

---

## Lab P-1: Introduction to MATLAB

---

**Pre-Lab and Warm-Up:** You should read at least the Pre-Lab and Warm-up sections of this lab assignment and go over all exercises in the Pre-Lab section before attending your lab session.

**Verification:** The Warm-up section of each lab must be completed **during your assigned Lab time** and the steps marked *Instructor Verification* must also be signed off **during the lab time**. One of the laboratory instructors must verify the appropriate steps by signing on the **Instructor Verification** line. When you have completed a step that requires verification, simply demonstrate the step to the instructor. Turn in the completed verification sheet before you leave the lab.

**Lab Report:** It is only necessary to turn in a report on Section 3 with graphs and explanations. You are asked to **label** the axes of your plots and include a title for every plot. In order to keep track of plots, include plots *inlined* within your report.

---

## 1 Pre-Lab

In this first week, the Pre-Lab will be extremely short and very easy. Make sure that you read through the information below prior to coming to lab.

### 1.1 Overview

MATLAB will be used extensively in all the labs. The primary goal of this lab is to familiarize yourself with using MATLAB. Please read Appendix B: *Programming in MATLAB* for an overview. Here are three specific goals for this lab:

1. Learn basic MATLAB commands and syntax, including the help system.
2. Learn to write and edit your own script files in MATLAB, and run them as commands.
3. Learn a little about advanced programming techniques for MATLAB, i.e., vectorization.

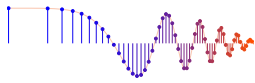
### 1.2 Movies: MATLAB Tutorials

In Appendix B, there are a large number of Real-media movies on basic topics in MATLAB, e.g., colon operator, indexing, functions, etc.

### 1.3 Getting Started

After logging in, you can start MATLAB by double-clicking on a MATLAB icon, typing `matlab` in a terminal window, or by selecting MATLAB from a menu such as the START menu under Windows-95/98/NT. The following steps will introduce you to MATLAB.

- (a) View the MATLAB introduction by typing `intro` at the MATLAB prompt. This short introduction will demonstrate some of the basics of using MATLAB.
- (b) Run the MATLAB help desk by typing `helpdesk`. The help desk provides a hypertext interface to the MATLAB documentation. The MATLAB preferences can be set to use Netscape or Internet Explorer as the browser for help. Two links of interest are **Getting Help** (at the bottom of the right-hand frame), and **Getting Started** which is under MATLAB in the left-hand frame.



- (c) Explore the MATLAB help capability available at the command line. Try the following:

```
help
help plot
help colon      %<--- a VERY IMPORTANT notation
help ops
help zeros
help ones
lookfor filter  %<--- keyword search
```

NOTE: it is possible to force MATLAB to display only one screen-full of information at once by issuing the command `more on`).

- (d) Run the MATLAB demos: type `demo` and explore a variety of basic MATLAB commands and plots.

- (e) Use MATLAB as a calculator. Try the following:

```
pi*pi - 10
sin(pi/4)
ans ^ 2      %<--- "ans" holds the last result
```

- (f) Do variable name assignment in MATLAB. Try the following:

```
x = sin( pi/5 );
cos( pi/5 )      %<--- assigned to what?
y = sqrt( 1 - x*x )
ans
```

- (g) Complex numbers are natural in MATLAB. The basic operations are supported. Try the following:

```
z = 3 + 4i, w = -3 + 4j
real(z), imag(z)
abs([z,w])      %<-- Vector constructor
conj(z+w)
angle(z)
exp( j*pi )
exp(j*[ pi/4, 0, -pi/4 ])
```

## 2 Warm-Up

### 2.1 MATLAB Array Indexing

- (a) Make sure that you understand the **colon** notation. In particular, explain in words what the following MATLAB code will produce

```
jkl = 0 : 6
jkl = 2 : 4 : 17
jkl = 99 : -1 : 88
ttt = 2 : (1/9) : 4
tpi = pi * [ 0:0.1:2 ];
```

- (b) Extracting and/or inserting numbers into a vector is very easy to do. Consider the following definition of `xx`:

```
xx = [ zeros(1,3), linspace(0,1,5), ones(1,4) ]
xx(4:6)
size(xx)
length(xx)
xx(2:2:length(xx))
```



Explain the results echoed from the last four lines of the above code.

- (c) Observe the result of the following assignments:

```
yy = xx; yy(4:6) = pi*(1:3)
```

Now write a statement that will take the vector `xx` defined in part (b) and replace the even indexed elements (i.e., `xx(2)`, `xx(4)`, etc) with the constant  $\pi$ . Use a vector replacement, not a loop.

