Lab 3: Integrating and Differentiating Continuous-Time Signals in MATLAB

Habib University
CE-251L /EE-252L Signal and Systems
Prepared by: Dr. Shafayat Abrar and Ms. Hira Mustafa

Objective: Evaluating integrals and derivatives of signals using symbolic and numerical methods.

Name: Basil khowaja bk08432
Shahmir chauhdry 08400

In-Lab:
Part I: Integrating a signal using symbolic and numerical tools.
Part II: Differentiating a signal using symbolic and numerical tools.
Post Lab:
Part III: Integrating and differentiating basic signal functions like unit-impulse and unit-step.

***Instructions**: If you are doing any analysis, calculation or sketching on paper then attach a **scanned image** of the picture taken; you can either use an application named CamScanner or any other online application to attach a scanned version of the image. **Do not attach raw image only attach the scanned images** in case of any confusion you can ask from any of the RA.*

Useful tip: In MS Word *, in order to check/resolve formatting issues go to the Menu bar Paragraph section you will find* ¶ this symbol*, click on it this will indicate the start of new Paragraph, spaces, tabs, Section breaks and page Breaks.*

Part I: Symbolic and Numerical Integration of a Signal

Here, we learn how to integrate a signal (function) both symbolically and numerically in MATLAB.

Consider a signal (□) which is obtained as the time derivative of another signal (□),

$$\square(\square) = \frac{\square}{\square\square}\square(\square) \qquad (1)$$

We witness such relations in electrical and mechanical dynamic systems in a variety of manner. Consider the scenario of a linear electric circuit which contains an inductor and a capacitor. In inductor, the voltage (□) across it is directly proportional to the time rate of change of the current flowing through the inductor, □(□), and the constant of proportionality is the inductance

□, so

$$\square_{\square}(\square) = \square \frac{\square}{\square\square} \square_{\square}(\square)$$

Similarly, in a capacitor, the current through the capacitor (□) is equal to capacitance times the time rate of change of capacitor voltage □(□), giving

$$□_{□}(□) = □ \frac{□}{□□} □_{□}(□)$$

Suppose, the current flowing through the inductor is given as (where $f_0 \neq 0$)

$$(□) = □ \sin(2□□_0□)$$

The voltage across the inductor may easily be obtained as:

$$(□) = □ \frac{□}{□□} □(□) = □ \frac{□}{□□} □ \sin(2□□_0□) = 2□□_0□□ \cos(2□□_0□)$$

Using symbolic toolbox, (□) is obtained as follows:

```
syms A f0 t pi L
current = A*sin(2*pi*f0*t); voltage =
L*diff(current,'t')
```
giving
```
voltage = 2*A*L*f0*pi*cos(2*f0*pi*t)
```

On the contrary, given the voltage across the inductor, (□), the current flowing through the inductor is obtained as

$$i_L(t) = \frac{1}{□} \int_{-\infty}^{□} v_L(\lambda) d\lambda$$

or, if the initial (reference) time is given to be $□_0$, then

$$(t) = i(t_0) + \frac{1}{□} \int_{□_0}^{□} v_L(\lambda) d\lambda$$

where, $□_{□}(□_0)$ is termed as the initial value of the current, or what you may regard as the *constant of integration*. Note, the variable □ serves as a dummy variable in time. In simple, words, at a given time *t*, the value of inductor current $□_{□}(□)$ is obtained by integrating the inductor voltage from the beginning of time (the initial time, $□_0$) till the given time *t*, plus the initial value of current

$(□_0)$ which happens to depend on the initial condition of the inductor. In principle, $(□_0)$ is

$$(t_0) = \frac{1}{□} \int_{-\infty}^{□_0} v_L(\lambda) d\lambda$$

Reverting back to the initial consideration, as expressed in equation (1), given (□), where $(□) = \frac{□}{□□}(□)$, the signal (□) is obtained by taking integral or anti-derivative of (□):

$$(t) = \int_{-\infty}^{□} (\lambda)\lambda = y(t_0) + \int_{□_0}^{□} x(\lambda) d\lambda$$

If $(□) = 0$ for $□ < □_0$, then $(□_0) = \int_{-\infty}^{□_0} (□)□□ = 0$.

Consider $(t) = A\sin(2\pi f_0 t)$ for $0 \le t \le 3$, $A = 1$ and $f_0 = 1$. Let us integrate the signal (t) to obtain $y(t) = \int x(t)dt$ in MATLAB. Assuming that $(t) = 0$ for $t < 0$, we get $(0) = 0$.

