UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Department of Electrical and Computer Engineering

ECE 498MH Signal and Image Analysis

Lab 0

Fall 2014

Assigned: Thursday, August 28, 2014 Due: Thursday, September 4, 2014

Reading: MIT OCW Introduction to Matlab,
`http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-094-introduction-to-matlab-january-iap-2010/`

General Instructions:

(a) **On what will I be graded?** Each part of this lab ends by specifying one or more figures. You will receive **three points per lab for submitting your lab on time.** The other fifteen points per lab are given for the following:

(1) your "runlab.m" script generates the required figures,

(2) the figures show the correct results,

(3) your code is carefully commented, meaning that every function file has a file header specifying your name and the date, and meaning that your "runlab.m" script has labeled headers specifying which block of code implements each part of the problem, and

(4) if the instructor reads any labeled section of your code, he finds code performing the specified task (code that displays precomputed results is not acceptable for lab 1).

(b) **How will the instructor grade my lab?** He will unpack your zip-file or tar-file, cd into the resulting directory, and type `diary gradingdiary.txt; run;`. If nothing happens, you will receive a zero.

(c) **How should I submit my lab results?** Create a "runlab.m" script that generates required figures, and contains required comments. Put any other necessary files into the same directory, including any data files that we sent you (don't assume that the instructor has anything on his computer except matlab and the toolboxes). Zip, tar, or otherwise package the directory, and e-mail it to the instructor.

(d) **What happens if there is a bug in my code?** If your "runlab.m" script doesn't generate all of the figures and/or they're nor correct, the instructor will send you an e-mail specifying which part failed, and with the "gradingdiary.txt" file as an attachment. You are free to debug your code, and resubmit, with up to two resubmissions per lab, on any date prior to the last minute of the last lecture of the semester. If you think there is a hardware or version incompatibility between your PC and the instructor's PC, you may come to office hours and demonstrate the code on your own computer; if you convince him that you've done the work yourself, then he can give you full credit.

(e) **Can I use code I found on the web?** You can not use external add-on toolboxes. You can, however, cut and paste code snippets into your own script and function files, but you must add comments to your code that do the following things: (1) specify the start and end of the copied code, (2) specify where you got it, and (3) specify what it does, with sufficient detail to convince the instructor that you understand it.

Lab 0

(a) In order to create 10kg of knife steel, a particular factory needs 8kg of raw iron, 0.5kg of zinc, and 1.5kg of carbon. To create 10kg of stainless steel requires 7kg of irong, 2kg of zinc, and 1kg of carbon. To create 10kg of cast iron requires 9kg of iron, 0.25kg of zinc, and 0.75kg of carbon.

On January 1, this factor ordered 15kg of iron, 2kg of zinc, and 2kg of carbon from their suppliers. What were they planning to make?

Use `figure(1);` to create figure 1, then use the `stem` command to show your results. The first bubble should show the amount of knife steel being produced, the second bubble should show the amount of stainless steel being produced, and the third bubble should show the amount of cast iron being produced. Add a title (something like "Production Targets," maybe) and axis labels (something like `xlabel('1=knife, 2=stless, 3=cast');ylabel('Target (kg)');`, perhaps). In general, plots without title and labels will be marked wrong.

(b) Create ten half-second waveforms, each playing a pure tone at a sampling rate of $F_s = 8000$ samples/second. The ten notes should be A1, A2, A3, A4, A5, A6, A7, A8, A9, and A10 (the last three notes don't exist on a standard piano).

Your "runlab.m" script should play these ten notes in order, at an 8kHz sampling frequency. For example, after you have appropriately defined the variables `FS` and `f`, you could use `for k=1:10, t=[0:(1/FS):0.5]; x=sin(2*pi*f(k)*t); input(sprintf('Hit enter to hear the tone at %d Hz',f(k))); soundsc(x,FS); end`, where the `input` command is useful primarily because it keeps matlab from playing all 10 notes at the same time.