EE2T11 - Telecommunications Practical

Midterm Report - Group A24

February 22, 2018

Nicolaas du Plessis 4203933

1: Convolution

Report 1

A step and impulse response was convoluted with a transfer function of a high pass and low pass filter, with coefficient "a" being 0.95 and -0.95 for each respective filter. The MATLAB script used can be found on page 8. The original input signals can be seen below in figure 1. N was chosen as 100 samples, as the increased resolution gives a better indication on the nature of the responses.

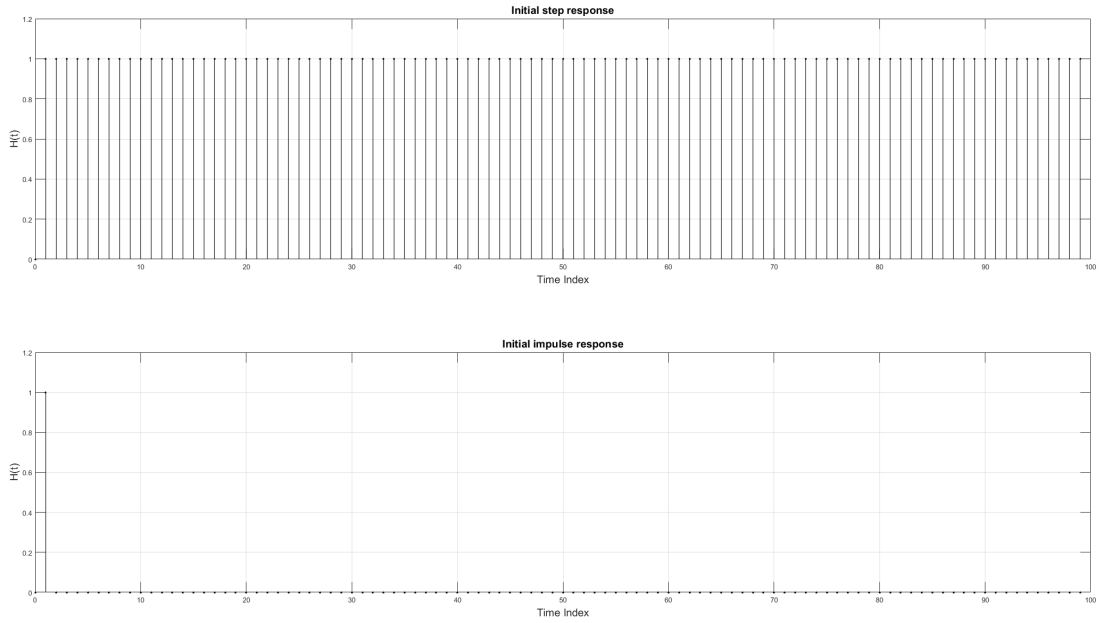

Figure 1: Input Impulse and Step Response

For the low pass signal response as seen in figure 2, the step response (which contains very high frequency components) in filtered, which only leaves the low frequecy components, meaning the filter takes a while to equalize (reach the input value), which is typical of a low pass filter. Similarly, the impulse response shows the effective settling time of the filter after an impulse, dependant on the filter coefficient.

