Project 2

Andrew Rozniakowski

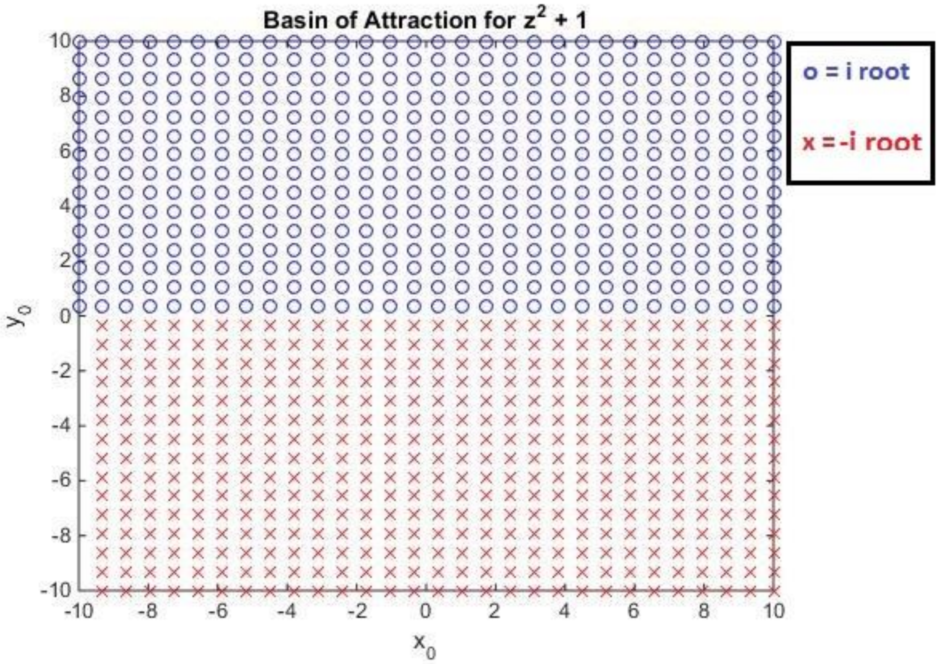
3/22/16

For this project I chose to pick problems 2 and 3. First we will look at problem 2.

2) For this question we had to use Newton's method to determine a basin of attraction for roots of multiple functions in the complex plane. A basin of attraction is a dividing line in which that area converges to a specific root. So we are looking for what initial values will converge to which root of our function. Then, after the basin of attractions are found, we must plot the Mandelbrot set in 2D and 3D. The Mandelbrot set is the set of complex numbers z_0 for which the function $z_{n+1} = z_n^2 + z_0$ is bounded.

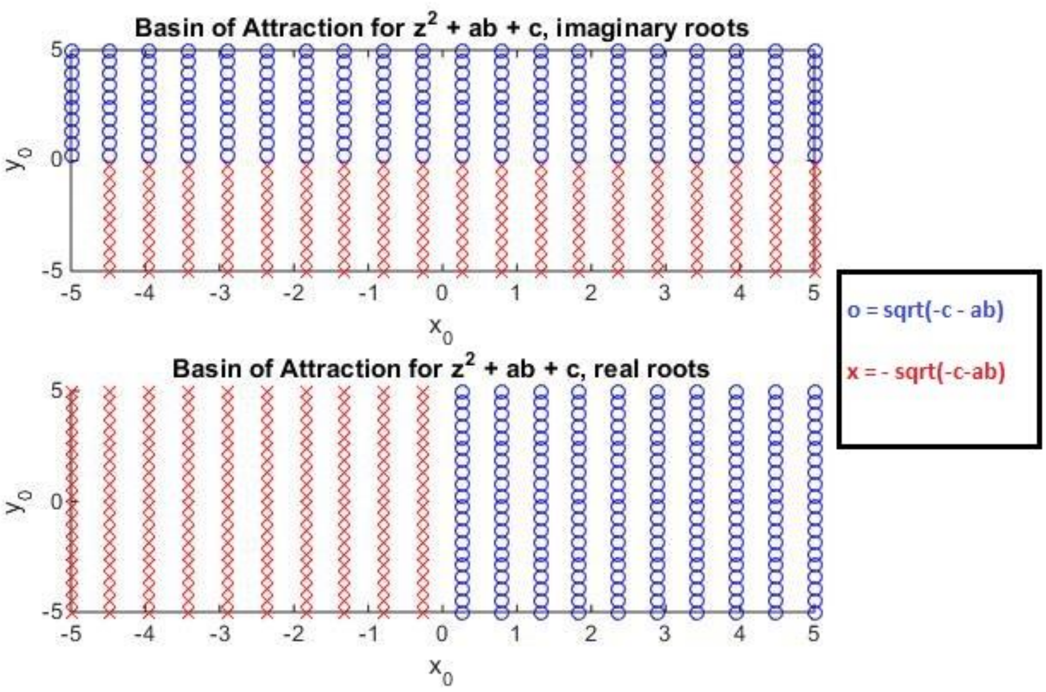
   For the first part of this question we use newton's method to determine the basin of attraction. My Newton's method program takes in an initial value (x_0, y_0) which correlates to an imaginary number x_0+y_0i. The program then plugs in our new imaginary number into the function and its derivative. We then follow newton's method where we divide the function by its derivative and continually do this until the program converges to a root.

