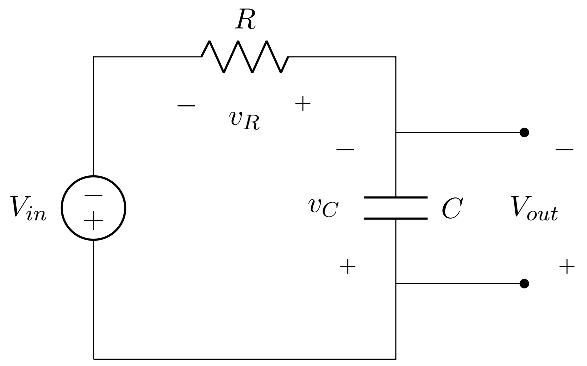
EE2-08C - Numerical Analysis of ODEs using Matlab - Coursework 2018

The coursework is due for submission on Friday 16 March 2018 either to me or the UG office room 607, before 4pm. You should hand in a *printed* copy of all your matlab files with thorough comments as well as the plots you generate, clearly labelled. Before 23:30pm on Friday 16 March, you should also submit the pdf of the report on Blackboard. Each group should submit only once: choose a member of your group to submit the items on behalf of the group. Do not submit text as image, only plots should be images. Send me an email with a zip file containing *at most* seven matlab files, as outlined below, subject line: *matlab files group X* .

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.

1 RC circuit

A low-pass filter takes an input signal V_{in} (for instance, an audio signal) and renders it smooth by removing all high-frequency components (for instance, thermal noise). We will now observe a high-pass filter usually used in NAB filters to cut-off frequencies above 1.592 kHz.



Consider the RC circuit as depicted above. For the circuit, we can write the following equations:

$$v_C(t) + v_R(t) = V_{in}(t)\,, \quad \frac{1}{C}\int_0^t i_C(t) + R\,i_C(t) = V_{in}(t)\,, \quad \frac{1}{C}\,q_C(t) + R\,\dot{q}_C(t) = V_{in}(t)\,.$$

Note:

1. the state is $q_C(t)$, with initial condition: capacitor pre-charged at time $t = 0$ with $q_C(0) = 500\,\text{nC}$.
2. the input is $V_{in}(t)$; the output is $V_{out} = \frac{1}{C}q_C(t)$,

To impose a cut-off frequency of above 1.592 kHz, in a standard NAB filter, we set:

$$R = 1\,\text{k}\Omega\,, \quad C = 100\,\text{nF}$$

Exercise 1. Model the behaviour of the RC-circuit using three second-order Runge-Kutta methods: Heun's method, the midpoint method and Ralston's method. The first two were covered in lectures, you will have to find the third one in the literature.

Please write **two files only**, one called **RK2.m** which implements all three methods. Given that they are second-order RK2 methods, you should be able to change from Heun to midpoint to Ralston with changes in two lines of code only. Submit it with the lines for one method active and the others commented out with %. It should be flexible enough to solve any first-order ODE of the form

$$\frac{dy}{dx} = f(x, y)$$

given an initial condition $y(x_0)$, a final value x_f, and a step-size h. You should write it using variable names k_1 and k_2, corresponding to the second-order RK methods.

The details for each case need to be passed as arguments to the function **RK2.m** in the file of the same name. See Problem sheet 1 for the Euler function example. Your second file **RK2-script.m** should contain calls to the function **RK2.m**, for a number of different examples, as outlined below.

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 $\bar{V}_{in} = 2.5\,\text{V}$;
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 $\bar{V}_{in} = 2.5\,\text{V}$ and $\tau = 100\,(\mu\text{s})^2$, resp. $\tau = 100\,\mu\text{s}$.
3. sine, square, and sawtooth waves with amplitude $\bar{V}_{in} = 5\,\text{V}$ and different periods $T = 100\,\mu\text{s}$, $T = 10\,\mu\text{s}$, $T = 500\,\mu\text{s}$, $T = 1000\,\mu\text{s}$. For square and sawtooth wave, try Matlab help: "square-", resp. "sawtooth wave".

Note that in each case you will need to make a decision on how large t_f should be: this too, will depend on what you can/want to show, such as, for example, transient vs. steady-state. Make sure your pictures show sensible choices.

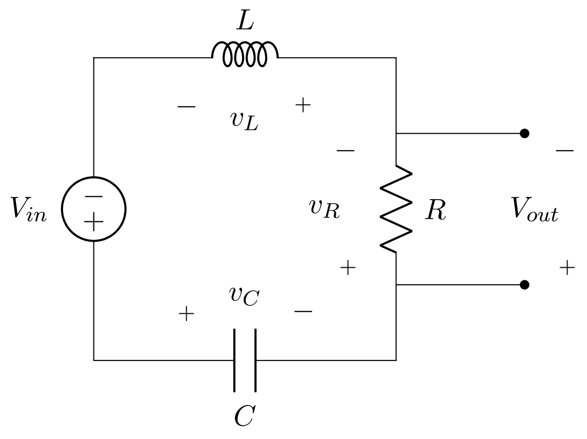
When you have gone through the given set of parameters, try some new ones: vary $R, C, q_C(0), \tau$, and T; try some different input functions V_{in}; see if you can get different kinds of behaviour; this part of the exercise is open-ended and you need to be inventive.

