ME 355 – System Modeling and Numerical Methods
Fall 2022

Project 1

Soft Due Date: Monday, September 26 by 11:59 pm
Hard Due Date: Friday, September 30 by 11:59 pm
Preliminary functions due Friday, September 23 by 11:59 pm
This project must be turned in on Blackboard.

Project 1 is due on the soft due date. However, since your Exam 1 is on Wednesday, September 28, you are allowed to submit up to the hard due date without any late penalties. Late penalties will be based on the hard due date.

You are an engineer at the company We Make Stuff, a contractor service to design and build on demand products. Your company recently won a build for a Renaissance fair to design and build a trebuchet. One of the design criteria is to have the trebuchet hit various targets at different heights and distances from the trebuchet. Though your team are proficient at building the trebuchet, they are having difficulty determining the launch parameters for the projectile.

You were able to develop a mathematical model which calculates the trajectory of the object as

$$y = (\tan\theta_0)\, x - \frac{g}{2v_0^2 \cos^2\theta_0} x^2 + y_0 \quad (1)$$

where the position of the object is defined by (x, y) coordinates with an initial launch angle of θ_0 and initial launch velocity of v_0. The launch height is y_0. You can assume gravity (g) is 9.81 m/s^2. The object is at position $(0, y_0)$ at release.

Your job is to specify the launch angle of the projectile necessary to hit a target x_{target} meters away at a height of y_{target} meters off the ground, based on an initial velocity v_0. After completing this program, you need to prove to the customer that your program works. To do this, you plan to perform verification and validation (V&V) on your program. You will perform V&V two ways.

1) By visualizing the path of the object for a given v_0 and computed θ_0. This plot should show that the trajectory of the object passes through point (x_{target}, y_{target})
2) By calculating the y_{target} value based on the v_0, x_{target}, and computed θ_0.

These two methods must be applied to each of the three conditions in Table 1.

Table 1. The test data used for the verification and validation documentation.

y_0, [m]	v_0, [m/s]	x_{target}, [m]	y_{target}, [m]	θ_0, [degrees]
16	50	250	10	
20	80	660	4	
9	40	30	20	

1 Functions

To complete this task, you will need to develop **3** MATLAB functions. Preliminary versions of these functions are due on the date specified above. To get full credit for the preliminary functions you must have made a legitimate attempt. You may make changes to the functions after the preliminary function due date and prior to the final due date.

Note: All functions will be run through an auto grader, with the code reviewed manually for comments. All inputs and outputs must be **EXACTLY** as listed below.

1. **trajectory.m**
   y = trajectory(x, y0, v0, theta0)
   The *trajectory* function calculates the y-position (y) of the thrown object using equation (1) based on the current x-position (x), initial launch height (y0), release velocity of the object (v0), and release angle **in degrees** of the object (theta0). Assume gravity is 9.81 m/s^2.

2. **falseposition.m**
   root = falseposition(f, xl, xu, tolerance)
   The *falseposition* function performs the false-position method where f is an anonymous function with one independent variable, xl and xu are the lower and upper bounds of the false-position method respectively, and tolerance is the acceptable value such that the false-position algorithm is ran until the relative approximate error is less than the tolerance. Note that **tolerance is a decimal**, NOT a %. It can be assumed xl < xu and tolerance > 0. Be sure check if there is a guaranteed root between xl and xu and have Matlab throw an error if not! Matlab can throw errors using the **error()** function. You can assume that the root never falls on xl or xu during your algorithm. You are not allowed to use any built in Matlab *falseposition* functions, you must code your own!

3. **launchangle.m**
   theta0 = launchangle(x_target, y_target, y0, v0, number_of_significant_figures)
   This function uses the *falseposition* function developed by you to calculate the launch angle (θ_0) **in degrees** necessary to hit the target located at (x_{target}, y_{target}) within a specified number of significant figures (number_of_significant_figures) based on the initial launch height (y0) and release velocity of the object (v0). This function must utilize the falseposition.m function and the trajectory.m function defined prior. You will need to assign your trajectory.m function as an anonymous function in a format that can be called by falseposition.m. For this function, it can be assumed the launch angle will always be between 0 and 45 degrees.

2 Validation and Verification Documentation

All your code for the V&V portion of the project should be written in a Matlab file called **project1.m** and submitted with your project! You are not allowed to use any root finding functions Matlab provides, including fzero. You must use your own! You will additionally be submitting a **V&V documentation PDF**.

For this project, you need to provide V&V documentation to prove to the customer that your code works. Your V&V documentation should include the following sections.

1. Title Page
2. Introduction
   This should be about a paragraph setting up what the purpose of your code even is! You must include a sketch of the problem. Note, there is no need to mention Matlab anywhere in this documentation.
3. Figures and Tables
   All plots are listed in the form of y-axis versus x-axis.
   All figures and tables must have a **descriptive** caption.
   All plots must be created in Matlab!
   a. Recreate Table 1 with the θ_0 values filled in. Calculate these θ_0 values using your launchangle.m function.
   b. Plots (3 in total) of your projectile trajectory and the target location for each of the three test cases listed in Table 1.
4. Calculations
   Calculate the y_{target} value based on the y_0, v_0, x_{target}, and computed θ_0 values for each of the three test cases listed in Table 1.
   For one of the test cases, show your work using Word's equation editor (or equivalent). Refer to the example report for how to use the equation editor. Do NOT do this by hand and paste in an image!
5. Brief explanation (about a paragraph) as to why your figures and calculations (parts 3 and 4 of your V&V documentation) prove that your code works.

3 Deliverables

Be sure to submit all 5 files!

1. trajectory.m
2. falseposition.m
3. launchangle.m
4. project1.m
5. V&V PDF

4 Rubric (Total 100 Points)

Functions 1 – **10 Points**
Functions 2 – **20 Points**
Functions 3 – **10 Points**
Turned in preliminary functions on time **– 10 points**

V&V Documentation – **40 Points**
- Introduction – **5 Points**
- Figures and Tables – **15 Points**
- Calculations – **10 Points**
- Discussion – **10 Points**

V&V Documentation format and Code Comments – **10 Points**

5 Notes for Report

For any **equations**, please use the equation editor.
For **Figures** and **Tables**, please follow the guidelines outlined in the example report on Blackboard.
Include a title page.

6 Notes for Functions

These notes do not apply to the **project1.m** file.
All functions must be named EXACTLY as listed above. This includes capitalization.
Functional inputs and outputs must be EXACTLY as defined in the Functions section.

7 Test Data

If you run the following code below in the Matlab command window, you should get the following output. Do note, this does NOT guarantee your code is correct.

```
a = trajectory(3.4, 10, 33.2, 30)
f = @(x) x^2 - 7;
b = falseposition(f, 0, 10, 0.01)
c = falseposition(f, 2, 10, 10)
d = launchangle(40, 1, 1.8, 20, 3)
e = launchangle(40, 1, 1.8, 20, 1)
f = falseposition(f, 9, 10, 1)
```

```
a = 11.8944
b = 2.6224
c = 2.4082
d = 36.3876
e = 37.6030
f = (This should throw an error)
```