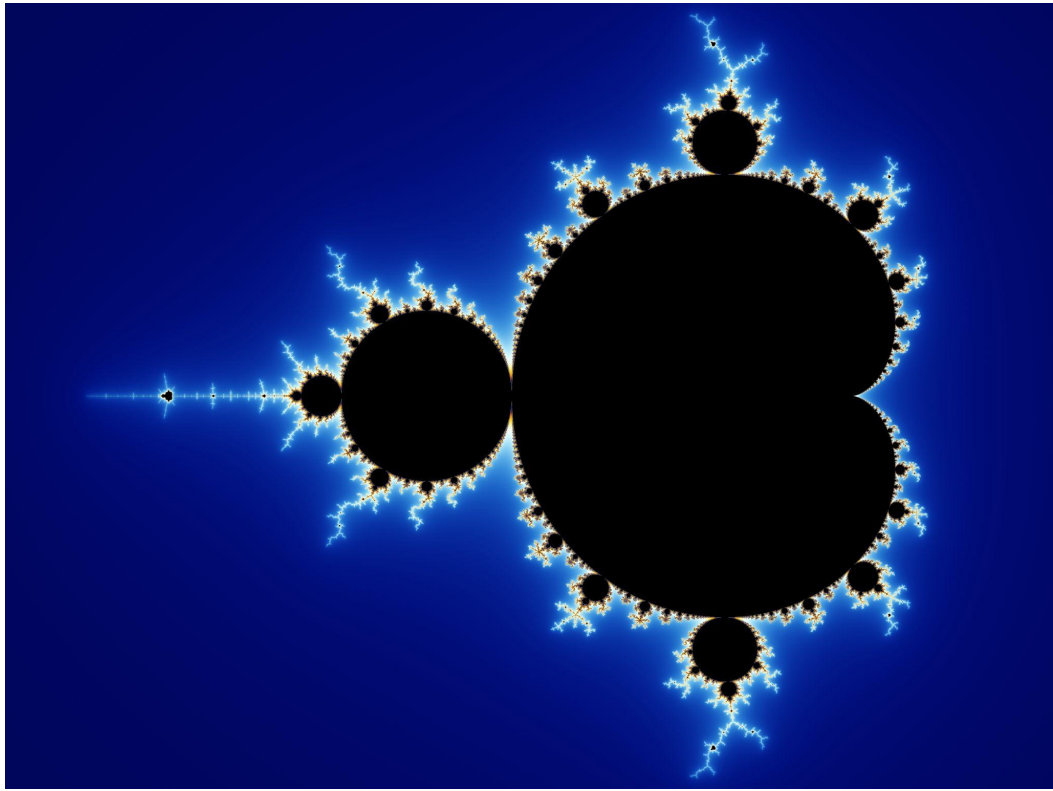
Mandelbrot Set Animation

Frenly Espino, Bennett Austin, Junlajak Jongpipattanakul

Mandelbrot Set

- A fractal within the complex plane defined by a recursive sequence

- Each term in the sequence must have a norm bounded by 2



$$\mathcal{M} = \{c \in \mathbb{C} \mid \{z_n\} \text{ is bounded, where } z_{n+1} = z_n^2 + c \text{ for all } n \in \mathbb{N}_0, \ z_0 = c\}$$

Coding the Mandelbrot Set

- Make a subset of the complex plane where we wish to define the Mandelbrot set

- We wrote a function that takes in bounds for x and y and then creates a matrix that acts as the subset of the complex plane

```
def makeMatrix(xmin, xmax, ymin, ymax, nx, ny):
    dx = (xmax-xmin)/nx
    dy = (ymax-ymin)/ny
    x = np.arange(xmin, xmax + 1, dx)
    y = np.arange(ymin, ymax + 1, dy)
    C = np.asarray([complex(x[k], y[j]) for k in range(0, nx) for j in range(0, ny)])
    C = np.reshape(C, (nx, ny))
    C = np.transpose(C)
    return C
```

Coding the Mandelbrot Set (cont.)

