MAE3456/MEC3456 LAB 05

Due: 10:00PM, Friday 27th May 2020 (Mid Week 10)

This lab should be completed INDIVIDUALLY. Plagiarism will result in a mark of zero. Plagiarism includes letting others copy your work and using code without citing the source. Collaborating with others to discuss algorithms and details of MATLAB syntax and structures is acceptable (indeed encouraged), however you *MUST* write your own MATLAB code. All assignments will be checked using plagiarism-detecting software and similarities in submitted code will result in a human making a decision on whether the similarity constitutes plagiarism.

INSTRUCTIONS

Download **`template.zip`** from Moodle and update the M-Files named **`Lab05_Q1.m`**, etc… with your Lab code. DO NOT rename the M-Files in the template or modify **`run_all.m`** Check your solutions to the questions by running **`run_all.m`** and ensuring all questions are answered as required.

SUBMITTING YOUR ASSIGNMENT

Submit your assignment online using Moodle. You must include the following attachments:

A **ZIP** file (**NOT .rar or any other format**) named in the following way:

Surname_StudentID_Lab_5.zip (e.g. Rudman_23456789_Lab_5.zip)

The zip file should contain one folder containing everything needed for Lab05, i.e. the following files:

a. All MATLAB m-files for lab tasks: **run_all.m**, **LabNN_ Q1.m**, etc…

b. Any additional MATLAB function files required by your code

c. All data files needed to run the code, **including any input data provided to you**

d. Any hand calculations or written responses asked for - scanned in as a ***SINGLE*** PDF file

YOUR ZIP FILE WILL BE DOWNLOADED FROM MOODLE AND ONLY THOSE FILES INCLUDED IN YOUR SUBMISSION WILL BE MARKED – YOU WILL GET ZERO MARKS FOR ANY MISSING FILES – DON'T ASK ME TO ACCEPT THEM LATE AS I WILL SAY NO.

We will extract (unzip) your ZIP file and mark the lab based on the output of **run_all.m** and any hand calculations or written responses. It is your responsibility to ensure that everything needed to run your solution is included in your ZIP file.

STRONG RECOMMENDATION: After uploading to Moodle, download your Lab to a NEW folder, extract it and open a ***NEW*** instance of **MATLAB**. Ensure your code runs as expected. ***CHECK YOUR PDF IS INCLUDED***. You can upload your submission as many times as you like ***BEFORE*** the submission deadline. After this time, it is not possible to resubmit your lab without incurring a late

penalty. If you forget to include some of your code or your PDF you cannot include it later without penalty.

MARKING SCHEME

This lab is marked out of 80 and full marks is worth 8% of your total unit grade for the semester. This lab will be graded using the following criteria:

1) `run_all.m` produces results **automatically** (no additional user interaction needed except where asked explicitly – NOTE, I have included pause commands in run_all so that intermediate answers can easily be viewed by the demonstrators – please don't remove them)
2) Your code produces correct results (printed values, plots, etc…) and is well written.
3) Programming style, efficiency of algorithm and quality of output (figures, tables, written text ...) will be assessed in this lab and marks will be lost for poor style.

ASSIGNMENT HELP

1) You can ask questions in the Discussion Forum on Moodle
2) Hints and additional instructions are provided as comments in the assignment template M-Files
3) Hints may also be provided during lectures
4) The questions have been split into sub-questions. It is important to understand how each sub-question contributes to the whole, but each sub-question is effectively a stand-alone task that does part of the problem. Each can be tackled individually.
5) I recommend you break down each sub-question into smaller parts too, and figure out what needs to be done step-by-step. Then you can begin to put things together again to complete the whole.
6) To make it clear what must be provided as part of the solution, I have used bold italics and a statement that (usually) starts with a verb (e.g. ***Write a function ..., Print the value...***, etc.)

