EE4252 LAB 1: INTRODUCTION TO MATLAB

1 Introduction

You will use MATLAB for all lab projects in EE4252. You can access MATLAB from the start menu of the PC or seek help from your Lab TA.

1.1 The ADSP Toolbox

You will be using the EE toolbox in MATLAB (which is actually the ADSP toolbox from Ambardar's text). It comprises a suite of MATLAB m-files for signal and system analysis and a set of graphical user interface (GUI) routines for easy visualization of basic signal processing concepts. The GUIs require no prior MATLAB skills, and their common features include a point-and-click interface, simple data entry, a help menu, plot and axis control, and cursor control.
Please download the zip files `DataFiles.zip`, `adspmfiles.zip` and `adspguis.zip` from
`http://www.ece.mtu.edu/faculty/akambard/book/text.html`
Create a folder, unzip the files to this folder and make sure the folder is in the MATLAB path.

1.2 When in MATLAB

To exit MATLAB, type `quit` at the MATLAB prompt `>>`.

To seek help on a MATLAB command, say `stem`, type `help stem`

To access the on-line documentation, type `doc`. This spawns a window with menu driven help

For a tour of MATLAB demos, type `expomap` (or `demo`).

For a tour of m-files in the ADSP toolbox, type `tour`

Most ADSP m-files are self-demonstrating if you type the filename (with no arguments) at the MATLAB prompt. For example, to seek help on the m-file `sysresp1.m`, type `sysresp1`. This will bring up a help screen and demonstrate a simple example of usage. You may also use the help facility of MATLAB by typing `help sysresp1`. In this case, you will only see the help screen but no example of usage.

For a tour of all the GUI (graphical user interface) programs, type `tourgui`

For a tutorial on basic MATLAB commands, type `tutorial`

1.3 Some General Guidelines

1. To exit MATLAB, type `quit` at the MATLAB prompt.
2. MATLAB works in double precision. It is case-sensitive. All built-in commands are in lower case.
3. Use `format long` to see results in double precision, and `format` to get back to the default display.
4. In MATLAB, the indexing starts at 1 (not 0).
5. A semicolon (;) after a statement or command suppresses on-screen display of results.
6. To abort an executing MATLAB command, or on-screen data display, use Ctrl-C.
7. Use the up/down "arrow keys" to scroll through commands and re-edit/execute them.

1.4 Array Operations

Most operations you will perform will be element-wise operations on arrays. Be aware of the following:

```
x=0:0.01:2; % Generate an array x at intervals of 0.01.
y=x .^ 2;   % Generate array of squared values; uses " .' "
z=x.';      % Transpose x from a row to column or vice versa; uses " .' "
x.*y;       % Pointwise product of equal size arrays x and y; uses " .* "
x./y;       % Pointwise division of equal size arrays x and y; uses " ./ "
l=length(x) % Returns the length of an array or vector x.
p=x(3)      % Third element of x (MATLAB indexing starts with 1)
```

2 A Tiny List of Useful Commands

Some of the MATLAB commands for mathematical operations are `exp, sin, cos, tan, acos, asin, atan, pi` (for π), `eps` (for default tolerance = $(2.22)10^{-16}$), `i` or `j` (for $\sqrt{-1}$), `real, imag, sqrt` (for square root), `abs` (for absolute value or magnitude), `angle` (for the angle or phase in *radians*), `log2` and `log10` (for logarithms to the base 2 or 10), `length` (of a vector or array), `size` (of a matrix) etc. Other useful commands are `min, max, int, fix, find, conj, sum, cumsum, diff, prod, eval, fliplr, roots` and `sign`. Use the help facility to learn more about them.

