MATH 2271 - Ordinary Differential Equations

Semester II - 2017/2018

Lab #2 - Euler's Method

1 Solving a 1st order ODE

1.1 Problem

1. Write a MATLAB function to find and make a plot of the numerical solution of the general first order initial value problem:
$$\frac{dy}{dx} = f\left(x, y\right), \qquad y\left(a\right) = y0,$$
over the interval $[a, b]$, using Euler's method, with N points.

2. Consider the IVP
$$\frac{dy}{dx} = x + y - 1, \qquad y\left(0\right) = 2, \qquad x \in \left[0, 0.75\right].$$
Use Euler's method to obtain a plot of the numerical solution of this problem over the interval $\left[0, 0.75\right]$, using 501 points.

3. Consider the IVP
$$\frac{dy}{dt} = 2 - ty, \qquad y\left(0\right) = 1, \qquad t \in \left[0, 2\right].$$
Use Euler's method to obtain a plot of the numerical solution of this problem over the interval $\left[0, 2\right]$, using 1001 points.

1.2 MATLAB Implementation

1. **Code for Euler's Method (for a single 1st order ODE):**

```
function [x,y] = Euler(a, b, N, y0, f)
% MATLAB code for using Euler's Method to solve the general 1st order IVP
% dy/dx=f(x,y) on the interval [a,b] with  y(a)=y0

% Inputs:   a and b define the interval [a,b]
%           N is the number of points
%           y0 is the value of y(a)
%           f is the function f(x,y) which must be defined separately

% Outputs:  x stores the discretized values of x in the interval [a,b]
%           y stores the corresponding approximate values of the solution

h=(b-a)/(N-1);  % determines the step-size needed
x=linspace(a,b,N);  % discretizes the interval for x
y=zeros(1,N);   % initializes y as a vector of zeros
y(1)=y0;    % stores the value of y(a) as the first entry in the y vector
for i=1:N-1   % using a for loop to compute the remaining y values
    y(i+1)=y(i)+h*feval(f,x(i),y(i));   % Algorithm for Euler's Method
end
plot(x,y)   % displays a plot of y versus x
end
```

```
function [x,y] = Euler(a, b, N, y0, f)
h=(b-a)/(N-1);
x=linspace(a,b,N);
y=zeros(1,N);
y(1)=y0;
for i=1:N-1
y(i+1)=y(i)+h*feval(f,x(i),y(i));
end
plot(x,y)
end
```

