Middle East Technical University
Department of Mechanical Engineering
ME 310 Numerical Methods
Spring 2024 (Dr. C. Sert)
Study Set 2

For Homework 2 submit the answers of questions 2, 3, 6 and 7. Their grade percentages may not be equal.

Assigned: 19/03/2024 – Due: 01/04/2024 10:30 (online) and in class

Homework Rules and Suggestions:

- This assignment can be done individually or as a team of two students. Everything in your report should be the result of your own work or your team's work. You are allowed to discuss the questions with your classmates and teaching staff up to a certain detail on ODTUClass. You are not allowed to use an AI tool such as ChatGPT in writing codes or other parts of your report.
- Put the following honor pledge at the top of your homework report and behave accordingly.
  "I understand that this is an individual/team assignment. I affirm that I have not given or received any unauthorized help on this assignment, and that this work is my own/team's."
- If you do the homework as a team, put the percent contribution of each member at the beginning of your report.
- If you've exchanged ideas with other students outside ODTUClass, you need to put their names and the extent of your discussion at the beginning of your report.
- You need to submit a printed report. It is what we will be grading. You also need to upload the same report as a PDF document (not a Word document) together with all other files (such as codes) to ODTUClass. Name your MATLAB files properly. Follow MATLAB file naming rules such as "File names cannot start with a number", "They cannot contain special characters or spaces", etc.
- Late submission is not allowed unless you have a valid excuse. In such a case, you need let the whole teaching team know about it before the submission deadline, unless it is an emergency.
- Make sure that the codes in your report are formatted properly. Use a small sized, fixed width font and make sure that lines are not wrapped. If your code is very long, you can shorten it by getting rid of its noncritical parts and putting a note about this. Note that we grade what we see in your printed report. Do not expect us to run your codes for you to generate results, figures, etc. You should do that yourself and put all the results in your report.
- In writing your codes, follow good programming practices such as "use explanatory header lines", "explain inputs and outputs of functions", "use self-explanatory variable names", "use comments", "use empty lines and spaces for readability", "use indentation for code blocks", "divide long lines into multiple lines using MATLAB's '…' syntax", etc.
- Pay attention to the format of your report. It should look like a serious academic work, not like a high school student work. Font types and sizes, page margins, empty spaces on pages, equations, figures, tables, captions, colors, etc. are all important to give the desired "academic work feeling". Language used is also important. Reports with poor use of English will be penalized.
- Do not provide an unnecessarily long report. The shorter your report, the better it is, as long as it answers the questions properly. Avoid wasting paper. Print on both sides of sheets. Avoid using color unless it is really necessary. Format your report properly with small but readable fonts, small margins, no unnecessarily large figures, no useless spaces, etc. to reduce the number of sheets. Do not use a cover page.
- There are more than 100 students, and we can spend only about 10 minutes to grade each report. Your report should be easy to read and understand. We should be able to find the results and judge their correctness easily. We should not get lost in your report. The more we struggle to understand your report, the lower your grade will be. Use figures and tables cleverly for this purpose.
- Reports with only figures, tables and codes, but no text, comments or discussions will not get a good grade. Start answering each question with one paragraph of introduction. Even when a question does not specifically ask for a discussion or a comment, you need to write a few sentences on the key points and your key findings/learnings.
- Figures and tables should be numbered and should have captions (at the bottom for figures and at the top for tables). Their titles should be self-explanatory, i.e., we should be able understand everything about the table or figure just by reading its title. They should all be referred properly in the written text (such as "… as shown in Fig. 3" or "… (See Table 2)").
- Do not use any Appendices in your report.
- Do not forget to put a numbered reference list at the end of your report if you use references. In that case, you need to refer to the references in the text.
- If you are inexperienced in programming, converting an idea/algorithm into a code and writing it in a bug-free way can be time consuming and frustrating. This is not something that can be done at the last minute. You are advised to start working on the assignments as soon as they are assigned.

Reading Assignments:

Self-learning is an important skill. Not everything can be discussed in lectures. You need to learn certain things by yourself.

