Digital Signal Processing

Final project

Name: Youssef Hassan

ID: 6259

Name: Nour El-Din Hazem

ID: 6261

Name: Adham Wael

ID: 6486

Code Part

```
function varargout = untitled(varargin)
% UNTITLED MATLAB code for untitled.fig
%      UNTITLED, by itself, creates a new UNTITLED or raises the existing
%      singleton*.
%
%      H = UNTITLED returns the handle to a new UNTITLED or the handle to
%      the existing singleton*.
%
%      UNTITLED('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in UNTITLED.M with the given input arguments.
%
%      UNTITLED('Property','Value',...) creates a new UNTITLED or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before untitled_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to untitled_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help untitled

% Last Modified by GUIDE v2.5 15-Jun-2021 20:51:05

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @untitled_OpeningFcn, ...
                   'gui_OutputFcn',  @untitled_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before untitled is made visible.
function untitled_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
```

```
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to untitled (see VARARGIN)

% Choose default command line output for untitled
vol = 5;
set(handles.slider17,'value',vol);
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes untitled wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = untitled_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in browse.
function browse_Callback(hObject, eventdata, handles)
% hObject    handle to browse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


%To get the wav file from the user
[filename pathname] = uigetfile({'*.wav'},'File Selector');
handles.fullpathname = strcat(pathname, filename);

set(handles.text3, 'String',handles.fullpathname) %showing fullpathname
guidata(hObject,handles)


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
```

```
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider2_Callback(hObject, ~, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider3_Callback(hObject, eventdata, handles)
% hObject    handle to slider3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes on slider movement.
function slider4_Callback(hObject, eventdata, handles)
% hObject    handle to slider4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider5_Callback(hObject, eventdata, handles)
% hObject    handle to slider5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider6_Callback(hObject, eventdata, handles)
% hObject    handle to slider6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
```

```
% --- Executes during object creation, after setting all properties.
function slider6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
% --- Executes on slider movement.
function slider9_Callback(hObject, eventdata, handles)
% hObject    handle to slider9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end


% --- Executes on slider movement.
function slider7_Callback(hObject, eventdata, handles)
% hObject    handle to slider7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
% --- Executes on slider movement.
function slider8_Callback(hObject, eventdata, handles)
% hObject    handle to slider8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

%Play button function
function play_equalizer(hObject, handles)
global player;

[handles.y,handles.Fs] = audioread(handles.fullpathname);
handles.Volume=get(handles.slider17,'value');   %adjust the volume

%get the values the user enter at each slider
handles.a1=get(handles.slider1,'value');
handles.a2=get(handles.slider2,'value');
handles.a3=get(handles.slider3,'value');
handles.a4=get(handles.slider4,'value');
handles.a5=get(handles.slider5,'value');
handles.a6=get(handles.slider6,'value');
handles.a7=get(handles.slider7,'value');
handles.a8=get(handles.slider8,'value');
handles.a9=get(handles.slider9,'value');

%set the text with the value of the corresponding slider
set(handles.text13,'String',handles.a1);
set(handles.text14,'String',handles.a2);
set(handles.text15,'String',handles.a3);
set(handles.text16,'String',handles.a4);
set(handles.text17,'String',handles.a5);
set(handles.text18,'String',handles.a6);
set(handles.text19,'String',handles.a7);
set(handles.text20,'String',handles.a8);
set(handles.text21,'String',handles.a9);

Y=real(fft(handles.y)); %Calculate the frequency Domain

figure; %plot the time & frequency domian of the original signal
```

```
subplot(2,1,1);plot(handles.y);title('Time Domain');
subplot(2,1,2);plot(Y);title('Frequency Domain');




% --- Executes on button press in play.
function play_Callback(hObject, eventdata, handles) %play button
global player;
play_equalizer(hObject, handles);
play(player);
guidata(hObject,handles)
% hObject    handle to play (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles) %pause button
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global player;
play_equalizer(hObject, handles);
pause(player);
guidata(hObject,handles)


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)%stop button
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global player;
play_equalizer(hObject, handles);
stop(player);
guidata(hObject,handles)


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles) %resume button
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global player;
play_equalizer(hObject, handles);
resume(player);
guidata(hObject,handles)


% --- Executes on slider movement.
function slider17_Callback(hObject, eventdata, handles)
% hObject    handle to slider17 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider


% --- Executes during object creation, after setting all properties.
function slider17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

%FIR filter function
function FIR_Filter(hObject,handles)
global player;
[handles.y,handles.Fs] = audioread(handles.fullpathname); %get y & Fs of the wav file
handles.Volume=get(handles.slider17,'value'); %get vloume value

%get the values from the position of each slider
handles.a1=get(handles.slider1,'value');
handles.a2=get(handles.slider2,'value');
handles.a3=get(handles.slider3,'value');
handles.a4=get(handles.slider4,'value');
handles.a5=get(handles.slider5,'value');
handles.a6=get(handles.slider6,'value');
handles.a7=get(handles.slider7,'value');
handles.a8=get(handles.slider8,'value');
handles.a9=get(handles.slider9,'value');

%set the value of the text with the corresponding slider
set(handles.text13,'String',handles.a1);
set(handles.text14,'String',handles.a2);
set(handles.text15,'String',handles.a3);
set(handles.text16,'String',handles.a4);
set(handles.text17,'String',handles.a5);
set(handles.text18,'String',handles.a6);
set(handles.text19,'String',handles.a7);
set(handles.text20,'String',handles.a8);
set(handles.text21,'String',handles.a9);

%Convert decibels to magnitude
handles.a1=db2mag(handles.a1);
handles.a2=db2mag(handles.a2);
handles.a3=db2mag(handles.a3);
handles.a4=db2mag(handles.a4);
handles.a5=db2mag(handles.a5);
handles.a6=db2mag(handles.a6);
handles.a7=db2mag(handles.a7);
```

```
handles.a8=db2mag(handles.a8);
handles.a9=db2mag(handles.a9);
handles.Fnew=get(handles.edit1,'value'); %get output sample factor


%assume order with 26 & set the denominatorominator with 1
order=26;
denominator=1;

%0-170 Hz
cut_off=170;
numerator1=fir1(order,cut_off/(handles.Fs/2),'low'); %calculate the numerator and normalize it
transfer1=tf(numerator1,denominator);   %calculate transfer function
[h1,w1]=freqz(numerator1,denominator);  %calcutate frequency response
[h11,t11]=impz(numerator1,denominator); %calculate impluse response
[h12,t12]=stepz(numerator1,denominator);%calculate step response
mag1=abs(h1);                           %calculate magnitude
phase1=angle(h1)*180/pi;                %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('0-170 Hz');
subplot(3,2,1);plot(w1/pi,mag1);title('Magnitude');
subplot(3,2,2);plot(w1/pi,phase1);title('Phase');
subplot(3,2,3);stem(t11,h11);title('Impluse Response');
subplot(3,2,4);stem(t12,h12);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer1);title('Zeros and poles');

yf1=handles.a1*filter(numerator1,1,handles.y); %calculate yf1 to plot it at the end

%170-310
f1=171;
f2=310;
numerator2=fir1(order,[f1/(handles.Fs/2) f2/(handles.Fs/2)],'bandpass');%calculate the numerator
and normalize it
transfer2=tf(numerator2,denominator);%calculate transfer function
[h2,w2]=freqz(numerator2,denominator);%calcutate frequency response
[h21,t21]=impz(numerator2,denominator);%calculate impluse response
[h22,t22]=stepz(numerator2,denominator);%calculate step response
mag2=abs(h2);%calculate magnitude
phase2=angle(h2)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('170-310 Hz');
subplot(3,2,1);plot(w2/pi,mag2);title('Magnitude');
subplot(3,2,2);plot(w2/pi,phase2);title('Phase');
subplot(3,2,3);stem(t21,h21);title('Impluse Response');
subplot(3,2,4);stem(t22,h22);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer2);title('Zeros and poles');

yf2=handles.a2*filter(numerator2,1,handles.y); %calculate yf2 to plot it at the end

% 310-600 Hz
f3=311;
f4=600;
numerator3=fir1(order,[f3/(handles.Fs/2) f4/(handles.Fs/2)],'bandpass');%calculate the numerator
and normalize it
```

```
transfer3=tf(numerator3,denominator);%calculate transfer function
[h3,w3]=freqz(numerator3,denominator);%calcutate frequency response
[h31,t31]=impz(numerator3,denominator);%calculate impluse response
[h32,t32]=stepz(numerator3,denominator);%calculate step response
mag3=abs(h3);%calculate magnitude
phase3=angle(h3)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('310-600 Hz');
subplot(3,2,1);plot(w3/pi,mag3);title('Magnitude');
subplot(3,2,2);plot(w3/pi,phase3);title('Phase');
subplot(3,2,3);stem(t31,h31);title('Impluse Response');
subplot(3,2,4);stem(t32,h32);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer3);title('Zeros and poles');

yf3=handles.a3*filter(numerator3,1,handles.y); %calculate yf3 to plot it at the end

% 600-1000 Hz
 f4=601;
 f5=1000;
 numerator4=fir1(order,[f4/(handles.Fs/2) f5/(handles.Fs/2)],'bandpass');%calculate the numerator
and normalize it
transfer4=tf(numerator4,denominator);%calculate transfer function
[h4,w4]=freqz(numerator4,denominator);%calcutate frequency response
[h41,t41]=impz(numerator4,denominator);%calculate impluse response
[h42,t42]=stepz(numerator4,denominator);%calculate step response
mag4=abs(h4);%calculate magnitude
phase4=angle(h4)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('600-1000 Hz');
subplot(3,2,1);plot(w4/pi,mag4);title('Magnitude');
subplot(3,2,2);plot(w4/pi,phase4);title('Phase');
subplot(3,2,3);stem(t41,h41);title('Impluse Response');
subplot(3,2,4);stem(t42,h42);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer4);title('Zeros and poles');

 yf4=handles.a4*filter(numerator4,1,handles.y); %calculate yf4 to plot it at the end

%1000-3000 Hz
 f5=1001;
 f6=3000;
 numerator5=fir1(order,[f5/(handles.Fs/2) f6/(handles.Fs/2)],'bandpass');%calculate the numerator
and normalize it
 transfer5=tf(numerator5,denominator);%calculate transfer function
[h5,w5]=freqz(numerator5,denominator);%calcutate frequency response
[h51,t51]=impz(numerator5,denominator);%calculate impluse response
[h52,t52]=stepz(numerator5,denominator);%calculate step response
mag5=abs(h5);%calculate magnitude
phase5=angle(h5)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('1000-3000 Hz');
subplot(3,2,1);plot(w5/pi,mag5);title('Magnitude');
subplot(3,2,2);plot(w5/pi,phase5);title('Phase');
subplot(3,2,3);stem(t51,h51);title('Impluse Response');
subplot(3,2,4);stem(t52,h52);title('Step Response');
```

```
subplot(3,2,[5 6]);pzmap(transfer5);title('Zeros and poles');

 yf5=handles.a5*filter(numerator5,1,handles.y); %calculate yf5 to plot it at the end

% 3000-6000 Hz
  f7=3001;
  f8=6000;
  numerator6=fir1(order,[f7/(handles.Fs/2) f8/(handles.Fs/2)],'bandpass');%calculate the
numerator and normalize it
 transfer6=tf(numerator6,denominator);%calculate transfer function
[h6,w6]=freqz(numerator6,denominator);%calcutate frequency response
[h61,t61]=impz(numerator6,denominator);%calculate impluse response
[h62,t62]=stepz(numerator6,denominator);%calculate step response
mag6=abs(h6);%calculate magnitude
phase6=angle(h6)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('3000-6000 Hz');
subplot(3,2,1);plot(w6/pi,mag6);title('Magnitude');
subplot(3,2,2);plot(w6/pi,phase6);title('Phase');
subplot(3,2,3);stem(t61,h61);title('Impluse Response');
subplot(3,2,4);stem(t62,h62);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer6);title('Zeros and poles');

  yf6=handles.a6*filter(numerator6,1,handles.y); %calculate yf6 to plot it at the end

% 6000-12000 Hz
  f9=6001;
  f10=12000;
  numerator7=fir1(order,[f9/(handles.Fs/2) f10/(handles.Fs/2)],'bandpass');%calculate the
numerator and normalize it
 transfer7=tf(numerator7,denominator);%calculate transfer function
[h7,w7]=freqz(numerator7,denominator);%calcutate frequency response
[h71,t71]=impz(numerator7,denominator);%calculate impluse response
[h72,t72]=stepz(numerator7,denominator);%calculate step response
mag7=abs(h7);%calculate magnitude
phase7=angle(h7)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('6000-12000 Hz');
subplot(3,2,1);plot(w7/pi,mag7);title('Magnitude');
subplot(3,2,2);plot(w7/pi,phase7);title('Phase');
subplot(3,2,3);stem(t71,h71);title('Impluse Response');
subplot(3,2,4);stem(t72,h72);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer7);title('Zeros and poles');

  yf7=handles.a7*filter(numerator7,1,handles.y); %calculate yf7 to plot it at the end

% 12000-14000 Hz
  f11=12001;
  f12=14000;
  numerator8=fir1(order,[f11/(handles.Fs/2) f12/(handles.Fs/2)],'bandpass');%calculate the
numerator and normalize it
 transfer8=tf(numerator8,denominator);%calculate transfer function
[h8,w8]=freqz(numerator8,denominator);%calcutate frequency response
[h81,t81]=impz(numerator8,denominator);%calculate impluse response
```

```
[h82,t82]=stepz(numerator8,denominator);%calculate step response
mag8=abs(h8);%calculate magnitude
phase8=angle(h8)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('12000-14000 Hz');
subplot(3,2,1);plot(w8/pi,mag8);title('Magnitude');
subplot(3,2,2);plot(w8/pi,phase8);title('Phase');
subplot(3,2,3);stem(t81,h81);title('Impluse Response');
subplot(3,2,4);stem(t82,h82);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer8);title('Zeros and poles');

  yf8=handles.a8*filter(numerator8,1,handles.y); %calculate yf8 to plot it at the end

 % 14000-16000 Hz
  f13=14001;
  f14=16000;
  numerator9=fir1(order,[f13/(handles.Fs/2) f14/(handles.Fs/2)],'bandpass');%calculate the
numerator and normalize it
 transfer9=tf(numerator9,denominator);%calculate transfer function
[h9,w9]=freqz(numerator9,denominator);%calcutate frequency response
[h91,t91]=impz(numerator9,denominator);%calculate impluse response
[h92,t92]=stepz(numerator9,denominator);%calculate step response
mag9=abs(h9);%calculate magnitude
phase9=angle(h9)*180/pi;   %calculate phase
figure;  %plot magnitude,phase,impluse reponse,step response,zeros&poles
suptitle('14000-16000 Hz');
subplot(3,2,1);plot(w9/pi,mag9);title('Magnitude');
subplot(3,2,2);plot(w9/pi,phase9);title('Phase');
subplot(3,2,3);stem(t91,h91);title('Impluse Response');
subplot(3,2,4);stem(t92,h92);title('Step Response');
subplot(3,2,[5 6]);pzmap(transfer9);title('Zeros and poles');

  yf9=handles.a9*filter(numerator9,1,handles.y); %calculate yf9 to plot it at the end
  handles.yfT=yf1+yf2+yf3+yf4+yf5+yf6+yf7+yf8+yf9; %calculate yfT
  handles.yfT=resample(handles.yfT,1,1);
  YfT=real(fft(handles.yfT)); %calculate frequency domain
player = audioplayer(handles.Volume*handles.yfT, handles.Fs); %play sound after filter
play(player);
figure; %plot time & frequency domain
subplot(2,1,1);plot(handles.yfT);title('Time Domain');
subplot(2,1,2);plot(YfT);title('Frequency Domain');

% --- Executes on button press in FIR.
function FIR_Callback(hObject, eventdata, handles)%fir button
% hObject    handle to FIR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
FIR_Filter(hObject,handles);


function IIR_Filter(hObject,handles)
global player;
[handles.y,handles.Fs] = audioread(handles.fullpathname);%calculate the y & fs of the wav file
handles.Volume=get(handles.slider17,'value');%get the volume from the slider
```

```
%get the values from the position of each slider
handles.a1=get(handles.slider1,'value');
handles.a2=get(handles.slider2,'value');
handles.a3=get(handles.slider3,'value');
handles.a4=get(handles.slider4,'value');
handles.a5=get(handles.slider5,'value');
handles.a6=get(handles.slider6,'value');
handles.a7=get(handles.slider7,'value');
handles.a8=get(handles.slider8,'value');
handles.a9=get(handles.slider9,'value');

%set the value of the text with the corresponding slider
set(handles.text13,'String',handles.a1);
set(handles.text14,'String',handles.a2);
set(handles.text15,'String',handles.a3);
set(handles.text16,'String',handles.a4);
set(handles.text17,'String',handles.a5);
set(handles.text18,'String',handles.a6);
set(handles.text19,'String',handles.a7);
set(handles.text20,'String',handles.a8);
set(handles.text21,'String',handles.a9);

%Convert decibels to magnitude
handles.a1=db2mag(handles.a1);
handles.a2=db2mag(handles.a2);
handles.a3=db2mag(handles.a3);
handles.a4=db2mag(handles.a4);
handles.a5=db2mag(handles.a5);
handles.a6=db2mag(handles.a6);
handles.a7=db2mag(handles.a7);
handles.a8=db2mag(handles.a8);
handles.a9=db2mag(handles.a9);

handles.Fnew=get(handles.edit1,'value'); %get output sample factor

fm = handles.Fs/2; % to normalize each Fs
order2 = 4; %assume order of iir filter with 4

% %0 - 170 - Calculations
f1 = 170/(fm);
[n1, d1] = butter(order2, f1);
transf1 = tf(n1, d1);
[h1, w1] = freqz(n1, d1);
mag1 = abs(h1);
phase1 = angle(h1) * 180 / pi;
%0 - 170 - Plotting
figure;
suptitle('0-170 Hz');
subplot(3,2,1); plot(w1/pi,mag1);title('Magnitude');
subplot(3,2,2); plot(w1/pi,phase1);title('Phase');
subplot(3,2,3); stem(impulse(transf1));title('Impulse Response');
subplot(3,2,4); stem(step(transf1));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf1);title('Zeros and Poles');
```

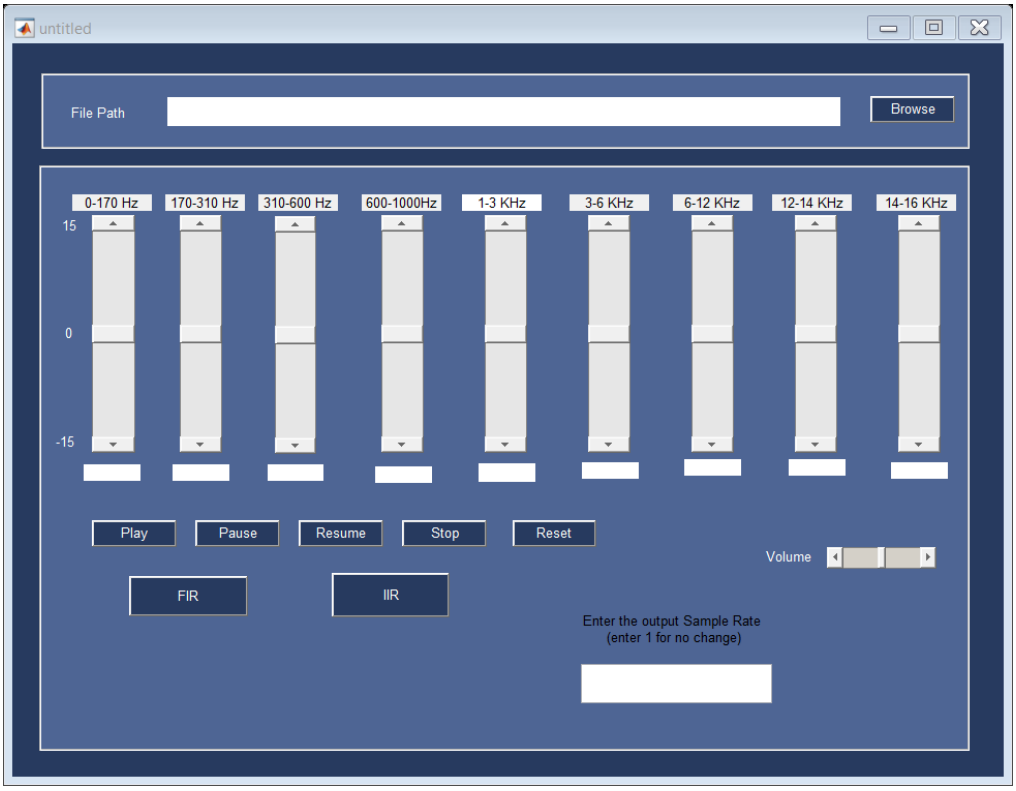
```
%composite 1
yi1 = handles.a1*filter(n1, d1, handles.y);

%170 - 310 - Calculations
f21 = 170;
f22 = 310;
f2 = [f21 f22]/(fm);
[n2, d2] = butter(order2, f2);
transf2 = tf(n2, d2);
[h2, w2] = freqz(n2, d2);
mag2 = abs(h2);
phase2 = angle(h2) * 180 / pi;
%170 - 310 - Plotting
figure;
suptitle('170-310 Hz');
subplot(3,2,1); plot(w2/pi,mag2);title('Magnitude');
subplot(3,2,2); plot(w2/pi,phase2);title('Phase');
subplot(3,2,3); stem(impulse(transf2));title('Impulse Response');
subplot(3,2,4); stem(step(transf2));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf2);title('Zeros and Poles');
%composite 2
yi2 = handles.a2*filter(n2, d2, handles.y);

%310 - 600 - Calculations
f31 = 310;
f32 = 600;
f3 = [f31 f32]/(fm);
[n3, d3] = butter(order2, f3);
transf3 = tf(n3, d3);
[h3, w3] = freqz(n3, d3);
mag3 = abs(h3);
phase3 = angle(h3) * 180 / pi;
%310 - 600 - Plotting
figure;
suptitle('310-600 Hz');
subplot(3,2,1); plot(w3/pi,mag3);title('Magnitude');
subplot(3,2,2); plot(w3/pi,phase3);title('Phase');
subplot(3,2,3); stem(impulse(transf3));title('Impulse Response');
subplot(3,2,4); stem(step(transf3));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf3);title('Zeros and Poles');
%composite 3
yi3 = handles.a3*filter(n3, d3, handles.y);

%600 - 1000 - Calculations
f41 = 600;
f42 = 1000;
f4 = [f41 f42]/(fm);
[n4, d4] = butter(order2, f4);
transf4 = tf(n4, d4);
[h4, w4] = freqz(n4, d4);
mag4 = abs(h4);
phase4 = angle(h4) * 180 / pi;
%600 - 1000 - Plotting
figure;
```

```
suptitle('600-1000 Hz');
subplot(3,2,1); plot(w4/pi,mag4);title('Magnitude');
subplot(3,2,2); plot(w4/pi,phase4);title('Phase');
subplot(3,2,3); stem(impulse(transf4));title('Impulse Response');
subplot(3,2,4); stem(step(transf4));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf4);title('Zeros and Poles');
%composite 4
yi4 = handles.a4*filter(n4, d4, handles.y);

%1K - 3K - Calculations
f51 = 1000;
f52 = 3000;
f5 = [f51 f52]/(fm);
[n5, d5] = butter(order2, f5);
transf5 = tf(n5, d5);
[h5, w5] = freqz(n5, d5);
mag5 = abs(h5);
phase5 = angle(h5) * 180 / pi;
%1K - 3K - Plotting
figure;
suptitle('1000 - 3000 Hz');
subplot(3,2,1); plot(w5/pi,mag5);title('Magnitude');
subplot(3,2,2); plot(w5/pi,phase5);title('Phase');
subplot(3,2,3); stem(impulse(transf5));title('Impulse Response');
subplot(3,2,4); stem(step(transf5));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf5);title('Zeros and Poles');
%composite 5
yi5 = handles.a5*filter(n5, d5, handles.y);

%3K - 6K - Calculations
f61 = 3000;
f62 = 6000;
f6 = [f61 f62]/(fm);
[n6, d6] = butter(order2, f6);
transf6 = tf(n6, d6);
[h6, w6] = freqz(n6, d6);
mag6 = abs(h6);
phase6 = angle(h6) * 180 / pi;
%3K - 6K - Plotting
figure;
suptitle('3000-6000 Hz');
subplot(3,2,1); plot(w6/pi,mag6);title('Magnitude');
subplot(3,2,2); plot(w6/pi,phase6);title('Phase');
subplot(3,2,3); stem(impulse(transf6));title('Impulse Response');
subplot(3,2,4); stem(step(transf6));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf6);title('Zeros and Poles');
%composite 6
yi6 = handles.a6*filter(n6, d6, handles.y);

%6K - 12K - Calculations
f71 = 6000;
f72 = 12000;
f7 = [f71 f72]/(fm);
[n7, d7] = butter(order2, f7);
```

```
transf7 = tf(n7, d7);
[h7, w7] = freqz(n7, d7);
mag7 = abs(h7);
phase7 = angle(h7) * 180 / pi;
%6K - 12K - Plotting
figure;
suptitle('6000-12000 Hz');
subplot(3,2,1); plot(w7/pi,mag7);title('Magnitude');
subplot(3,2,2); plot(w7/pi,phase7);title('Phase');
subplot(3,2,3); stem(impulse(transf7));title('Impulse Response');
subplot(3,2,4); stem(step(transf7));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf7);title('Zeros and Poles');
%composite 7
yi7 = handles.a7*filter(n7, d7, handles.y);

%12K - 14K - Calculations
f81 = 12000;
f82 = 14000;
f8 = [f81 f82]/(fm);
[n8, d8] = butter(order2, f8);
transf8 = tf(n8, d8);
[h8, w8] = freqz(n8, d8);
mag8 = abs(h8);
phase8 = angle(h8) * 180 / pi;
%12K - 14K - Plotting
figure;
suptitle('12000-14000 Hz');
subplot(3,2,1); plot(w8/pi,mag8);title('Magnitude');
subplot(3,2,2); plot(w8/pi,phase8);title('Phase');
subplot(3,2,3); stem(impulse(transf8));title('Impulse Response');
subplot(3,2,4); stem(step(transf8));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf8);title('Zeros and Poles');
%composite 8
yi8 = handles.a8*filter(n8, d8, handles.y);

%14K - 16K - Calculations
f91 = 14000;
f92 = 16000;
f9 = [f91 f92]/(fm);
[n9, d9] = butter(order2, f9);
transf9 = tf(n9, d9);
[h9, w9] = freqz(n9, d9);
mag9 = abs(h9);
phase9 = angle(h9) * 180 / pi;
%14K - 16K - Plotting
figure;
suptitle('14000-16000 Hz');
subplot(3,2,1); plot(w9/pi,mag9);title('Magnitude');
subplot(3,2,2); plot(w9/pi,phase9);title('Phase');
subplot(3,2,3); stem(impulse(transf9));title('Impulse Response');
subplot(3,2,4); stem(step(transf9));title('Step Response');
subplot(3,2,[5 6]);pzmap(transf9);title('Zeros and Poles');
%composite 9
yi9 = handles.a9*filter(n9, d9, handles.y);
```

```
handles.yiT = yi1+yi2+yi3+yi4+yi5+yi6+yi7+yi8+yi9;
handles.yiT=resample(handles.yiT,1,1);
YiT = real(fft(handles.yiT));
 player = audioplayer(handles.Volume*handles.yiT, handles.Fs);
 play(player);

figure;
 subplot(2,1,1);plot(handles.yiT);title('Time Domain');
subplot(2,1,2);plot(YiT);title('Frequency Domain');


% --- Executes on button press in IIR.
function IIR_Callback(hObject, eventdata, handles)
% hObject    handle to IIR (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
IIR_Filter(hObject,handles);


% --- Executes on button press in Reset.