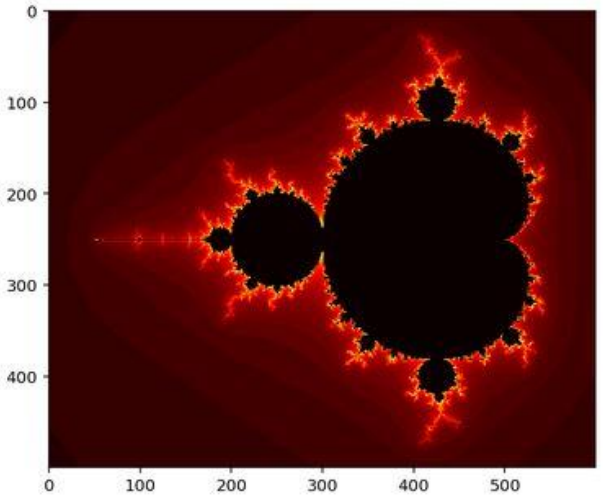
- Start with a zero matrix N of the same size as the previously defined matrix C
- For each coordinate z in C, we run the recursion on z
- If z's sequence is bounded, then we leave the corresponding entry in N alone
- Otherwise, the sequence diverges at a specific iteration n. Change the corresponding entry in N to n

```
def mandelbrotSet(C, maxiter):
    N = np.zeros((len(C[:, 1]), len(C[1, :])), dtype = int)
    for i in range(0, len(C[:, 1])):
        for j in range(0, len(C[1, :])):
            n = 0
            zold = complex(0, 0)
            z = complex(0, 0)
            while abs(z) <= 2 and n <= maxiter:
                z = zold**2 + C[i, j]
                zold = z
                n += 1
            if n >= maxiter:
                n = 0
            N[i, j] = n
    return N
```

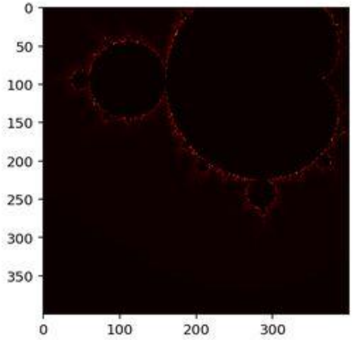
Zooming into the Mandelbrot Set

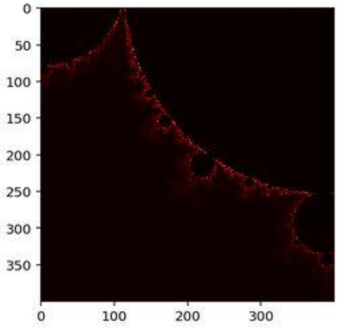
```
num_iterations = 300

with writer.saving(fig, "mandelbrotZOOMmore.mp4", dpi=200):
    for i in range(1,num_iterations+1):

        ax.clear() # first clear the figure

        zlimit=0.0000046   #zoom factor stuff
        slope=(zlimit-1.5)/299
        yint= 1.5
        zoomin=(slope*(i-1))+yint

        C = makeMatrix(-0.55075-zoomin, -0.55075+zoomin, 0.53165-zoomin, 0.53165+zoomin, 400, 400)
        maxiter = ((800/299)*(i-1))+200
        N = mandelbrotSet(C, maxiter)

        ax.imshow(N,cmap="hot")
        plt.draw()
        writer.grab_frame()
```

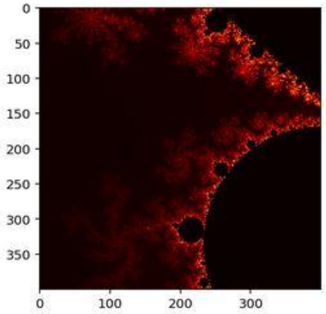
```
C = makeMatrix(-2.25, 0.75, -1.25, 1.25, 600, 500)
```

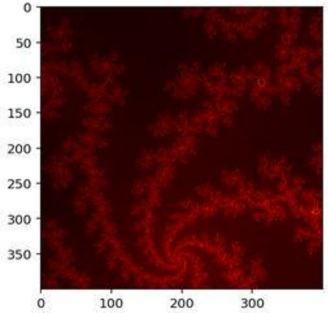


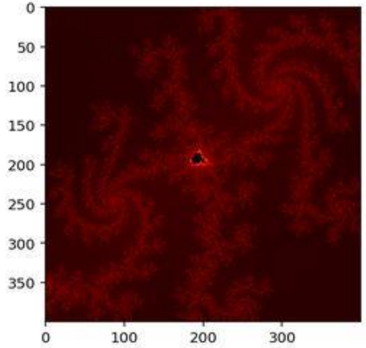
Desired Animation

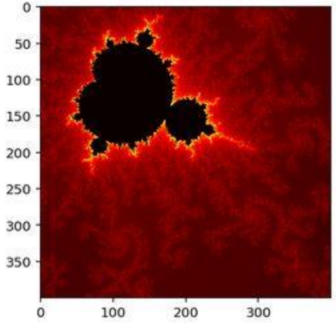


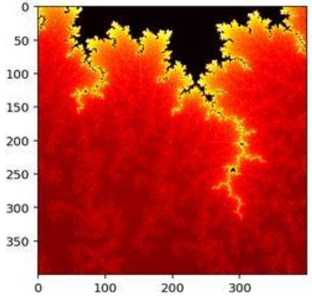












Animation Video

References

Weisstein, Eric W. “Mandelbrot Set.” *Wolfram MathWorld*, 2021, https://mathworld.wolfram.com/MandelbrotSet.html.