# CTA200H Assignment #2

Nikki Frazer
SURP Student

May 11, 2020

## Question 1

For each point in the complex plane $c = x + iy$, with $-2 < x < 2$ and $-2 < y < 2$, set $z_0 = 0$ and iterate the equation $z_{i+1} = z_i^2 + c$. Note what happens to the $z_i$'s: some points will remain bounded in absolute value $|z|^2 = \Re(z)^2 + \Im(z)^2$, while others will run off to infinity. Make an image in which your points $c$ that diverge are given one color and those that stay bounded are given another. (Once you have done this, you can try coloring the points that diverge using a colorscale that indicates the iteration number at which the given point diverged.) Try zooming in on a portion of the image and trying again.

### Method

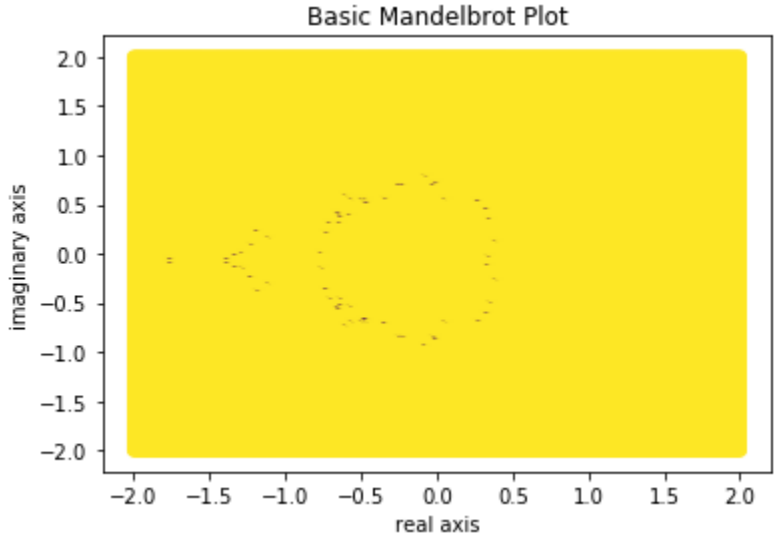


Figure 1: Mandelbrot Plot: Early Attempt

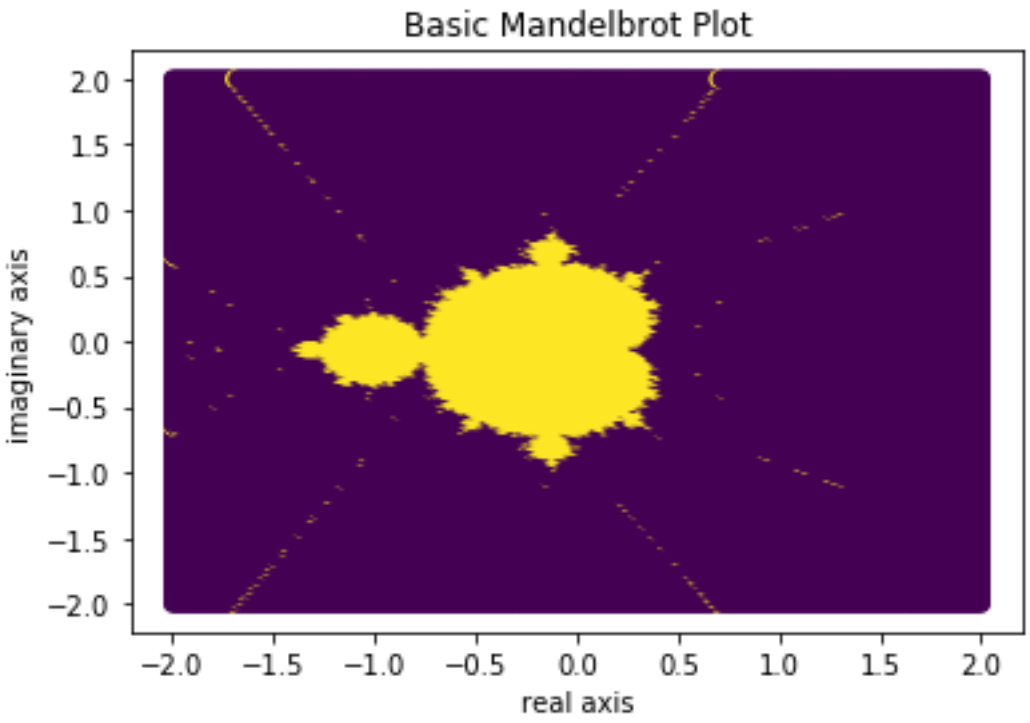


Figure 2: Mandelbrot Plot:Diverging vs Converging Points

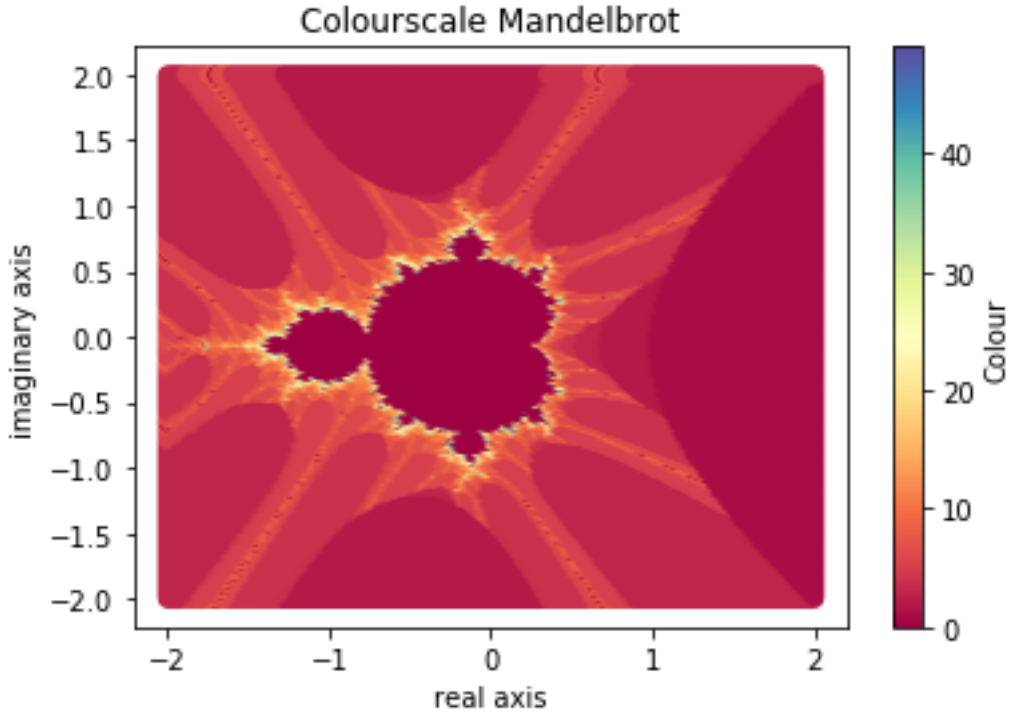


Figure 3: Mandelbrot Plot: Diverging points colourscaled

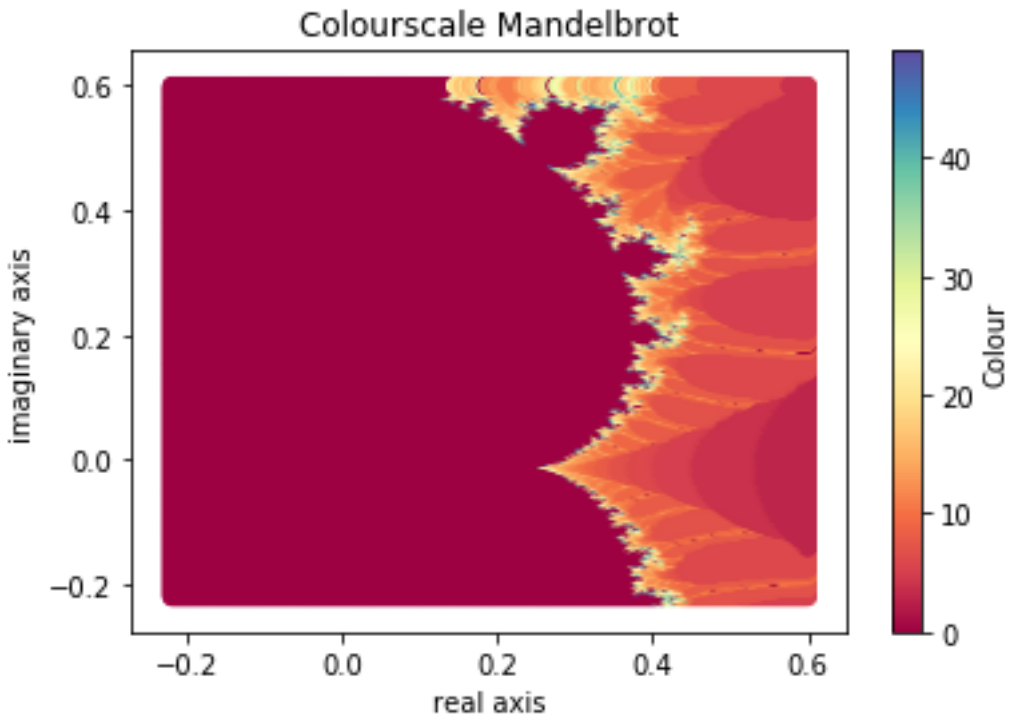


Figure 4: Mandelbrot Plot: Zoomed In

I began by drawing out the complex plane and computing iterations of Z by hand when given a point c. Then, I created a function that will determine whether a point diverges or converges after a certain number of iterations of Z, or, which points are actually in the Mandelbrot set. I computed some tests, and noticed that by splitting Z