QUESTION 1 [8 MARKS TOTAL]

Background – Adams-Moulton schemes

The Adams-Moulton integration schemes are one of the many ways to find solutions of the ODE $dy/dt=f(t,y)$. In particular they aim to approximate the integral in the update

$$y_{i+1} = y_i + \int_{t_i}^{t_{i+1}} f(t,y)dt \qquad \text{(Eq 1)}$$

They are based on fitting a polynomial to the set of data points (t_{i-n}, f_{i-n}), (t_{i-n+1}, f_{i-n+1}), … (t_i, f_i), (t_{i+1}, f_{i+1}) and integrating this polynomial, but ***ONLY*** over the interval t_i, t_{i+1}. The underlying idea is that the polynomial fitting uses information in the "past" to provide a better information of what is happening over the next step (i.e. t_i to t_{i+1}). The drawback of the AM schemes is that the solution at the new (unknown) time level t_{i+1} is required, (i.e. we need y_{i+1} so we can calculate f_{i+1}). This is generally approximated from a predictor step that can be obtained from (for example) an Adams-Bashforth estimate. In the question below, we will assume that such an estimate of f_{i+1} is available. See Lecture 20 for more detail.

Q1

Using three points equally spaced in time, (t_{i-1}, t_i, t_{i+1}) and the associated function values at these times (f_{i-1}, f_i, f_{i+1}) derive the 2 segment Adams-Moulton scheme (AM2) for estimating the integral on the RHS of Eqn 1.

Hints: you should make the following assumptions:

1. Let the (constant) spacing in time be equal to h
2. Without loss of generality, shift your time axis so that $t_{i-1} = -h$, $t_i = 0$ and $t_{i+1} = h$.
3. Fit a quadratic polynomial to the (t, f) points, i.e. $f = At^2+Bt+C$ (determine A, B, C, in terms of h and f_{i-1}, f_i, f_{i+1})
4. Analytically integrate the resulting polynomial over the time interval $[t_i, t_{i+1}]$
5. Write the resulting formula for the AM2 update (i.e. the AM2 approximation of Eqn 1)

SHOW ALL WORKING in your PDF

QUESTION 2 [8 MARKS TOTAL]

Background:

Consider the second order ODE

$$\frac{d^2q}{dx^2} - \left(\sin^2 x\right)\frac{dq}{dx} + I\, q = 0 \qquad \frac{-p}{2} \le x \le \frac{p}{2}$$

Eqn 2

$$\text{with BCs} \qquad q\left(\frac{-p}{2}\right) = 0 \quad \text{and} \quad q\left(\frac{p}{2}\right) = 0$$

These boundary conditions can only be satisfied for certain values of the parameter, λ, known as eigenvalues. Below, you will be asked to analyse this equation.

Q2a

In your PDF, write the finite difference equation that approximates this equation at an arbitrary point, x_i assuming a uniform mesh spacing, h.

Q2b

Using the relaxation method and a mesh spacing of h=π/4, by hand explicitly write the matrix equation that approximates the eigenvalue problem. Given that you have specified BCs *at* $x = \pm\pi/2$ your system should be a 3x3 matrix problem for unknowns defined at x = -π/4, 0, π/4.

DO NOT ATTEMPT TO SOLVE THE SYSTEM

Q2c

Write the second order equation (Eqn 2) as two first order equations that can be solved using one of the ODE integrators we have considered in this unit given a value of λ. Clearly specify what boundary conditions or constraints you will need to use at x=-π/2 in your numerical integration, and what boundary condition (or constraint) you want to satisfy at x=π/2. Your adjustable parameter is λ.

QUESTION 3 [28 MARKS TOTAL]

Q3a

Using ANY accurate ODE solver (I suggest your RK4A code from Lab 04, but feel free to use ODE45 if you like) ***write a MATLAB function*** that integrates the system of 1st order ODEs that is equivalent to **Eqn 2** from x = -π/2, to x = π/2 given a guess for the eigenvalue, λ.

This function should have a header like

```
function [qRHS] = BValR(lambda)
```

where the input parameter is

- lambda – the eigenvalue guess

and the output parameter is

- qRHS is a scalar value that is the estimate of q at x = π/2, given lambda.

You will need to call RK4A (or ODE45) inside this function.

Q3b

Modify the template in Lab05_Q3.m so that it calculates the RHS value of q at $x = \pi/2$ for values of λ in the range $\lambda \in [0,50]$. (Choose as many values as you think are needed to make a good plot).

Plot the predicted value of qRHS as a function of λ for values of λ in the range $\lambda \in [0,50]$.

Q3c

Recall that the eigenvalues of Eqn 2 are those values of λ which satisfy the correct BC at $x = \pi/2$ after we have integrated from $x = -\pi/2$, i.e. they are the zeros of the function specified by BValR(λ).

Modify the template in Lab05_Q3.m so that it finds the first 5 eigenvalues of the equation ***automatically*** (i.e. using MATLAB code, not your eyes).