R1) Read section 5.3.2 Modified False Position (page 141 of 8th edition) to learn about the "trick" that can be used to speed up convergence when the interval gets smaller only from one side due to the high non-linearity of the function.

R2) Read Box 6.1 Convergence of Fixed-Point Iteration (page 151 of 8th edition) and Box 6.2 Derivation and Error Analysis of the Newton-Raphson Method (page 154 of 8th edition) to see the convergence rate proofs of these methods.

R3) Read section 6.5 Multiple Roots (pages 167-168 of 8th edition) to learn the approaches used in determining multiple roots.

R4) Read section 8.4 Pipe Friction (pages 214-217 of 8th edition) which is a Mechanical Engineering case study about friction factor calculation for flows inside pipes. It is related to Q7 of this study set.

R5) Read the Epilogue section of Part 2 (pages 231-234 of 8th edition). Part 2 includes 4 chapters and its epilogue is at the end of Chapter 8. What we call "Chapter 2 Root Finding" in our lectures is this 2nd part of the textbook. We completely skipped the 7th chapter of the textbook, which is about finding the roots of polynomials. You can also skip it. You are NOT responsible for it.

Questions:

Q1. a) Discuss the advantages and disadvantages of the following methods; a) bisection, b) false-position, c) fixed-point iteration, d) Newton-Raphson, e) Secant, f) modified secant, g) quadratic interpolation, h) inverse quadratic interpolation.

b) If an iterative method roughly squares the error every two iterations, what is its convergence rate?

c) Both the false-position method and the secant method perform linear interpolation between function values at two previously calculated root estimates. Are they the same method? What is their main difference?

d) What is wrong in using relative error for convergence check when the root we are looking for is at zero or very close to zero? What should we do in such a case?

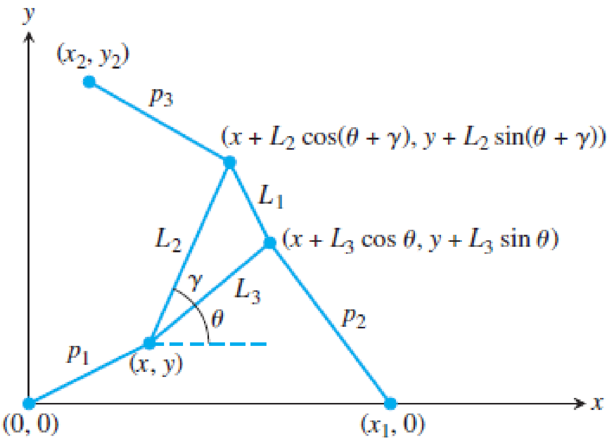
e) Why do we use logarithmic vertical axis for convergence plots that show Error vs. iterations?

Q2. (Reference: Sauer's book) A Stewart platform consists of six variable length struts (or prismatic joints), supporting a payload. Prismatic joints operate by changing the length of the strut, usually pneumatically or hydraulically. As a six-degree-of-freedom robot, the Stewart platform can be placed at any point and inclination in 3D space that is within its reach. Have a look at the following videos and similar ones to see one in operation.

https://www.youtube.com/watch?v=xiECumcaEx0

https://www.youtube.com/watch?v=uxYMh2VpDSc

To simplify matters, here we consider a 2D version of this mechanism, as shown on the right. This is a manipulator composed of a triangular platform controlled by three struts. The inner triangle represents the planar Stewart platform whose dimensions are defined by the three lengths L_1, L_2, and L_3. γ is the angle across from side L_1. The position of the platform is controlled by p_1, p_2, and p_3, the variable lengths of the struts. Finding the position of the platform, i.e. finding (x, y) and θ for any given set of inputs is a forward (or direct) kinematics problem. For motion planning, it is important to solve this problem as fast as possible, often in real time. Unfortunately, no analytical solution is known.