Task (a) □ Solve the following integral by hand:

$$(t) = (0) + \int^{□} (\lambda)d\lambda.$$

$$y(t) = y(0) + A\int_0^t x(\lambda)\,d\lambda$$

$$= A\sin(0) + A\int_0^t \sin(\lambda)\,d\lambda \qquad \lambda = 2\pi f_0 t$$

$$= 0 + A\left[\cos\lambda\right]_0^t$$

$$= A\,2\pi f_0\left[\cos^{2\pi f_0} t - \cos 0\right] = \frac{A\,\cancel{2\pi f_0}}{2\pi f_0}\left[\cos 2\pi f_0 t - 1\right]$$

$$= \boxed{\frac{A\cos 2\pi f_0 t}{2\pi f_0} - \frac{A}{2\pi f_0}}$$

0

Task (b) □

Using symbolic tool box, try the following piece of code to integrate (□) to obtain (□):

```
syms A f0 t pi
xt = A*sin(2*pi*f0*t); yt =
int(xt,'t')
```

giving

```
yt =
-(A*cos(2*f0*pi*t))/(2*f0*pi)
```

To integrate (□) with limits from **0**to **t**, try the following code:

yt = int(xt,'t',[0,t])

Is the value of **yt**obtained above is same as that in task (a)? explain.

```
syms A f0 t pi
xt = A*sin(2*pi*f0*t);
yt = int(xt,'t')

yt = int(xt,'t',[0,t])
```

yt =

$$-\frac{A\cos(2\,f_0\,\pi\,t)}{2\,f_0\,\pi}$$

yt =

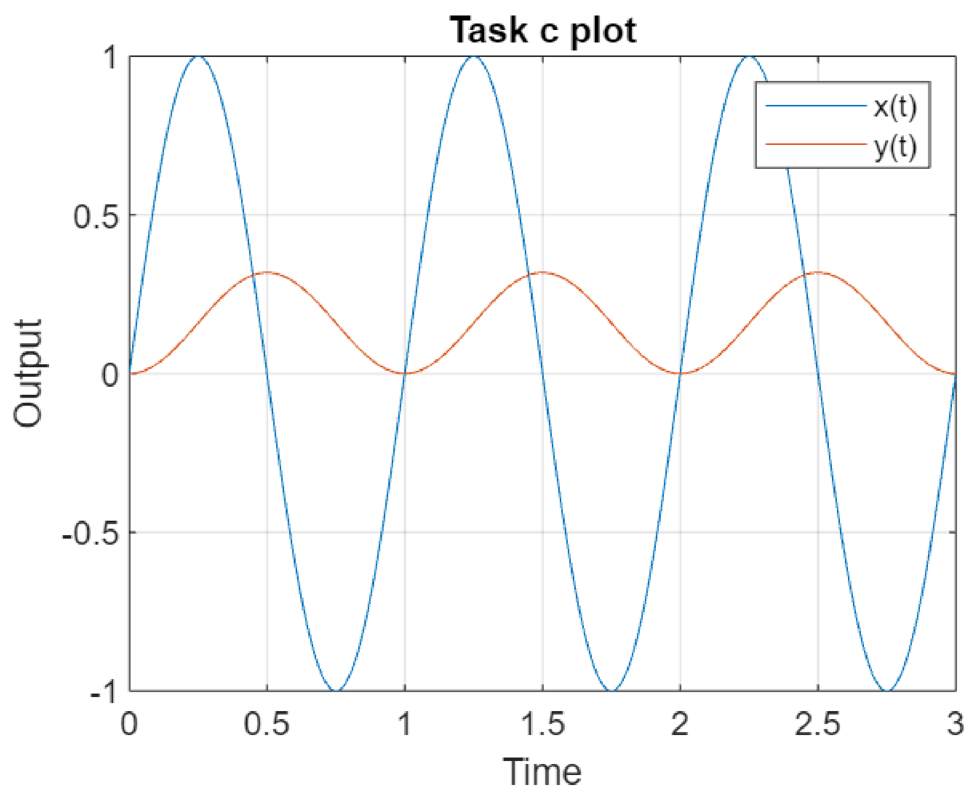
$$\frac{A\sin(f_0\,\pi\,t)^2}{f_0\,\pi}$$

The answer obtained is is excatly the one we obtained in task a. But the difference is that matlab has used an identity and then showed the asnwer.

Task (c) □ Write a MATLAB code to plot (t) and (t) for $0 \le t \le 3$. Select time step-size equal to 0.01 second. Obtain both plots on the same figure. Your code should make appropriate

```
clc
clear all
close all

t = 0:0.01:3;
xt = sin(2*pi*t);
yt = (sin(pi*t).^2)/pi;
plot (t, xt);
grid on;
hold on;
title ("Task c plot");
xlabel("Time")
ylabel("Output")
plot (t, yt)
legend ("x(t)", "y(t)")
```



In the following, we learn performing integration operation numerically in MATLAB. Owing to which, the signal (□) may be computed numerically without requiring to integrate (□) explicitly either manually or using some symbolic toolbox. The numerical integration is a *cumulative sum* operation where the numerical value of (□) at time □ is obtained as the cumulative sum of (□) from its initial value (0) till its value at time □. Conceptually, we have

$$(t) = (t_0) + \int_{\square_0}^{\square} (\lambda) d\lambda \approx y(t_0) + \mathrm{dt} \sum_{k=\square_0}^{\square} \mathrm{x}[\mathrm{k}]$$

This may easily be done in MATLAB using the built-in function **cumsum**as follows:

$$\mathrm{yt} = \mathrm{yt0} + \mathrm{dt} * \mathbf{cumsum}(\mathrm{xt}) \tag{2}$$

where **dt** is a (small) time step-size as assigned in MATLAB by the user, **yt0** is the initial value of $y(t)$ (if there is any), and **xt** denotes the discrete values of $x(t)$ as computed in MATLAB. This simplest form of numerical integration is called the rectangular or Euler's integration method.

