# ELEC 403 Lab 3 Script - EXPERIMENT 3

DESIGN OF A BAND-LIMITED DIFFRERENTIATOR FOR VELOCITY ESTIMATION BASED ON NOISY POSITION DATA % This script is used to fulfill the requirements in the lab manual. %

3.6.1 - following Sec. 3.3 to generate noise-free position signal s[n] and v[n] 1

3.6.2 - get a conventional differentiator of length 23 by following Sec. 3.3 3

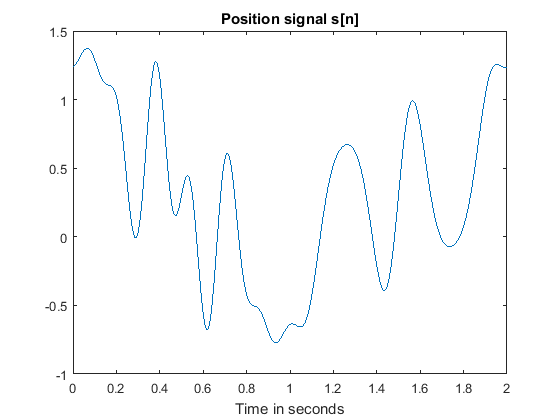
3.6.3 - generate noise w[n] and noise-corrupted position signal x[n] 5

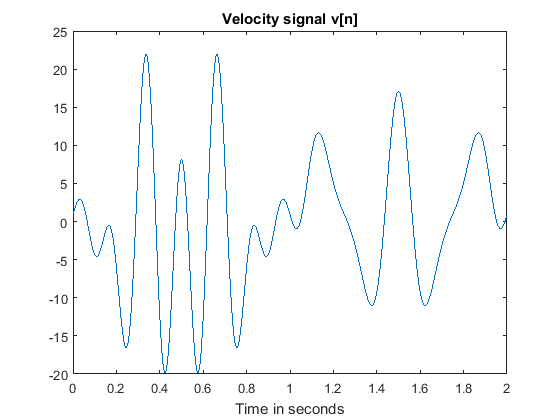
3.6.4 - design BLD with N =23, wp = 0.2pi, wa=0.5pi and Ts = 1/512 7

3.6.5 - apply BLD designed in 3.6.4 to x[n] save figures. 9

## 3.6.1 - following Sec. 3.3 to generate noise-free position signal s[n] and v[n]

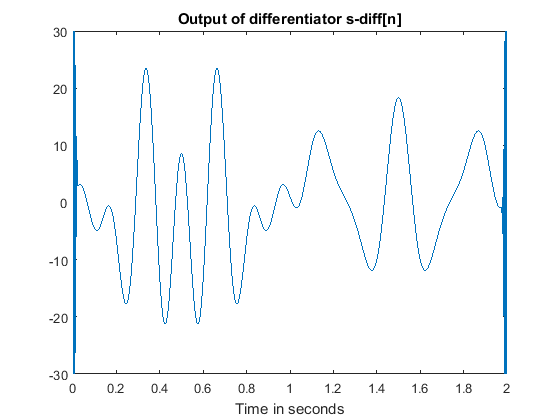
close all  
clear all  
  
t = 0:1/512:(2-1/512);  
t = t(:);  
randn('state', 7)  
a = 0.25\*randn(7,1);  
randn('state', 19)  
b = 0.25\*randn(7,1);  
s = 0.3\*ones(1024,1);  
v = zeros(1024,1);  
for i = 1:7,  
 s1 = a(i)\*sin(2\*pi\*i\*t);  
 s2 = b(i)\*cos(2\*pi\*(i-0.5)\*t);  
 s = s + s1 + s2;  
 v1 = i\*a(i)\*cos(2\*pi\*i\*t);  
 v2 = (i-0.5)\*b(i)\*sin(2\*pi\*(i-0.5)\*t);  
 v = v + 2\*pi\*(v1 - v2);  
end  
figure  
plot (t,s)  
xlabel('Time in seconds')  
title('Position signal s[n]')  
print('Reproduced3\_1a','-dpng','-r300')  
figure  
plot (t,v)  
xlabel('Time in seconds')  
title('Velocity signal v[n]')  
print('Reproduced3\_1b','-dpng','-r300')

