Signals and Systems
Matlab Homework #2

Table of Contents

Introduction

- Author: Will Burgess
- Class: ESE 351
- Date: Created 1/25/2024, Last Edited 2/06/2024
- With contributions from: Mack Larosa, Tasha Igic, Mischa Tranor

Variable Initialization

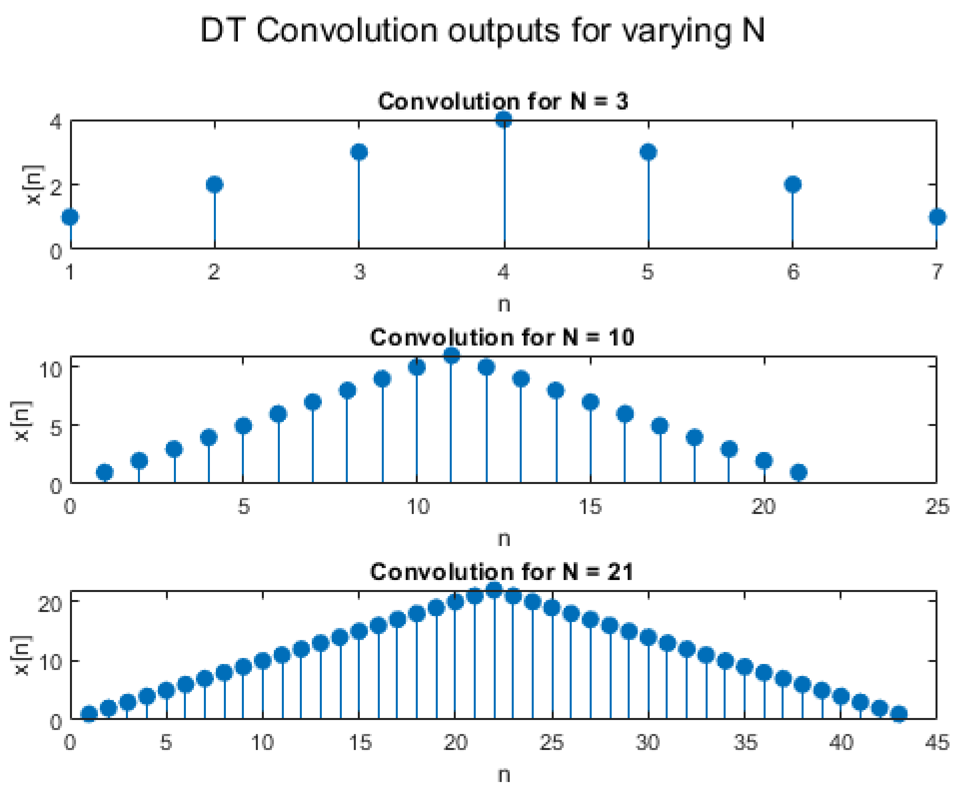
```
oldparam = sympref('HeavisideAtOrigin',1);
sympref('HeavisideAtOrigin',oldparam);
close all
R = 1e3 ; %Resistance in Ohms
C = 5e-6 ; %Capacitence in Farads
tau = R*C;
sampleFreq = 44.1e3; %Sampling freq in Hz
samplePeriod = 1/sampleFreq;
```

```
timeRange = 0:samplePeriod:15*tau;
```

Part 1: Compute discrete-time Convolution for a finite-length Function

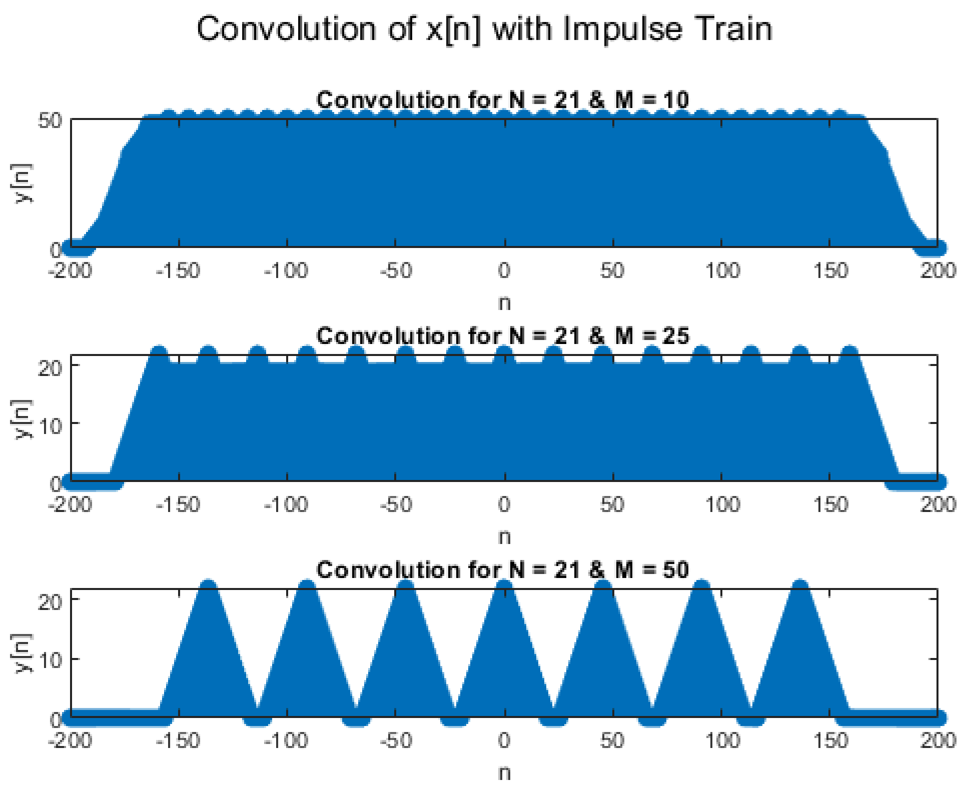
Part 1 A

```
figure();
NRange = [3,10,21];
for i = 1:3 % range of 3 due to 3 N values
N = NRange(i);

r = ones(1,N+1);
x = conv(r,r);
val = (-1:1:length(x)-1);

% Subplot results
subplot(3,1,i)
stem(x,'filled') %Use Stem plot for DT
title(['Convolution for N = ', num2str(N)]);
xlabel('n');
ylabel('x[n]');
end
sgtitle('DT Convolution outputs for varying N')
```



