University of Illinois at Urbana-Champaign
Department of Electrical and Computer Engineering

ECE 311: Digital Signal Processing Lab
Chandra Radhakrishnan
Peter Kairouz

LAB 1: **Introduction to MATLAB**

Summer 2011

---

1 Overview

The goal of this lab is to get familiar with MATLAB and to learn how to represent, manipulate, and analyze discrete time signals in MATLAB. We will also look at some tools and commands to construct signals in MATLAB. Links to a number of resources and tutorials will be posted on the course web page. You can refer to these links for further reading.

2 Getting Started

- Get familiar with MATLAB by using tutorials and demos found in MATLAB. You can type `showdemo intro` to start a comprehensive help screen.
- At the prompt, type `help` followed by the command to get information about the command, try the following:

```
>>help plot
>>help stem
>>help cos
>>help sin
>>help exp
>>help for
>>help ones
>>help zeros
```

- What if you do not know a command name? A keyword search can be done by using the `lookfor` command.

```
>>lookfor frequency
```

- Variables in MATLAB can hold numbers, vectors or matrices:

```
>>x = randn(1,1)
>>x = randn(1,10)
>>x = randn(10,1)
>>x = randn(10,10)
>>x = rand(10,10)-0.5
```

You can see what `rand` and `randn` do using the `help` command!

- Putting a semicolon after a variable assignment prevents unnecessary echoing:

```
>>x = randn(10,10);
>>x = rand(10,10)-0.5;
```

- Result of a computation can be assigned to another variable. If the result is not assigned to a variable, MATLAB stores it in the variable `ans`

```
>>x = 0.25*0.25;
>>x
x=
      0.0625
>>0.25*0.25
ans =
      0.0625
```

- It is very easy to manipulate complex numbers in MATLAB. You can assign a complex number of the form `a+bi` directly to a variable and perform arithmetic operations:

```
>>x = 2+3i
>>y = 3+4i
>>x+y
ans =
      5.000 + 7.000i
```

Use `help` to see what the following functions do : `conj`, `abs`, `real`, `imag`.

- Loops in MATLAB: A simple `for` loop can be written as,

```
>> for cnt = 1 : 10
cnt
end
```

The loop prints out the values of `cnt`. Remember you can suppress the echo using semicolon. You can also use `while` loop instead of `for`. In general MATLAB is not very efficient with loops. We will see later that one can avoid loops by using the `semicolon` operator.

3 MATLAB functions for representing Signals

- The `colon` operator: The colon operator can be used to define a vector. Let us say we want to create a vector `x` which holds the integers from 0-100. One way is to use a loop. Another way to do this is shown below,

```
>> x = [0:100];
```

- Try the following,

```
>> x = [-0.5:0.1:0.5];
>> x = [0.5:-0.1:-0.5];
```

Use `help` to check what `linspace` does.

- Discrete-time complex exponentials form an important class of functions 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 $\alpha = e^{\pm i\omega}$,

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

MATLAB has functions `cos`, `sin` and `exp` to manipulate these signals.

- In general, signals are represented by row or column vectors, depending on the context. All vectors in MATLAB are indexed starting with 1. One may have to create an additional vector to properly keep track of signal index. Consider the following example,

$$x[n] = \begin{cases} 2n, & -3 \le n \le 3, \\ 0, & \text{otherwise} \end{cases}$$

  One can then do the following,

```
>> n=[-3:3]
>> x = 2*n;
```

- We can plot this signal using `stem` command,

```
>> stem(n,x)
```

- One can also extend the range of the sequences. For instance, if you want to plot the signal over the range $-5 \le n \le 5$, you can extend the index vector `n` and add additional elements to `x`,

```
>>n = [-5:5];
>>x = [0 0 x 0 0];
```

- What if want to extend the range by a large value? Use `zeros`:

```
>>n = [-100:100];
>>x = [zeros(1,95) x zeros(1,95)];
```

