MATLAB Assignment 1

Alex Benasutti

ELC 321 / Signals and Systems

April 16th, 2019

TCNJ

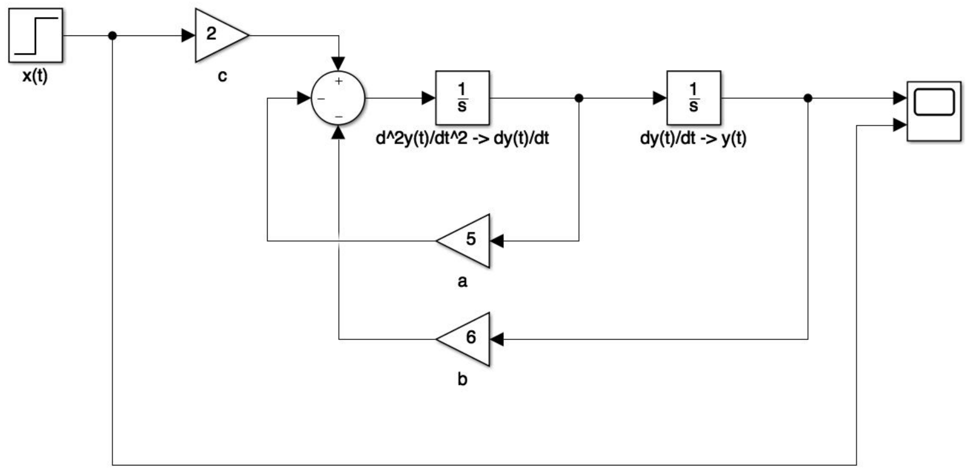
THE COLLEGE OF NEW JERSEY

SCHOOL OF ENGINEERING

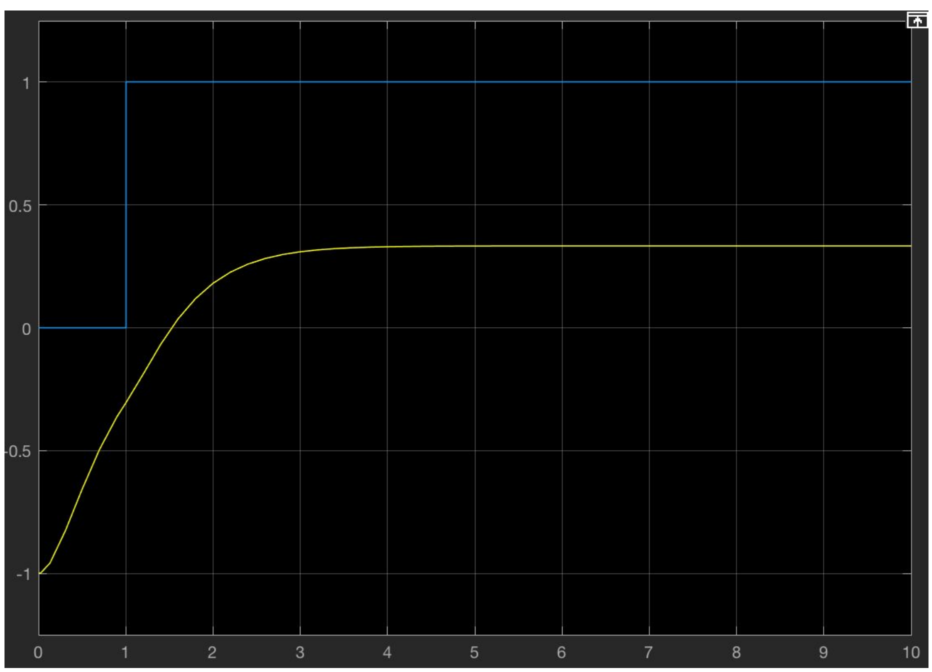
Results

Problem 1.

Simulation Diagram



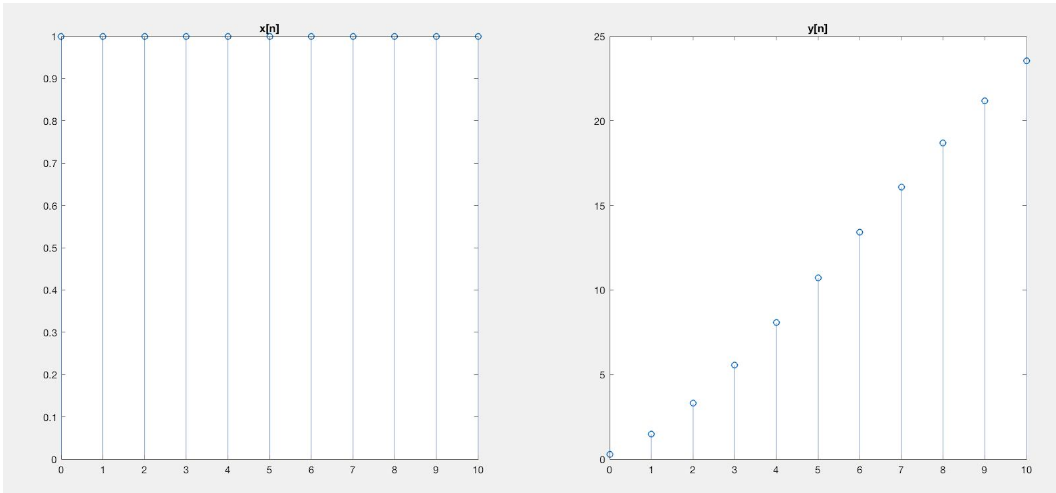
Output waveforms (blue = x(t), yellow = y(t))



