CLab-1 — "Operations on Signals, Sinusoidal Signals, Periodicity, and Convolution"

Lab Week: Week 5
Total Marks: 10
Contribution to Final Assessment: 2%
Grading: Lab is marked based on satisfactory completion of tasks. Attendance is compulsory.
Relevant Textbook Sections: 1.2, 1.3 and 2.1

1 Learning Outcomes

After completing this lab, the students should be able to:

- implement various signal operations using **MATLAB** code;
- explain the periodicity of discrete sinusoidal signals, and compute their fundamental frequency; and
- demonstrate a thorough understanding of the concept of signals convolution.

2 Background on MATLAB

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Math and computation
- Algorithm development
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including Graphical User Interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.

The name **MATLAB** stands for matrix laboratory. **MATLAB** was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects (superceded now by LAPACK).

MATLAB has evolved over a period of years with input from many users. In university environments, it is a common instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, **MATLAB** is used for research, development, and analysis.

2.1 Instructions to Execute MATLAB Code

For some given **MATLAB** code, you can type the commands in line-by-line or, alternatively, you can use an editor to type the commands into a file. If you type them in line-by-line, then they are executed as soon as you type them. If you put them in a file, then it should have a name that ends with `.m`; in that case, the commands will be executed when you type the name of the file at the **MATLAB** prompt (for example, if you name the file `Lab1.m`

and put it in the current directory, then the commands will be run when you type `Lab1` at the **MATLAB** prompt).

If you have not worked in the **MATLAB** before, you should get familiar with the **MATLAB** system, the language syntax and how to write and invoke `m`-files. A deep understanding of **MATLAB** is not required to successfully complete this lab.

Note: Students should already be familiar with **MATLAB** from the **ENGN2219** Computing for Engineering Simulation course in Semester 1.

3 Basic Operations on Signals

In lectures, we have studied that combined time scaling, time reversal and time shifting operations on any signal $f(t)$ can be represented by $f(\alpha t + \beta)$ for $\alpha, \beta \in \mathbb{R}$. Here, we implement and visualize these operations.

Task 1

For this task, assume the signal

$$f(t) = \exp(-t)\,\cos(2\pi t)\,u(t).$$

✱ **Task 1-1:** The following **MATLAB** code plots $f(t)$ over $-3 \leq t \leq 3$ seconds.

```
tinc = 0.01;
t = -3:tinc:3;
f = exp(-t).*cos(2*pi*t).*heaviside(t);
plot(t,f);
xlabel('time, t (seconds)');
ylabel('f(t)');
```

If you run it in **MATLAB** confirm you get the graph shown in Figure 1.

✱ **Task 1-2:** Plot the following signals over $-3 \leq t \leq 3$ seconds by modifying the **MATLAB** code above.

A. $f(-t)$
B. $f(-3t)$
C. $f(-t/3)$
D. $f(3t-1)$

First plot all of the signals and then show to the tutor.

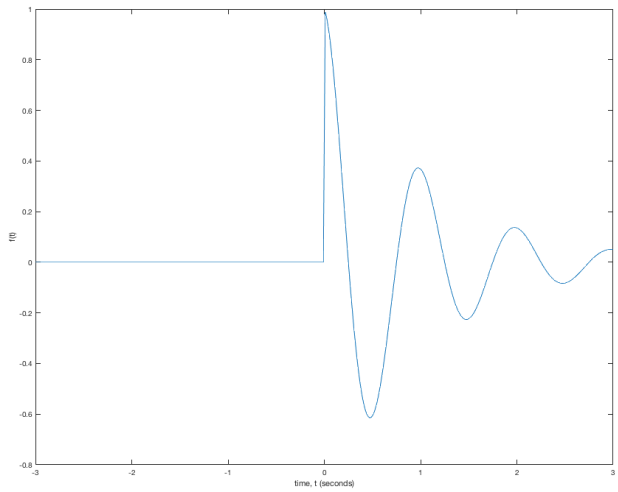

Figure 1: *Signal f(t) in Task 1*

4 Discrete-Time Sinusoidal Signals

Complex exponentials play an important role in the analysis of discrete-time signals and systems. A discrete-time complex exponential has the form α^n, where α is a complex scalar. The discrete-time sine and cosine signals can be built from complex exponential signals by setting $\alpha = e^{\pm j\omega}$, namely,

$$\cos(\omega n) = \frac{1}{2}\left(e^{j\omega} + e^{-j\omega}\right)$$
$$\sin(\omega n) = \frac{1}{2j}\left(e^{j\omega} - e^{-j\omega}\right).$$

In this part, you will create and analyze a number of discrete-time sinusoids.