- We can define $x_1[n]$ to be the discrete-time unit impulse function and $x_2[n]$ to be the time-advances version of $x_1[n]$, i.e., $x_1[n] = \delta[n]$ and $x_2[n] = \delta[n+2]$. We can do this by,

```
>>nx1 = [0:10];
>>x1 = [1 zeros(1,9)];
>>nx2 = [-5:5];
>>x1 = [zeros(1,3) 1 zeros(1,7)];
```

- Let us now plot the signals,

```
>>stem(nx1,x1)
>>stem(nx2,x2)
```

- Discrete-time sinusoids and exponents can also be generated using `cos`, `sin`, and `exp`,

```
>>n = [0:32];
>>x = exp(i*(pi/8)*n)
```

Note the variable `x` is a vector of complex values now! Try the following commands,

```
>>stem(n,real(x))
>>stem(n,imag(x))
>>stem(n,abs(x))
>>stem(n,angle(x))
```

Note `angle` returns the phase in radians.

- MATLAB also allows you to add, subtract, multiply, divide, scale and exponentiate signals. let us define the signals `x1` and `x2`,

```
>>x1 = sin((pi/4)*[0:15]);
>>x2 = cos((pi/7)*[0:15]);
```

Try the following

```
>>y1 = x1 + x2
>>y2 = x1 - x2
>>y3 = x1 .* x2
>>y4 = x1 ./ x2
>>y5 = 2*x1
>>y6 = x1 .^2
>>y7 = x1 * x2
>>y8 = x1 * x2'
```

For multiplying, dividing, and exponentiating on a term by term basis, you must precede the operator with a period `.*` instead of `*`. Also note `x2'` converts the row vector `x2` into a column vector, i.e .it computes the hermitian transpose (conjugate transpose) of the argument. If you want to transpose `x2` without conjugating it use `x2.'`.

- MATLAB has several commands to help you label the plots appropriately, as well as to print them out. `title` places its argument over the current plot as the title. `xlabel` and `ylabel` allow you to label the axes. Every plot or graph you generate must have a title and the axes must be labeled clearly.

```
>>n = [0:32];
>>x = exp(i*(pi/8)*n);
>>stem(n,angle(x))
>>title('Phase of exp(i*(pi/8)*n)')
>>xlabel('n (Samples)')
>>ylabel('Phase of x[n] (radians)')
```

4 MATLAB scripts and functions

- MATLAB allows us to create `m-files`. There are two types of `m-files`: `scripts` and `functions`
- A command script is a text file of MATLAB commands whose filename ends in a **.m** in the current working directory or elsewhere in your MATLAB path. If you type the name of the file (without `.m`) at the command prompt the commands contained in the script file will be executed.
- The following `script` file generates a discrete time cosine signal. It then computes the mean of the signal and plots the signal. You can create a script file by using the MATLAB editor.

```
%example1.m
n = [0:16];
x1 = cos(pi*n/4);
y1 = mean(x1);
stem(n,x1)
title('x1 = cos(pi*n/4)')
xlabel('n (samples)')
ylabel('x1[n]')
```

Note `%` is used to comment in MATLAB. To execute the script file,

```
>>example1
```

- An `m-file` implementing a function is a text file with a title ending `.m` whose first word is a `function`. The rest of the line specifies the input and output arguments.
- The following `m-file` is a function called foo. It accepts input `x` and returns `y` and `z` which are equal to `2*x` and `5/9*(x-32)` respectively

```
function [y,z] = foo(x)

%[y,z] = foo(x) accepts a numerical argument x and
% returns two arguments y and z, where y is 2*x and z is (5/9)*(x-32)

y = 2*x;
z = (5/9)*(x-32);
```

Try the following

```
>> help foo
>>[y,z] = foo(-40)
>>[y,z] = foo(225)
```

5 Homework - Due 06/21/2011 at 5:00 PM

1. (a) On a single plot, representing the complex plane, graph:
   i. The complex number $z = 1 + 0.3j$ (mark the location with an asterisk, i.e. '*') (The commands **figure**, **plot**, **real**,and **imag** may be useful). Also give the magnitude and phase of this number (the commands **sqrt**, **abs**, **angle**, **exp**, and/or **hold** could come in handy).

ii. The complex number $exp(-(1+0.2j))$ (mark the location with 'o').

iii. The unit circle, by plotting 1001, (e.g., using the vector [0:1000]), complex points $exp(j\theta_k)$ for θ_k taking on 1001 values from 0 to 2π (The MATLAB command **axis equal** can make the circle).