It is possible to reduce the problem to a single non-linear equation and solve it. Simple trigonometry applied to the above figure gives

$$\begin{aligned} p_1^2 &= x^2 + y^2 \\ p_2^2 &= (x + A_2)^2 + (y + B_2)^2 \\ p_3^2 &= (x + A_3)^2 + (y + B_3)^2. \end{aligned} \tag{1}$$

where

$$\begin{aligned} A_2 &= L_3 \cos\theta - x_1 \\ B_2 &= L_3 \sin\theta \\ A_3 &= L_2 \cos(\theta + \gamma) - x_2 = L_2[\cos\theta \cos\gamma - \sin\theta \sin\gamma] - x_2 \\ B_3 &= L_2 \sin(\theta + \gamma) - y_2 = L_2[\cos\theta \sin\gamma + \sin\theta \cos\gamma] - y_2. \end{aligned} \tag{2}$$

Eqn. (1) solves the inverse kinematics problem, which is to find p_1, p_2, p_3, given x, y, θ. Your goal is to solve the forward problem, namely, to find x, y, θ, given p_1, p_2, p_3. Multiplying out the last two lines of Eqn. (1) and using the first yields

$$\begin{aligned} p_2^2 &= x^2 + y^2 + 2A_2x + 2B_2y + A_2^2 + B_2^2 = p_1^2 + 2A_2x + 2B_2y + A_2^2 + B_2^2 \\ p_3^2 &= x^2 + y^2 + 2A_3x + 2B_3y + A_3^2 + B_3^2 = p_1^2 + 2A_3x + 2B_3y + A_3^2 + B_3^2, \end{aligned} \tag{3}$$

which can be solved for x and y as

$$\begin{aligned} x &= \frac{N_1}{D} = \frac{B_3(p_2^2 - p_1^2 - A_2^2 - B_2^2) - B_2(p_3^2 - p_1^2 - A_3^2 - B_3^2)}{2(A_2B_3 - B_2A_3)} \\ y &= \frac{N_2}{D} = \frac{-A_3(p_2^2 - p_1^2 - A_2^2 - B_2^2) + A_2(p_3^2 - p_1^2 - A_3^2 - B_3^2)}{2(A_2B_3 - B_2A_3)}, \end{aligned} \tag{4}$$

as long as $D = 2(A_2B_3 - B_2A_3) \neq 0$. Substituting these expressions for x and y into the first line of Eqn. (1), and multiplying by D^2, yields

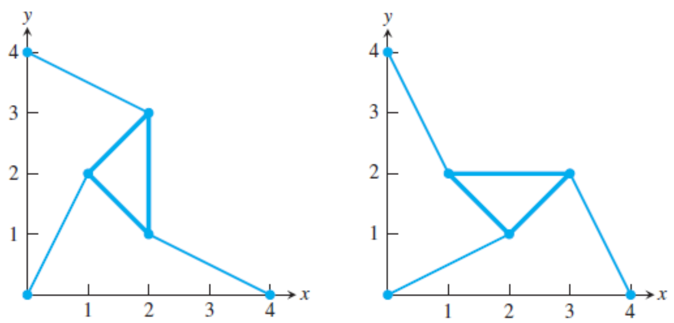
$$f = N_1^2 + N_2^2 - p_1^2 D^2 = 0 \tag{5}$$

in the single unknown θ (Recall that $p_1, p_2, p_3, L_1, L_2, L_3, \gamma, x_1, x_2, y_2$ are known). If the roots of $f(\theta)$ can be found, the corresponding x and y values follow immediately from Eqn. (4). Note that $f(\theta)$ is a polynomial in $\sin(\theta)$ and $\cos(\theta)$, so, given any root θ, there are other roots $\theta + 2\pi k$ that are equivalent for the platform. For that reason, we can restrict our attention to θ in $[-\pi, \pi]$. It can be shown that $f(\theta)$ has at most six roots in that interval.

a) Write a MATLAB code that will calculate all the roots of Eqn. (5) for a given set of input parameters and plot all the possible configurations of the platform in a single figure window. Do not use MATLAB's built-in root function methods, but use your own root finding code. You can use MATLAB's built-in functions for testing purposes, but that is not we will be grading. Make sure that your code is efficient and provides an answer as fast as possible. Speed is important. Hint: Plotting $f(\theta)$ for a number of different input parameter sets will help you understand what kind of a root finding problem is being solved.