2.1 String Functions and Logical Operations

A string function (symbolic expression) is entered in single quotes, for example `x = '4*t.*exp(-2*t)'`. MATLAB displays it on screen without quotes. To use its results, or plot the expression, it must first be evaluated using `eval`. For example:

```
x = '4*t.*exp(-2*t)'     % String or symbolic expression
t=0:0.05:4;              % Pick a time range
xe = eval(x);            % Evaluate the expression x
y = 4*t.*exp(-2*t)       % Numeric result (only after array t is defined).
plot(t,xe)               % Plot
```

Logical operations are useful in many situations. Here are some examples:

```
i=find(xe>=0.7);         % Index of values in xe that equal or exceed 0.7
iz=i(1);                 % Index of first such value in i
tz=t(iz);                % Approximate time of first zero
k=find(xe==max(xe));     % Index of maximum in xe. Note the ==
tm=t(k);                 % Time(s) of maximum
```

2.2 Saving a MATLAB session

You can cut-and-paste commands or statements from the MATLAB window to a text editor.
You may also use the `diary` command. In a MATLAB window, type

```
diary mywork.m     % Example:  Open a file which will be called mywork.m
```

This opens a text file called `mywork.m`. All subsequent MATLAB on-screen statements and displays are appended to this file. To close the file, type `diary off`. To add more statements, type `diary on`. You can alternate between the `diary on` and `diary off` commands.

3 Frequently Asked Questions

Q. How many data points do I need to obtain a smooth curve when plotting analog signals?
A. A good rule of thumb is at least 200 points.

Q. Why do I get an error while using `plot(t,x)`?
A. Both `t` and `x` must have the same dimensions (size). To check, use `size(t)` and `size(x)`.

Q. Why do I get an error when generating $x(t) = e^{-t}\sin(2t)$?
A. You probably did not use array operations (such as `.*` or `.^`).

Q. How can I plot several graphs on the same plot?
A. Use `>>plot(t,x),hold on,plot(t1,x1),hold off` OR `>>plot(t,x,t1,x1)`

Q. How do I use `subplot` to generate 6 windows (2 rows of 3)?
A. Use `>>subplot(2,3,n)` to plot in the nth (count by rows) window. Note that `n` goes from 1 to 6. Note also that each subplot has its own xlabel, ylabel, axis, title, etc.

3.1 Some Guidelines on Plotting

Here are some examples of how to plot continuous and discrete signals and add some bells and whistles:

```
t=-10:0.1:10;xc=2*sinc(t/5);      % Create a time array and signal
n=-10:10;xd=2*sinc(n/5);          % Create a discrete index and a discrete signal
subplot(2,1,1);plot(t,xc)         % Split the screen, plot in top window
axis([-5 5 -1 3]);grid on         % Set x-limits (-5, 5),y-limits (-1, 3),grid
xlabel('Time in seconds')         % Label for x-axis.  Note single quotes
ylabel('Amplitude')               % Label for y-axis.  Title has similar format
subplot(2,1,2); dtplot(n,xd,'.')  % Discrete plot in second window
print                             % Hardcopy printout of the screen
```

4 Project 1: Continuous Systems

1. Consider a continuous system whose transfer function is $H(s) = \dfrac{1}{s+4}$.
   - **(a)** Use `ctfgui` to display the frequency response over $0 \leq f \leq 5$ Hz. Save all variables including the **impulse response** $h(t)$ and **step response** $s(t)$ (these two will be saved as string variables in *symbolic* form).
   - **(b)** In MATLAB, use the `eval` command to evaluate $h(t)$ and $s(t)$ obtained from `ctfgui` over $0 \leq t \leq 4$ and overplot. Add appropriate labels, legends, a title etc. Obtain a hardcopy.
   - **(c)** In MATLAB, use the variables saved from `ctfgui` to plot the magnitude of $H(f)$ over the frequency range $0 \leq f \leq 5$ Hz. Obtain a hardcopy.
   - **(d)** Use `bodegui` to display the Bode magnitude plot for $H(s)$. Obtain a hardcopy.
   - **(e)** Use the ADSP routine `trbw` to find the rise-time and bandwidth of the system. Use these values to compute the **rise-time–bandwidth product**.

5 Project 2: Discrete Systems

1. Consider a *relaxed* (zero initial conditions) digital filter described by $y[n] - 0.5y[n-1] = 2x[n]$.
   - **(a)** Use `dtsimgui` to find and save expressions for its step response $s[n]$, and its response to the signal $x[n] = (0.4)^n u[n]$. In MATLAB evaluate these responses over $0 \leq n \leq 10$ and plot in two subplots.
   - **(b)** The numerical solution of a *relaxed* difference equation may also be obtained from the MATLAB routine `filter`. Type `>>help filter` to learn its syntax. Use `filter` and `dtplot` to obtain and plot the step response over $0 \leq n \leq 10$. Obtain a hardcopy. Compare the values obtained from `filter` with the values obtained from `dtsimgui`.

