Lab 2: Solving Differential Equations via Euler's Method

Goals for Lab 2:

1. Implement Euler's Method of numerically solving a differential equation
2. Compare to an analytical solutions and quantify error

<u>Prelab – Part 1:</u> Introduction to Euler's Method

Over the course of ENGR 232, we will be solving many differential equations analytically and numerically. It therefore becomes important to see how a basic numeric differential equation solver works and gain an appreciation of the inherent error produced with them.

The **Euler's Method** was named after Leonhard Euler, and provides an intuitive approach to solving a first-order ordinary differential equation (ODE) given some initial value. Consider a function ***y(t)***. We can estimate its first derivative, ***y'(t)***, over some small increment of time Δt, as the following:

$$y'(t) \approx \frac{y(t+\Delta t) - y(t)}{\Delta t}$$

By the formal definition of the derivative, we know that as the time step Δt get smaller the approximation shown above should get better. Doing some simple rearrangement yields the following:

$$\boxed{y(t+\Delta t) \approx y(t) + y'(t)\Delta t}$$

Thus given some increment Δt and a differential equation ***y'(t) = f(y,t)***, we can iteratively build up the solution of the system (***y(t)***). The algorithm is thus as follows:

<u>**GIVEN:**</u> Differential equation ***y'(t)***, Initial condition $y(t_0)$**,** Time step Δt:

<u>**STEPS:**</u>

1. Substitute $y(t_0)$ and t_0 into the differential equation to solve for $y'(t_0)$
2. Use the boxed equation to solve for the next value, $y(t_0+\Delta t)$
3. Denote the next time value in the solution as $t_1 = t_0+\Delta t$
4. Repeat (1) by using $y(t_1)$ and t_1 with the differential equation to solve for $y'(t_1)$. Continue this process until the desired ending time.

For additional clarification, a block diagram is shown in Figure 1 illustrating the above steps.

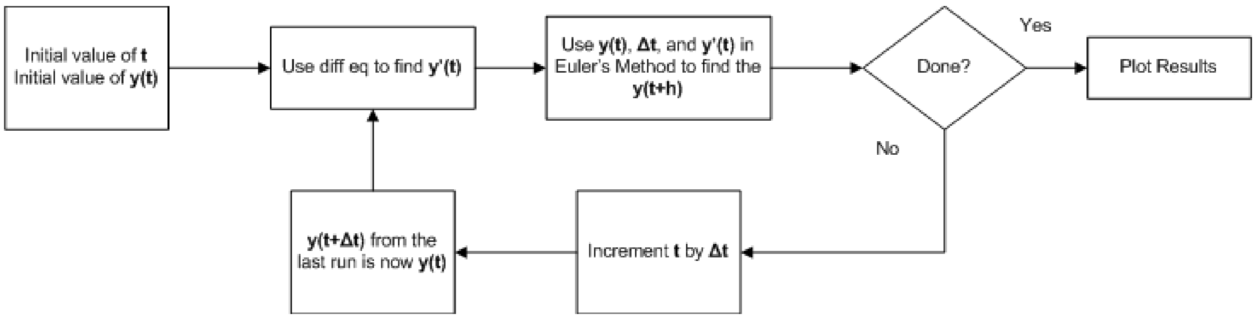

Initial value of t
Initial value of y(t)
Use diff eq to find y'(t)
Use y(t), Δt, and y'(t) in Euler's Method to find the y(t+h)
Done?
Yes
Plot Results
No
Increment t by Δt
y(t+Δt) from the last run is now y(t)

Figure 1. Block diagram of Euler's Method

<u>Prelab – Part 2:</u> Implementation Example of Euler's Method

We would like to numerically estimate the solution of the following initial value problem:

$$y' = -0.5y + 1.5 - t \qquad y(0) = 1$$

In this example, let ***Δt* = 0.5**. Table 1 shows the computations. Following the numbered list on the first page, we take t_0 and $y(t_0)$ and calculate $y'(t_0)$. We then substitute $y'(t_0)$ into the boxed equation to compute the next value, $y(t_0 + \Delta t) = y(t_1)$ and repeat.

Table 1. Calculated values of the differential equation of Part 2 via Euler's Method. The initial value is highlighted in yellow and recursive elements are colored accordingly. An iterative index 'k' is also shown.

k	**t**	**y(t)**	**y'(t)**	**y(t+Δt)**
1	**0.0**	**y(0) = 1**	y'(0) = 1.0	y(0.5)=1.5
2	0.5	y(0.5) = 1.5	y'(0.5) = 0.25	y(1.0) = 1.625
3	1.0	y(1.0) = 1.625	y'(1.0) = -0.3125	y(1.5) = 1.469
…	…	…	…	…
numel(t)	4.5	y(4.5) = -2.451	y'(4.5) = -0.734	y(5.0) = -3.338

The easiest way to implement this in MATLAB is via the use of a for loop. The overall approach is shown in the code below and the resulting figure is shown on the next page.