Task 2

✱ **Task 2-1:**

Consider the signal

$$x_M[n] = \sin\left(\frac{2\pi M n}{N}\right),$$

and assume $N = 12$. For $M = 4, 5, 7,$ and 10, plot $x_M[n]$ on the interval $0 \leq n \leq 2N - 1$.

Use the `stem` function (instead of `plot`) to create your figures and be sure to appropriately label (`xlabel`, `ylabel`) your axes.

✱ **Task 2-2:**

- What is the fundamental period of each signal?
- In general, how can the fundamental period be determined from arbitrary integer M and N? (Remember that M can be greater than N.)

✱ **Task 2-3:**

Consider the signal

$$x_k[n] = \sin\left(\omega_k n\right),$$

where $\omega_k = 2\pi k/5$. For $x_k[n]$ given by $k = 1, 2, 4$, and 6, use `stem` to plot each signal on the interval $0 \leq n \leq 9$. Use `subplot` to plot each signal on separate axes on the same figure. How many *unique* plots did you obtain? Explain your observations.

✱ **Task 2-4:**

Now plot each of the following signal on the interval $0 \leq n \leq 31$:

$$x_1[n] = \sin\left(\frac{\pi n}{4}\right)\cos\left(\frac{\pi n}{4}\right),$$
$$x_2[n] = \cos^2\left(\frac{\pi n}{4}\right),$$
$$x_3[n] = \sin\left(\frac{\pi n}{4}\right)\cos\left(\frac{\pi n}{8}\right).$$

What is the fundamental period of the each signal?

✱ **Task 2-5:**

- What conclusion can you draw from Task 2-4, above?
- Is the multiplication of periodic signals necessarily periodic? Explain your answer.

5 Discrete Time Convolution

Here we implement and visualize the convolution of discrete time signals. We also carry out a simple task to study the effect of convolution.

The convolution of discrete time signals $f[n]$ and $h[n]$ results in an output signal $g[n]$ given by

$$g[n] = f * h \triangleq \sum_{m=-\infty}^{\infty} f[m]h[n-m].$$

In **MATLAB**, the convolution of two signals can be performed using the command **conv**. For example, the convolution of two signals $f[n] = (0.8)^n u[n]$ and $h[n] = (0.5)^n u[n]$ is computed and plotted using the following code for $-20 \leq n \leq 20$, as follows:

```
n=-20:20;
unitstep = double(n>=0); % define a discrete heaviside (unit step) function
f = (0.8).^n.*unitstep;
h = (0.5).^n.*unitstep;
g = conv(f,h); % convolution between f and h
subplot(3,1,1);
stem(n,f);
xlabel('time, n'); ylabel('f[n]');
subplot(3,1,2);
stem(n,h);
xlabel('time, n'); ylabel('h[n]');
subplot(3,1,3);
N = ceil(length(h)/2);
stem(n,g(N:end-N+1));
xlabel('time, n '); ylabel('g[n]');
```

Task 3

✱ Task 3-1:

Try to understand the provided **MATLAB** code. Use **MATLAB** help for the function **conv** to learn why we have defined N in the code above. Note down your understanding and show it to the tutor at the end of the lab.

✱ Task 3-2:

Download the **testsound.wav** file from Wattle. Generate a sound signal $f[n]$ and play it using the following **MATLAB** code:

```
[f,Fs] = wavread('test_sound');
sound(f,Fs);
```

f is a sound signal and **Fs** is the sampling frequency at which the sound signal is sampled. The **MATLAB** function **wavread** reads the data from the sound file.

✱ Task 3-3:

Generate an output $g = f * h$ by convolving the signal $f[n]$ with the following signal $h[n]$:

$$h[n] = \begin{cases} 1 & 0 \leq n \leq 60 \\ 0 & \text{otherwise.} \end{cases}$$

✱ Task 3-4:

Play the signal $g[n]$ using the command:

```
sound(g,Fs);
```

✱ Task 3-5:

Observe the difference between the original sound signal $f[n]$ and the convolution output signal $g[n]$. Record your observations and write down the intuitive reason of the difference in $f[n]$ and $g[n]$. At the end of the lab, show your working to the tutor.