(b) On the same graph, plot 100 samples, over $n = [0:99]$, of the signal $e^{0.025(1-j)n}$, marking each location with an **'x'**.

(c) In two new figures, plot the real and imaginary parts of these signal samples, respectively, using digital stem plot using the stem command. Label the **x-axis** as **'samples'** (**xlabel**) and the **y-axis** as **'real amplitude'** or **'imaginary amplitude'** (**ylabel**) as appropriate. What similarities and differences do you note between the real and imaginary parts?

(d) In Matlab, create a 100×100 matrix containing the magnitude of the complex function $f(z) = \frac{z-0.3}{z-0.85e^{\frac{j\pi}{4}}}$, for values of z over the ranges 0 to 1 in both the real and imaginary parts (that is, for all combinations of $\Re_e(z) = (0, 0.01, 0.02, \ldots, 0.99)$ and $\Im_m(z) = (0, 0.01, 0.020.99)$, and plot that using either the command mesh or surf, whichever you prefer. (The **for** and **zeros** commands may be useful.)

2. Consider the discrete-time signal

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

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 **stem** to create your plots, and be sure to appropriately label your axes. What is fundamental period of each signal? In general, how can the fundamental period be determined from arbitrary integer values of M and N?

3. Consider the signal

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

where $\omega_k = \frac{2\pi k}{5}$. For $x_k[n]$ given by $k = 1, 2, 4, and 6$, use **stem** to plot each signal on the interval $0 \leq n \leq 9$. All of the signals should be plotted with separate axes in the same figure using **subplot**. How many unique signals have you plotted? If two signals are identical, explain how different values of ω_k can yield the same signal.

4. Plot each of the following signals on the interval $0 \leq n \leq 31$:

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

What is the fundamental period of each signal? For each of these three signals, how could you have determined the fundamental period without relying upon MATLAB?

5. (a) Define a MATLAB vector nx to be the time indices $-3 \leq n \leq 7$ and the MATLAB vector x to be the values of the signal $x[n]$at those samples, where $x[n]$ is given by

$$x[n] = \begin{cases} 2, & n = 0, \\ 1, & n = 2, \\ -1 & n = 3, \\ 3 & n = 4, \\ 0 & \text{otherwise}, \end{cases}$$

Plot this discrete-time sequence by typing **stem(nx,x)**.

(b) Define vectors $y1$ through $y4$ to represent the following discrete-time signals:

$$y_1[n] = x[n-2]$$
$$y_2[n] = x[n+1]$$
$$y_3[n] = x[-n]$$
$$y_4[n] = x[-n+1]$$

To do this, you should define $y1$ through $y4$ to be equal to x. The key is to define correctly the corresponding index vectors $ny1$ through ny_4. First, you should figure out how the index of a given sample of $x[n]$ changes when transforming to $y_i[n]$. The index vectors need not span the same set of indices as nx, but they should all be at least 11 samples long and include the indices of all nonzero samples of the associated signal.

(c) Generate plots of $y_1[n]$ to $y_4[n]$ using **stem**. Based on your plots, state how each signal is related to the original $x[n]$, e.g. "delayed by 4" or "flipped and advanced by 3".

Deliverables

- Email your code, figures, calculation and answers as a *.pdf* or *.doc* file to **ece311lab.uiuc@gmail.com**. Be sure to name your document in the form- **ECE311Lab1_firstname_lastname.doc/pdf**.
- Late reports will reduce the grade by 20% per day.
- Make sure to present a clear and concise report having figures labeled and centered.
- **Reminder: Homework is due on 06/21/2011**