<u>Ex 1:</u>

```
%% ENGR 232: Lab 2, Prelab Example 1
% First we pick a step size and an initial value
dt = 0.5; yI = 1; tI = 0;

% Now let's make the entire range of t we wish to evaluate.
% Note that the first value in here is the initial value of t.
tEnd = 5;
t = tI:dt:tEnd;

% The values of y should be an array the same size as t, so we can plot it.
y = zeros(1,numel(t));
y(1) = yI;              %Set the initial value into the y array
```

```
% Now we can implement this all using a for loop
% In each run of the for loop, we'll calculate the next value of y(t). We'll
% start the indexing variable at k = 2 since the second position is the very
% first value we'll need to calculate (i.e. y(t_1)).

% k - Position/Index for y(t_(k-1)) and t_(k-1) (i.e. k = 2 is y(t_1))
% y(k) and t(k) - Current y and t values (Remember indexing rules from 231!)
% y(k-1) and t(k-1) - Previous y and t values
% yPrime - Current value of the derivative

% This produces the following:
for k = 2:numel(t)
    yPrime = -0.5*y(k-1) + 1.5 - t(k-1);
    y(k) = y(k-1) + dt*yPrime;
end

% Now plot the final solution
plot(t,y)
grid on
title(['ENGR 232: Lab 2 Prelab Example, dt = ' num2str(dt)])
xlabel('Time')
ylabel('y(t)')
```

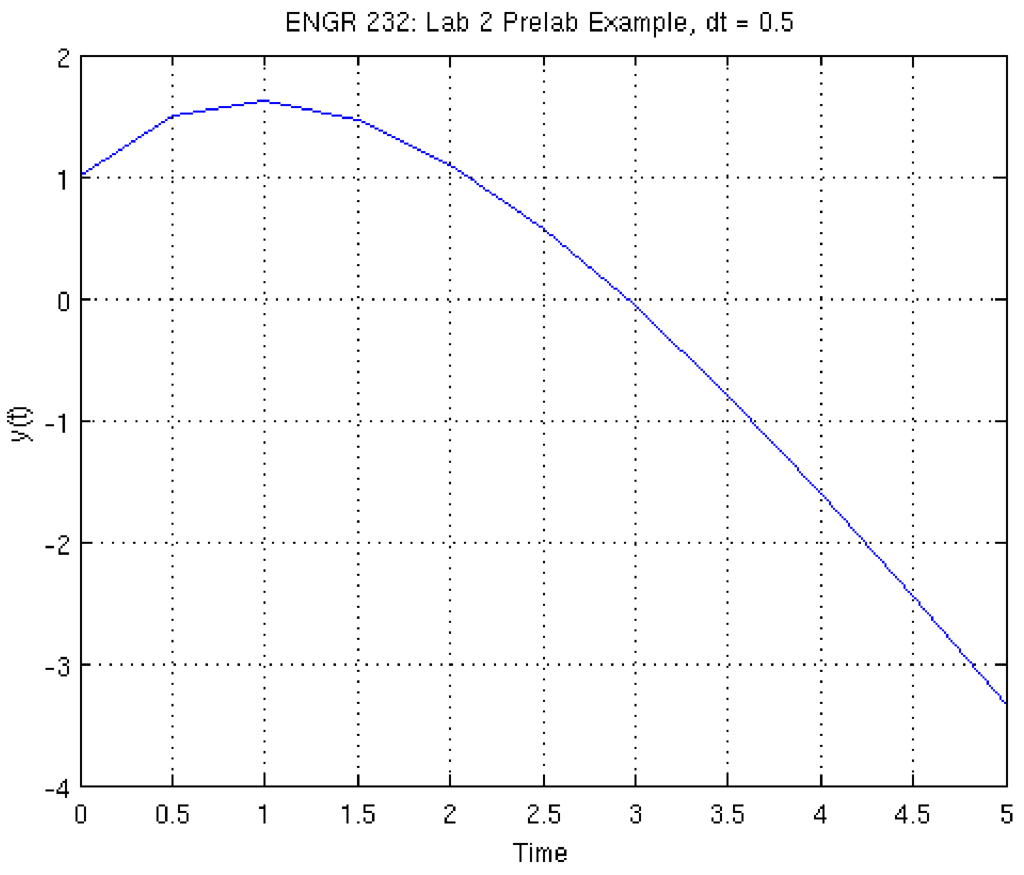


Figure 2. Resulting plot from Ex 1.