2. **Code for the function f:**

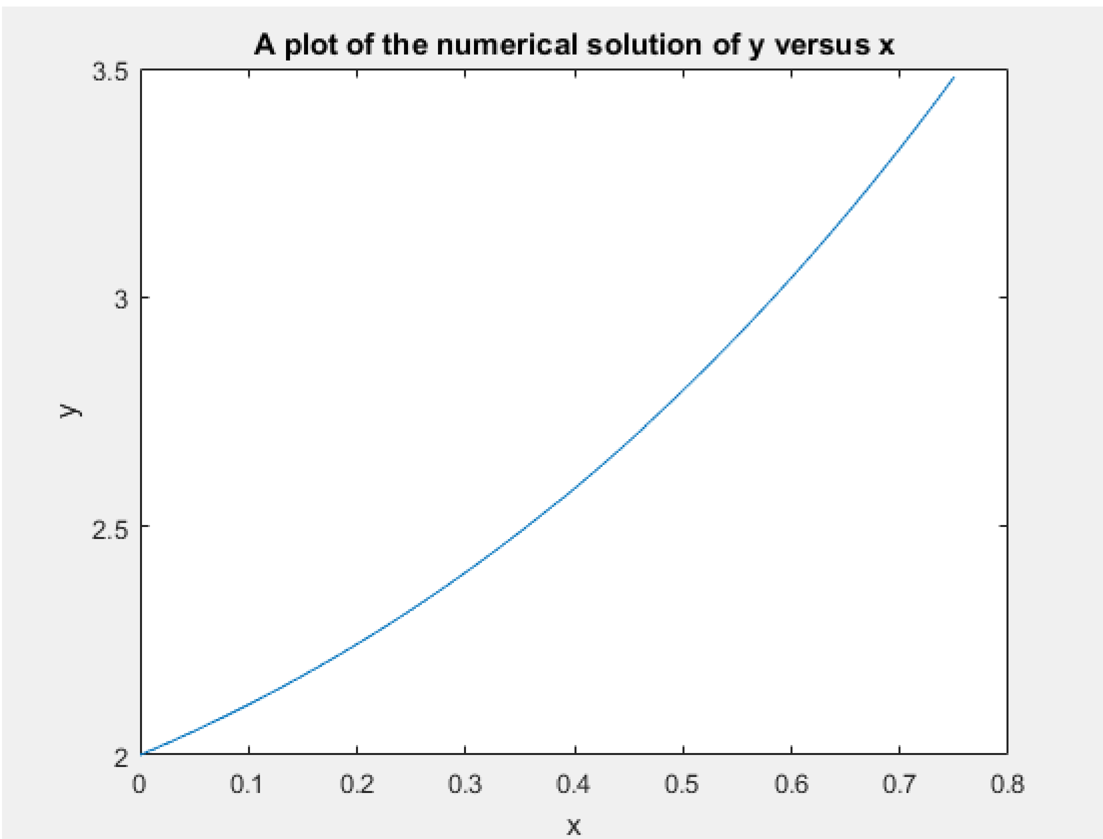
   Note that the function f is defined here as $f(x, y) = x + y - 1$.

```
function [u] = f( x,y )
u=x+y-1;
end
```

Command Window:

```
>> [x,y]=Euler(0, 0.75, 501, 2, 'f');
>> xlabel('x')
>> ylabel('y')
>> title('A plot of the numerical solution of y versus x')
```

Plot:



3. **Code for the function g:**

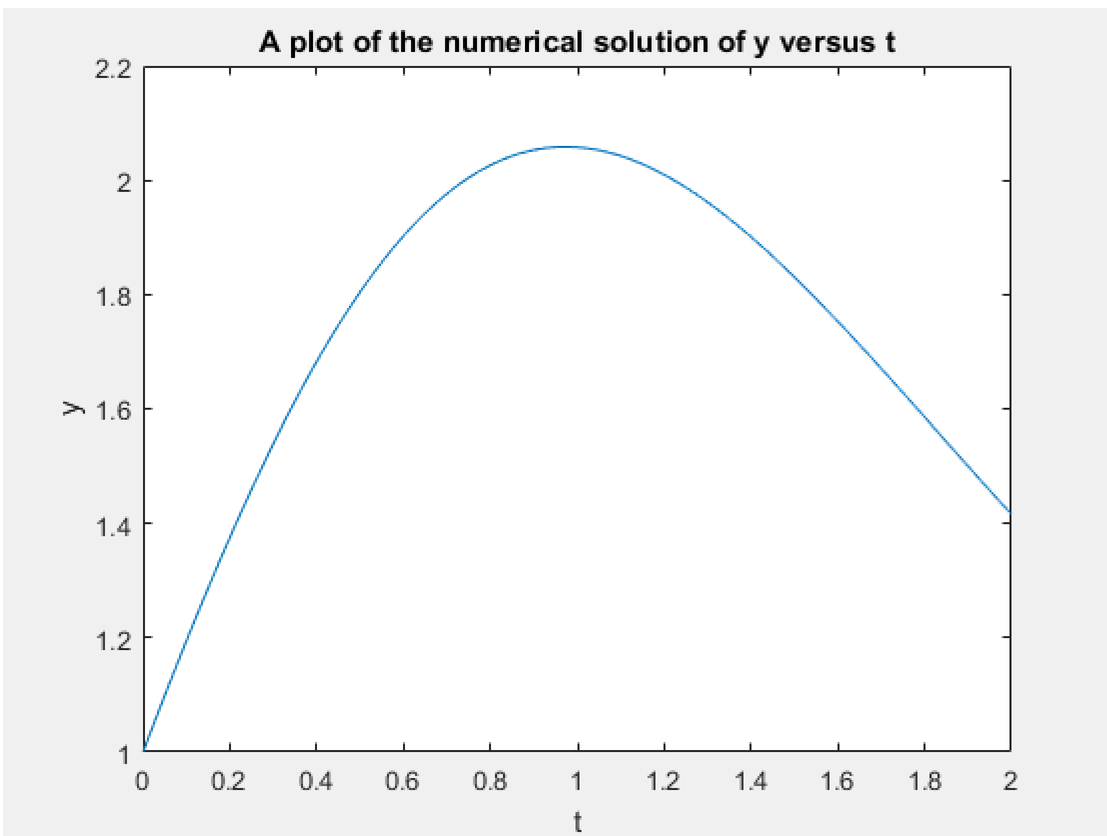
Note that the function here is defined as $f(t, y) = 2 - ty$. Since we already have a function **f** saved in the Current Folder, we shall name this function **g** instead.

```
function [u] = g( t,y )
u=2-t.*y;
end
```

