Fractals and dynamical systems
Berkeley Math Circle, October 23, 2019
Jacob Shapiro

1. What is a Fractal?

Somewhat disappointingly, there is no commonly agreed-upon mathematical definition of the term "fractal". However, if X possesses the following somewhat imprecise properties, we typically regard A as a fractal:

- A is a (usually bounded) subset of Euclidean space $\mathbb{R}^n$ or the complex plane $\mathbb{C}$.
- A has some kind of "rich" or complicated structure. One example of this is A looking the same at all scales. Roughly speaking, if you were to use a microscope to "zoom in" on A more and more, you would see the same patterns as before. Sometimes this property is called **self-similarity**. As non-examples, squares and spheres are usually not usually considered to be self-similar, because as you zoom in on a small patch of a square or sphere, you just see a line or plane.

Interestingly, many objects/phenomena in nature are fractal-like. So, in a way, fractals describe nature more accurately than the classical geometric shapes we are used to.

Main takeaway from this lecture: While fractals themselves are very complex, we can use simple mathematical concepts and formulae to generate and understand them.

Three examples of fractals are the Cantor set, the Sierpinksi Triangle, and the Mandelbrot Set.

2. Notation

We use X to denote a set of objects. If an object x belongs to X, we express that in mathematical notation by $x \in X$.

For example, X may be the set of real numbers $\mathbb{R}$ or complex numbers $\mathbb{C}$. Then $2 \in \mathbb{R}$, $\pi \in \mathbb{R}$, and $3 + i\pi \in \mathbb{C}$.

Or, X could be some set of higher dimensional objects, like row or column vectors in $\mathbb{R}^2$. Further still, the objects of X may be sets themselves—X could be the set of all closed and bounded subsets of $\mathbb{R}^2$.

A function $f : X \to X$ is simply a "rule" or formula that assigns to each point $x \in X$ another the point $y = f(x) \in X$. The **domain** of f is the set of all values f can take as an input, in this case X. The **range** or **image** of f, denoted by $\operatorname{Ran} f$, is the set of all output values that f assumes:

$$\operatorname{Ran} f := \{y \in X : \text{there exists } x \in X \text{ such that } f(x) = y\}.$$

For example, if $f(x) = \sin x : \mathbb{R} \to \mathbb{R}$, then the range of $\sin x$ is the closed interval $[-1, 1]$.

3. The Cantor set

The first fractal we construct is the classical Cantor set C, which is a subset of the closed interval $[0, 1]$. We obtain C by successive deletion of middle third open intervals:

$$I_0 = \{[0, 1]\},$$
$$I_1 = \left\{ \left[0, \frac{1}{3}\right], \left[\frac{2}{3}, 1\right] \right\},$$
$$I_2 = \left\{ \left[0, \frac{1}{9}\right], \left[\frac{2}{9}, \frac{1}{3}\right], \left[\frac{2}{3}, \frac{7}{9}\right], \left[\frac{8}{9}, 1\right] \right\},$$
$$\vdots$$

We define the Cantor set to be the intersection of all these sets:

$$C := \bigcap_{n=0}^{\infty} I_n = \{x \in [0, 1] : x \in I_n \text{ for all } n\}.$$

Exercise 3.1. Identify two points of $[0, 1]$ which are *not* in the Cantor set. What are some whole subintervals of $[0, 1]$ that are not in C?

Exercise 3.2. Identify at least five points which *are* in the Cantor set.

Exercise 3.3. Determine the length of the Cantor set. Or, put another way, does the Cantor set contain any interval of positive length? *Hint*: How many intervals make up I_n and what is the length of each interval?

4. Contraction mappings

Recall that we represent points of the Euclidean space $\mathbb{R}^n$ by n-tuples $(a_1, a_2, \dots a_n)$, where each $a_i \in \mathbb{R}$. The distance between two points of $\mathbb{R}^n$,

$$a = (a_1, a_2, \dots, a_n), \qquad b = (b_1, b_2, \dots, b_n),$$

is given by

$$|a - b| := \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_n - b_n)^2}.$$

Let $X \subseteq \mathbb{R}^n$. A function $f : X \to X$ is called a contraction if there exists $0 \leq s < 1$ such that, for all $x, y \in X$,

$$|f(x) - f(y)| \leq s|x - y|.$$

The number s is called a **contractivity factor** for f.

Example 4.1. if $f(x) = ax + b$, $a, b \in \mathbb{R}$, a necessary and sufficient condition that f be a contraction on $\mathbb{R}$ is $|a| < 1$.

Exercise 4.1. Consider the affine transformation:

$$T : \mathbb{R}^2 \to \mathbb{R}^2,$$
$$T(x, y) = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = \begin{bmatrix} ax + by + e \\ cx + dy + f \end{bmatrix}.$$

Show that T is a contraction if $\max\{|a|, |b|, |c|, |d|\} < 1/2$.

Exercise 4.2. Show that $f : [0, 1] \to [0, 1]$,

$$f(x) = \frac{1}{4}x^2,$$

is a contraction mapping.

Theorem 4.1 (Contraction mapping principle)**.** *Let X be a closed subset of $\mathbb{R}^n$ and suppose $f : X \to X$ is a contraction. Then f has a unique fixed point. That is, there is one and only one point $x_f \in X$ such that*

$$f(x_f) = x_f.$$

*Moreover, if we choose any initial condition $x_0 \in X$ and successively compute the **orbit** of x_0:*

$$x_0,\ f(x_0),\ f(f(x_0)),\ f(f(f(x_0))), \dots,$$

then this sequence converges to the fixed point x_f.

We denote the nth point in the orbit of x_0 by x_n or $f^{\circ n}(x_0)$. Note it is important that the domain and range of f be the same, so that the orbit of f is well-defined.

The proof of this theorem is beyond the scope of this lecture—it even holds in the much more general setting of a complete metric space. However, we can illustrate the main idea of this theorem with the following exercise:

Exercise 4.3. Consider the map

$$f : \mathbb{R} \to \mathbb{R},$$
$$f(x) = \frac{1}{2}x + \frac{1}{2}.$$

Show that f has a unique fixed point x_f. Prove that the orbit of any initial condition x_0 converges to x_f.

5. Iterated function systems

An **iterated function system** (IFS) is a closed and bounded subset X of $\mathbb{R}^n$, together with a collection $w_1, \ldots, w_m$ of contraction mappings,

$$w_i : X \to X, \qquad i = 1, \ldots, m,$$

where w_i has contractivity factor $0 \leq s_i < 1$. We will write an IFS succintly as

$$\{X; w_1, \ldots, w_m\}.$$

Let $H(X)$ denote the collection of all closed and bounded subsets of X. For each IFS $\{X : w_1, \ldots, w_m\}$, there is an associated **Hutchinson operator** acting on elements $E \in H(X)$:

$$H(X) \ni E \mapsto W(E) := \bigcup_{i=1}^{m} w_i(E) =$$
$$\{y \in \mathbb{R}^n : \text{there exists } x \in X \text{ and some } i \text{ so that } y = w_i(x)\}.$$

This map is a contraction mapping, with contractivity factor

$$s = \max_{i \in \{1,\ldots,m\}} s_i,$$

with respect to the **Hausdorff distance** between two elements of $H(X)$,

$$h(E_1, E_2) = \max\{\text{dist}(E_1, E_2), \text{dist}(E_2, E_1)\},$$

where

$$\text{dist}(E_1, E_2) = \max_{x \in E_1} \min_{y \in E_2} |x - y|.$$

Therefore, the Hutchinson operator W has a unique fixed point A,

$$W(A) = A,$$

which we call the **attractor** of the IFS. By the contraction mapping principle, A may be approximated by computing the orbit $W^{\circ k}(E)$ for any initial condition $E \in H(X)$.

The attractor A itself may have a very complicated structure, but the initial E we choose can be very simple (e.g., just a point, or X itself). By computing just a few iterates of E under the Hutchinson operator, we begin to see some of the important properties of A.

Exercise 5.1. Determine the attractor of the IFS

$$\left\{[0,1]; w_1(x) = \frac{1}{2}x, w_2(x) = \frac{1}{2}x + \frac{1}{2}\right\}.$$

Example 5.1. The Cantor set is the attractor of the IFS

$$\left\{[0,1]; w_1(x) = \frac{1}{3}x,\ w_2(x) = \frac{1}{3}x + \frac{2}{3}\right\}.$$

This fact can be used the show that the Cantor set consists of all points in [0,1] whose base three representation contains only 0's and 2's,

$$C = \left\{ x \in [0,1] : x = \sum_{j=1}^{\infty} \frac{\rho_j}{3^j},\ \rho_j \in \{0, 2\} \right\}.$$

Historical note: Iterated function systems were popularized by two Australian mathematicians, John Hutchinson and and Michael Barnsley. Hutchinson first introduced IFS's in a 1981 paper. Barnsley later developed several applications for them, including an application to image compression.

6. Python code to generate Sierpinski triangle

```
import numpy as np
import matplotlib.pyplot as plt

SIZE = 512

pic = np.zeros((SIZE, SIZE))
m = np.zeros((SIZE, SIZE))

for x in range(SIZE):
    for y in range(SIZE):
        pic[x,y] = 255

a1 = 0.5; b1 = 0; c1 = 0; d1 = 0.5; e1 = 0; f1 = 0
a2 = 0.5; b2 = 0; c2 = 0; d2 = 0.5; e2 = 256; f2 = 0
a3 = 0.5; b3 = 0; c3 = 0; d3 = 0.5; e3 = 128; f3 = 256

for z in range(8):

    for x in range(SIZE):
        for y in range(SIZE):
            m[x,y] = 0

    for x in range(SIZE):
        for y in range(SIZE):
            if pic[x,y]==255:
                m[int(a1*x+b1*y+e1),int(c1*x+d1*y+f1)] = 255
                m[int(a2*x+b2*y+e2),int(c2*x+d2*y+f2)] = 255
                m[int(a3*x+b3*y+e3),int(c3*x+d3*y+f3)] = 255

    pic[:] = m[:];

plt.figure()
plt.imshow(pic, cmap='gray')
```

7. Julia sets of quadratic polynomials

We consider complex polynomials,

$$Q_c : \mathbb{C} \to \mathbb{C}, \qquad c \in C,$$
$$Q_c(z) = z^2 + c.$$

The **filled-in Julia set** of the polynomial Q_c is

$$K(Q_c) := \{z \in \mathbb{C} : \text{the orbit } Q_c^{\circ k}(z) \text{ is bounded}\}.$$

The **Julia set**, $J(Q_c)$, of Q_c, is defined to be the boundary of $K(Q_c)$.

Exercise 7.1. Show that the Julia set of the polynomial $Q_0(z) = z^2$ in the unit circle in the complex plane.

While the preceding exercise furnishes us with a very simple Julia set, in general Julia sets have a very rich fractal structure.

The Mandelbrot set is

$$\mathcal{M} := \{c \in \mathbb{C} : \text{the orbit } Q_c^{\circ k}(0) \text{ is bounded}\},$$

Exercise 7.2. Show that if $c \in \mathbb{C}$ is such that $|c| > 2$, then c is *not* in the Mandelbrot set.

The Mandelbrot set has a very surprising and beautiful connection to the Julia sets of the polynomials Q_c,

$$\mathcal{M} = \{c \in \mathbb{C} : J(Q_c) \text{ is connected}\}.$$

8. Python code to generate Mandelbrot set

```
import numpy
import matplotlib.pyplot as plt

def mandelbrot(Re, Im, max_iter):
    c = complex(Re, Im)
    z = 0.0j

    for i in range(max_iter):
        z = z*z + c
        if(z.real*z.real + z.imag*z.imag)>4:
            return i

    return max_iter

columns = 2000
rows = 2000

result = numpy.zeros([rows, columns])
for row_index, Re in enumerate(numpy.linspace(-2,1, num=rows)):
    for column_index, Im in enumerate(numpy.linspace(-1,1,num=columns)):
        result[row_index, column_index] = mandelbrot(Re, Im, 100)

plt.figure(dpi=100)
plt.imshow(result.T, cmap='hot', extent=[-2,1,-1,1])
plt.xlabel('Re')
plt.ylabel('Im')
plt.show()
```

9. Python code to generate Julia sets (with comments)

```
import numpy as np
import matplotlib.pyplot as plt

#size of image
SIZE = 1000

image = np.zeros((SIZE, SIZE))

shades = 30

#view box
XMIN = -2.0
XMAX = 2.0
YMIN = -2.0
YMAX = 2.0

#c value for polynomial
c = -1/4 - (1/2)*1j
```

```
#roots
r1= 0.5*(1 - (1 - 4*c)**(0.5))
r2= 0.5*(1 + (1 - 4*c)**(0.5))
#r1= 0.5*(1 - (1 + 4*c)**(0.5))
#r2= 0.5*(1 + (1 + 4*c)**(0.5))

#modr1 = abs(r1)
#modr2 = abs(r2)

#iteration
def F(z):
    new=z**2 + c
    return new

#loop over all pixels
for x in range(SIZE):
    for y in range(SIZE):

         #convert units
         re = XMIN+x*(XMAX-XMIN)/SIZE
         im = YMIN+y*(YMAX-YMIN)/SIZE

         #set up point to iterate
         z=re+im*1j

         for i in range(shades):

             #use escape time
             #note multiple fixed points!
             #what do they do

             if abs(z) > 100 : break


             if abs(z-r1)<0.002*shades:break
             if abs(z-r2)<0.002*shades:break

             #this is the fixed point on the Julia set
             #if abs(z-r2)<0.01:break

             z = F(z)
         #first coordinate corresponds to imaginary axis
         #need to reflect because of the unusual coordinate
         #conventions in Python
         image[-y,x] = i


plt.imshow(image - np.min(image),cmap = 'rainbow')
plt.colorbar()
```