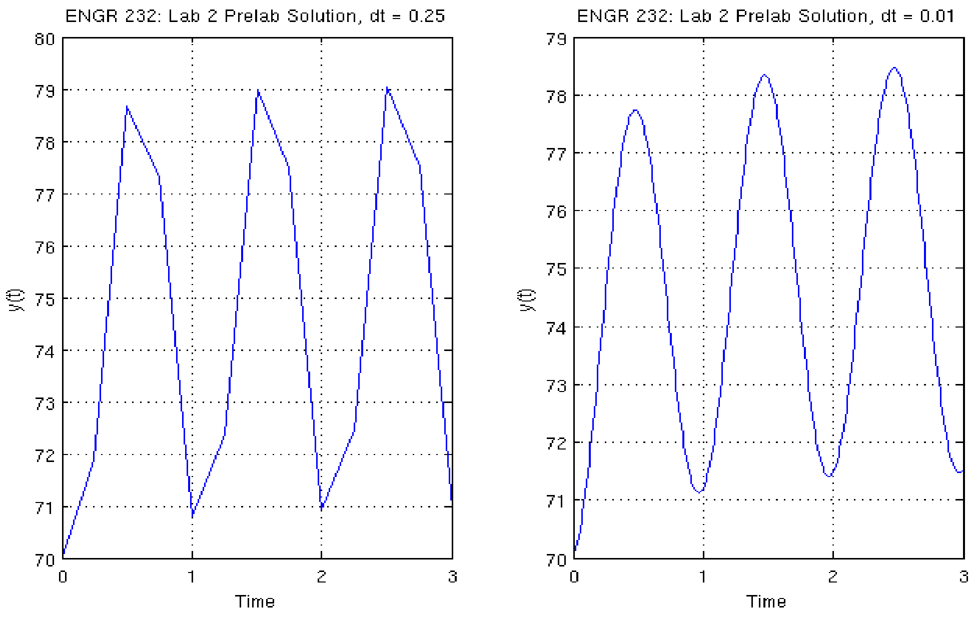
Prelab Assignment:

Write a script that uses Euler's method to solve the equation:

$$\frac{du}{dt} = -k[u - T(t)]$$

with k = 1.5, $T(t) = 75 + 15\sin 2\pi t$, and $u(0)$ = 70. Evaluate the result for $0 \le t \le 3$ and Δt = 0.25 and 0.01. ***Display your results with a 1 x 2 subplot (see MATLAB help for how subplots work).*** Annotate all plots with appropriate titles, xlabels, and ylabels.

EXPLAIN YOUR FINDINGS! Make sure that you follow the appropriate requirements for publishing your prelab results in a cell mode script. If done correctly, your plot should resemble the following:



In Class Exercise: Comparing to an analytical solution

Suppose now that we would like to quantify the error between our numerical solution and the analytical solution of a differential equation. Let's go back to our example used in the prelab. The analytic solution of the differential equation:

$$y' = -0.5y + 1.5 - t \qquad y(0) = 1$$

is known to be:

$$\boxed{y(t) = 7 - 6e^{-0.5t} - 2t}$$

Now we can take the same code from Example 1 and modify it such that we evaluate the boxed equation over the same interval of time. This way, we will have a one-to-one matching in time as to what our solver estimated the solution values to be and what they really are.

Finally, we would like to quantify the error between the analytic solution and numerical approximation with a single number. In ENGR 231's Lab 8, we used the ***MSE*** and provided formal definition. This quantity can be obtained however via a shortcut in MATLAB through the use of the data's mean and variance. With this, the MSE can be expressed as:

$$MSE = Var(y_{num} - y_{act}) + \left(mean(y_{num} - y_{act})\right)^2$$

Rather than hardcode this, we refer to the MATLAB functions `var()` and `mean()`[1]. All of this is shown below in example 2 with corresponding results in Figure 3.

<u>Ex 2:</u>

```matlab
%% ENGR 232: Lab 2, Prelab Example 2
%% Numerical Approximation
% First we pick a step size and an initial value
dt = 0.5;
yI = 1;
tI = 0;

tEnd = 5;
t = tI:dt:tEnd;
y = zeros(1,numel(t));
y(1) = yI;

% Calculate the numerical answer
for k = 2:numel(y)
    yPrime = -0.5*y(k-1) + 1.5 - t(k-1);
    y(k) = y(k-1) + dt*yPrime;
end

%% Analytical Solution
% Evaluate yA over the same range of t from above, using the dot operator if
% necessary
yA = 7-6*exp(-0.5*t)-2*t;

%% Final Results - MSE Error
MSE = var(y-yA)+mean(y-yA)^2

%% Final Results - Plot
plot(t,y,'r-.',t,yA,'b-')
title(['Lab 2 In Class Example, dt = ' num2str(dt) ', MSE = ' num2str(MSE)])
xlabel('Time'), ylabel('y(t)'), grid on
legend('Numerical Solution', 'Analytical Solution')
```

[1] Note that the var() function in MATLAB computes sample variance which is normalized by a factor of (n-1), where n is the size of the dataset. The true MSE requires that the variance term be normalized by a factor of n. For large datasets however, this difference becomes negligible. See the help file for variance for more information.