Part 1 B

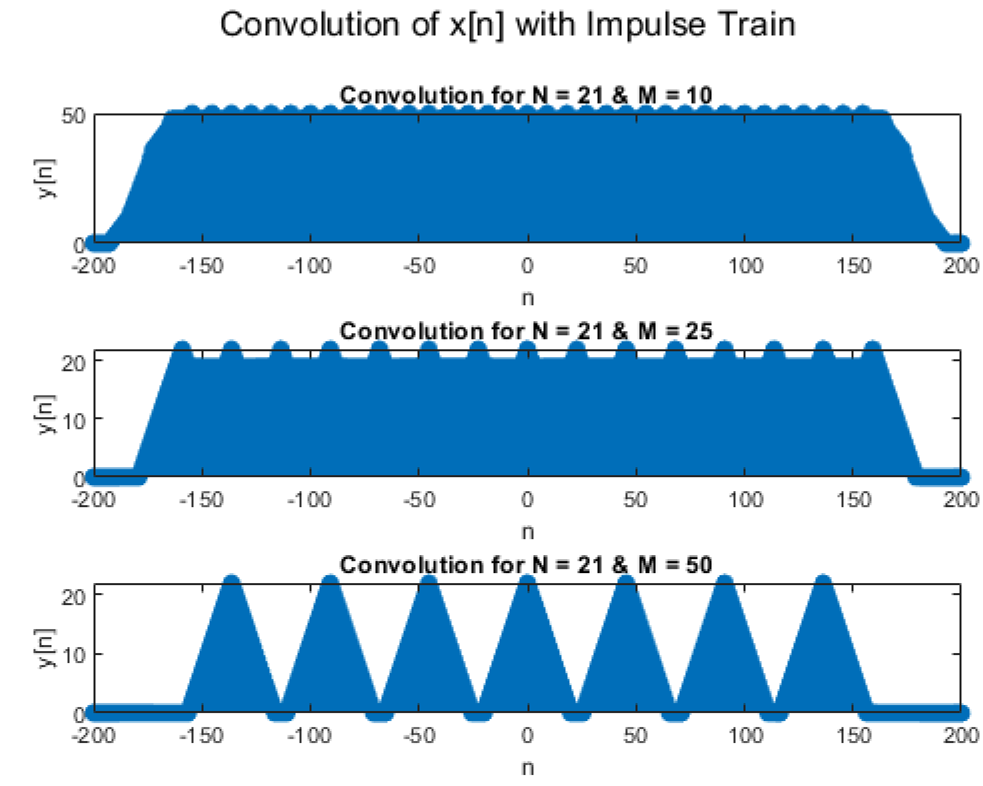
```
%Initialize
N = 21;
n = -200:1:200;
r = ones(1,N+1);
x = conv(r,r);

MRange = [10,25,50];

%Incrementaly perform series with varying M values
figure()
for i = 1:3 %range of 3 due to 3 M values
M = MRange(i);
sum = zeros(1,range(n)-1);
    for k = 1:(range(n)-1)
        if mod(k,M) == 0
            sum(k)=1;
        end
    end
y = conv(x,sum);

subplot(3,1,i)
stem(linspace(-200,200, length(y)),y,'filled')
title(['Convolution for N = 21 & M = ',num2str(M)]);
xlabel('n');
ylabel('y[n]');
sgtitle('Convolution of x[n] with Impulse Train');
end
```