function Reset_Callback(hObject, eventdata, handles) %reset function
% hObject    handle to Reset (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%Set the values of the slider with 0
handles.a1=0;
handles.a2=0;
handles.a3=0;
handles.a4=0;
handles.a5=0;
handles.a6=0;
handles.a7=0;
handles.a8=0;
handles.a9=0;

%set the texts with the new values
set(handles.text13,'String',handles.a1);
set(handles.text14,'String',handles.a2);
set(handles.text15,'String',handles.a3);
set(handles.text16,'String',handles.a4);
set(handles.text17,'String',handles.a5);
set(handles.text18,'String',handles.a6);
set(handles.text19,'String',handles.a7);
set(handles.text20,'String',handles.a8);
set(handles.text21,'String',handles.a9);

%set the slider with the new values
set(handles.slider1,'value',handles.a1);
set(handles.slider2,'value',handles.a2);
set(handles.slider3,'value',handles.a3);
set(handles.slider4,'value',handles.a4);
set(handles.slider5,'value',handles.a5);
set(handles.slider6,'value',handles.a6);
```

```
set(handles.slider7,'value',handles.a7);
set(handles.slider8,'value',handles.a8);
set(handles.slider9,'value',handles.a9);



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

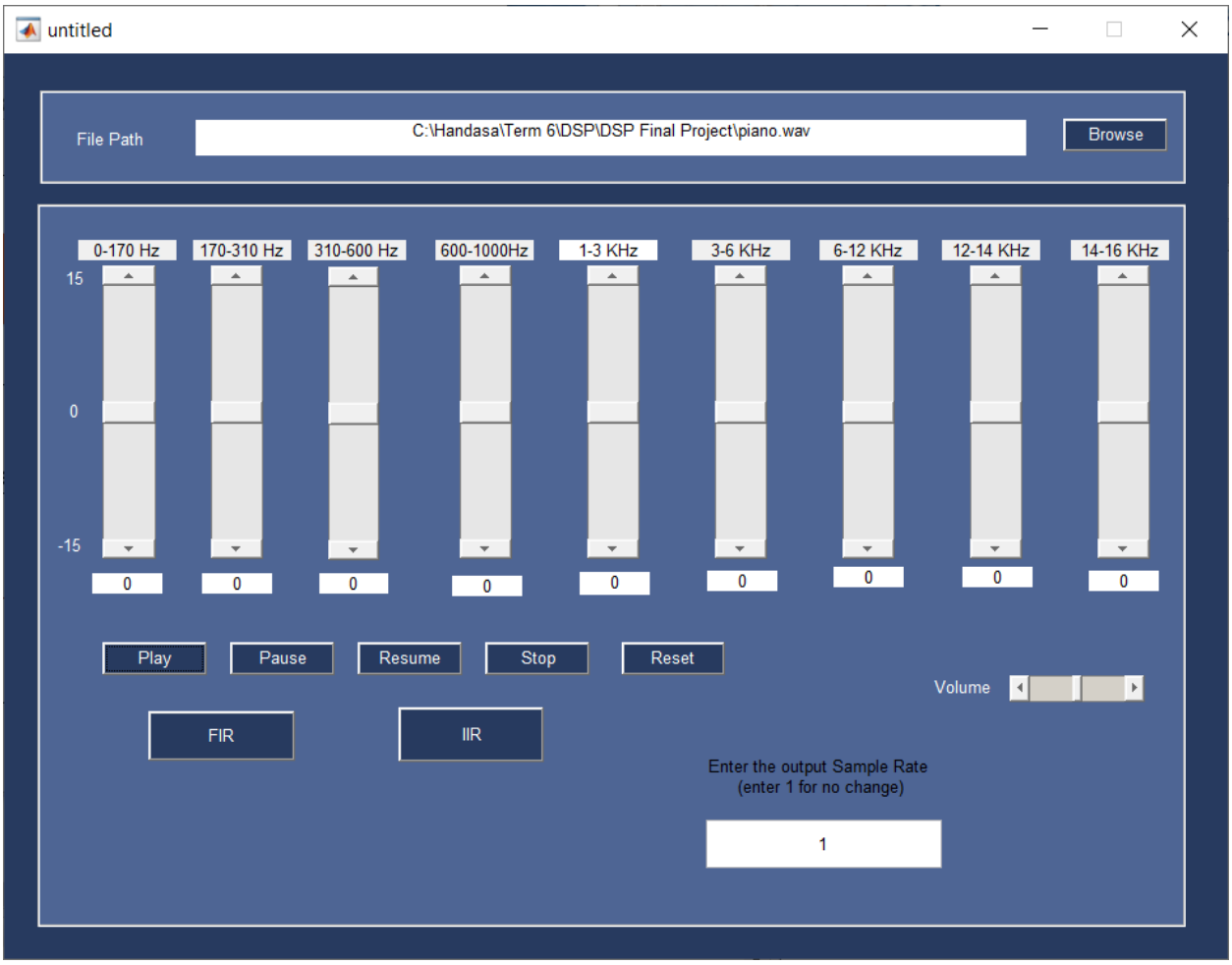


Published with MATLAB® R2016a