Explain in detail the logic used in your code.

b) Run your code with the inputs $L_1 = 2, L_2 = L_3 = \sqrt{2}$, $\gamma = \pi/2$, $p_1 = p_2 = p_3 = \sqrt{5}, x_1 = 4, x_2 = 0, y_2 = 4$. Provide all the calculated θ values and the generated figure, similar to the one shown on the right.



c) Run your code again with the inputs $L_1 = L_3 = 3, L_2 = 3\sqrt{2}$, $\gamma = \pi/4$, $p_1 = p_2 = 5$, $p_3 = 3, x_1 = 5, x_2 = 0, y_2 = 6$. Provide all the calculated θ values and the generated figure.

d) Solve part (c) again by changing $p_2 = 7$. Provide all the calculated θ values and the generated figure.

Q3. The polynomial $x^3 - 2x^2 + 4/3x - 8/27$ has a triple root at $x = 2/3$, which is difficult to determine numerically with high accuracy. This is because the problem is ill-conditioned.

a) Use MATLAB's `fzero` function to find the root by using an initial interval of $[0,2]$. Provide the calculated answer and the iteration details (See Handout 2). What is the true error of the result? How many decimal places were calculated correctly? Try to lower the error by changing the default tolerance values? Is the performance of `fzero` in solving this problem expected? Discuss.

b) Use the uploaded Bracket.m code to solve the problem using the Bisection method. Start with the interval $[0,2]$. Use a very small tolerance value to achieve very high accuracy. Provide the output that shows the iteration number, calculated root, the absolute value of $f(x)$ at the calculated root and the absolute value of the true error. Is the result accurate? Is it expected? Discuss. If you want, you can modify the uploaded code in any way you want. Hint: Do not forget to change the exact value of the root.

c) Write a Newton-Raphson code and repeat part (b). Start with an initial guess of 2.

d) For a root finding problem given as $f(x) = 0$, it is possible to define a forward error and a backward error. If x_r is the exact value of the root and x_a is an approximation of it, the forward error is $|x_r - x_a|$ and the backward error is $|f(x_a)|$. For an ill-conditioned root finding problem, for which an equation with multiple roots is an example, the backward error is often much smaller than the forward one. Is this the case here? Calculate the forward and backward errors for the final iteration values given in parts (a), (b) and (c).

Q4. An iterative root finding method is said to converge with rate r if

$$\lim_{i\to\infty} \frac{E_{t_{i+1}}}{\left(E_{t_i}\right)^r} = C \quad , \qquad i\text{: iteration number}$$

for some finite constant $C > 0$. If $r = 1$ and $C < 1$, the convergence rate is linear. If $r = 2$, the convergence rate is quadratic. If $1 < r < 2$, the convergence rate is superlinear. Determine the convergence rates of the following solutions, where true errors of successive iterations are given.

a) $10^{-1}, 5 \times 10^{-2}, 0.5 \times 10^{-2}, 0.125 \times 10^{-2}, \ldots$

b) $10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, \ldots$

c) $10^{-1}, 2 \times 10^{-2}, 4 \times 10^{-4}, 8 \times 10^{-8}, \ldots$

d) $10^{-1}, 1.26 \times 10^{-2}, 4.54 \times 10^{-4}, 2.24 \times 10^{-6}, \ldots$

Q5. The equation $f(x) = x^2 - 3x + 2 = 0$ has a root at $x = 2$. Each of the following functions yield an equivalent fixed-point problem.

$$g_1(x) = \frac{x^2 + 2}{3}, \qquad g_2(x) = \sqrt{3x - 2}, \qquad g_3(x) = 3 - \frac{2}{x}, \qquad g_4(x) = (x^2 - 2)/(2x - 3)$$

a) Analyze the convergence properties of the corresponding fixed-point iteration schemes by evaluating $|g'(2)|$. Which ones will converge and which ones will diverge? Which of the converging ones will converge the fastest?

b) Confirm your analysis by performing 5 iterations and finding the roots numerically. Show the details of your calculations. Start with an initial guess of 3.

c) Plot the given $g(x)$ functions using a computer and show the iteration paths by hand drawings.

