Project 2

Numerical methods

Matlab has many built in functions that allow us to solve nonlinear equations (or systems of equations), ordinary differential equations (or systems of ODEs), and integrals. These built-in functions use advanced algorithms that are based on simple numerical methods.

Newton's method (for solving algebraic equations)

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

where f is the function of interest.

Euler's method (for solving differential equations)

$$y_{n+1} = y_n + hf(x_n, y_n)$$

where f is the differential equation and h is the step size.

Midpoint rule (for integrating an equation)

$$\sum f\left(\frac{x_{i+1} + x_i}{2}\right) \cdot h$$

where f is the function that is being integrated, and h is the step size.

These basic numerical algorithms were discussed during class, and significant documentation can be found online.

For this project, you need to

1) Write a basic function for each one of these methods (see rubric for input requirements)
2) Create a visual-heavy (and text-light) report that addresses the following:
   A. Newton's Method: (be sure the final numerical answer is clearly displayed for each of these, in addition to the other requirements)
      a. Solve

$$\frac{\sin(x)}{(x-1)^2} + 5(\cos(0.5x^2))^2 + 3x = -1.5$$

      with Newton's method and visually display how the algorithm works.
      b. Solve

$$y = 2x\sin(1.5x)$$
$$y = 4(x - 6.7)^5 - 2$$

      with Newton's method. Display algorithm steps (including, but not limited to, iteration number and values of y and x) in a reasonably sized table.

c. Solve

$$z = 3x - 2y - 4$$
$$z = -x + 3y - 8$$
$$z = x - 2$$

with Newton's method. (No table or plot necessary).

d. Solve for at least three roots of

$$y = 2x^4 - 6x^3 - 8x^2 + 24x$$

with different initial guesses, and visually display the how the algorithm works for each of the different initial guesses.

B. Euler's Method: (your final answer for each of these should be a professional looking plot, in addition to the other requirements. Note that you may label your plot and create a legend *outside* of the function itself.)

a. Solve

$$\frac{dy}{dx} = -4\sin\left(\frac{\pi}{6}\ln(y)\right) \cdot (x - 1), \quad y(0) = 2$$

from x=0 to 4 and visually display how the algorithm works.

b. Solve this system of equations:

$$\frac{dy}{dx} = -\sin\big(z \cdot (2x + y)\big)$$
$$\frac{dz}{dx} = e^{-0.05x}\cos(4y)$$
$$y(0) = -1, \quad z(0) = 0$$

from x=0 to 7.

c. Solve

$$\frac{dy}{dx} = y + x^2 - e^{1.5y}, \quad y(0) = 0$$

from x=0 to 10 with different time steps. Determine an optimum step size and support your decision.

d. (For excellence, not minimally acceptable) repeat parts a, b, and c with other Runge Kutta methods. You will need to write other functions for these algorithms.

C. Midpoint Method: (be sure the final numerical answer is clearly displayed for each of these, in addition to the other requirements)

a. Solve

$$\int_0^{2.5} \tan\left(x - \frac{\pi}{3}\right) + 4\cos(x^3)\,dx$$

and visually display how the algorithm works.

b. Solve

$$\int_0^{4\pi} \sin(2x) + \cos(3x)\,dx$$

with different time steps. Determine an optimum step size and support your decision.

c. (For excellence, not minimally acceptable) repeat parts a and b with other numerical integration methods. You will need to write other functions for these algorithms.

You will need to write a script for part 2, but you can and should create the report in another program, like Microsoft Word, LaTeX, Google Docs, etc. You should not be using the publish command in Matlab to create the report. The only things you should do in the text editing program is format tables, resize an entire figure (labels and all) and type text.

You are allowed to write a separate function or section of the script to visually demonstrate how each algorithm works, since the information you want to plot is likely more than what you need for the basic solution. The 3 methods sections in the rubric will be looking at the basic function, not the "visual output" function/section of script. This "visual output" function/section will not be graded extensively for efficiency (use for loops when needed, etc.), but will be considered in the code organization/commenting portion of the rubric.

Some things to keep in mind – a longer report is not (always) better. More information in a single plot is not (always) better. Plotting the same data in different ways can (sometimes) be better. Inset plots can (sometimes) be a good way to zoom in on sections of a plot. To show how an algorithm works, you might want to supplement the (accurate) solution with a less-than-stellar step size/starting guess. You might need to try different initial guesses/step sizes to get a good answer. Text-light does not mean no text; a good rule of thumb is to include captions, and brief statements about the final answers (1-2 sentences). We will be looking at these reports on a computer screen, which means you don't need to worry about the conversion from color to grayscale. For Euler's method, you may create labels and legends in the script and not inside the function itself.

Like Project 1, we would like you to demonstrate anything that's excellent (ex: optional inputs, error messages, etc.) Please put these examples into an "excellence" script that can be run on its own. If you do not provide examples, you will not be eligible for the excellent points.

You may do this project either in Python or in Matlab.

Project logistics:

Due April 25th at 5 PM

This project is worth 15% of your overall grade. (~25% of project grade)

You may work in pairs, but do not have to.

You cannot work with the same partner as you did in project 1.

Please submit your partner report to the Partner Report assignment, and not with your code.

There is a 10% penalty if the project report is turned in between 5 and 8 PM. There is a 25% penalty if the project report is turned in between 8 PM and 11:59 PM. There is a 50% penalty if the project report is turned in between 12 AM and 8 AM on April 25th. The project will not be accepted if it is turned in after that.

The rubric contains all of the details about the project and how to earn excellent score, etc. Meeting the minimum requirements means that you get a passing grade, not an A.