2. Consider the *relaxed* second order digital filter described by $y[n] - 1.4y[n-1] + Cy[n-2] = x[n]$. Use `dtsimgui` for the following problems.
   - **(a)** Plot the step response over $0 \leq n \leq 60$ for $C = 0.5,\ 0.6,\ 0.7,\ 0.8$. What final value does the response settle to for each choice of C? For which values of C is the response overdamped? underdamped? critically damped? Obtain a hardcopy only for $C = 0.5$ and $C = 0.7$.
   - **(b)** Plot the step response over $0 \leq n \leq 60$ for $C = 0.4,\ 0.3,\ 0.2$. Describe the nature of the response as you decrease C.
   - **(c)** The *characteristic equation* for this system is $z^2 - 1.4z + C = 0$. The system is stable if *all* **root magnitudes** of the characteristic equation are **less than unity**. For which values of C (chosen above) is the system unstable?

REPORT

Turn in a report (with this sheet stapled on top) on each project including:

1. All computations and a careful **discussion** of the results, not to exceed 2 pages.
2. All **relevant plots/tables** requested in the problems, with appropriate titles, labels, etc.
3. An appendix containing your MATLAB **code with comments and explanations**.

NOTE: HANDWRITTEN WORK IS NOT ACCEPTABLE

Lab 1 Report

Introduction to MATLAB

Alex Hirzel

Submitted to Yang Liu for EE4252

Due September 27, 2012

Michigan Technological University

Introduction

This report presents the information from Lab 1, which consists of the analysis of the following three systems:

$$H(s) = \frac{1}{s+4} \qquad y[n] - 0.5y[n-1] = 2x[n] \qquad y[n] - 1.4y[n-1] + Cy[n-2] = x[n]$$

Project 1

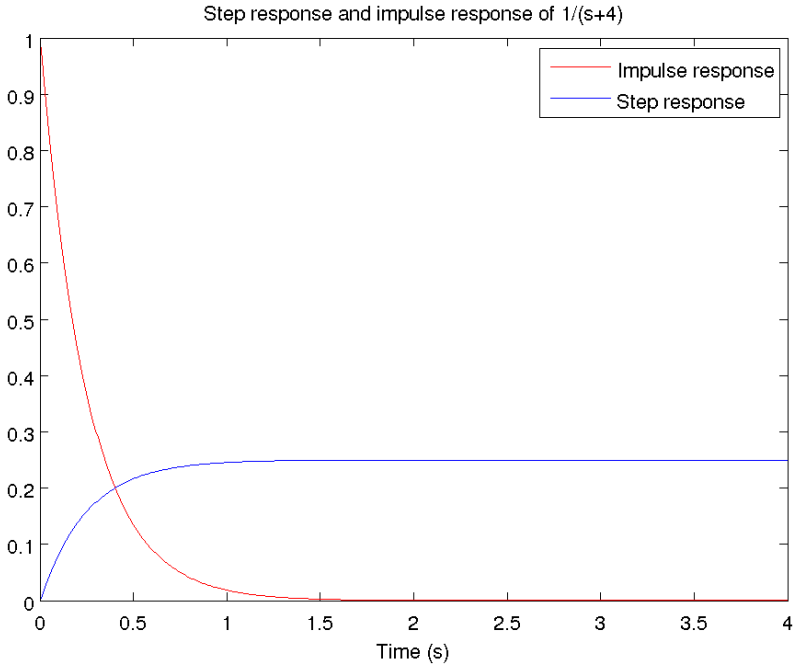

Figure 1: Step response and impulse response of $1/(s+4)$ as generated for question 1(b).

