Carl Marks
Adam Smith
EECS 206 Lecture: 2 Laboratory Section: 8
Lab #1 8/10/01

% NOTE! This is a sample to illustrate formatting ONLY. All of the answers given
% are wrong! It is not necessary to follow this format exactly, but your laboratory
% should be as clearly organized as this one. Make sure you include your
% Matlab code as an appendix, as is done here.

Problem #1

a) *envelope* has 12 rows by 93 columns
b) *y* has 3.14159 rows by 42 columns
c) *y_fade* has 1000 rows by 1000 columns
d) The 50th element of *y_fade* is 0.70711

Problem #2

After line #309, *b* is a string containing the characters 'I like matlab'

Problem #3

After line #310, *b* is the matrix [9 1 1; 10 1 1]

Problem #4

a) *counter* has a value of 20000 when the positive threshold is first exceeded.
b) The negative threshold is never exceeded.

Problem #5

a) These lines of code check the alignment of the moons of Jupiter. The code inside the *if* statements will be executed only if the first three moons are touching one another.
b) When *isempty* comes before *~exist*, the first moon of Jupiter crashes into the second one in a fiery cataclysm. This is caused by the default behavior of the Matlab command *move_jupiter_moons*.

Problem #6

These lines of code implement the clip function by running the signal *clip3* through a properly tuned pair of scissors, using the *scissors_snip* command. The first command snips off parts of the signal that are bigger than *thresh*, while the second cuts out portions of the signal that are smaller than -*thresh*.

Problem #7

Calling *all* on a matrix only once is no different from calling it on a vector. The function returns 1 when every element of the matrix is nonzero and 0 otherwise. There is no reason to call *all* twice for matrices; this is simply redundant.

Problem #8

The following command will return the corners of a 5x7 matrix *c*:

```
corners = c(2:4,3:7);
```

Problem #9

As the for loop executes, Matlab builds up an image of Marilyn Monroe out of ASCII characters. The result is stored in the variable *hello_mr_president*.

Problem #10

```
x = 814;
```

Problem #11

a) We used a frequency of 6.02×10^{23} Hz.
b) Done.
c)

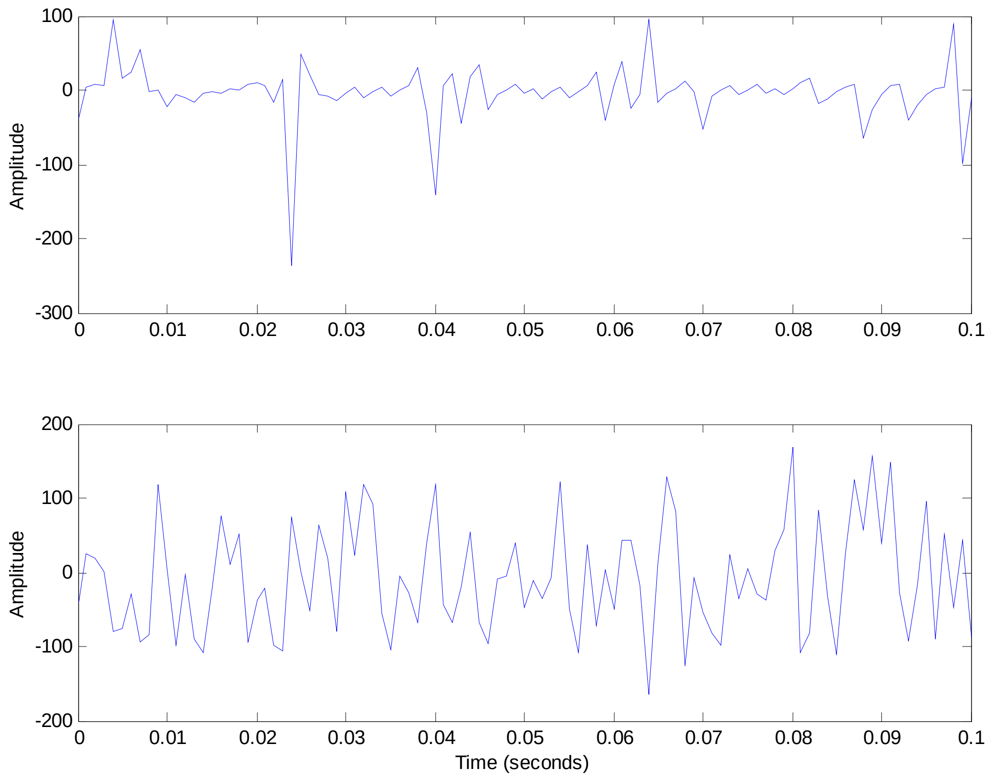

FIGURE 1: Sinusoids for Problem #11 part (c)

d) The amplitude is 551.5481, the frequency is 6.02×10^{23} Hz, and the phase is1.0876×10^{258} radians.