Command Window:

```
>> [x,y]=Euler(0, 2, 1001, 1, 'g');
>> xlabel('t')
>> ylabel('y')
>> title('A plot of the numerical solution of y versus t')
```

Plot:



2 Solving a system of two coupled 1st order ODEs

2.1 Problem

1. Write a MATLAB function to find and make a plot of the numerical solution of the first order initial value problem:
$$\frac{du}{dx} = f\left(x, u, v\right), \qquad u\left(a\right) = u0,$$
$$\frac{dv}{dx} = g\left(x, u, v\right), \qquad v\left(a\right) = v0,$$
over the interval $[a, b]$, using Euler's method, with step size h.

2. Consider the IVP
$$\frac{du}{dx} = 2v - 1, \qquad u\left(1\right) = 2,$$
$$\frac{dv}{dx} = v - u + x^2, \qquad v\left(1\right) = 2,$$
where $x \in [1, 5]$. Use Euler's method to obtain a plot of the numerical solution of this problem over the interval $[1, 5]$, using step size $h = 0.001$.

2.2 MATLAB Implementation

1. **Code for Euler's Method (for a coupled system of two 1st order ODEs):**

```
function [ x,u,v ] = Euler2( a,b,h,u0,v0,f,g )
% MATLAB code for using Euler's Method to solve two 1st order ODEs
% on the interval [a,b] with given initial conditions

% Inputs:   a and b define the interval [a,b]
%           h is the step size
%           u0 and v0 are the values of u(a) and v(a)
%           f is the function f(x,u,v), defined separately
%           g is the function g(x,u,v), defined separately

% Outputs:  x stores the discretized values of x in the interval [a,b]
%           u and v store the corresponding approximate solution values


x=a:h:b;     % discretizes the interval for x
N=length(x);    % determines the number of points
u=zeros(1,N);   % initializes u
v=zeros(1,N);   % initializes v
u(1)=u0; v(1)=v0;    % defines the initial values of u and v
for i=1:N-1     % using a for loop to compute the remaining values
    u(i+1)=u(i)+h*feval(f,x(i),u(i),v(i));  % Euler's method on u
    v(i+1)=v(i)+h*feval(g,x(i),u(i),v(i));  % Euler's method on v
end
plot(x,u,x,v)    % displays an overlaid plot of u and v vs. x
end
```


```
function [ x,u,v ] = Euler2( a,b,h,u0,v0,f,g )
x=a:h:b;
N=length(x);
u=zeros(1,N);
v=zeros(1,N);
u(1)=u0; v(1)=v0;
for i=1:N-1
u(i+1)=u(i)+h*feval(f,x(i),u(i),v(i));
v(i+1)=v(i)+h*feval(g,x(i),u(i),v(i));
end
plot(x,u,x,v)
end
```

2. **Code for the functions f1 and g1:**

   Note that the functions here are defined as $f(x,u,v) = 2v - 1$ and $g(x,u,v) = v - u + x^2$. Since we already have function **f** and **g** saved in the Current Folder, we shall name these functions **f1** and **g1** instead.

