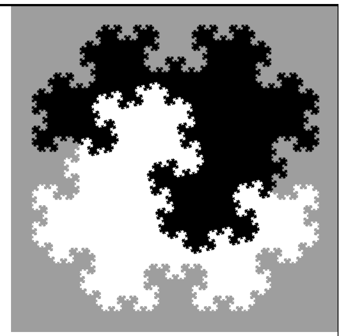


Fractals and the Mandelbrot Set

ENGR 151, Lecture 24: 3 Dec 14

Announcements

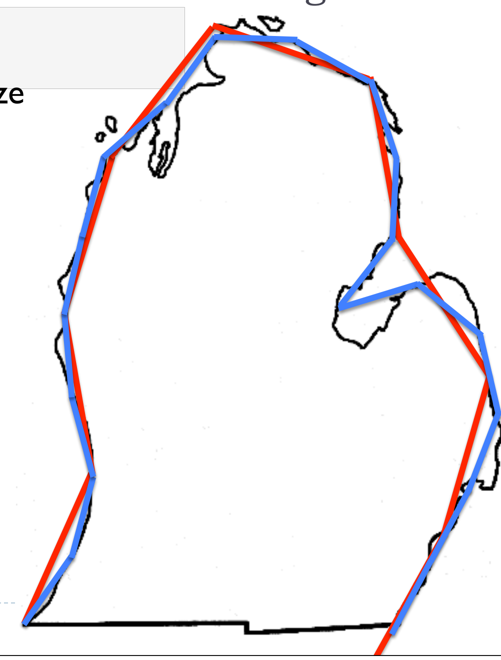
- Project 8 due Wed 10 Dec 11pm
- Final exam: Wed 17 Dec 4pm
  - Review: lecture of Wed 10 Dec
- Don't forget: course evaluations

How Long is the Coastline of Michigan (L.P.)?

- A: It depends on the size of your ruler

8.7 units

9.5 units

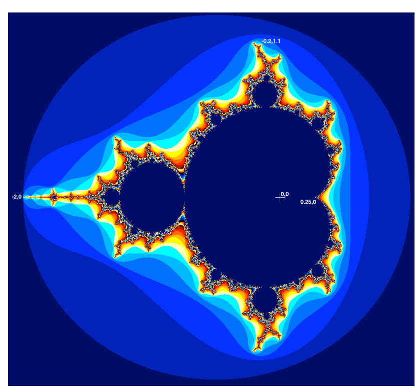


Fractals

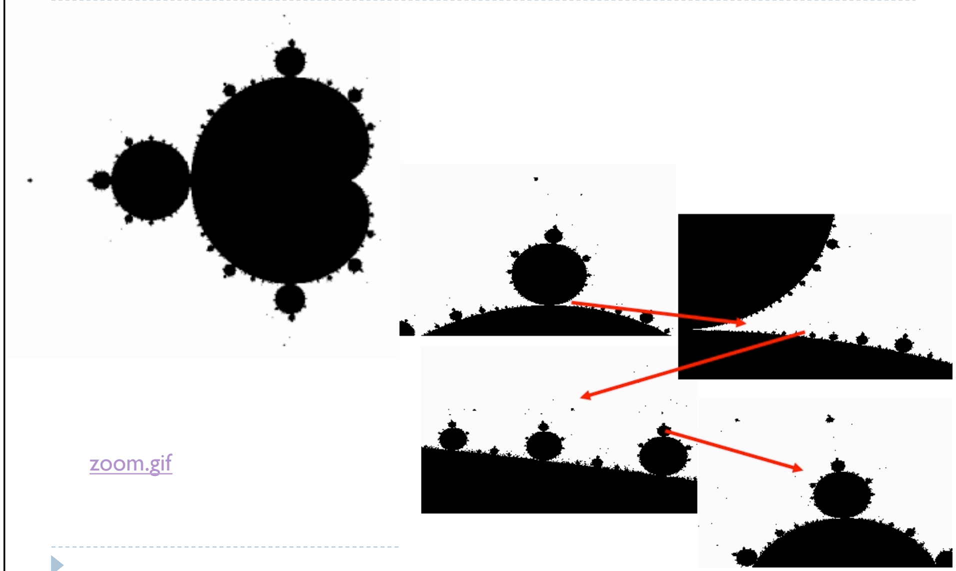
- Geographic shape exhibiting self-similarity
  - Parts maintain characteristics of the whole, in reduced size
- Name and definition due to mathematician Benoit Mandelbrot

1924-2010

- Most famous fractal: the Mandelbrot set



Fractal Boundary of the Mandelbrot Set



zoom.gif

Generating the Mandelbrot Set

- Consider the family of recurrence equations:

$$z_1 = 0$$

$$z_{m+1} = z_m^2 + c$$

c	1	0.1
z_1	0	0
z_2	1	0.1
z_3	2	0.11
z_4	5	0.1121
z_5	26	0.1126
z_6	677	0.1127
z_7	458330	0.1127
…	…	…

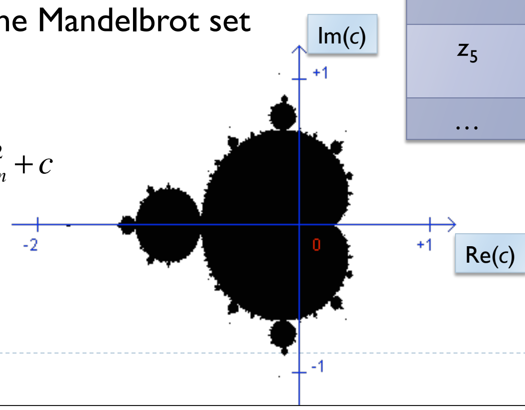
Grows very quickly for $c = 1$
very slowly for $c = 0.1$

The Mandelbrot Set (Definition)

- Let *c* be a complex number
  - real and imaginary part
  - can plot in 2-d
- If the z series stays finite, then *c* is in the Mandelbrot set

$$z_1 = 0$$

$$z_{m+1} = z_m^2 + c$$



c	$2i$	$0.5i$
z_2	$2i$	$0.5i$
z_3	–4+2i	0.25+0.5i
z_4	12–14i	–0.1875 +0.25i
z_5	–52–334i	–0.0273 +0.4062i
…	…	…

Magnitude of a Complex Number

- Fact: Once the z series reaches magnitude 2, it will escape to infinity.
- What is the magnitude (absolute value) of a complex number $a + b\mathbf{i}$?