into its real and imaginary parts that by 50 iterations, points with real values over 2 will diverge, or become close to infinity. This function then creates a list of zeroes and ones depending on if a point converges or diverges. In order for the code to execute, it was important to include code that recorded the exact iteration when a point started to diverge. Next, I created another function which will plot the Mandelbrot set. I used meshgrid() from the numpy library to create the coordinates in the grid, and then plotted the imaginary axis, the real axis, and the colour code of zeros and ones together to create the Mandelbrot shape. For the colourscale plot, I adapted my first function to count how many iterations a point goes through before it diverges, and then used that data to map colour onto the plot.

## Analysis

My first few attempts at creating a plot showed me the general outline of the Mandelbrot plot, which helped me to refine the image by creating a density variable. The Mandelbrot set plot shows a distinct ink splotch pattern, which remarkably has been coloured according to which points diverge or converge according to the iterations of Z. The colourscale plot shows the pattern of the diverging points, and creates a striking visual. Zooming in on the plot shows the distinct swirls in different colours. It"s amazing that a short python script created such a complex plot!

# Question 2

The SIR model is a simple mathematical model of disease spread in a population. The model divides a fixed population of size $N$ into three groups, which vary as a function of time, $t$:

- $S(t)$ is those that are susceptible but not yet infected
- $I(t)$ is the number of infected individuals
- $R(t)$ is those individuals that have recovered and are now immune

The model can be described by a set of 3 first order differential equations for each of the variables as

$$\frac{dS}{dt} = -\frac{\beta SI}{N}, \tag{1}$$
$$\frac{dI}{dt} = \frac{\beta SI}{N} - \gamma I, \tag{2}$$
$$\frac{dR}{dt} = \gamma I \tag{3}$$

Using the ODE integrator of your choice (must be callable in Python, we recommend using Scipy as will be covered in lecture on Friday), integrate the equations with $N = 1000$ from $t = 0$ to $t = 200$ for various values of $\gamma$ and $\beta$ (at least 3-4 values, justify your choices physically).
Use the initial condition $I(0) = 1, S(0) = 999, R(0) = 0$ (you can also experiment with other initial conditions if you wish). Plot the curves for $S, I, R$ on the same figure with a legend. make separate plots for each choice of the parameters.
Bonus: Add a 4th parameter $D$ for deaths and justify the addition of a 4th differential equation as well as any RHS terms that must be changed on the initial 3 equations. Integrate the new set of equations for some choice of parameters and comment on the results compared to the SIR model. ...