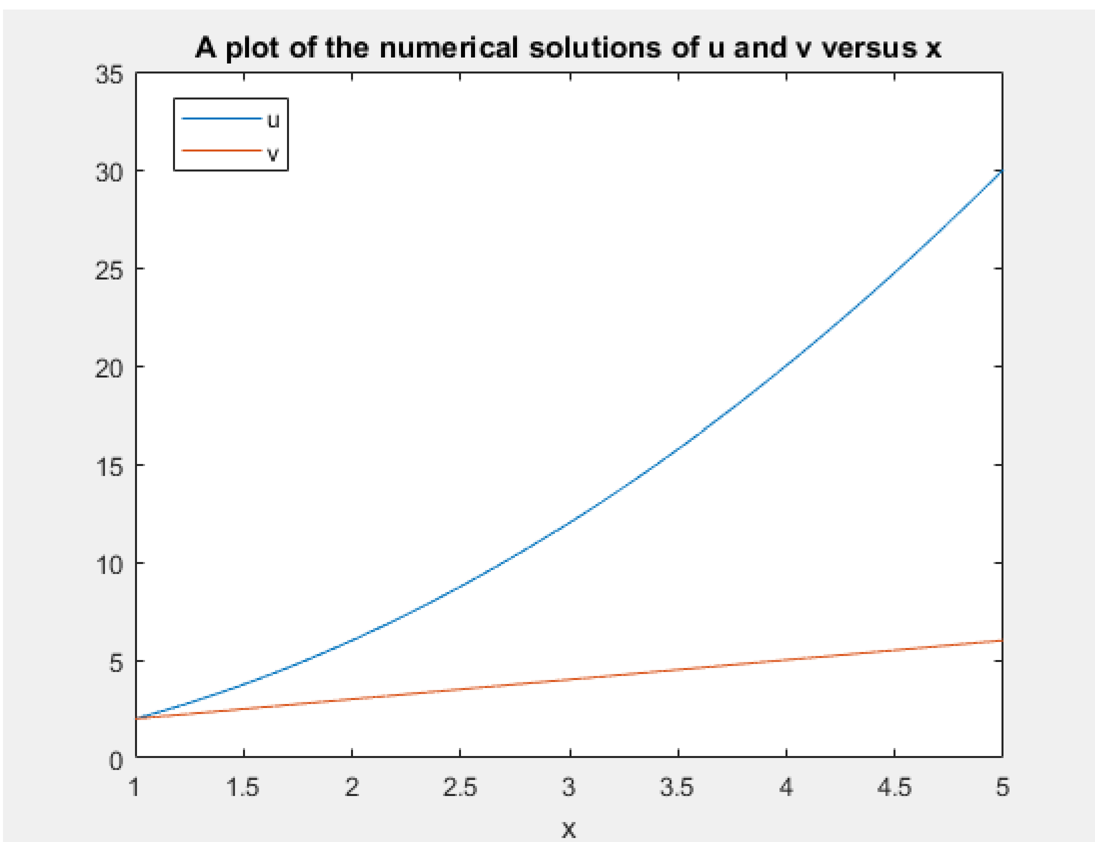
```
function [ t ] = f1( x,u,v )
t=2*v-1;
end
```

```
function [ t ] = g1( x,u,v )
t=v-u+x.^2;
end
```

Command Window:

```
>> [ x,u,v ] = Euler2( 1, 5, 0.001, 2, 2, 'f', 'g' );
>> xlabel('x')
>> legend('u','v')
>> title('A plot of the numerical solutions of u and v versus x')
```

Plot:



3 Solving a 2nd order ODE

3.1 Problem

1. Write a MATLAB function to find and make a plot of the numerical solution of the general second order initial value problem:

$$\frac{d^2y}{dx^2} = f\left(x, y, \frac{dy}{dx}\right), \qquad y\left(a\right) = y0, \qquad \frac{dy}{dx}\left(a\right) = u0,$$

over the interval $[a, b]$, using Euler's method, with step size h.