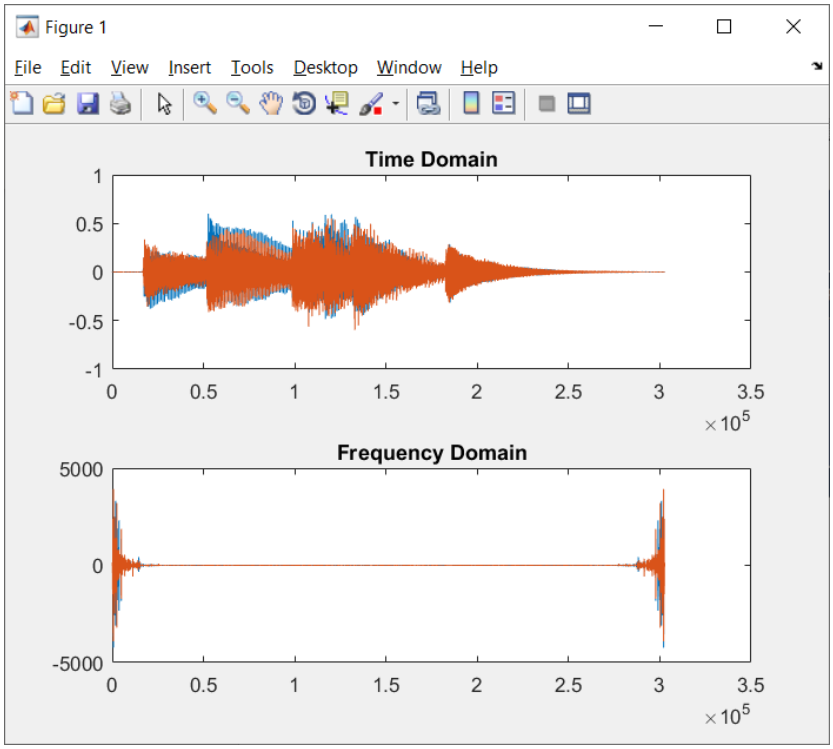
Sample Runs

1) IF the output Sample remain the same

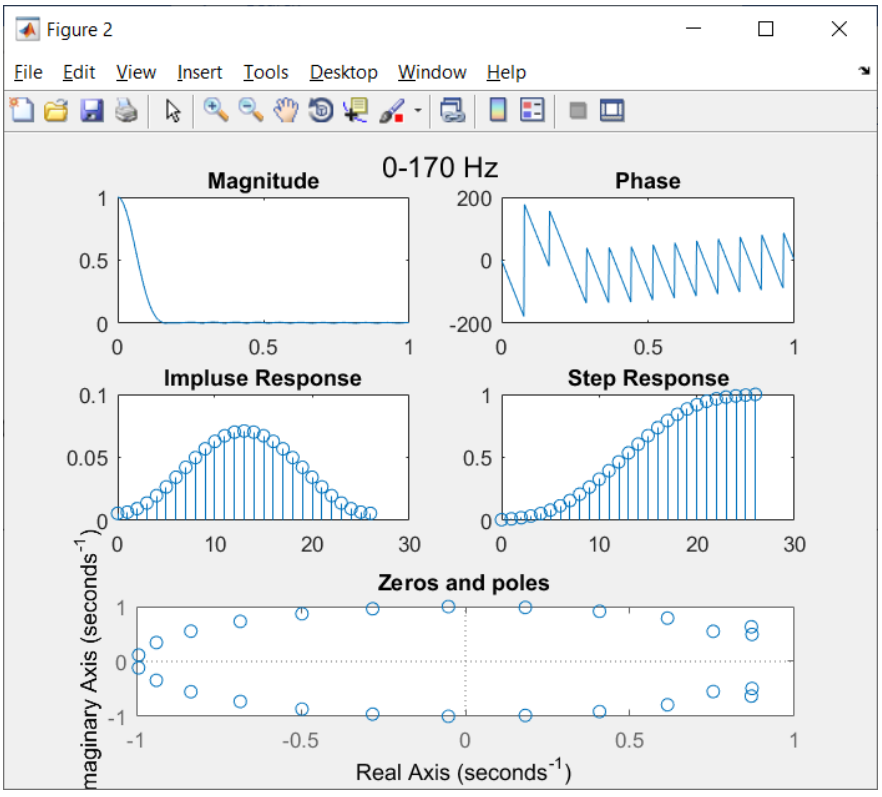
Gui view



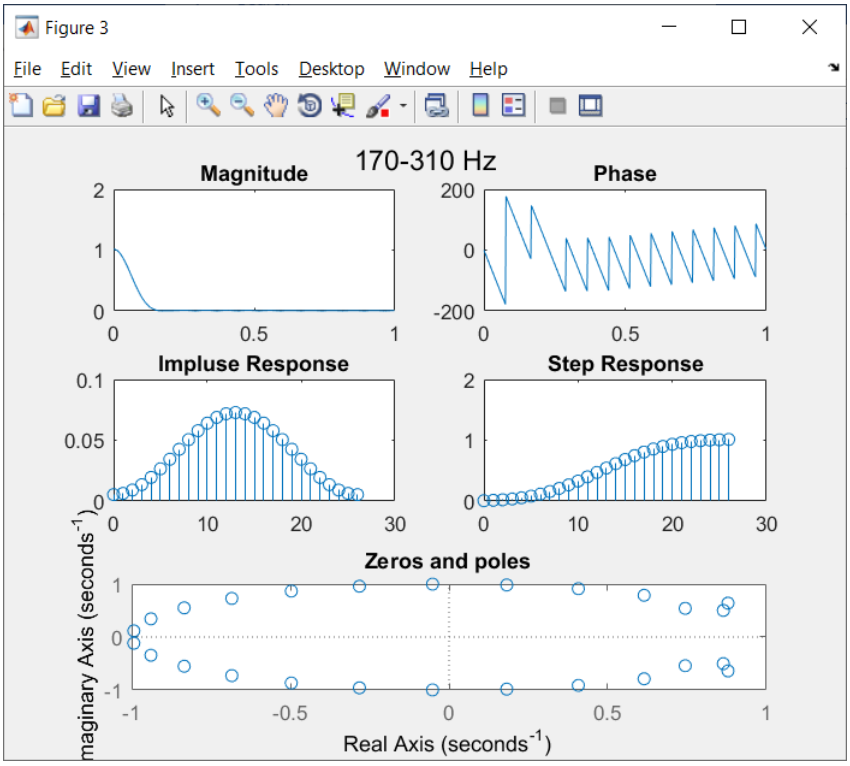
Time Domain & Frequency Domain of original signal



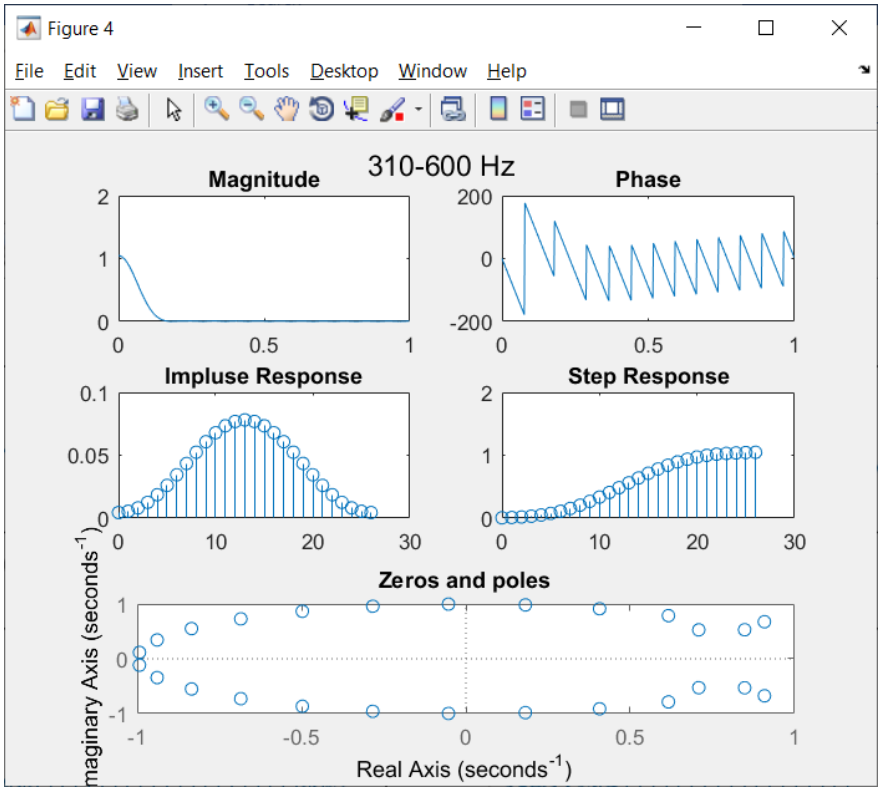
Signal after using FIR filter (0-170 Hz)



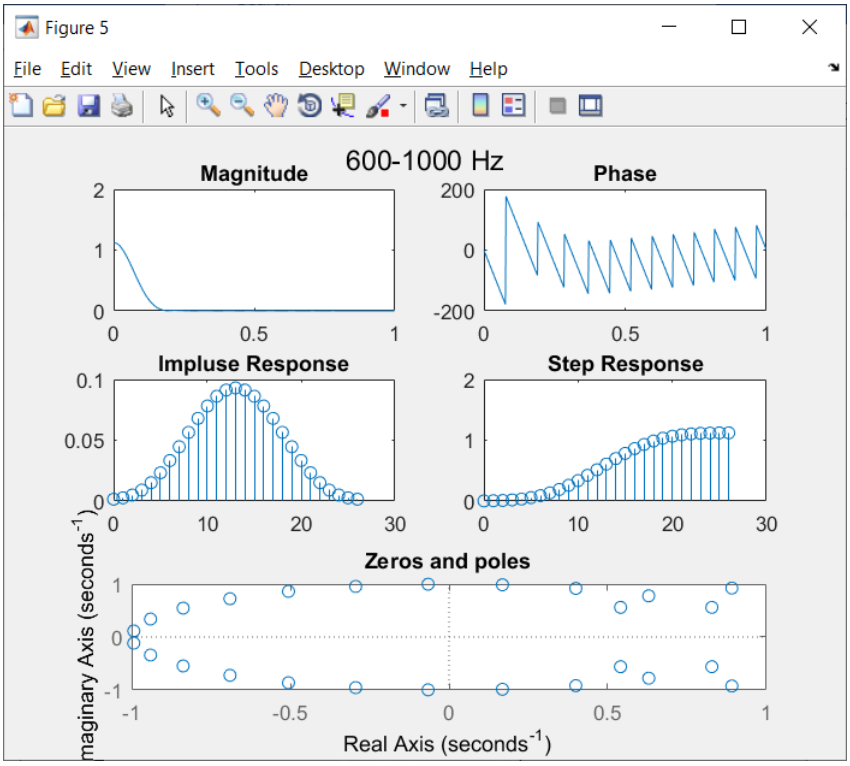
(170-310Hz)



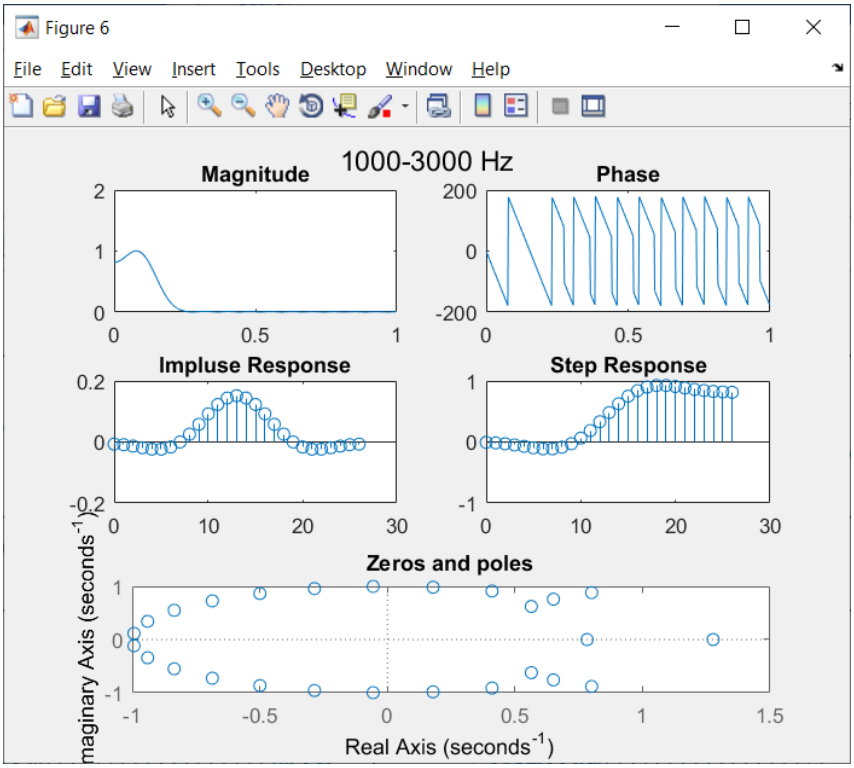
(310-600 Hz)



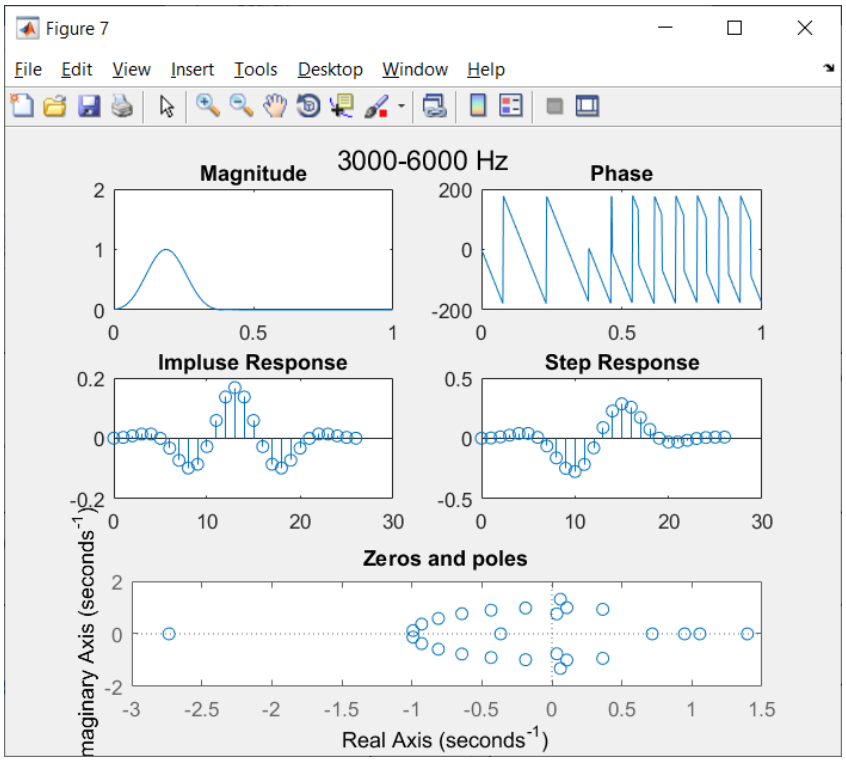
(600-1000 Hz)



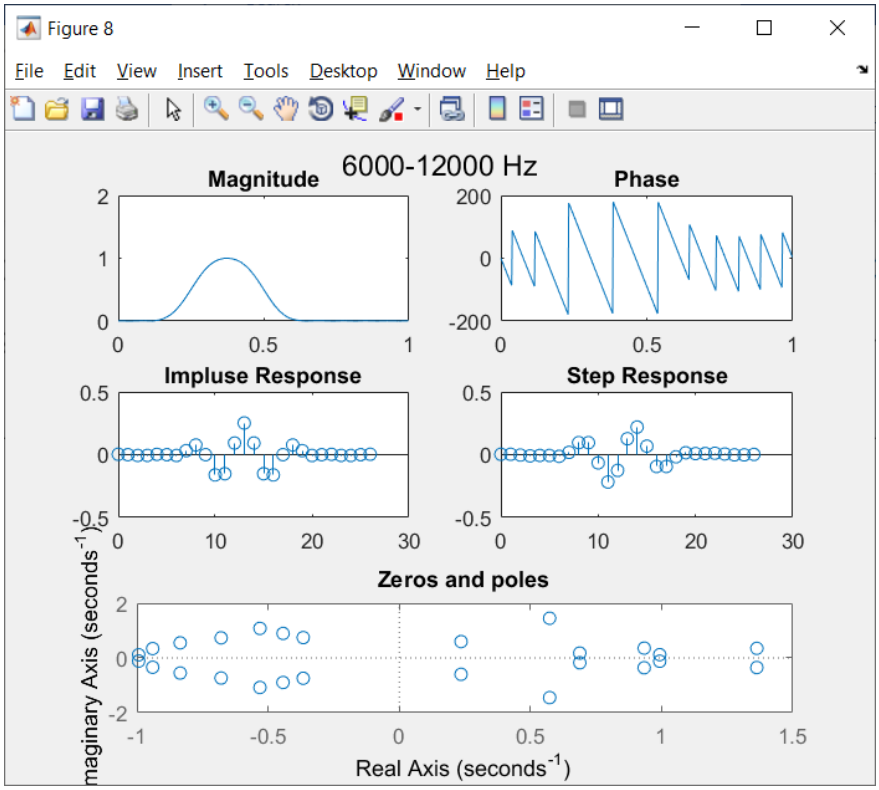
(1000-3000 Hz)



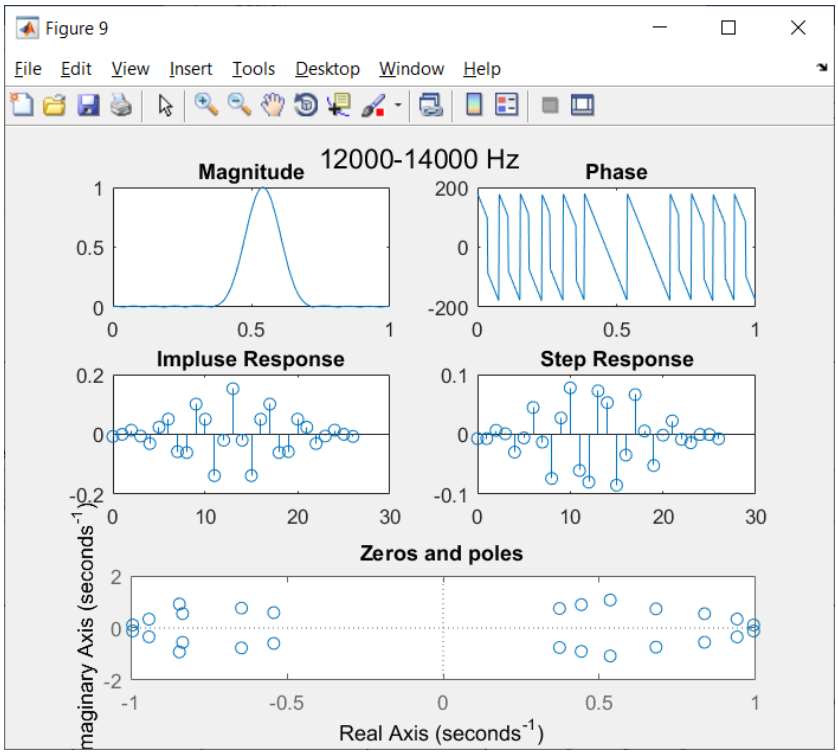
(3000-6000 Hz)



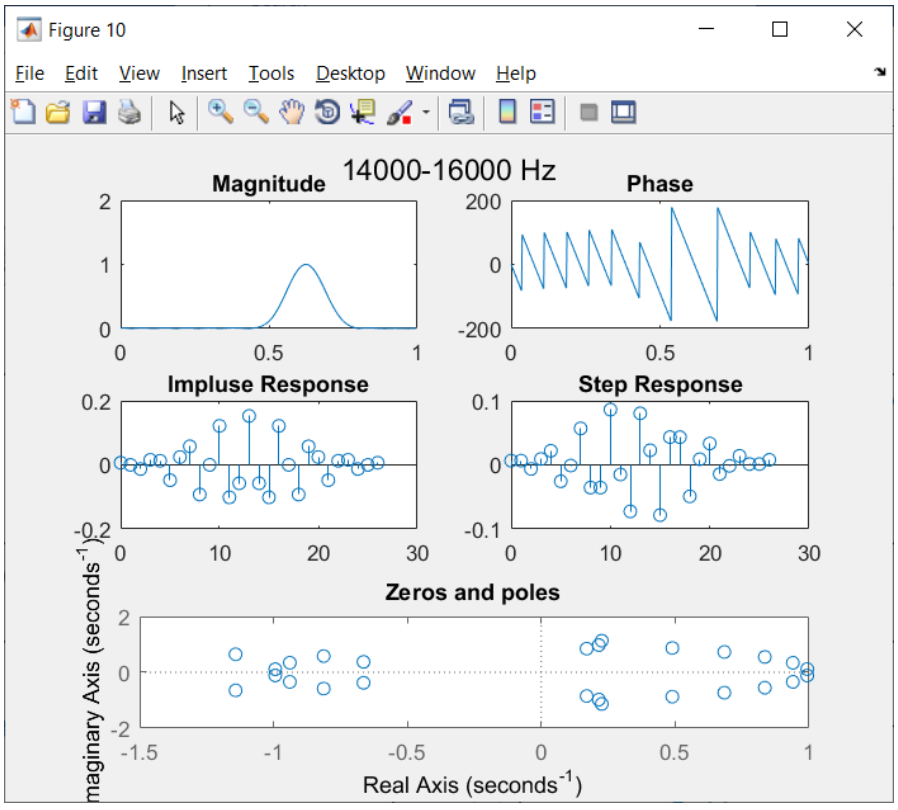
(6000-12000 Hz)



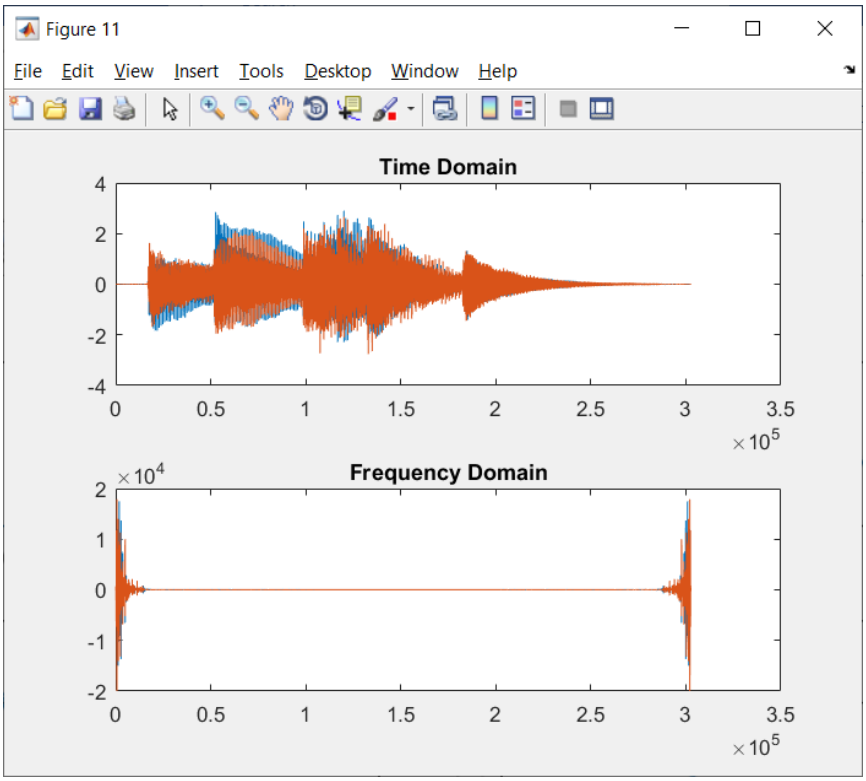
(12000-14000 Hz)



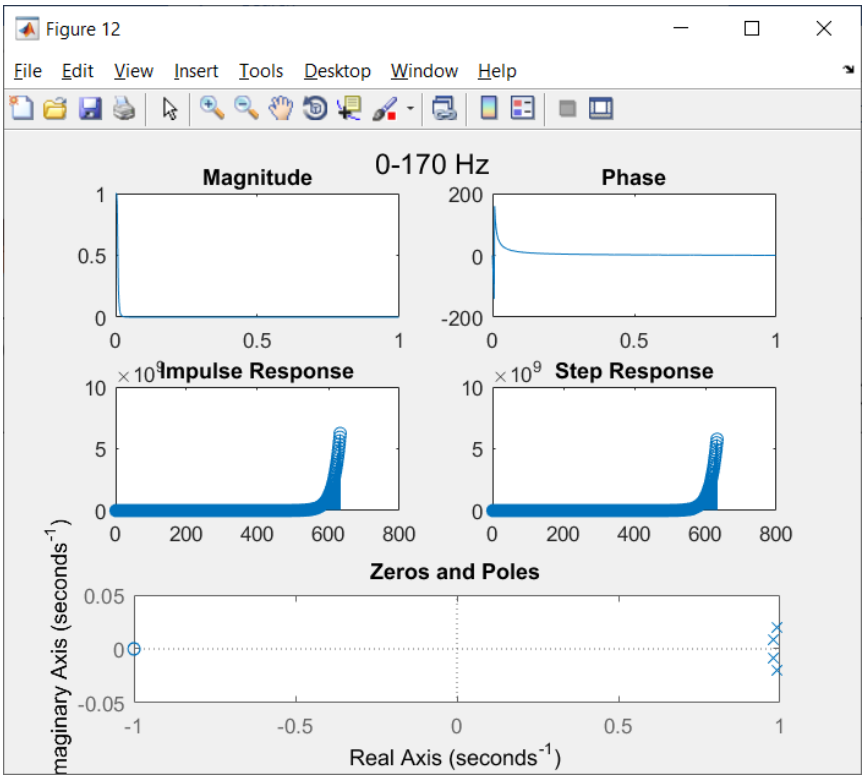
(14000-16000 Hz)



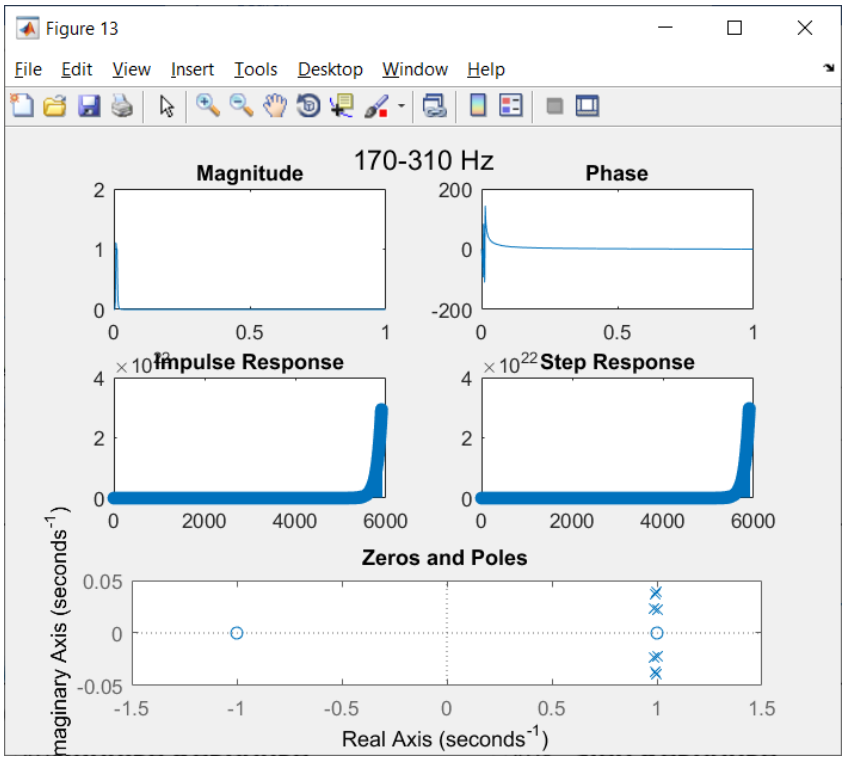
Time Domain & Frequency Domain of composite signal (after FIR filter)



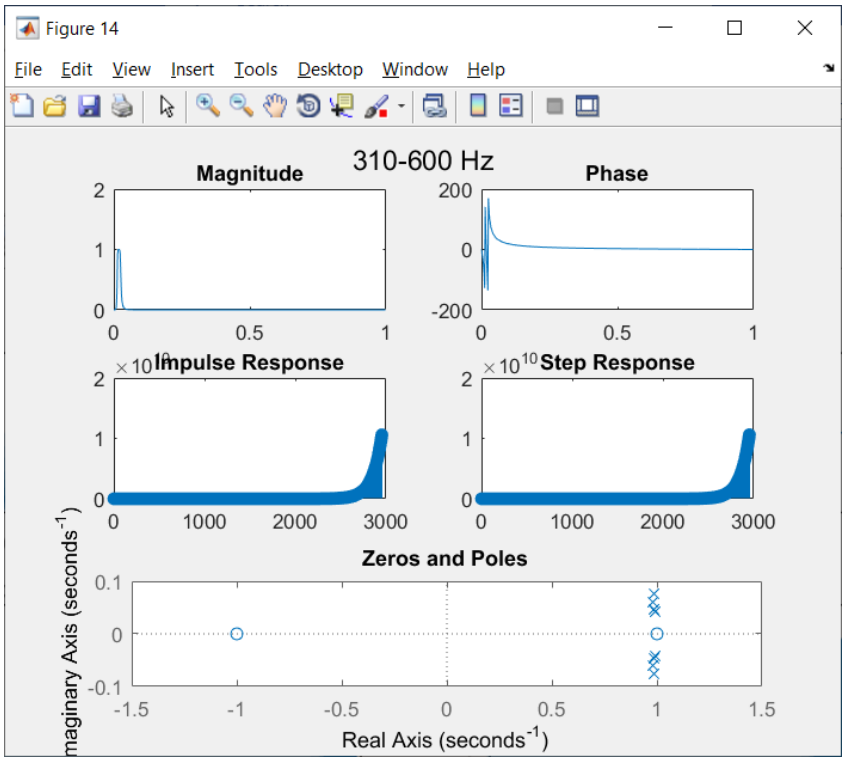
Signal after using IIR filter (0-170 Hz)



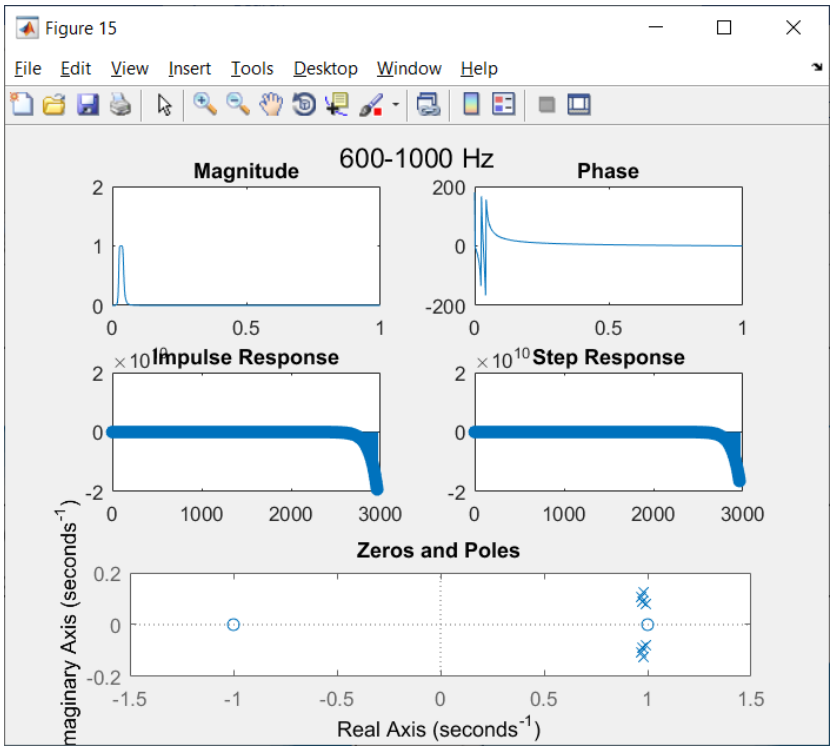
(170-310 Hz)



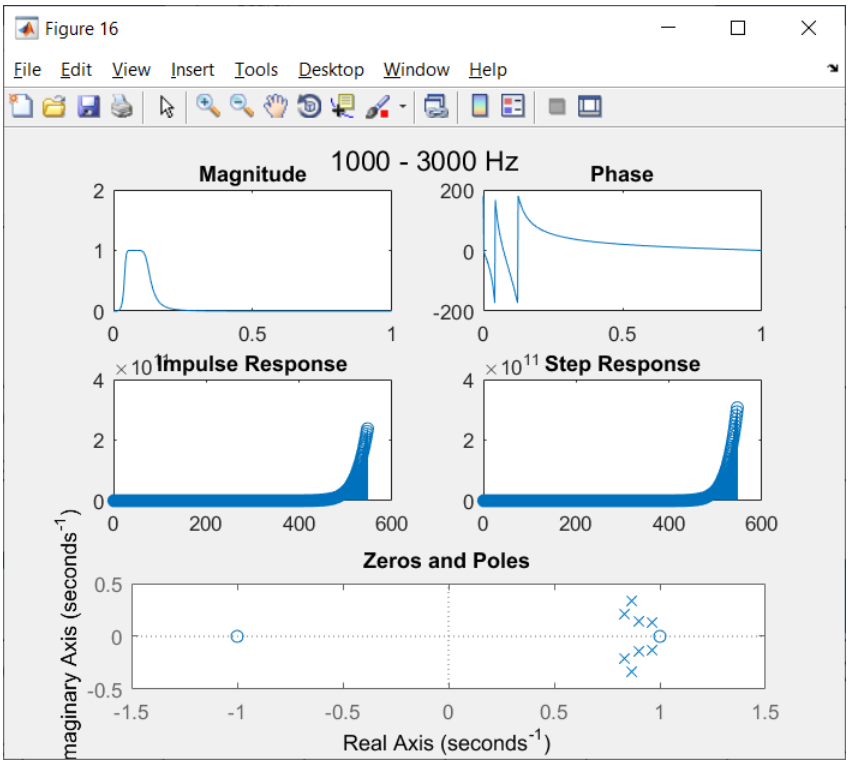
(310-600 Hz)



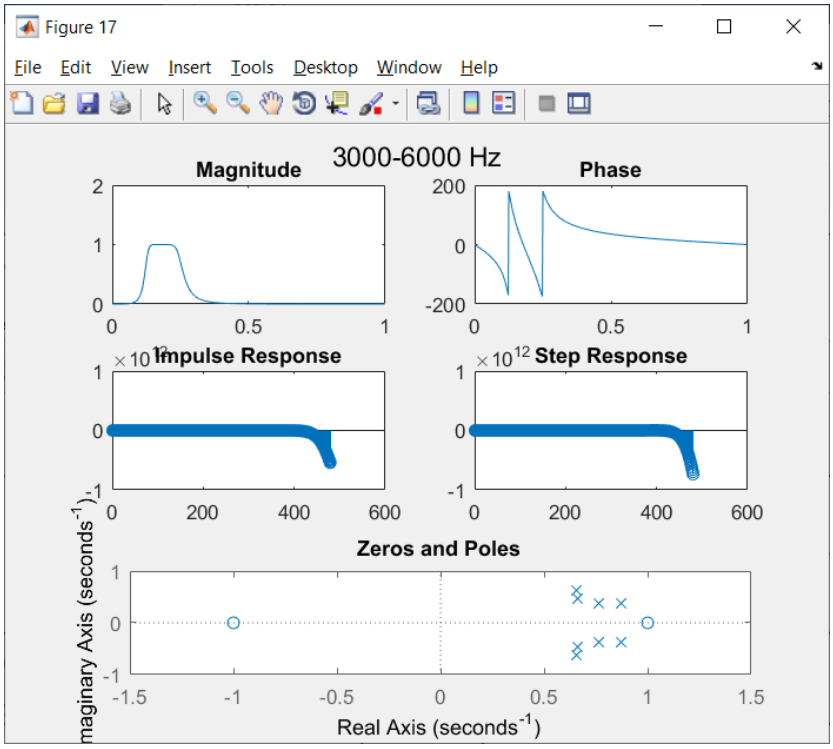
(600-1000 Hz)



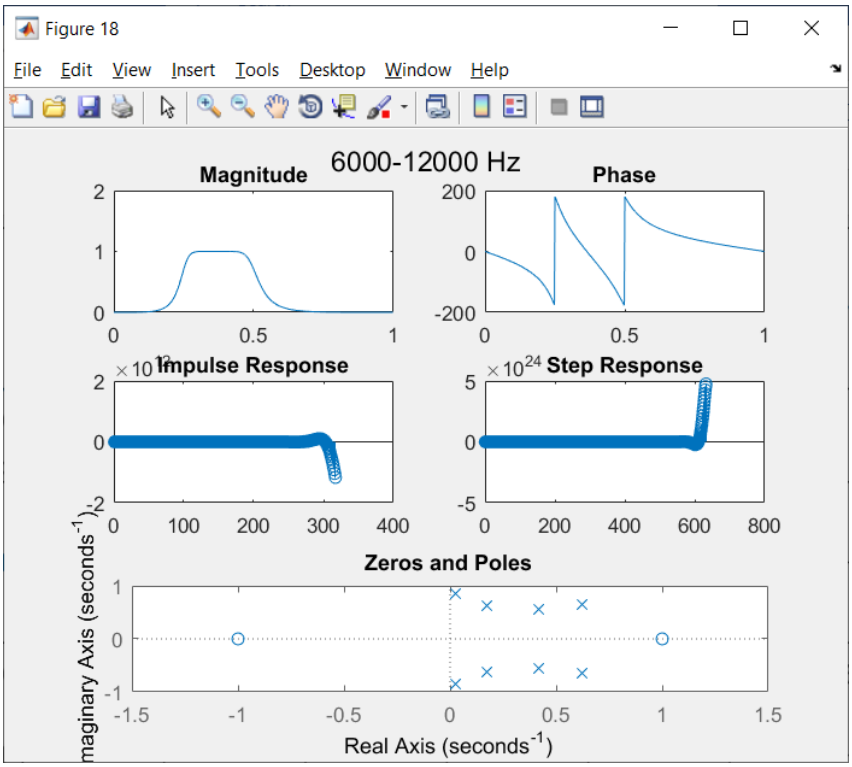
(1000-3000 Hz)