Task (d) □ To understand **cumsum**, consider

a = [1 2 3 4 5];

try

b = **cumsum**(a)

What do you notice? Provide your comments below:

```
a = [1 2 3 4 5];
b = cumsum(a)
```

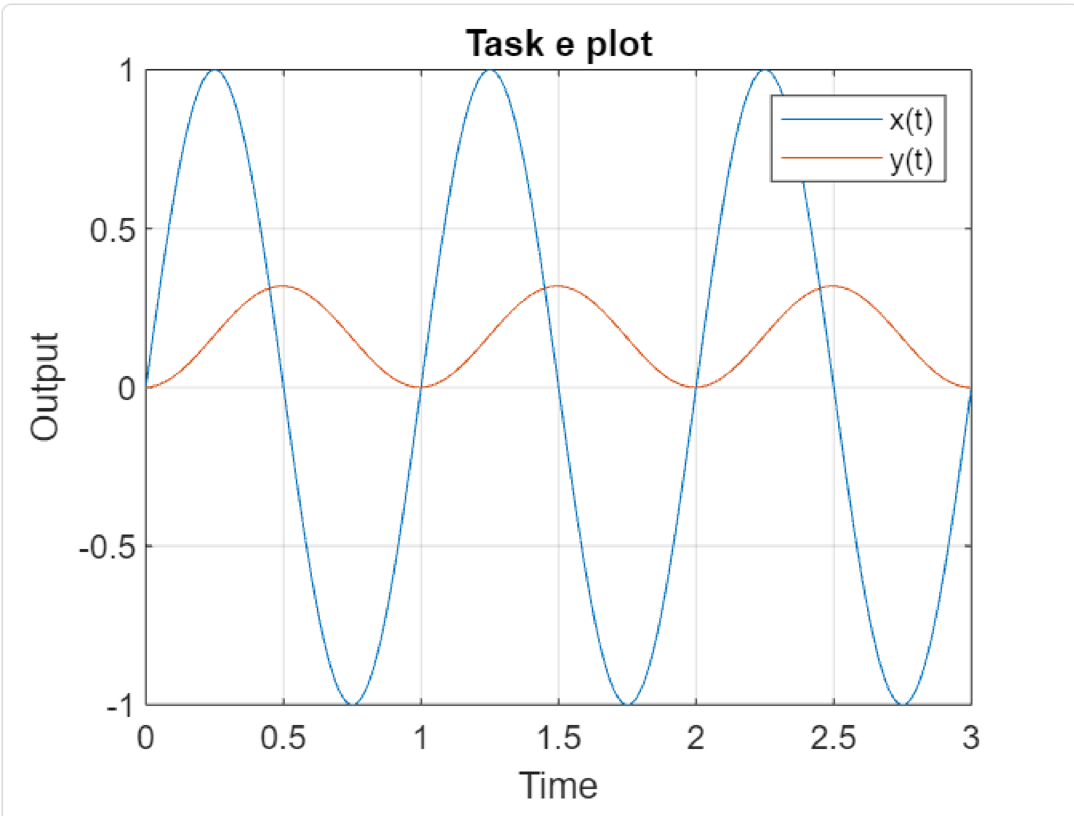
The cumsum command is adding the current element which is being accessed with the sum of previous elements. 1, 1+3, 1+3+6, 1+3+6+4, 1+3+6+4+10

```
b = 1×5
      1     3     6    10    15
```

Task (e) □ Write a MATLAB code to obtain (□) from (□) = □ sin(2□□$_0$□) using **cumsum**. Select a time step-size dt (like 0.01). Obtain plots of (□) and (□) on the same figure. Your code should make appropriate use of functions **hold on**; **grid on**; **figure**; **plot**; **legend**; **xlabel**, **ylabel**; and title. Show your work to lab demonstrator or Instructor.

Provide MATLAB code and plot here

```
A = 1;
fo = 1;
yo = 0;
t = 0:0.01:3;
xt = A*sin(2*pi*fo*t);
yt = cumsum(xt)*0.01;
figure; plot (t, xt);
title ("Task e plot");
xlabel("Time")
ylabel("Output")
hold on;
plot (t, yt)
grid on;
legend ("x(t)", "y(t)")
hold off;
```



Task (f) □ Compare the numerical values of (□) as obtained in the MATLAB simulation with the analytical result as obtained earlier in tasks (a) and (b), and note down your observations.