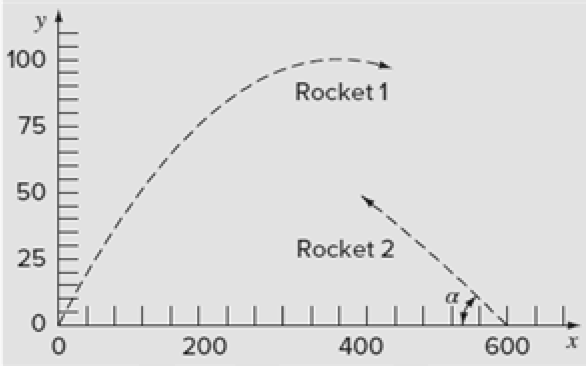
d) It is possible to show that the Newton-Raphson method is nothing but the fixed-point iteration method with a specially selected $g(x)$ such that $g'(x_{\text{root}}) = 0$. Does any of the given $g(x)$ functions satisfy this condition? If yes, show that solving the question with the Newton-Raphson method is the same as solving it using fixed-point iteration with that special $g(x)$.

Q6. This question is a great oversimplification of a missile defense application. The paths of two rockets in the x-y plane are described by the following parametric equations.

Rocket 1: $x_1 = 100t\,, \quad y_1 = 80t - 22t^2$

Rocket 2: $x_2 = 700 - 140\cos(\alpha)\,t\,, \quad y_2 = 120\sin(\alpha)\,t - 1.3t^2$

where $t =$ time and $\alpha =$ launch angle of the second rocket. The two paths may intersect but the rockets will only hit if they arrive at the same point at the same time. The task is to determine the value of t and α so that a hit occurs. The intercepting rocket need to be guided in a clever missile defense system, which is not the case here.



a) Solve the problem graphically by plotting the two functions that need to be solved together and locating their intersection points. Note that they are not the curves seen on the above figure. You need to plot functions obtained by equating x_1 to x_2, and y_1 to y_2. This plot should be on the αt plane, not xy plane. Use α in radians.

b) Perform hand calculations on paper for 2 Newton-Raphson iterations starting from the initial guess of $\alpha = 0.25, t = 5$. Show all the details of the solution.

c) Write a MATLAB code to complete the solution of part (b). Show the results of each iteration. Is this a converging solution? If not, change the initial guess to get a converging solution.

Q7. (You may want to read the reading assignment R4 first) A crucial quantity in pipeline design is the pressure drop due to friction, which is described by the friction factor f, a unitless quantity that can be determined using the empirical Colebrook equation

$$\frac{1}{\sqrt{f}} = -2\log_{10}\left(\frac{\varepsilon/D}{3.7} + \frac{2.51}{\mathrm{Re}\sqrt{f}}\right)$$

where D is the inside pipe diameter, ε is the roughness height of the pipe interior, and Re is the Reynolds number of the flow.

a) Write a code that performs root finding by using the modified secant method to calculate the friction factor for given ε/D and Re values. Use your code to calculate the friction factor for $\varepsilon/D = 5 \times 10^{-4}$ and $\mathrm{Re} = 10^5$. Friction factor values should be positive and typically lie in the range [0.01, 0.1], which may give you an idea about the initial guess you can use. You can also plot the function for which you are trying to find the root to understand the problem at hand better. Provide the output of your code, showing iteration details.

b) Fix ε/D as in part (a), and modify your code to calculate the friction factor for several Reynolds numbers between 10^4 and 10^8. Make a plot of f versus Re using logarithmic axes (use MATLAB's `loglog` command).

