Ordinary Differential Equations
MATH 308 - 523
A. Bonito

Due Week of April 27
Spring 2015

Last Name(s): _____

Matlab Project

- Build a group of maximum 3 people.
- Everyone in the group will receive the same grade, independently of the participation in the project.
- A report should be prepared for the last week of class and returned during the oral discussion with the instructor where the entire group must be present. The time and date for the discussion will be determined in due time.
- By returning your project, you agree to follow the university code of academic integrity.

This project is based on the discussion in Section 2.9, 8.1, 8.2, 8.3 of the textbook. *Read them first!*
Denote by $[0,T]$ the time interval we are interested in solving the first order equation

$$\frac{d}{dt}y = f(t,y), \qquad y(0) = y_0.$$

Any numerical method start with a partition of $[0,T]$ into sub-intervals. For this, let N be a positive integer and define $h = T/N$ and $t_n = nh$ for $n = 0, ..., N$. This creates a *uniform* partition $0 = t_0 < t_1 < ... < t_N = T$ of $[0,T]$. The goal of numerical methods is to provide approximations Y^n of the exact solution $y(t_n)$, $n = 0, ..., N$.
The performance of each particular numerical algorithm is measured by computing (for instance)

$$\max_{n=0,...,N} |y(t_n) - Y^n|$$

and in particular how the above quantity behaves with N, the number of steps in the method.
For the Euler method, there exists a constant C independent of N and the solution such that

$$\max_{n=0,...,N} |y(t_n) - Y^n| \leqslant CN^{-1} \sup_x \left| \frac{d^2}{dx^2} y \right|,$$

provided the exact solution y is twice differentiable. This means that if the number of subintervals is doubled, then the error is divided by a factor 2. In general, a method is said to be of order r if

$$\max_{n=0,...,N} |y(t_n) - Y^n| \leqslant CN^{-r} \sup_x \left| \frac{d^{r+1}}{dx^{r+1}} y \right|,$$

provided y is $r+1$ times differentiable.
In practice, the validation of the implementation is performed as follow:

- Decide for a numerical method and derive an estimate of the type

$$e_N := \max_{n=0,\dots,N} |y(t_n) - Y^n| \leqslant CN^{-\alpha},$$

  for some $\alpha > 0$. The variable α is called the order of the method.

- Manufacture a non trivial exact solution $y(t)$, for instance, $y(t) = \cos(\pi t)e^{-3t}$.

- Deduce what should be $f(t, y)$ and y_0. In the example above, $f(t, y) = \frac{d}{dt}y = -(\pi \sin(\pi t) + 3\cos(\pi t))e^{-3t}$ and $y_0 = y(0) = 1$.

- Run your code with this $f(t, y)$ and y_0 and record e_N for different values of N (e.g. $N = 100, 200, 400, 800, ...$).

- Plot e_N versus N in a $log - log$ scale and check you have a straight line (for large N) with a slope of $-\alpha$. The matlab code for this last point would be something like:

```
loglog(arrayN,arrayErrors,arrayN,arrayN.^(-alpha));
legend('Errors','Expected decay');
```

  where $arrayN$ is an array with all the values of N considered, $arrayErrors$ is an array with the corresponding errors e_N and $alpha$ is the constant derived in the first step.

Exercise 1

We propose to test the Forward Euler method and start by providing its matlab implentation

```
%% FILE euler.m %%

function [X,Y]=euler(f,x0,xf,y0,N)
% Solve dy/dx = f(x,y)  y(x0)=y0
% for x0 <= x <= xf and with N number of steps
% X are the x coordinates
% Y are the solutions

clear Y;
clear X;


Y(1)=y0; % initial values
X(1)=x0;

h=(xf-x0)/N;  % increment

for i=2:N+1
Y(i) = Y(i-1)+h*f(X(i-1),Y(i-1)); % approximation
X(i)=X(i-1)+h;  %increment the position
end

%%% END FILE %%%
```

If you want to solve the ODE

$$\frac{d}{dx}y = 2y \quad \text{for } 0 < x \leqslant 2, \qquad y(0) = 1,$$

using 100 points, save the above text in a file named "euler.m" and execute the following commands in matlab

```
f = @(x,y) 2*y;
[X,Y]=euler(f,0,2,1,100);
plot(X,Y);
```

Warning: make sure that the file *"euler.m"* is in the working directory. This can be checked using the command *ls*.
Now we start the assignement and consider the following ODE

$$\frac{d}{dx}y = -y + 2\ x\ \cos(x^2)e^{-x} \quad \text{for } 0 < x \leqslant 2, \qquad y(0) = 0.$$