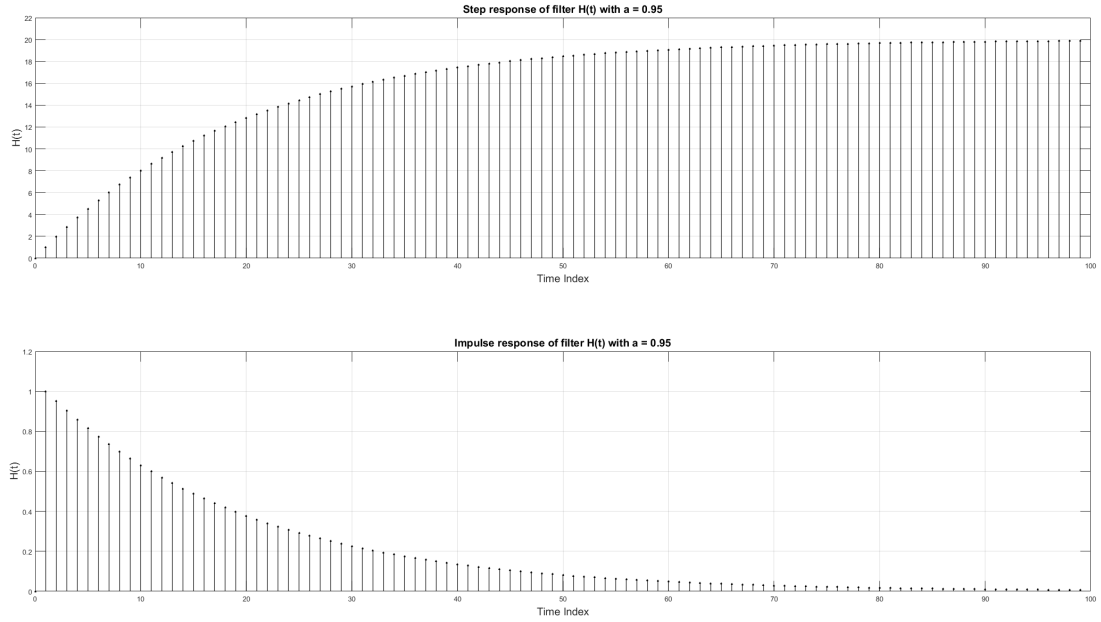

Figure 2: Low Pass Filter Response with a=0.95

For the high pass signal response as seen in figure 3, the difference in amplitude for the step response is magnified greatly, as the high frequency content of the filter allows for the quick rise time. However, as these frequencies are not dampened by the filter, they dent to oscillate, which can also be seen in the figure. The impulse signal results as shows an oscillatory behaviour, as the high frequency components of the impulse are greatly magnified, but not dampened.

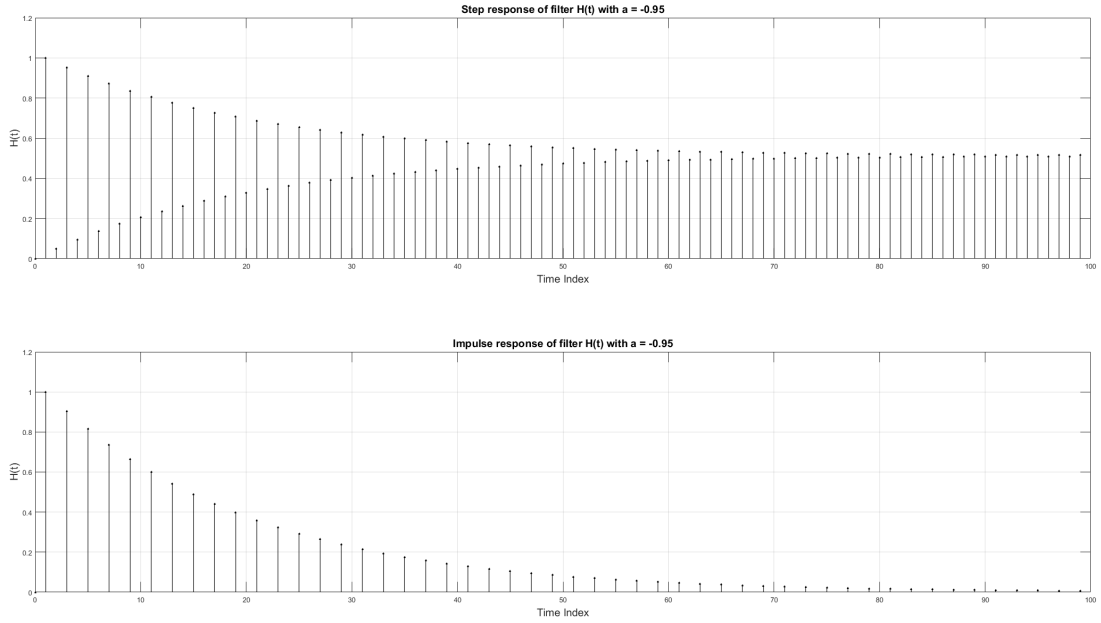

Figure 3: High Pass Filter Response with a=-0.95

Report 2

The second exercise simulated sound travelling from a source to a target, with first and second order reflections from the walls being included. The MATLAB script used can be found on page 8. A damping

factor of 0.5 was used to allow the signals to decrease over time, as it would in reality. An impulse response was sent as the source, and from the simulated received pulses the room channel impulse response was calculated through a convolution of the original impulse response and the simulated received response. Figure 4 shows that both the received signal and calculated response are the same, as expected.