## Method

For question 2, I began by reviewing that a solution to such a system of ordinary differential equations would be geometrical, and that I should use a numerical integration method that could solve the system and plot its solution. From the SciPy Integrate library, solve ivp requires equations, an interval, initial values, and the number of points in the interval. I started by setting initial values for the model and then I defined a function that would return the right hand sides of the equations. I experimented with different gamma and beta values, and figured out what the variables meant. Beta is the contract rate for the virus, and gamma is the recovery rate. Then, I called solve ivp to create the data sets for the three groups of the population. I then plotted the data against the time axis, with four different sets of beta and gamma.

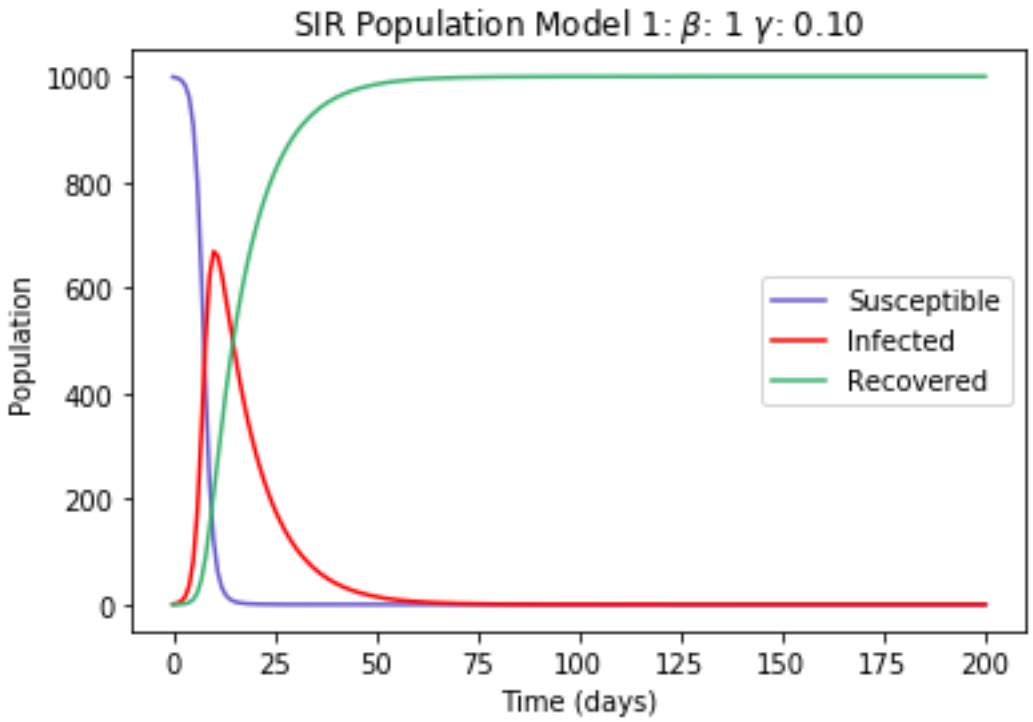


Figure 5: SIR Model Plots

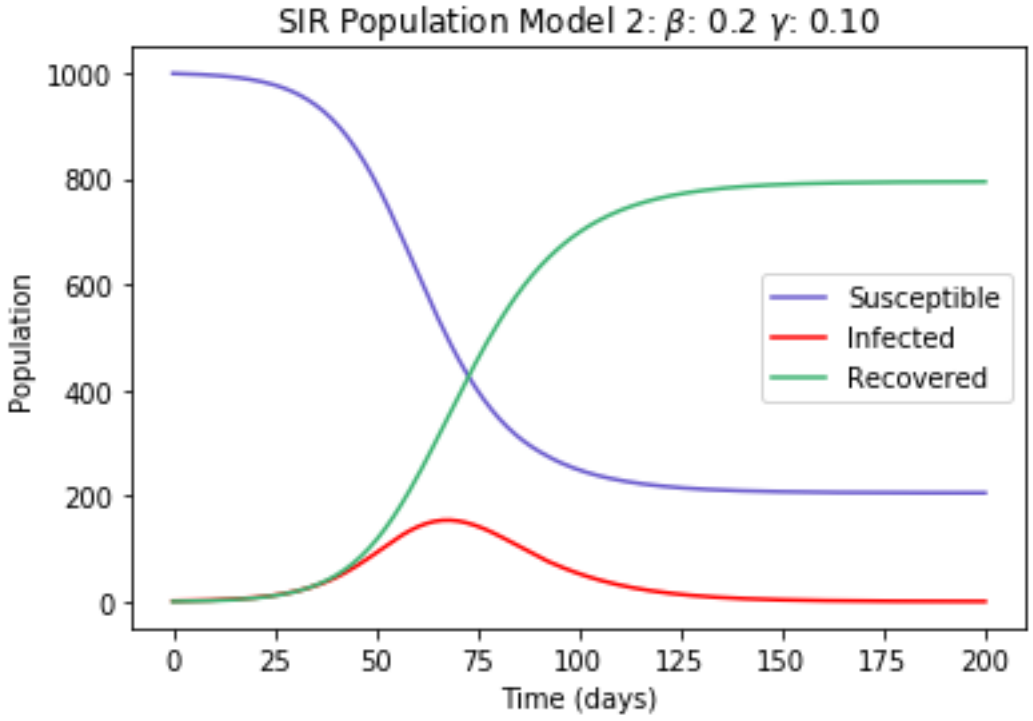


Figure 6: SIR Model Plots

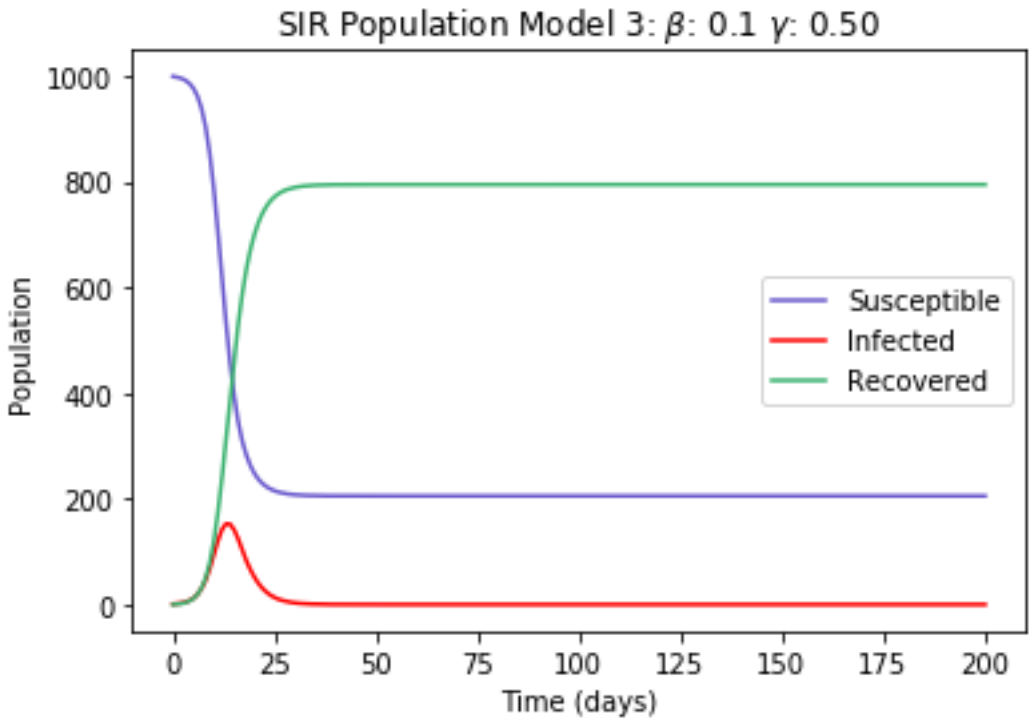


Figure 7: SIR Model Plots

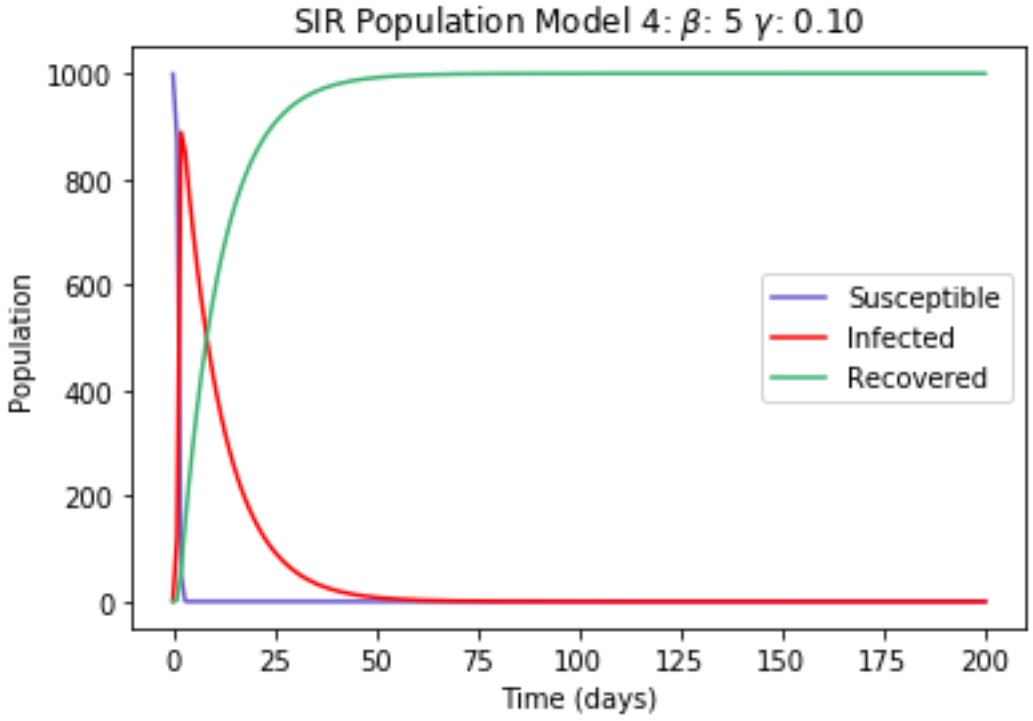


Figure 8: SIR Model Plots

## Analysis

The four plots show interesting results. I picked beta and gamma values to model distinct situations. Plot 1 models a situation where a large portion of the population becomes infected within the first 25 days, but the recovery rate also increases rapidly, flattening the number of people susceptible to the disease close to zero quickly. Plot 2 models a situation where a minority of the population becomes infected after 50 days of the disease spreading, but while the recovery rate is still steep, the number of susceptible people does not drop to zero. Plot 3 models a situation where a minority of the population becomes infected rapidly, but recovers quickly. However, again, the number of susceptible people does not reach zero. Plot 4 models a situation where almost all of the population becomes infected before the first 25 days, but the rapid recovery rate shows that the number of infected and susceptible people drops to zero before 50 days. Each situation models a unique disease, and while the SIR model simplifies pandemic situations, it also offers a unique perspective into how variables such as beta and gamma can drastically change how a disease spreads throughout a population. The SIR model has been used to roughly model the COVID-19 global pandemic in different regions, and while it oversimplifies the situation, it does offer a way to show people how a disease can spread without preventative measures in place.