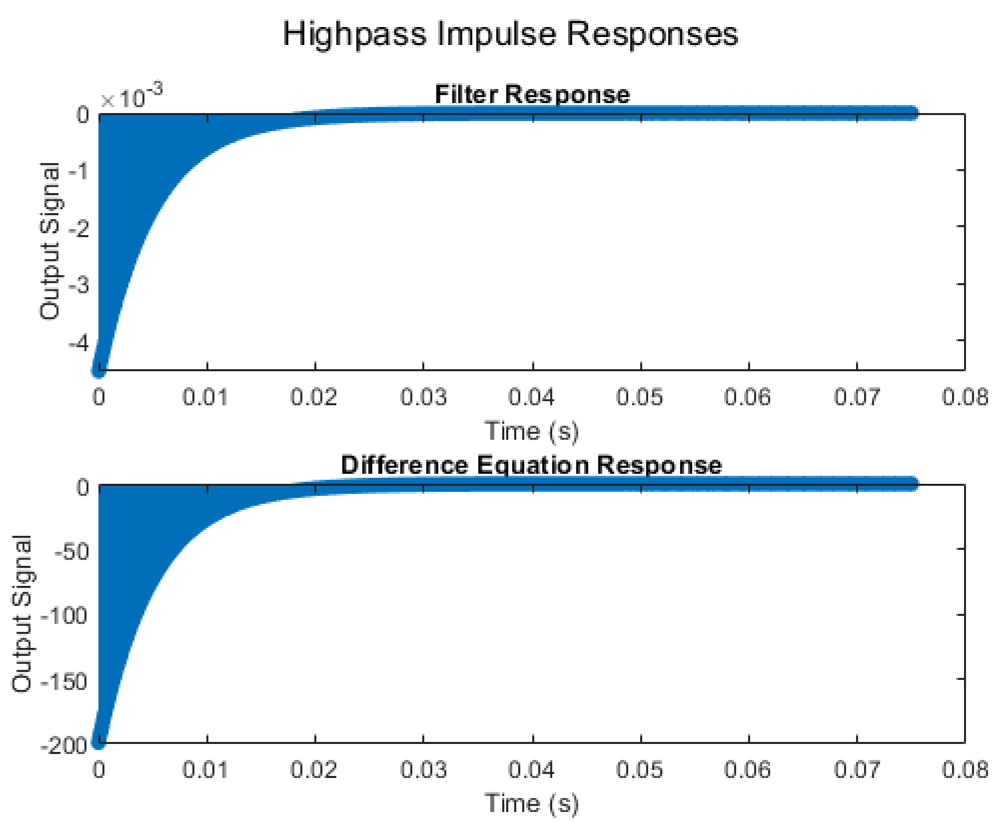


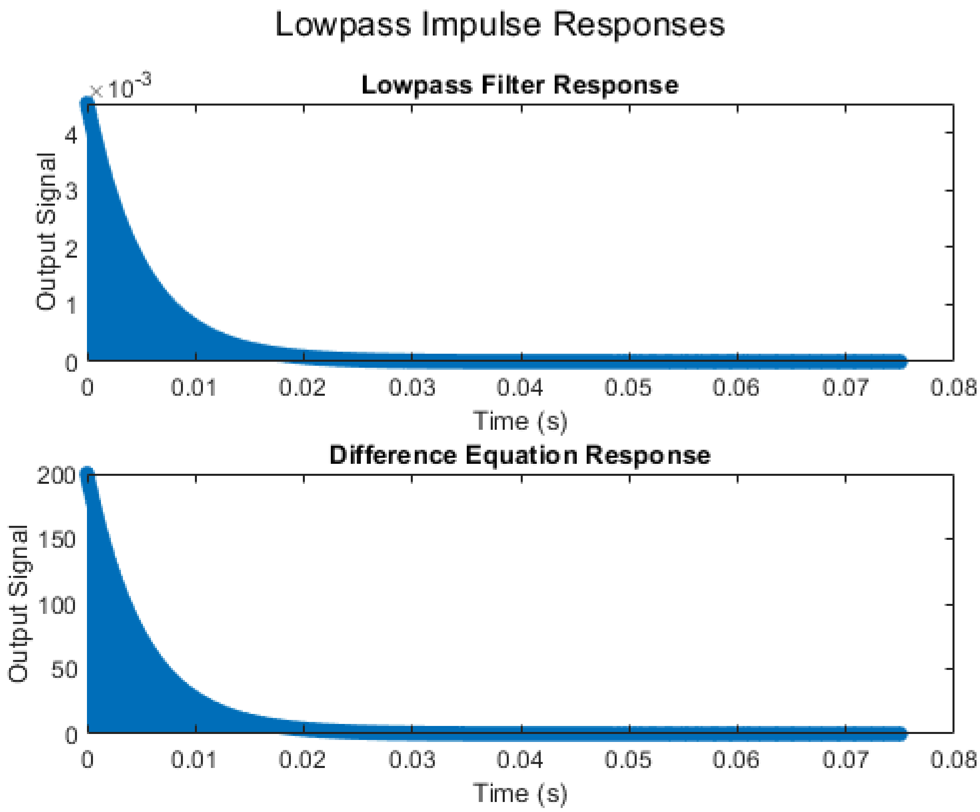
Part 2 A

```
%Initialize variables
R = 1e3; % Resistance in Ohms
C = 5e-6; % Capacitence in Farads
tau = R*C;
sampleFreq = 44.1e3; % Sampling frequency in Hz
samplePeriod = 1/sampleFreq;
t = 0:samplePeriod:15*tau;

%Define inputFunction as a impulse at t=0
inputFunction = zeros(1,length(t));
inputFunction(1) = 1;

%Calculate responses via difference equaton
%Lowpass
h_low = @(t) 1/tau * exp(-t/tau);
h_low = h_low(t);

%Highpass
h_hi = @(t) heaviside(t) - 1/tau *exp(-t/tau);
h_hi = h_hi(t);

%Calculate reponses via filter
%Lowpass
```

```
a = [1, samplePeriod/tau-1];
b = samplePeriod/tau;
lowResponse_impulse = filter(b,a,inputFunction);

%Highpass
a = [1,(samplePeriod/tau) - 1];
b = [1,-1];
hiResponse_impulse = filter(b,a,inputFunction);

%Plot highpasses
figure()
hold on
subplot(2,1,1)
stem(t(2:end),hiResponse_impulse(2:end),'o','MarkerSize',5);
title('Filter Response');
xlabel('Time (s)')
ylabel('Output Signal')

subplot(2,1,2)
stem(t,h_hi,'o','MarkerSize',5);
title('Difference Equation Response');
xlabel('Time (s)')
ylabel('Output Signal')
sgtitle('Highpass Impulse Responses')
hold off

%Plot lowpasses
figure()
hold on
subplot(2,1,1)
stem(t(2:end),lowResponse_impulse(2:end),'o','MarkerSize',5);
title('Lowpass Filter Response');
xlabel('Time (s)')
ylabel('Output Signal')

subplot(2,1,2)
stem(t,h_low,'o','MarkerSize',5);
title('Difference Equation Response');
xlabel('Time (s)')
ylabel('Output Signal')
sgtitle('Lowpass Impulse Responses')
hold off
```


Convolution of x[n] with Impulse Train
Convolution for N = 21 & M = 10
y[n]
n
Convolution for N = 21 & M = 25
y[n]
n
Convolution for N = 21 & M = 50
y[n]
n

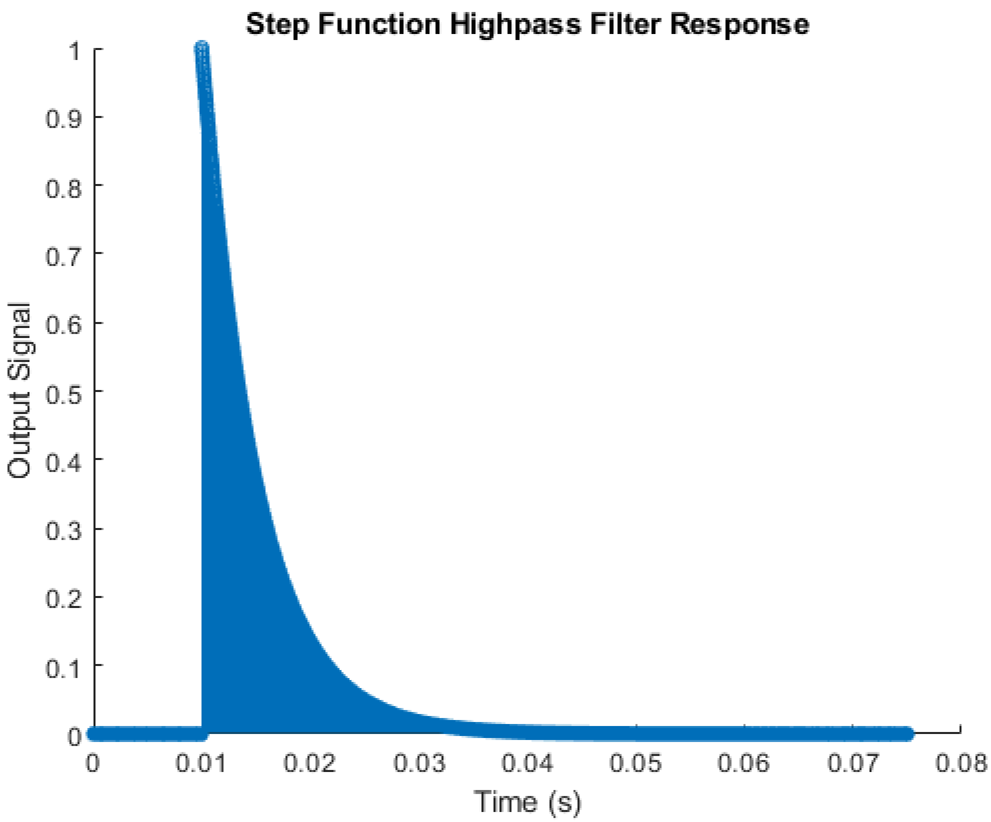

Highpass Impulse Responses
Filter Response
Output Signal
Time (s)
Difference Equation Response
Output Signal
Time (s)

