MATHS 2107 Statistics and Numerical Methods
Assignment 5
Due: Tuesday 19th October 5pm - (Week 11)

When presenting your solutions to the assignment, please include some explanation in words to accompany your calculations. It is not necessary to write a lengthy description, just a few sentences to link the steps in your calculation. Messy, illegible or inadequately explained solutions may be penalised. The marks awarded for each part are indicated in boxes.

This assignment has 2 questions, for a total of 19 marks.

---

1. Consider the general initial value problem

$$y' = f(t, y), \quad y(t_0) = y_0.$$

A particular Runge–Kutta method for solving the general initial value problem is

$$\begin{aligned} y_{k+1} &= y_k + \tfrac{1}{6}\big[F_1 + 4F_2 + F_3\big], \\ F_1 &= hf(t_k, y_k), \\ F_2 &= hf(t_k + \tfrac{1}{2}h, y_k + \tfrac{1}{2}F_1), \\ F_3 &= hf(t_k + h, y_k - F_1 + 2F_2), \end{aligned} \tag{1}$$

where $y_k \approx y(t_k)$, $t_k = t_0 + kh$, h is the step size and $k = 0, 1, 2, \ldots$

$\boxed{4}$ (a) Use one step of the Runge–Kutta method (1) to find an approximation of $y(h)$ for the initial value problem

$$y' = -\lambda y, \quad y(0) = y_0, \quad \lambda > 0. \tag{2}$$

Show each step of your calculation. Write your answer in terms of λ, h and y_0.

$\boxed{2}$ (b) Write down the first six terms of the Taylor series of the exact solution $y(h)$ for the intial value problem (2). Hence determine the order of error of the approximation found in (a) with respect to step size h.

$\boxed{3}$ (c) Determine the maximum step size h for which

$$\lim_{k\to\infty} y_k = \lim_{t\to\infty} y(t) = 0,$$

where y_k is the Runge–Kutta approximation of the solution $y(t_k)$ to the initial value problem (2).

- Write your answer in terms of λ and a coefficient which you should determine to one decimal place accuracy.
- A graphical method for finding the coefficient is acceptable, but you must include the plot in your report.

2. A simple model for large-scale atmospheric wind speeds is given by the following, in which x refers to east/west wind speeds and y to north/south wind speeds.

$$\frac{\mathrm{d}x}{\mathrm{d}t} = c_1 - Al_1 \sin(k_1 x)\cos(l_1 y) - \epsilon l_2 \sin\{k_2[x - (c_2 - c_1)t]\}\cos(l_2 y) \tag{3a}$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = Ak_1 \cos(k_1 x)\sin(l_1 y) + \epsilon k_2 \cos\{k_2[x - (c_2 - c_1)t]\}\sin(l_2 y) \tag{3b}$$

$\boxed{2}$ (a) Write a MATLAB function called `rossby.m` that computes the derivatives (3) for

$$A = 1,\quad k_1 = \pi,\quad k_2 = 1,\quad l_1 = \pi/2,\quad l_2 = 2,\quad c_1 = 1/2,\quad c_2 = \pi,\quad \epsilon = 0.3.$$

Your function must use and document the interface

```
function dxydt = rossby(t,xy)
```

where `t` is time and `xy` is a column vector containing the dynamic variables $\begin{pmatrix} x \\ y \end{pmatrix}$. The output `dxydt` is a column vector containing the time derivatives $\begin{pmatrix} x' \\ y' \end{pmatrix}$. For full marks, the function must handle the case where the components x and y of the input `xy` are themselves column vectors. In that case each entry in x corresponds to an entry in y, and you are simulating multiple trajectories simultaneously; your output `dxdydt` should be a column vector with dimension equal to the `xy`.

- Document your function with comments. The minimum documentation for a function is a description of what the function does, descriptions of the inputs and outputs, your name, and the modification date.
- Include a copy of the function in your report.
- Submit your function to MATLAB GRADER on MyUni for marking. For full marks, write a vectorised function that computes dx/dy and dy/dy for each input, without using loops. Partial marks are available for a function that uses a loop, and (fewer) partial marks are available for a function that only takes 2-D `xy` input.
- I suggest that in the first lines of your function, you create variables `x` and `y` by defining them appropriately using `xy`. You can then code `dxdt` and `dydt` separately, and the final line of your function can unify them: `dzdt = [dxdt; dydt];`

