GEORGIA INSTITUTE OF TECHNOLOGY

SCHOOL of ELECTRICAL and COMPUTER ENGINEERING

ECE 2026 Fall 2013
Lab #0: Introduction to MATLAB

Date: 19 – 23 August 2013

Important Note: *This is the first lab of the semester and is mainly to guide you through the* MATLAB *tool.* ***This initial lab is different from the rest of the labs as it does not require you to be present in Klaus 2440, the ECE2026 Lab****. Normally, each lab assignment requires a student to complete three stages: Pre-Lab, Exercise and Verification, and Lab Questions. The Pre-Lab stage requires you to read the assignment and to familiarize yourself with what the lab is about and what additional tools or equipment you may need before you come to the lab session. The Exercise and Verification stage will take place in Klaus 2440. You need to follow the instructions to complete the exercise, which will enable you to answer questions and to record necessary data on the Lab sheet (attached to the end of each lab assignment). Lab exercise is a critical component of the course grade; please see syllabus for details.*

For this lab ONLY, students can complete the exercise without going to the assigned lab sessions during the first week. Starting from Lab 1, the normal procedure has to be strictly followed and all students are required to go to the ***assigned*** *lab sessions.*

All labs are held in room 2440 of the Klaus building. Your GT login will work, as long as you specify the "Windows domain" to be AD. If you have difficulty logging in, send an E-mail to

help@ece.gatech.edu,

or visit the web site: **www.ece-help.gatech.edu**

If you have MATLAB installed on your laptop, you are welcome to use your laptop in lab instead of the lab desktop computers if you wish.

General Lab Instruction

Pre-Lab: You should read the Pre-Lab section of the lab and do all the steps in the Pre-Lab section ***before your assigned lab time.***

Exercise and Verification: The Exercise section 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, raise your hand and demonstrate the step to the TA or instructor.

Lab Questions: *Instructor Verification* is part of the Lab Sheet which has another component with a few lab related questions that need to be answered at your own pace. The completed lab sheet is, unless noted otherwise, due at the beginning of the next lab.

The predecessor to the three-credit-hour class ECE2026, which was a four-credit-hour class called ECE2025, consisted of labs that require a substantial amount of work to be done outside of lab and an elaborate lab report. To change the workload commensurate with losing one credit hour, we have reworked the labs such that most of the lab exercises can be completed in individual lab sessions and no lab report (except one in the semester) is required. To make this work well, you will need to ***come into lab prepared***.

Forgeries and plagiarism are a violation of the honor code and will be referred to the Dean of Students for disciplinary action. You are allowed to discuss lab exercises with other students and you are allowed to consult old lab reports, but you cannot give or receive written material or electronic files. Your submitted work should be original and it should be your own work.

PRINTING BUDGET: For the printers in the ECE labs, you have a quota. Please limit your printing to essential items for the labs. If you need to print lecture slides and other large documents, use the central (OIT) printing facilities.

1. Introduction

The objective of this lab is to help you regain your familiarity with MATLAB, a programming tool most suitable for signal processing studies. All labs of ECE2026 will involve use of MATLAB in some form.

2. Pre-Lab

Please read through the information below prior to attending your lab.

2.1 Overview

MATLAB will be used extensively in all the labs. The primary goal of this lab is to familiarize a student 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. 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.

2.2 Movies: MATLAB Tutorials

On the t-square course page, there are a number of movies on basic topics in MATLAB, e.g., colon operator, indexing, functions, etc. You will find these movies useful.

2.3 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. 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. Two links of interest are **Getting Help** and **Getting Started with MATLAB**.

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

```
help
help edit
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. **`control-C`** will stop the execution of any MATLAB command. For example, using **`ctl-C`** while **`lookfor`** is running will force it to stop and print out all the results it has found so far.

e. Run the MATLAB demos: type **`demo`** and explore a variety of basic MATLAB commands and plots.

f. Use MATLAB as a calculator. Try the following:

```
pi*sqrt(2)
0.5*pi*pi - 1.5
sin(2*pi/3+0.1)
```

```
ans ^ 0.8118            %<--- "ans" holds the last result
```

g. Do variable name assignment in MATLAB. Try the following:

```
x = sin( 2*pi/5 );
cos( pi/4 );        %<--- assigned to what?
y = sqrt( 2.2 - x*x )
ans                 %<--- what value is contained in ans?
```

h. Complex numbers are natural in MATLAB. The basic operations are supported. Try the following:

```
z = -4 + 3i, w = 3 - 4j
real(z), imag(z)
abs([z,w])          %<-- Vector (or Matrix) constructor
conj(z+w)
angle(z)
exp( -j*pi )
exp(j*[ pi/2, 0, -pi/2 ])
```

3. Exercise

3.1 Logging into ITS (Intelligent Tutoring System)

Exercises on ITS will be integrated into homework assignments and grades this semester. The first thing you need to do in this lab is to try to log into ITS at **http://its.vip.gatech.edu** . Spend some time to familiarize yourself with ITS and its capabilities.

Verification Sheet Fill-in (separate page)

3.2 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 = -4 : 1
jkl = -2 : 3 : 24
jkl = 87 : -2 : 63
ttt = 2 : (1/8) : 3.1
tpi = pi * [ 0:1/3:2.5 ];
```

b. Extracting and/or inserting numbers in a vector is very easy to do. Consider the following definition of **xx**:

```
xx = [ zeros(1,5), linspace(0,1,6), ones(1,3) ]
xx(5:8)
size(xx)
length(xx)
xx(2:2:length(xx))
```

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

c. Observe the result of the following two assignments:

```
yy = xx; yy(3:2:7) = exp(0.6)*(1:2:5)
```

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 π^e. *Use a vector replacement, not a loop.* Record this statement on the Verification Sheet.

Verification Sheet Fill-in (separate page)

3.3 MATLAB Script Files

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

```
xk = cos( pi*(-1:2:25)/3 ); %<---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 **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=-50:50
yy(k+51) = exp( -(k/20).*(k/20) );
end
plot(yy)
```

Explain why it is necessary to write **yy(k+51)**. What happens if you use **yy(k)** instead? Also, explain the labels on the x-axis. This functional form is called a *Gaussian* form. Record your explanations on the Verification Sheet.

Verification Sheet Fill-in (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**. Try the following:

```
x = [-2.8 -1.2 0 1.2 2.8 ];
y = x.*x - 1.5*x;
plot( x, y )
z = x + y*sqrt(-1); % a complex number consists of two parts
plot( z )          % complex values: plot real vs. imaginary
```

Use **help arith** to learn how the operation **xx.*xx** works when **xx** is a vector; compare array multiplication (**dot-star**) to matrix multiplication.

When unsure about a command, use **help**.

d. Learn to do arithmetic in vector and matrix forms. Try the following:

```
x = [-2 -1 0 1 2];
y = x.*x + (x*x')*x;
plot( x, y )
```

Use **help arith** to further learn how the operation **xx*xx'** works when **xx** is a vector. The apostrophe means transpose of a vector, i.e., turning a row vector into a column vector and vice versa. It of course also works for a matrix. (Note that there are more than one "apostrophe"-looking symbol; make sure which one works in MATLAB.)

Verification Sheet Fill-in (separate page)

e. Use the built-in MATLAB editor to create a script file called **`mylab0.m`** containing the following lines:

```
tt = -0.5 : 0.005 : 0.5;
xx = 2.* cos( 4*2*pi*tt + 0.3);
plot( tt, xx, 'b-', tt, xx, 'r--' ), grid on %<--- plot a
                                                %sinusoid
title('TEST PLOT of a SIN WAVE')
xlabel('TIME (sec)')
```

Mark the value of **`xx`** on your plot when it intersects with the time axis at 0 sec.

Note: *Do not save* this file or any of your MATLAB files to the local hard disk. Your computer account contains a private networked directory where you can store your own files. Use the MATLAB command **`addpath()`** to allow MATLAB to "see" your personal directory (usually the **`Z:`** drive).

From the plot which should look smooth, find the values of **`tt`** at which the wave attains a peak value. Which of these locations (values of **`tt`**) is the closest to the origin and what is the peak value? Give your answers on the Verification Sheet.

Verification Sheet Fill-in (separate page)

f. Run your script from MATLAB. To run the file **`mylab0`** that you created previously, try

```
Mylab0              %<---will run the commands in the file
type mylab0         %<---will type out the contents of mylab0.m
                    %    to the screen
```

Lab #0
ECE-2026 Fall-2013
LAB SHEET

Turn this page in to your grading TA at the beginning of Lab 1

Name: _____ Date of Lab:_____

Part 3.1 Logging into ITS

☐ Yes, I have done so and am now familiar with ITS.

Part 3.2 Vector replacement using the colon operator:

Part 3.3(b) Write your vectorized statement below.

In addition, explain why it is necessary to write **`yy(k+51)`** in the loop. What happens if you use **`yy(k)`** instead? Explain the labels on the horizontal axis in the plot of the *Gaussian*. How would you change the plot command so that the extent of the x-axis would be from –40 to +40?

Part 3.3(d) Write out the result of **`y`**.

`y =`

Part 3.3(e) Give answers to the following:

Peak value =	**`tt`** =

Question 1-1:

Find the sum of all even-indexed output values generated by **`mylab0.m`**.