1. Check that the solution is given by
$$y(x) = e^{-x}\sin(x^2).$$
2. Modify the provided matlab code such that it computes the error
$$e_N := \max_{i=1,\ldots,N+1} |Y(i) - y(X(i))|.$$
3. run the code for $N = 100$, 200, 400, 800, 1600 and plot the errors e_N versus N in a log log plot together with the function $g(x) = x^{-1}$. Discuss your results.
4. Now consider a different ODE
$$\frac{d}{dx}y = f(x) \quad \text{for } 0 < x \leqslant 5, \qquad y(0) = 0,$$
where
$$f(x) := \begin{cases} 1, & 0 < x < \pi \\ 200\pi & x \geqslant \pi. \end{cases}$$
Repeat the steps 1-3 above (you will need to find analytically the exact solution).
Hint: in matlab, the function f is produced using

```
f=@(x,y) ...
( 0 ).* (x<=0) + ...
( 1 ).* (0<=x & x<pi) + ...
( 200*pi ) .* (x>=pi);
```

Exercise 2

We propose to implement a Heun method to solve the following generic first order ODE

$$\frac{d}{dx}y = f(x,y), \qquad y(x_0) = y_0$$

for $x \in (x_0, x_f)$.
Let N any positive integer and define $h := (x_f - x_0)/N$. The following numerical algorithm yield approximation Y_i of y at $X_i = x_0 + (i-1)\ h$ for $i = 1, .., N+1$:

$x \leftarrow x_0$
$y \leftarrow y_0$
$X_0 \leftarrow x$
$Y_0 \leftarrow y$

FOR $i = 2, ..., N+1$
  $k_1 \leftarrow f(x, y)$
  $k_2 \leftarrow f(x + h, y + h\ k_1)$
  $x \leftarrow x + h$
  $y \leftarrow y + \frac{h}{2}(k_1 + k_2)$
  $X_i \leftarrow x$
  $Y_i \leftarrow y$
END FOR

1. Implement the above Heun algorithm including the computation of the error

$$e_N := \max_{i=1,\dots,N+1} |Y_i - y(X_i)|.$$

2. Run your code for $N = 100, 200, 400, 800, 1600$ and plot the errors e_N versus N in a log log plot when $f(x, y) = -y + 2x\cos(x^2)e^{-x}$, $x_0 = y_0 = 0$, $x_f = 2$ and thus $y(x) = e^{-x}\sin(x^2)$.

3. Estimate $\alpha \in \mathbb{R}$ such that for some constant C independent of N there holds

$$e_N \leqslant CN^{-\alpha}.$$

4. Discuss your results. In particular, compare with the Euler method introduced in the previous assignement.

Exercise 3

We now propose to implement a modified Heun method which select adaptively the time step h at each stage instead of being constant during the entire computation. Again, we focus on the solution to

$$\frac{d}{dx}y = f(x, y), \qquad y(x_0) = y_0$$

for $x \in (x_0, x_f)$.
The method works as follow. Given an approximation Y_i at X_i and a step size h_i, we run simultaneously *one step* of the Heun and Euler methods to get two approximations:

$$Y_{i+1}^H := Y_i + \frac{h_i}{2}\left(f(X_i, Y_i) + f(X_i + h_i, Y_i + h_i k_1)\right),$$

for the Heun approximation and

$$Y_{i+1}^E := Y_i + h_i f(X_i, Y_i)$$

for the Euler approximation.
The following criteria is proposed to define h_{i+1} and decide whether Y_{i+1}^H is good enough or if the computation should be rejected, thereby restarting from Y_i with a modified h_i. Given a relative tolerance Rel and absolute tolerance Abs define

$$err = \max(|Y_i|, |Y_{i+1}^H|)\mathrm{Rel} + \mathrm{Abs}$$

and the ratio

$$r = \frac{|Y_{i+1}^H - Y_{i+1}^E|}{err}.$$

The latter is expected to satisfy

$$r \approx h_i^2$$

because the Heun method is globally second order while the Euler method is globally first order. Hence, the heuristic "optimal" step size is chosen as

$$h_{opt} = h_i \min\left(\frac{3}{2}, \max\left(\frac{1}{2}, \frac{8}{10}r^{-\frac{1}{2}}\right)\right);$$

where the 1.5 is to make sure the step size does not increase too much in one step. In opposition the $\frac{1}{2}$ coefficient prevents the step size to be reduced by more than half, while $\frac{8}{10}r^{-\frac{1}{2}}$ is designed so that the next step will not be rejected with high probability. Finally the solution is $Y_{i+1} = Y_{i+1}^H$ is accepted if $r \leqslant 1$ and $h_{i+1} = h_{opt}$. The solution is rejected and the algorithm restarts at X_i with $h_i = h_{opt}$.
Accordingly, the following numerical algorithm yields approximations Y_i of y at X_i starting with a initial mesh size of h_0.