$\boxed{8}$ (b) Write a script that uses the MATLAB function `ode45` to solve `rossby` by expanding the following script:

```
%% A description of the script
% any critical usage information
% your name, the month and year


clear
close all

rngl(1)
M = 1; %set to 50 for full marks; M is the number of
   initial conditions
```

```
tSpan =[0 2] ;
xy0 = [2*rand(M,1); %x component
       3*rand(M,1) %y component
       ];
dxydt = @rossby; %'rossby' is the name of the m-file
   function for dxydt
odeSolver = @ode45; %'ode45' can be replaced by any
   integrator that takes the same arguments


%% Option 1: simulate using for loop

%%%%This option is simpler to code, but will result in
   fewer marks.


%% Option 2: simulate using vectorised ode


%% plot x vs y


%% plot t vs x and t vs y on subplots
```

- Document your script. The minimum documentation for a script is a description of what the script does, your name and the modification date.
- The lines I have written do the following:
  - Set `M`, the number of (x, y) trajectories to simulate.
  - Set the time interval for simulation.
  - Set the initial conditions in a repeatable fashion.
  - Set a function `dxydt` and a solver `odeSolver`. One reason to do this is to specify as early as possible what problem is being modelled, and what solver is being used.

  You should not change any of these lines except M. Your final submission should have either `M=1`, for partial marks, or `M=50`. If you managed to vectorise `rossby.m`, then you will be able to simulate all 50 trajectories in one go using `[tn,xn] = odeSolver(dxydt,tSpan,xy0);`
- Your script should produce three plots:
  - A plot of all trajectories showing x on the horizontal axis and y on the vertical. Each initial condition should be plotted with a symbol, then plot $x(t)$ vs $y(t)$ for all simulation times t using a line.
  - A plot of all trjectories, plotting time t on the horizontal axis, vs $x(t)$ on the vertical.
  - A plot of all trjectories, plotting time t on the horizontal axis, vs $y(t)$ on the vertical.

Include labels on axes. There is no need to indicate units.

- Include a copy of your script in this report.
- You do not need to submit your script to Matlab Grader.

Include the plots in your report. Make sure you include titles or captions for your figures.

a) $y' = -\lambda y \quad , \quad y(0) = y_0 \quad , \quad \lambda = 0$

$$F_1 = hf(t_0, y_0)$$
$$= hf(0,0)$$
$$= -\lambda h y_0$$

$$F_2 = hf(t_0 + \tfrac{1}{2}h, y_0 + \tfrac{1}{2}F_1)$$
$$= hf(0 + \tfrac{h}{2}, 0 + \tfrac{F_1}{2})$$
$$= -\lambda h(y_0 - \frac{\lambda h y_0}{2})$$

$$F_3 = hf(t_0 + h, y_0 - F_1 + 2F_2)$$
$$= hf(t_0 + h, y_0 + \lambda h y_0 - 2\lambda h(y_0 - \tfrac{1}{2}\lambda h y_0))$$
$$= -\lambda h(y_0 + \lambda h y_0 - 2\lambda h(y_0 - \tfrac{1}{2}\lambda h y_0))$$
$$= -\lambda h y_0(1 + \lambda h - 2\lambda + \lambda^2 h^2$$
$$= -\lambda h y_0(1 - \lambda h + \lambda^2 h^2)$$

$$y_{k+1} = y_k + \tfrac{1}{6}[F_1 + 4F_2 + F_3]$$

$$y_1 = y_0 + \tfrac{1}{6}[-\lambda h y_0 + y_0(\frac{\lambda^2 h^2}{2} - \lambda h) - \lambda h y_0[1 - \lambda h + \lambda^2 h^2]]$$
$$= y_0 + \tfrac{1}{6}[-\lambda h y_0 + y_0(2\lambda^2 h^2 - 4\lambda h) - \lambda h y_0[1 - \lambda h + \lambda^2 h^2]]$$
$$= y_0 + \frac{\lambda h y_0}{6}[-6 + 3\lambda h - (\lambda h)^2]$$
$$= y_0[1 - \lambda h + \tfrac{1}{2}\lambda^2 h^2 - \tfrac{1}{6}\lambda^3 h^3]$$

$\therefore \; y(h) = y_0 e^{-\lambda h}$

b) $y(h) = y_0[1 - \lambda h + \tfrac{1}{2}\lambda^2 h^2 - \tfrac{1}{6}\lambda^3 h^3 + \tfrac{1}{24}\lambda^4 h^4 - \tfrac{1}{120}\lambda^5 h^5] + O(h^6)$

where approx. value for y_1 is the first four terms in the exact solution. (As highlited green above) $\therefore$ the error is the same as the first unidentical term (highlited red) which is $O(h^4)$.