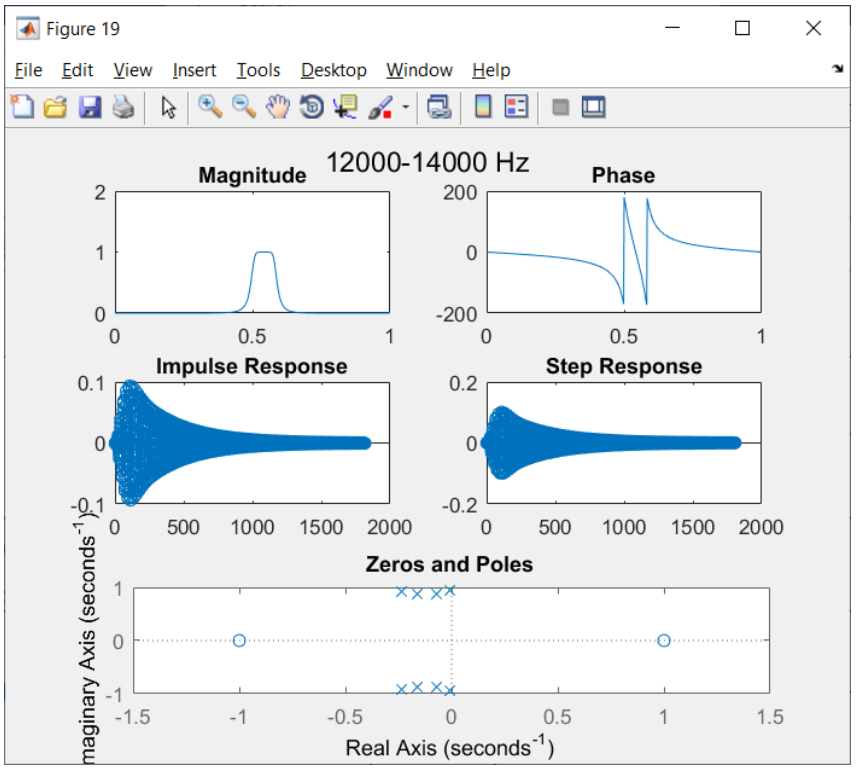
(3000-6000 Hz)



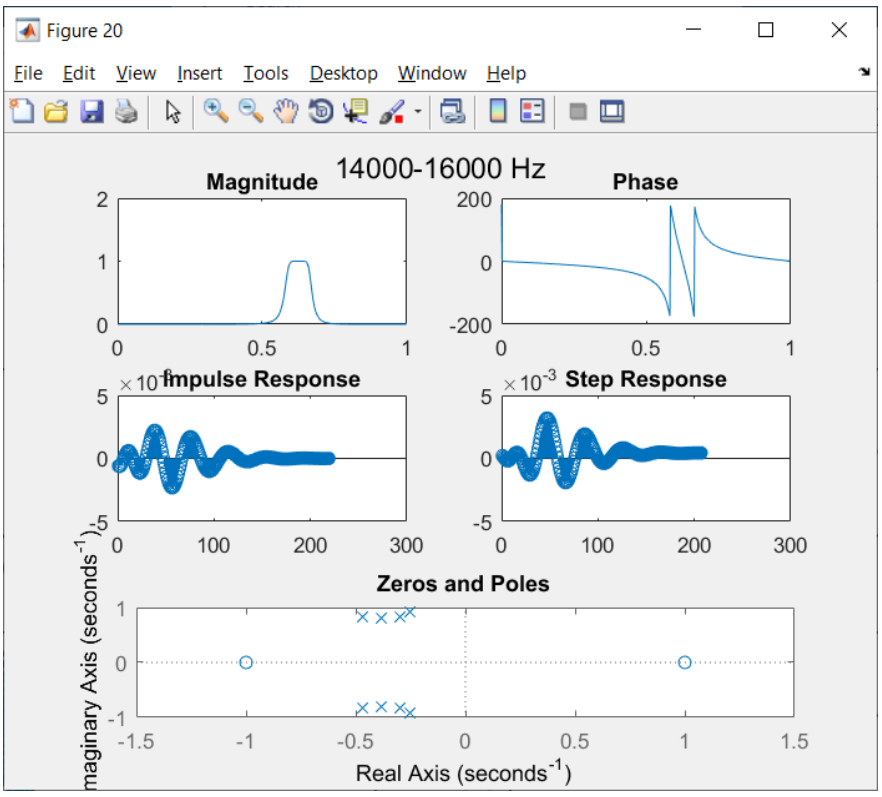
(6000-12000 Hz)



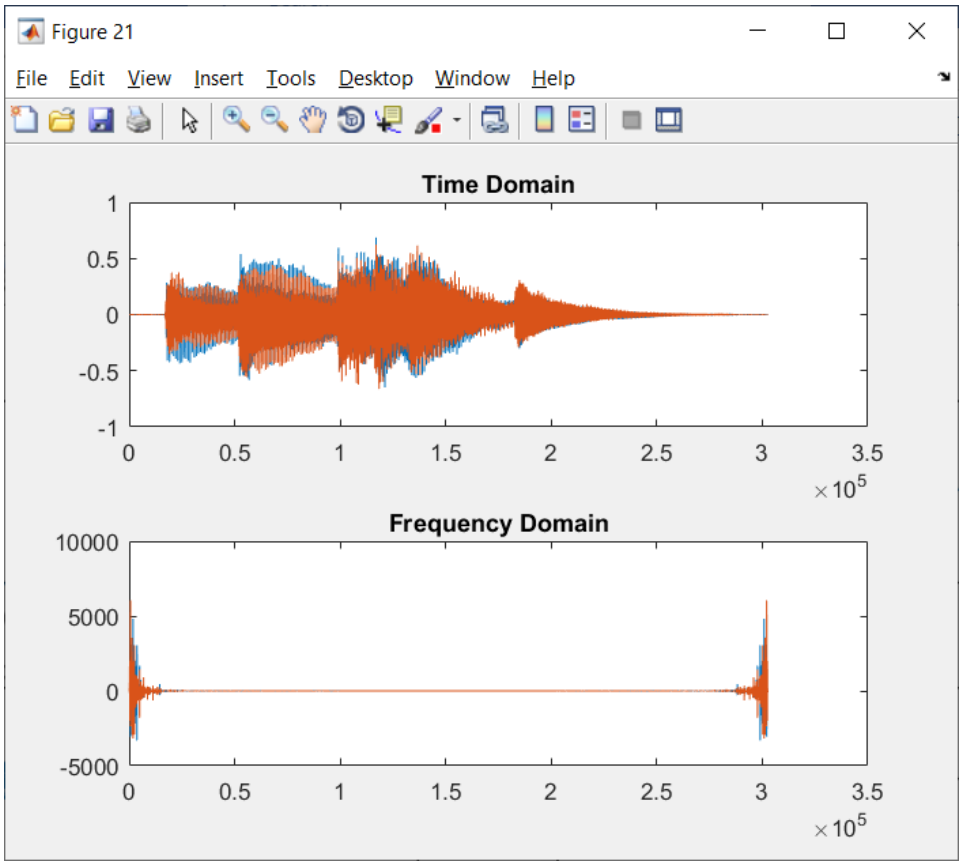
(12000-14000 Hz)



(14000-16000 Hz)

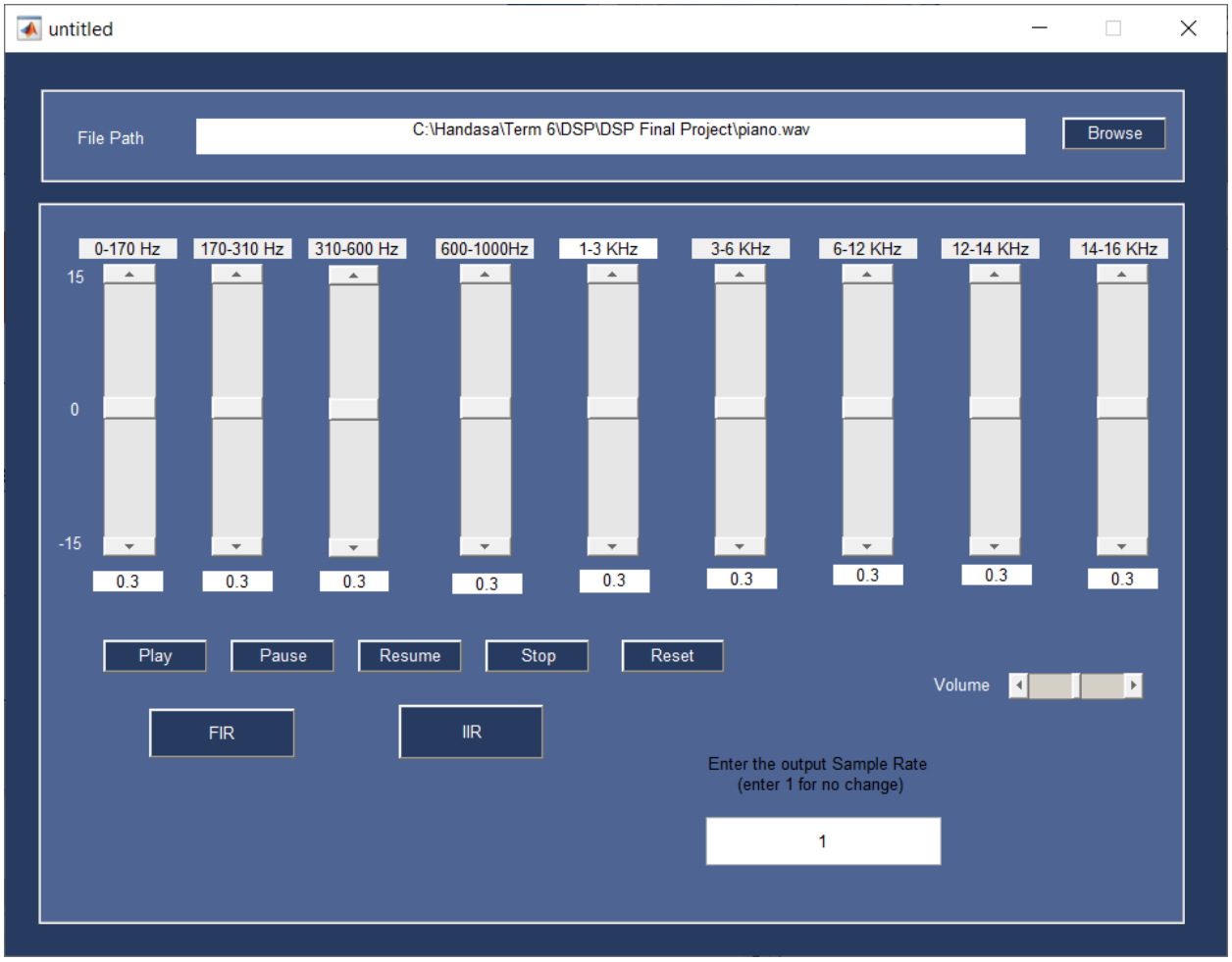


Time Domain & Frequency Domain of composite signal (after IIR filter)

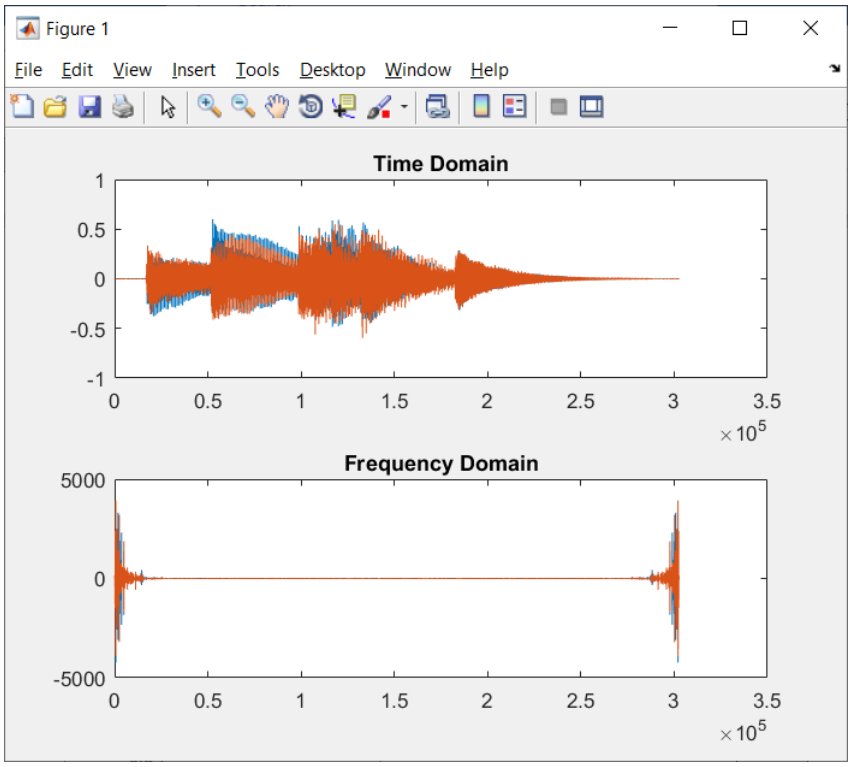


2) IF the output Sample is the Same (gain 0.3)

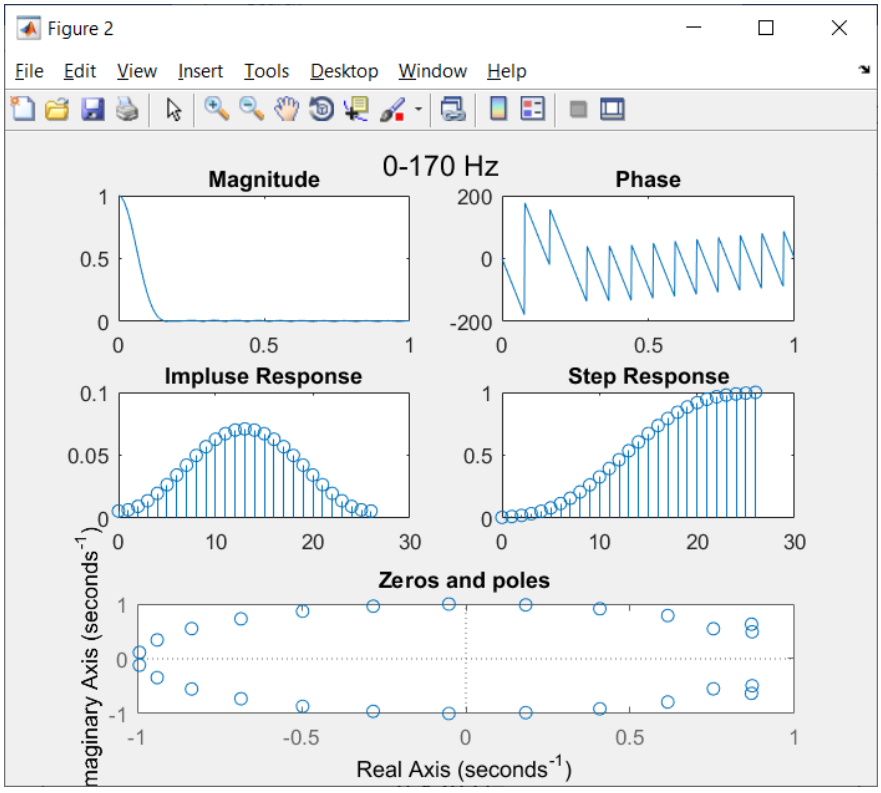
Gui view



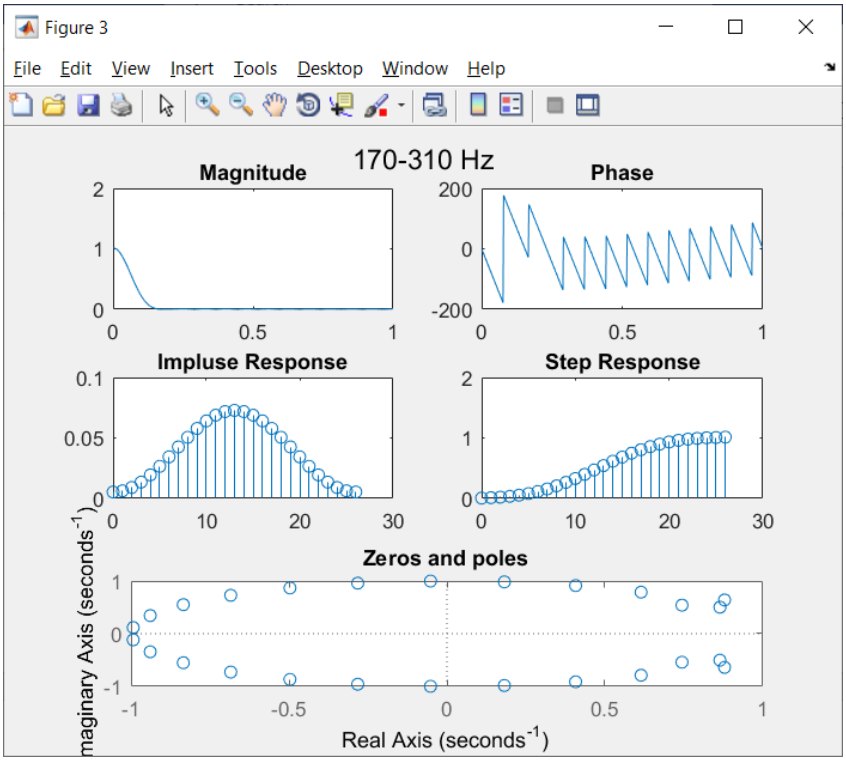
Time Domain & Frequency Domain of original signal



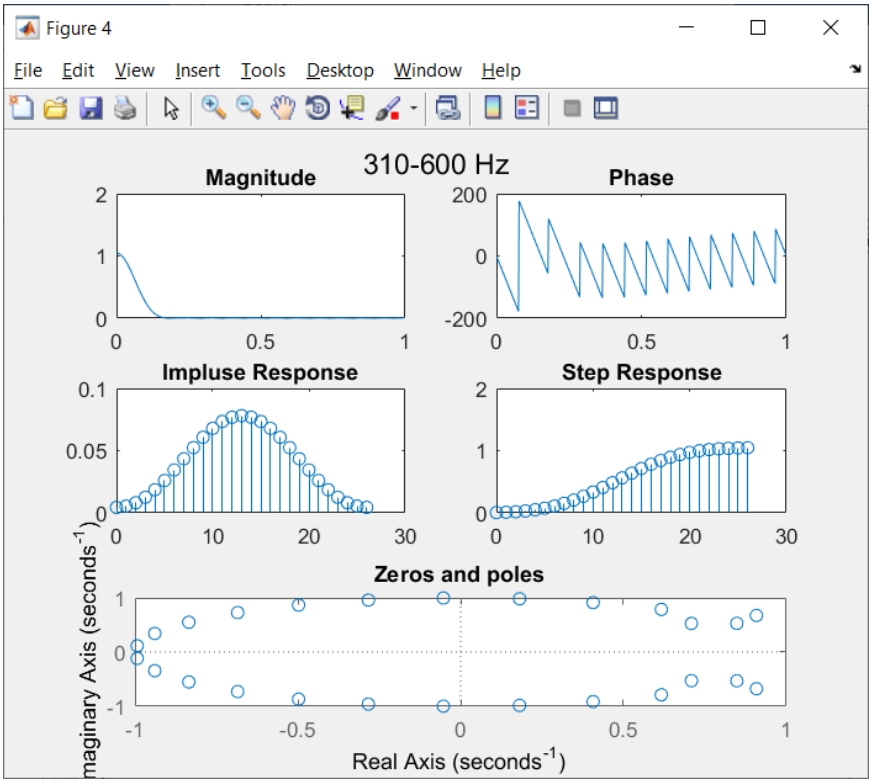
Signal after using FIR filter (0-170 Hz)



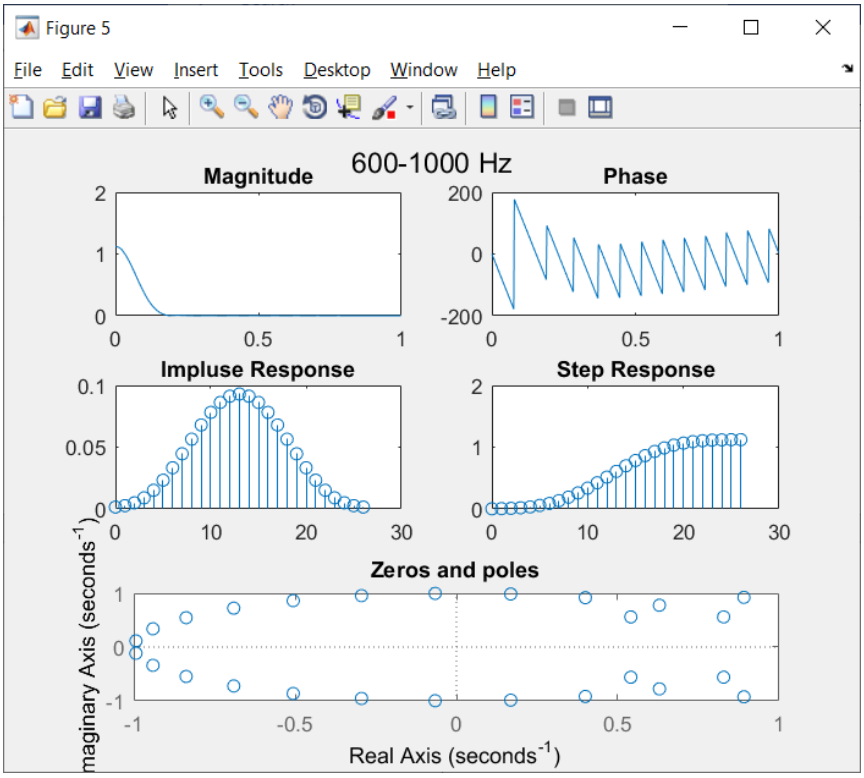
(170-310Hz)



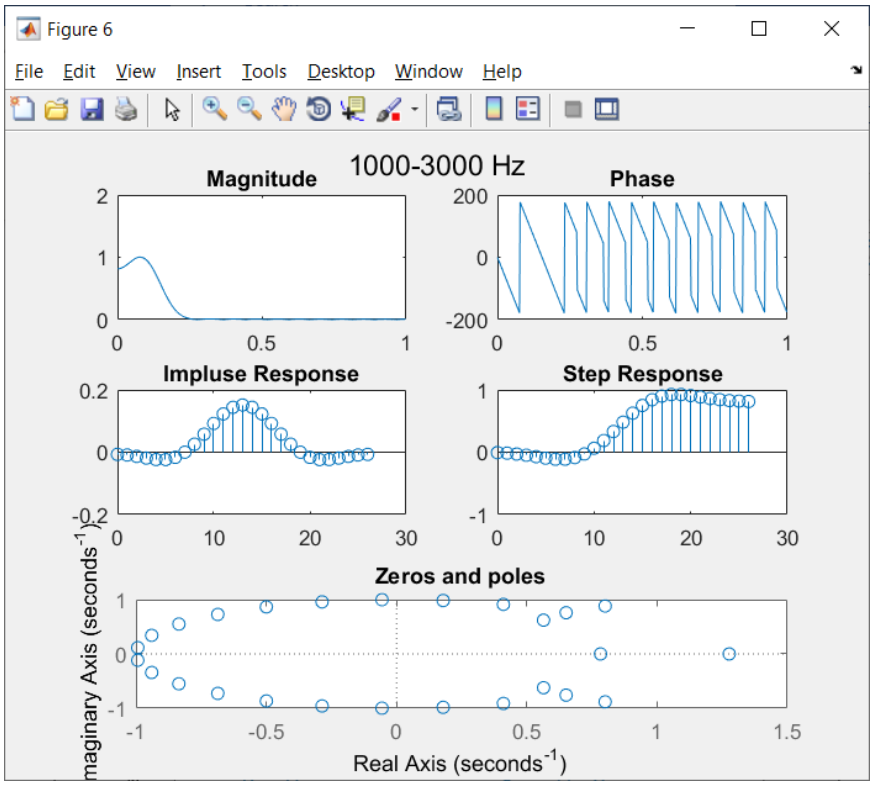
(310-600 Hz)



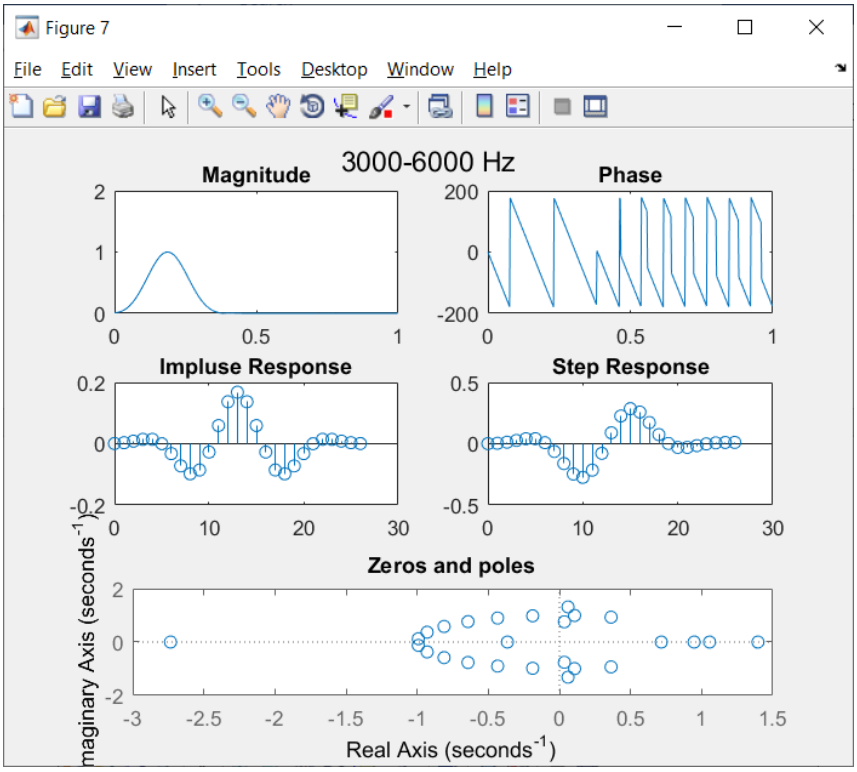
(600-1000 Hz)



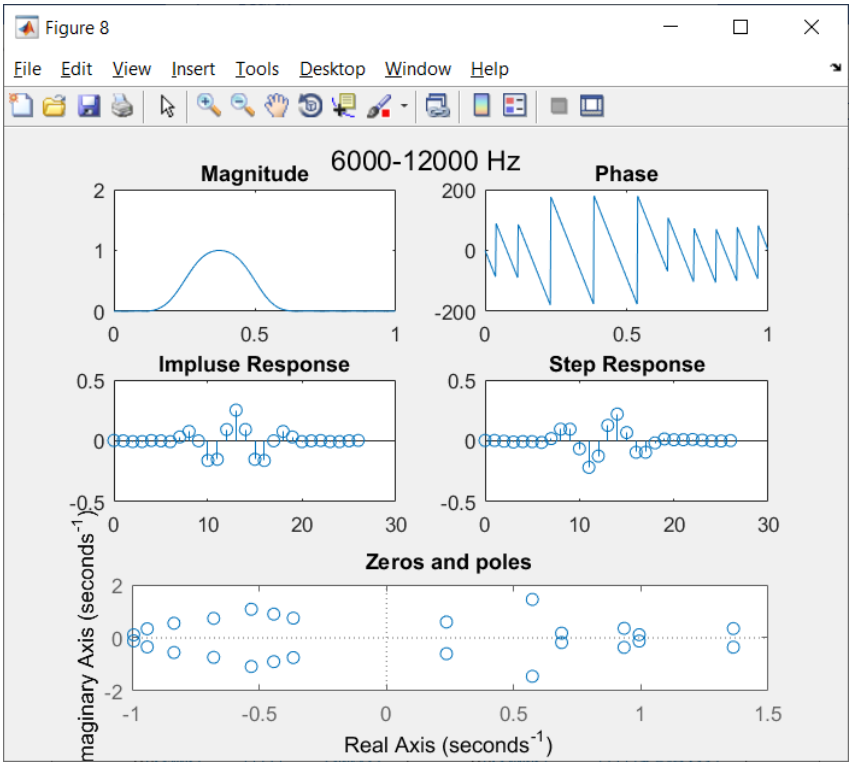
(1000-3000 Hz)



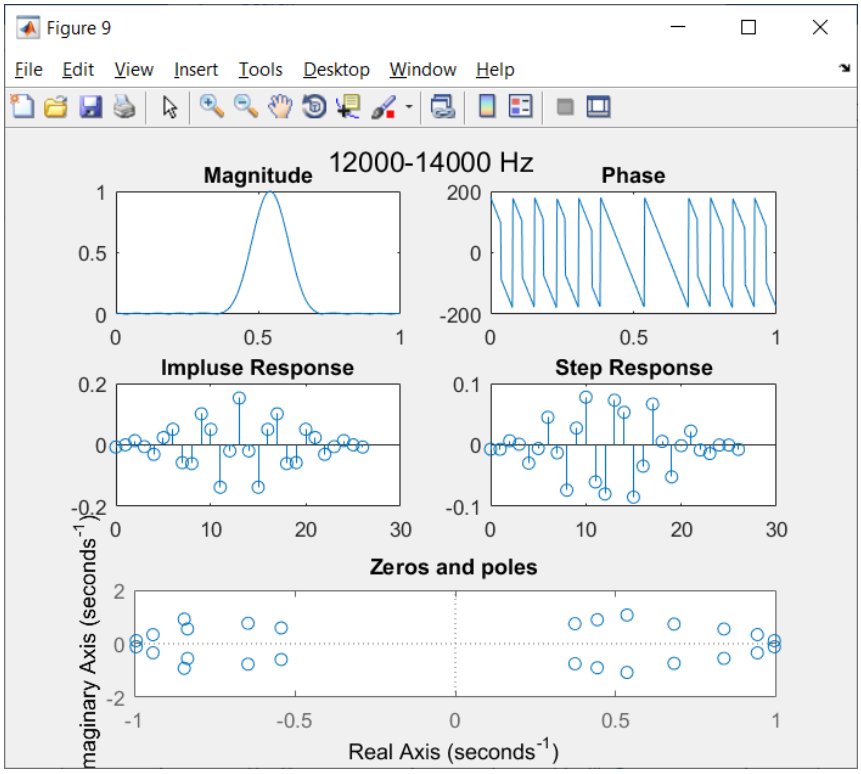
(3000-6000 Hz)



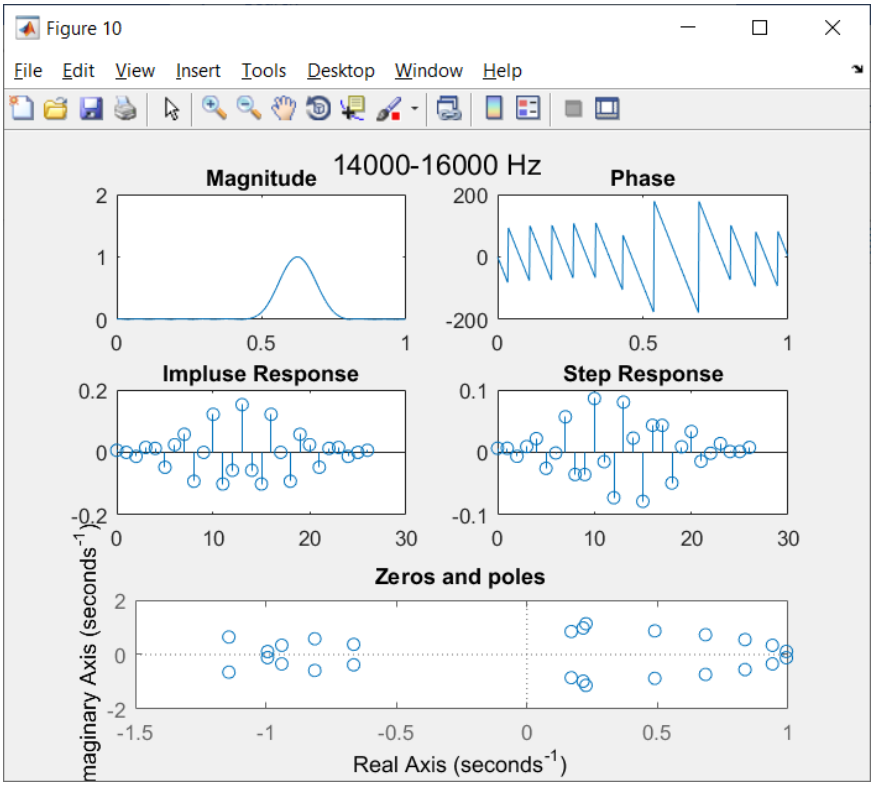
(6000-12000 Hz)



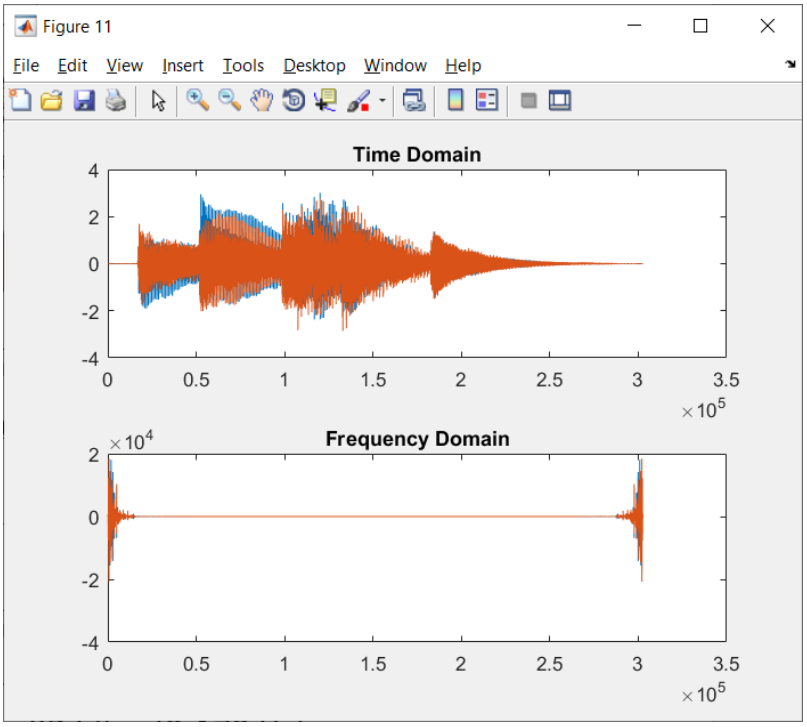
(12000-14000 Hz)



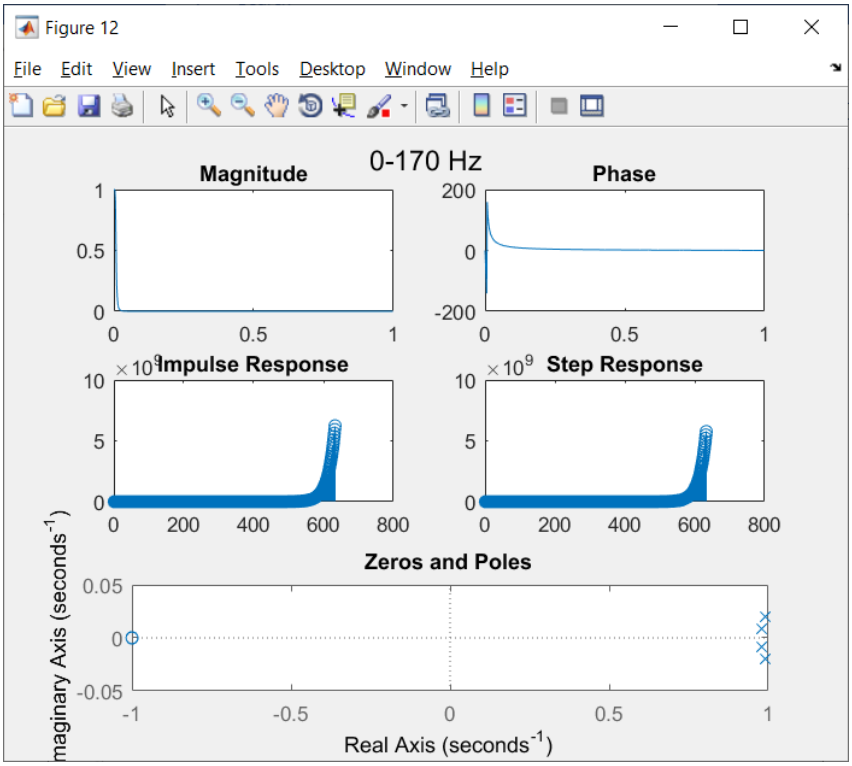
(14000-16000 Hz)



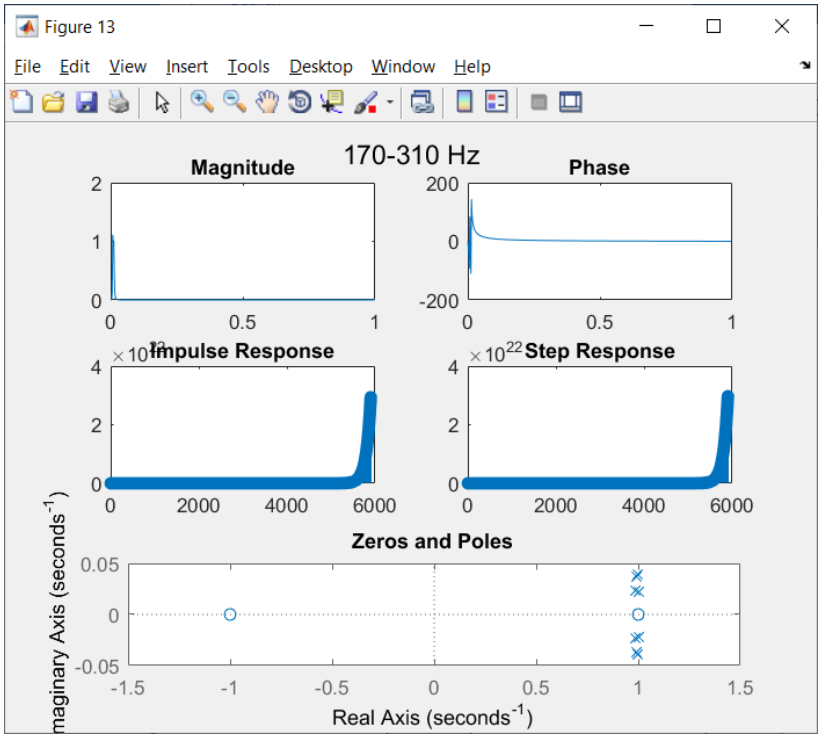
Time Domain & Frequency Domain of composite signal (after FIR filter)



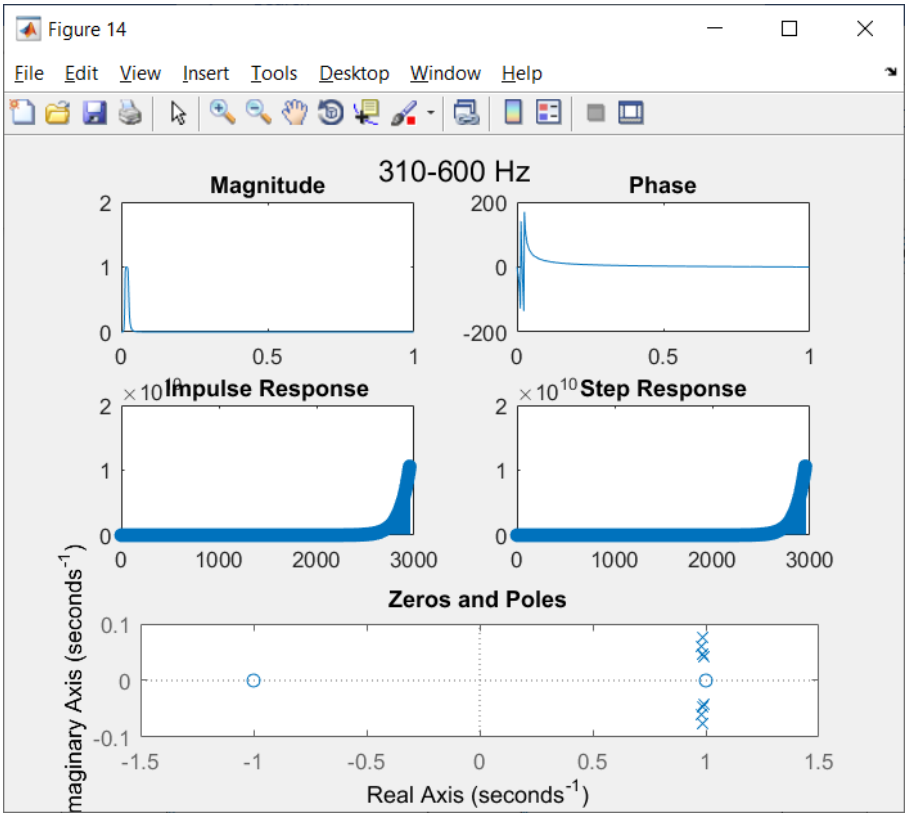