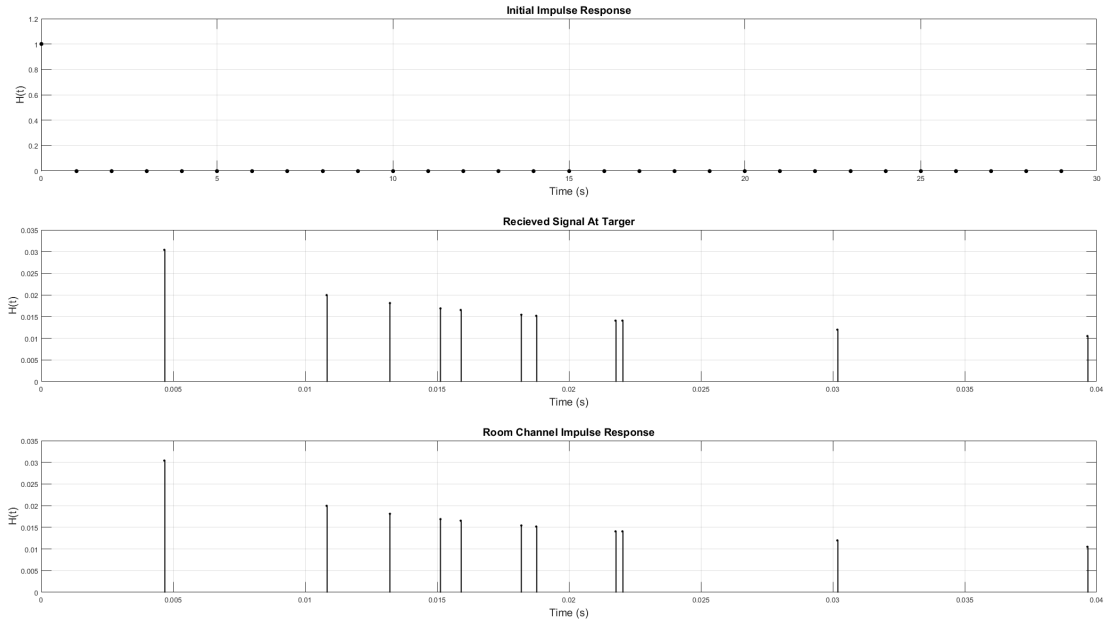

Figure 4: Room Channel Impulse Response & Reflected Signals

2: Fourier Transformation

Report 3

In figure 5, time and frequency domain plots of three audio signals are shown. The frequency domain plots where obtained using the MATLAB function `fft`. The frequency spectrum is symmetrical around the DC component $f = 0$Hz. The symmetry is a result of the discrete fourier transform, which is complex conjugate symmetric. Plotting only the positive frequencies would be a logical choice, since no information is lost, but was not done here in order to illustrate the mirrored spectrum. The MATLAB script can be found on page 10.