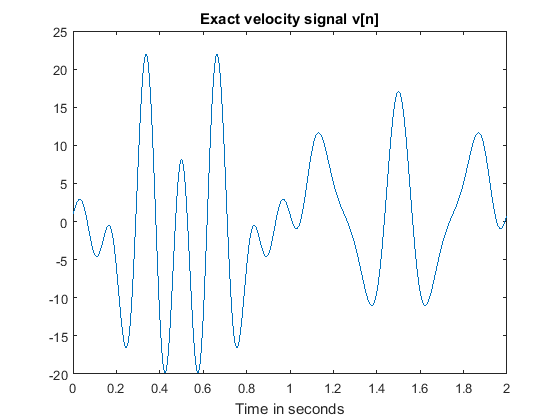




## 3.6.2 - get a conventional differentiator of length 23 by following Sec. 3.3

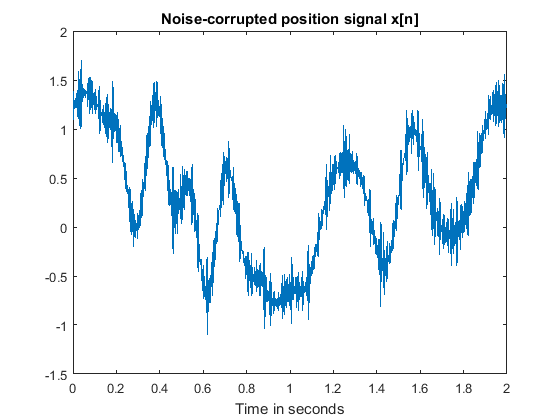
fs = 512;  
Ts = 1/fs;  
N = 23;  
M = (N-1)/2;  
n = 1:1:M;  
h = cos(n\*pi)./(Ts\*n); % compute desired impulse response using (3.5)  
h = [-fliplr(h) 0 h]; % compute desired impulse response using (3.4)  
win = hamming(N); % construct 41-point Hamming window  
h\_diff = win(:).\*h(:); % generate the impulse response of the differentiator  
  
% We now apply the differentiator to the position signal s[n] by discrete convolution:  
sw = conv(h\_diff, s); % perform digital differentiation by discrete convolution  
s\_diff = sw(12:1035); % reduce the output length to 1024  
  
figure  
plot (t, s\_diff)  
title('Output of differentiator s-diff[n]')  
xlabel('Time in seconds')  
axis([0 2 -30 30])  
print('Reproduced3\_2a','-dpng','-r300')  
  
figure  
plot(t,v)  
title('Exact velocity signal v[n]')  
xlabel('Time in seconds')  
print('Reproduced3\_2b','-dpng','-r300')

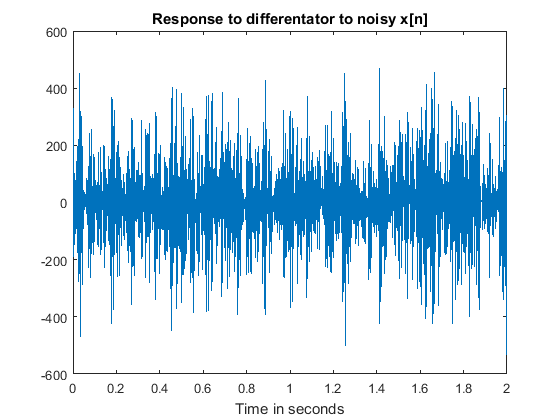




## 3.6.3 - generate noise w[n] and noise-corrupted position signal x[n]

randn('state',9); % sets a seed state for generating a random sequence  
w0 = randn(1024,1); % generate 1024 Gaussian white random samples  
mw = mean(w0); % evaluate its mean value  
w0 = w0 - mw; % modify w0 to have a zero-mean  
c = 0.3/sqrt((w0'\*w0)/1024);  
w0 = c\*w0; % modify w0 to have a standard deviation = 0.3  
h = fir1(250,0.7,'high'); % get a good highpass FIR filter with cutoff freq. = 0.7  
w1 = conv(h,w0); % apply highpass filtering to the white sequence  
w = w1(126:1149); % cut the filtered sequence to a right size  
w = w(:); % make sure w[n] is a column vector  
x = s + w; % Generate noisy position signal  
  
figure  
plot(t,x)  
title('Noise-corrupted position signal x[n]')  
xlabel('Time in seconds')  
print('Reproduced3\_3a','-dpng','-r300')  
  
figure  
sww = conv(h\_diff, x); % perform digital differentiation by discrete convolution  
w\_diff = sww(12:1035); % reduce the output length to 1024  
plot(t,w\_diff)  
title('Response to differentator to noisy x[n]')  
xlabel('Time in seconds')  
print('Reproduced3\_3b','-dpng','-r300')