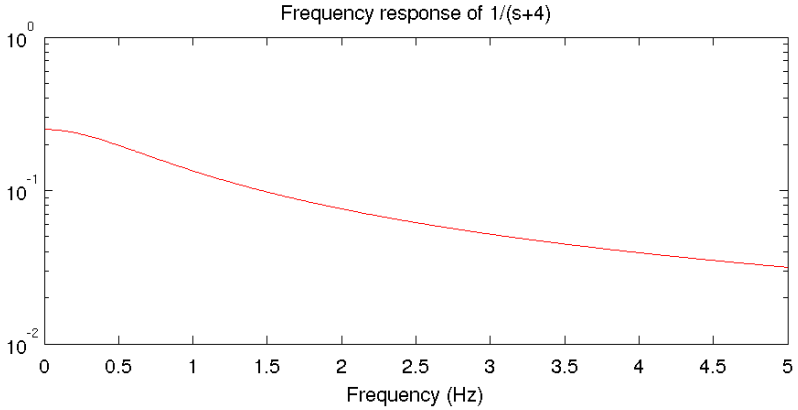

Figure 2: Frequency response of $1/(s+4)$ as generated for question 1(c).

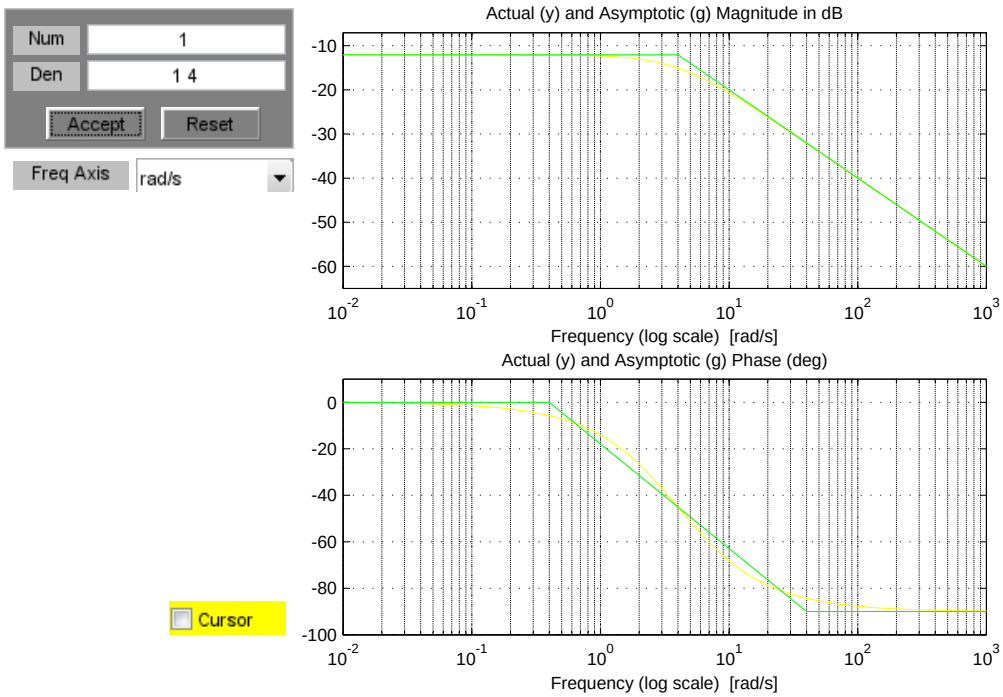

Figure 3: Bode plot of $1/(s+4)$ as captured for question 1(d).