The integration performed using the int() function and the cumsum operation both result in the same result as their plotted graphs are identical.

Task (g) □ What advantages you may mention of using numerical integration? Think about the difficulty we may have to encounter in solving a complicated integration; think about the real scenarios, where we happen to receive signal in the form of data not in the form of some function; etc.

And advantage is when handling discrete data from a sensor. For Such values we may not have a mathematical function available thus numerical integration allows you to process it. Additionally, there may be some mathematical functions too complex to solve directly.

Task (h) □ Solve the following integral numerically using **cumsum**:

$$y(t) = \int_{0}^{□} x(\lambda)d\lambda$$

where $x(t) = \mathrm{sinc}(t-6)$, for $0 \leq t \leq 12$.

Note, the (normalized) sincfunction is defined as:

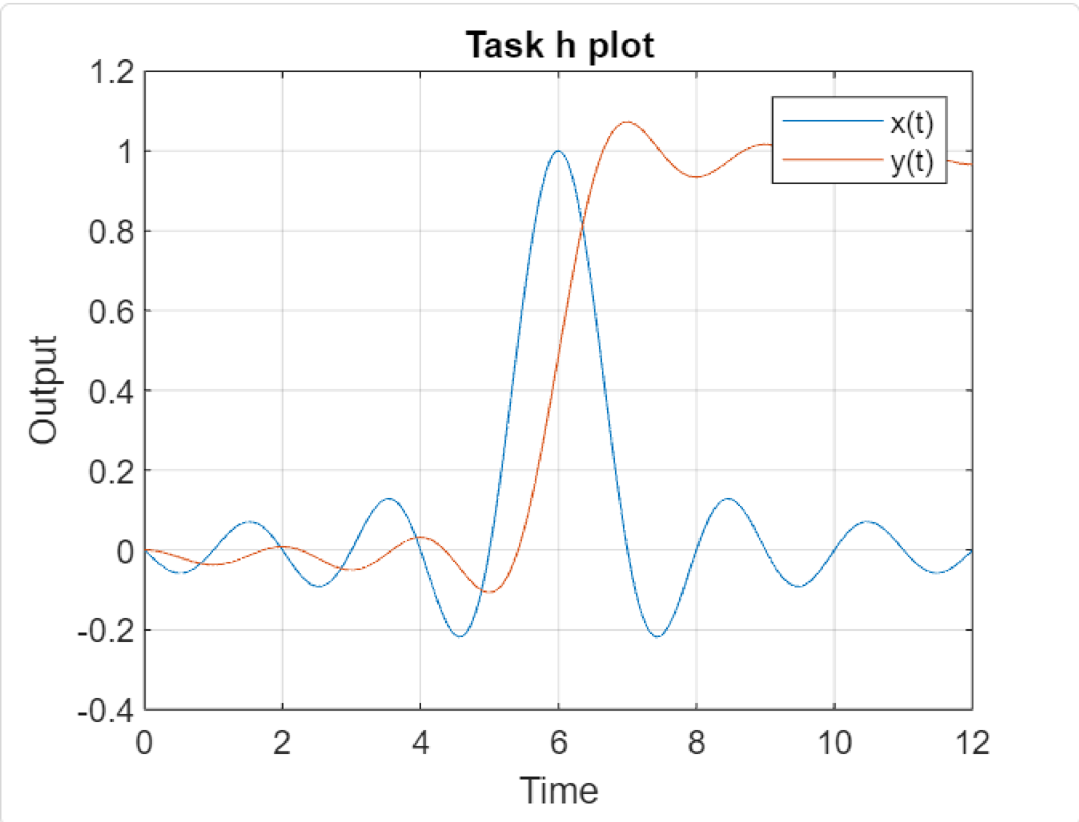
$$\mathrm{sinc}(□□) = \frac{\sin(□□□)}{□□□}$$

For more information, refer to https://en.wikipedia.org/wiki/Sinc_function.

Assign a time vector from 0 to 12 with a small step-size **dt**. Obtain the plot of (□) using MATLAB built-in **sinc** function. Obtain the value of (□) using **cumsum** and plot on the same figure. Your code should make appropriate use of **hold on**; **grid on**; **figure**; **plot**; **legend**; **xlabel**, **ylabel**; and **title**. Select the location of **legend** such that it does not over the traces of the plots. Seek help in MATLAB to learn how **legend** may be located at desirable places.