**Instructor Verification** (separate page)

## 2.2 MATLAB Script Files

- (a) Experiment with vectors in MATLAB. Think of the vector as a set of numbers. Try the following:

```
xk = cos( pi*(0:11)/4 ) %<---comment: compute cosines
```

Explain how the different values of cosine are stored in the vector `xk`. What is `xk(1)`? Is `xk(0)` defined?

NOTES: the semicolon at the end of a statement will suppress the echo to the screen. The text following the `%` is a comment; it may be omitted.

- (b) (A taste of vectorization) Loops can be written in MATLAB, but they are NOT the most efficient way to get things done. It's better to **always avoid loops** and use the colon notation instead. The following code has a loop that computes values of the cosine function. (The index of `yy()` must start at 1.) Rewrite this computation without using the loop (follow the style in the previous part).

```
yy = [ ]; %<--- initialize the yy vector to be empty
for k=-5:5
    yy(k+6) = cos( k*pi/3 )
end
yy
```

Explain why it is necessary to write `yy(k+6)`. What happens if you use `yy(k)` instead?

**Instructor Verification** (separate page)

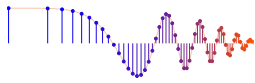
- (c) Plotting is easy in MATLAB for both real and complex numbers. The basic plot command will plot a vector `y` versus a vector `x` connecting successive points by straight lines. Try the following:

```
x = [-3 -1 0 1 3 ];
y = x.*x - 3*x;
plot( x, y )
z = x + y*sqrt(-1)
plot( z ) %<---- complex values: plot imag vs. real
```

Use `help arith` to learn how the operation `xx.*xx` works when `xx` is a vector; compare to matrix multiply.

When unsure about a command, use `help`.

- (d) Use the built-in MATLAB editor (on Windows-95/98/NT), or an external one such as EMACS on UNIX/LINUX, to create a script file called `mylab1.m` containing the following lines:



```
tt = -1 : 0.01 : 1;
xx = cos( 5*pi*tt );
zz = 1.4*exp(j*pi/2)*exp(j*5*pi*tt);
plot( tt, xx, 'b-', tt, real(zz), 'r--' ), grid on      %<--- plot
a sinusoid
title('TEST PLOT of a SINUSOID')
xlabel('TIME (sec)')
```

Explain why the plot of  $\text{real}(zz)$  is a sinusoid. What is its phase and amplitude? Make a calculation of the phase from a time-shift measured on the plot.

**Instructor Verification** (separate page)

- (e) Run your script from MATLAB. To run the file `mylab1` that you created previously, try

```
mylab1          %<---will run the commands in the file
type mylab1     %<---will type out the contents of
% mylab1.m to the screen
```

### 2.3 MATLAB Sound (optional)

The exercises in this section involve sound signals, so you should bring headphones to the lab for listening.

- (a) Run the MATLAB sound demo by typing `xpsound` at the MATLAB prompt. If you are unable to hear the sounds in the MATLAB demo then ask an instructor for help.

When unsure about a command, use `help`.

- (b) Now generate a tone (i.e., a sinusoid) in MATLAB and listen to it with the `soundsc()` command.<sup>1</sup> The first two lines of code in part 2.2(d) create a vector `xx` of values of a 2.5 Hz sinusoid. The frequency of your sinusoidal tone should be 2000 Hz and its duration should be 0.9 sec. Use a sampling rate (`fs`) equal to 11025 samples/sec. The sampling rate dictates the time interval between time points, so the time-vector should be defined as follows:

```
tt = 0:(1/fs):dur;
```

where `fs` is the desired sampling rate and `dur` is the desired duration (in seconds). Read the online help for both `sound()` and `soundsc()` to get more information on using this command. What is the length (number of samples) of your `tt` vector?

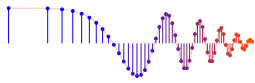
**Instructor Verification** (separate page)

## 3 Lab Exercise: Manipulating Sinusoids with MATLAB

Now you're on your own. **Include a short summary of this Section with plots in your Lab report.**

Write a MATLAB script file to do steps (a) through (d) below. Include a listing of the script file with your report.

<sup>1</sup>The `soundsc(xx, fs)` function requires **two** arguments: the first one (`xx`) contains the vector of data to be played, the second argument (`fs`) is the sampling rate for playing the samples. In addition, `soundsc(xx, fs)` does automatic scaling and then calls `sound(xx, fs)` to actually play the signal.