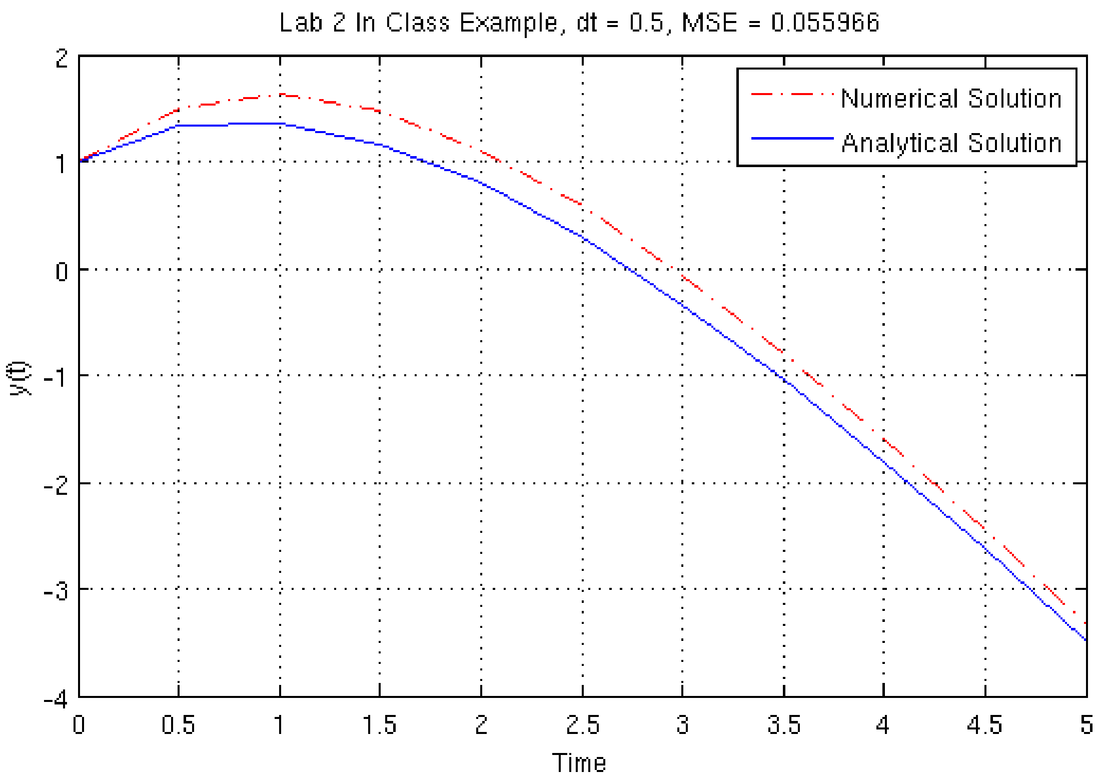


Figure 3. Results from Ex 2 of the lab. Notice differences between numerical and actual solutions.

In the prelab we introduced the idea of subplots as an alternative to overlaying multiple plots. In this instance, we see that it is better to try and overlay the plots of the estimated solution and actual solution for better comparison. Subplots tend to be more appropriate for grouping similar images together when direct comparison between them is not necessary.

Finally, we may also graph the error between the two lines. This is known as a ***residual plot*** and may be useful to help and pinpoint where inaccuracies in the estimation procedure occur.

Your lab instructor will distribute the in class assignment upon arriving to your lab section.

ENGR 232
Dynamic Engineering Systems

Lab 2 – Implementing and Evaluating Approximation Techniques via MATLAB
Verification Sheet

Student name____Section____________

1. **Prelab Assignment** – Implementation of Euler's Method (6 points total)
STAPLE YOUR PRELAB TO THIS VERIFICATION SHEET WHEN SUBMITTING!

Correct for loop usage (3 pts)	_______
Correct use of subplots (1 pts)	_______
Correct Final Plots (1 pts)	_______
Formatting Correct. Explanations sufficient (1 pts)	_______

Verified ____________________________Date/Time ______________

---

2. **In Class Assignment** – Comparing to an analytical solution (4 points total)

Correct Numerical Solutions (2 pts)	_______
Correct Residual Plots (1 pts)	_______
Correct MSE values (1 pts)	_______

Verified ____________________________Date/Time ______________

TOTAL POINTS __________ / 10 pts