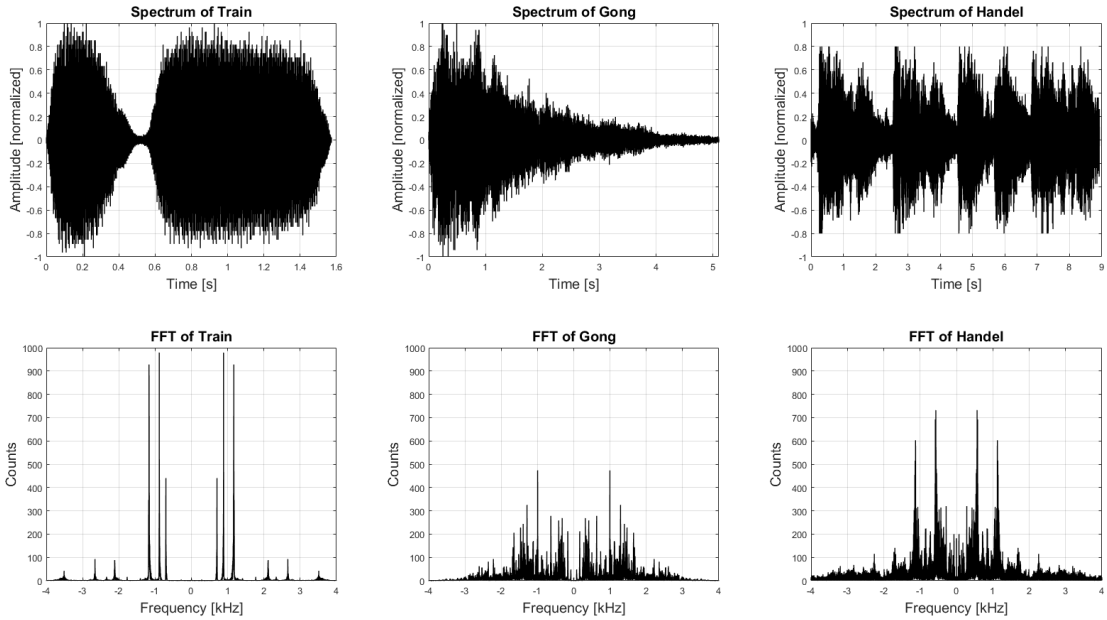

Figure 5: Time Domain & Frequency Domain plots of standard MATLAB sounds

Interesting to note here is that, particularly for the "Handel" file, that the limited sampling rate of the sounds means that no frequencies above 4kHz can be heard when played back. As the clarity in listening applications lies between 14kHz and 20kHz, and the perceived presence of audio around 8-12kHz, you can hear these lacking in the audio files.

Report 4

In figure 6, the previously used train signal is reshaped into segments of 20ms, on which the DFT is applied to each segment. Enlarging the segment size will give a better resolution on the frequency axis, but lower on the time axis. The DFT gives the frequency content of a the whole sample, but does not distinguishing when this frequency was present. By splinting up the sample, we can see the frequency content change in steps of 20ms through time. However, while smaller segment size gives better time resolution, it results in worse frequency resolution, with the opposite also being true. This is due to the number of samples in the window decreasing for smaller audio segments, meaning the resulting frequency domain graph also has less defined frequencies it displays.

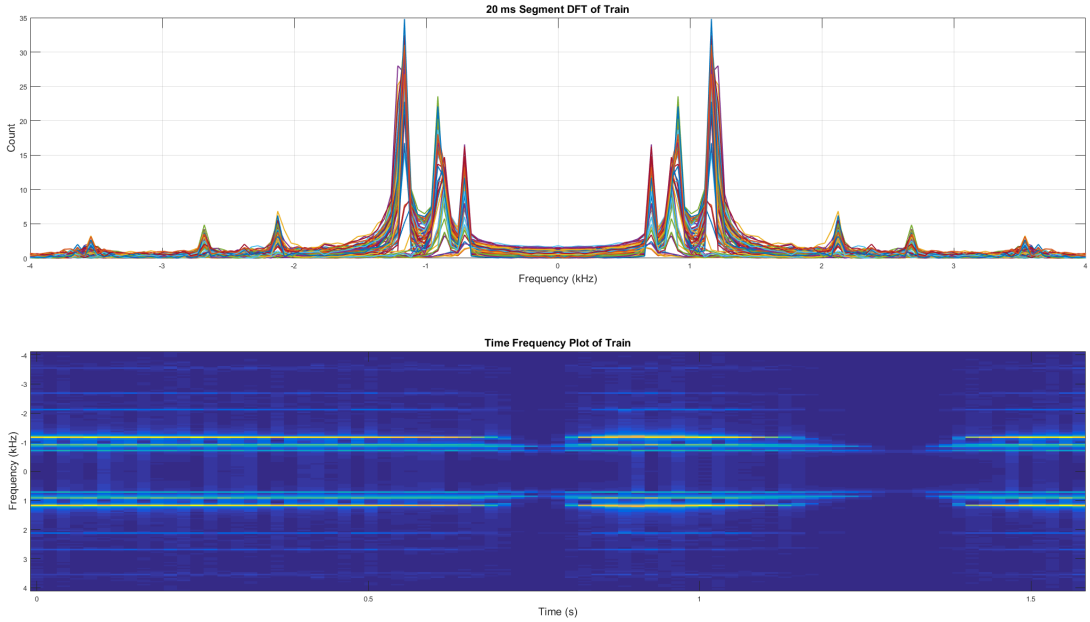

Figure 6: Train - Segmented FFT & Time Frequency Plot

Report 5 + 6

When the sample size is small, the resolution of corresponding FFT is limited as demonstrated in the previous report. If the time domain signal is zero-padded, the longer time signal will yield an interpolated signal in the frequency domain. This effect is shown below in figure 7, where the original signal and zero padded signal show the clear interpolation that occurs as a result of zero padding. The MATLAB script that was used can be found on page 11.