Signal after using IIR filter (0-170 Hz)



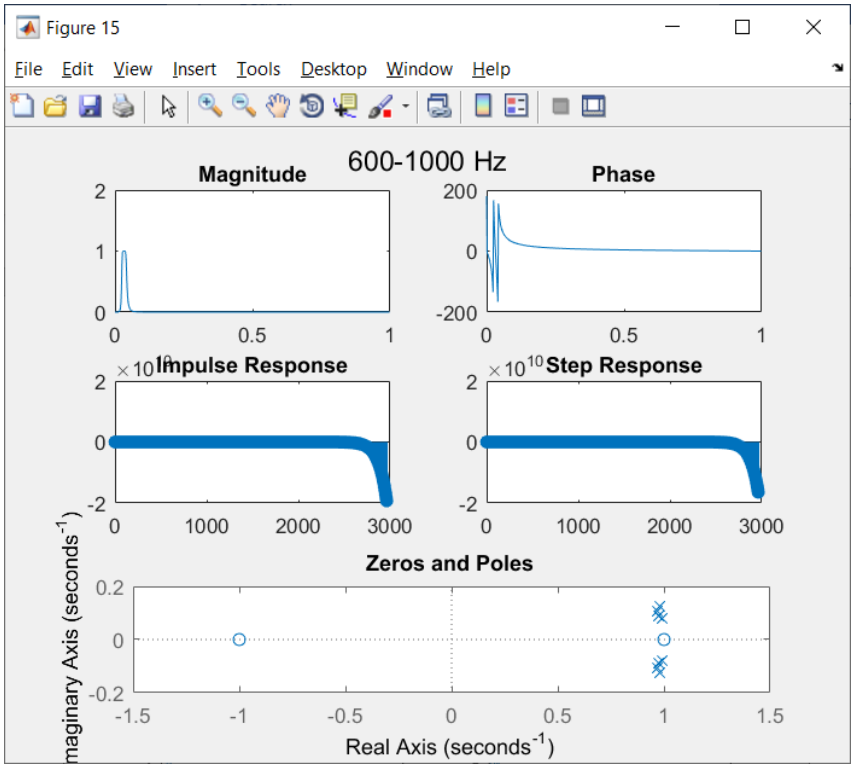
(170-310 Hz)



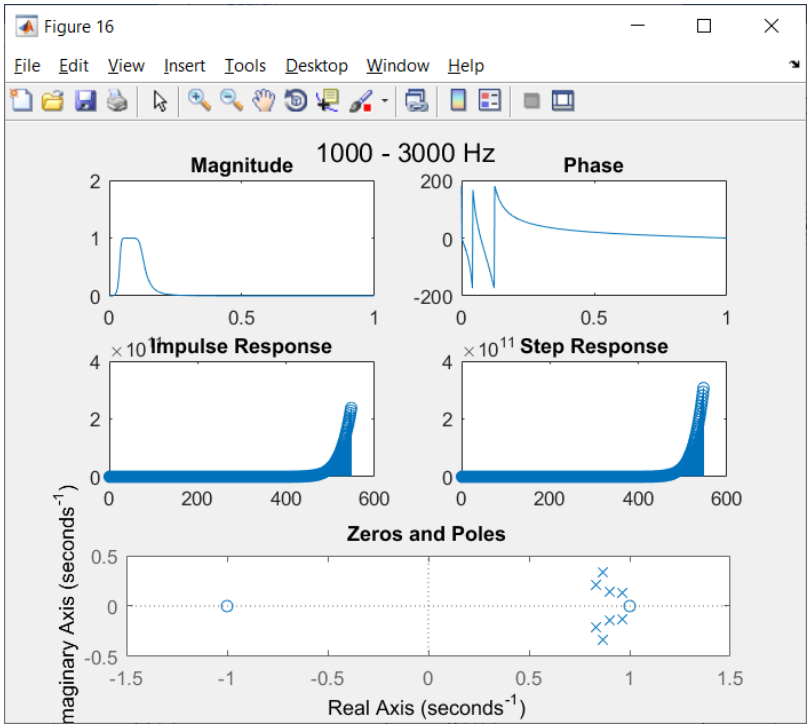
(310-600 Hz)



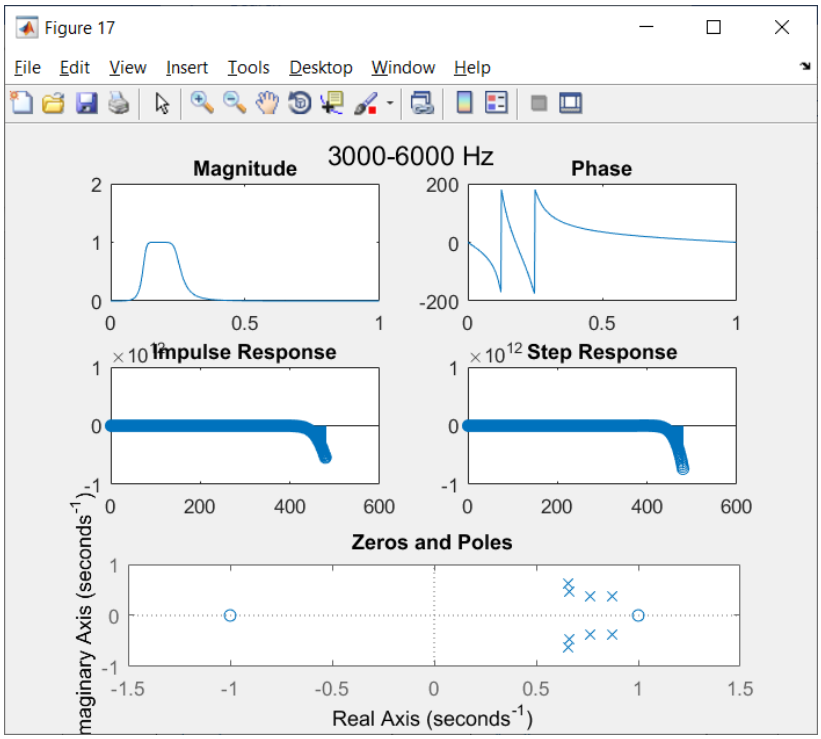
(600-1000 Hz)



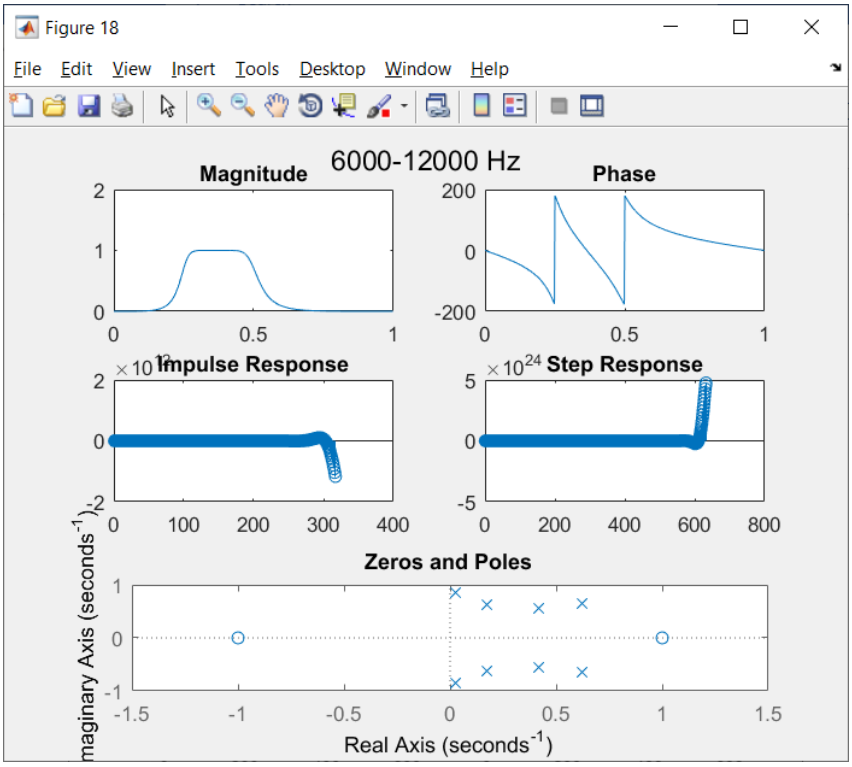
(1000-3000 Hz)



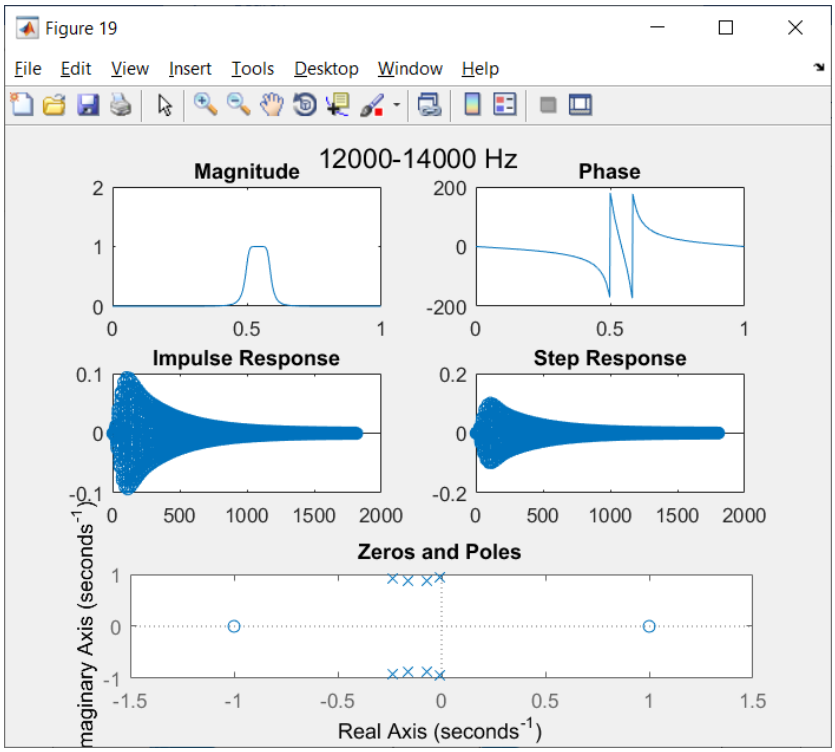
(3000-6000 Hz)



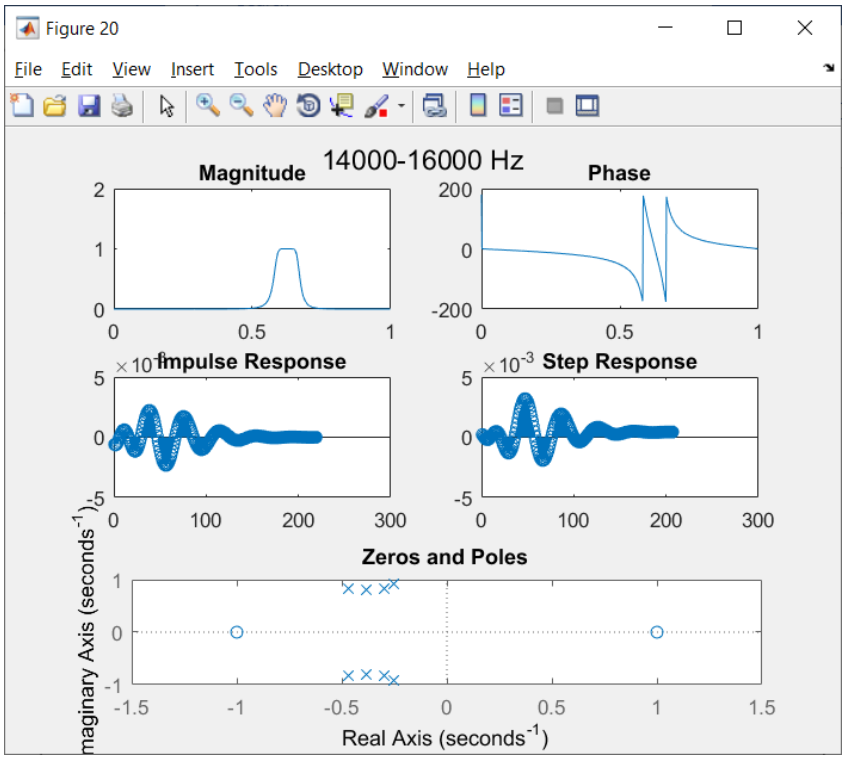
(6000-12000 Hz)



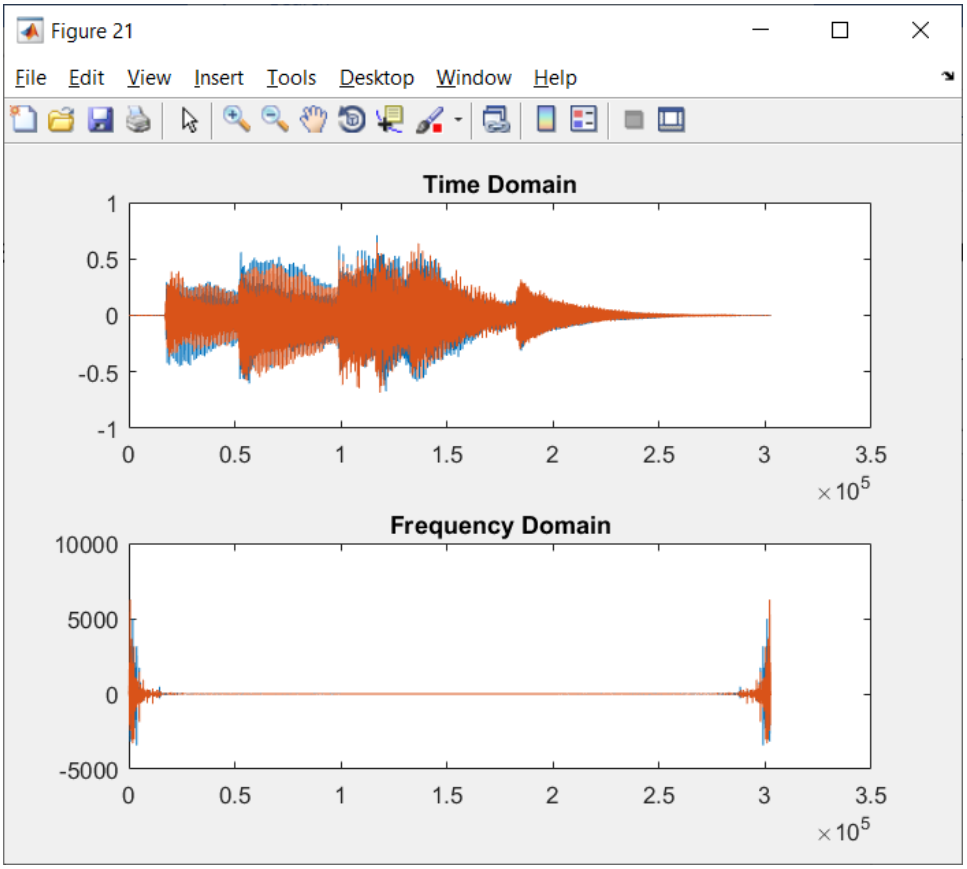
(12000-14000 Hz)



(14000-16000 Hz)



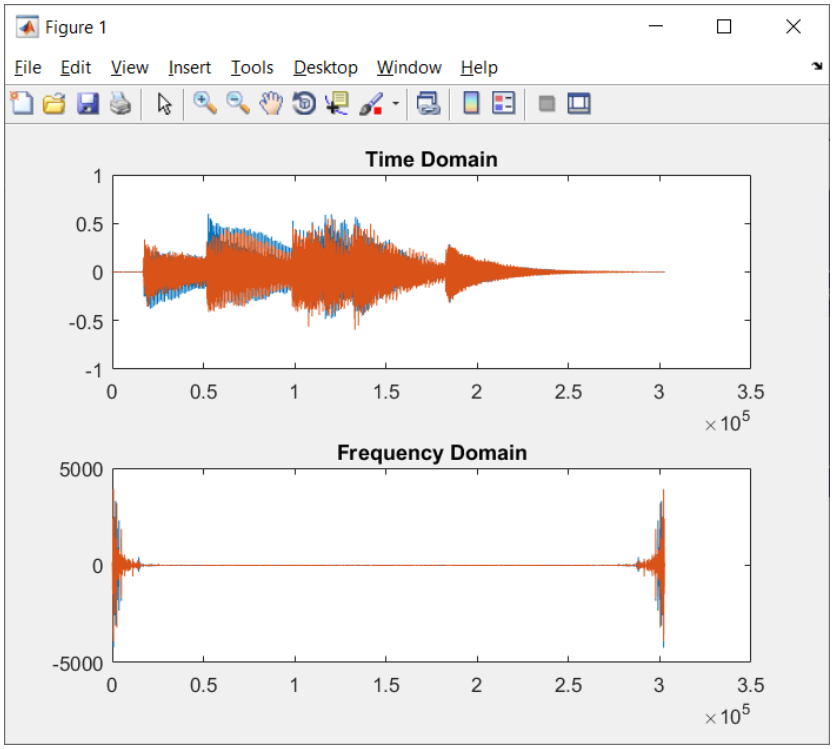
Time Domain & Frequency Domain of composite signal (after IIR filter)



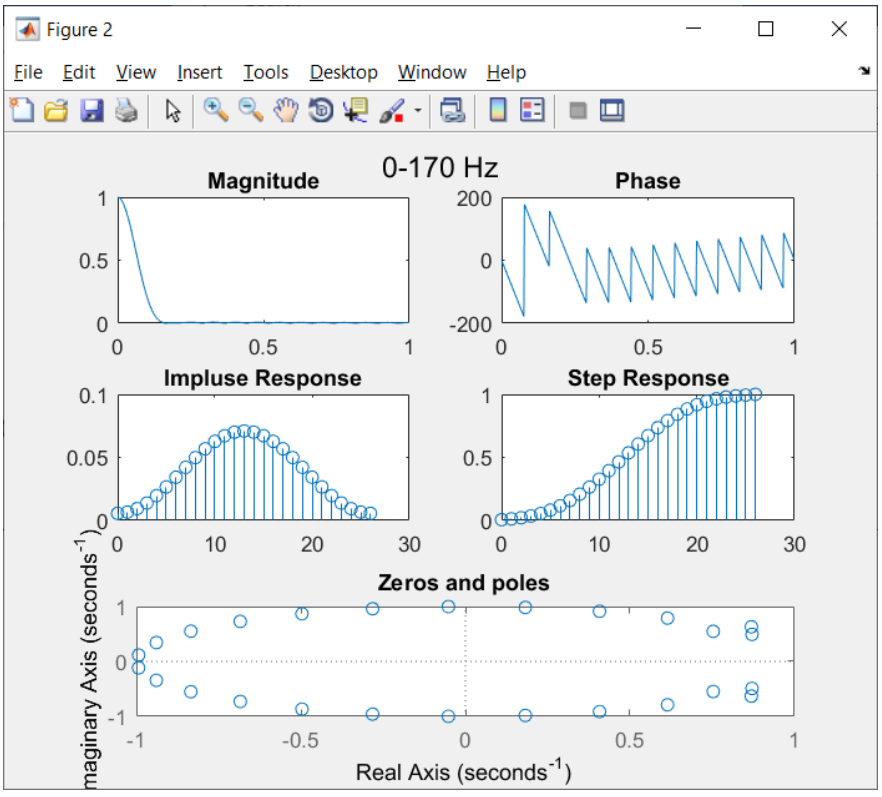
3) IF the output Sample is Doubled

Gui view

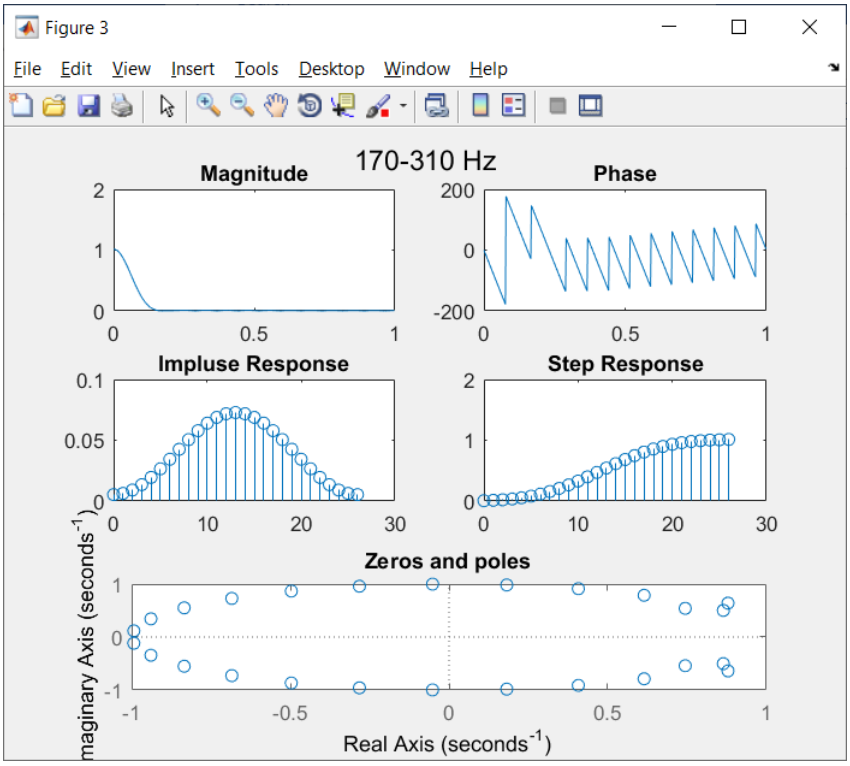
Time Domain & Frequency Domain of original signal



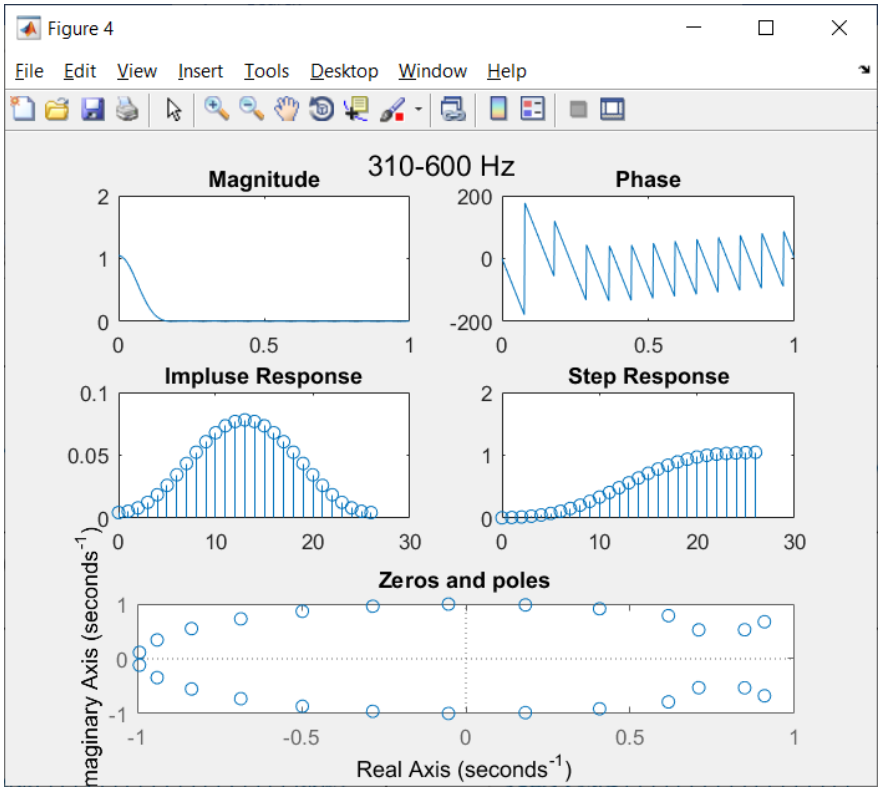
Signal after using FIR filter (0-170 Hz)



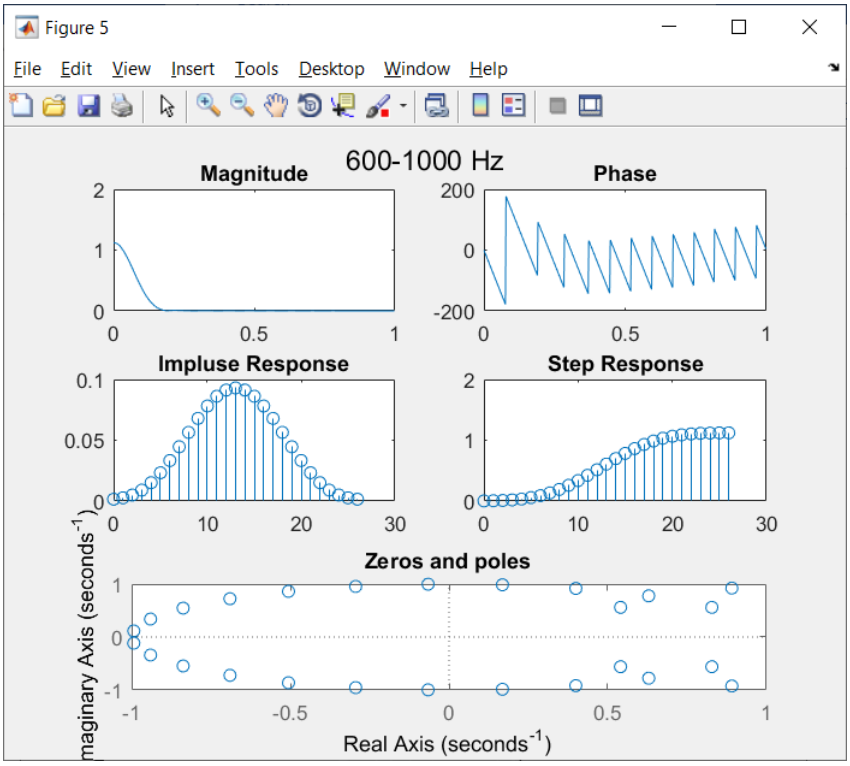
(170-310Hz)



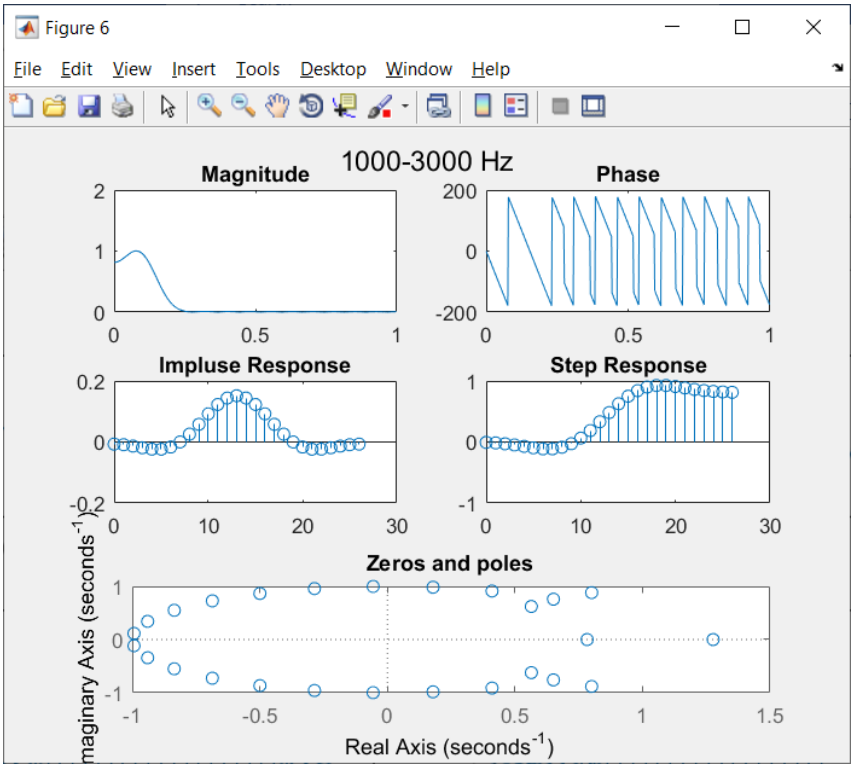
(310-600 Hz)



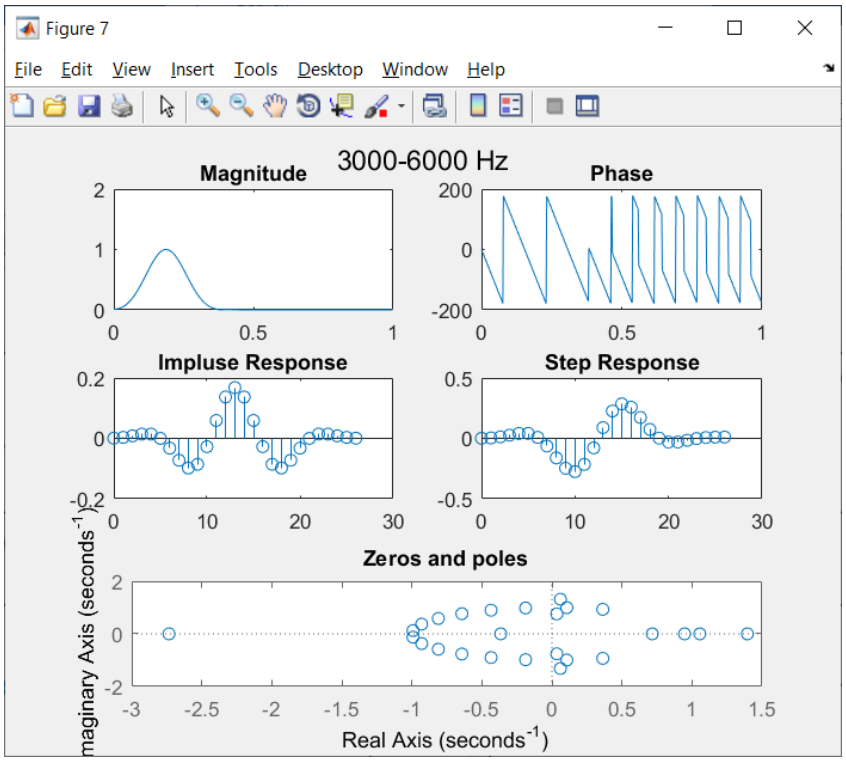
(600-1000 Hz)



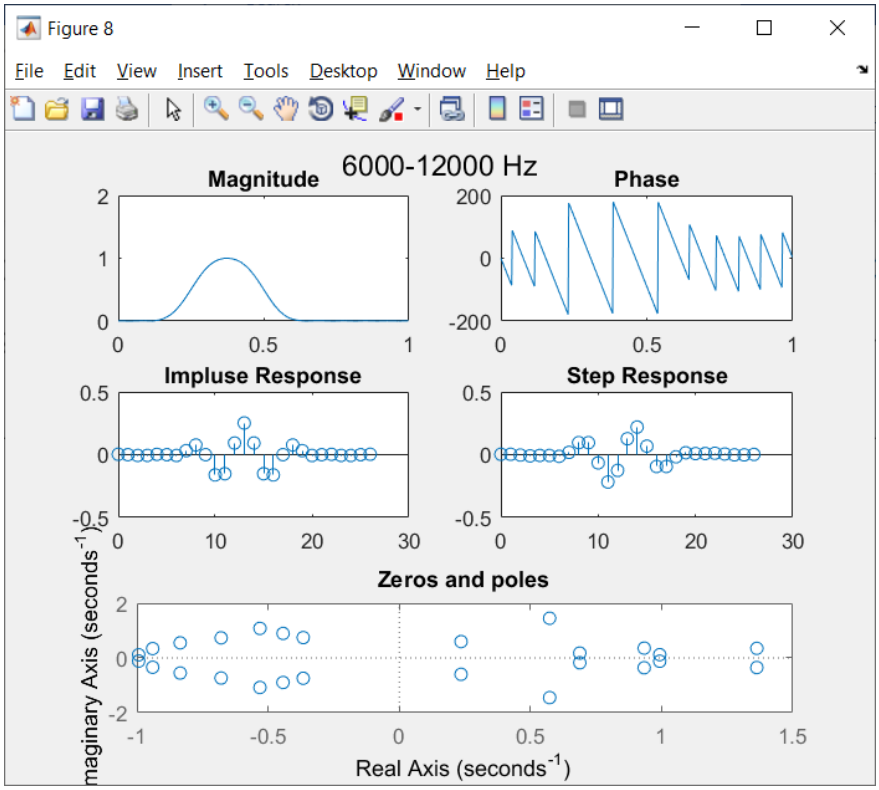
(1000-3000 Hz)



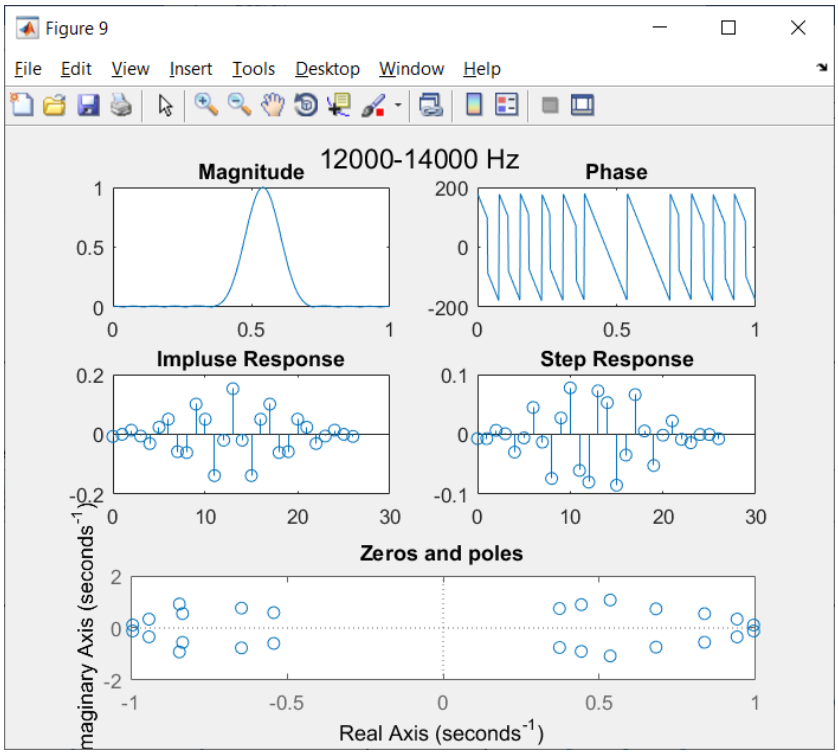
(3000-6000 Hz)



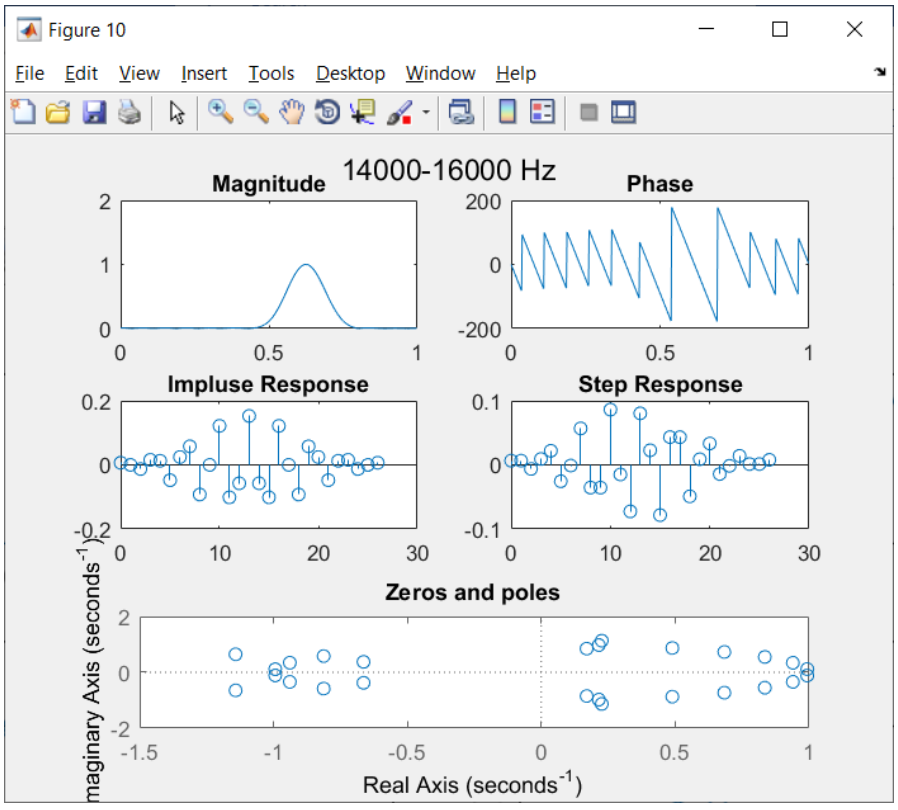
(6000-12000 Hz)



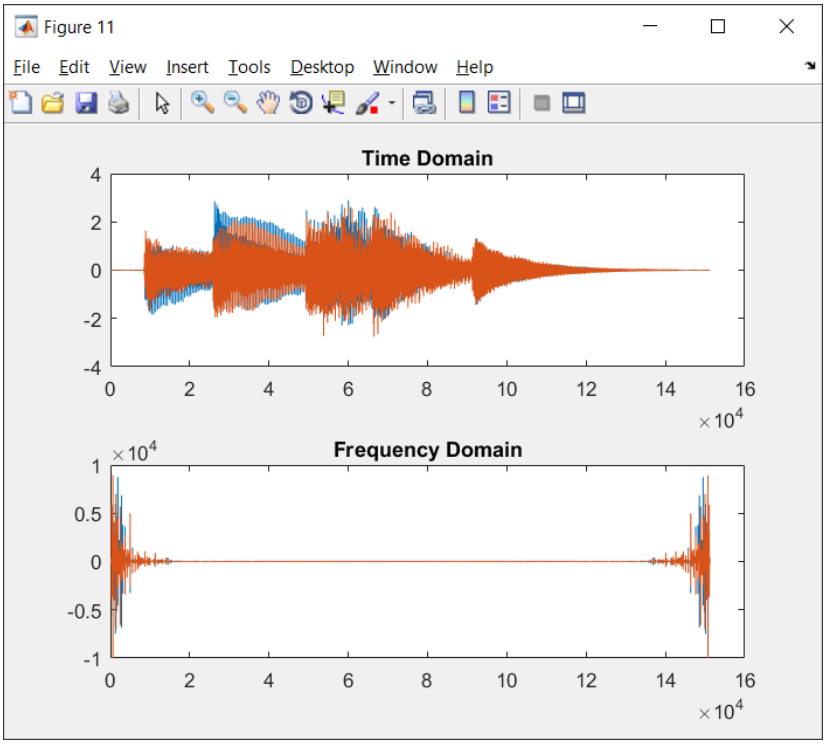
(12000-14000 Hz)