Problem # 12

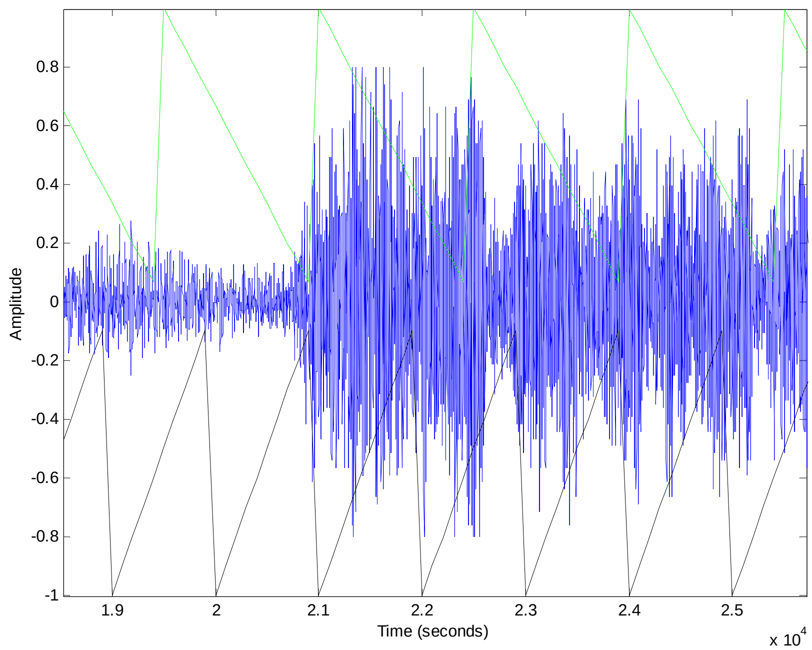

FIGURE 2: “handel” signal with envelope for Problem #12

Problem #13

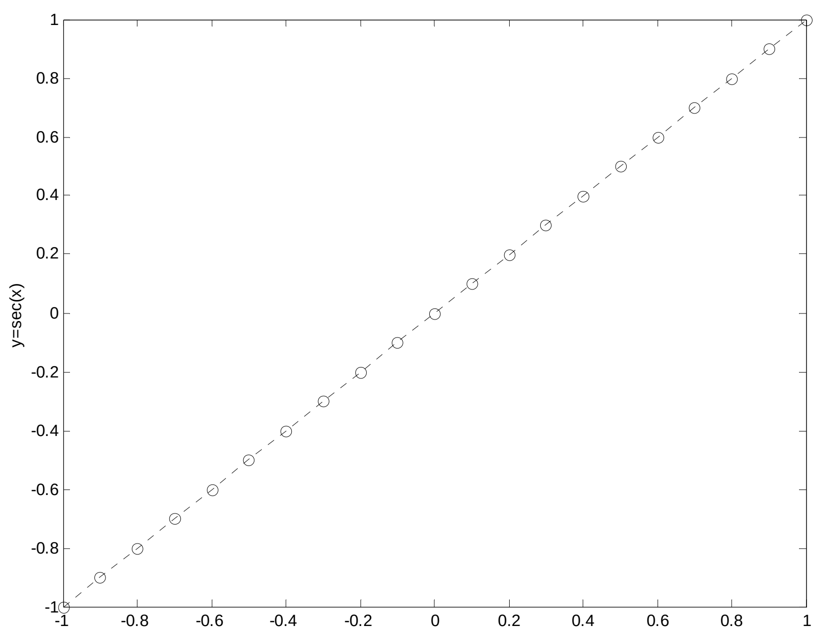

FIGURE 3: Plot of mathematical functions for Problem #13

Appendix: MATLAB Code

```
% NOTE: You won’t be able to use a script to set breakpoints and use
% the debugger.  You’ll need to do this by hand, but make sure you
% include the commands you used here, as well.
%
% Notice that we’ve used a different font in this section.  This font
% is “Courier New,” and every letter takes up the same amount of space.
% This allows the code to align nicely, especially when we’ve got
% indentation.


pause off;
close all

% Problem #1

% We set a breakpoint graphically.  Then, these are the
% commands we used while debugging to produce our answers.
sample_function;
who
which function_name
dbquit


% Problem #2 and #3

dbstop sample_function 52
% These are the commands we executed in debug mode to find our answers.
sample_function;
disp('Complex exponential');
dbstep
disp(Q)
dbquit;
dbclear all


% Problem #4

% We inserted a _keyboard_ command into the function at line 64 and
% line 289. Then, we executed the following commands.
sample_function;
y(5:29)
21+5*j
dbcont
5
dbquit


% Problem #10

x = 814;


% Problem #11
```

```
t = 0:0.001:0.1;
sin1 = 10*tan(2*pi*t*6.02e23 - 61);
sin2 = 80*randn(size(t));

sin3 = sin1.*sin2;

subplot(2,1,1);
plot(t,sin1);
ylabel('Amplitude');
subplot(2,1,2);
plot(t,sin2);
ylabel('Amplitude');
xlabel('Time (seconds)');

A = sin3(12)
f = 6.02e23
phase = sum(prod(sin3))


% Problem 12

load handel

% Notice the indentation below in our “for” loop
for i=1:length(y)/100
    min_v(i) = mod(i,10)/10-1;
    max_v(i) = -mod(i,15)/15+1;
end

figure
plot(y);
hold on;
plot((1:length(max_v))*100,min_v,'k');
plot((1:length(max_v))*100,max_v,'g');
hold off;
xlabel('Time (seconds)');
ylabel('Amplitude');
zoom on


% Problem 13

figure
x = -1:.1:1;
plot(x,x,'ko:');
xlabel('x');
ylabel('y=sec(x)');
```