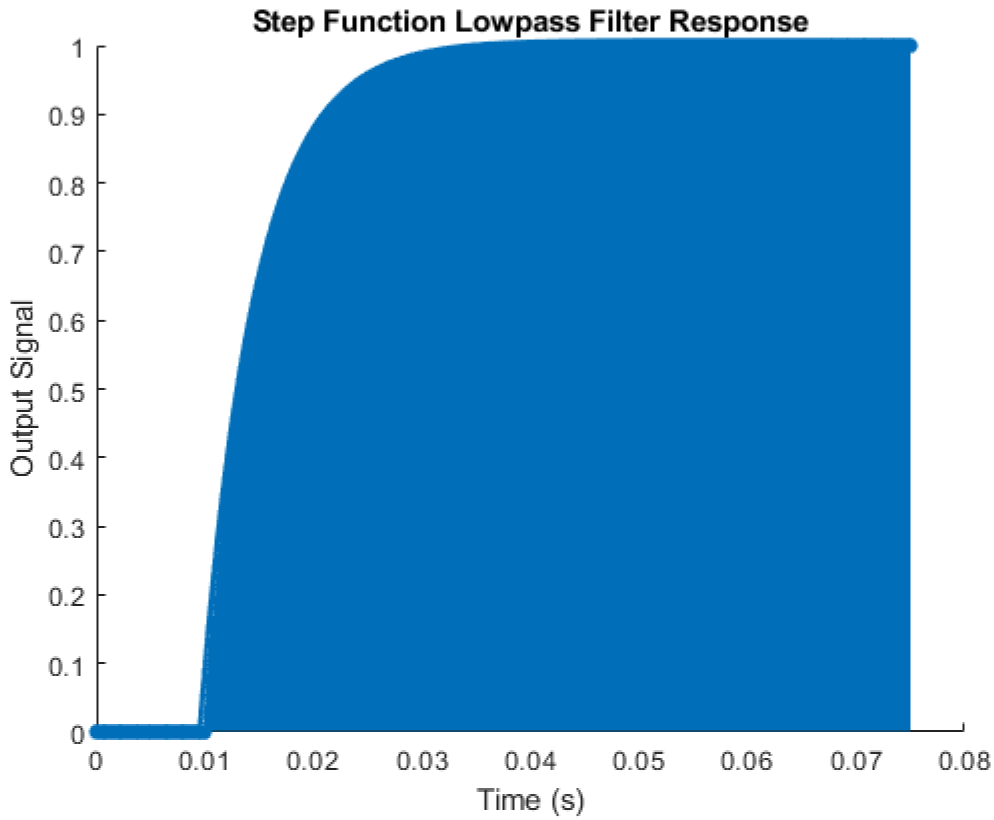


Part 2 B i : Compututation using difference filter

```
%Define inputFunction as unit step function
stepFunction = zeros(1,length(t));
stepFunction(2*tau*sampleFreq:end) = 1;

%Calculate reponses via filter
%Highpass
a = [1, samplePeriod/tau-1];
b = samplePeriod/tau;
lowResponse_step = filter(b,a,stepFunction);

%Highpass
a = [1,(samplePeriod/tau) - 1];
b = [1,-1];
hiResponse_step = filter(b,a,stepFunction);

%Plot highpasses
figure()
hold on
stem(t(2:end),hiResponse_step(2:end),'o','MarkerSize',5);
title('Step Function Highpass Filter Response');
xlabel('Time (s)')
```

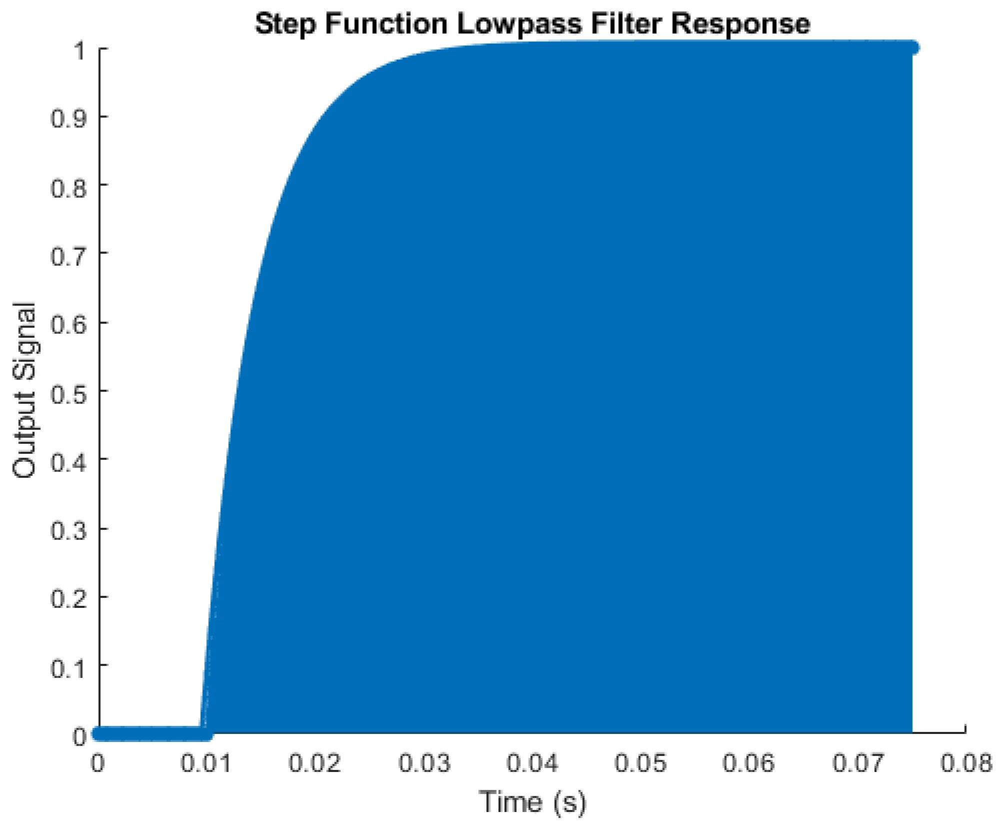
```
ylabel('Output Signal')
hold off

%Plot lowpasses
figure()
hold on
stem(t(2:end),lowResponse_step(2:end),'o','MarkerSize',5);
title('Step Function Lowpass Filter Response');
xlabel('Time (s)')
ylabel('Output Signal')
hold off
```

