1 Matlab Review

All lab tasks in the ECE438 lab will be performed in Matlab. Matlab (matrix laboratory) is a technical computing environment for numerical analysis, matrix computation, signal processing, and graphics. The sections below will review some of its basic functions.

Some short tutorials are also available here .

2 Matlab Help

The following is a list of links to on-line Matlab help files:

audio	How to load, play and write audio
cputime	How to time a routine
fft	Computing the Fast Fourier Transform
filter	Apply a linear constant-coefficient difference equation
functions	Writing Matlab functions
hist	Obtain a histogram of data
image	Displaying Images in Matlab
load	Loading .mat files
mesh	3-D Plots of Matricies
meshgrid	Creating a grid for mesh
normpdf	Obtain the Gaussian density function
plot	How to plot vectors / CT-signals
print	How to print
random	Generating random sequences
firpm	Generate an equiripple filter
scripts	How to write .m-files
Simulink	Simulink
specgram	Obtaining a spectrogram
stem	How to obtain a stem plot of DT-signals
subplot	Plotting several functions in one figure

2.1 Starting Matlab and Getting Help

You can start Matlab (version 7.0) on your workstation by typing the command

```
matlab
```

in a command window. After starting up, you will get a Matlab window. To get help on any specific command, such as "plot", you can type the following

```
help plot
```

in the "Command Window" portion of the Matlab window. You can do a keyword search

Questions or comments concerning this laboratory should be directed to Prof. Charles A. Bouman, School of Electrical and Computer Engineering, Purdue University, West Lafayette IN 47907; (765) 494-0340; bouman@ecn.purdue.edu

for commands related to a topic by using the following

```
lookfor topic
```

You can get an interactive help window using the function

```
helpdesk
```

or by following the Help meun in the top of the Matlab window.

2.2 Matrices and Operations

Every element in Matlab is a matrix. So, for example, the Matlab command

```
a = [1 2 3]
```

creates a matrix named "a" with dimensions of 1×3. The variable "a" is stored in what is called the Matlab workspace. Alternatively, the operation

```
b = a.'
```

stores the transpose of "a" into the vector "b". In this case, "b" is a 3×1 vector.

Since each element in Matlab is a matrix, the operation

```
c = a*b
```

computes the **matrix** product of "a" and "b" to generate a **scalar** value for "c" of $14 = 1*1 + 2*2 + 3*3$.

Often, you may want to apply an operation to each element of a vector. For example, you many want to square each value of "a". In this case, you may use the following command.

```
c = a.*a
```

The dot before the * tells Matlab that the multiplication should be applied to each corresponding element of "a". Therefore the `.*` operation is *not* a matrix operation. The dot convention works with many other Matlab commands such as divide `./`, and power `.^`.

Note also that while the operation `a.'` performs a transpose on the variable "a", the operation `a'` performs a conjugate transpose on "a" (transposes the matrix and conjugates each number in the matrix).

2.3 Matlab Scripts and Functions

Matlab has two methods for saving sequences of commands as standard files. These two methods are called *scripts* and *functions*. Scripts execute a sequence of Matlab commands just as if you typed them directly into the Matlab command window. Functions differ from scripts because they take inputs and return outputs.

A script-file is a text file with the filename extension ".m" . The file should contain a sequence of Matlab commands. The script-file can be run by typing its name at the Matlab prompt without the .m extension. This is equivalent to typing in the commands at the prompt. Within the script-file, you can access variables you defined earlier in Matlab. All variables in the script-file are global, i.e. after the execution of the script-file, you can access its variables at the Matlab prompt. For more help on scripts **click here.**

To create a function call "func", you first create a file called "func.m". The first line of the file must be

```
function output = func(input)
```

where "input" designates the set of input variables, and "output" are your output variables. The rest of the function file then contains the desired operations. All variables in the function are local; that means the function cannot access Matlab workspace variables that you don't pass as inputs. After the execution of the function, you cannot access internal variables of the function. For more help on functions **click here.**