   a. For part a of question 2 we have the function $f(z) = z^2 + 1$. To determine the basin of attraction first I figured out the roots for this function, which are obviously $\pm i$. Once the roots are determined I wrote a program to test every combination of (x_0, y_0) between -10 and 10 to see which root they converge to and if a pattern can be found. The graph below was created by my program:



   As we can clearly see x_0 does not affect which root the initial condition will converge to; however, it tells us if $y_0 > 0$ then it will converge to the positive root i and $y_0 < 0$ converges to –i. So our basin of attraction's dividing line is $y_0 = 0$.

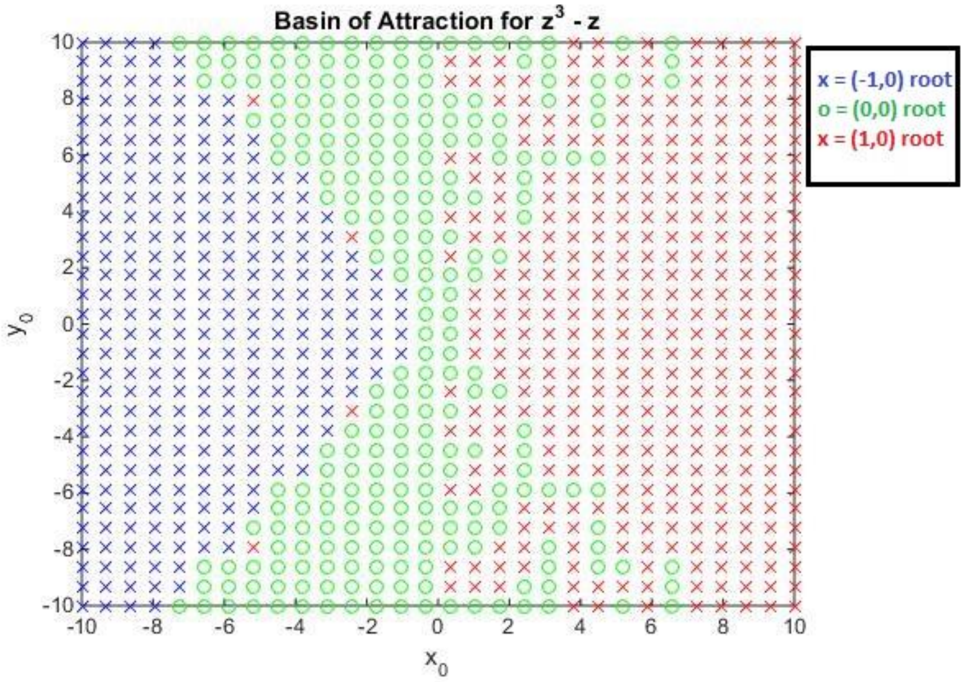
   d. For part d of question 2 I also wrote a program to help me find the basin of attraction for the function $f(z) = z^2 + ab + c$ where a, b and c are real numbers. To determine the basin of attraction first we must determine the roots. The roots for this function are: $\pm\sqrt{-c-ab}$, thus if $c+ab>0$ then we will end up with imaginary roots, while the roots will be real if $c+ab<0$. Therefore, I wrote my program to plot two different graphs, one for when the roots are real and one for when they are imaginary. In my program I chose to only show values of a, b, c between -1 and 1 because changing them will only change the root to $\pm\sqrt{-c-ab}$ but won't affect the basin of attraction. Thus, I felt the a, b and c values I picked show the basin of attraction well enough. My program created the graphs:


Basin of Attraction for z^2 + ab + c, imaginary roots

Basin of Attraction for z^2 + ab + c, real roots

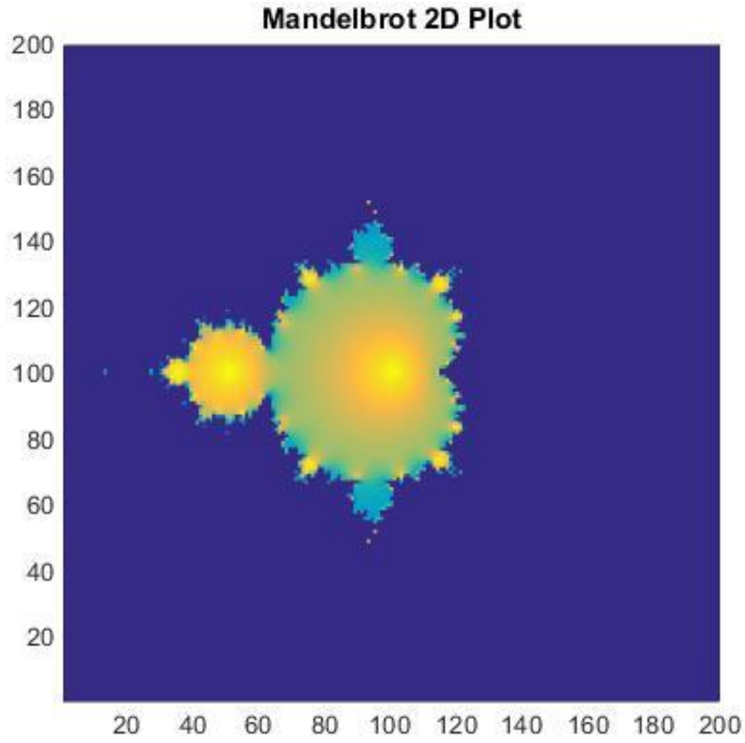
Looking at the first graph for the imaginary roots of our equation we can see the basin of attractions dividing line is at $y_0 = 0$. We also see in the second graph that for the real roots our dividing line is at $x_0 = 0$.

e. For part e of question 2 I once again wrote a program to determine the basin of attraction for the equation f(z) = $z^3 - z$. Before I could write my program I had to find the roots for this equation which are z = 0, ± 1. Similar to my programs for part a and d this program graphs all combinations of (x_0, y_0) from -10 to 10 based on which root they converge. The graph I received is:



As we can see from the graph we get a very interesting pattern. After making other graphs with different boundaries for (x_0, y_0) I found the pattern remains the same. It seems as though the basin of attraction line happens on the line $x_0 = y_0$ and $x_0 = - y_0$ except for -1 to 1, with random points thrown in.

f. I plotted the Mandelbrot set using matlab in my program named Mandelbrot2D.m. To create the Mandelbrot plot I looked at -2 to 2 for both the x and y axis. I used the meshgrid command in matlab to create an X and Y grid. I then created my complex numbers c as X+Yi using my grid, which gives me every combination of complex numbers from -2 to 2 for both x and y. I then substituted my complex numbers c into the quadratic polynomial $z_{n+1} = z_n^2 + c$. This gave me a vector z. I then plotted the vector z on my grid [X,Y] and I was able to achieve the plot:



g. I was unable to plot the 3D Mandelbrot set.

3) For this project I also chose to do question 3. For this question we are given a directed graph where each of the vertices on the graph is a company labeled 1 to 6. An arrow from company i to j means that company i is likely to buy products from company j. For this question we must create a program which takes in an arbitrary stochastic matrix M, which is a matrix where every value is between 1 and 0, an x vector which consists of the money each company has to spend, as well as an integer n for how many iterations we want to run. The program should then output the nth iteration of the fixed point iteration described by x_{k+1} = Mx_k. This is the fixed point iteration for a Markov chain which by definition is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event. Looking at the definition we clearly have a Markov in this question.

   a. For this part we must create a 6x6 matrix in which the (i, j)-th element is a 1 if there is an arrow from j to i, and zero otherwise. Looking at the graph given in the problem we end up with the matrix:

0	0	1	0	1	0
1	0	1	0	0	1
1	1	0	1	1	0
0	0	1	0	1	0
1	0	0	1	0	0

0	1	0	0	0	0

We then divide each column by the sum of that column to get our stochastic matrix:

0	0	1/3	0	1/3	0
1/3	0	1/3	0	0	1
1/3	½	0	½	1/3	0
0	0	1/3	0	1/3	0
1/3	0	0	½	0	0
0	½	0	0	0	0

Using this matrix we interpret the position M(i, j) as the probability that company j is selling products to company i. For example, the position M(1, 3) = 1/3 tells us there is a 33.33% chance company 3 buys products from company 1.

b. In the situation where x =

$50,000
$200,000
$300,000
$225,000
$125,000
$100,000

We see that M*x =

$141,670
$216,670
$270,830
$141,670
$129,170
$100,000

We interpret these numbers of our vector as the amount of money each company is making from selling products to other companies. For example the first spot in the vector = $141,670 which tells us company 1 makes $141,670 dollars through selling products to other companies. Now that every company has made a new amount of money, shown in vector M*x, that company now has that amount of money to spend.