2. Consider the IVP

$$\frac{d^2y}{dx^2} = \frac{dy}{dx} - y + x^3, \qquad y\left(0\right) = -6, \qquad \frac{dy}{dx}\left(0\right) = 0,$$

where $x \in [0, 2\pi]$. Use Euler's method to obtain a plot of the numerical solution of this problem over the interval $[0, 2\pi]$, using step size $h = 0.001$.

Note:

A second order IVP can be solved by first converting it to a system of 2 first order ODEs, as follows:
First, let $\frac{dy}{dx} = u$.
This gives $\frac{d^2y}{dx^2} = \frac{d}{dx}\left(\frac{dy}{dx}\right) = \frac{du}{dx}$.
Note also that $u\left(a\right) = \frac{dy}{dx}\left(a\right) = u0$.
Therefore, we have 2 first order ODEs to solve:

$$\frac{dy}{dx} = u, \qquad y\left(a\right) = y0$$

$$\frac{du}{dx} = f\left(x, y, \frac{dy}{dx}\right), \qquad u\left(a\right) = u0$$

The algorithm for applying Euler's Method to solve for y is

$$y\left(x_{j+1}\right) \approx y\left(x_j\right) + h \cdot u\left(x_j\right), \quad j = 0, 1, \ldots N - 1.$$

The algorithm for applying Euler's Method to solve for u is

$$u\left(x_{j+1}\right) \approx u\left(x_j\right) + h \cdot f\left(x_j, y\left(x_j\right), u\left(x_j\right)\right), \quad j = 0, 1, \ldots N - 1.$$

3.2 MATLAB Implementation

1. Code for Euler's Method (for 2nd order ODE):

```
function [x,y] = Euler3(a, b, h, y0, u0, f)
% MATLAB code for using Euler's Method to solve a general 2nd order IVP

% Inputs:   a and b define the interval [a,b]
%           h is the step size
%           y0, u0 are the values of y(a) and dy/dx(a)
%           f is the function f(x,y,dy/dx), defined in another file

% Outputs:  x stores the discretized values of x in the interval [a,b]
%           y stores the corresponding approximate values of y

x=a:h:b;  % discretizes the interval for x
N=length(x); % determines the number of points
y=zeros(1,N);   % initializes y as a vector of zeros
u=zeros(1,N);   % initializes u as a vector of zeros
y(1)=y0;    % stores the value of y(a) as the first entry in y
u(1)=u0;    % stores the value of dy/dx(a) as the first entry in u
for i=1:N-1   % using a for loop to compute the remaining y and u values
    y(i+1)=y(i)+h*u(i);  % Euler's method on y
    u(i+1)=u(i)+h*feval(f,x(i),y(i),u(i));  % Euler's method on u
end
plot(x,y)   % displays a plot of y versus x
end
```

```
function [x,y] = Euler3(a, b, h, y0, u0, f)
x=a:h:b;
N=length(x);
y=zeros(1,N);
u=zeros(1,N);
y(1)=y0;
u(1)=u0;
for i=1:N-1
y(i+1)=y(i)+h*u(i);
u(i+1)=u(i)+h*feval(f,x(i),y(i),u(i));
end
plot(x,y)
end
```

2. **Code for the function h:**

   Note that the function f is defined here as $f(x, y, u) = u - y + x^3$. Since we already have a function **f** saved in the Current Folder, we shall name this function **h** instead.

```
function [ t ] = h( x,y,u )
t=u-y+x.^3;
end
```

Command Window:

```
>> [x,y] = Euler3(0, 2*pi, 0.001, -6, 0, 'h');
>> xlabel('x')
>> ylabel('y')
>> title('A plot of the numerical solution of y versus x')
```

Plot: