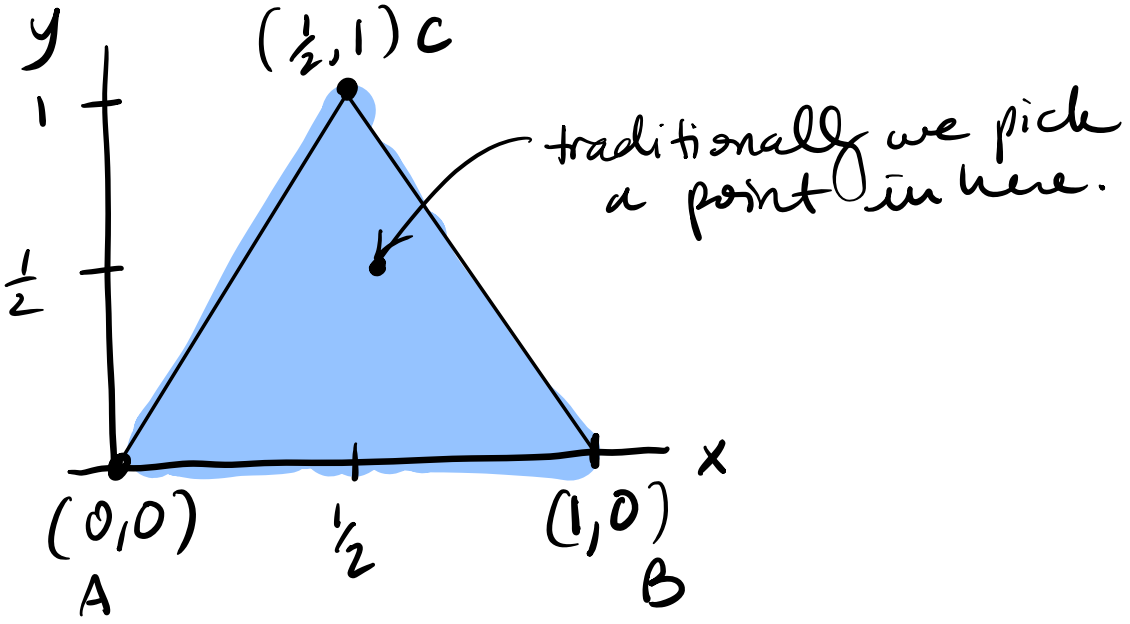
Chaos Game Explainer 1/19/23

- review what we saw w/the Chaos game previously

- Lets put some notation on our chaos game space:



Now, to understand the chaos game, let's see what happens to our point depending on which vertex we pick.

It moves halfway to our chosen vertex, so we can express that w/3 transformation functions:

$$f_A(x,y) = \left(\frac{x}{2}, \frac{y}{2}\right)$$

↑ moves x, y to $\frac{1}{2}(x,y)$ which is halfway to $(0,0) = A$

$$f_B(x,y) = \left(\underbrace{\frac{x}{2} + \frac{1}{2}}, \frac{y}{2}\right)$$

↑ height goes halfway to bottom

x goes halfway to 0 but then flips across $x = \frac{1}{2}$

moves us halfway to $B = (1,0)$

$$f_C(x,y) = \left(\underbrace{\frac{x}{2} + \frac{1}{4}}_{\text{takes us halfway to } \frac{1}{2}}, \underbrace{\frac{y}{2} + \frac{1}{2}}_{\text{takes us halfway to height 1}}\right)$$

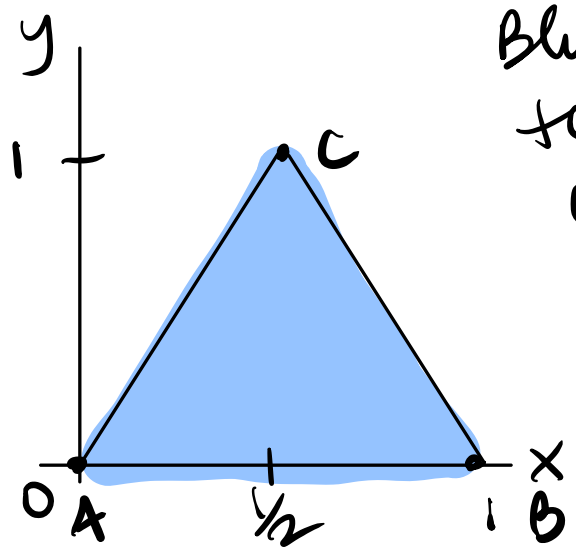
$$\frac{x + \frac{1}{2}}{2} = \frac{x}{2} + \frac{1}{4}$$

this moves us halfway to vertex $C = \left(\frac{1}{2}, 1\right)$

So these functions describe how each step of the chaos game will transforms our point at each step of the chaos game.

To understand what these functions do, instead of tracking a single point and randomly selecting a vertex each time, let's:

- Map where every point we could possibly have chosen goes, and
- let's run all 3 transformations every time.



Blue area is all the points we could have chosen.

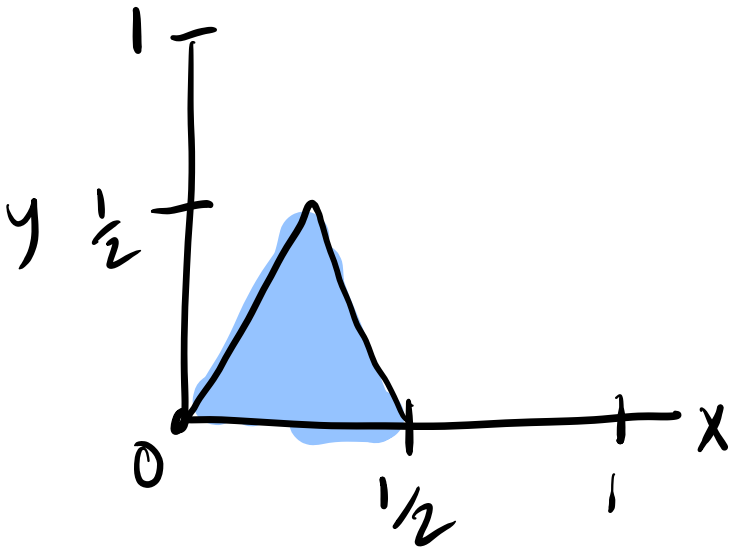
If we apply f_A, where do they go?

We can check the vertices and then we know.

$$f_A(A) = f_A(0,0) = (0,0)$$

$$f_A(B) = f_A(1,0) = (\tfrac{1}{2},0)$$

$$f_A(C) = f_A(\tfrac{1}{2},1) = (\tfrac{1}{4},\tfrac{1}{2})$$



$f_A(\triangle)$

It makes a copy of △ in the bottom left!

If we check f_B & f_C we find:

$$f_B(A) = f_B(0,0) = \left(\frac{1}{2}, 0\right)$$

$$f_B(B) = f_B(1,0) = (1,0)$$

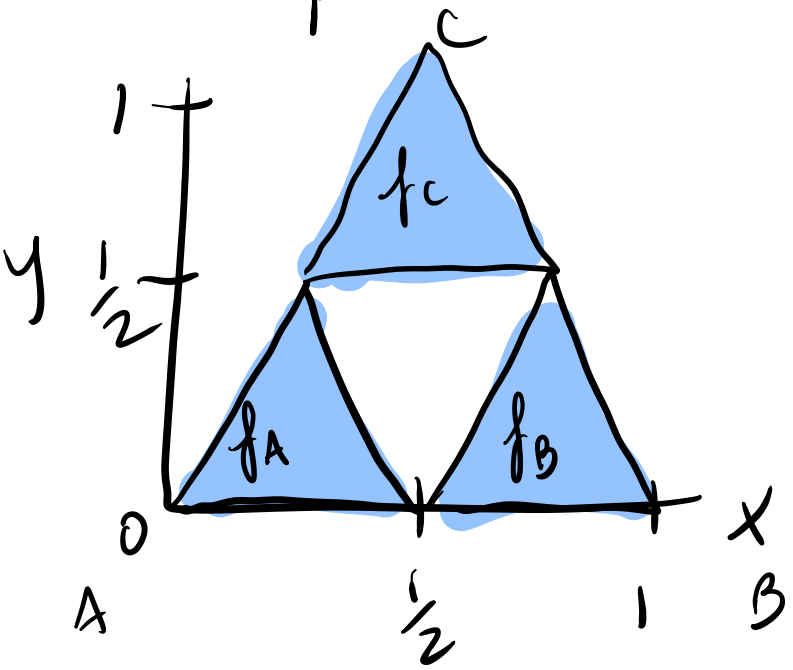
$$f_B(C) = f_B\left(\frac{1}{2}, 1\right) = \left(\frac{3}{4}, \frac{1}{2}\right)$$

$$f_C(A) = f_C(0,0) = \left(\frac{1}{4}, \frac{1}{2}\right)$$

$$f_C(B) = f_C(1,0) = \left(\frac{3}{4}, \frac{1}{2}\right)$$

$$f_C(C) = f_C\left(\frac{1}{2}, 1\right) = \left(\frac{1}{2}, 1\right)$$

Let's plot:



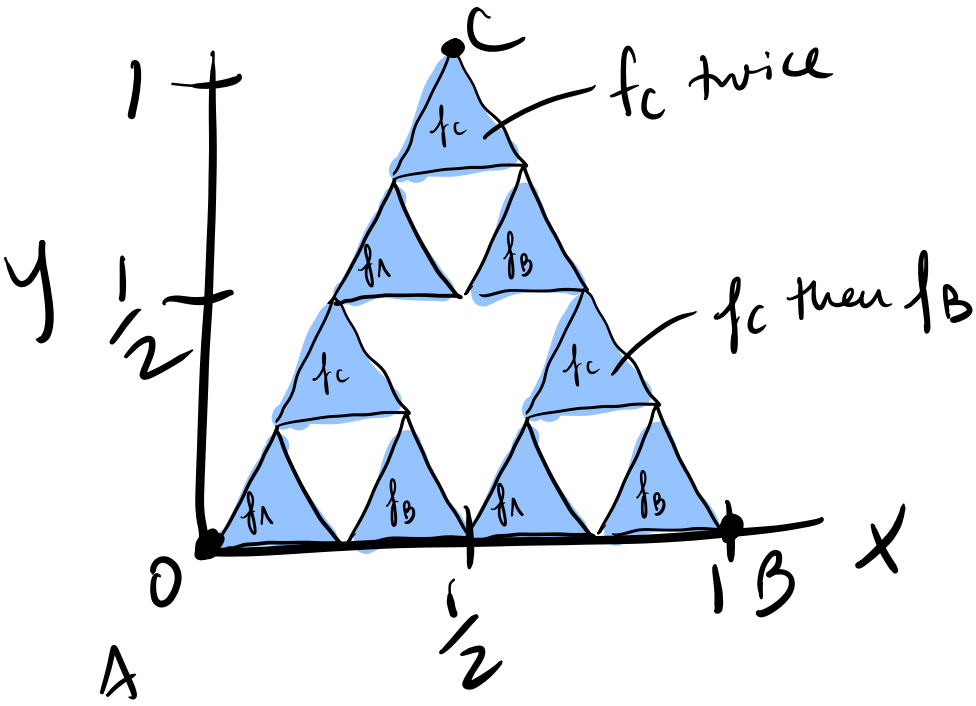
each transformation makes a copy of the original image scaled down $\frac{1}{4}$ and nearest the vertex it's named for!

This picture shows all the places we could end up from the first point we picked — any point in the blue original triangle will map to the corresponding point in each of the three triangles depending on the vertex we picked.

Note that this means that after one step, we have no points in the big central triangle! That forms an empty space (like the Sierpinski gasket).

Now, let's look at what happens to all of these points if we do the transformations again.

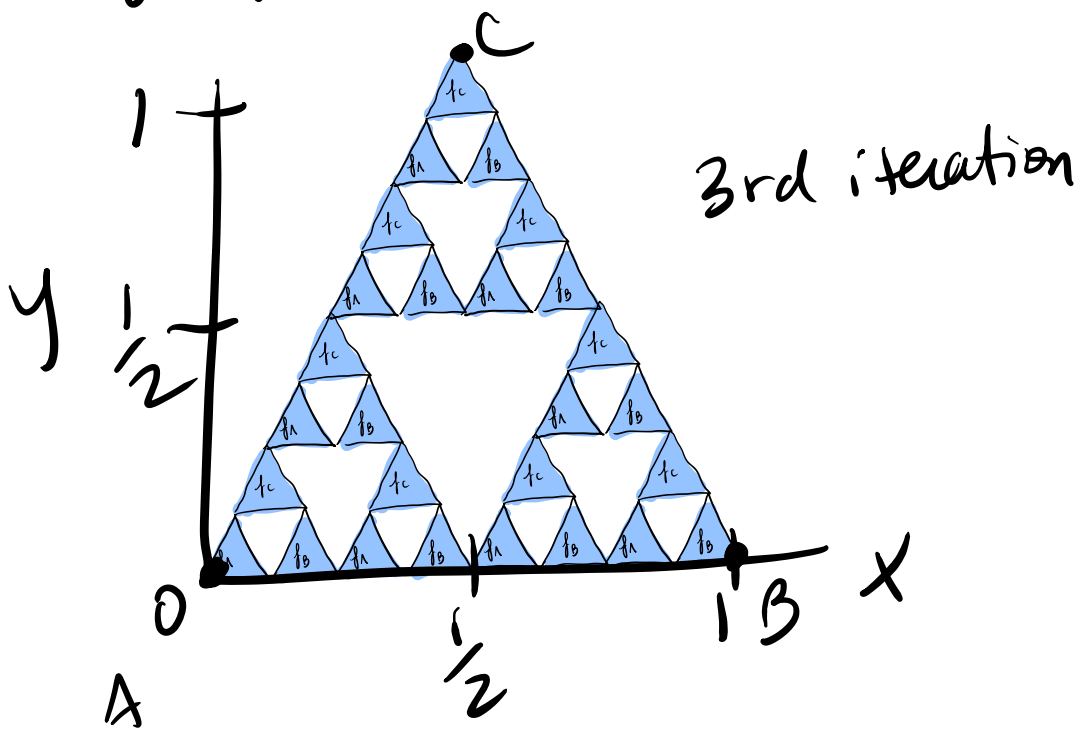
Note we could do f_A then f_C etc, so we take the whole image & shrink it.

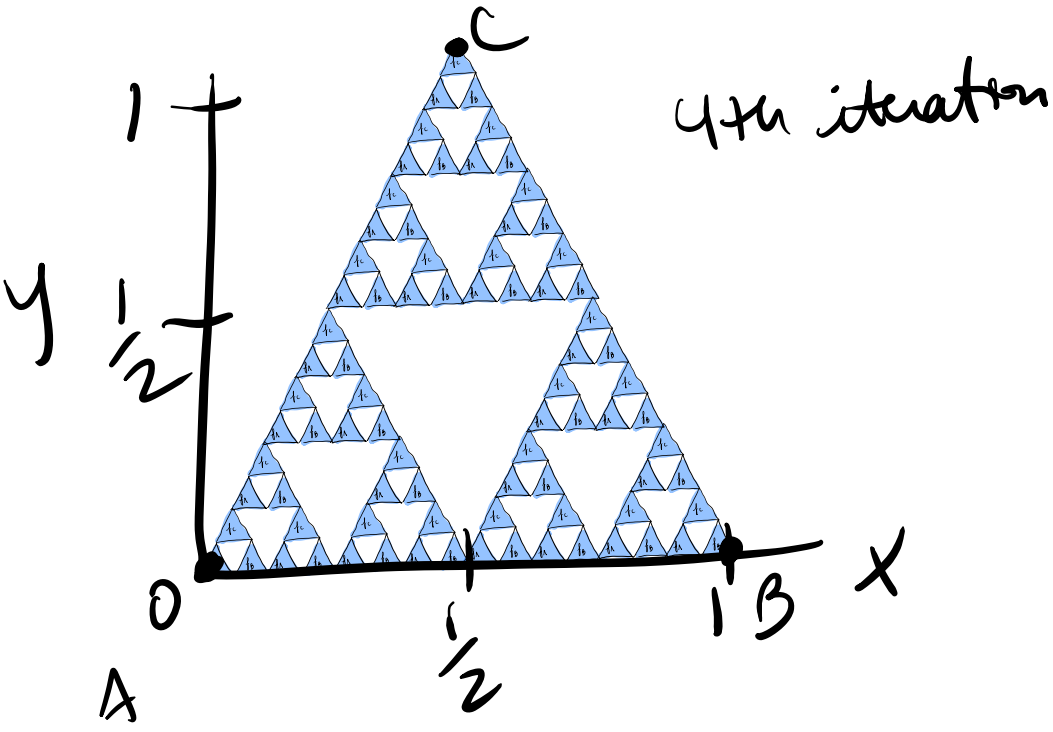


This shows all the places we could m on the second point.

Also note that this and subsequent steps have an empty main central triangle, so we can see that you can only ever have one point in the big central empty spot, 2 in the next smallest empty triangles, etc. So subsequent steps will land on further down "layers" of the triangle.

Let's try a few more steps:



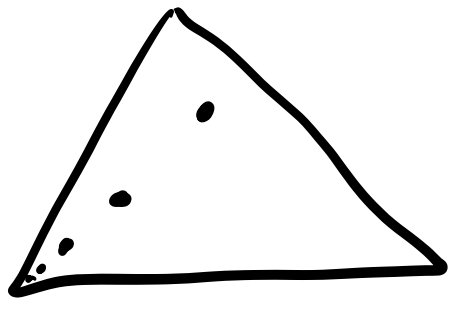


So we see that with each iteration we

land on a more refined version of the sierpinski Gasket!

Because we choose which function randomly, we apply each 1 ⅓ of the time, over time it eventually fills

The key here is the random choice of vertex — if we picked the same vertex over and over, we would just get a sequence of points in a straight line like zeno's paradox.



But – because we choose randomly, each time we pick f_A (for example) after having picked f_B or f_C previously, we will hop around between different chunks of the triangle. So eventually, it's as though we select many different points to apply each function to. (effectively we do many different sequences/paths in the pics above)

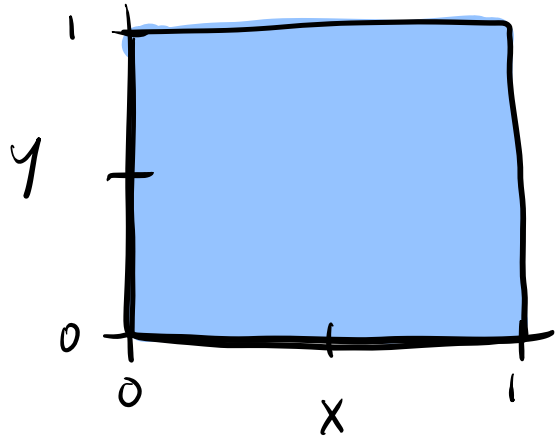
Important to note that you can only have up to n points in the

n^{th} largest empty space — i.e. after 3 steps no further points will be in the 3rd largest empty triangles, etc.

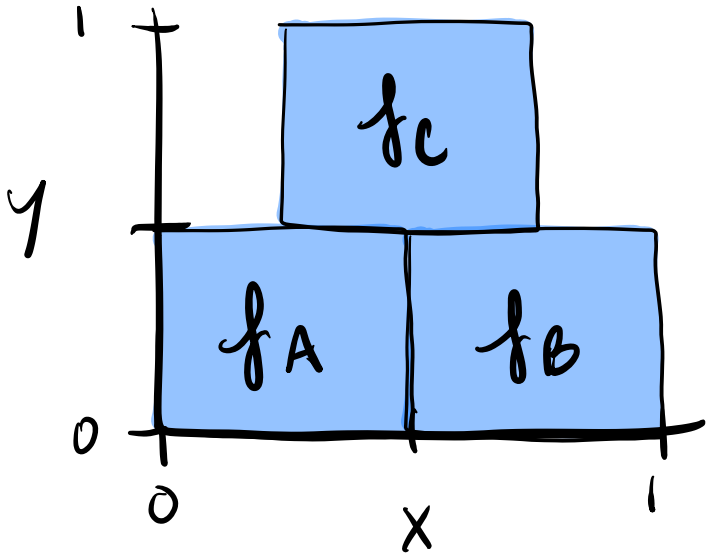
Why does it work if we pick a point outside the triangle?

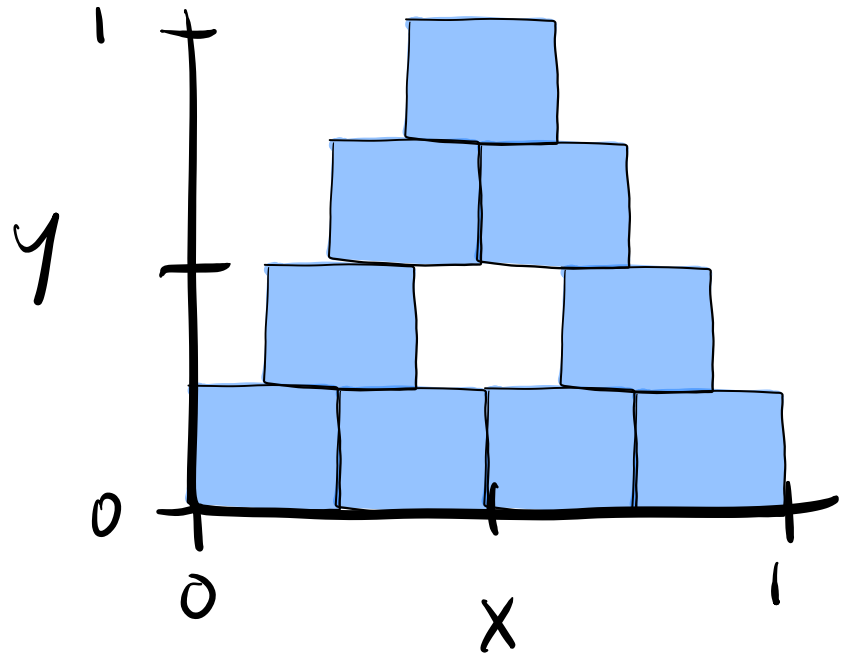
First, note that if we choose a point outside the triangle, going halfway to any vertex will bring us closer to the triangle, so the points will eventually approach the triangle — and so long as the furthest vertex is chosen often enough it will reach the interior of the triangle.

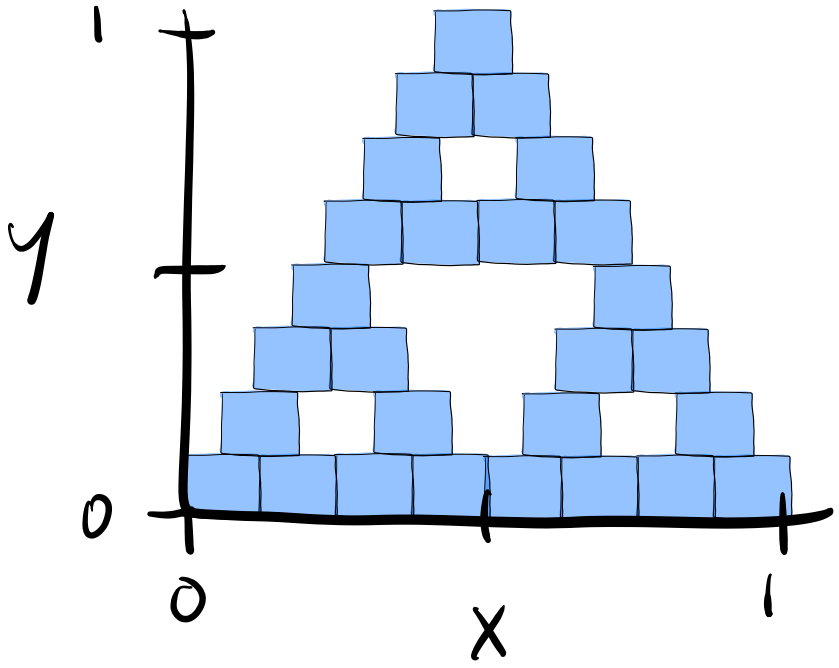
We can also see that the argument works even if we don't use the triangle but instead pick a point in the unit square:

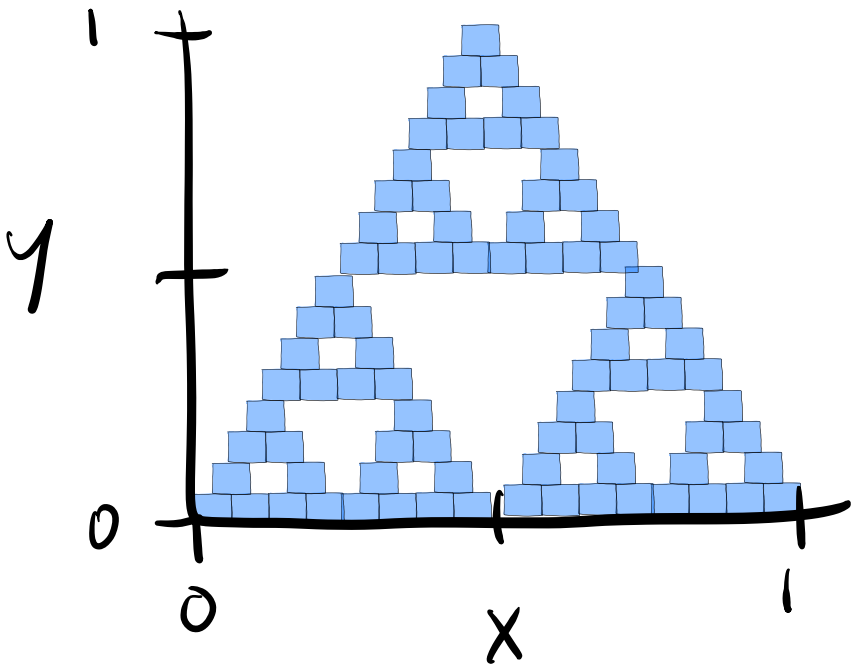


Now let's apply all three functions:




1
y
0
0
x
1


1
y
0
0
x
1



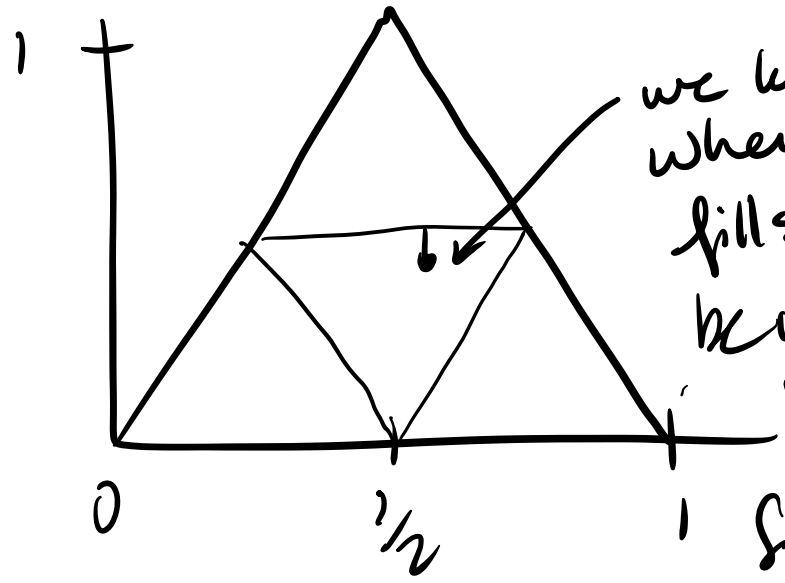
It will still converge correctly!

Another way to look at this —

- Points that start on the Sierpinski Triangle will stay on it. For example if we track a vertex at any step — it will always move to another vertex.

- What happens if you pick a point in the empty part of the Sierpinski Triangle?

E.g. what if we did:

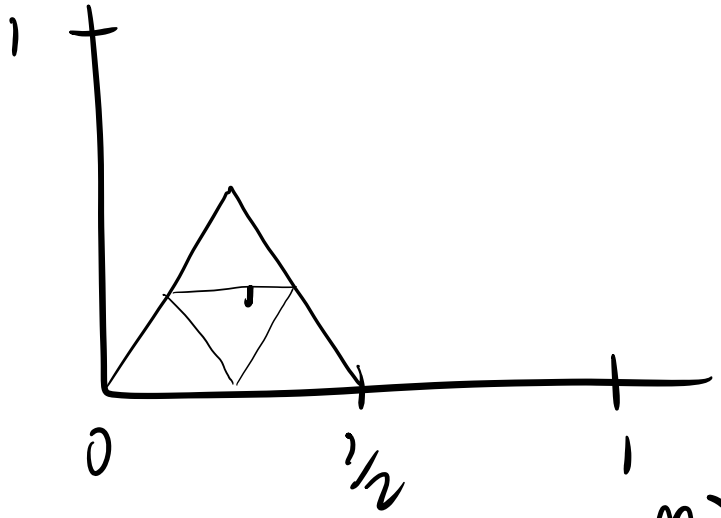


we know that when it all fills in, this will be mostly empty, w/only one point.

So what happens?

Well, 1st note that it's still true that the 2nd point will be in the empty spot on the next size down empty triangle, and in the empty spot on

the third size down, etc. But note that each iteration brings us closer to the filled triangle parts!



Eventually it will be (visually) indistinguishable from the points on the triangle anyhow – and it's still the case that we can have at most one point in the largest empty triangle, at most two in the next largest set of empty triangles etc.

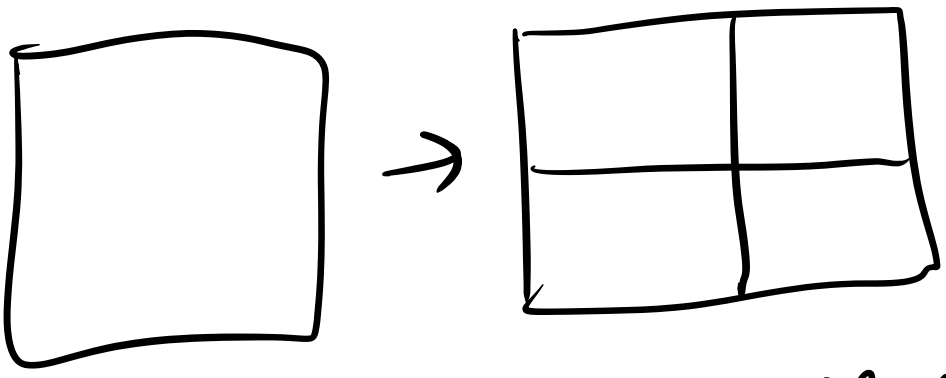
Also note that the precision of the computer helps us out a bit here — hard to distinguish once we're small enough

The Sierpinski Triangle acts as an attractor for this process.

What about other shapes?

Yep, we can do — can often intuit what the functions should be and therefore what is needed to make the fractal.

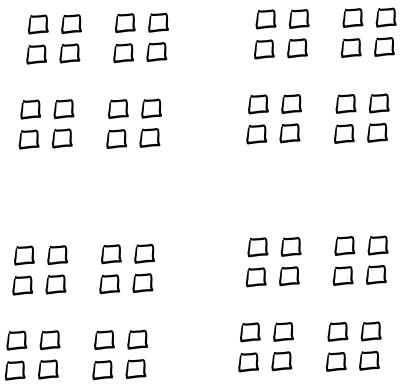
This also illustrates why $\frac{1}{2}$ doesn't work for the square:

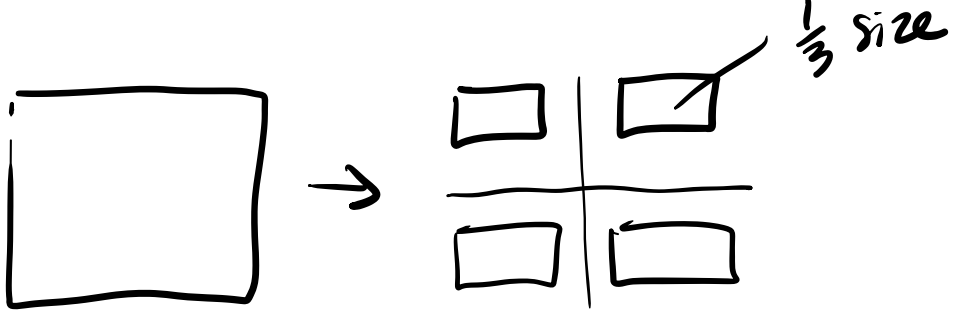


It fills the space! $\bigcup_i \text{im}(f_i(x,y))$ = the full square so it's possible that at each step you could land potentially anywhere. And also each further iteration will give the same picture – the whole square.

What if we shrink more than $\frac{1}{2}$?

Cantor Square / Cantor Dust





So — four vertices w/ 1/3 scaling.

$$f_A(x,y) = \left(\frac{x}{3}, \frac{y}{3}\right)$$

$$f_B(x,y) = \left(\frac{x}{3}, \frac{y}{3} + \frac{2}{3}\right)$$

$$f_C(x,y) = \left(\frac{x}{3} + \frac{2}{3}, \frac{y}{3}\right)$$

$$f_D(x,y) = \left(\frac{x}{3} + \frac{2}{3}, \frac{y}{3} + \frac{2}{3}\right)$$

Similarly the Sierpinsky carpet - needs 8 copies @ each step, so 8 functions

Also 1D probably could do!

1/2 wouldnt work but might be able to make something work. (e.g. cantor set)

A lot of these work by either changing the fraction toward a vertex or altering the order of vertices (e.g. can't be same one twice)

Can ever do non polygon shapes like the Barnsley Fern!