- (a) Generate a time vector (`tt`) to cover a range of  $t$  that will exhibit approximately two cycles of the 4000 Hz sinusoids defined in the next part, part (b). Use a definition for `tt` similar to part 2.2(d). If we use  $T$  to denote the period of the sinusoids, define the starting time of the vector `tt` to be equal to  $-T$ , and the ending time as  $+T$ . Then the two cycles will include  $t = 0$ . **Finally, make sure that you have at least 25 samples per period of the sinusoidal wave.** In other words, when you use the colon operator to define the time vector, make the increment small enough to generate 25 samples per period.
- (b) Generate two 4000 Hz sinusoids with arbitrary amplitude and time-shift.

$$x_1(t) = A_1 \cos(2\pi(4000)(t - t_{m_1})) \quad x_2(t) = A_2 \cos(2\pi(4000)(t - t_{m_2}))$$

Select the value of the amplitudes and time-shifts as follows: Let  $A_1$  be equal to your age and set  $A_2 = 1.2A_1$ . For the time-shifts, set  $t_{m_1} = (37.2/M)T$  and  $t_{m_2} = -(41.3/D)T$  where  $D$  and  $M$  are the day and month of your birthday, and  $T$  is the period.

Make a plot of both signals over the range of  $-T \leq t \leq T$ . For your final printed output in part (d) below, use `subplot(3,1,1)` and `subplot(3,1,2)` to make a three-panel figure that puts both of these plots in the same figure window. See `help subplot`.

- (c) Create a third sinusoid as the sum:  $x_3(t) = x_1(t) + x_2(t)$ . In MATLAB this amounts to summing the vectors that hold the values of each sinusoid. Make a plot of  $x_3(t)$  over the same range of time as used in the plots of part (b). Include this as the third panel in the plot by using `subplot(3,1,3)`.
- (d) Before printing the three plots, put a title on each subplot, and include your name in one of the titles. See `help title`, `help print` and `help orient`, especially `orient tall`.

### 3.1 Theoretical Calculations

Remember that the phase of a sinusoid can be calculated after measuring the time location of a positive peak,<sup>2</sup> if we know the frequency.

- (a) Make measurements of the “time-location of a positive peak” and the amplitude from the plots of  $x_1(t)$  and  $x_2(t)$ , and write those values for  $A_i$  and  $t_{m_i}$  directly on the plots. Then calculate (by hand) the phases of the two signals,  $x_1(t)$  and  $x_2(t)$ , by converting each time-shift  $t_{m_i}$  to phase. Write the calculated phases  $\phi_i$  directly on the plots.

Note: when doing computations, express phase angles in radians, not degrees!

- (b) Measure the amplitude  $A_3$  and time-shift  $t_{m_3}$  of  $x_3(t)$  directly from the plot and then calculate the phase ( $\phi_3$ ) by hand. Write these values directly on the plot to show how the amplitude and time-shift were measured, and how the phase was calculated.
- (c) Now use the phasor addition theorem. Carry out a phasor addition of complex amplitudes for  $x_1(t)$  and  $x_2(t)$  to determine the complex amplitude for  $x_3(t)$ . Use the complex amplitude for  $x_3(t)$  to verify that your previous calculations of  $A_3$  and  $\phi_3$  were correct.

### 3.2 Complex Amplitude

Write one line of MATLAB code that will generate values of the sinusoid  $x_1(t)$  above by using the complex-amplitude representation:

$$x_1(t) = \Re\{Xe^{j\omega t}\}$$

Use appropriate constants for  $X$  and  $\omega$ .

<sup>2</sup>Usually we say time-delay or time-shift instead of the “time location of a positive peak.”



---

## Lab: Introduction to MATLAB INSTRUCTOR VERIFICATION SHEET

Turn this page in to your lab grading TA before the end of your scheduled Lab time.

Name: \_\_\_\_\_ LoginUserName: \_\_\_\_\_ Date: \_\_\_\_\_

Part 2.1 Vector replacement using the colon operator:

Verified: \_\_\_\_\_ Date/Time: \_\_\_\_\_

Part 2.2(b) Explain why it is necessary to write  $yy(k+6)$ . What happens if you use  $yy(k)$  instead?

Verified: \_\_\_\_\_ Date/Time: \_\_\_\_\_

Part 2.2(d) Explain why the plot of  $\text{real}(zz)$  is a sinusoid. What is its amplitude and phase? In the space below, make a calculation of the phase from time-shift.

Verified: \_\_\_\_\_ Date/Time: \_\_\_\_\_

Part 2.3 (optional) Use `soundsc()` to play a 2000 Hz tone in MATLAB:

Verified: \_\_\_\_\_ Date/Time: \_\_\_\_\_