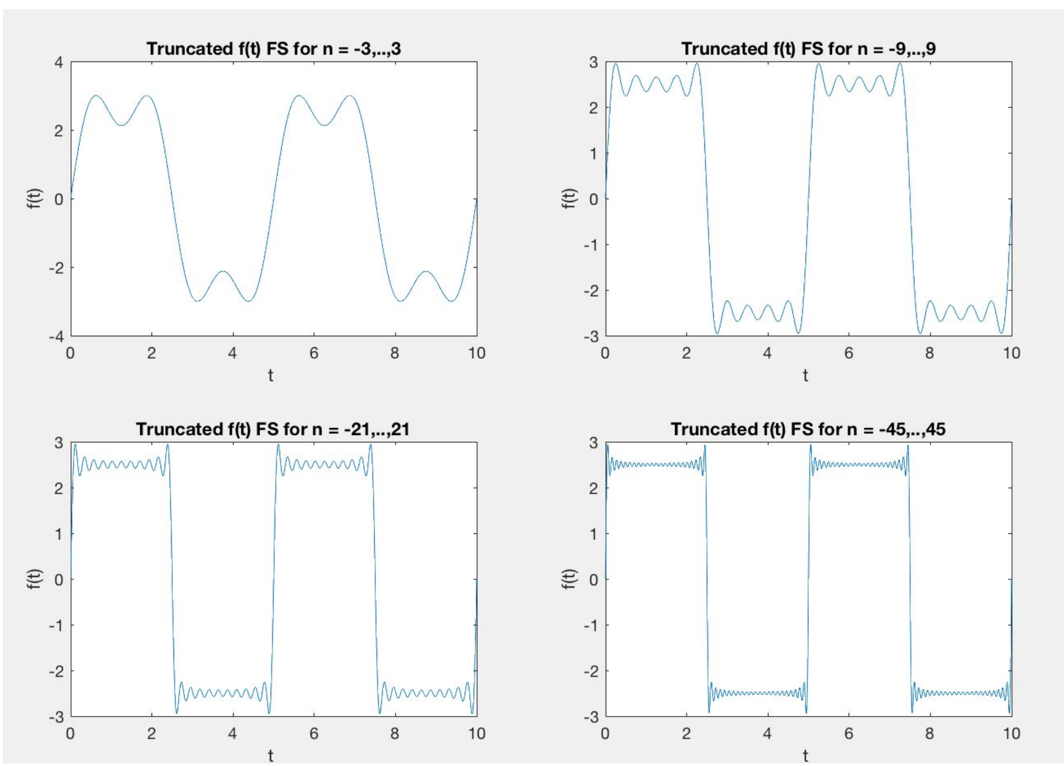
Problem 2.

Results are in audio format. See Appendix for code and Assignment1Problem2.m for audio.

Problem 3.



Problem 4.



Appendix

Problem 2 Code

```
load handel.mat

file = 'handel.wav';

[x,Fs] = audioread(file); % save original data to array with sampling rate

sound(x,Fs); % play original full file
pause(11);

tau = 0.5; % 500 ms delay
alpha = 0.7; % attenuation factor
D = tau*Fs; % total delay

y = zeros(size(x)); % set up echo matrix
y(1:D) = x(1:D);

for n=D+1:length(x)
    y(n) = alpha*y(n-D) + x(n); % using equation from (1)
end

sound(y,Fs); % play echoed file
```

Problem 3 Code

```
t = zeros(1,11);    % Define time range
y = zeros(1,11);    % Define y[n] range
x = y;              % Define x[n] range
x(t>=0) = 1;        % Define x[n] as step function

y1init = 0;         % Define inital conditions: y[-1]
y2init = 1;         % y[-2]

a = 1.7;            % Define gain values
b = -0.72;

for n=1:11          % Apply transfer function based on value of n
    if n==1
        y(n) = x(n) + a*y1init + b*y2init; % y[n] = x[n] + 1.7y[-1] - 0.72y[-2]
        t(n) = n-1;
    elseif n==2
        y(n) = x(n) + a*y(n-1) + b*y1init; % y[n] = x[n] + 1.7y[0] - 0.72y[-1]
        t(n) = n-1;
    else
```

```
        y(n) = x(n) + a*y(n-1) + b*y(n-2); % y[n] = x[n] + 1.7y[n-1] -
0.72y[n-2]
        t(n) = n-1;
    end
end

subplot(1,2,1);     % Plot x[n]
stem(t,x)
title('x[n]')
subplot(1,2,2);     % Plot y[n]
stem(t,y)
title('y[n]')
```

Problem 4 Code

```
figure(1); clf;         % Open and clear Figure 1

To = 5;                 % Define fundamental period, frequency and j
wo = 2*pi/To;
j = sqrt(-1);

t = 0:0.01:10;          % Define time range

N = [3 9 21 45];        % Define +/- harmonic values at which to truncate FS

for i = 1:4             % Compute truncated FS for harmonic values
    f = zeros(size(t)); % start out with DC bias term

    for k = -N(i):2:N(i)                % Loop over index k (odd)
        Ck = 2/(j*k*wo);                % FS coefficient
        f = f + real(Ck*exp(j*k*wo*t)); % FS computation
    end

    subplot(2,2,i); % Plot truncated FS representation of square wave
    plot(t,f);      % and actual signal
    hold on;

    xlabel('t ');
    ylabel('f(t)');
    titlevec = ['Truncated f(t) FS for n = '
num2str(-N(i)),',..,',num2str(N(i))];
    title(titlevec);
end
```