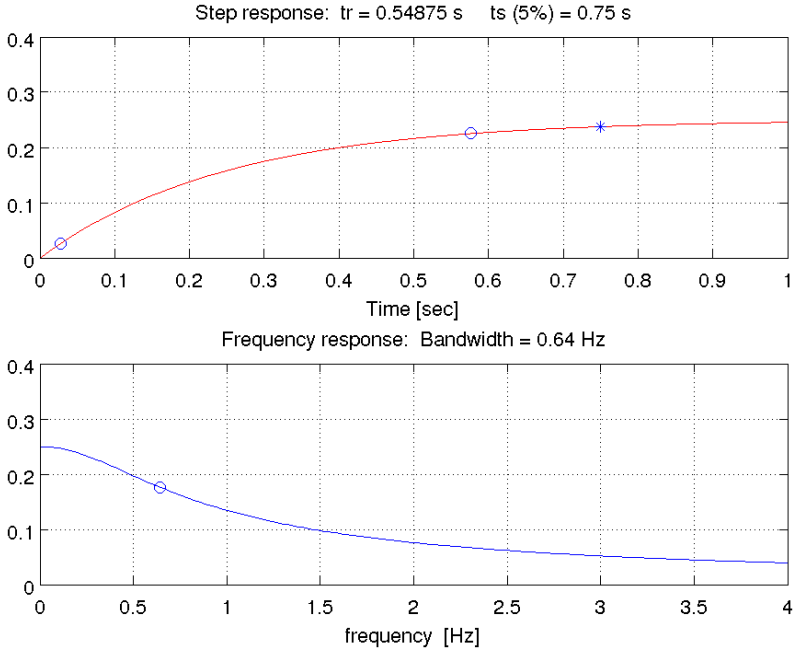

Figure 4: Plot of $1/(s+4)$ as generated from `trbw` for question 1(e). The rise-time–bandwidth product is $0.64\text{Hz} \cdot 0.54875\text{s} = \boxed{0.3512}$.

Project 2

Two systems are analyzed in this project:

$$y[n] - 0.5y[n-1] = 2x[n] \tag{1}$$
$$y[n] - 1.4y[n-1] + Cy[n-2] = x[n] \tag{2}$$

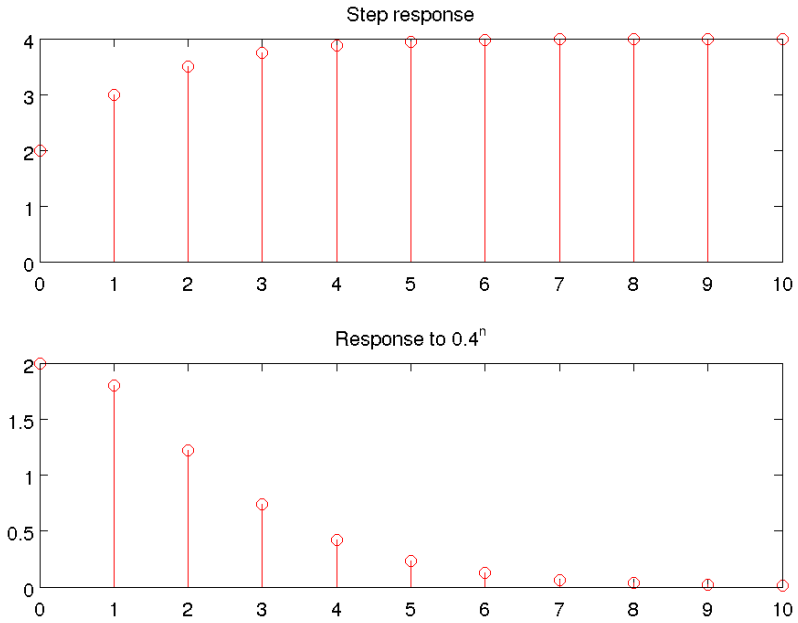

Figure 5: The step response and response to $0.4^n u(t)$ of system (1) as generated by `dtsimgui` for question 1(a).

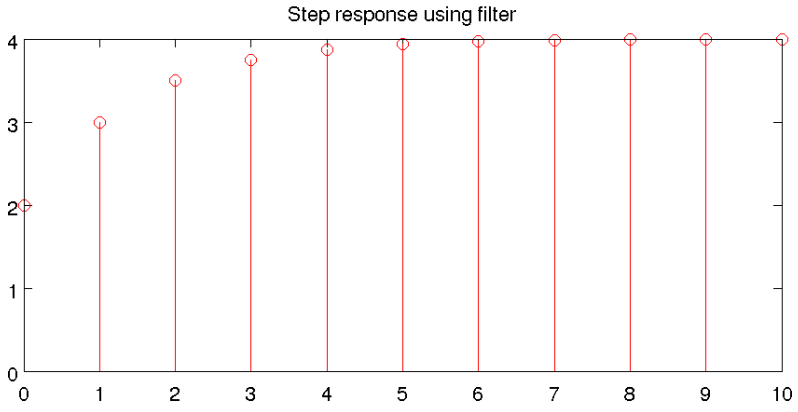

Figure 6: The step response of system (1) as generated by `filter` for question 1(b).

Shown in figures 5 and 6 (respectively) are the responses of system (1) to a step input. These two figures are in exact agreement on the step response, which is to be expected.