## 3.6.4 - design BLD with N =23, wp = 0.2pi, wa=0.5pi and Ts = 1/512

format long  
h\_BLD = lab3\_363(23,1/512,0.2\*pi,0.5\*pi);  
h\_BLD

p =  
  
 Columns 1 through 3  
  
 40.686020377278155 72.029997840609937 87.241378322871554  
  
 Columns 4 through 6  
  
 83.874163751577640 64.339814440168666 35.017056807228130  
  
 Columns 7 through 9  
  
 4.263870745255236 -20.034761157085999 -32.633158103349196  
  
 Columns 10 through 11  
  
 -32.169957942439694 -21.172824565516400  
  
  
q =  
  
 Columns 1 through 3  
  
 0.861793301924978 -0.531283440388385 0.164290969791942  
 -0.531283440388385 1.026084269608764 -0.106107378607646  
 0.164290969791942 -0.106107378607646 1.148539524585688  
 0.425176062996481 0.286746224868288 -0.209603359847701  
 0.122455220840790 0.321680099301193 0.297205127125491  
 -0.103495957773908 0.132914183847191 0.413386424691162  
 0.010458873370765 -0.011789685918372 0.073473169617056  
 0.091706249068981 -0.048982091017164 -0.172172019577499  
 -0.059441055005861 -0.068676020848451 -0.088609561706681  
 -0.160382437396598 -0.099068396605395 0.005378476006910  
 -0.039627312563967 -0.086327839719034 -0.080431475586132  
  
 Columns 4 through 6  
  
 0.425176062996481 0.122455220840790 -0.103495957773908  
 0.286746224868288 0.321680099301193 0.132914183847191  
 -0.209603359847701 0.297205127125491 0.413386424691162  
 1.158998426828271 -0.117897034238076 0.237764142189861  
 -0.117897034238076 1.099557441874591 -0.278279393929659  
 0.237764142189861 -0.278279393929659 1.059930072001250  
 0.253004065922314 0.198136772316520 -0.204224976940981  
 0.033845799793747 0.327058482948997 0.216771753913668  
 -0.098117571193358 0.052480781412049 0.291842721359883  
 -0.069974479303084 -0.133333347142564 0.091841416993021  
 -0.029837425590578 -0.030613843501887 -0.042616003903041  
  
 Columns 7 through 9  
  
 0.010458873370765 0.091706249068981 -0.059441055005861  
 -0.011789685918372 -0.048982091017164 -0.068676020848451  
 0.073473169617056 -0.172172019577499 -0.088609561706681  
 0.253004065922314 0.033845799793747 -0.098117571193358  
 0.198136772316520 0.327058482948997 0.052480781412049  
 -0.204224976940981 0.216771753913668 0.291842721359883  
 1.078565053597575 -0.239440720914006 0.256132389528398  
 -0.239440720914006 1.117925689211494 -0.148723378027174  
 0.256132389528398 -0.148723378027174 1.125975706841395  
 0.382560081830462 0.264182407158481 -0.216955130677360  
 0.099891434609813 0.314328329220408 0.237764147283939  
  
 Columns 10 through 11  
  
 -0.160382437396598 -0.039627312563967  
 -0.099068396605395 -0.086327839719034  
 0.005378476006910 -0.080431475586132  
 -0.069974479303084 -0.029837425590578  
 -0.133333347142564 -0.030613843501887  
 0.091841416993021 -0.042616003903041  
 0.382560081830462 0.099891434609813  
 0.264182407158481 0.314328329220408  
 -0.216955130677360 0.237764147283939  
 1.099557446966611 -0.196292831636055  
 -0.196292831636055 1.077942522550796  
  
  
h\_BLD =  
  
 -0.797923862820184  
 -0.676208173791139  
 2.375172129586918  
 5.611656469894399  
 1.165735106609191  
 -12.499355678469954  
 -19.819051269390684  
 0.643297572310985  
 46.809278069707453  
 81.820406520359597  
 65.510591958352848  
 0  
 -65.510591958352848  
 -81.820406520359597  
 -46.809278069707453  
 -0.643297572310985  
 19.819051269390684  
 12.499355678469954  
 -1.165735106609191  
 -5.611656469894399  
 -2.375172129586918  
 0.676208173791139  
 0.797923862820184

## 3.6.5 - apply BLD designed in 3.6.4 to x[n] save figures.

xx = conv(h\_BLD, x); % perform digital differentiation by discrete convolution  
x\_diff = xx(12:1035); % reduce the output length to 1024  
figure  
plot(t,x\_diff);  
axis([0 2 -30 30])  
title('Applied BLD to noisy x[n]')  
xlabel('Time in seconds')  
print('Pic3\_5','-dpng','-r300')  
  
figure  
sssx = conv(h\_BLD, s);  
sss\_diff = sssx(12:1035);  
plot(t,sss\_diff);  
axis([0 2 -30 30])  
xlabel('Time in seconds')  
title('Applied BLD to s[n]')  
print('Pic3\_5b','-dpng','-r300');

[Published with MATLABｮ R2016a](http://www.mathworks.com/products/matlab)