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 for the basic case with $R, C, q_C(0)$ as above, and as input a cosine wave of period $T = 100\,\mu\text{s}$ and amplitude $\bar{V}_{in} = 5\,\text{V}$. Obtain the exact solution of the ODE, and for each method of the three from Exercise 1, obtain a numercial solution: again, switching methods by only commenting out lines of code appropriately. Compare the numerical solution with the exact solution, obtaining the error as a function of t. Plot the error function. 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. Be sure to explain and justify your empirical observation.

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. For the circuit, we sum voltages: $v_L(t) + v_R(t) + v_C(t) = V_{in}(t)$ and derive the ODEs:

$$L\,\frac{d}{dt}i_L(t) + R\,i_L(t) + \frac{1}{C}\int_0^t i_L(t) = V_{in}(t) \qquad \equiv \qquad L\,\frac{d^2}{dt^2}q_C(t) + R\,\frac{d}{dt}q_C(t) + \frac{1}{C}\,q_C(t) = V_{in}(t)\,.$$

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 $q_C(0) = 500\,\mathrm{nC}$. Moreover, no initial current flows through the inductor at time $t = 0$: $i_L(0) = \frac{d}{dt}q_C(0) = 0\,\mathrm{A}$. The values for resistance, capacitance, and inductance are:

$$R = 250\,\Omega\,, \quad C = 3.5\,\mu\mathrm{F}\,, \quad L = 600\,\mathrm{mH}\,.$$

Exercise 3 Write a matlab script called **RLC_script.m** and a matlab function called **RK4.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 will include a call to **RK4.m**, the matlab function you will write to implement the *classic fourth-order Runge-Kutta* algorithm for **any** system of two coupled first order equations:

$$z' = f_1(x, y, z) \qquad \text{and} \qquad y' = f_2(x, y, z)$$

with initial conditions $z = z_0$ and $y = y_0$ when $x = 0$; note that both y and z are functions of x.

You will have to think about which functions and parameters you need to pass as arguments in the call to **RK4.m** (Please note: second-order refers to the ODE, fourth-order to the order of the RK algorithm.)

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:

1. step signal with amplitude $\bar{V}_{in} = 5\,\mathrm{V}$;
2. impulsive signal with decay $V_{in} = \bar{V}_{in}\exp\left\{-\frac{t^2}{\tau}\right\}$, with $\bar{V}_{in} = 5\,\mathrm{V}$ and $\tau = 3\,(\mathrm{ms})^2$.
3. square wave with amplitude $\bar{V}_{in} = 5\,\mathrm{V}$ and different frequencies $f = 109\,\mathrm{Hz}$, $f = 5\,\mathrm{Hz}$, $f = 500\,\mathrm{Hz}$;
4. sine wave with amplitude $\bar{V}_{in} = 5\,\mathrm{V}$ and different frequencies $f = 109\,\mathrm{Hz}$, $f = 5\,\mathrm{Hz}$, $f = 500\,\mathrm{Hz}$.

As in Exercise 1, when you have gone through the given set of parameters, try some new ones: vary $R, C, Lq_C(0), q_C'(0), \tau$, and f; try some different input functions V_{in}; see if you can get different kinds of behaviour. Note that in each case you will need to make a decision on how large t_f should be: this too, will depend on what you can/want to show, such as, for example, transient vs. steady-state, and how frequencies interact. 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 Relaxation

The Poisson equation in the plane

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = g(x, y)\,,$$

which gives Laplace's equation when $g(x, y) = 0$, can be solved using finite differences in the form of the *relaxation* method, as outlined in lectures.

Exercise 4. Write a matlab script called **relaxation1.m** to implement the relaxation method for Laplace's equation on the unit square, with boundary conditions

$$\begin{aligned}
u(0, y) = u(1, y) &= 0\,, && \text{for} \quad 0 < y < 1; \\
u(x, 1) &= 0\,, && \text{for} \quad 0 < x < 1; \\
u(x, 0) &= 0\,, && \text{for} \quad 0 < x < 0.2 \quad \text{and} \quad 0.8 < x < 1; \\
u(x, 0) &= 1\,, && \text{for} \quad 0.2 < x < 0.8.
\end{aligned}$$

Things to consider:

- your choice of gridsize and residual size;
- efficient testing of residual for convergence.
- Once you have the relaxation method working, try it out for some different, more challenging boundary conditions: this is open ended, be inventive.
- For example, what happens when they BCs don't match at the corners? Does your method reproduce this?
- All code, your choices, all plots into report, well commented.

Now *you choose* a quadratic function in x, y:

$$u(x, y) = ax^2 + by^2 + cx + dy + exy + f$$

with coefficients $a \ldots f \in \{-2, -1, 0, 1, 2\}$, with at least four terms, and at most one of the coefficients a, b equal to zero, for example $u(x, y) = x^2 + y + 2xy - 1$. Obtain a surface plot of your function $u(x, y)$ over the unit square.

Differentiate appropriately to find the Poisson equation satisfied by this function. Obtain also the boundary conditions $u(x, 0)$, $u(x, 1)$, $u(0, y)$, and $u(1, y)$ which the function satisfies.

Exercise 5. Write a matlab script called **relaxation2.m** to implement the relaxation method to include a forcing term and the more complicated boundary conditions. Solve the Poisson equation you have derived and make sure your numerical solution is the same as the surface plot you have obtained. As before: all code, choices, calculations, plots go into the report.

Bonus Exercise Implement the solutions in Exerices 4 and 5, using Successive Over-Relaxation, SOR.