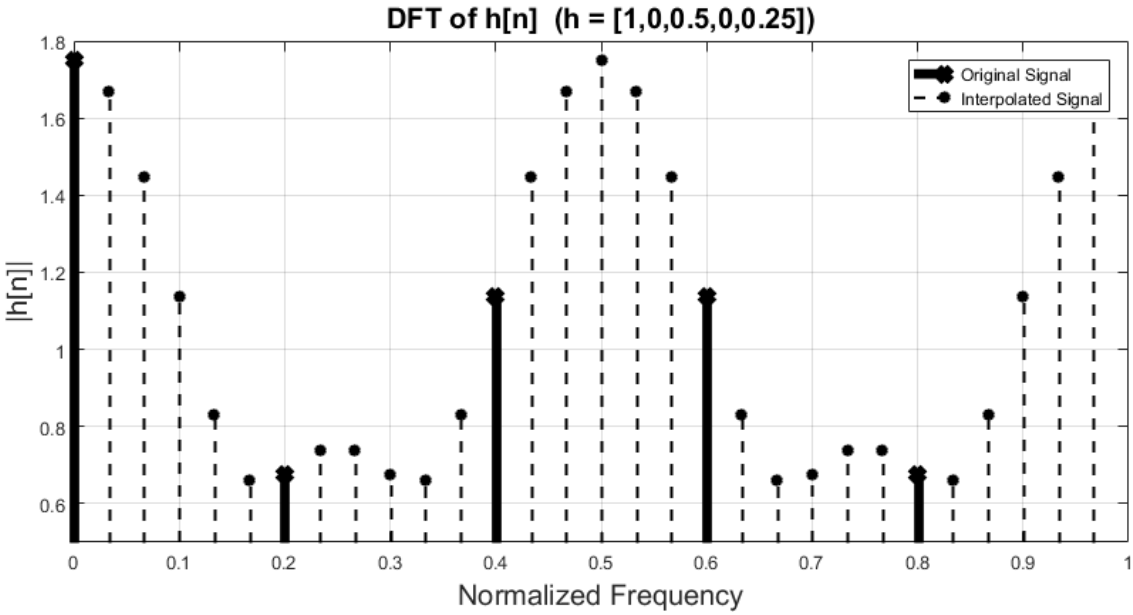

Figure 7: Interpolated sequence with DFT through zero padding

Report 7

In figure 8, the result of the convolution with the train audio signal x and the filter h is shown. Both plots are identical, indicating that the convolution $y[n] = x[n] * h[n]$ can be archived using the property