Provide MATLAB code and plot here.

```
A = 1;
fo = 1;
yo = 0;
t = 0:0.01:12;
xt = sinc(t - 6);
yt = cumsum(xt)*0.01;
figure; plot (t, xt);
title ("Task h plot");
xlabel("Time")
ylabel("Output")
hold on;
plot (t, yt)
grid on;
legend ("x(t)", "y(t)")
hold off;
```



Part II: Numerical Differentiation of a Signal

Given a signal (□), another signal (□) is obtained as time derivative of (□) as follows:

$$□(□) = \frac{□}{□} □(□)$$

□

In MATLAB, this may be done numerically by using the function **diff**. So given the numerical data of signal (□) in some vector **yt**, the numerical time-derivative of **yt**is obtained as

xt = diff(yt) / dt;

where **dt** is the time step-size. The function **diff**, when applied on an array, obtains successive differences between consecutive elements of the array. The resulting output vector **xt** becomes smaller in size than the input vector **yt** by one element. To make both vectors (arrays) have same number of elements, either a dummy *zero* or *not-a-number* (nan) element is usually appended (or inserted) in **xt** as follows:

xt = [0 xt];

or

xt = [nan xt];

Task (i) □ To understand **diff**, consider a = [1 2 3 4 5];

and evaluate b = **diff**(a)

What do you notice?

```
a=[1,2,3,4,5]
b=diff(a)
```

```
a = 1×5
       1       2       3       4       5
```

Task (j) □ Solve the following differentiation by hand:

(F)

$x(t) = \frac{d}{dt}(y(t)),$

$y(t) = 2te^{-t}$

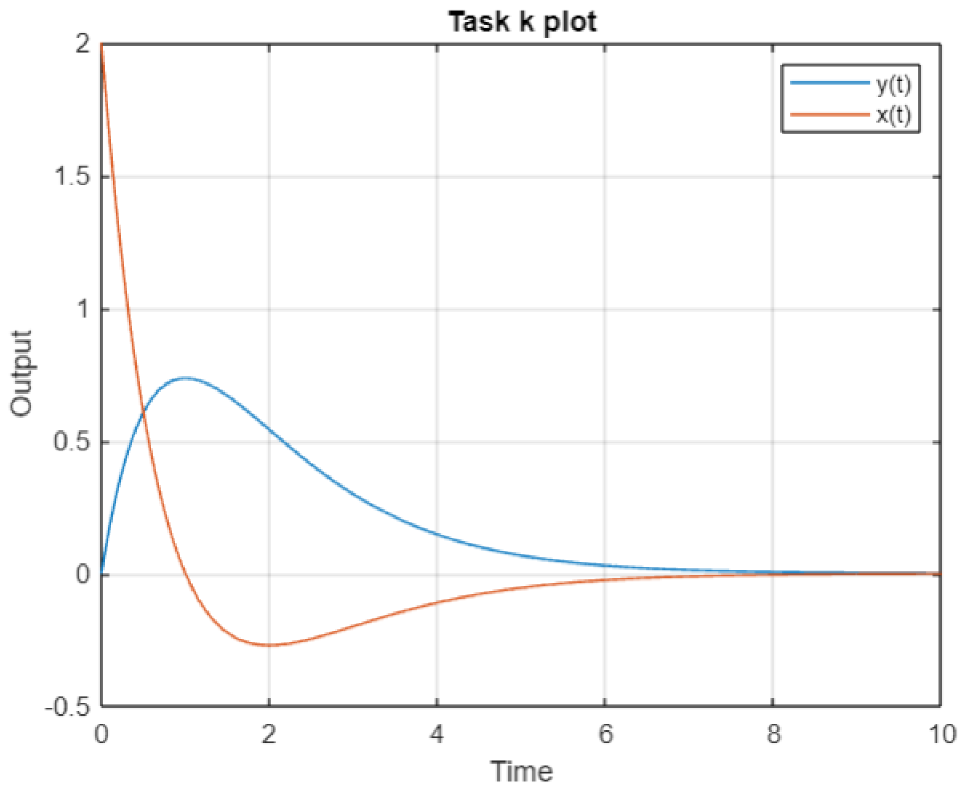
$\frac{dy}{dt} = 2e^{-t} - 2te^{-t} = e^{-t}(2-2t)$

$$□(□) = \frac{□}{□□}(t), \qquad \text{where } (t) = 2te^{-t}$$

Task (k) □ Write a MATLAB code to plot $(t) = 2te^{-t}$ and the expression of (t) (as obtained above in task (j)) for $0 \le t \le 10$ on a single figure. Select a small-time step-size. Your code should make appropriate use of functions **hold on**; **grid on**; **figure**; **plot; legend; xlabel, ylabel**; and **title**.

Put MATLAB code here:

```
t = 0:0.01:10;
yt = 2.*t.*exp(-t);
xt = 2.*exp(-t).*(1-t);
figure; plot (t, yt);
title ("Task k plot");
xlabel("Time")
ylabel("Output")
hold on;
plot (t, xt)
grid on;
4
legend ("y(t)", "x(t)")
hold off;
```

