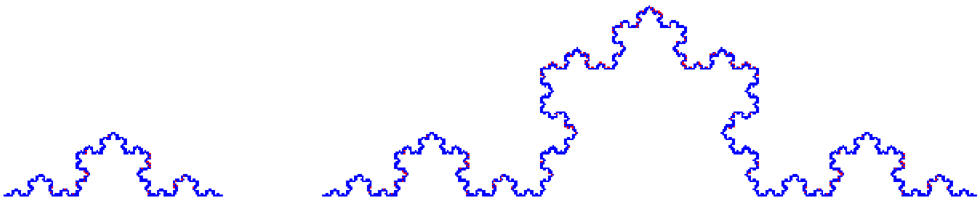
今, 縦, 横, 高さがそれぞれ 1 である立方体を考えます. この場合, 底面積は, 体積は共に 1 となります.

[*1] 4 次元以上の集合に対しても, 「体積」は定義できます.

それでは, この立方体を「拡大」した場合, 底面積や高さはどのように変化するでしょうか?

例えば辺の長さが 3 倍になった場合, 底面の面積は $3 \times 3 = 9$ 倍, 体積は $3 \times 3 \times 3 = 27$ 倍になります. つまり, 図形の縮尺を t 倍にすると, 長さは t 倍, 面積は t^2 倍, 体積は t^3 倍されるのです.

では, 先のコッホ曲線の場合はどうでしょうか? これは「曲線」と呼ばれてはいますが, 1 次元の体積である「長さ」を求めることはできません[*2]. そこで, コッホ曲線の定義に戻って考えることにします. 自己相似という性質から, コッホ曲線の大きさを 3 倍に拡大した場合には, 最初のコッホ曲線と同じものが 4 つ生じていることがわかります.



つまり, 縮尺を 3 倍にすると, 長さが 4 倍になるのです. 3 は 3 の 1 乗, 9 は 3 の 2 乗でしたが, 4 は 3 の何乗でしょうか? 実は, $3^a = 4$ を満たす a はきちんと存在することが知られており, $a = 1.261\cdots$ という値 (無理数) です. この a を, 数学では $\log_3 4$ と表します. 注意してほしいことは, この値は 1 よりも大きく, 2 よりも小さい, ということです. つまり, この集合は 1 次元と 2 次元の間にあるような性質を持っているのです. この値を, **ハウスドルフ次元**と呼びます.

一般に, 次のように定義されます[*3].

t 倍の縮尺に対して, 体積が s 倍になる集合のハウスドルフ次元を $\log_t s$ で定める.

ある集合のハウスドルフ次元がもとの次元よりも大きな場合, この集合をフラクタルと呼び, この値を, **フラクタル次元**と呼びます. フラクタル次元は, フラクタルの「複雑さ」を表す指標になっています.

雲や山の形など, 自然界で表れるものの一部もフラクタルで表すことができることが知られています. 単純な物理法則も時間をかけて繰り返すことによって複雑な現象を作り出すことができるわけです. コンピュータを用いることによってそのような構造を手軽に構成することが可能となりました.

[*2] 実際には, 無限大になってしまいます

[*3] 本当はハウスドルフ次元の厳密な定義があるのですが, 今はこうして求めたものと一致します.

2 カオス

今まで見たように, 単純な作業でもそれを繰り返すことで複雑な図形を描くことができました. これは, 単純な法則の元でも, 非常に複雑な現象が得られることを意味しています. それは**カオス**として研究されています. ここでは, カオスの一般論について論じることはせずに, 「複雑な現象」の典型例である, マンデルブロー集合について見ることで, そこに潜むカオスを調べることにします.

2.1 マンデルブロー集合

平面上の点を考える場合, その点の位置をどのように表わすか? ということを考えます.

高校までは, 主に (x, y) という座標表示を主に用いていました. これは, 基準となる点 (原点) と 2 つの直交する方向 x 軸方向, y 軸方向を定めて, 例えば x 軸方向に 3, y 軸方向に -2 進んだ先の点を $(3, -2)$ と表すという方法です.

ですが, 今は次の 2 つの情報によって点の位置を定めることを考えます[*4]:

「原点からの距離」, 「x 軸方向の向きとの角度」

この方法でも, 平面上の全ての点を決めることができます. 実際, 原点以外の全ての点は 1 対 1 に決まります.