c. For this step I created a program which takes in a stochastic matrix M, an initial vector x_0 and a max iteration n. The program then calculates the fixed point iteration of the Markov chain described by the function $x_{k+1} = Mx_k$. The program stops iterating when the max iterations are reached or convergence is found when the absolute value of $x_k - x_{k-1} < .00001$. By inputting the given M and x_0 given in question 3 with a max iteration of 100, we see that the fixed point iteration does indeed converge. My program took 27 iterations to converge to x_k =

.1224
.2585

.2653
.1224
.1020
.1293

We interpret x_k in terms of this problem first by multiplying by 1 million, just like we divided by 1 million in part c. Thus, x_k tells us the amount of money each company makes and thus has to spend over a long period of time. Since this converges x_k is essentially the average of each company has to spend. Over a long period of time each company has exactly that much money to spend.

d. To test the convergence of large stochastic matrices I created a 10x10 matrix as well as a 20x20 matrix to test using the program I created in part c. Both matrices can be found in my code Project2part3.m labeled as M_d and M_2d respectively, including an initial value labeled x_1 and x_2. Running my program on M_d, the 10x10 matrix, I was able to achieve convergence on its 18th iteration, while M_2d, the 20x20 matrix, also achieved convergence after 18 iterations. Therefore, my program keeps the same convergence rate as my matrices increase size. This leads me to believe given any stochastic matrix M its fixed point iteration will converge.

e. Prove if M is stochastic than there exists as x such that Mx = x.

We start by assuming we have an arbitrary matrix M =

A_1	B_1	C_1	...	N_1
A_2	B_1	C_1	...	N_2
A_3	B_1	C_1	...	N_3
...	...	...		...
A_N	B_N	C_N	...	N_N

with an arbitrary x =

X_1
X_2
X_3
X_4
...
X_N

If we multiply these together we end up with the system of equations:

$A_1 * X_1 + B_1 * X_2 + C_1 * X_3 + ... + N_1 * X_N = K_1$
$A_2 * X_1 + B_2 * X_2 + C_2 * X_3 + ... + N_2 * X_N = K_2$
$A_3 * X_1 + B_3 * X_2 + C_3 * X_3 + ... + N_3 * X_N = K_3$
...
$A_N * X_1 + B_N * X_2 + C_N * X_3 + ... + N_N * X_N = K_{N,}$

where $K_{1....N}$ is a vector of constants. Next we can add our solutions up and factor out our x's and this results in

$X_1 * (A_1 + A_2 + ...+ A_N) + X_2 *(B_1 + B_2 +...+ B_N) + ... + X_N *(N_1 + N_2 +...+ N_N) = K_1 + K_2 + ...+ K_N.$

Since this is a stochastic matrix we know all the columns add to one, so we end up with

X_1* (1) + X_2*(1) + ... + X_N*(1) = K_1 + K_2 + ...+ K_N.

Thus this must mean $X_1 = K_1$ and so on. So we have proved if a given matrix M is stochastic then there exists an x such that Mx = x.

f. Studying up on eigenvalues and eigenvectors I found that our steady state x, our x when Mx = x, is the same as the eigenvector corresponding to the dominant eigenvalue of a matrix, which is the eigenvector corresponding to the greatest eigenvalue.

g. Using my 10x10 and 20x20 matrix I used earlier I compared the speed of my fixed point iteration to the eigenvalue function of matlab. After doing this test it is clear the matlab function was able to run more efficiently. I believe this is the case because finding the eigenvalues is a more efficient process. When finding the eigenvalues it doesn't take a lot of operations to achieve your result. It takes a lot more operations to do the fixed point iterations over and over again and thus it is much less efficient.

h. Monopoly is a perfect game to be represented by Markov chains. There are 40 different spots that you could possibly land on thus we have a 40x40 stochastic which tells us the probability of getting from one spot to another. For example, if we are at the position (5,8) of our stochastic matrix then the value of this position is the probability that we will get from spot 5 to spot 8. This probability is based off the roll of the dice or in this case, 2 die. Continuing our example, to get from spot 5 to 8 the player must roll a 3. With 2 die there are 36 total possible rolls but there are only 2 possible ways to roll a 3. Thus the probability of getting to spot 8 from spot 5 is 1/18. Monopoly is not won by getting to a certain position on the board so there is no average number of rolls to finish but what is interesting is since the probability of landing on a spot is not based off how many times a player lands on that spot the for any fixed starting position the probability of landing on the other 39 spots will never change. The best sided die to use for monopoly is a 20 sided die because when using 2 of these you will be able to reach every position no matter what position you are starting at.