ECE 2610 Lab Worksheet: MATLAB Intro & Complex Arithmetic
1/24/2011

MATLAB as a Complex Number Calculator

- Functions used: real(), imag(), abs(), angle()
- Compare the three angle producing functions: angle(), atan2(), and atan()

Practice Problems (very similar to Set #1)
For each of the problem below work out the answer using both MATLAB and your calculator

1. Write \( z = 127 - j75 \) in polar form; find the angle in both radians and degrees.

\[
>> z = 127 - j75;
>> \text{abs}(z)
ans = 147.4924
>> \text{angle}(z)
ans = -0.5334 \quad \text{In radians}
>> \text{angle}(z)*180/pi
ans = -30.5640 \quad \text{In degrees}
\]

Hand/Calculator workspace:

From TI89 using the numerical evaluate mode (green diamond button, then press enter)

- Since this number lies in quadrant IV, we are OK using the \( \text{atan}( ) \) to find the angle
- Watch out if in quadrant II or III
- Best to plot the complex number as a vector

2. Write \( z = 22 \angle -110^\circ \) in rectangular form.

\[
>> z = 22*\text{exp}(-j*110*\text{pi}/180)
\]

\[
z = -7.5244 -20.6732i
\]

\[
>> \text{real}(z)
ans = -7.5244
>> \text{imag}(z)
ans = -20.6732
\]
3. Evaluate \( z = (15 - j37) - 60 \angle 45^\circ \) to a rectangular form solution.

**MATLAB Steps:**

\[
\begin{align*}
\text{ans} &= (15 - j*37) - 60*\text{exp}(j*45*\text{pi}/180) \\
&= -2.7426e+01 - 7.9426e+01i
\end{align*}
\]

4. Evaluate \( z = (15 - j37)/60 \angle 45^\circ \) to a rectangular form solution.

**MATLAB Steps:**

\[
\begin{align*}
\text{ans} &= (15 - j*37)/(60*\text{exp}(j*45*\text{pi}/180)) \\
&= -2.5927e-01 - 6.1283e-01i
\end{align*}
\]
MATLAB for Plotting Data and Functions

- Functions used: plot(), xlabel(), ylabel(), title(), grid, and axis

1. Plot \( x(t) = 25\sin(\pi t/5 + \pi/4) \) for \( 0 \leq t \leq 15 \) s. Include a grid and axis labels.

\[
\begin{align*}
\text{>> } & t = 0:.1:15; \% \text{ create a time axis vector with sample spacing 0.1s} \\
\text{>> } & \text{plot}(t,25*\sin(\pi t/5+\pi/4)) \\
\text{>> } & \text{grid} \\
\text{>> } & \text{xlabel('Time (s)')} \\
\text{>> } & \text{ylabel('Amplitude')} \\
\text{>> } & \text{print -depsc -tiff wks_fig1.eps}
\end{align*}
\]

![Plot of x(t) with grid and axis labels](image1)

For the \( x(t) \) above, plot \( x(t-2) \) for \( 0 \leq t \leq 15 \) s, overlaid on the plot of \( x(t) \) of part (1).

\[
\begin{align*}
\text{>> } & \text{hold on } \% \text{ will hold the previous plot so you can overlay a new plot} \\
\text{>> } & \text{hold} \\
\text{Current plot held} \\
\text{>> } & \text{plot}(t,25*\sin(\pi *(t-2)/5+\pi/4), 'r') \\
\text{>> } & \text{print -depsc -tiff wks_fig2.eps}
\end{align*}
\]

![Plots of x(t) and x(t-2) overlaid](image2)
User Defined Functions in MATLAB

One of the most power capabilities of Matlab is being able write your own user defined functions. Consider a custom trig function of the form

\[ y(t) = 3 \cos(t) + 4 \sin(3t) \]  \hspace{1cm} (1)

The input to this function is time, \( t \), and the output is \( y \). The function prototype we require is of the form:

```matlab
function y = my_trig(t)
% y = my_trig(t) is a function that evaluates the simple trig
% based function y = 3*cos(t) + 4*sin(3*t).
% 
% Author: My Name
% Date: January 2011
% 
% function body
% ... 
% make sure that you return output to variable y
```

Write the Function

```matlab
function y = my_trig(t)
% y = my_trig(t) 
% Mark Wickert, January 2011 
% 
% y = 3*cos(t) + 4*sin(3*t);
```

Test the Function

To test the function input a time vector that runs from -2s to 10s using a time step of 0.05s. Output the results in a plot using `plot(t,y)`.

```matlab
>> t = -2:.05:10;
>> y = my_trig(t);
>> plot(t,y)
>> grid
>> xlabel('Time (s)')
>> ylabel('Amplitude')
>> print -tiff -depsc fig1.eps
```