EE2-08C - Numerical Analysis of Differential Equations using Matlab - Coursework 2019

The coursework is due for submission on Monday 18 March 2019 either to me or the UG office room 608, before 4pm. You should hand in a *printed* copy of the report. Before 22:00pm on Monday 18 March, you should also submit the pdf of the report on Blackboard. Please send the matlab files *separately* by email. Each group should submit only once: choose a member of your group to submit the pdf on BB and email the matlab files on behalf of the group. Include your group number in all correspondence.

Before embarking on this coursework it is useful to work through the exercises in Example Sheets 1 and 2, covering Euler's method, Heun's method and second-order equations.

Your report should include: 1) all the code, well commented/explained. That means explain how the algorithm works and how it solves the given ODE/PDE, or how error analysis is carried out and what it means. 2) All plots, with brief comments explaining the physical significance of the solutions in the plots, e.g. things like resonance, transient/steady-state, corner frequency, high/low-pass, etc. In Q1-3, that is in the context of the circuit.

Above all, you should write as if for one of your peers: your audience is someone with your level of knowledge of Mathematics and Circuits, but who doesn't know this particular topic, and you are explaining it to them. Don't assume the reader already knows everything!

1 RL circuit

A high-pass filter takes an input signal V_{in} (for instance, an audio signal) and let only the high-frequency components pass. Usually, high-pass filters are made with capacitors instead of inductors due to the fact that inductors can be more easily manufactured and are generally physically smaller. However, the example of RL circuit we are going to consider is still very important. In fact, RL circuits are usually used in the modeling of real physical systems such as DC motors.

Consider the RL circuit as depicted in Figure 1. For the circuit, we can write the following equations:

$$\begin{aligned} v_L(t) + v_R(t) &= V_{in}(t) \\ L\,\frac{d}{dt} i_L(t) + R\, i_L(t) &= V_{in}(t)\,. \end{aligned}$$

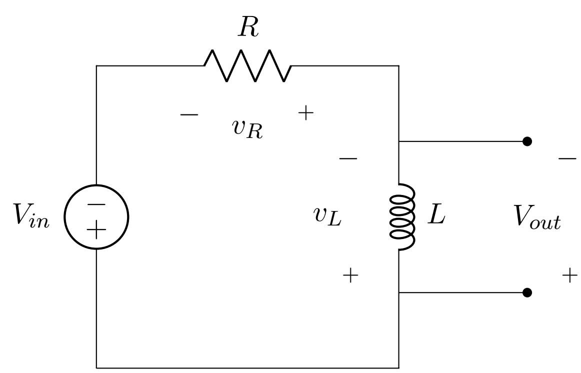


Figure 1: RL circuit

The state is $i_L(t)$. The input is $V_{in}(t)$. The output is $V_{out} = V_{in}(t) - Ri_L(t)$. Assume that the initial current through the inductor at time $t = 0$ is $i_L(0) = 0\,\mathrm{A}$. We want to model a DC motor with inertia $250\,\mu\mathrm{Nm/s}^2$ and maximum torque gain $50\,\mathrm{mN\,m\,A^{-1}}$. The corresponding values for the inductance and resistance are:

$$\begin{aligned} R &= 0.5\,\Omega \\ L &= 1.5\,\mathrm{mH}\,. \end{aligned}$$

Exercise 1. Take three second-order Runge Kutta methods: Heun's and Midpoint, covered in lectures and a new one which you will devise and call MyMethod. Recall the conditions required for a second-order RK method to be valid. Explain why you make the choices that you make.

For each method, write a matlab function, so for Heun's method, there will be a matlab function called **heun.m**, implementing the Heun method to model the RL-circuit, with first line:

```
function [t,vout]. = heun(?,?,?,...)
```

The arguments for the function heun.m need to be carefully chosen and one will need to be a function; another argument needs to be the final time t_f. The function should solve any first-order ODE $y' = f(x, y)$, not just the one corresponding to this circuit; likewise for the other two methods.

(ii) The other file should be a script called **RK2_script.m**, in which you call heun.m (and the other two ***.m functions) to model the circuit for different types of input, and plot the output. When you vary the system parameters and input function, give some thought about choosing t_f carefully to allow the features of the output to be seen in the plots, Simulate the system and obtain plots for the following different input signals and observe how the amplitude of the output signal changes:

1. step signal with amplitude $\bar{V}_{in} = 3.5\,\mathrm{V}$;
2. impulsive signal and decay:
$$V_{in} = \bar{V}_{in} \exp\left\{-\frac{t^2}{\tau}\right\} \qquad V_{in} = \bar{V}_{in} \exp\left\{-\frac{t}{\tau}\right\}$$
with $\bar{V}_{in} = 3.5\,\mathrm{V}$ and $\tau = 150\,(\mu\mathrm{s})^2$, resp. $\tau = 150\,\mu\mathrm{s}$.
3. sine, square, and sawtooth waves with amplitude $\bar{V}_{in} = 4\,\mathrm{V}$ and different periods $T = 150\,\mu\mathrm{s}$, $T = 15\,\mu\mathrm{s}$, $T = 400\,\mu\mathrm{s}$, $T = 1100\,\mu\mathrm{s}$. For square and sawtooth wave, try Matlab help: "square-", resp. "sawtooth wave".