$h \leftarrow h_0$

$x \leftarrow x_0$
$y \leftarrow y_0$
$X_0 \leftarrow x$
$Y_0 \leftarrow y$

i=1
While $(X_{i-1} \leqslant x_f)$
$k_1 \leftarrow f(X_{i-1}, Y_{i-1})$
$k_2 \leftarrow f(X_{i-1} + h, Y_{i-1} + h\ k_1)$
$Y_i^H \leftarrow Y_{i-1} + h/2(k_1 + k_2)$
$Y_i^E \leftarrow Y_i + h\ k_1$
$r = \frac{|Y_i^H - Y_i^E|}{\max(|Y_{i-1}|, |Y_i^H|)\text{Rel}+\text{Abs}}$
if $(r \leqslant 1)$ then $X_i = X_{i-1} + h$ and $Y_i = Y_i^H$ and $i \leftarrow i + 1$
end if
$h \leftarrow h \min\left(\frac{3}{2}, \max\left(\frac{1}{2}, \frac{8}{10} r^{-\frac{1}{2}}\right)\right)$
END While

1. Implement the above algorithm including the computation of the error

$$e_N := \max_{i=1,\ldots,N} |Y_i - y(X_i)|.$$

   where N is the number of accepted iterations within the while loop.

2. Run your code for $Rel = Abs = 0.01 \times 2^{-i}$, $i = 0, ..., 5$ and provide the log-log plot for each of the following to ODEs

   (a)
$$\frac{d}{dx} y = -y + 2\ x\ \cos(x^2) e^{-x} \quad \text{for } 0 < x \leqslant 2, \qquad y(0) = 0.$$

   (b)
$$\frac{d}{dx} y = f(x) \quad \text{for } 0 < x \leqslant 5, \qquad y(0) = 0,$$

   where
$$f(x) := \begin{cases} 1, & 0 < x < \pi \\ 200\pi, & x \geqslant \pi. \end{cases}$$

3. For comparison, run the basic Heun method on the same test cases with N steps where N is for each case the number of accepted steps by the adaptive method. Compare the log-log plots.

From now on, the adaptive Heun method is used. However, you will need to extend the above algorithm to solve *systems* of ODEs. Think carefully on how to perform this task and test your extended algorithm on manufactured solutions.

Exercise 4

We consider the mass-spring system

$$y''(t) + 0.1y'(t) + y(t) = \sin(\omega t)$$

with $y(0) = y'(0) = 0$.
Write the above second order ODE as a system of two first order ODEs and find (numerically) the value of ω leading to the highest amplitude. How does the computed predictions compare the theoretical values for the optimal frequency ω and resulting amplitude R? Recall that in this case

$$\omega = \sqrt{0.995}, \qquad R \approx 10.0125 \quad .$$

Exercise 5

We want to see how difficult it is to predict weather forecast. A simple model for the water prediction is given by the Lorentz equations relating critical component of the atmosphere

$$\begin{aligned}\frac{dx}{dt} &= 10(-x+y)\\ \frac{dy}{dt} &= 28x - y - xz\\ \frac{dz}{dt} &= -\frac{8}{3}z + xy.\end{aligned}$$

- Extend your implementation and chose a non trivial exact solution to check you obtain the correct decay for the error.
- Solve the Lorentz system for $t \in (0, 21)$ with $x(0) = y(0) = z(0) = 5$. Plot the evolution of x vs t and the phase portrait (enjoy the butterfly) $y(t)$ vs $x(t)$ for different targeted accuracy. Discuss your results.
- Assume that there is a slight different initial condition, say $x(0) = 5.01$, $y(0) = z(0) = 5$. Perform the same computations in this context and discuss the differences. This is called the butterfly effect (chaotic behavior). Included in your discussion until what time you estimate you can trust your predictions and justify why. Discuss how this numerical experiment relates to weather prediction.