そこで, 平面上に以下のように 2 つの規則で得られる点列を考えます:

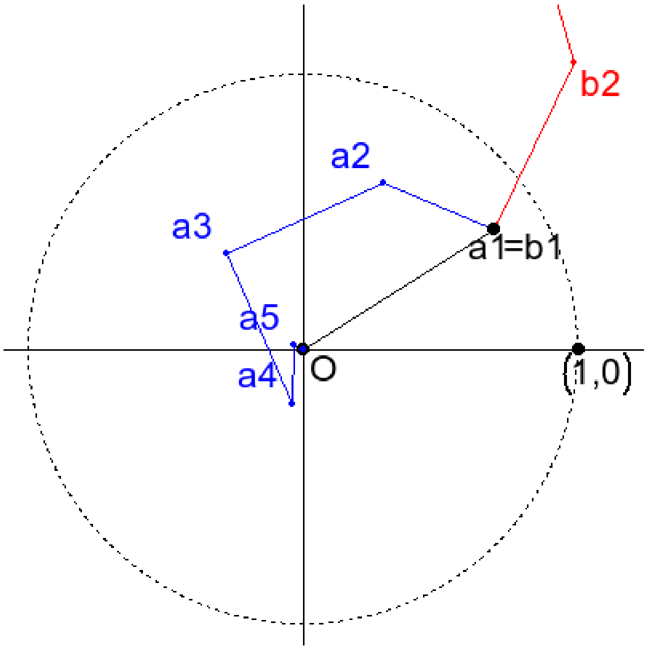
(操作 A)
(1) 原点から一定の方向に平行移動させた点を考えます.
(2) その点に対し, 長さを 2 乗, 角度を 2 倍した点を取ります.
(3) 以下, (2) の操作を繰り返します.

(操作 B)
(1) 原点から一定の方向に平行移動させた点を考えます.
(2) その点に対し, 長さを 2 乗, 角度を 2 倍した点を取ります.
(3) それを (1) で決めた方向と同じだけ平行移動します.
(4) 以下, (2),(3) の操作を繰り返します.

こうして得られた 2 つの点列はどのような挙動を示すでしょうか? KSEG で調べてみましょう.

[*4] 極座標表示, と呼びます

実験 3. KSEG から, mandel.seg を開いて下さい. 中心 O から 点 (a1=b1) へ行った点が, その後どのように動くかを線分で表わしています. 赤い折れ線は (操作 A) の挙動であり, 青い折れ線は (操作 B) の挙動を表しています. 緑色の点線は (操作 B) で得られた 50 番目の点の位置を示しています. 点 (a1=b1) を動かしてみて, 2 つの点列の動きがどのように変わるかを見て下さい.



複素平面を知っている場合, この操作は, 非常に簡単に表すことができます. 複素数は, 実数 x, y を用いて $x + yi$ と表すことができますが, 複素平面とは, これに対して, (x, y) という座標を対応させるものです. このとき, 複素数における和は平行移動, 積は回転と縮尺を表すことになります.

今, 最初の平行移動に対応する複素数を c とします. すると, 先の (操作 A) で決まる点列は

$$a_1 = c, \qquad a_{n+1} = a_n^2 \quad (n = 1, 2, \ldots)$$

で得られる (複素数の) 数列 $\{a_n\}_{n=1}^{\infty}$ を考えることに対応し, (操作 B) は

$$b_1 = c, \qquad b_{n+1} = b_n^2 + c \quad (n = 1, 2, \ldots)$$

で得られる数列 $\{b_n\}_{n=1}^{\infty}$ を考えることに相当します.

このように考えると, $\{a_n\}_{n=1}^{\infty}$ の挙動は簡単に分かります. 実際, $a_n = c^{2^{n-1}}$ となります. ゆえに, $\{a_n\}_{n=1}^{\infty}$ がはるか彼方へと飛び去ることと, c の絶対値 (原点からの距離) $|c|$ が 1 よりも大きくなることが同値になります[*5]. これは, 実験でも確かめることができます.

[*5] ちなみに, $|c| < 1$ ならば原点に収束し, $|c| = 1$ の場合は円周を回り続けます.

それでは, $\{b_n\}_{n=1}^{\infty}$ の方はどうでしょうか? 場合分けをして考えてみましょう.

$\{b_n\}_{n=1}^{\infty}$ は最初に与えられた複素数 c によって, 以下のいずれか 1 つの振舞いをします:

(1) はるか彼方に飛び去る ($\lim_{n\to\infty} |b_n| = \infty$: 無限大に発散).
(2) 平面上のある 1 点 d に近づく ($\lim_{n\to\infty} b_n = d$: 収束).
(3) N 個の同じ場所を回り続ける ($b_n = c_r$ $(n = qN + r)$: 周期的な軌道).
(4) それ以外 ($b_n =?$).

どのような振舞いをするかは, 最初の c の値を決めることで確定します. 点列が発散しないような c の集合, つまり, (操作 B) において, 条件 (1) 以外の性質をもつような c の点全体を**マンデルブロー集合**と呼びます.

