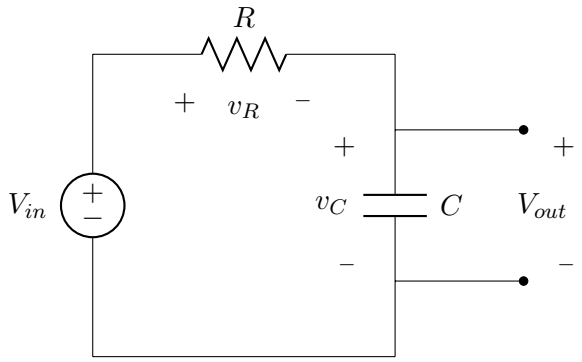
EE2-08C - Numerical Analysis of ODEs using Matlab - Coursework 2016

The coursework is due for submission on Friday 11 March 2016 either to me or the UG office room 608, 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 11 March, you should also submit the pdf of the report as well as the six matlab files *separately* on Blackboard. Each group should submit only once: choose a member of your group to submit the seven items on behalf of the group.

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:

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

Note:

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

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

$$\begin{aligned} R &= 1\,\text{k}\Omega \\ C &= 100\,\text{nF}\ . \end{aligned}$$

Exercise 1. Write two matlab files, one a matlab function called **midpoint.m**, implementing the midpoint method to model the RC-circuit, with first line:

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

The arguments for the function midpoint.m need to be carefully chosen and one will need to be a function to pass the input function $V_{in}(t)$. Another argument needs to be the final time t_f, chosen carefully to allow the features of the output to be seen in the plots. The function should solve the ODE for any values of R, C, t_f and $q_C(0)$, not just those given above. The other file should be a script called **midpoint_script.m**, in which you call midpoint.m to model the circuit for different types of input, and plot the output.

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 $\bar{V}_{in} = 2.5\,\mathrm{V}$;
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 $\bar{V}_{in} = 2.5\,\mathrm{V}$ and $\tau = 100\,(\mu\mathrm{s})^2$, resp. $\tau = 100\,\mu\mathrm{s}$.

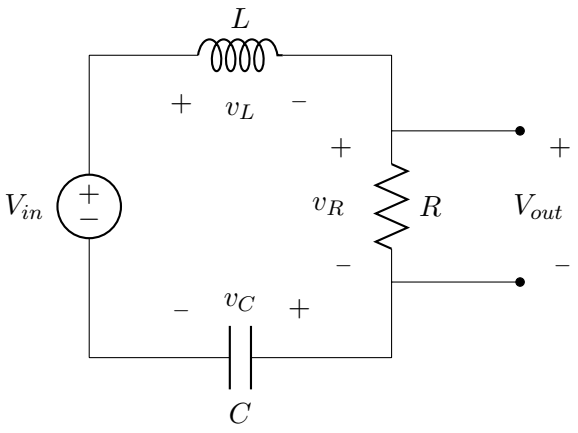
3. sine, square, and sawtooth waves with amplitude $\bar{V}_{in} = 5\,\mathrm{V}$ and different periods $T = 100\,\mu\mathrm{s}$, $T = 10\,\mu\mathrm{s}$, $T = 500\,\mu\mathrm{s}$, $T = 1000\,\mu\mathrm{s}$. For square and sawtooth wave, try Matlab help: "square-", resp. "sawtooth wave".

Exercise 2. Write a script called **error_script.m** in which you carry out error analysis given as input a cosine wave of period $T = 100\,\mu\mathrm{s}$ and amplitude $\bar{V}_{in} = 5\,\mathrm{V}$. 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. 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.

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:



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

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 **rukasecond.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 **rukasecond.m** which should be written to implement the *classic forurth-order Runge-Kutta* algorithm, see lecture notes, for a **single time-step**, to obtain x_{i+1} and y_{i+1} from $x_i\,, y_i$ and t_i. The function call should include arguments $x_i\,, y_i$ and t_i (and others!) and the function should return x_{i+1} and y_{i+1}. (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:

- step signal with amplitude $\bar{V}_{in} = 5\,\mathrm{V}$;
- impulsive signal with decay

  $$V_{in} = \bar{V}_{in} \exp\left\{-\frac{t^2}{\tau}\right\}$$

  with $\bar{V}_{in} = 5\,\mathrm{V}$ and $\tau = 3\,(\mathrm{ms})^2$.
- square wave with amplitude $\bar{V}_{in} = 5\,\mathrm{V}$ and different frequencies $f = 109\,\mathrm{Hz}$, $f = 5\,\mathrm{Hz}$, $f = 500\,\mathrm{Hz}$;
- sine wave with amplitude $\bar{V}_{in} = 5\,\mathrm{V}$ and different frequencies $f = 109\,\mathrm{Hz}$, $f = 5\,\mathrm{Hz}$, $f = 500\,\mathrm{Hz}$.

3 Finite Differences

When the ODE is a Boundary Value Problem (BVP) the method of finite differences can be used. The most general second-order linear inhomogenous ODE with constant coefficients is:

$$A\frac{d^2x}{dt^2} + B\frac{dx}{dt} + Cx = f(t)\,, \qquad x(t_0) = x_0\,, x(t_1) = x_1\,, \tag{2}$$

where $f(t)$ is an input/forcing function, boundary conditions are $x(t_0) = x_0\,, x(t_1) = x_1$, and A, B, C are real constants.

Exercise 4. Write a matlab script called **finite_script.m** to implement the finite difference method, as illustrated in lectures, for the above general ODE. The first lines of your script should read

```
h=0.2;  A=1;  B=-20;  C=0;  t0=0;  tf=1;  x0=1;  xf=0;  func=@(t) -1;
```

This sets the general ODE (2) to the case considered in lectures, of the model reaction-diffusion equation:

$$x'' - 20x' = -1\,, \quad \text{with } x(0) = 1\,, x(1) = 0\,.$$

All code after this should make reference to the parameter names $A, B, C, t_0, x_0, t_f, x_f$ and the function handle func, corresponding to $f(t)$ in (2). This way you can change the ODE to any other case of (2) by changing only a few lines.

Things to consider:

- Obtaining N from t_0, t_f and h. The length of the vectors t and x needs to be matched;
- Calculating the parameters a, b, c from $A, B, C, t_0, x_0, t_f, x_f$, and h – here's one of them: $b = C - 2A/h^2$;
- Given that u_i corresponds to t_i, $f(t_i)$ will be the value on the r.h.s. of the equation for u_i for all $0 < i < N$. What about $i = 0, N$?
- A loop to assemble the vector vec with values of `func(`t_i`)`, etc. This has values $i = 1 \ldots N-1$. Think about what happens for $i = 0, N$;
- Calling the matlab function `solvetridiag(N,a,b,c,vec)` correctly.
- plotting the results t, x.

Once you have replicated the results seen in lectures/problem class for the convection-diffusion equation with $h = 0.2$ and smaller, choose another set of values $A, B, C, t_0, x_0, t_f, x_f$ and a simple (but non-constant) input function $f(t)$, and implement the finite differences method. Make sure your BVP has a solution.

Bonus Exercise You can get full marks by answering all of the above. If you want to improve your mark, you may try the following, but the maximal mark will still be 100%.

Use Taylor series to derive the finite-difference approximations

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

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