You should submit a zip file that contains, at minimum, a Matlab or Python script that shows how you got the answers in the report, along with a pdf of your report. You may also include extra files if you wrote external functions or have other excellent examples to show me.

Rubric:

Pre-project check in: 4 points	Excellent: 4 points Partner selection is completed on time (you must sign up for a group even if working alone)	Fair: 2 points Partner selection is <48 hours late			Did not meet requirements: Partner selection is >48 hours late
Newton's Method /nonlinear equations (10 points)	Excellent: 10 points All minimally acceptable and also outputs a programmer-generated exit flag. The function accepts *optional* inputs about maximum number of steps and whether or not the user wants details about the solving process displayed in the command window. The algorithm is written efficiently.	Good: 9 points	Fair: 8 points	Minimally acceptable: 7 points Accepts inputs of a function, its derivative (there are exceptions to this – see Dr. Dahlke if you don't want to have this as an input!), the starting guess, and the tolerance (how close the answer must be to 0). Outputs the root. The same function works for **a single or system of equations**. There is an attempt at making the function efficient.	Did not meet requirements:
Euler's Method /differential equations (10 points)	Excellent: 10 points All minimally acceptable, and also outputs a programmer-generated exit flag. The function is terminated if a mathematical error is encountered or if the solution blows up, along with a displayed error message. The function accepts an *optional* input about whether or not the user wants details about the solving process displayed in the command window. The algorithm is written efficiently. At least two other Runge Kutta methods are implemented for part d that meet the same requirements as the Euler's algorithm.	Good: 9 points	Fair: 8 points	Minimally acceptable: 7 points Accepts inputs of a function, step size, initial conditions, and domain of interest. Outputs the independent and dependent variable(s). Creates a figure (does not need to be appropriately labeled *in* the function, but should be labeled elsewhere). The same function works for a single or system of equations. There is an attempt at making the function efficient.	Did not meet requirements:

Midpoint Method /integration (10 points)	Excellent: 10 points All minimally acceptable, and also outputs a programmer-generated exit flag. The function is terminated if a mathematical error is encountered or if the solution blows up, along with a displayed error message. The function accepts an *optional* input about whether or not the user wants details about the solving process displayed in the command window. The algorithm is written efficiency. At least two other algorithms (trapezoid and left/right rectangle) are implemented for part d that meet the same requirements as the midpoint algorithm.	Good: 9 points	Fair: 8 points	Minimally acceptable: 7 points Accepts inputs of a function, step size, and integration limits. Outputs the numerical answer. There is an attempt at making the function efficient.	Did not meet requirements:
Format: 5 points **Note: ask for clarification if you are confused**. I will not be giving points back for misunderstanding the directions.	Excellent: 5 points Up to two scripts are provided (one for excellence demonstration, and another to show how you got report answers), along with a PDF of the report. External functions are also allowed. Everything is submitted in a zip file			3 points: Report is not in PDF form but is provided OR more than two scripts are provided (one for excellence, one to "show your work".) External *functions* are okay. Everything is submitted in a zip file.	Did not meet requirements:
Code readability/ organization: 10 points	Excellent: 10 points Your code is well commented (not overcommented) and each function contains a robust description of the algorithm. It is clearly organized and another programmer can easily see what the code is doing.	Good: 9 points	Fair: 8 points	Minimally acceptable: 7 points There is a genuine attempt at commenting and organizing your code. Any packages required are clearly listed at the top if you are using Python.	Did not meet requirements:

Correctness: 36 points Each answer (numerical for Newton and Midpoint, and plots for Euler's) are worth ~1-2 points. (1 point if correct, 0 points if incorrect) Each visual display of the method (or table) is worth 2 points – 2 for a good/excellent, 1 for fair/minimally acceptable, 0 if visual display is poor, -1 if it is missing. Each figure should have a caption/description associated with it. ~1-2 for a good caption/description, 0 for a poor one, -1 if it is missing. This is a rough breakdown – if your visual displays are fine, but not great, you might get 6/8 for one section. If I would score each individually, you'd get 4/8. If you want a more detailed breakdown, you will very likely receive fewer points.					
Answers: 5 points	Excellent: 5 points Attention is paid to decimals for numerical answers. Answers are clearly indicated in the report. Optimum step size arguments are thorough and accounts for computation time.	Good: 4 points	Fair: 3 points	Minimally acceptable: 2 points Answers are all located in the report.	Did not meet requirements:
Report organization: 10 points	Excellent: 10 points Pages are filled appropriately - not too many plots and not too much white space. Styling is consistent in the full report. Different sections are utilized for the different methods.	Good: 9 points	Fair: 8 points	Minimally acceptable: 5 points The report is in order. Margins can be down to 0.5 inches.	Did not meet requirements:
Report figures: 5 points	Excellent: 5 points Figures are appropriately colored. Legends do not overlap any of the plot. All axes are appropriately scaled. All figures and accompanying labels are appropriately sized.	Good: 14-13 points	Fair: 13-12 points	1 point All figures are appropriately labeled (x and y axis labels, legends where appropriate).	Did not meet requirements:

Newton	Answer	Visual display	Caption/Description (4 points)	
A	1	2		
B	1	2 (table)		
C	2.5			
D	2.5	2		
Eulers	Answer (plot)	Visual display (method)	Caption/Description (3 points)	
A	1	2		
B	2	0		
C	2		1 (optimum step size discussion)	
D				
Midpoint	Answer	Visual display (method)	Caption/Description (2 points)	
A	1	2		
B	2	0	1 (optimum step size discussion)	
C				
Total: 36 points				