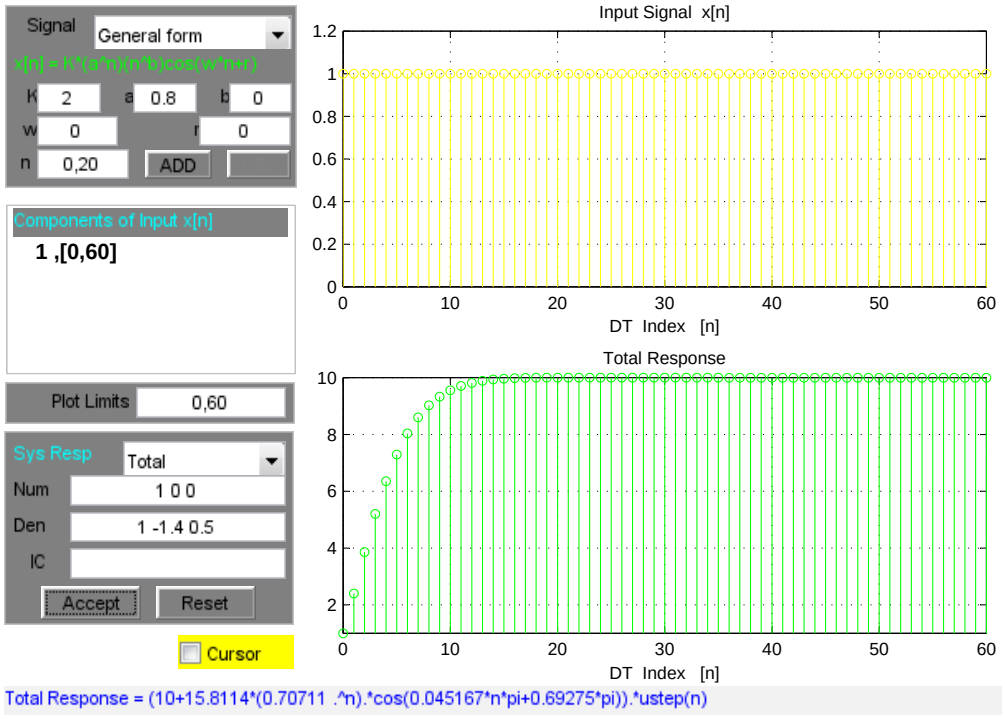

Figure 7: Hard copy of `dtsimgui` GUI for $C = 0.5$.

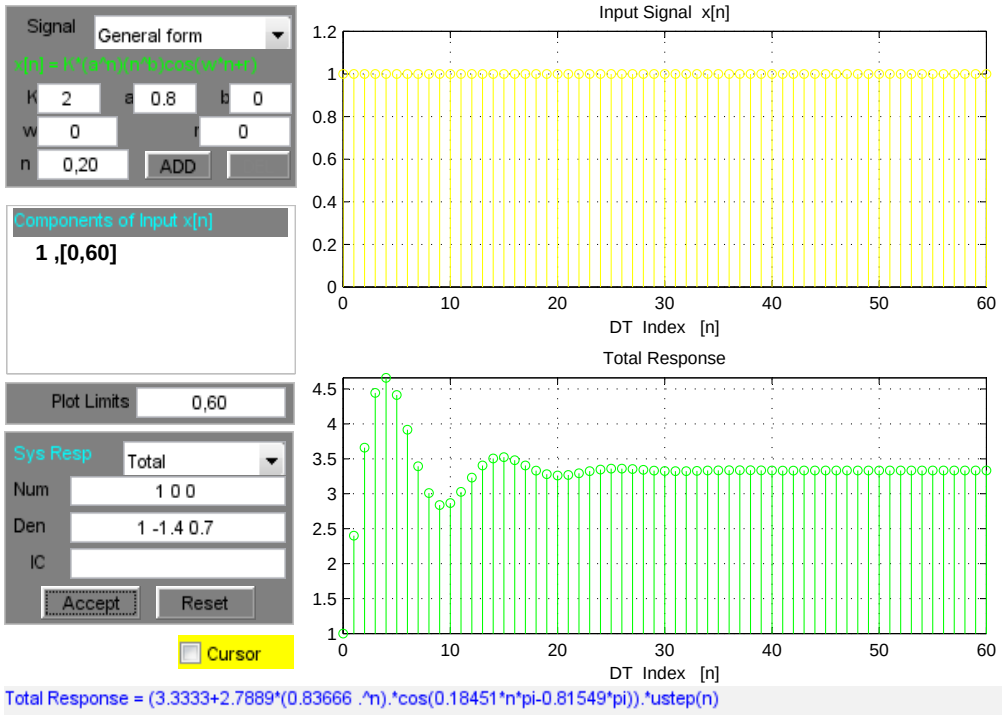

Figure 8: Hard copy of `dtsimgui` GUI for $C = 0.7$.