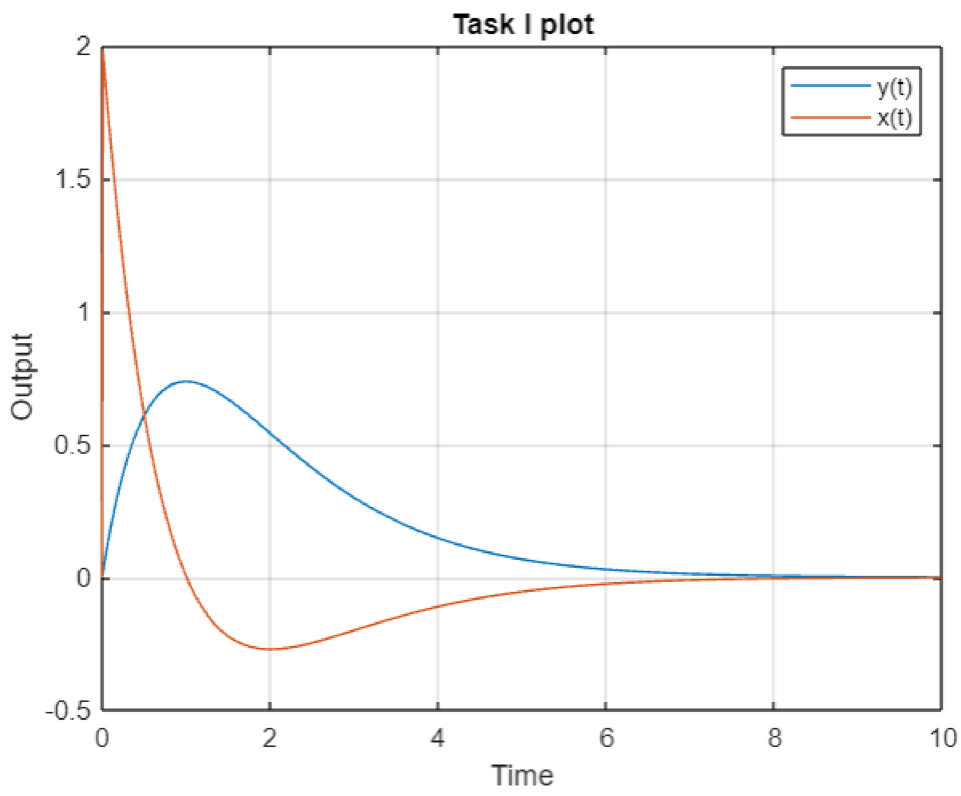


Provide MATLAB plot here:

Task (l) □ Write a MATLAB code to obtain (t) numerically from $(t) = 2te^{-}$, $0 \leq t \leq 10$, using the function **diff**. Select a reasonable time step-size **dt**. Obtain plots of (t) and (t) on the same figure. Your code should make appropriate use of functions **hold on**; **grid on**; **figure**; **plot**; **legend**; **xlabel**, **ylabel**; and title.

Provide your MATLAB code and plot here:

```
t = 0:0.01:10;
yt = 2.*t.*exp(-t);
xt = diff(yt)/0.01;
xt=[0 xt];
figure; plot (t, yt);
title ("Task l plot");
xlabel("Time")
ylabel("Output")
hold on;
plot (t, xt)
grid on;
legend ("y(t)", "x(t)")
hold off;
```

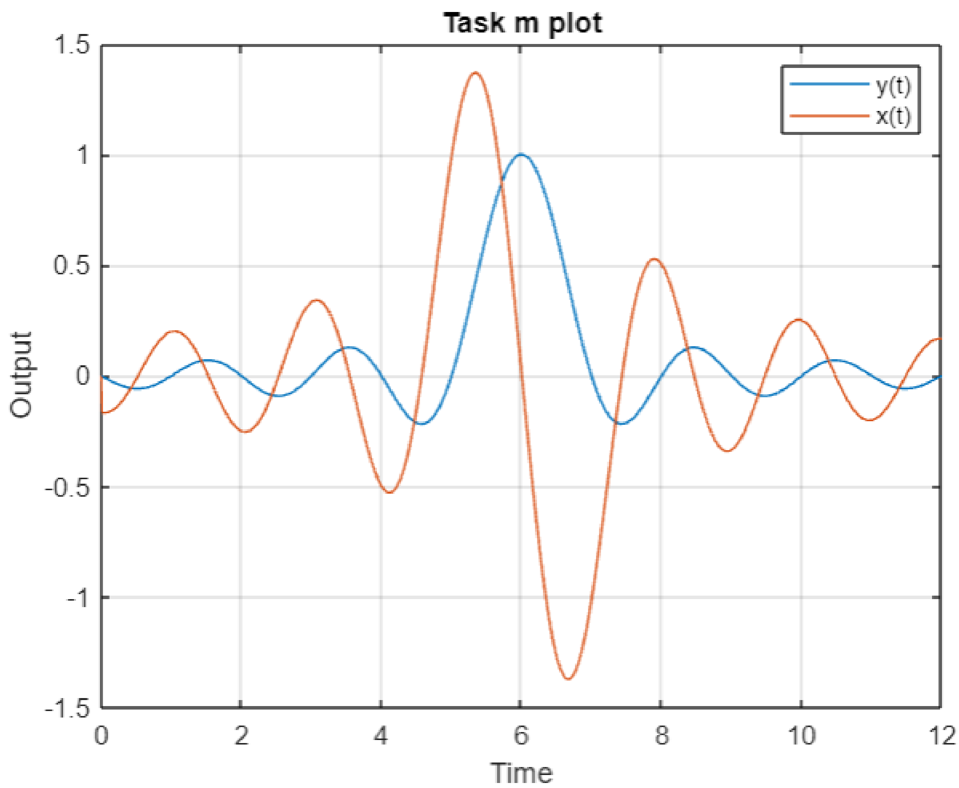

Task I plot
y(t)
x(t)
Output
Time
2
1.5
1
0.5
0
-0.5
0
2
4
6
8
10

Task (m) □ Solve the following differentiation numerically using function **diff**:

$$□(□) = \frac{□}{□ \; □}(t), \qquad \text{where } (t) = \text{sinc}\,(t-6)$$

for $0 \leq t \leq 12$. Select a proper time step-size. Obtain plots of (t) and (t) on the same figure. Your code should make appropriate use of functions **hold on**; **grid on**; **figure**; **plot**; **legend**; **xlabel**, **ylabel**; and **title**. Select the location of **legend** such that it does not interfere with the plots. Seek help in MATLAB to learn how location of **legend** may be changed.

```
t = 0:0.01:12;
yt = sinc(t-6);
% xt = 2.*exp(-t).*(1-t);
xt = diff(yt)/0.01;
xt=[0 xt];
figure; plot (t, yt);
title ("Task m plot");
xlabel("Time")
ylabel("Output")
hold on;
plot (t, xt)
grid on;
legend ("y(t)", "x(t)")
hold off;
```



Post Lab

Part III: Integrating and differentiating basic signal functions like unit-impulse and unit-step

Unit-impulse function: The unit-impulse function plays an important role in mathematical modeling and analysis of signals and linear systems. It is defined by the following equations:

$$(t) = \{ \begin{matrix} 0, t \neq 0 \\ = 0 \end{matrix} \text{ undefined, } t$$

Note that this is an incomplete definition of the function (t) since the amplitude of it is defined only when $t \neq 0$, and is undefined at the time instant $t = 0$. This is not possible to draw function (t) in MATLAB due to its incomplete definition. However, it is possible to plot (t) using the following asymptotic alternate definitions:

$$(\square) = \lim_{a \to 0} \frac{1}{a\sqrt{\pi}} e^{-t^2/a^2}$$

or

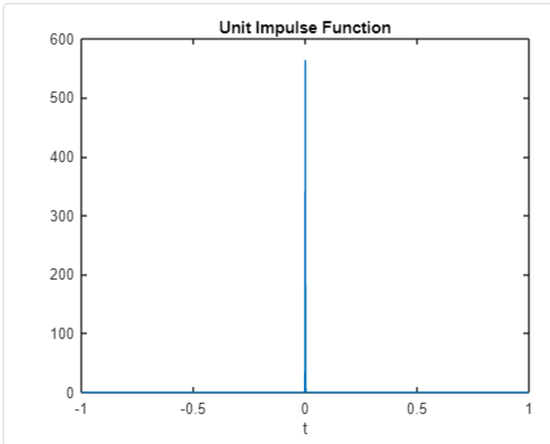
$$(\square) = \lim_{a \to 0} \frac{\square}{(\pi t)^2} \sin^2 (\frac{\square\square}{a})$$

These definitions also ensure that the area under the curve of $(\square)$ is unity, i.e., $\int_{-\infty}^{\infty} (\square)\square\square = 1$.

Task (n) □ Write a MATLAB code to plot $(\square)$ using its alternate asymptotic definitions.

Caution: select a reasonably very small step-size for time axis like **dt = 0.001**. Also consider a very small value for □, like **a = 0.001**.

```
% Defining the parameters
syms a t % Defining symbolic variables
t = -1:0.001:1;
a = 0.001;
dt = 0.001;
% Creating the unit impulse function using the two different functions
unit_impulse = 1/(a*sqrt(pi)) * exp(-t.^2/a^2);
% Plotting
figure;
plot(t, unit_impulse);
title('Unit Impulse Function');
xlabel('t');
```



Task (o) □ Show that the total area under the curve of function (□) as obtained in MATLAB above is **unity**. Hint: sum all values of (□) using the command **sum**and multiply it with **dt**.

```
area = sum(unit_impulse * dt)
```

```
area = 1.0001
```