When you have gone through the given set of parameters, try some new ones: vary $R, L, i_L(0), \tau$, T, etc; try some different input functions V_{in}; see if you can get different kinds of behaviour; this part of the exercise is **open-ended** and you need to be inventive, adventurous! Don't hesitate to explore: if you obtain non-sense, think about why this happened, and maybe include that in the report.

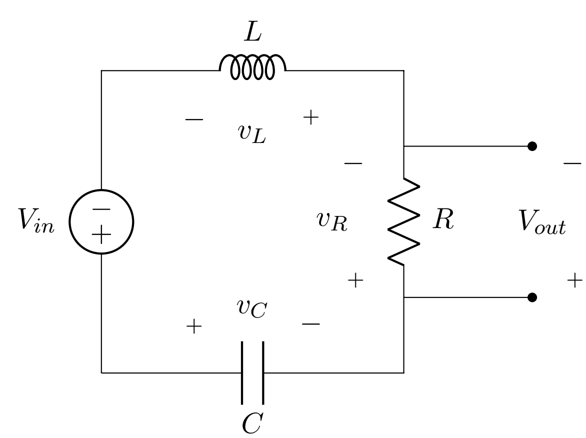
The behaviour of the output can be explained in terms of the input, parameters, the nature of the circuit. In each case, explain briefly what you observe. Code, pictures of the output and comments go in the report.

Exercise 2. Write a script called **error_script.m** in which you carry out error analysis given as input a cosine wave of period $T = 150\,\mu\mathrm{s}$ and amplitude $\bar{V}_{in} = 6\,\mathrm{V}$. Use your favourite method to obtain the exact solution of the ODE and compare the numerical solution with the exact solution, obtaining the error as a function of t. Plot the error function for each method. Vary the time step h for a suitable number and range of values, and obtain a log-log plot to show the order of the error. Compare the errors between the three methods, do you notice anything? What do you notice, and why do you think this is happening?

2 RLC circuit

An RLC circuit represents a standard example of a harmonic oscillator, namely a device able to resonate to a sinusoidal input signal. Among the many applications for this circuit, there is the tuning of analogue radio receivers. Usually, they are used as band-pass filters.

Consider the RLC circuit as depicted in the Figure. For the circuit, we can write the following equations:



$$\begin{aligned} v_L(t) + v_R(t) + v_C(t) &= V_{in}(t) \\ L\frac{d}{dt}i_L(t) + R\,i_L(t) + \frac{1}{C}\int_0^t i_L(t) &= V_{in}(t) \\ L\frac{d^2}{dt^2}q_C(t) + R\frac{d}{dt}q_C(t) + \frac{1}{C}q_C(t) &= V_{in}(t)\,. \end{aligned} \tag{1}$$

The state is $q_C(t)$. The input is $V_{in}(t)$. The output is $V_{out} = v_R = R\frac{d}{dt}q_C(t)$. Assume that the capacitor is pre-charged at time $t = 0$ with $q_C(0) = 500\,\text{nC}$. Moreover, no initial current flows through the inductor at time $t = 0$: $i_L(0) = \frac{d}{dt}q_C(0) = 0\,\text{A}$. The values for resistance, capacitance, and inductance are:

$$R = 250\,\Omega\,, \quad C = 3\,\mu\text{F}\,, \quad L = 650\,\text{mH}\,.$$

Exercise 3 Write a matlab script called **RLC_script.m** and a matlab function called **RK4second.m**. In the script you should set up the two coupled first order ODEs in q and q' to solve the RLC second order ODE (1) for q. The script should include calls to **RK4second.m** which should be written to implement the fourth-order *Runge-Kutta 3/8* algorithm, for a **single time-step**, to obtain x_{i+1} and y_{i+1} from $x_i\,, y_i$ and t_i. The algorithm to be used is the "Runge-Kutta 3/8 " method, and you will need to find this in the literature.

The function call should include arguments $x_i\,, y_i$ and t_i and the function should return x_{i+1} and y_{i+1}. As in Exercise 1, the function RK4second.m should be flexible enough to solve *any* pair of coupled first-order ODEs, not just the ones in this exercise, while everything specific to this exercise will be in the script file.