c) for $(k+1)$th step

$$y_{k+1} = y_k[1 - \lambda h + \tfrac{1}{2}\lambda^2 h^2 - \tfrac{1}{6}\lambda^3 h^3]$$

assume $y_k = A\sigma^k \;\therefore\; y_0 = A\sigma^0 \;\therefore\; A = y_0$

$$y_k = y_0[1 - \lambda h + \tfrac{1}{2}\lambda^2 h^2 - \tfrac{1}{6}\lambda^3 h^3]^k$$

for $\lim_{t \to \infty} y(t) = 0$

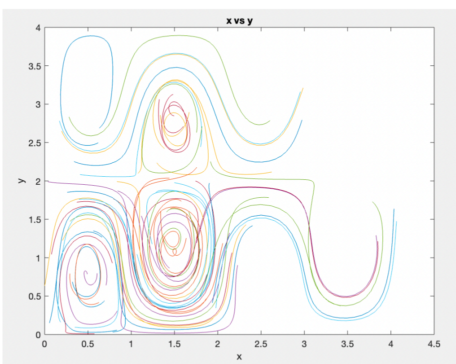
$$|1 - \lambda h + \tfrac{1}{2}\lambda^2 h^2 - \tfrac{1}{6}\lambda^3 h^3] < 1$$

$\therefore$ tried to plot but didnt work

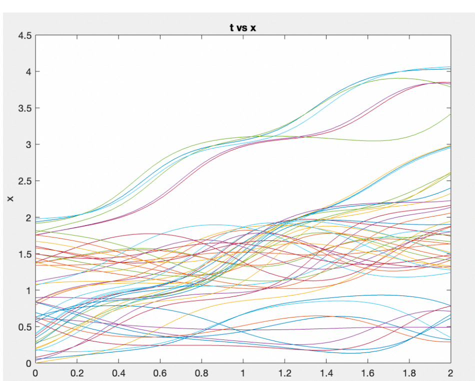
2 a) submitted to websub

b)

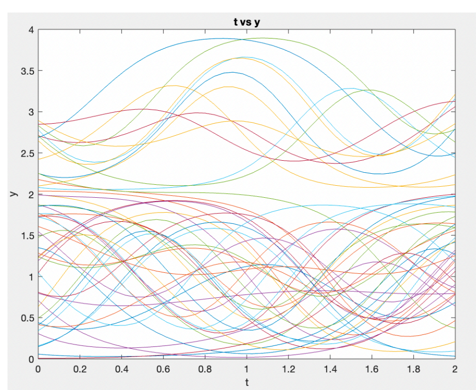
```
%Sarah Telford
%a1810750
%18.10.21 - started
%20.10.21 - finished (needed to fix rossby to take in  vectors)

%rossby model trajectories for large-scale atmospheric wind
clear
close  all
rng(1)
M = 50;

tSpan  =[0 2] ;
xy0 = [2* rand(M,1); 3*rand(M,1)];
dxydt = @rossby;
odeSolver = @ode45;

% Option  2:  simulate  using  vectorised  ode
[tn, xn] = odeSolver(dxydt,tSpan,xy0);

%get y values from xn starting at postition 1 and ending at position M
x1 = xn(:,1:M);

%get y values from xn starting at postition m+1 and ending at the end of the vector
y1 = xn(:,M+1:end);

% plot x vs y
figure(1)
plot(x1,y1);
xlabel("x");
ylabel("y");
title("x vs y")

% plot t vs x and t vs y on  subplots
figure(2)
plot(tn,x1)
xlabel("t");
ylabel("x");
title("t vs x");
figure(3)
plot(tn,y1)
xlabel("t");
ylabel("y");
title("t vs y");

function dxydt = rossby(t, xy)

    %parameters to change the vector locations
    x = xy(1:(end/2));
    y = xy(((end/2)+1):end);

    %parameter values - provided in tasksheet
    A=1; k1=pi; k2=1; l1=pi/2; l2=2; c1=1/2; c2=pi; e=0.3;

    %equation one - provided in tasksheet
    first = c1-A.*l1.*sin(k1.*x).*cos(l1.*y)-e.*l2.*sin(k2.*(x-(c2-c1).*t)).*cos(l2.*y);

    %equation two - provided in tasksheet
    second = A.*k1.*cos(k1.*x).*sin(l1.*y)+e.*k2.*cos(k2.*(x-(c2-c1).*t)).*sin(l2.*y);

    %output vector that evaluates the ODE
    dxydt = [first;second];

end
```



plot : height vs length of trajectory.



plot : length of trajectory vs time



plot : height of trajectory vs time