Print the first 5 eigenvalues to the command window to 5 decimal places and with suitable identification (i.e. don't just write 5 numbers).

HINT: One way to do this is to use the data you calculated in Q3b to determine when uRHS changes sign and use the corresponding values of lambda as bounds in a bracketed root finding method such as bisection. If you have a different way to find the roots, you can use it provided you ***write a brief statement to the command window*** outlining your method. HOWEVER, whatever method you use, it must ***not*** require human intervention and must be entirely contained in the algorithm logic.

Q3d

For the first, second and fourth eigenvalue, ***calculate*** the associated eigen solution. ***Plot*** both in the same figure with a solid line (a different colour for each).

QUESTION 4 [8 MARKS TOTAL]

Q4 Hand calculations (see Lectures 21 and 22 for relevant content)

Consider the 1-D diffusion equation:

$$\frac{\partial q}{\partial t} = k \frac{\partial^2 q}{\partial x^2} \qquad \textbf{Eqn 3}$$

Write the centered in space, Crank-Nicolson (CN) discretization of this equation.

Consider just ONE Fourier mode of the representation of the error, and write it in the simplified form (defined at time level n, and node j) as

$$\varepsilon_j^n = A^n e^{iKj\delta x} \qquad \textbf{Eqn (4)}$$

Using this Fourier mode, ***undertake a von Neumann stability analysis*** for the CN discretization and derive (and clearly state) the amplification factor as a function of the timestep δt. You can assume that the error satisfies the same difference equation as the solution for Q, (because it does).

SHOW ALL YOUR WORKING

Compare the amplification factor for CN to the amplification factor for the Forward in Time, Centered in Space (FTCS) discretization derived in lectures. Is the CN method stable, conditionally stable or unstable? Comment in your PDF.

QUESTION 5 [28 MARKS TOTAL]

The theta scheme is a variation on the Crank-Nicolson scheme that does not exactly average the spatial derivative at times n and n+1. For the 1D diffusion equation (Eqn 3) the theta scheme is written:

$$q_j^{n+1} = q_j^n + \frac{1}{2}\frac{k dt}{dx^2}\left\{(1-\theta)\left(q_{j+1}^n - 2q_j^n + q_{j-1}^n\right) + (1+\theta)\left(q_{j+1}^{n+1} - 2q_j^{n+1} + q_{j-1}^{n+1}\right)\right\} \qquad \textbf{Eqn (5)}$$

In Eqn 5, the parameter θ is ***usually*** chosen to be a positive number with $\theta << 1$ (although we will choose different values below). When $\theta = 0$, we recover the Crank-Nicolson scheme, when $\theta = 1$ we recover the BTCS scheme and when $\theta = -1$ we recover the FTCS scheme.

See Lecture 22 and associated PDFs for a discussion of both Crank-Nicolson and the theta scheme.

You will be asked solve the diffusion equation using the theta scheme for $q(x,t)$ on a domain $0 \le x \le 10$ with boundary conditions given at the LHS by $q(0,t) = 1.0$ (homogeneous Dirichlet BC) and $\partial q/\partial x = 0$ at $(x = 10)$ (i.e. homogeneous Neumann BCs).

The ***initial conditions*** are the step function

$$q(x,0) = -1 \text{ for } 0 < x \le 5 \quad \text{and} \quad q(x,0) = +1 \text{ for } 5 < x \le 10 \qquad \textbf{Eqn (6)}$$

Q5a

If all unknown values in Eqn 5 (at time n+1) are moved to the LHS and all others kept on the right Eqn 5 can be written as a matrix

$$[\mathbf{L}]\mathbf{q}^{n+1} = [\mathbf{R}]\mathbf{q}^n + [\mathbf{BC}] \qquad \text{OR} \qquad \mathbf{q}^{n+1} = [\mathbf{L}]^{-1}[\mathbf{R}]\mathbf{q}^n + [\mathbf{L}]^{-1}[\mathbf{BC}] \qquad \textbf{Eqn (7)}$$

Assume the spatial grid is discretized with "M+1" grid points, with the LHS value of the solution given by $q_0 = q(0)$, and the solution at the RHS by $q_M = q(10)$. Because the BC at 0 is known, we can remove q_0 from the set of unknowns and thus we will have just "M" unknowns, starting with $q_1 = q(\delta x)$.