Once you have the matlab code working, use it to simulate the system, and obtain plots, for different input signals and observe how the amplitude of the output signal changes:

- step signal with amplitude $\bar{V}_{in} = 5\,\text{V}$;
- impulsive signal with decay
$$V_{in} = \bar{V}_{in} \exp\left\{-\frac{t^2}{\tau}\right\}$$
with $\bar{V}_{in} = 5\,\text{V}$ and $\tau = 3\,(\text{ms})^2$.
- square wave with amplitude $\bar{V}_{in} = 5\,\text{V}$ and different frequencies $f = 5\,\text{Hz}$, $f = 100\,\text{Hz}$, $f = 500\,\text{Hz}$;
- sine wave with amplitude $\bar{V}_{in} = 5\,\text{V}$ and different frequencies $f = 5\,\text{Hz}$, $f = 100\,\text{Hz}$, $f = 500\,\text{Hz}$.

When you vary the system parameters and input function, give some thought about choosing t_f carefully to allow the features of the output to be seen in the plots,

When you have gone through the given set of parameters, try some new ones: vary R, L, C, τ, T, initial conditions, etc; try some different input functions V_{in}; see if you can get different kinds of behaviour; this part of the exercise is **open-ended** and you need to be inventive, adventurous! Don't hesitate to explore: if you obtain non-sense, think about why this happened, and maybe include that in the report.

The behaviour of the output can be explained in terms of the input, parameters, the nature of the circuit. In each case, explain briefly what you observe. Code, pictures of the output and comments go in the report.

3 Finite Differences for PDE

The 1-D heat equation $\dfrac{\partial y}{\partial t} = \dfrac{\partial^2 y}{\partial x^2}\,, \quad 0 < x < 1\,, \quad t > 0$

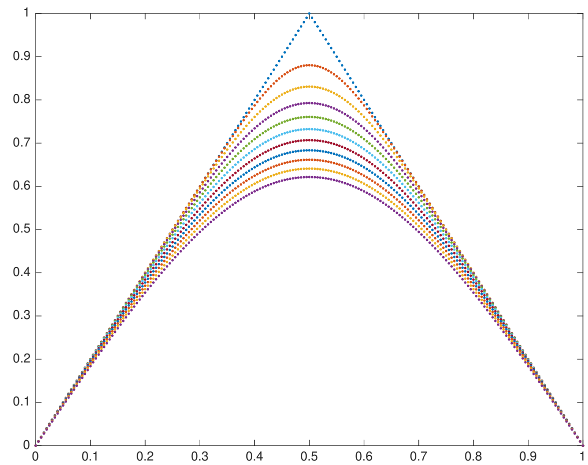
with zero boundary conditions $y(0,t) = y(1,t) = 0$, and initial condition $y(x,0) = y_0(x)$ can be solved numerically using the finite difference method outlined in lectures.

Exercise 4. Write a matlab script called **finite_script.m** to implement the finite difference method, and solve the heat equation. The initial condition given by $y(x,0) = y_0(x)$ should be tested for the two basic cases seen in lectures, a tent function and a sinusoidal function:

$$(i)\, y_0(x) = \begin{cases} 2x, & \text{for } x \in [0, 0.5] \\ 2 - 2x & \text{for } x \in [0.5, 1] \end{cases} \qquad \text{and} \qquad (ii)\, y_0(x) = \sin(2\pi x)\,.$$

Test it also for $y_0(x) = |\sin(2\pi x)|$. Can you explain the behaviour you observe? Finally, test it for two further initial conditions of your choice: one should include polynomial terms, the other exponential term(s). Be sure to supply plots. Some points you need to consider:

- How to allocate the boundary conditions U_0^m and U_N^m;
- How to implement the central algorithm $U_j^{m+1} = vU_{j-1}^m + (1-2v)U_j^m + vU_{j+1}^m$;
- How to choose h, k;
- How to plot U_j^m for a sensible choice of values of m, to be able to visualize progress as time increases. Here is what you expect for the tent function:



Each set of coloured dots represents U_j^m for one value of m and $j = 0 \ldots N$. The top set is the initial condition; as time increases, the values all decrease.

Exercise 5

(i) Extend your work on the PDE solution in (**4**) to include an initial condition that does not match one or both boundary conditions. For example, take zero boundary conditions and initial condition $y_0(x) = 1 - x$ or $y_0(x) = \cos(\pi x/2)$: both match the BC at $x = 1$, but not the other BC at $x = 0$. Experiment! You will need to take care with the visualization here.

(ii) Further extend by including constant, non-zero boundary conditions.

(iii) Further extend to include time-varying boundary conditions.

Exercise 6

Use Taylor series to derive the finite-difference approximations

$$\frac{y(x_i + h) - y(x_i - h)}{2h} \approx \frac{dy}{dx} \quad \text{and} \quad \frac{y(x_i + h) - 2y(x_i) + y(x_i - h)}{h^2} \approx \frac{d^2y}{dx^2},$$

and obtain the order of the error incurred in the approximation.