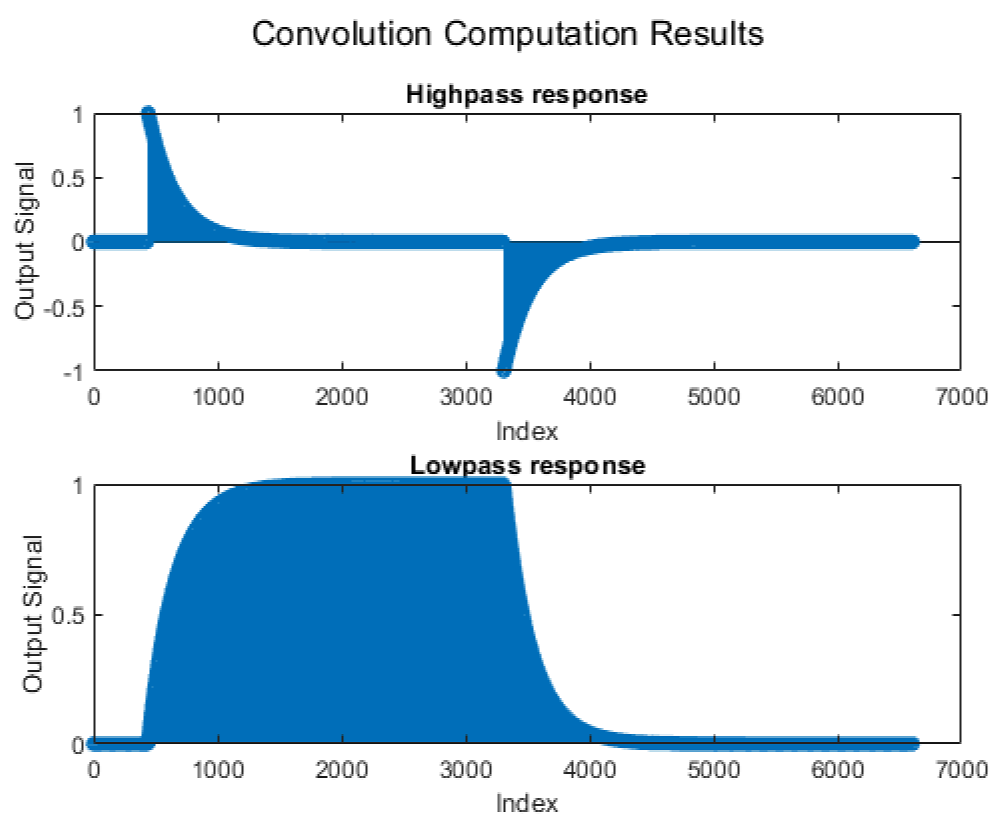




Part 2 B ii : Computation via convolution

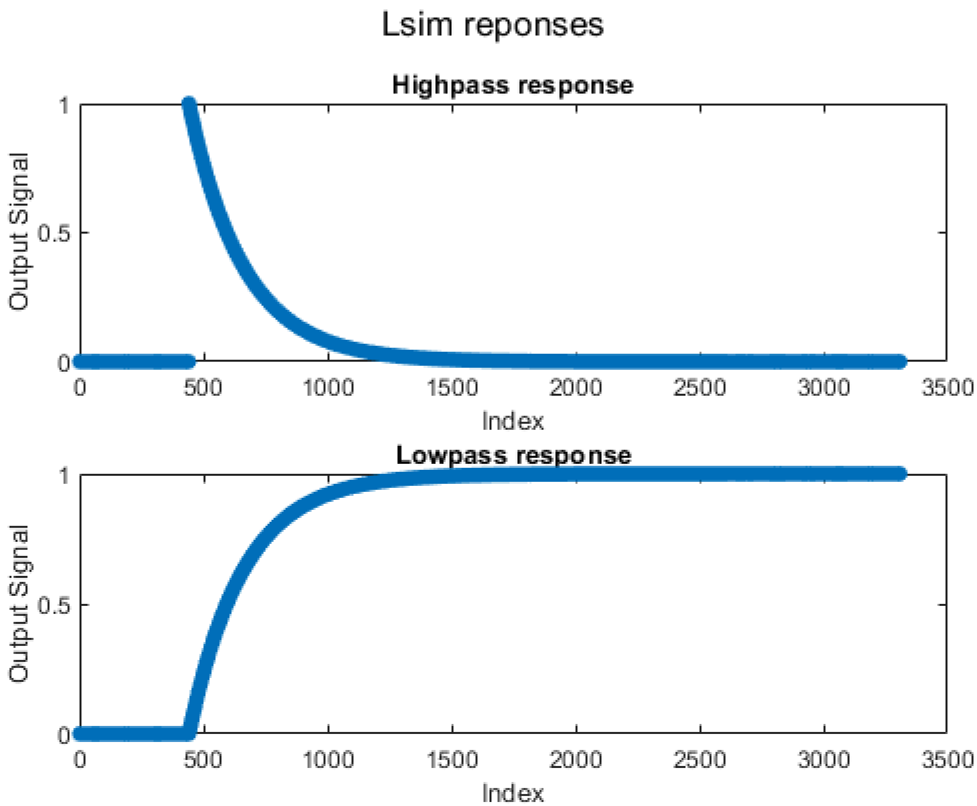
```
%Highpass
convImpulse_hi = conv(stepFunction,hiResponse_impulse);
%Lowpass
convImpulse_lo = conv(stepFunction,lowResponse_impulse);

figure()
hold on
subplot(2,1,1)
stem(convImpulse_hi,'o','MarkerSize',5);
title('Highpass response');
xlabel('Index')
ylabel('Output Signal')

subplot(2,1,2)
stem(convImpulse_lo,'o','MarkerSize',5);
title('Lowpass response');
xlabel('Index')
ylabel('Output Signal')

sgtitle('Convolution Computation Results')
hold off
```


Step Function Lowpass Filter Response
Output Signal
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)


Convolution Computation Results
Highpass response
Output Signal
1
0.5
0
-0.5
-1
0
1000
2000
3000
4000
5000
6000
7000
Index
Lowpass response
Output Signal
1
0.5
0
0
1000
2000
3000
4000
5000
6000
7000
Index

