MAE 3210 - Spring 2020 - Project 1

Project 1 is due **online** through Canvas by 11:59PM on Friday, February 28.

IMPORTANT REMARKS (Please read carefully):

- **Projects are to be treated as take-home exams with NO collaboration or discussion with other students**. Plagiarism will be monitored and considered as cheating. However, you are permitted to reach out to the course instructor (Geordie Richards), lab TA (Aditya Parik), and course grader (Jacob Bryan) for advice with projects (see syllabus for contact information). **In addition to your project PDF, you are required to sign and submit an honor pledge which can be found adjacent to the project handout, under assignments in Canvas.** Each project is worth 50 points, and a 10 point loss per day will apply to late projects.

- You are required to submit code for all functions and/or subroutines built to solve these problems, which is designed to be easy to read and understand, in your chosen programming language, **and which you have written yourself**. The text from your code should both be copied into a single PDF file submitted on canvas. **Your submitted PDF must also include responses to any assigned questions, which for problems requiring programming should be based on output from your code**. For example, if you are asked to find a numerical answer to a problem, the number itself should be included in your submission.

- In addition to what is required for homework submission, for each numerical answer you reach based on running code you have written yourself, **your submitted project PDF must include a copy of a screenshot** that shows your computer window after your code has been executed, with the numerical answer displayed clearly on the screen.

- Note that, in problem 1 below, you are asked to use code that you have already written for homework 3. **You need to submit any code that you use for this project, including any code that you already wrote and submitted for homework 3.**

- If you are asked to use code that you have written yourself to solve a given problem (e.g. in 1(c),(d) below), but you are unable to get that code working, you may choose, instead, to submit numerical answers based on running built-in functions (e.g. algebraic solvers already available in MATLAB), and you will receive partial credit. **However, you are required to write all code**

yourself, without relying on built-in functions, in order to get full points.

1. Linear algebraic equations can arise in the solution of differential equations. For example, the following *heat equation* describes the equilibrium temperature $T = T(x)$ (°C) at a point x (in meters m) along a long thin rod,

$$\frac{d^2T}{dx^2} = h'(T - T_a), \tag{1}$$

where $T_a = 10\,°\text{C}$ denotes the temperature of the surrounding air, and $h' = 0.03\,(\text{m}^{-2})$ is a heat transfer coefficient. Assume that the rod is 10 meters long (i.e. $0 \le x \le 10$) and has boundary conditions imposed at its ends given by $T(0) = 20\,°\text{C}$ and $T(10) = 100\,°\text{C}$.

(a) Using standard ODE methods, which you do not need to repeat here, the general form of an analytic solution to (1) can be derived as

$$T(x) = A + Be^{\lambda x} + Ce^{-\lambda x}, \tag{2}$$

where A, B, C, and λ are constants. Plug the solution of type (2) into both sides of equation (1). This should give you an equation that must be satisfied for all values of x, for $0 \le x \le 10$, for some fixed constants A, B, C, and λ. Analyze this conclusion to determine what the values of A and λ must be.

(b) Next, impose the boundary conditions $T(0) = 20\,°\text{C}$ and $T(10) = 100\,°\text{C}$ to derive a system of 2 linear algebraic equations for B and C. Provide the system of two equations you have derived.

(c) Use one of the numerical algorithms you developed for homework 3 (Gauss elimination *or* LU decomposition) to solve the algebraic system you derived in question 2(b) above, and obtain an *analytic solution* to (1) of the form (2). By *analytic solution* we mean an explicit solution to equation (1) which is valid for each x in the interval $[0, 10]$.

(d) Next we will discuss how to obtain a *numerical solution* to (1). That is, we will seek to obtain an approximate solution to (1) which describes the value of T at 9 intermediate points inside the interval $[0, 10]$. More precisely, the equation (1) can be transformed into a linear algebraic system for the temperature at 9 interior points $T_1 = T(1)$, $T_2 = T(2)$, $T_3 = T(3)$, $T_4 = T(4)$, $T_5 = T(5)$, $T_6 = T(6)$, $T_7 = T(7)$, $T_8 = T(8)$, $T_9 = T(9)$, by using the following finite difference approximation for the second derivative at the i^{th} interior point,

$$\frac{d^2T_i}{dx^2} = \frac{T_{i+1} - 2T_i + T_{i-1}}{(\Delta x)^2}, \tag{3}$$

where $1 \leq i \leq 9$, $T_0 = T(0) = 20\,^\circ\text{C}$, $T_10 = T(10) = 100\,^\circ\text{C}$, and Δx is the equal spacing between consecutive interior points (i.e. with 9 equally spaced interior points inside $[0, 10]$ it holds that $\Delta x = 1$). Use (3) to rewrite (1) as a system of 9 linear algebraic equations for the unknowns $T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_8$, and T_9. Provide the system of 9 equations you have derived.

(e) Use one of the numerical algorithms you developed for homework 3 (Gauss elimination *or* LU decomposition) to solve the system derived in question 1(d) above. Validate your numerical solution by comparison to the analytic solution that you obtained in 1(c) through depicting the two solutions on plots over the interval $0 \leq x \leq 10$.

(f) Write a function that takes as input the number of interior nodes n desired for your numerical solution (i.e. $n = 9$ in 1(d) above), and outputs the numerical solution to (1) in the form of the interior node values $T_1 = T(\Delta x)$, $T_2 = T(2\Delta x), \ldots, T_n = T(n\Delta x)$.

(g) Produce and submit 4 plots that compare your analytic solution to (1) derived in question 2(b) to the numerical solution generated in question 2(f) for $n = 1$, $n = 4$, $n = 9$, and $n = 19$, respectively.

2. Develop an algorithm that uses the golden section search to locate the minimum of a given function. Rather than using the iterative stopping criteria we have previously implemented, design the algorithm to begin by determining the number of iterations n required to achieve a desired absolute error $|E_a|$ (not a percentage), where the value for $|E_a|$ is input by the user. You may gain insight by comparing this approach to a discussion regarding the bisection method on page 132 of the textbook. Test your algorithm by applying it to find the minimum of $f(x) = 2x + \frac{6}{x}$ with initial guesses $x_l = 1$ and $x_u = 5$ and desired absolute error $|E_a| = 0.00001$.

3. Given $f(x, y) = 2xy + 2y - 1.5x^2 - 2y^2$,

(a) Start with an initial guess of $(x_0, y_0) = (1, 1)$ and determine (by hand is fine) two iterations of the steepest ascent method to maximize $f(x, y)$.

(b) What point is the steepest ascent method converging towards? Justify your answer without computing any more iterations.