Numerical Mathematics Assignment 1 Peter-Jan Derks

Sine function

The function should be called with two arguments as can be seen in figure 1, x, the number or numbers of which the sine needs to be calculated and 'terms', the number of terms of the Taylor expansion used to calculate the sine. If you want to calculate the sine of multiple numbers using the functions once the function needs to be called using a vector.

```
>> sine([1 2 3],30)

ans =

    0.8415
    0.9093
    0.1411
```

Figure 1

The function calculates the sine using the Taylor series around 0, also known as the Maclaurin series.

The accuracy of a Taylor series can easily be calculated using the remainder of a Taylor series. The formula to find the remainder term for the Taylor series of the sine is as follows:

$$|R_n x| \leq \frac{|x|^{n+1}}{(n+1)!}$$

For x = 2π, if 30 terms are requested the remainder can be calculated using MATLAB:

>> ((2*pi)^32)/(factorial(32))

ans =

1.3231e-10

So the error is smaller or equal than $\approx 1.3231 \cdot 10^{-10}$. We can easily test if this is true by letting my function calculation the sine of 2π (which should be 0)

>> sine(2*π)

ans =

3.3009e-15

This number is much smaller than the maximum error (remainder).

My function is not periodic meaning it does not give the same answer for some values x and x + 2 π, while it should. Having to do with this my function is not accurate for large values. I could have solved this by making a while loop that subtracts 2π off or adds 2π to the input until the value is between -2 π and 2 π.

The Attractor

The function "attractor" should be called with a λmin and λmax. The input variable N which decides the number of values of λ between λmin and λmax for which the attractor is calculated, is optional. Example: >> attractor(0,4,200).

The function produces a picture with $\lambda \in [\lambda min, \lambda max]$ along the horizontal axis and values of xn for "large" n along the vertical axis.

I decided to use three different methods for three different ranges of lambda. For these different ranges of lambda the limiting value can be calculated differently. From lambda 0 to 1 I plotted x = 0, from lambda 1 to 3 I plotted x = lamda – 1/ lambda, and for lambda 3 to 4 I plotted $xn=\lambda xn-1(1-xn-1)$ for a large n, in my case n = 100. To make the function compute the limiting values for lambda 3 to 4 was the most complicated. My function makes a matrix with 100 rows and the 'length of lambda 3 to 4' columns. I set up an array of x's with the value of x at lambda = 3, equal in length to lambda 3 to 4. (The length of lambda 3 to 4 depends on the step size which depends on input variable N). By doing this I could use matrix element multiplication (in MATLAB: .*) to calculate all x's of a year/row without looping through the different values of lambda.

My program has difficulty finding the limiting values for lambda for which the x doesn't converge quickly. An example of this is the point at lambda $\approx 3 + \sqrt{6}$. I could solve this by increasing the amount of 'years' my program runs, meaning the amount of times $xn+1=\lambda xn(1-xn)$ is calculated. Currently I have it compute this 100 times for every lambda bigger than 3 and I decided not to increase this because it would make my function run slower.

To produce figure 2 I used the following values:
x0 = 0.5
n = 200
The input satisfies $0 \leq \lambda min \leq \lambda max \leq 4$.

The function was called as following:
>> attractor(3,4,200)

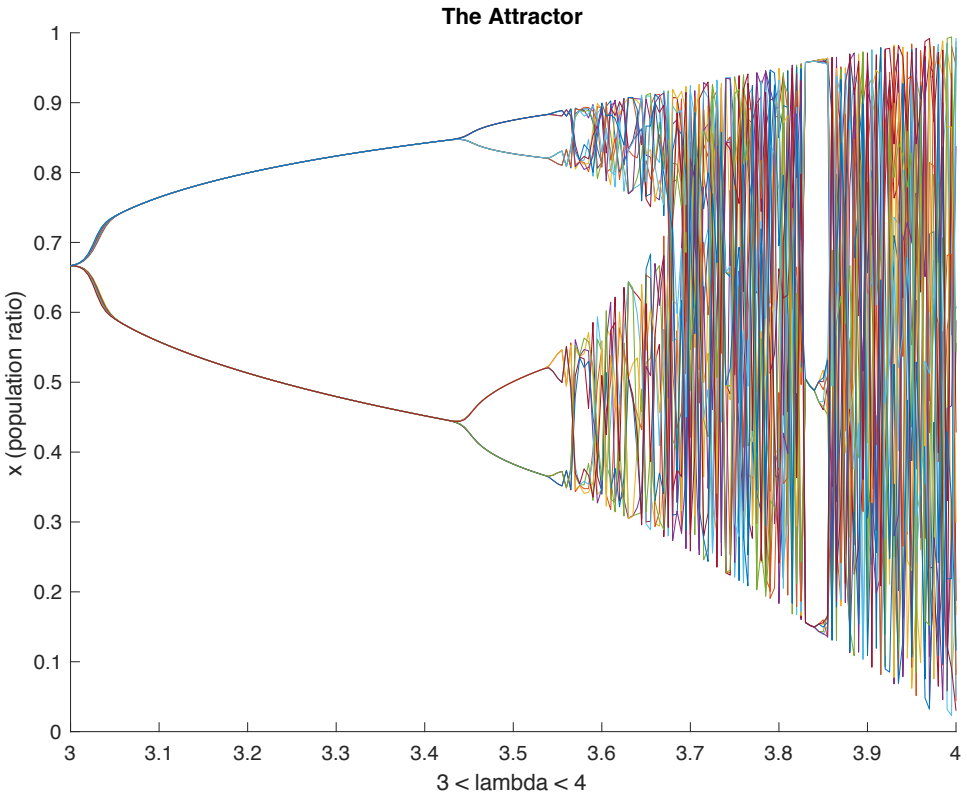
The Attractor
x (population ratio)
3 < lambda < 4
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
3
3.1
3.2
3.3
3.4
3.5
3.6
3.7
3.8
3.9
4

Figure 2