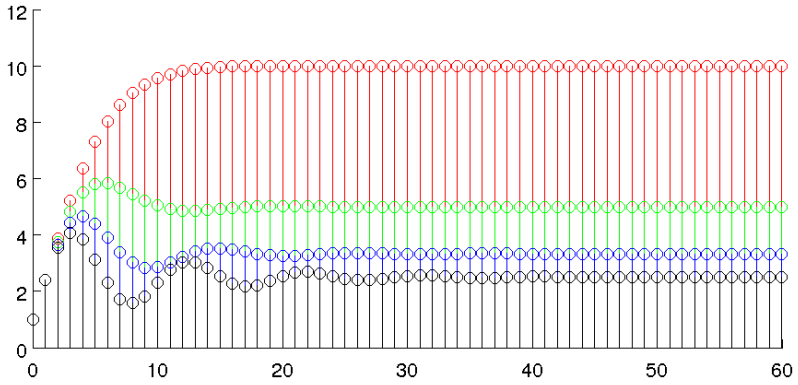

Figure 9: Step responses of system (2) as generated for question 2(a). Shown are step responses—from top to bottom—for $C = 0.5$ (red), $C = 0.6$ (green), $C = 0.7$ (blue) and $C = 0.8$ (black).

Shown in many preceeding figures are the responses of system (2) to a step input. Table 1 shows the final value taken on by the system as a function of the constant C. This varying coefficient allows the exploration of the stability of the filter because for some values of C the filter is both stable and divergent.

Table 1: Final value of the step response of $y[n] - 1.4y[n-1] + Cy[n-2] = x[n]$ for varying C.

C	Final value
0.5	10
0.6	5
0.7	3.3334
0.8	2.5027

The characteristic equation $z^2 - 1.4z + C = 0$ of system (2) has roots according to the quadratic equation[1]. These roots are less than unity when

$$\left|\frac{1.4 \pm \sqrt{1.96 - 4C}}{2}\right| < 1 \quad \Rightarrow \quad \boxed{C > 0.4 \text{ or } C > 0.4}.$$

It is under the opposite of this condition that the system is unstable. That is to say: when $C < 0.4$, the system is unstable and when $C = 0.4$ the system is marginally stable. This stability criterion is rooted in the iterative nature of the filter. Specifically, looking at the difference equation form of the system,

$$y[n] = x[n] - 1.4y[n-1] + Cy[n-2]$$

it can be inferred that if approximately steady-state conditions are assumed—specifically that $y[n]$ is not changing much during each iteration (making $y[n-1] \approx y[n-2]$) and $x = 1$ (because a step response is under consideration)—then the final value of $y[n]$ is approximated by:

$$\begin{aligned} \tilde{\tilde{y}}[n] &= 1 - 1.4y[n-1] + Cy[n-1] \\ &= 1 - (1.4 - C)y[n-1] \end{aligned} \tag{3}$$

which will grow without bound unless $1.4 - C < 1$ (i.e. $C > 0.4$) due to the recursive nature of the filter. As can be seen in (3), if $C = 0.4$, then $y[n] = 1 - y[n-1]$. This implies marignal stability about $y[n] = 1$ with $C = 0.4$.

[1] $ax^2 + bx + c = 0 \iff x = (-b \pm \sqrt{b^2 - 4ac})/(2a)$.