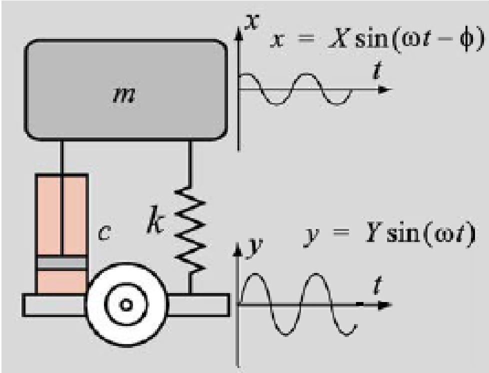
c) Moody diagram that you've studied in ME 306 (an online version can be seen here) is a graphical version of the Colebrook equation. It has several curves similar to the one you generated in part (b). Each curve corresponds to a different ε/D value, which are listed on the right of the plot. Horizontal axis is the Reynolds number and vertical axis on the left is the friction factor. Modify your code so that it performs root finding solutions for various ε/D values (such as 0.05, 0.025, 0.01, 1e-3, 1e-4, 1e-5, 1e-6) and generates a figure with several curves in it, just like the Moody diagram. Hint: In MATLAB, the function `log10` is used to calculate a logarithm in base 10.

Q8. The van der Waals equation of state,

$$\left(p + \frac{a}{v^2}\right)(v - b) = RT$$

relates the pressure p, specific volume v, and temperature T of a gas, where R is a universal constant and a and b are constants that depend on the particular gas. In appropriate units, $R = 0.082054$, and for carbon dioxide, $a = 3.592$ and $b = 0.04267$. Compute the specific volume v for a temperature of 300 K and for pressures of 1 atm, 10 atm, and 100 atm. Compare your results to those for the ideal gas law, $pv = RT$. The latter can be used as a starting guess for an iterative method to solve the van der Waals equation.

Q9. A simplified model of the suspension of a car consists of a mass m, a spring with stiffness k and a damper with damping coefficient c. A bumpy road is modeled by a sinusoidal up-and-down motion of the wheel $y = Y\sin(\omega t)$. From the solution of the equation of motion for this model, the up-and-down motion of the car (mass m) is given by $x = X\sin(\omega t - \phi)$. The ratio between the amplitude X and the amplitude Y is given by



$$\frac{X}{Y} = \sqrt{\frac{\omega^2 c + k^2}{(k - m\omega^2) + (\omega c)^2}}$$

For $m = 2500$ kg, $k = 3 \times 10^5$ N/m and $c = 3.6 \times 10^4$ Ns/m, find the frequency ω for which $X/Y = 0.4$.

Q10. A silicon chip measuring 5 × 5 × 1 mm is embedded in a substrate. At steady state, the chip releases 0.03 W of waste heat. Although the bottom and sides are insulated, the top surface is exposed to air flow and subject to both radiation and convective heat transfer. The radiation heat flux, J_{rad} (W/m²), can be determined via the Stefan-Boltzmann law, $J_{rad} = \varepsilon\sigma\left(T_{a,\infty}^4 - T_{a,s}^4\right)$, and the convective heat flux, J_{conv} (W/m²), by $J_{conv} = h\left(T_{a,\infty} - T_{a,s}\right)$, where $\varepsilon =$ emissivity, $\sigma =$ Stefan Boltzmann constant [= 5.67 × 10^{-8} W/(m² K⁴)], $T_{a,\infty}$ = ambient temperature (K), $T_{a,s}$ = the chip's surface temperature (K), and h = convective heat transfer coefficient (W/(m² K)). Use a steady-state heat balance (equate the rate of waste heat released from chip's top surface (0.03 W / 25 mm²) to the rates of radiation plus convection heat transfer) to compute the chip's surface temperature using the following parameter values: $\varepsilon = 0.9$, $T_{a,\infty} = 300$ K, and $h = 60$ (W/(m² K)).