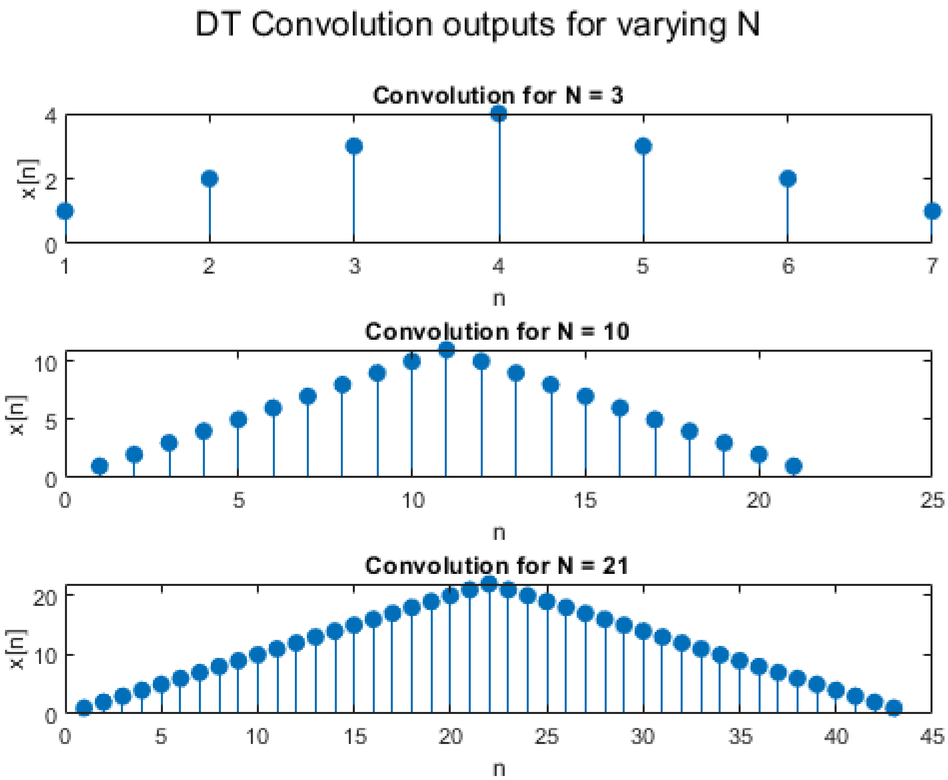
Part 2 B iii : Computation via lsim

```
%Highpass
a = [1, 1/tau] ;
b = [1,0];
hi_sim = lsim(b,a,stepFunction,t);

%Lowpass
a = [1,1/tau];
b = 1/tau;
lo_sim = lsim(b,a,stepFunction,t);

figure()
hold on
subplot(2,1,1)
plot(hi_sim,'o','MarkerSize',5);
title('Highpass response');
xlabel('Index')
ylabel('Output Signal')

subplot(2,1,2)
plot(lo_sim,'o','MarkerSize',5);
title('Lowpass response');
xlabel('Index')
ylabel('Output Signal')

sgtitle('Lsim reponses')
hold off
```



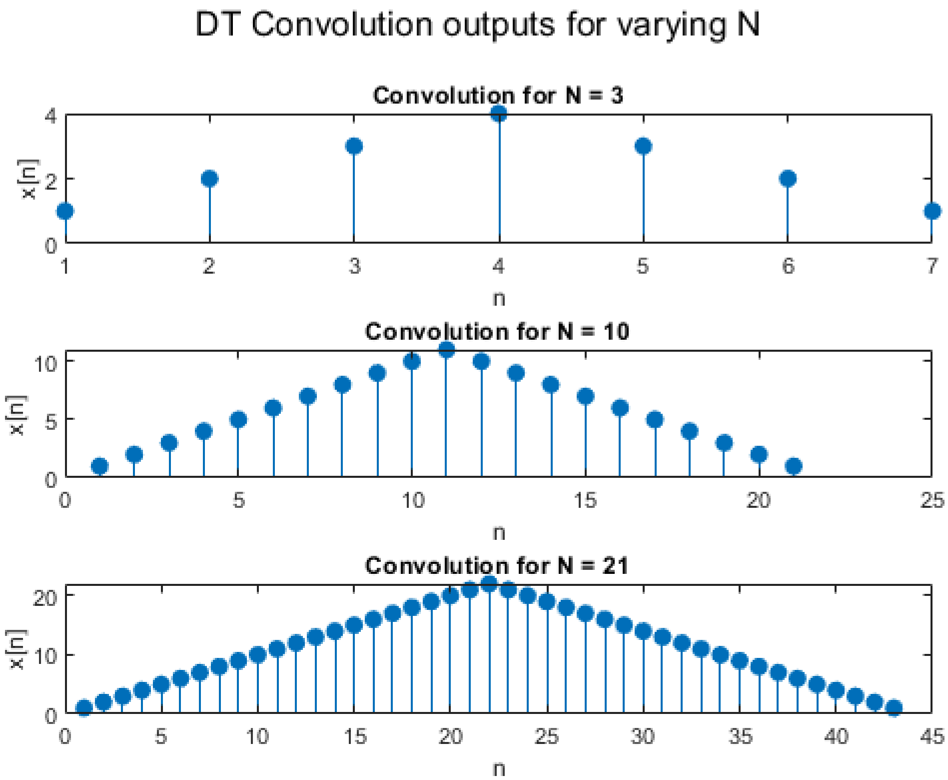
Part 2 C

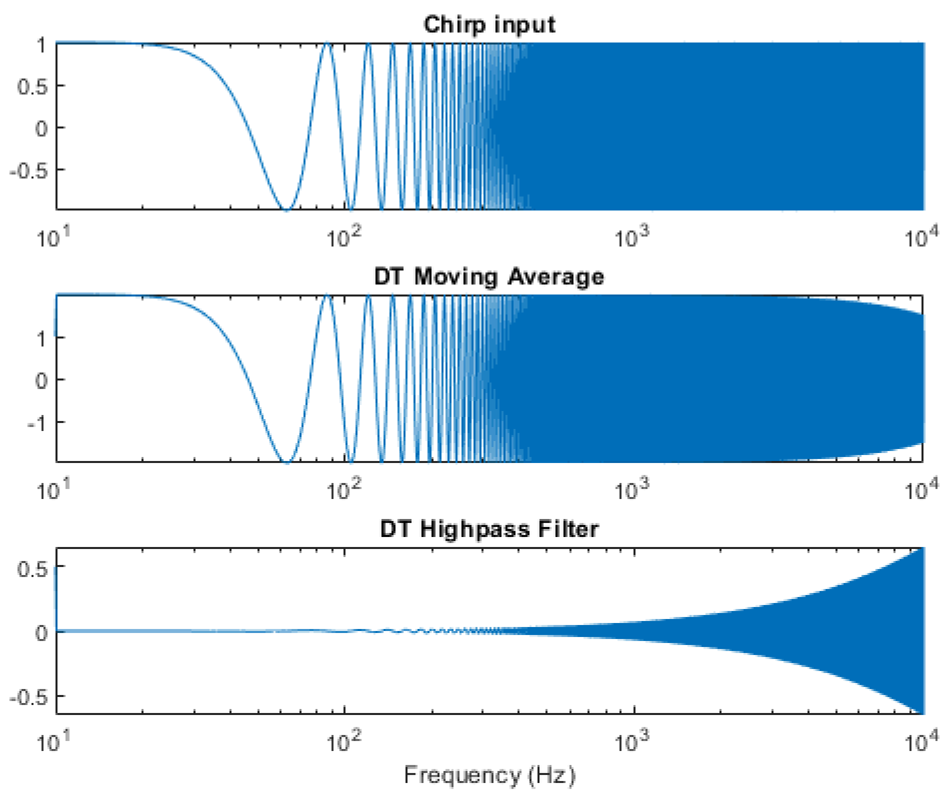
```
figure()
NRange = [3,10,21];
for i = 1:3 % range of 3 due to 3 N values
N = NRange(i);

r = zeros(1,(2*N)+1);
r(1,1:N+1)= 1;

x = filter(r,1,r); %calculate convolution by usin the filter function

% Subplot results
subplot(3,1,i)
stem(x,'filled') %Use Stem plot for DT
title(['Convolution for N = ', num2str(N)]);
xlabel('n');
ylabel('x[n]');
end
sgtitle('DT Convolution outputs for varying N')
```