As done in the lectures, ***by hand write the structure*** of these matrices (**L**, **R**) and the **BC** vector and include in your PDF. Focus on the equations for the first unknown (node 1), a "typical" unknown away from the boundary (say node j) and the equation at the RHS (node M).

Q5b

Write a MATLAB function that solves the 1D heat equation using the theta scheme with boundary conditions as specified above. Ensure that your function has the function header

```
function [tsol, qsol] = theta_solve(tmax, q0,dx,dt,kappa,theta)
```

where the input parameters are

- tmax is the end time of the integration (assume it is an integer multiple of the timestep, dt)
- q0 is a vector of the initial condition (time 0) of the solution (i.e. $q(x,0)$) including the BCs
- dx is the uniform grid spacing in x
- dt is the constant time step
- kappa is the diffusion coefficient
- theta is the theta coefficient

and the output parameters are

- tsol – a vector with the times that the solution has been calculated at
- qsol is a matrix, the n^{th} column of which is the solution $q(x,t^{n+1})$ (includes the BCs)

HINT: Using a matrix formulation of this solution method as discussed in **Q5a** is essential because it is an implicit scheme where all unknowns are coupled to each other. The matrices **L** and **R** (and the inverse $\mathbf{L}^{-1}$) only need to be determined ONCE because the timestep and BCs do not change in time.

How to implement the BCS:

1. The boundary condition at $x = 0$ is easily implemented using $q_0 = q(0) = 1$. This BC should be explicitly included in the difference equations so that q_0 is NOT one of your unknowns.
2. You ***MUST*** set the boundary condition at $x = 10$ using a ***CENTRED finite difference*** approximation to the derivative. Thus (as per the details in Lecture 22) at the RHS boundary you will need to include the diffusion equation (i.e. Eqn 5) centered ***AT*** the boundary node (node "M") as one of your equations, ***BUT*** with the unspecified variable (q_{M+1}) replaced using the centered finite difference approximation to the BC at node M.

Q5c

Modify the file **Lab05_Q5.m** so that it solves Eqn 5 by calling theta_solve with the following conditions:

- Uses a value of kappa = 1
- Discretizes the domain with 100 intervals (101 points).
- Uses tmax = 20.0
- Sets the initial condition given in Eqn 6
- Chooses a theta parameter of θ = -1 (i.e. the FTCS method)

From the lectures, you know what the stability timestep restriction is on the 1D heat equation ***IF*** you were to use the FTCS scheme – ***calculate this value in your code*** for your parameters and call it dtmaxC. ***Print dtmaxC*** to the command window with a suitable statement (e.g. " Time step restriction for FTCS scheme = …")

Integrate forward in time from the initial condition to t=20 with 3 different timesteps of: 0.5*dtmaxC, 0.99*dtmaxC and 1.001*dtmaxC.

- For each different timestep, ***plot ONE FIGURE.*** This figure will contain the predicted solution at times given by t = 0, 4, 8, 12, 16, 20). After this, you should have 3 plots, each with 6 curves).
  (DO NOT be concerned about ensuring the plots are at EXACTLY the times requested above, although they should be close. For this lab only, near enough in time is good enough.)

Write a brief statement to the command window that describes what you observe in each plot. Include answers to whether the FTCS method is stable for all these dt and if your results are what you expect, or not?

This method of implementing the FTCS scheme is not very efficient – ***write a sentence to the command window*** that says why it is not efficient.

Q5d

Repeat **Q5c** (including plotting the figures) with the following changes:

1. Use θ = 0.0 which corresponds to the Crank-Nicolson method.
2. Use timesteps of 0.5, 1 and 2

Modify the file **Lab05_Q5.m** as required to implement the above.

Write a brief statement to the command window that describes what you observe in each plot. Include answers to whether the CN method stable for all these dt and if your results are what you expect, or not? Ensure you mention how these dt compare to dtmaxC calculated in **Q5c**.

Q5e

Repeat **Q5d** (including plotting the figures) with the following changes:

1. Use θ = 0.1 which corresponds to the genuine theta scheme

Modify the file **Lab05_Q5.m** as required to implement the above.

Write a brief statement to the command window that describes what you observe compared to the Crank-Nicolson solution. Based on your results, write a statement that addresses the statement "The theta scheme will remove oscillating errors that can arise in the CN scheme."

Poor Programming Practices [-10 Marks]

(Includes, but is not limited to, poor coding style or insufficient comments or unlabeled figures, etc.)

(END OF LAB)