2.2 バタフライ効果

点が「飛び去らないと」といっても, ある一点に近づいていく場合もありますし, 幾何模様を描くように同じ場所を回り続けることもあります. 各 c に対してそれぞれの点列の挙動を調べていてもマンデルブロー集合の全体像はあまり見えて来ません. コンピュータを利用して, この集合の概形を見ることにしましょう.

実験 4. XaoS というソフトウェアを起動して下さい. 起動して出た図形で, 黒い部分がマンデルブロー集合です. 他の色がついているのは, 発散する部分です. その色が薄いほど, 早く発散することを示しています. マウスの左, 右クリック, で図形をそれぞれ拡大, 縮小できます. 両方をクリックしている間は移動できます.

数字の「1」キーを押すと最初の状態に戻ります. 他の数字のキーを押すことによって, マンデルブロー集合とは別の規則で作られた図形を見ることができます.

実際に見て頂けると分かると思いますが, 非常に「複雑」な図形です. 注意深く見ると, (厳密なものではないのですが) 自己相似性も持っています.

この集合について, これまで多くの数学者が研究をしており, 現在も続けられています. 黒い部分は全てつながっている (連結) ことや, 縁 (境界) の「曲線」のハウスドルフ次元が 2 であることが知られています.

どうしてこのような複雑な図形なのか? それは, 先程の分類で条件 (4) を満たす点[*6]が存在することに起因します. このような点は, **カオス的な振舞いをする**と呼ばれています. それでは, 何故このような振舞いをする点が出て来るのでしょうか? その理由の 1 つとして, 「無限」の操作を行なっていることが挙げられます. 一般に, あるデータを使って何

[*6] 言い換えると, 条件 (1), (2), (3) のいずれも満たさない点

かのデータを出す場合, 最初の差が微小である場合は, 出て来たデータの差も微小であることが多いです[*7]. マンデルブロー集合で行なった操作—長さを 2 乗, 角度を 2 倍にして, 一定だけ平行移動させる—は確かに連続なのですが, この操作を何度も繰り返すうちに, 最初の小さな差がどんどん大きくなり, 結局全く異なった挙動を示す場合があるわけです. 数学の世界ではこのような性質を「**初期値鋭敏性**」と呼んでいます. カオスの研究者は, このような性質を持ったものを扱っています[*8].

以上のように

単純な操作の積み重ねから思いがけない挙動が生じることがある

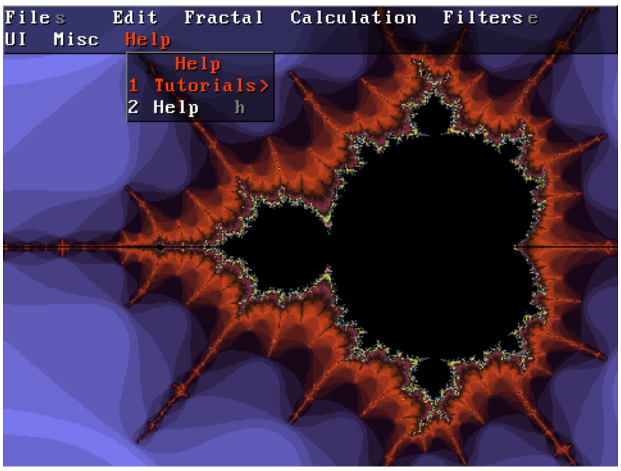
ということは, 逆に

ランダムに見える現象も実際は単純なシステムから構成されている可能性がある

と考えることもできます. 様々な研究によって, 大気の状態 (天気や雲の形状) や水滴の落下といった自然現象だけではなく, 人口の増加や株価の動向などといった社会的な現象もカオス的な振舞いをしていることが判明しています. 上記の「初期値鋭敏性」は「**バタフライ効果**」とも呼ばれます. これは「北京の蝶のはばたきがニューヨークの天気に影響を及ぼす (可能性がある)」という例えを指しています.

このように, 今まで手に負えなかった現象も, 「複雑さ」や「乱雑さ」もうまく定式化することで数学の世界でも扱うことができるようになりました. コンピュータはこのようなものを捉えるためにも大いに役に立っています.

最後に, XaoS のメニューから, Help の Tutorials から, 今回のようなお話のより詳しい解説を (英語ではありますが) 見ることができます. 興味のある方はご覧下さい.



[*7] このような性質を数学では「連続」と呼びます

[*8] 実はカオスの定義は研究者によって異なっているのですが, 多くの研究者の定義するカオスは上記の性質を持っています.