MA 321

Project #1

Project 1

Problem 1

User Guide:
Bis.M : This function is used as the bisection method. The bisection method is used to find the roots of a polynomial equation.
The calling Syntax is bis(a,b,tolb,func) where tolb is the error for the bisection method.
This method returns the found root, a, and, b.
Nwt.M: This function resembles Newton's Method, with the intial guess p0. Newtons method is another numerical method used to find the roots of a polynomial equation.
The calling syntax is nwt(p0,toln,nmax,func,dfunc) where toln is the error for Newton's Method. and dfunc is the derivative of func.
This method returns the root,found, a logicial variable indicating weather the root has been found, and the iteration count.
Bisnwt.m: This function resembles a mix of newtons method and the bisection method. We are using both of these methods combined to find the roots of the polynomial equation. This equation is solely based upon algorithm one in the project description.
The calling syntax is bisnwt(a,b,tolb,toln,nmax,func,dfunc). This includes the calling syntax of both the bisection method and newtons method combined.
This method returns the root, iter, and itern, where iter is the number of iterations used in the while loop. If the while loops fails to achieve an error less than or equal to the toln, it outputs "Method failed to achieve error toln".
Tanom.M: this functions solves the kepler equation and gives the position of the satellite in its orbital plane.
The calling syntax is tanom(T,e,n,varargin). T is the period of the orbit in hours, e is the eccentricity of the orbit, and n is the number of points where we want to localize the satellite on its orbit.
This method returns orbit, which is a marix consiting of [t, E, v, r, x, y]. ti = iT/n, Ei is the solution to Kepler Equation, i is the true anomaly corresponding to ti, ri is the distance between the satellite and the earth at ti and (xi,yi) are the cartesian coordinates.
demo.M: This method calls tanom and displays the orbit in a graph.
The calling syntax

Problem 2

The argument toln in the Matlab function tanom.m represents the error that we allow in the approximation of the Kepler equation. If we want the error in the coordinates of the orbit to be smaller than 106, which value should we use for toln? Justify your answer.
Toln would stay the same because Matlab has precision of up to 16 decimal points. TolN would be equal to it's original value of 1e-12.

Problem 3

– Intuitively, root-finding solvers are meant for x Rand thus, the points in the orbit can be found one by one, applying Algorithm 1 and the transformations described in Section 2.1 for each time of the period. A slight change, though, would allow us to work with a vector x Rn and find all the points of the orbit at once. Discuss about the differences between the two approaches.

It would be a lot easier to debug the first algorithm over the second algorithm. Although algorithm 2 may be faster, it may not be easier to navigate. With algorithm 2 taking multiple errors at once it appears harder to debug than Algorithm 1.

Problem 4

Describe the tests that you have done to validate your code, what you have discovered, the problems encountered, and their respective solutions. You must include at least the following test T = 4, e = 0.25, n = 100, and tolerance 1e-12 in “bisnwt.m”.
Test 1: During the bisection method I checked if the signs were the same in the func(a) and the func(b). If they were the same, it would not meat the requirement for the bisection method. I would have to return.
Test 2: During newtons method, my first if statement was checking if the function value at p0 was equal to zero. If it was that meant I did not need to go through the method as the root would be found to be zero.
Test 3: In the Bisection/Newton method, I had the function display after each iteration of the while loop if the method failed to achieve error ¡= toln. This enabled me to see the iterations in my command window so I would be able to visualize the loop and predict any potential errors in code.
Test 4: When using the test with T=4, e=0.25, n = 100 and tolerance 1e to the -12, One of the main problems I was having was creating the full aspect of the orbit. At first, only about half of the entire orbit was being shown. I knew this was happening in the bottom half of the orbit, around pi to 2 pi. In order to fix this issue I had to add an if statement in tanom.m to add pi to my v value when E was between pi and 2pi.