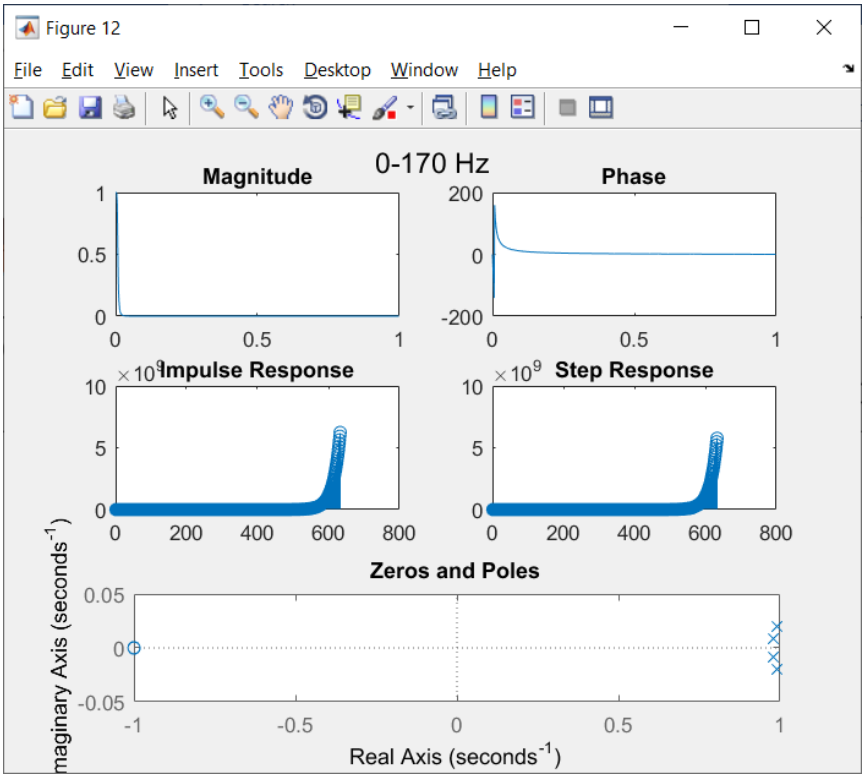
(14000-16000 Hz)



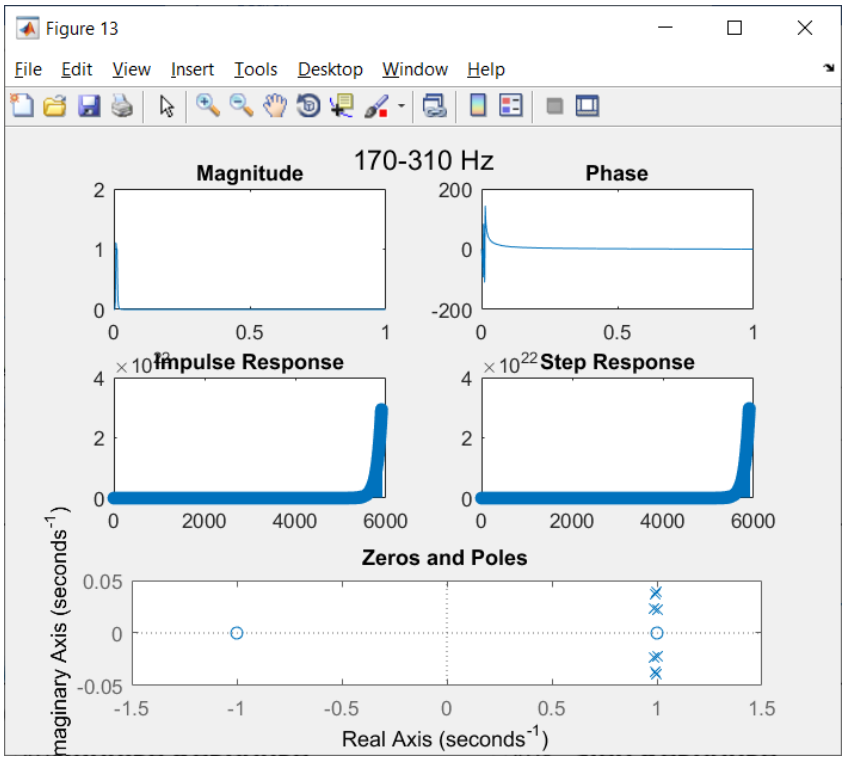
Time Domain & Frequency Domain of composite signal (after FIR filter)



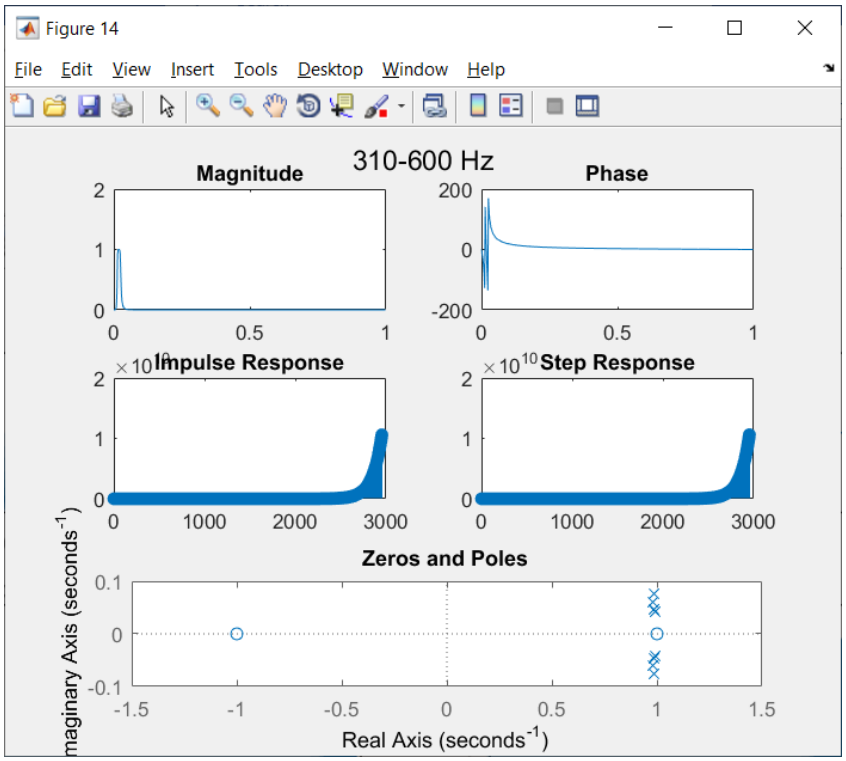
Signal after using IIR filter (0-170 Hz)



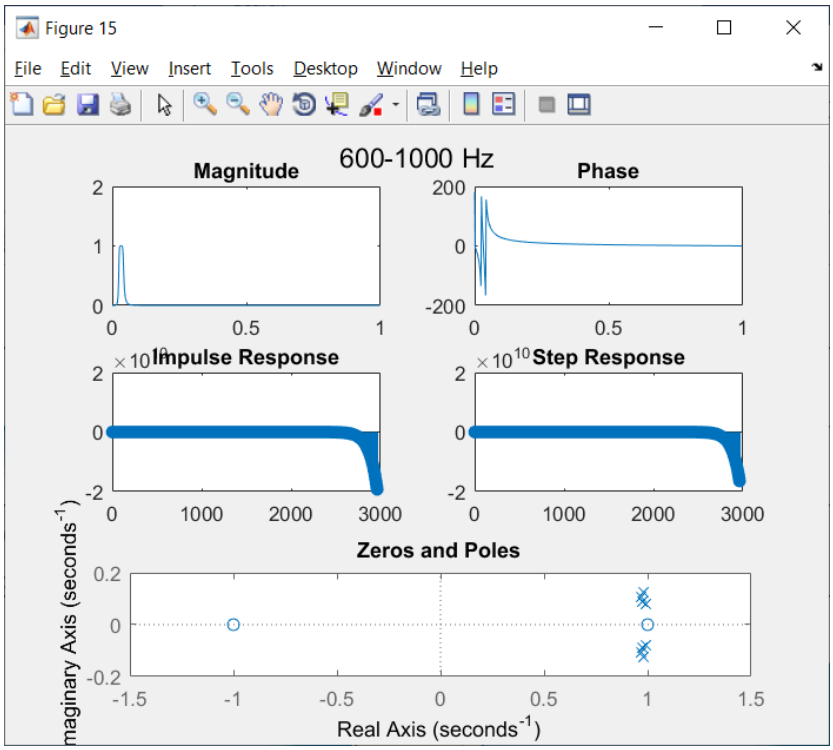
(170-310 Hz)



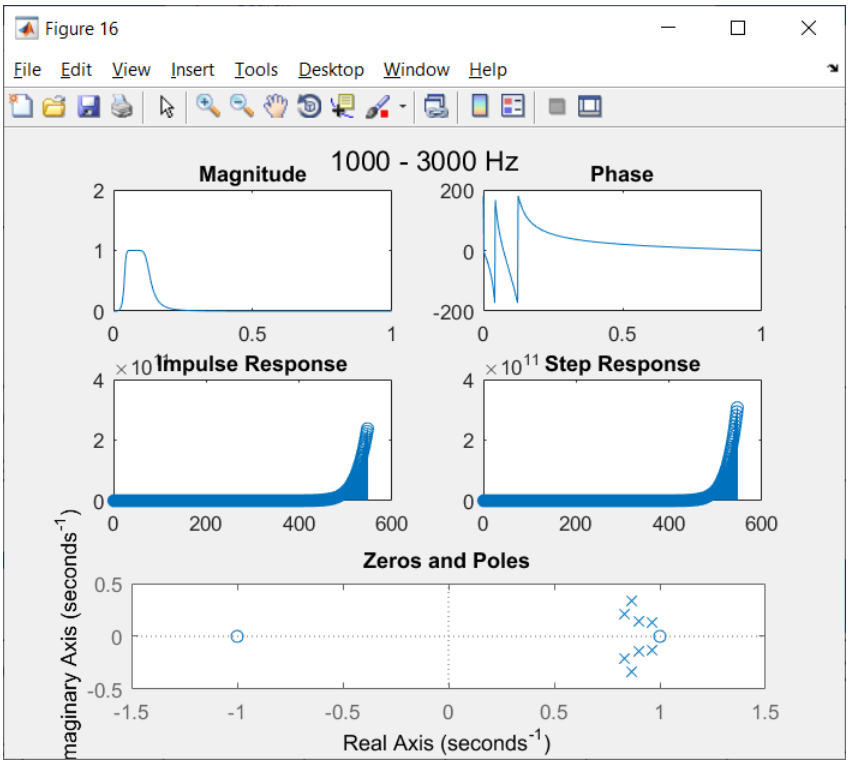
(310-600 Hz)



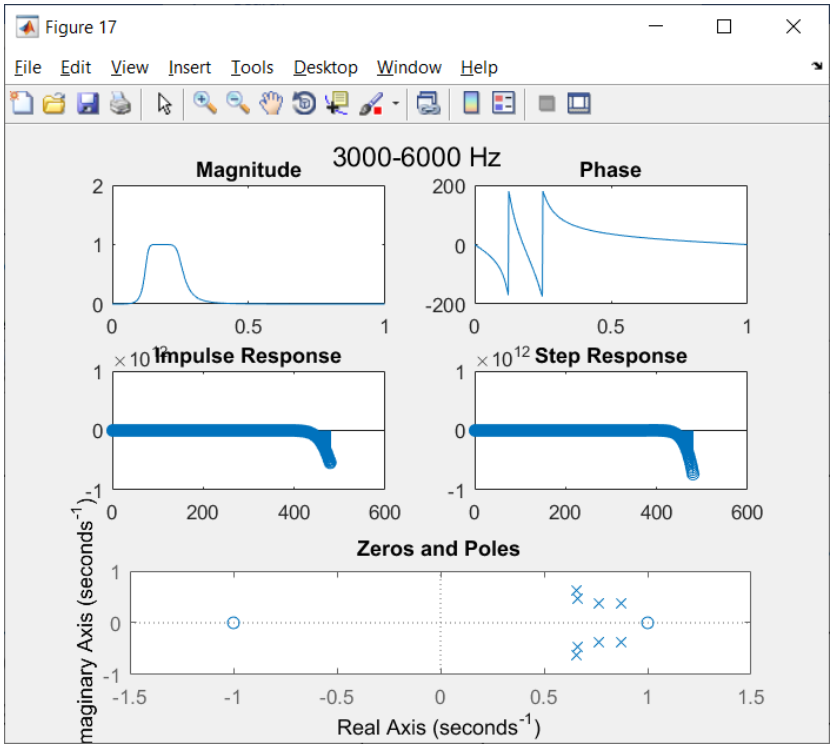
(600-1000 Hz)



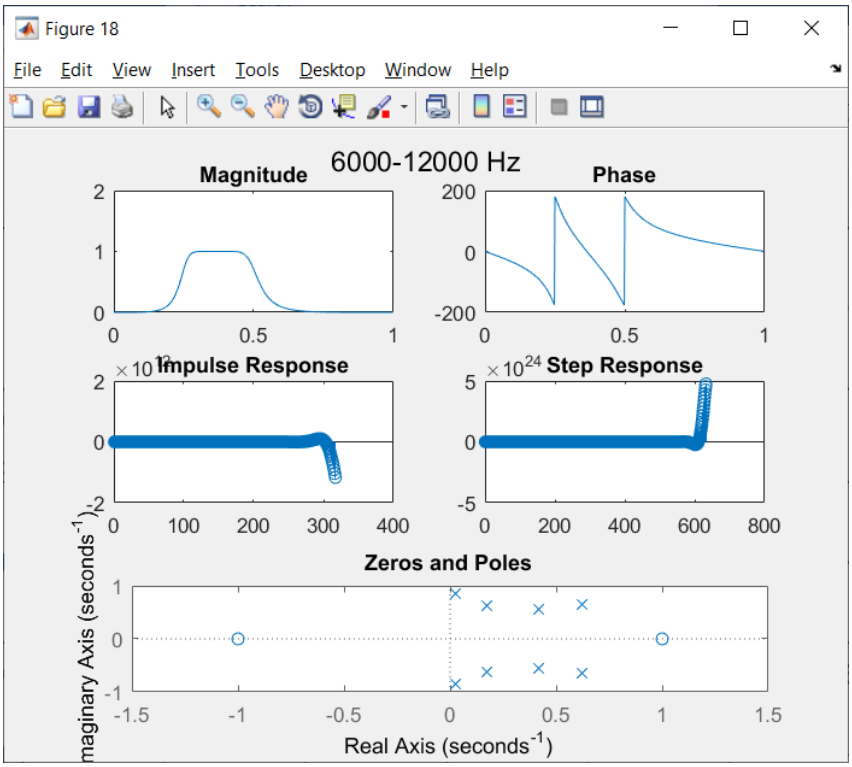
(1000-3000 Hz)



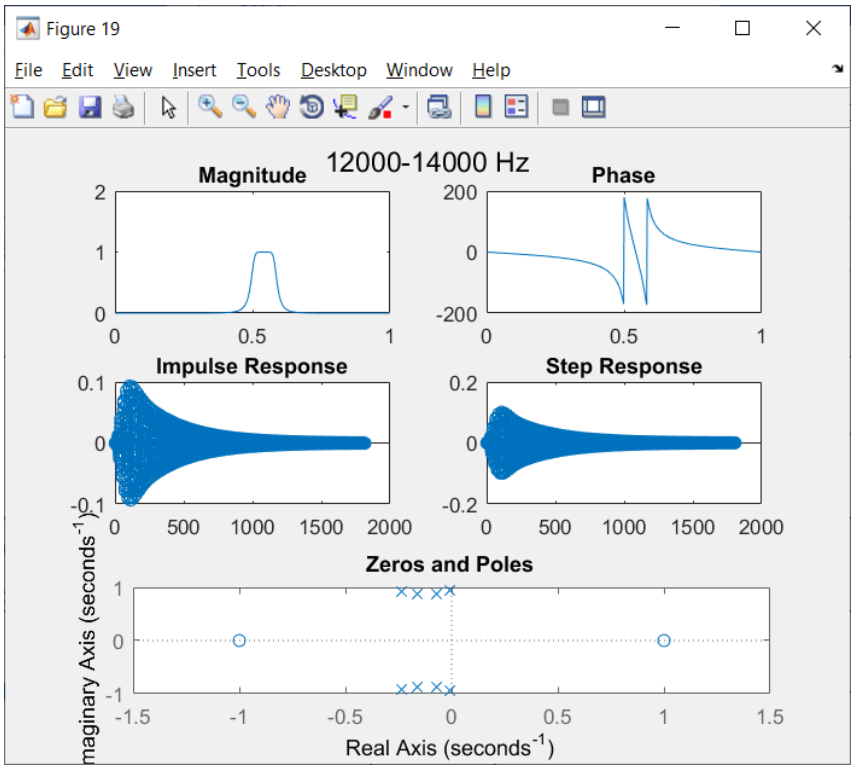
(3000-6000 Hz)



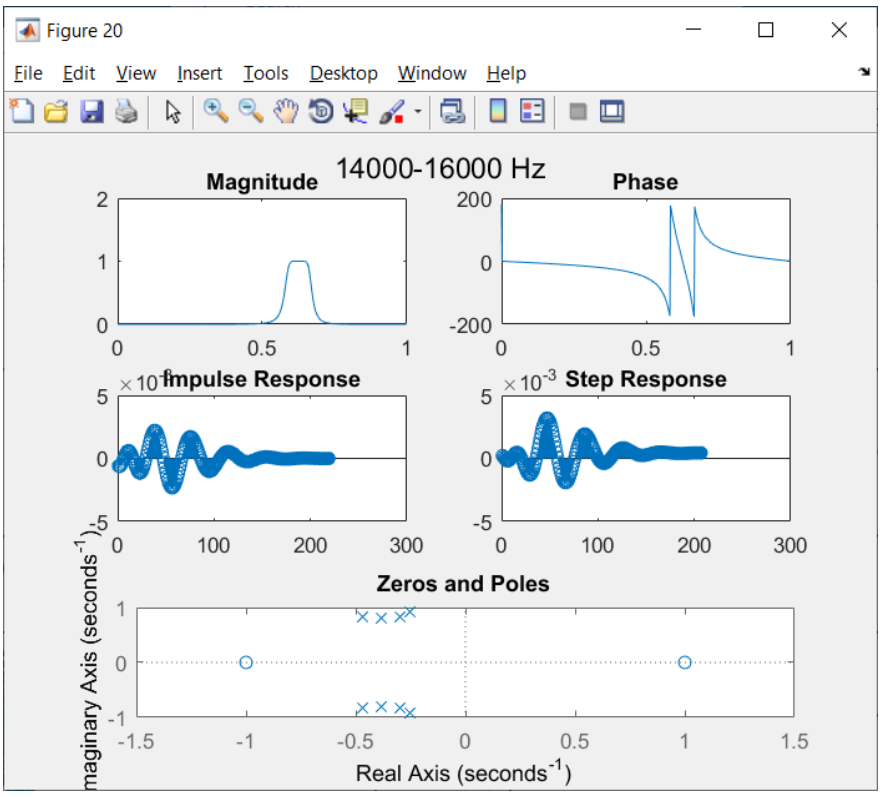
(6000-12000 Hz)



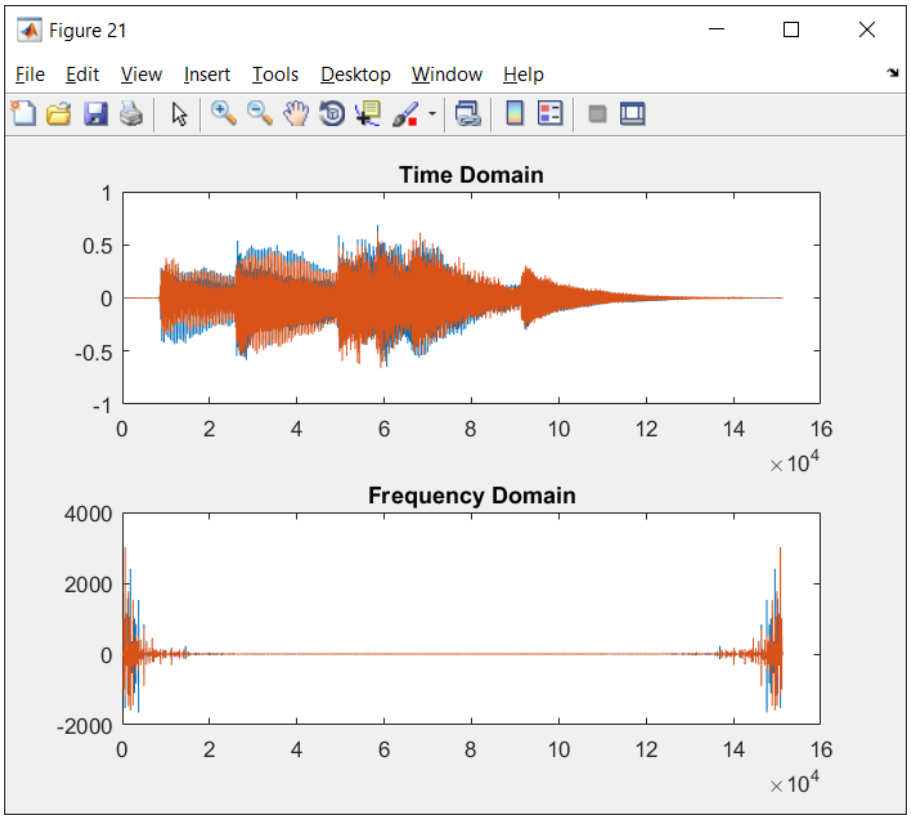
(12000-14000 Hz)



(14000-16000 Hz)



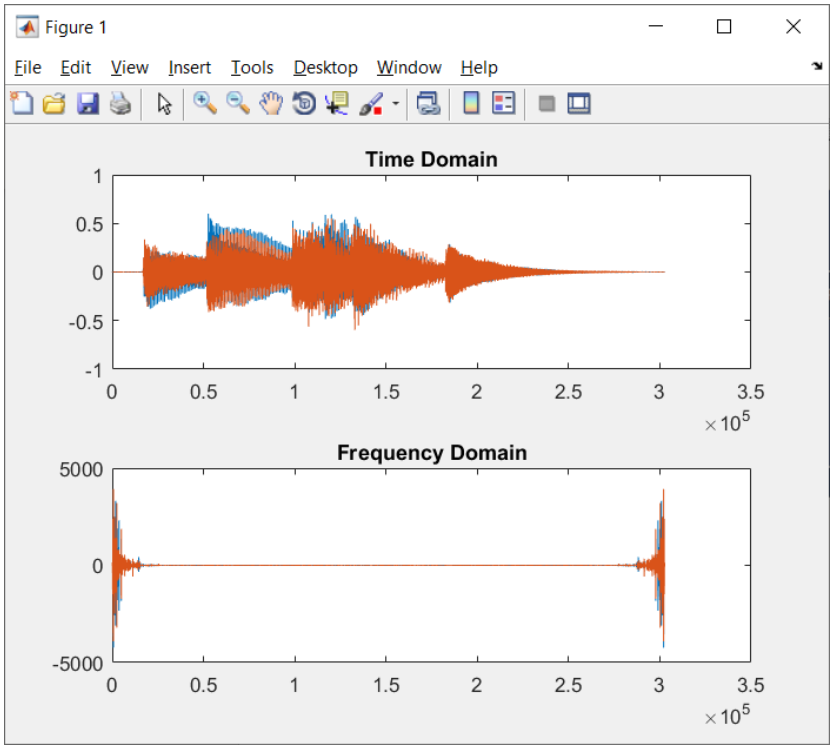
Time Domain & Frequency Domain of composite signal (after IIR filter)



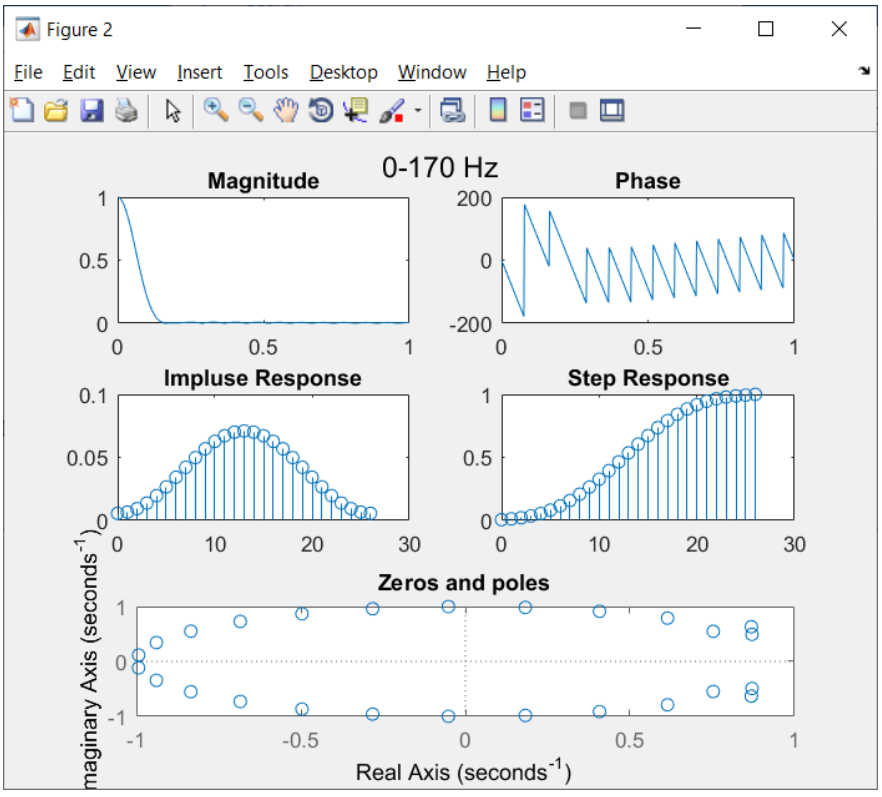
4) IF the output Sample is Decreased to half

Gui view

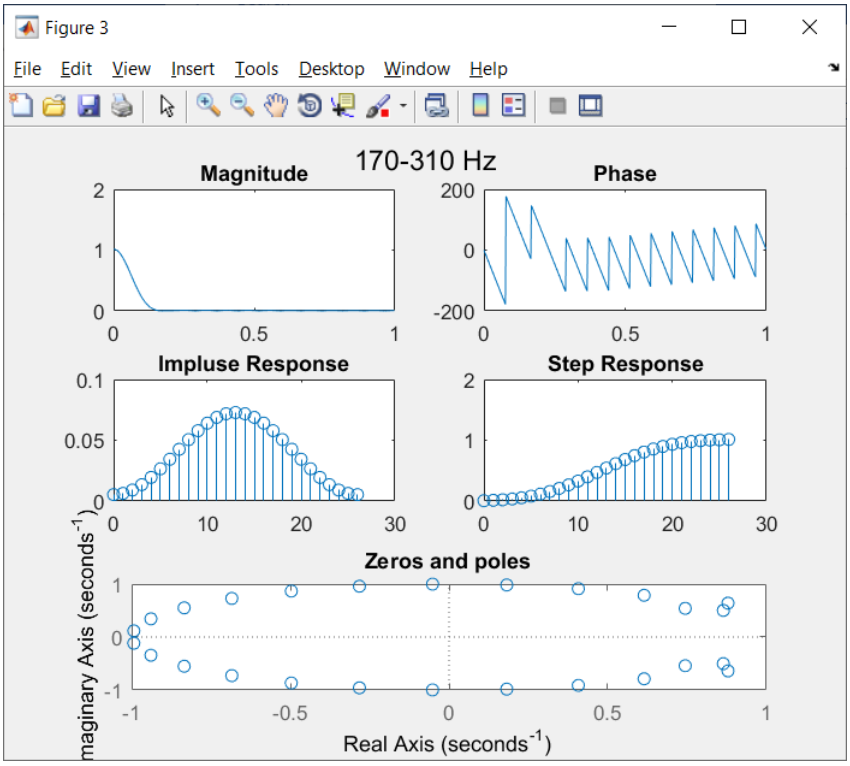
Time Domain & Frequency Domain of original signal



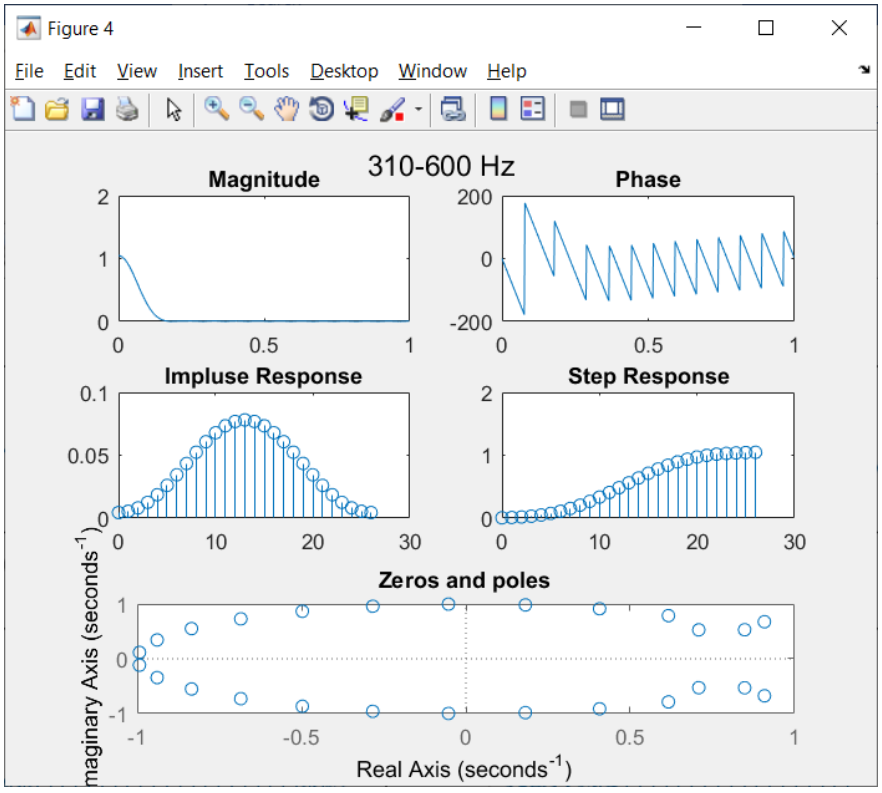
Signal after using FIR filter (0-170 Hz)



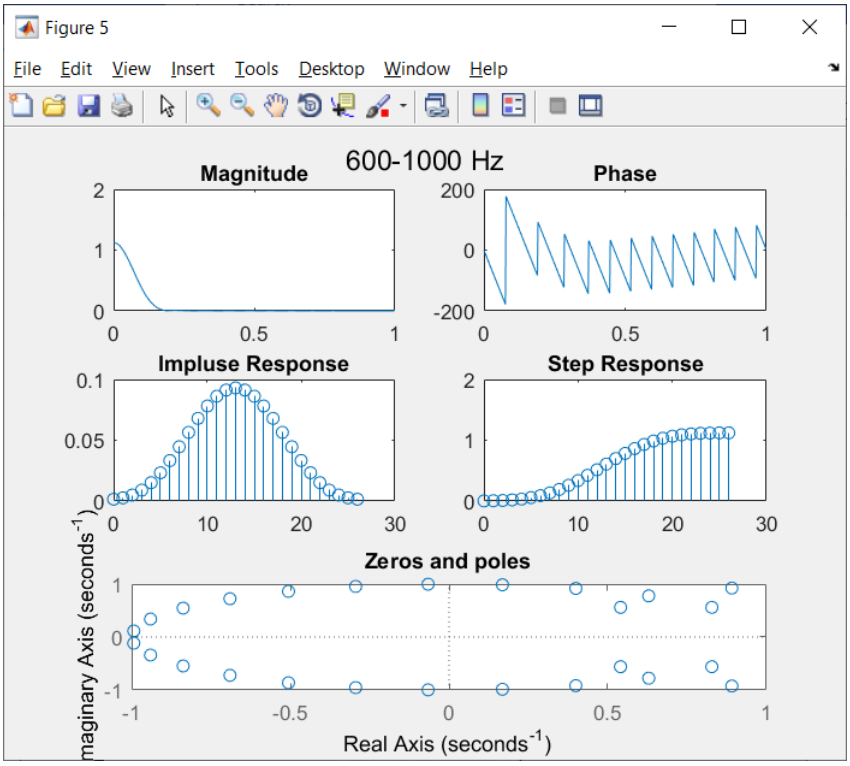
(170-310Hz)



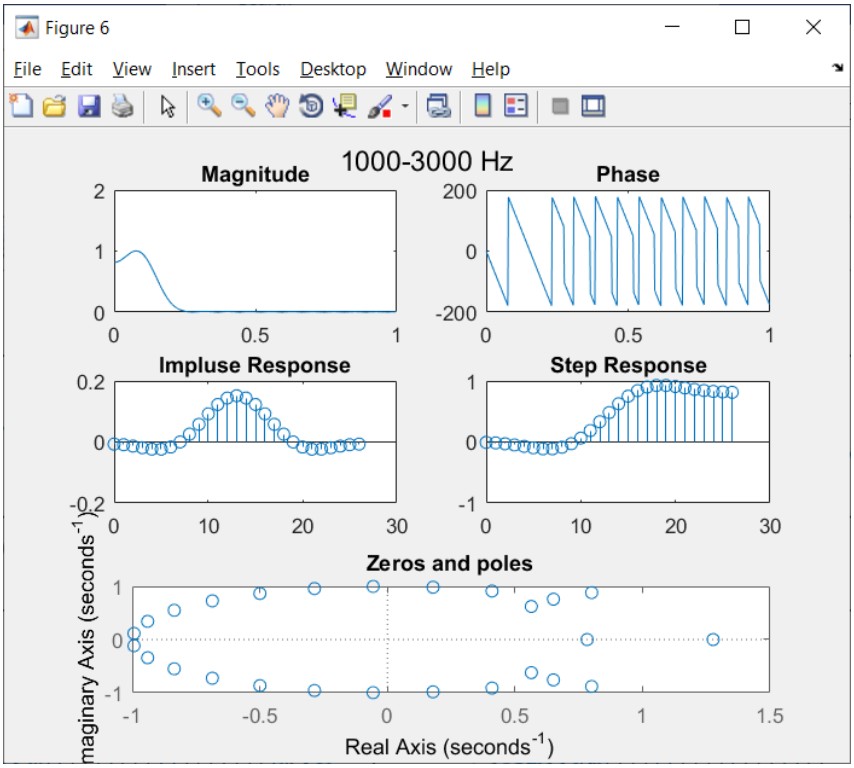
(310-600 Hz)



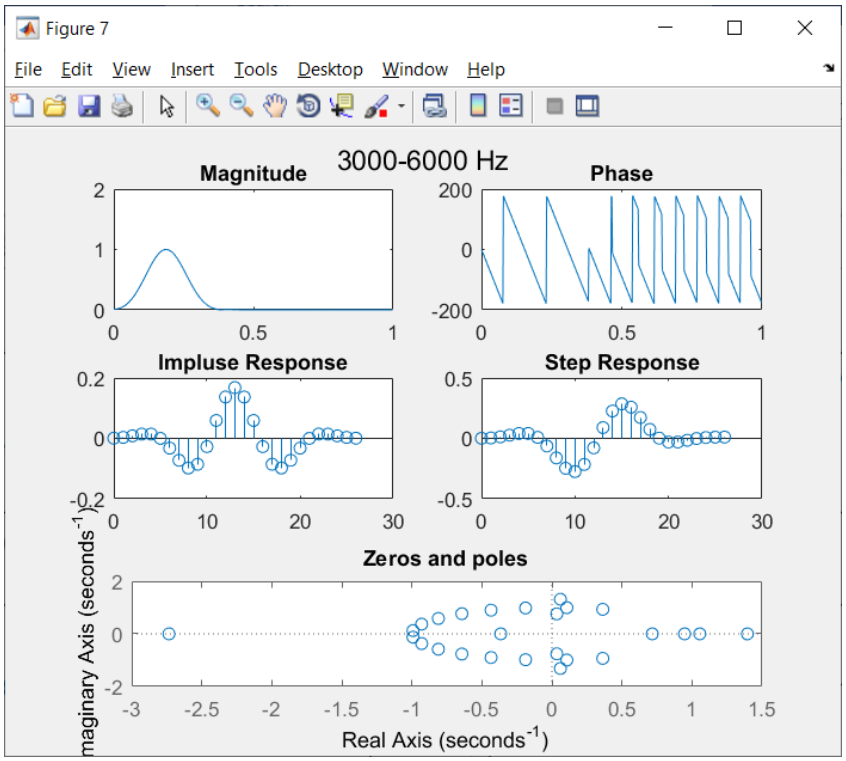