Chirp Sample Code

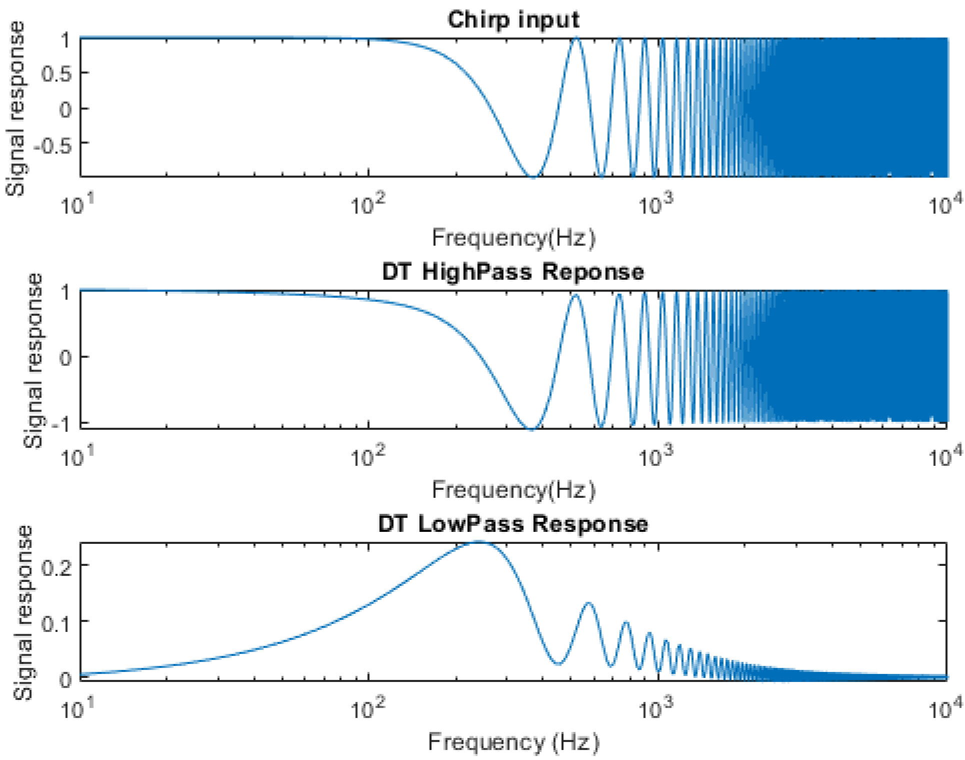
```
% sample code to define chirp signal and filter it to show approximate
% system frequency response. Shown here for simple FIR HPF and LPF
fs = 44.1e3; % sampling frequency
dT = 1/fs; % sampling period
t = 0:dT:3; % time vector
fmin = 10; fmax = 10e3; % 10000; % min and max frequencies for chirp
fchirp = (fmax-fmin).*t/max(t)+fmin; % chirp instantaneous frequency
xchirp = cos(2*pi*fchirp/2.*t); % chirp signal
% xchirp = chirp(t,fmin,max(t),fmax,'logarithmic'); % use of Matlab chirp with
 logarithmic frequency variation
sound(xchirp,fs)
ychirpHPF = filter(.5*[1 -1],1,xchirp);
ychirpMA = filter(ones(2,1),1,xchirp);
% visualization - log-frequency linear-amplitude
figure, subplot(3,1,1), plot(fchirp,xchirp); title('Chirp input')
subplot(3,1,2), plot(fchirp,(ychirpMA)); title('DT Moving Average')
subplot(3,1,3), plot(fchirp,(ychirpHPF)); title('DT Highpass Filter'),
 xlabel('Frequency (Hz)')
% to visualize with log or linear scale for frequency or amplitude, use this
 code accordingly
for i = 1:3, subplot(3,1,i), set(gca,'XScale','log'),
 set(gca,'YScale','linear'), axis tight, end
```


DT Convolution outputs for varying N
Convolution for N = 3
x[n]
n
Convolution for N = 10
x[n]
n
Convolution for N = 21
x[n]
n


Chirp input
DT Moving Average
DT Highpass Filter
Frequency (Hz)