Unit-step function: The unit-step function is useful in situations where we need to model a signal that is turned on or off at a specific time instant. It is defined as follows:

$$(\square) = \begin{cases} 1, \square > 0 \\ 0, \square < 0 \end{cases}$$

The relationship between the unit-step function and the unit-impulse function is important. The unit-step function can be expressed as a running integral of the unit-impulse function:

$$u(t) = \int_{-\infty}^{\square} \delta(\lambda)\, d\lambda$$

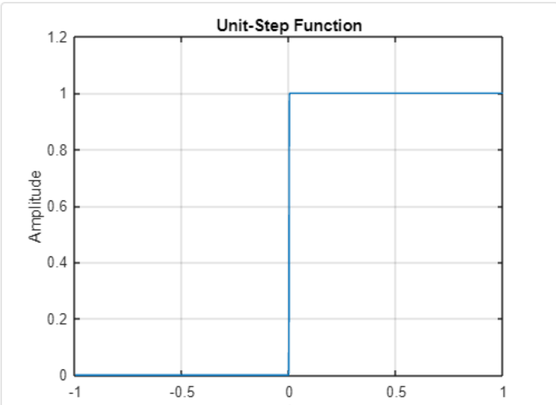
Task (p) □ Integrate the function (□) in MATLAB, as obtained in task (m), and show that it yields (□). **Hint:** the time integration may easily be implemented using **cumsum**and multiplying with **dt**.

```
plot(t, unit_impulse);
title('Unit Impulse Function');
xlabel('t');


area = sum(unit_impulse * dt)




% Obtaining unit step function by numerically integrating the unit impulse function
ut = dt*cumsum(unit_impulse);
% Plotting the unit step function
figure;
plot(t, ut);
title('Unit-Step Function');
xlabel('t');
ylabel('Amplitude');
grid on;
```



Task (q) □ Since, (□) is the integral of (□); therefore, (□) is the time derivative of □(□),

$$\square(\square) = \frac{\square}{\square}\square(\square)$$

□

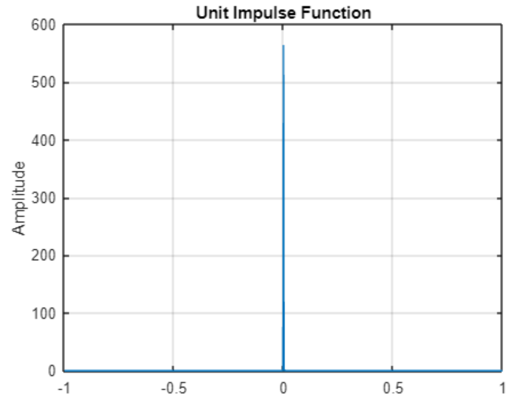
Use the MATLAB function **diff**in numerical mode, take **diff**of expression (□) as obtained in task (o), divide it with **dt**, and show that it yields the unit impulse function

Provide your MATLAB code and plot here:

```
xlabel('t');
ylabel('Amplitude');
grid on;



% Obtaining unit impulse function by differentiating unit step function
unit_impulse_again = diff(ut)/dt;
unit_impulse_again = [0 unit_impulse_again]; % Making sure size of vectors match
% Plotting the unit impulse function
figure;
plot(t, unit_impulse_again);
title('Unit Impulse Function');
xlabel('t');
ylabel('Amplitude');
grid on;
```



It is observed from the plot taking the time derivative of u(t), the unit-step function gives us the unit-impulse function.

Lab 3: Integrating and Differentiating Continuous-Time Signals in MATLAB

Habib University

EE-252 Signal and Systems

Name: ________________________________ Student ID.: ____________________

Marks distribution:

<table>
<tr><th></th><th></th><th>LR2</th><th>LR5</th><th>LR9</th><th>AR4</th></tr>
<tr><td rowspan="8">In-Lab</td><td>Task a-b</td><td></td><td>4</td><td>4</td><td rowspan="8">12</td></tr>
<tr><td>Task c-d</td><td>4</td><td>8</td><td rowspan="7">6</td></tr>
<tr><td>Task e-f</td><td>4</td><td>8</td></tr>
<tr><td>Task g-h</td><td>4</td><td>8</td></tr>
<tr><td>Task i-j-k</td><td>12</td><td>12</td></tr>
<tr><td>Task l-m</td><td>4</td><td>8</td></tr>
<tr><td>Task n-o</td><td>8</td><td>8</td></tr>
<tr><td>Task p-q</td><td>8</td><td>8</td></tr>
<tr><td colspan="2">Max Marks = 130</td><td>44</td><td>64</td><td>10</td><td>12</td></tr>
</table>

Marks obtained:

<table>
<tr><th></th><th></th><th>LR2</th><th>LR5</th><th>LR9</th><th>AR4</th></tr>
<tr><td rowspan="8">In-Lab</td><td>Task a-b</td><td></td><td></td><td></td><td rowspan="8"></td></tr>
<tr><td>Task c-d</td><td></td><td></td><td></td></tr>
<tr><td>Task e-f</td><td></td><td></td><td></td></tr>
<tr><td>Task g-h</td><td></td><td></td><td></td></tr>
<tr><td>Task i-j-k</td><td></td><td></td><td></td></tr>
<tr><td>Task l-m</td><td></td><td></td><td></td></tr>
<tr><td>Task n-o</td><td></td><td></td><td></td></tr>
<tr><td>Task p-q</td><td></td><td></td><td></td></tr>
<tr><td colspan="2">Marks Obtained</td><td></td><td></td><td></td><td></td></tr>
</table>