A. abs(a)
B. abs($a + b$)
C. max(a, b)
D. sqrt($a^2 + b^2$)
E. None of the above

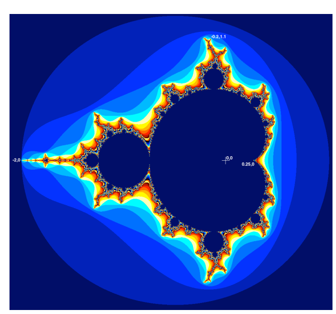
Jonathan Coulton

- music video

video by Pisut Wisessing

What about the Pretty Colors?

- Define the color value of a point according to how many iterations it takes to grow greater than 2 magnitude
- Normalize values over color range: divide iterations by max allowed
- For example, with max 10 iterations:
  - M(0.45) = 0.6 (6 iterations)
  - M(0.1) = 1 (stays less than 2)



Calculating Mandelbrot in MATLAB

- Write a function that takes as input:
  - a matrix `c`, representing a grid of complex numbers
  - a max number of iterations, `niters`
- Iteratively calculate the `z` series for each elt of `c`
  - Initial `z` (z_1) is 0
  - Repeatedly apply the recurrence, up to `niters` times

mandelbrotIterate

```
function res = mandelbrotIterate (c, niters)

z = zeros(size(c));
res = ones(size(c));

for m = 1:niters
  z = z.^2 + c;
end
```

Selective Iteration

- No point in iterating the `z` series once we exceed magnitude 2
- Use logical array called `active`, true for elements of `z` such that `abs(z)` ≤ 2
- Which assignment selectively updates active values?

A. `z = z.^2 + c;`
B. `z(active) = z.^2 + c;`
C. `z = z(active).^2 + c(active);`
D. `z(active) = z(active).^2 + c(active);`
E. None of the above

mandelbrotIterate (cont.)

```
function res = mandelbrotIterate (c, niters)

z = zeros(size(c));
res = ones(size(c));
active = (z==0);

for m = 1:niters
  z(active) = z(active).^2 + c(active);
  active = abs(z) <= 2;
end
```

Final step: set `res` according to which iteration (if any) `z` exceeded the magnitude threshold.

mandelbrotIterate (cont.)

```
function res = mandelbrotIterate (c, niters)

z = zeros(size(c));
res = ones(size(c));
active = (z==0);

for m = 1:niters
  z(active) = z(active).^2 + c(active);
  newactive = abs(z) <= 2;
  res(active & ~newactive) = m/niters;
  active = newactive;
end
```

One of These is not Like the Others

- Suppose we assign:

  `z = zeros(size(c));`

- which of these is different?

  A. `z == 0`
  B. `ones(size(c))`
  C. `~zeros(size(c))`
  D. `~(z == 1)`
  E. `logical(z+1)`

The Mandelbrot Set

- Now we have a function mandelbrotIterate that calculates the result for a matrix of complex values given a max number of iterations
- Next: design a function to create the matrix, call mandelbrotIterate, and plot the result in a pseudocolor plot.
- We call this function drawMandelbrot and pass it:
  - a range [xmin xmax ymin ymax]
  - a resolution (number of boxes in each direction)
  - a number of iterations (niter).

```
xmin=6, xmax=12, resolution=2 → xvals = [6, 12]
xmin=6, xmax=12, resolution=3 → xvals = [6, 9, 12]
xmin=6, xmax=12, resolution=5 → xvals = [6, 7.5, 9, 10.5, 12]
```

drawMandelbrot

```
function drawMandelbrot(range,resolution,niter)
% drawMandelbrot generates a Mandelbrot set plot

%  we will have to create c here

res = mandelbrotIterate(c,niter);
pcolor(res);
shading flat;
colormap jet;
```

drawMandelbrot (cont.)

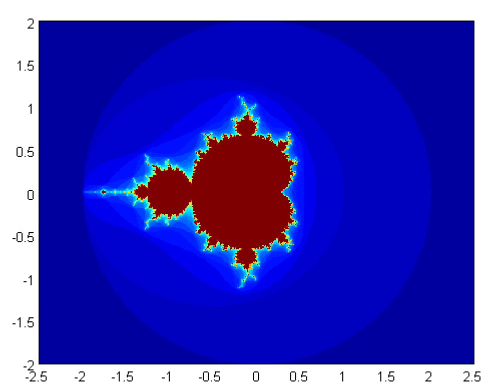
```
function drawMandelbrot(range,resolution,niter)
% drawMandelbrot generates a Mandelbrot set plot

xvals = linspace(range(1),range(2),resolution);
yvals = linspace(range(3),range(4),resolution);
[re, im] = meshgrid(xvals, yvals);
c = re + i*im;

res = mandelbrotIterate(c,niter);
pcolor(res);
shading flat;
colormap jet;
end
```

Generating the Mandelbrot Set Plot

- `drawMandelbrot([-2.5 2.5 -2 2],400,40)` produces the image:



A Deep Zoom

- https://www.youtube.com/watch?v=0jGaio87u3A