$$Y(\omega) = X(\omega)H(\omega).$$

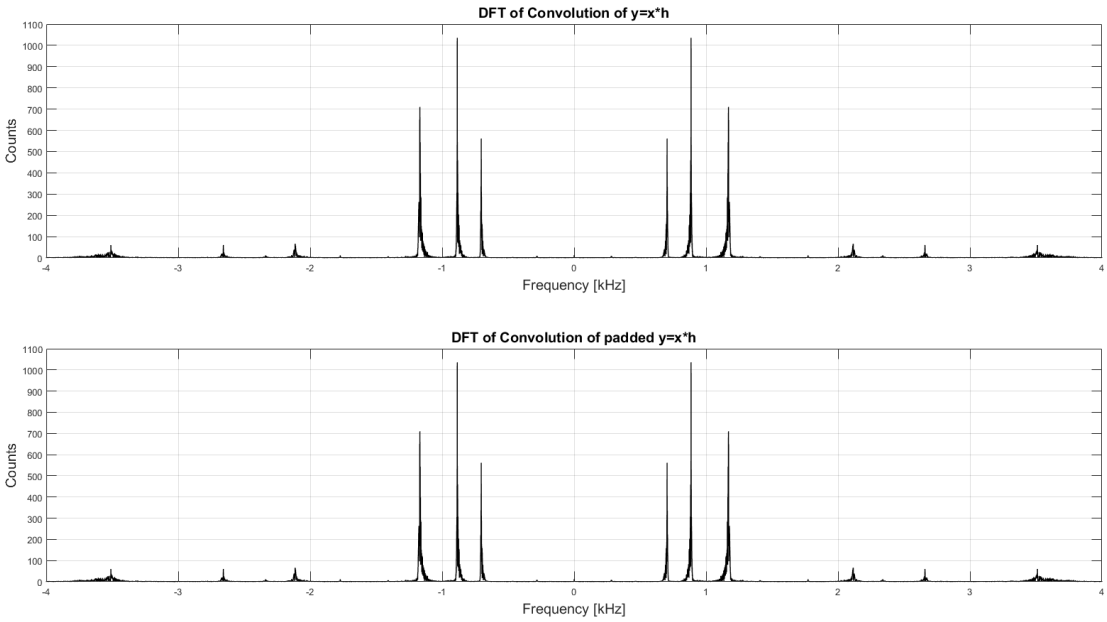

Figure 8: Comparison of DFT of convolution for original and padded signals

Appendix: Matlab Scripts

Script for Report 1

```
%Low pass and High pass filter on impulse and step response
%Author: Nicolaas du Plessis
close all; clear; clc;

N = 100; %number of samples
x_step = [0 ones(1,N-1)]; %step response
x_impulse = [0 1 zeros(1, N-2)]; %impulse response
n = 0:N-1; %time index

%Low Pass Filter
a = 0.95;
y_lp_step = filter(1, [1 -a], x_step);
y_lp_impulse = filter(1, [1 -a], x_impulse);
%High Pass Filter
a = -0.95;
y_hp_step = filter(1, [1 -a], x_step);
y_hp_impulse = filter(1, [1 -a], x_impulse);

%Plot Input Pulses
figure(1)
subplot(211); stem(n,x_step,'filled','LineStyle','-','LineWidth',0.8,'Color','black','Marker',
    '^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time␣Index','Fontsize',15); grid on;
title('Initial␣step␣response','Fontsize',15); axis([0 N 0 1.2])

subplot(212); stem(n,x_impulse,'filled','LineStyle','-','LineWidth',0.8,'Color','black','
    Marker','^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time␣Index','Fontsize',15); grid on;
title('Initial␣impulse␣response','Fontsize',15); axis([0 N 0 1.2])

%Low Pass Filter
figure(2)
subplot(211); stem(n,y_lp_step,'filled','LineStyle','-','LineWidth',0.8,'Color','black','
    Marker','^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time␣Index','Fontsize',15); grid on;
title('Step␣response␣of␣filter␣H(t)␣with␣a␣=␣0.95','Fontsize',15); axis([0 N 0 22])

subplot(212); stem(n,y_lp_impulse,'filled','LineStyle','-','LineWidth',0.8,'Color','black','
    Marker','^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time␣Index','Fontsize',15); grid on;
title('Impulse␣response␣of␣filter␣H(t)␣with␣a␣=␣0.95','Fontsize',15); axis([0 N 0 1.2])

%High Pass Filter
figure(3)
subplot(211); stem(n,y_hp_step,'filled','LineStyle','-','LineWidth',0.8,'Color','black','
    Marker','^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time␣Index','Fontsize',15); grid on;
title('Step␣response␣of␣filter␣H(t)␣with␣a␣=␣-0.95','Fontsize',15); axis([0 N 0 1.2])

subplot(212); stem(n,y_hp_impulse,'filled','LineStyle','-','LineWidth',0.8,'Color','black','
    Marker','^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time␣Index','Fontsize',15); grid on;
title('Impulse␣response␣of␣filter␣H(t)␣with␣a␣=␣-0.95','Fontsize',15); axis([0 N -1.2 1.2])
```

Script for Report 2

```
%Room Channel Impulse Response
%Author: Nicolaas du Plessis
close all; clear; clc;

%Due to the time limit, instead of making making a script that can
%calcualte nth reflection based on input coordinates, we instead hardcoded
%the reflection into the scripts. Each reflection is seen as a virtual
```

```
%source mirrored in the appropriate plane.

sound_speed = 340; %speed of sound in m/s
beta_damp = 0.5; %dampning factor of the air
num_virt_source = 17; %number of virtual sources

%Impulse Signal
N = 30; %number of samples
x = [1 zeros(1, N-1)]; %impulse response
n = 0:N-1; %time index

%Source Coordiantes
Tx_x = 1.2; %meters
Tx_y = 0.3; %meters
%Target Coordinates
Rx_x = 3.1; %meters
Rx_y = 3.3; %meters
%Reflection Boundary Coordinates
Room_x = 4; %meters
Room_y = 4; %meters

%Virtual Source Vector
sources = zeros(num_virt_source,2); %sources(:,1) is x coordiantes, sources(:,2) is y
    coordiantes
sources(1,1) = Tx_x; sources(1,2) = Tx_y; %Original Source

%Primary Reflections
sources(2,1) = -Tx_x; sources(2,2) = Tx_y; %X=0
sources(3,1) = (2*Room_x)-Tx_x; sources(3,2) = Tx_y; %X=4
sources(4,1) = Tx_x; sources(4,2) = -Tx_y; %Y=0
sources(5,1) = Tx_x; sources(5,2) = (2*Room_y)-Tx_y; %Y=4

%Secondary Reflections
sources(6,1) = sources(2,1); sources(6,2) = (2*Room_y)-sources(2,2); %X=0 -> Y=4
sources(7,1) = sources(2,1); sources(7,2) = -sources(2,2); %X=0 -> Y=0
sources(8,1) = (2*Room_x)-sources(2,1); sources(8,2) = sources(2,2); %X=0 -> X=4

sources(9,1) = sources(3,1); sources(9,2) = (2*Room_y)-sources(3,2); %X=4 -> Y=4
sources(10,1) = sources(3,1); sources(10,2) = -sources(3,2); %X=4 -> Y=0
sources(11,1) = -sources(3,1); sources(11,2) = sources(3,2); %X=4 -> X=0

sources(12,1) = sources(4,1); sources(12,2) = -sources(4,2); %Y=4 -> Y=0
sources(13,1) = -sources(4,1); sources(13,2) = sources(4,2); %Y=4 -> X=0
sources(14,1) = (2*Room_x)-sources(4,1); sources(14,2) = sources(4,2); %Y=4 -> X=4

sources(15,1) = sources(5,1); sources(15,2) = (2*Room_y)-sources(5,2); %Y=4 -> Y=0
sources(16,1) = -sources(5,1); sources(16,2) = sources(5,2); %Y=4 -> X=0
sources(17,1) = (2*Room_x)-sources(5,1); sources(17,2) = sources(5,2); %Y=4 -> X=4

%Calculations
distances = sqrt(((sources(:,1)-Rx_x).^2) + ((sources(:,2)-Rx_y).^2)); %Distance Vector
attenuations = (distances./sound_speed); %Attenutions
times = beta_damp./(distances.^2); %Travel Times

atten_sorted = sort(attenuations,'descend'); %Higher attenuation means longer journey
times_sorted = sort(times); %Sort times to match above attenuation vector

%Convolution of input signal and resulting recording
response = conv(x,atten_sorted);
response = response(1:num_virt_source); %truncate padded zeros

%Plots
subplot(311)
stem(n,x,'filled','LineStyle','-','LineWidth',0.8,'Color','black','MarkerSize',5);
ylabel('H(t)','Fontsize',15); xlabel('Time␣(s)','Fontsize',15); grid on;
title('Initial␣Impulse␣Response','Fontsize',15); axis([0 N 0 1.2]);

subplot(312)
stem(times_sorted,atten_sorted,'filled','LineStyle','-','LineWidth',1.5,'Color','black','
```

```
    Marker','^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time (s)','Fontsize',15); grid on;
title('Recieved Signal At Targer','Fontsize',15);

subplot(313)
stem(times_sorted,response,'filled','LineStyle','-','LineWidth',1.5,'Color','black','Marker','
    ^','MarkerSize',2);
ylabel('H(t)','Fontsize',15); xlabel('Time (s)','Fontsize',15); grid on;
title('Room Channel Impulse Response','Fontsize',15);
```

Script for Report 3

```
%Time Domain and Frequency Domain
%Author: Nicolaas du Plessis
close all; clear; clc;

load train
%Time axis
time_length = (length(y)/Fs);
t = linspace(0,time_length,length(y));
freq=linspace(-Fs/2000,Fs/2000,length(y));
Y = fftshift(abs(fft(y)));

subplot(231)
plot(t,y,'LineWidth',1,'Color','black');
axis([0 1.6 -1 1]); grid on;
xlabel('Time [s]','Fontsize',15); ylabel('Amplitude [normalized]','Fontsize',15)
title('Spectrum of Train','Fontsize',15)

subplot(234)
plot(freq,Y,'LineWidth',1,'Color','black');
axis([-4 4 0 1000]); grid on;
xlabel('Frequency [kHz]','Fontsize',15); ylabel('Counts','Fontsize',15)
title('FFT of Train','Fontsize',15)

load gong
%Time axis
time_length = length(y)/Fs;
t = linspace(0,time_length,length(y));
freq=linspace(-Fs/2000,Fs/2000,length(y));
Y = fftshift(abs(fft(y)));

subplot(232)
plot(t,y,'LineWidth',1,'Color','black');
axis([0 5.1 -1 1]); grid on;
xlabel('Time [s]','Fontsize',15); ylabel('Amplitude [normalized]','Fontsize',15)
title('Spectrum of Gong','Fontsize',15)

subplot(235)
plot(freq,Y,'LineWidth',1,'Color','black');
axis([-4 4 0 1000]); grid on;
xlabel('Frequency [kHz]','Fontsize',15); ylabel('Counts','Fontsize',15)
title('FFT of Gong','Fontsize',15)

load handel
%Time axis
time_length = length(y)/Fs;
t = linspace(0,time_length,length(y));
freq=linspace(-Fs/2000,Fs/2000,length(y));
Y = fftshift(abs(fft(y)));

subplot(233)
plot(t,y,'LineWidth',1,'Color','black');
axis([0 9 -1 1]); grid on;
xlabel('Time [s]','Fontsize',15); ylabel('Amplitude [normalized]','Fontsize',15)
title('Spectrum of Handel','Fontsize',15)

subplot(236)
```

```
plot(freq,Y,'LineWidth',1,'Color','black');
axis([-4 4 0 1000]); grid on;
xlabel('Frequency [kHz]','Fontsize',15); ylabel('Counts','Fontsize',15)
title('FFT of Handel','Fontsize',15)
```