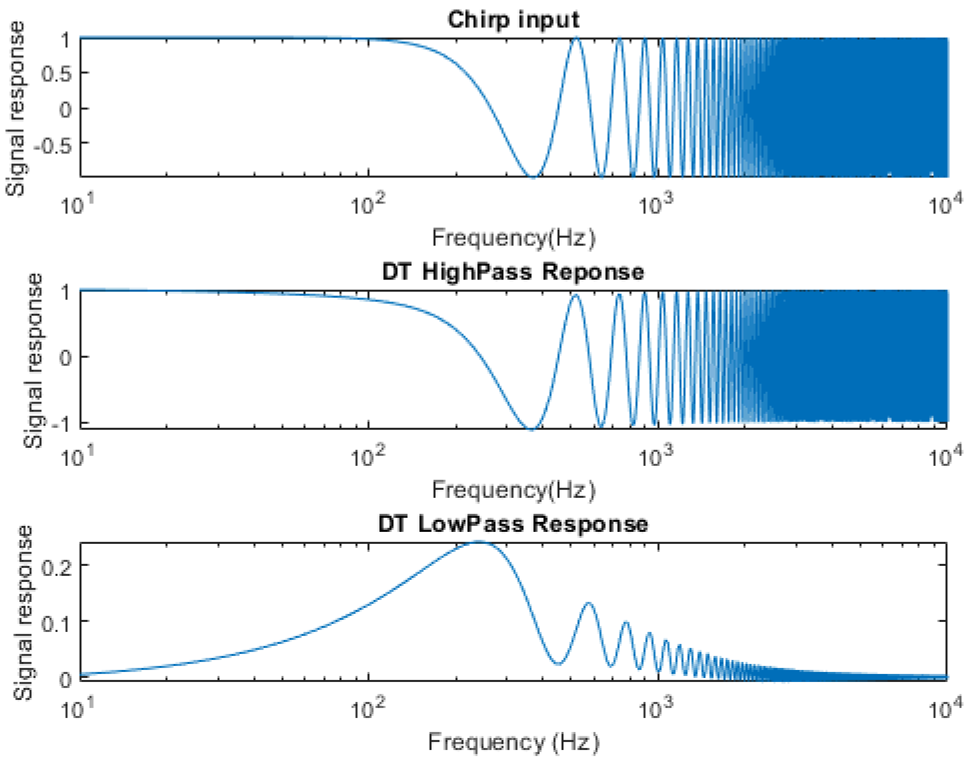
Part 3 A

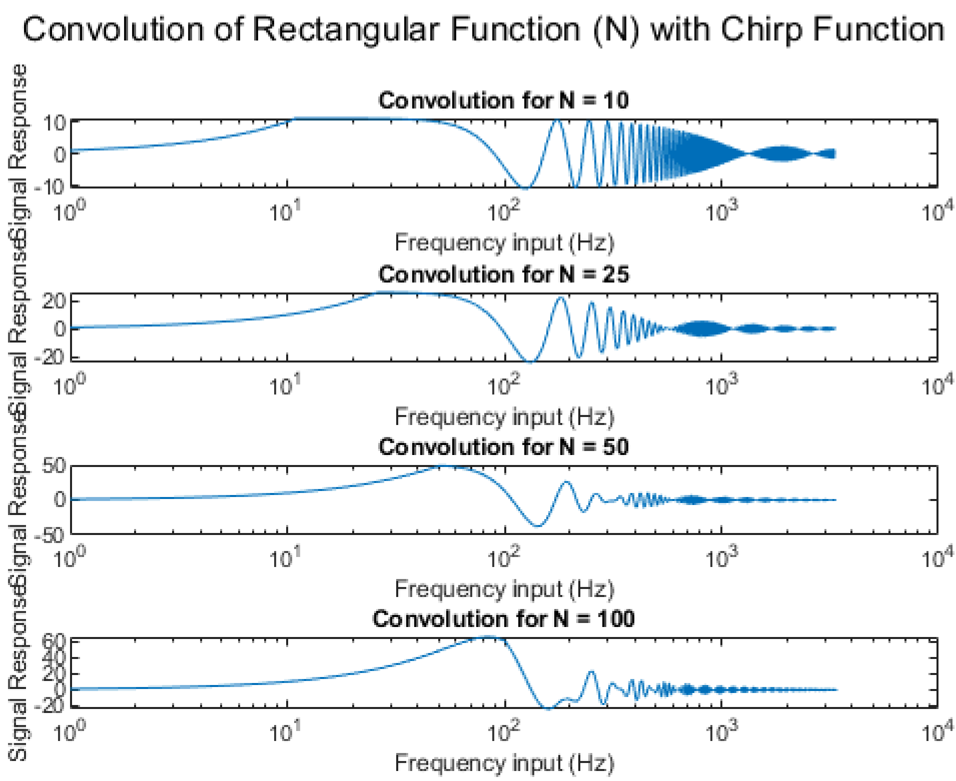
```
fs = 44.1e3; % sampling frequency
dT = 1/fs; % sampling period
t = 0:dT:15*tau; % time vector
fmin = 10; fmax = 10e3; % 10000; % min and max frequencies for chirp
fchirp = (fmax-fmin).*t/max(t)+fmin; % chirp instantaneous frequency
xchirp = cos(2*pi*fchirp/2.*t); % chirp signal

%Highpass repsonse
a = [1,(samplePeriod/tau) - 1];
b = [1,-1];
chirpFilter_hi = filter(b,a,xchirp);

%Lowpass response
a = [1, samplePeriod/tau-1];
b = samplePeriod/tau;
chirpFilter_lo = filter(b,a,xchirp);

figure, subplot(3,1,1), plot(fchirp,xchirp); title('Chirp input')
xlabel('Frequency(Hz)')
ylabel('Signal response')
subplot(3,1,2), plot(fchirp,(chirpFilter_hi)); title('DT HighPass Reponse')
xlabel('Frequency(Hz)')
ylabel('Signal response')
subplot(3,1,3), plot(fchirp,(chirpFilter_lo)); title('DT LowPass Response'),
 xlabel('Frequency (Hz)')
ylabel('Signal response')
% to visualize with log or linear scale for frequency or amplitude, use this
 code accordingly
for i = 1:3, subplot(3,1,i), set(gca,'XScale','log'),
 set(gca,'YScale','linear'), axis tight, end
sound(chirpFilter_lo)
```



Part 3 B

```
NRange = [10, 25, 50, 100];
figure();
for i = 1:length(NRange)
N = NRange(i);
r = ones(1,N+1);

y = conv(r,xchirp);
subplot(4,1,i)
plot(y)
set(gca,'XScale','log')
title(['Convolution for N = ',num2str(N)]);
xlabel('Frequency input (Hz)');
ylabel('Signal Response');
sgtitle('Convolution of Rectangular Function (N) with Chirp Function');
end
```


Chirp input
Signal response
Frequency(Hz)
DT HighPass Reponse
Signal response
Frequency(Hz)
DT LowPass Response
Signal response
Frequency (Hz)


Convolution of Rectangular Function (N) with Chirp Function
Convolution for N = 10
Convolution for N = 25
Convolution for N = 50
Convolution for N = 100
Signal Response
Frequency input (Hz)

Part 4: System 3A

```
[inputFunction, Fs] = audioread('Sound.mp3','native');
inputFunction = inputFunction(:,1);

%Hipasss
a = [1,(samplePeriod/tau) - 1];
b = [1,-1];
hi_soundA = filter(b,a,inputFunction);

%Lowpass
a = [1, samplePeriod/tau-1];
b = samplePeriod/tau;
lo_soundA = filter(b,a,inputFunction);
```

Play Hi A

```
sound(hi_soundA,Fs)
```

Play Lo A

```
sound(lo_soundA,Fs)
```

Part 4: System 3B

```
NRange = [10, 25, 50, 100];
```

Play N=10

```
N = NRange(1);
r = ones(1,N+1);
y10 = conv(r,inputFunction);
sound(y10,Fs);
```

Play N=25

```
N = NRange(2);
r = ones(1,N+1);
y25 = conv(r,inputFunction);
sound(y25,Fs);
```

Play N=50

```
N = NRange(3);
r = ones(1,N+1);
y50 = conv(r,inputFunction);
sound(y50,Fs);
```

Play N=100

```
N = NRange(4);
r = ones(1,N+1);
y100 = conv(r,inputFunction);
sound(y100,Fs);
```

Final Question 4 Observations

```
%The chosen audio file was unafected by the highpass filter, while the
%sound output was extremely damped by the lowpass filter. This is due to
%the frequency of the audio signal being higher than the cutoff frequency
%denoted by R and C. The convolution with the DT rectangular function
%yeilds interesting results. The audio file is completely distorted,
%resembling the effects of high audio compression or blown out speakers.
%Noticably, this adverse effect is amplified by the size of N. This makes
%sense because the higher the N, the longer the square wave being applied,
%allowing the distortion to have a higher effect on the audio signal.
```

Published with MATLAB® R2022a