(600-1000 Hz)



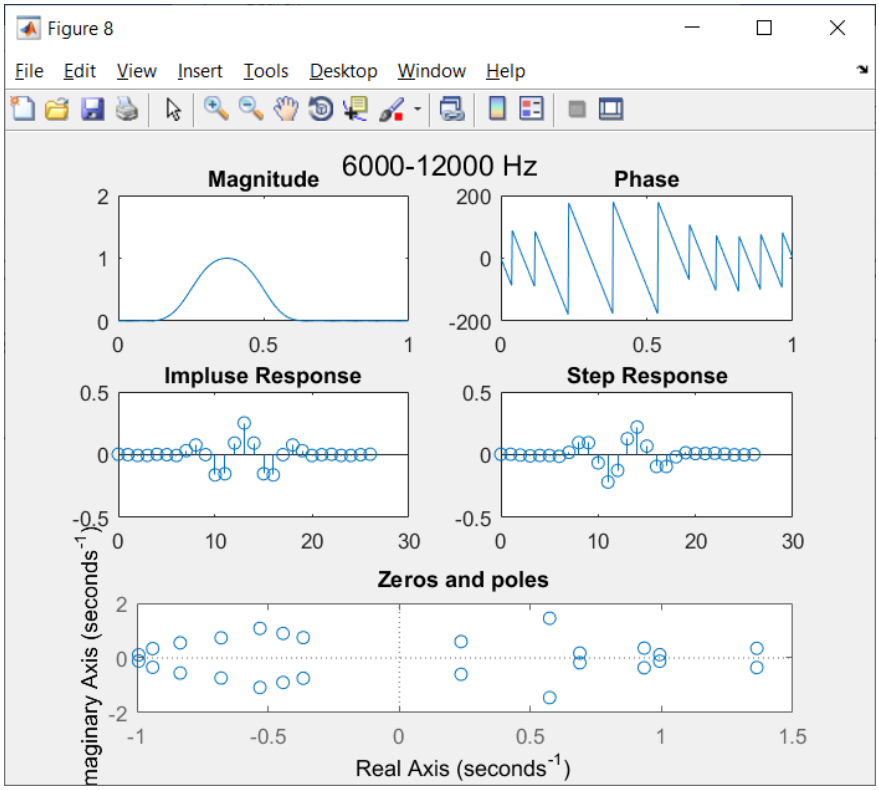
(1000-3000 Hz)



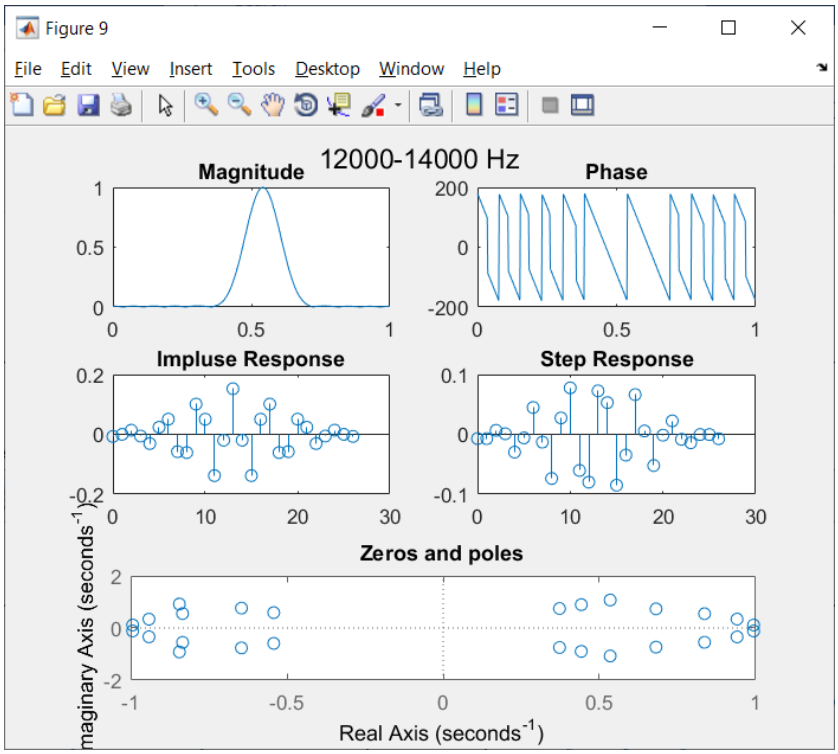
(3000-6000 Hz)



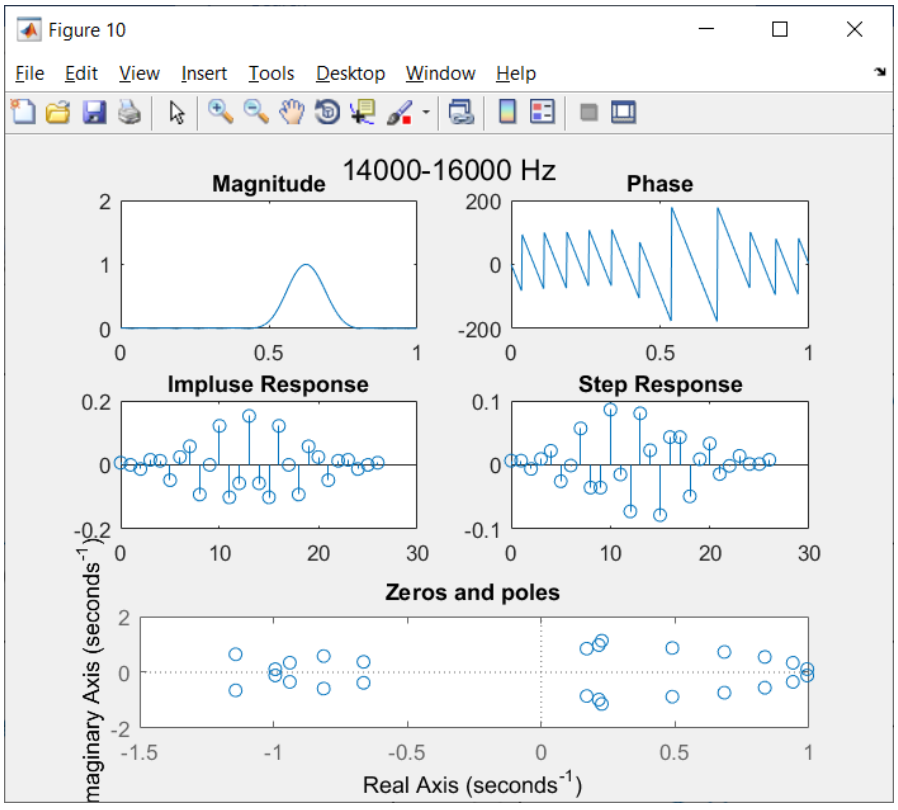
(6000-12000 Hz)



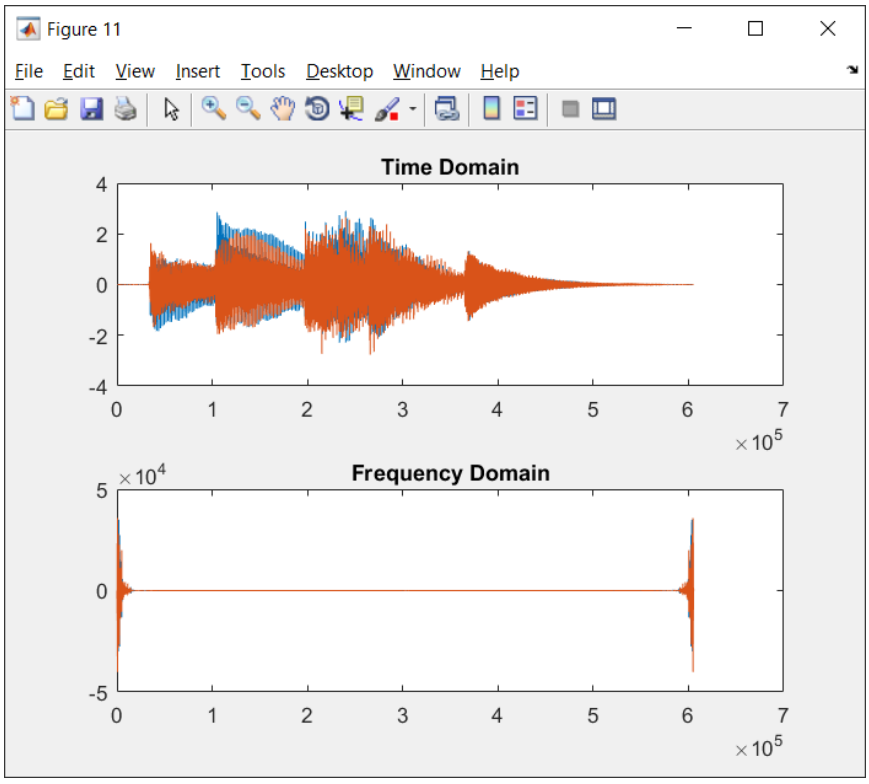
(12000-14000 Hz)



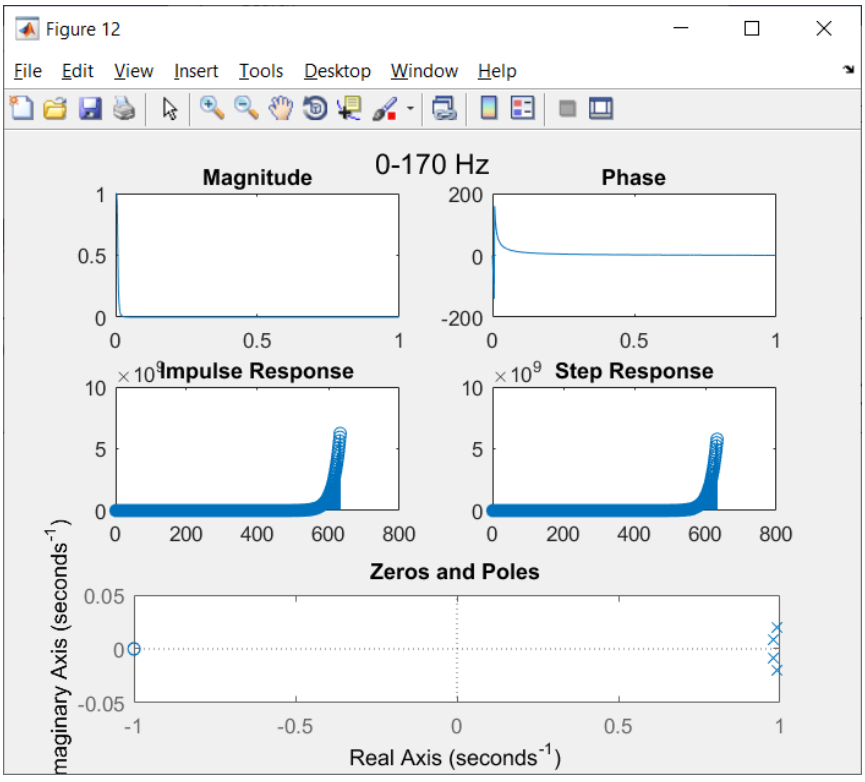
(14000-16000 Hz)



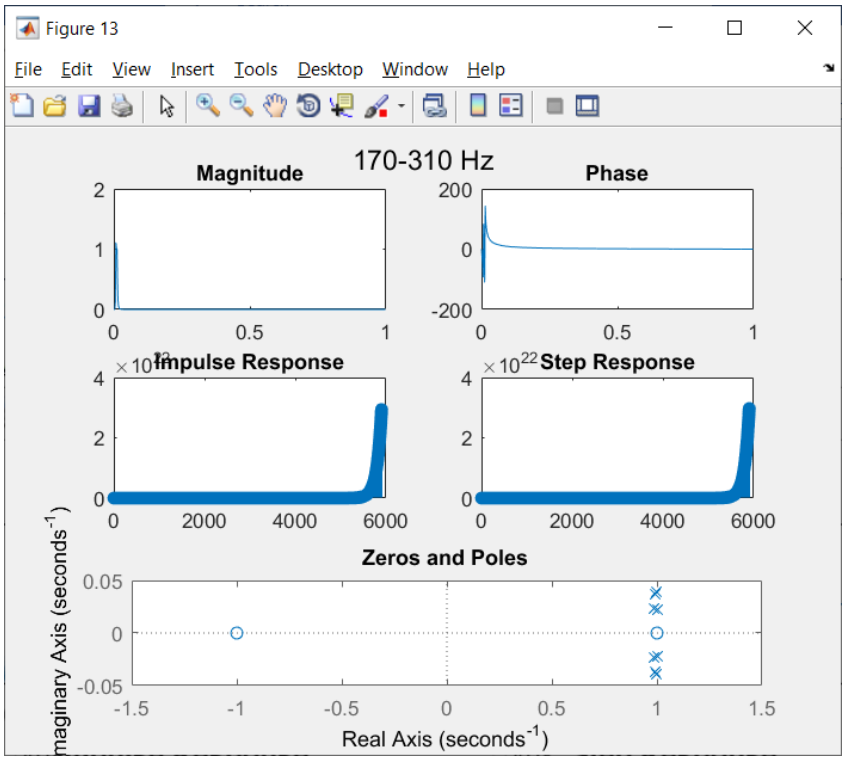
Time Domain & Frequency Domain of composite signal (after FIR filter)



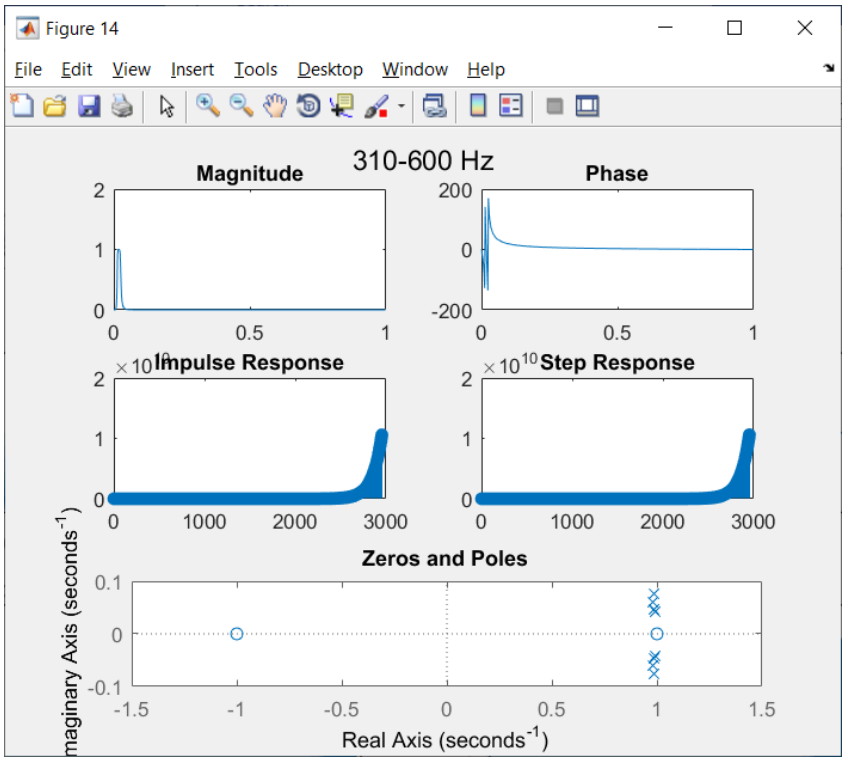
Signal after using IIR filter (0-170 Hz)



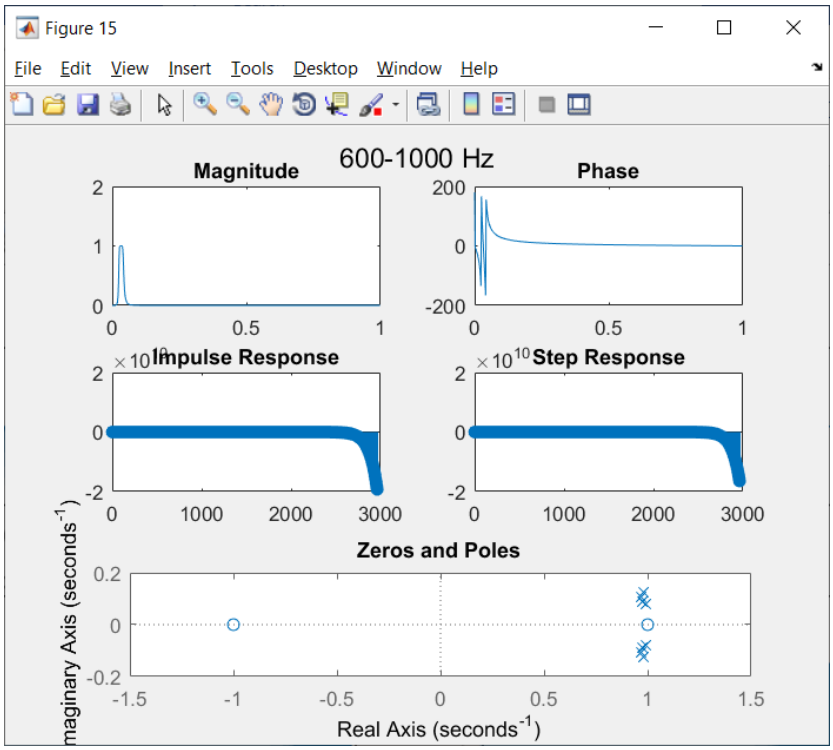
(170-310 Hz)



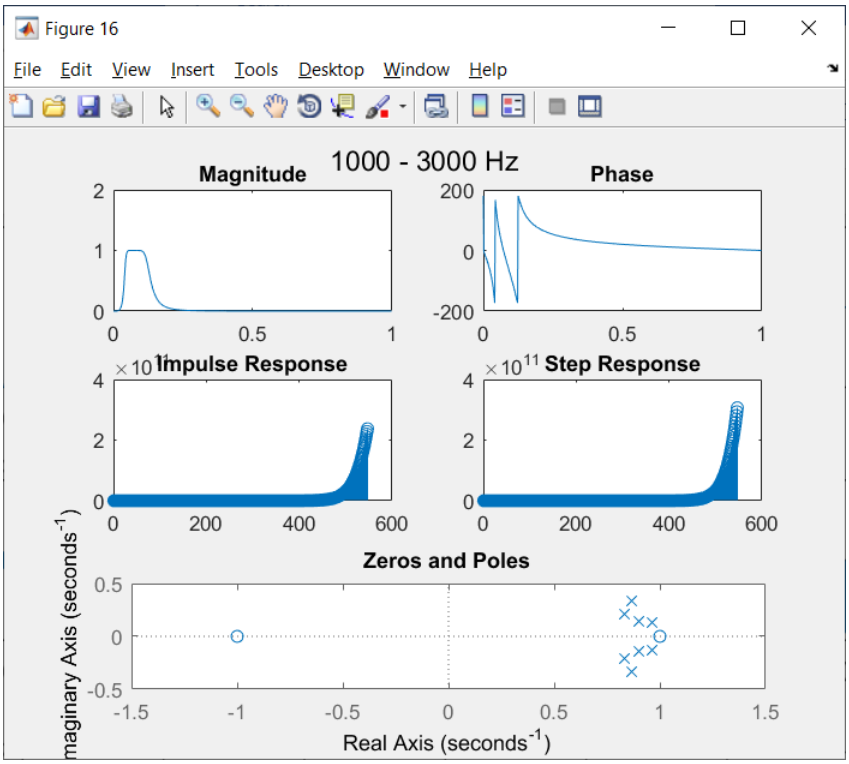
(310-600 Hz)



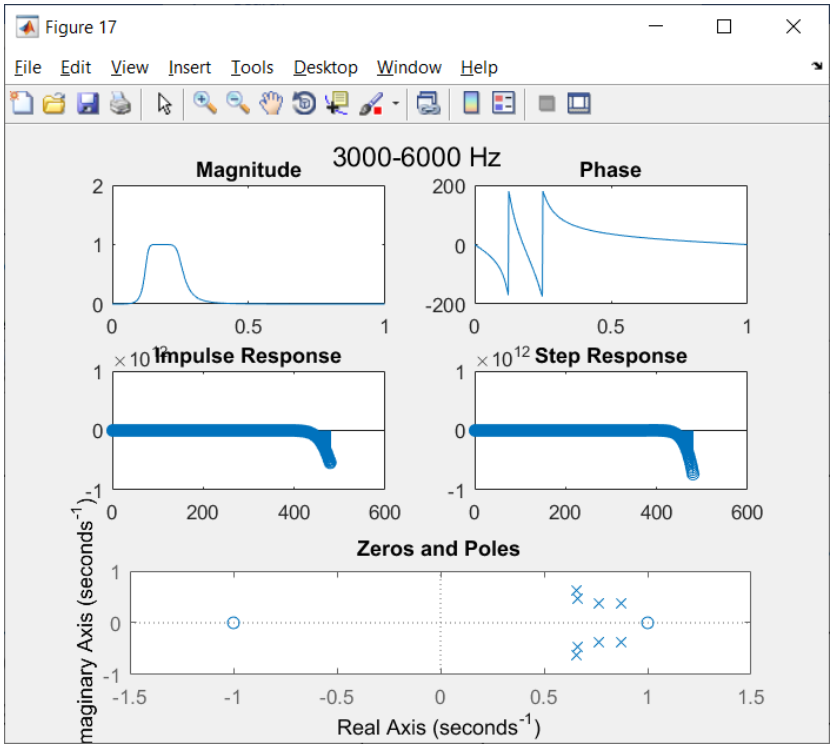
(600-1000 Hz)



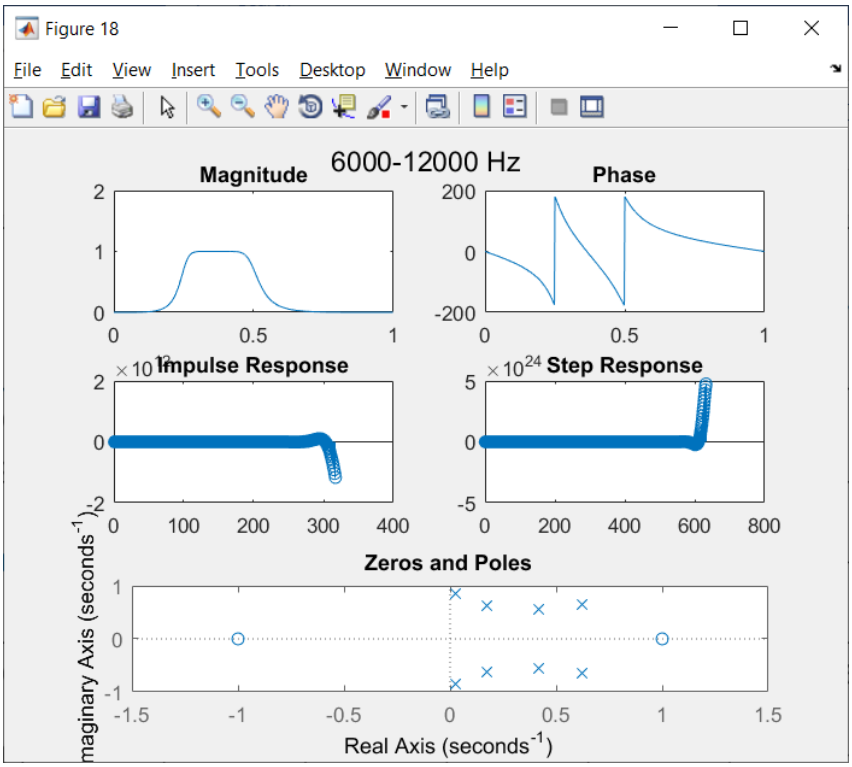
(1000-3000 Hz)



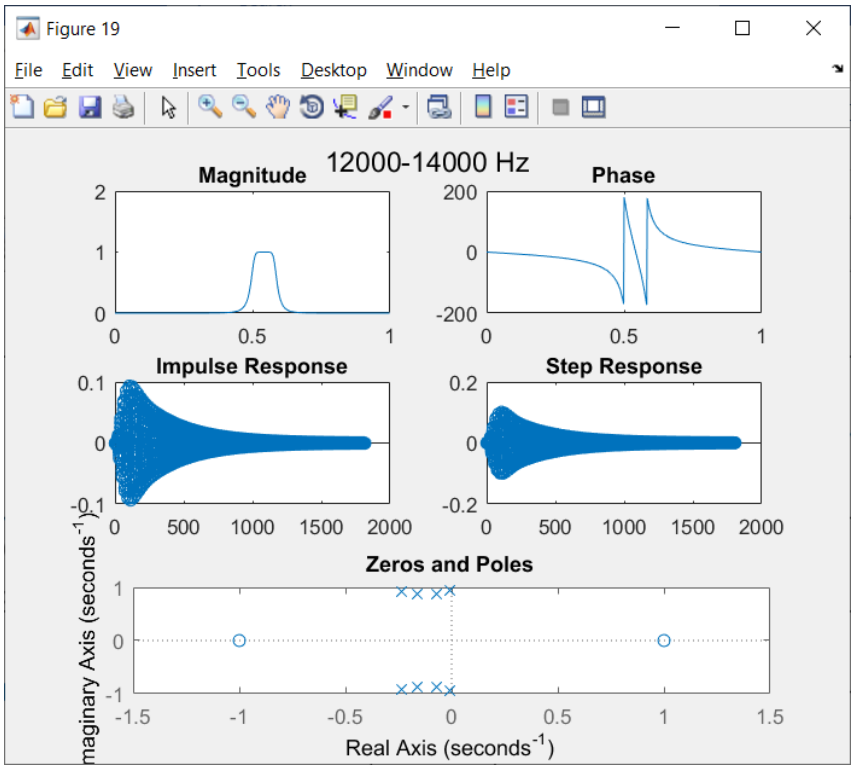
(3000-6000 Hz)



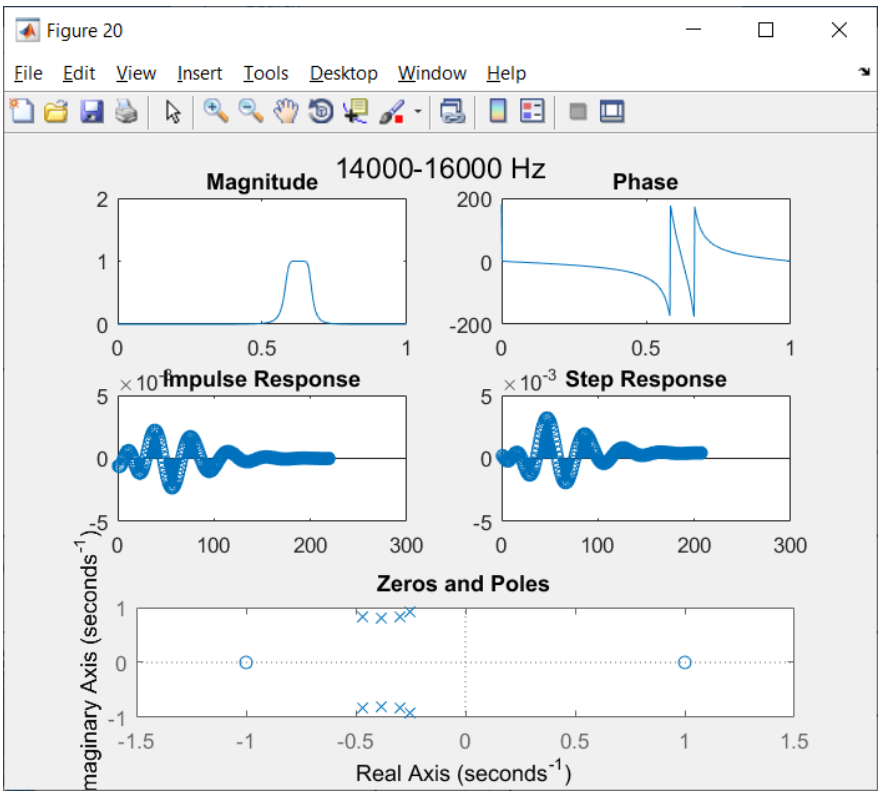
(6000-12000 Hz)



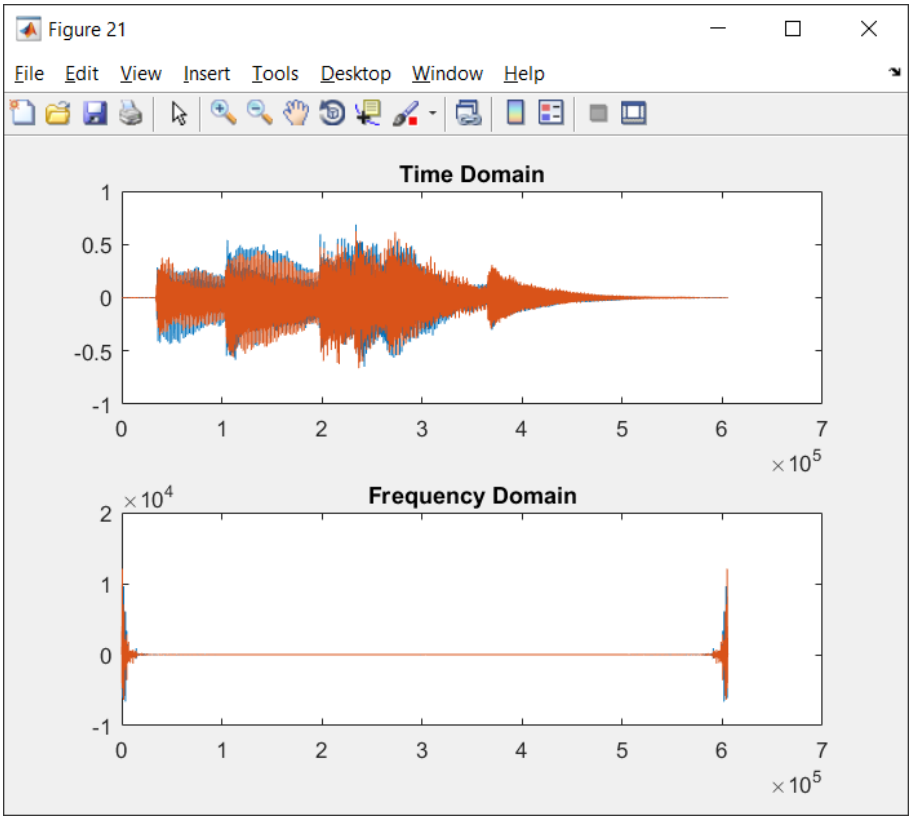
(12000-14000 Hz)



(14000-16000 Hz)

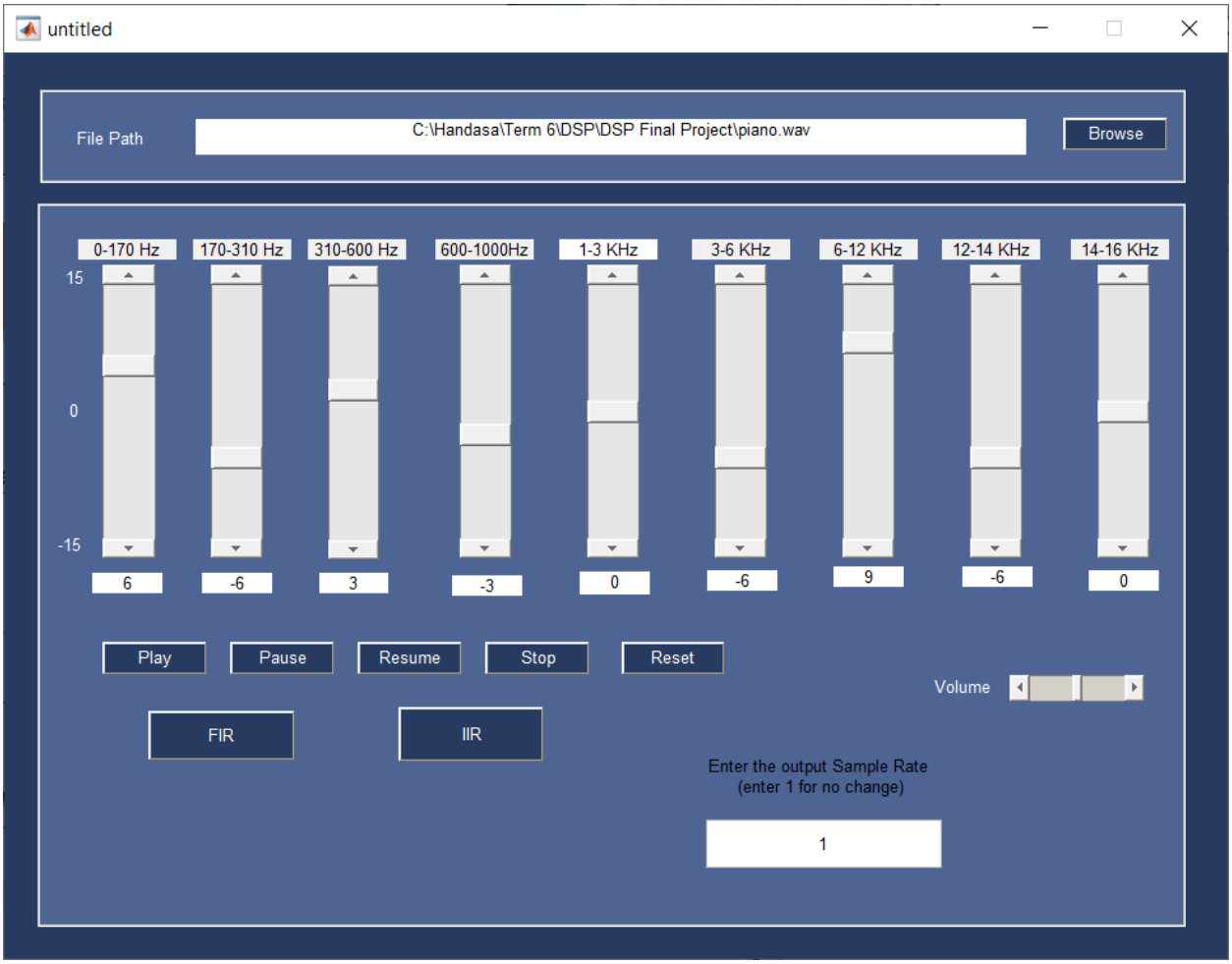


Time Domain & Frequency Domain of composite signal (after IIR filter)

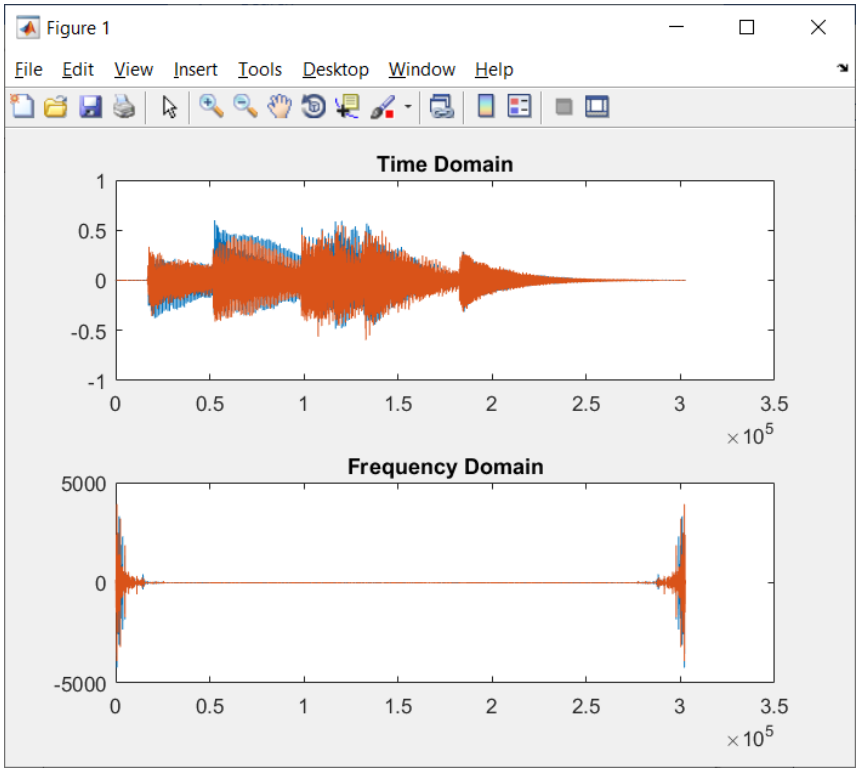


5) IF the output Sample same with different gains

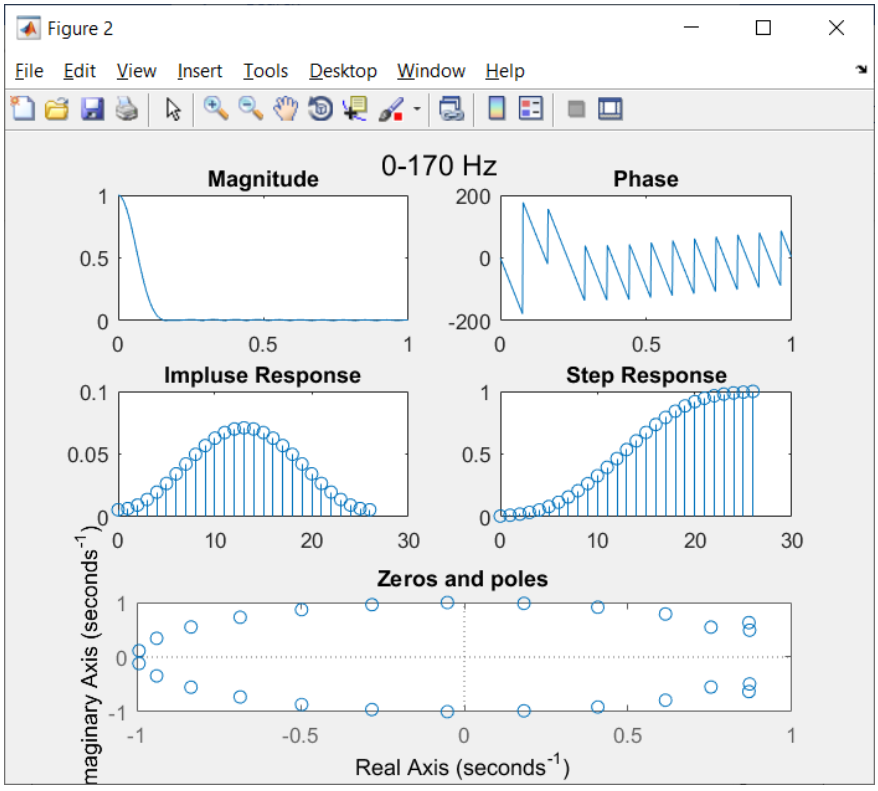
Gui view



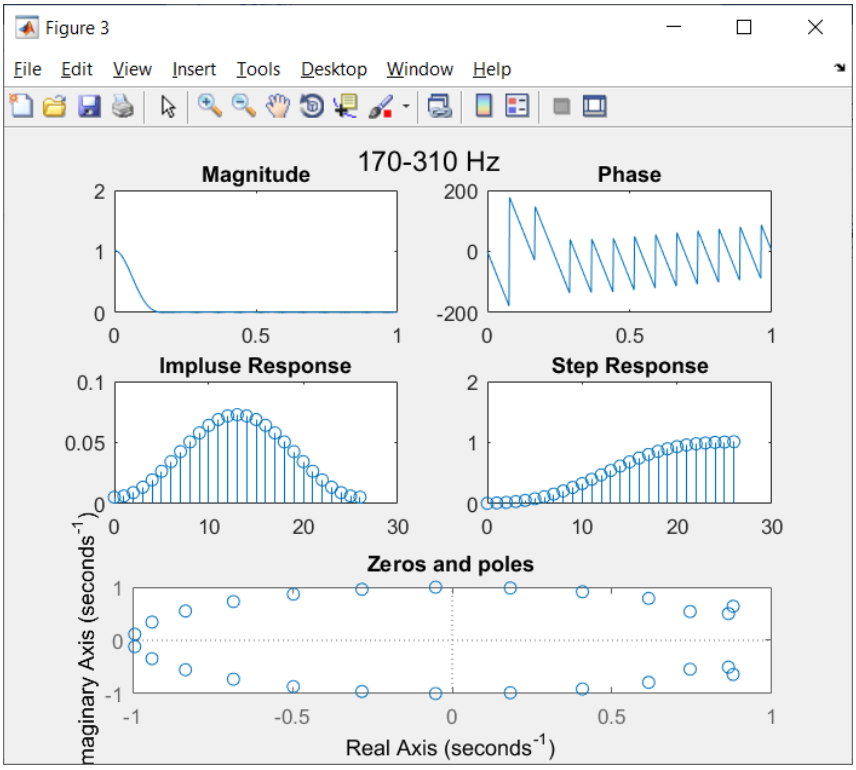