Script for Report 4

```
%Time Frequency Plot
%Author: Nicolaas du Plessis
close all; clear; clc;

load train

%Time axis
time_length = length(y)/Fs;
Sample_lengths = 0.02;

coloms = fix(Fs.*Sample_lengths); %Amount of Samples
rows = fix(length(y)/coloms); %Amount of Blocks
trunc_l = coloms * rows;
y_trunc = y(1:trunc_l);

X = reshape(y_trunc, coloms, rows);
Y = fftshift(abs(fft(X)));

t = linspace(0,time_length,length(y));
f = linspace(-Fs/2000,Fs/2000,coloms);

subplot(211)
plot(f,Y,'LineWidth',1.5);
axis([-4 4 0 35]); grid on
xlabel('Frequency (kHz)','Fontsize',15); ylabel('Count','Fontsize',15)
title('20 ms Segment DFT of Train','Fontsize',15)

subplot(212)
imagesc(t,f,Y)
xlabel('Time (s)','Fontsize',15); ylabel('Frequency (kHz)','Fontsize',15)
title('Time Frequency Plot of Train','Fontsize',15)
```

Script for Report 5+6

```
%Zero Padding
%Author: Nicolaas du Plessis
close all; clear; clc;

h = [1 0 1/2 0 1/4];
H = fftshift(abs(fft(h)));
N = 5;
f = 0: 1/N : (N-1)/N;

stem(f,H,'LineStyle','-','LineWidth',5,'Color','black','Marker','x','MarkerSize',8);
axis([0 1 0.5 1.8]); grid on
xlabel('Normalized Frequency','Fontsize',15); ylabel('|h[n]|','Fontsize',15)
title('DFT of h[n]  (h = [1,0,0.5,0,0.25])','Fontsize',15)

hold on

N = 30;
f = 0: 1/N : (N-1)/N;
h = [1 0 1/2 0 1/4 zeros(1,25)];
H = fftshift(abs(fft(h)));

stem(f,H,'filled','LineStyle','--','LineWidth',1.5,'Color','black','Marker','o','MarkerSize'
    ,5);

legend('Original Signal','Interpolated Signal')
```

Script for Report 7

```
%Convolution Property
%Author: Nicolaas du Plessis
close all; clear; clc;

load train
freq=linspace(-Fs/2000,Fs/2000,length(y)); %Factor 1000 here to plot in kHz

%Original Signals
h_short = [1 0 1/2 0 1/4]; %Impulse
Y_short = conv(y,h_short,'same');
Y_short = Y_short(1:length(y)); %truncate padded zeros

%Plot Original
subplot(211)
plot(freq,fftshift(abs(fft(Y_short))),'LineStyle','-','LineWidth',1,'Color','black');
axis([-4 4 0 1100]); grid on
xlabel('Frequency [kHz]','Fontsize',15); ylabel('Counts','Fontsize',15)
title('DFT of Convolution of y=x*h','Fontsize',15)

%Padded Signals
h_padded = [1 0 1/2 0 1/4, zeros(1,12884-5)]; %Filter impulse
Y_padded = conv(y,h_padded);
Y_padded = Y_padded(1:length(y)); %truncate padded zeros

%Plot Padded
subplot(212)
plot(freq, fftshift(abs(fft(Y_padded))),'LineStyle','-','LineWidth',1,'Color','black');
axis([-4 4 0 1100]); grid on
xlabel('Frequency [kHz]','Fontsize',15); ylabel('Counts','Fontsize',15)
title('DFT of Convolution of padded y=x*h','Fontsize',15)
```