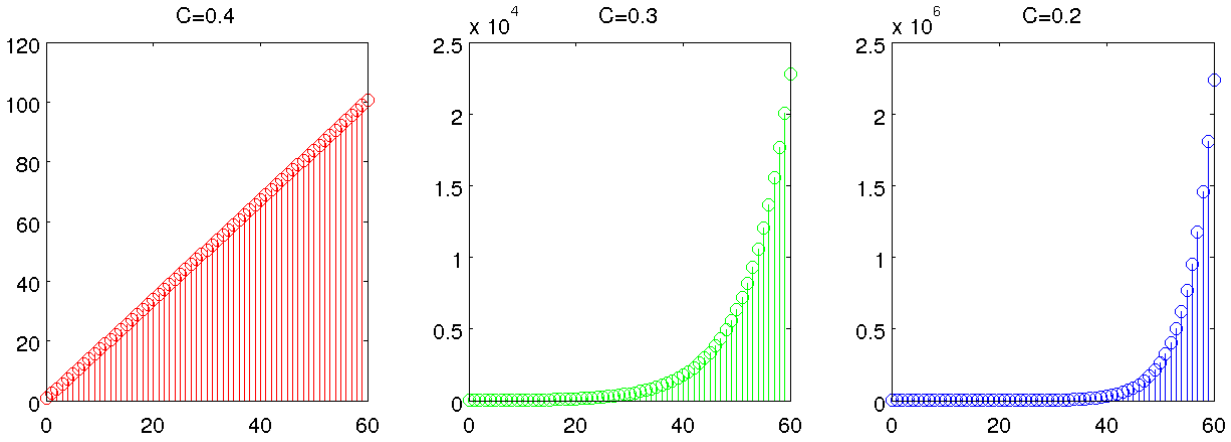

Figure 10: Step responses of system (2) as generated for question 2(b). Shown are step responses—from top to bottom—for $C = 0.4$ (red), $C = 0.3$ (green) and $C = 0.2$ (blue).

The same story is told by Figure 10, which shows the step response with $C = 0.4, 0.3, 0.2$. The left-most plot ($C = 0.4$) shows marginal stability in the form of a divergent, perfectly linear output. Decreasing values of C force more rapid divergence.

Appendix: MATLAB Source Code

What follows is a listing of the MATLAB source code (listing 1)—and the output of this code (listing 2)—used to generate the figures and other information presented in this report.

Listing 1: The MATLAB script used for this report, `Lab01_ahirzel.m`.

```
% EE4252 Lab 1
% Alex Hirzel <ahirzel@mtu.edu>
% 2012-09-20
% Requires data saved from the GUIs in project[12]_data.mat

addpath ../../ClassWorkspace;
load('project1_data');
load('project2_data');

% Part 1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
t = 0:0.01:4; % seconds
f = 0:0.01:5; % Hz

plot(t, eval(h), 'r', t, eval(s), 'b')
xlabel('Time (s)')
title('Step response and impulse response of 1/(s+4)')
legend('Impulse response', 'Step response')
mysaveas('p1_step_impulse', 7, 5)

semilogy(f', Hmag, 'r')
xlabel('Frequency (Hz)')
title('Frequency response of 1/(s+4)')
mysaveas('p1_frequency', 7, 3)

trbw([1], [1 4], 5);
mysaveas('p1_rise_time', 7, 5)

% Part 2.1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
t = 0:10;

subplot(2, 1, 1)
dtplot(t, step, 'o')
title('Step response')
subplot(2, 1, 2)
dtplot(t, resp, 'o')
title('Response to 0.4^n')
mysaveas('p2_1_responses', 7, 5)

dtplot(t, filter([2], [1 -0.5], ustep(0:10)), 'o')
title('Step response using filter')
mysaveas('p2_1_filter_response', 7, 3)

% Part 2.2 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mycolors = ['r'; 'g'; 'b'; 'k'];

for i = 4:-1:2
    d = eval(['p22c' num2str(i)]);
    hold on;
    subplot(1, 3, 5-i);
    dtplot(0:60, d, 'o', mycolors(5-i));
    title(['C=0.' num2str(i)])
    xlim([0 60])
end
mysaveas('p2_1_unstables', 10, 3)

delete 'diary.txt'; diary 'diary.txt'; diary on
hold on
for i = 5:8
    d = eval(['p22c' num2str(i)]);
    disp(['Final value when C = 0.' num2str(i) ' is ' ...
        num2str(d(end))])
    dtplot(0:60, d, 'o', mycolors(i-4));
end
diary off
mysaveas('p2_1_final_values', 7, 3)
```

Listing 2: The output of listing 1, `diary.txt`.

```
Final value when C = 0.5 is 10
Final value when C = 0.6 is 5
Final value when C = 0.7 is 3.3334
Final value when C = 0.8 is 2.5027
```