Time Domain & Frequency Domain of original signal



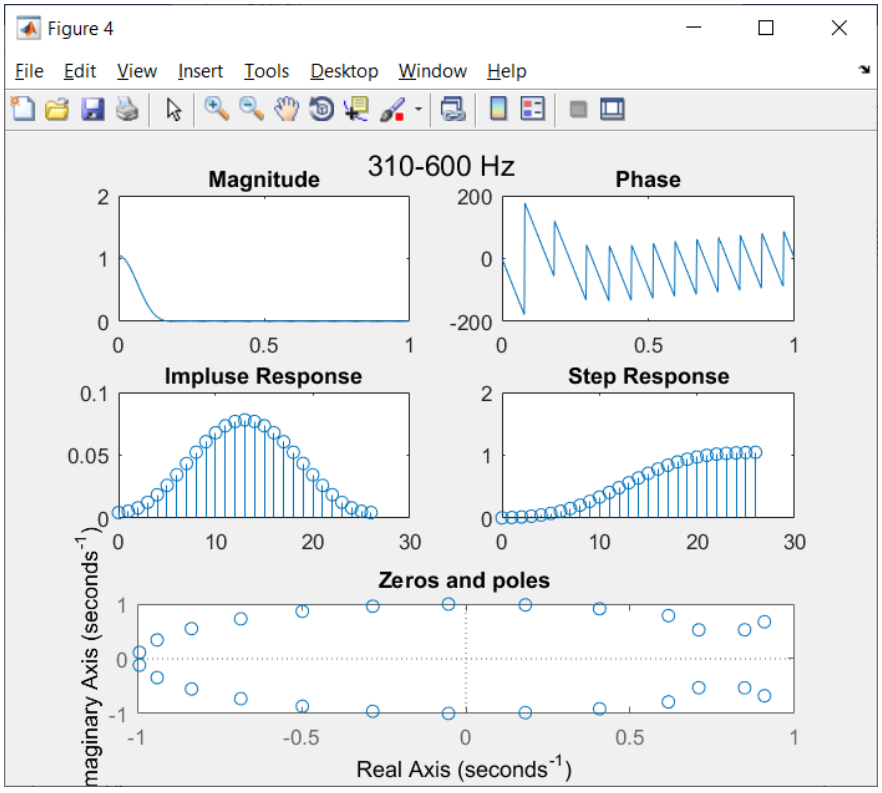
Signal after using FIR filter (0-170 Hz)



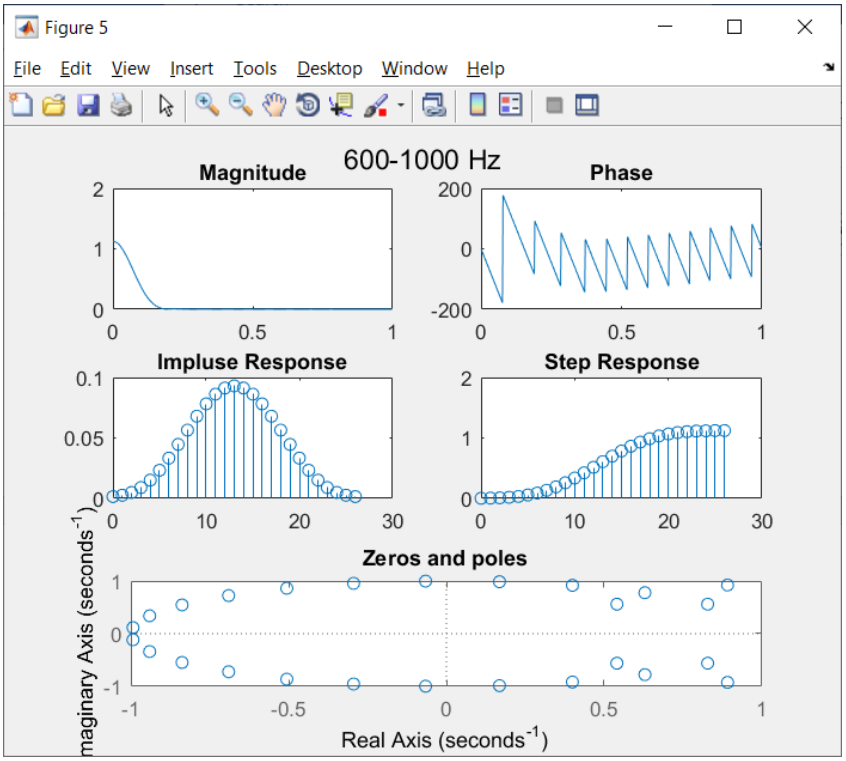
(170-310Hz)



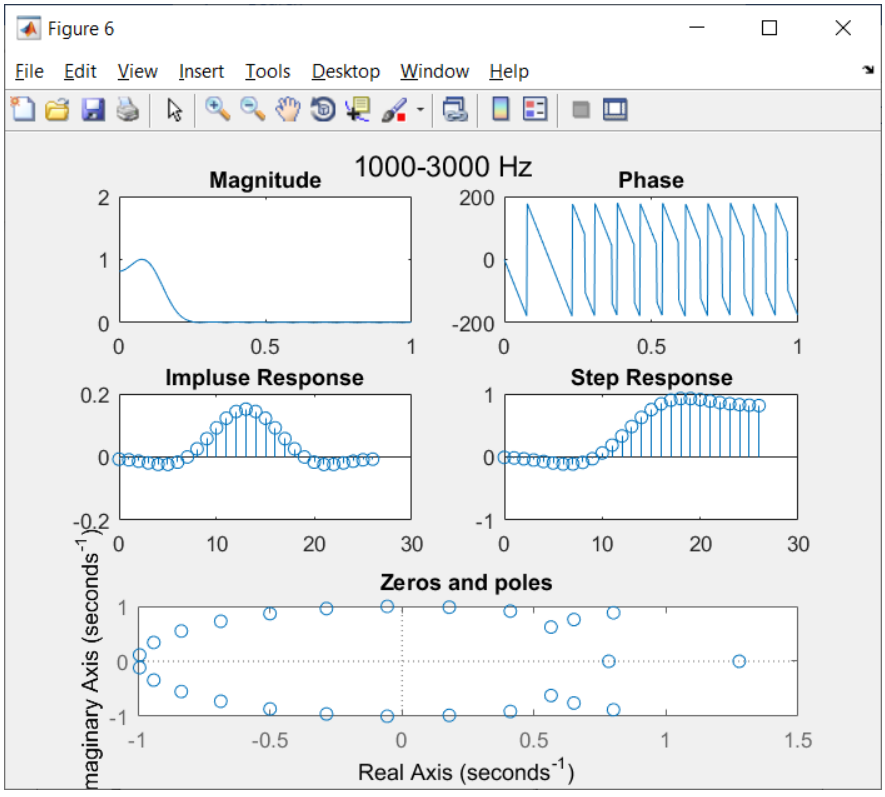
(310-600 Hz)



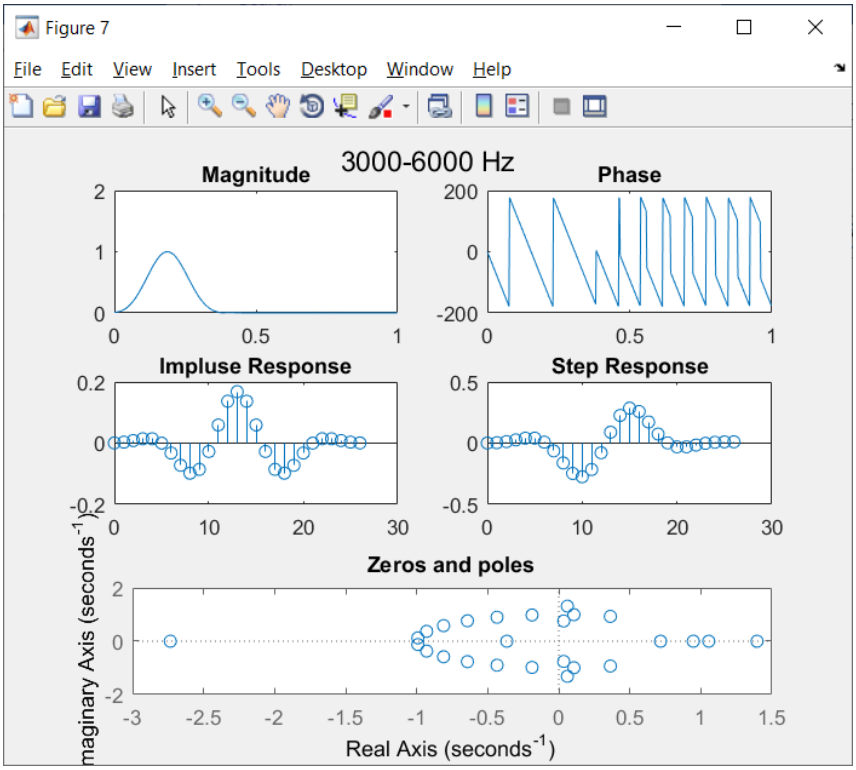
(600-1000 Hz)



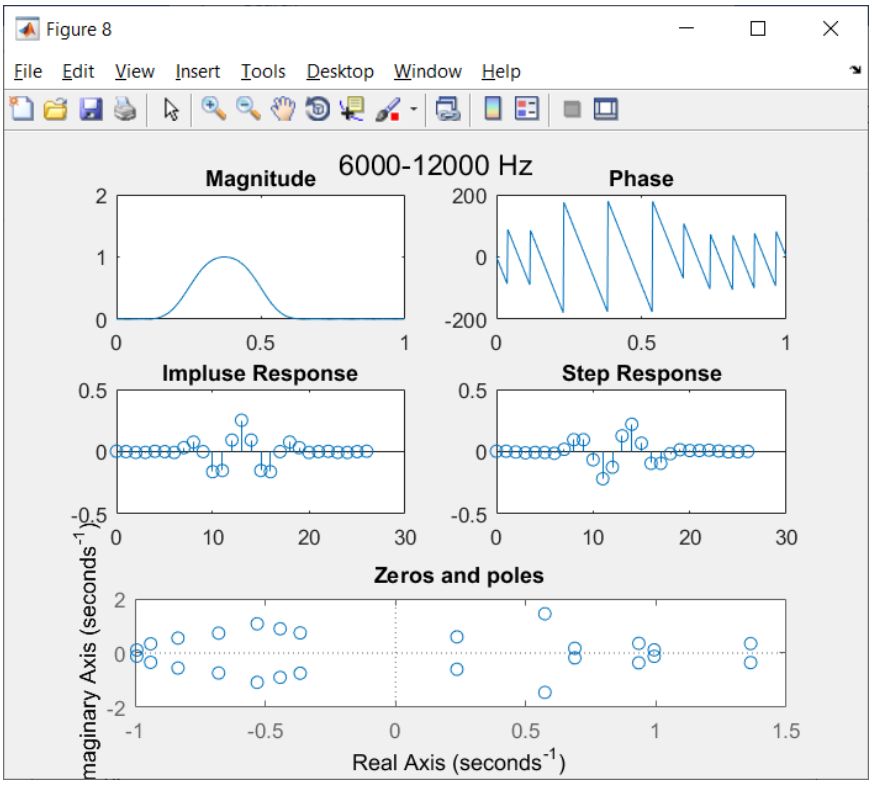
(1000-3000 Hz)



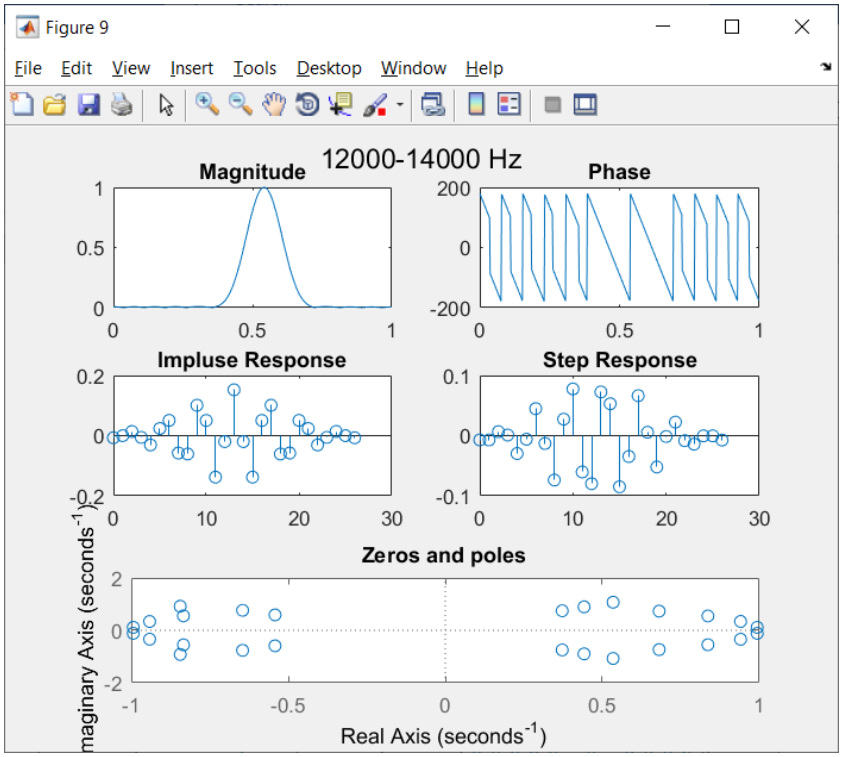
(3000-6000 Hz)



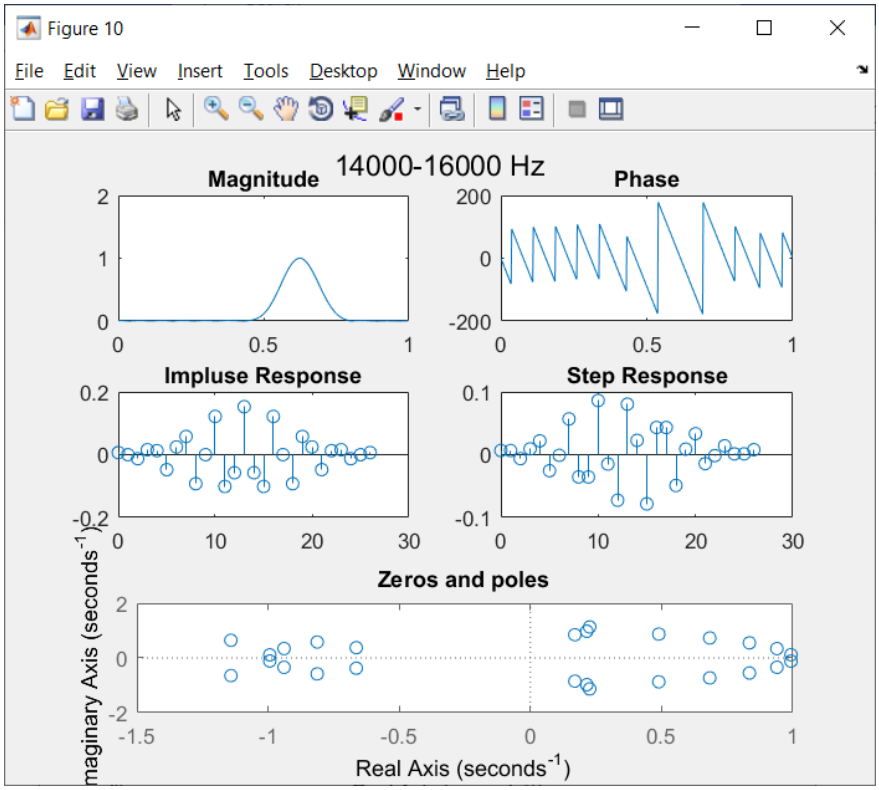
(6000-12000 Hz)



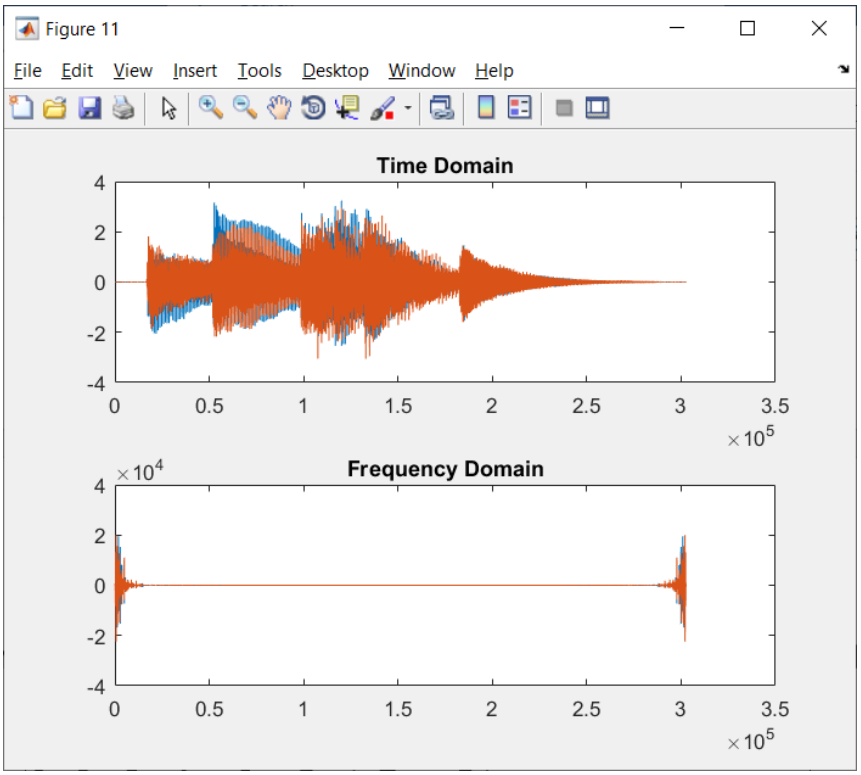
(12000-14000 Hz)



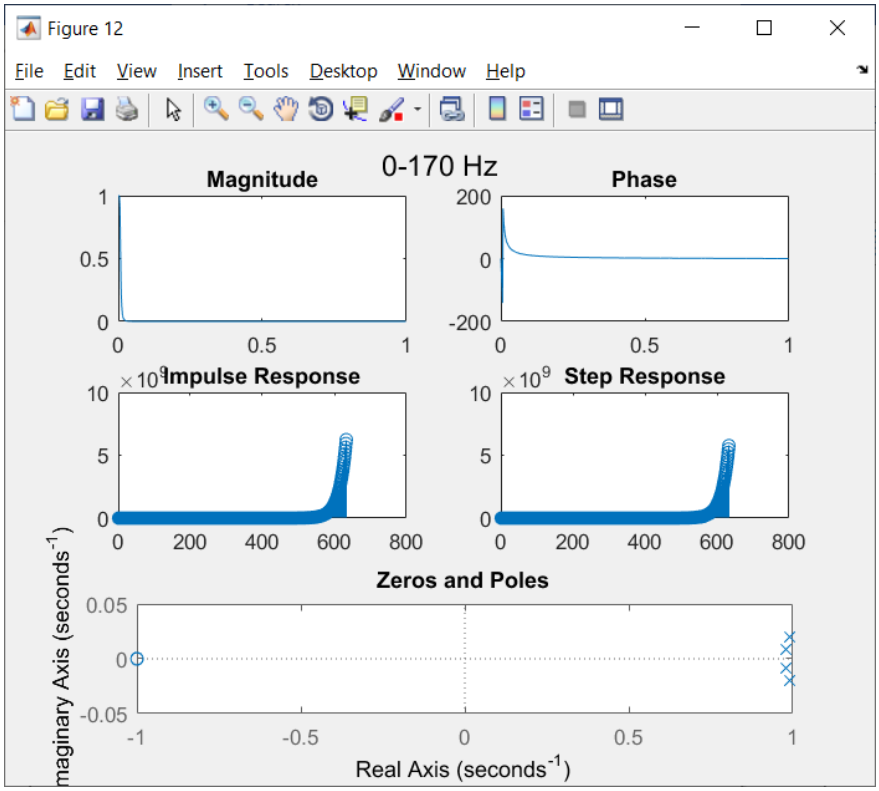
(14000-16000 Hz)



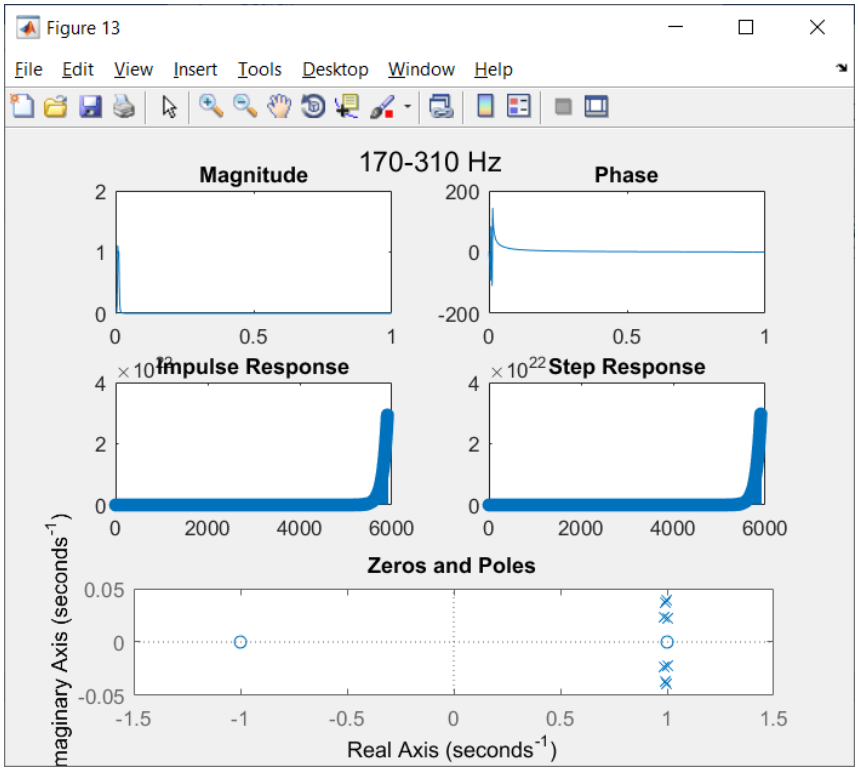
Time Domain & Frequency Domain of composite signal (after FIR filter)



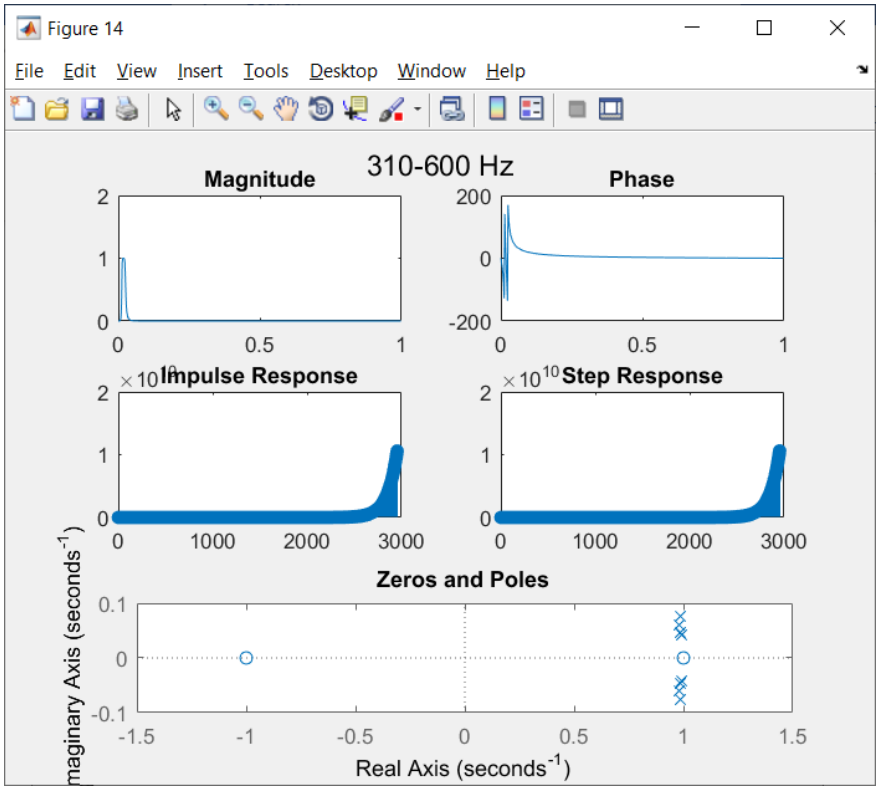
Signal after using IIR filter (0-170 Hz)



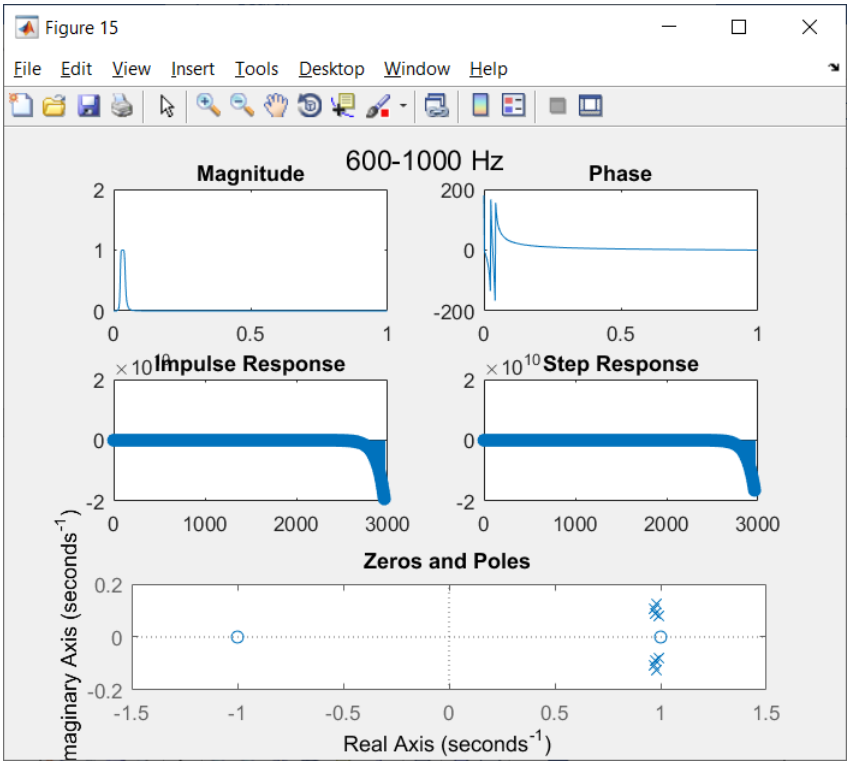
(170-310 Hz)



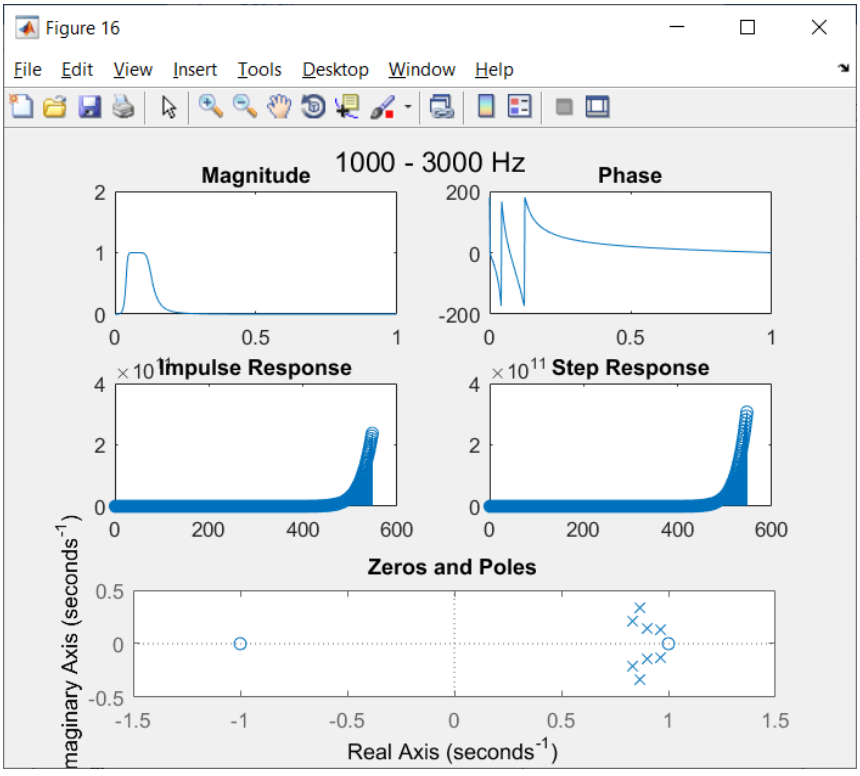
(310-600 Hz)



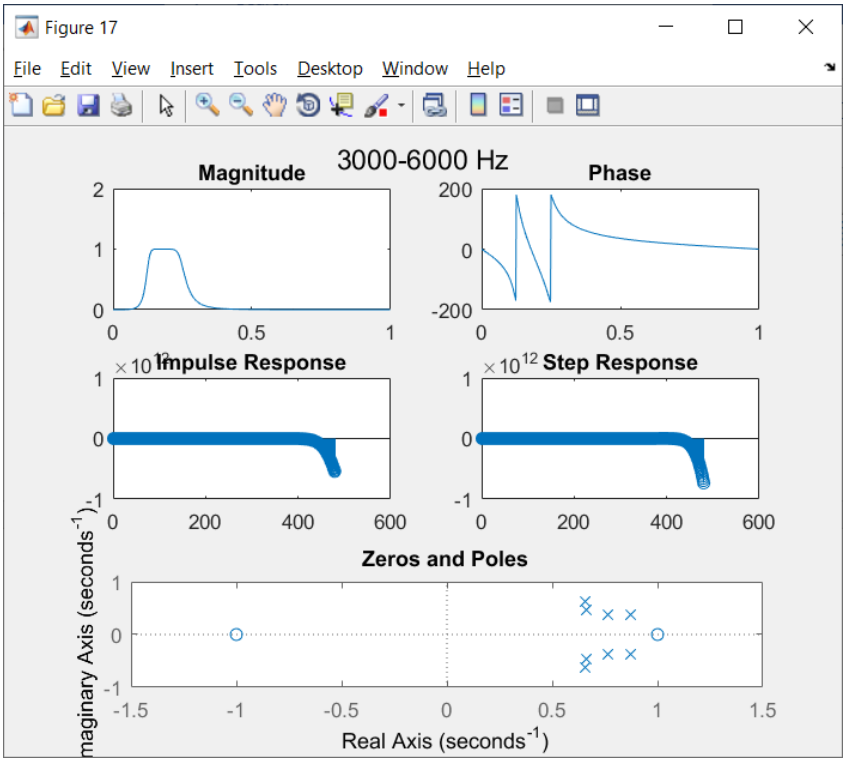
(600-1000 Hz)



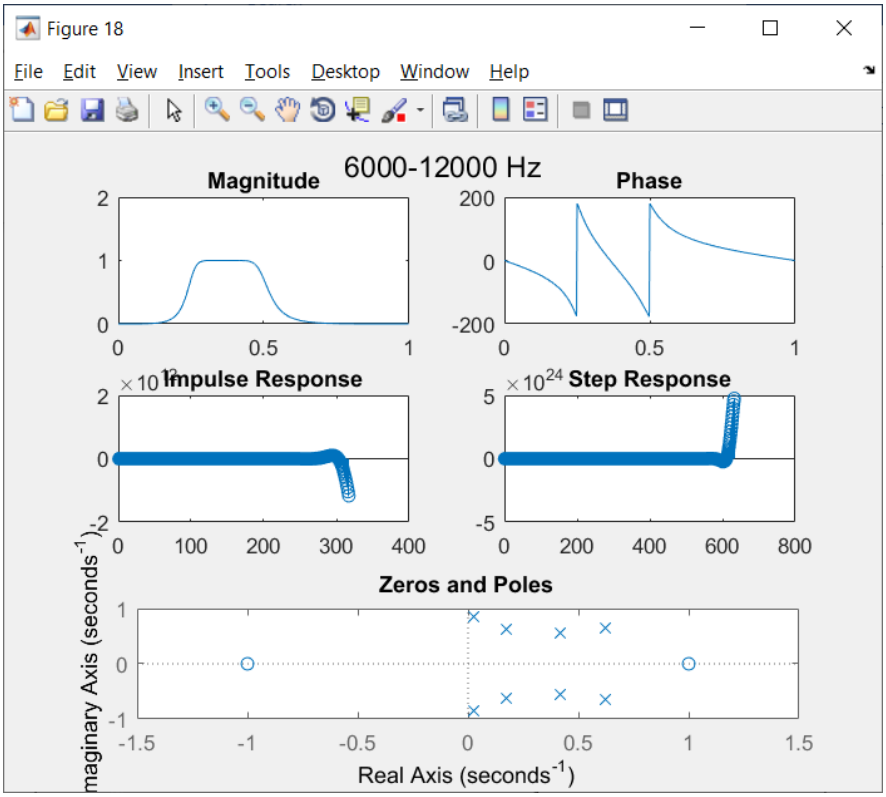
(1000-3000 Hz)



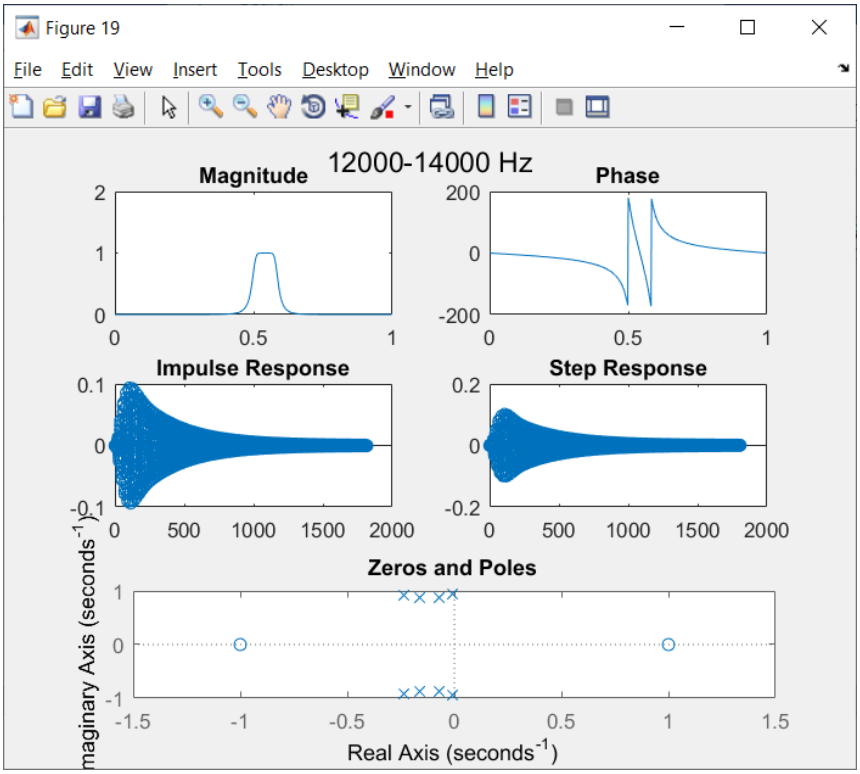
(3000-6000 Hz)



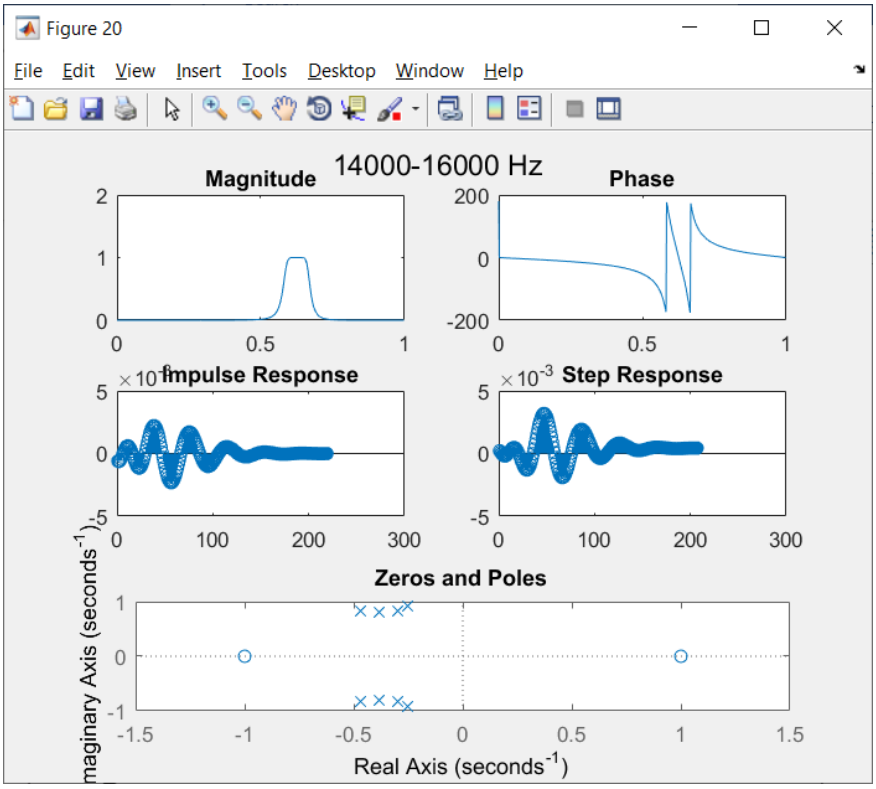
(6000-12000 Hz)



(12000-14000 Hz)



(14000-16000 Hz)



Time Domain & Frequency Domain of composite signal (after IIR filter)