Vibrations Wing Lab #3

```
clear all
close all
clc
format short
fontSize = 16.0;

%=========================================================================
%Define manual filter that will in this case remove the first 5
 Fourier
%coefficients in the frequency domain and use ifft to get a new time
 series
%that will remove some of the noise. This new dataset will be called
 "Cor"
%=========================================================================

%used for function ManFilt call.

df = 0.3125;
dt=0.00039062501;
ave='noave';
Freq = [5 10 13.50 15 20 25 30 35 40 45 50 51.70 55];
Options1 = {'no', 1};
Options2 = {'no', 2};

% Random data signals

for i = 1:5;
    FileName = strcat('Random_',int2str(i),'.mat');
    load(FileName,'-mat');

    Random.ch1.Raw(:,i) = Time_chan_1;
    Random.ch2.Raw(:,i) = Time_chan_2;
    Random.ch3.Raw(:,i) = Time_chan_3;
    Random.ch4.Raw(:,i) = Time_chan_4;

end

% Noise.ch1.Mean = mean(Noise.ch1.Raw')'; Noise.ch2.Mean  =
 mean(Noise.ch2.Raw')';
% Noise.ch3.Mean = mean(Noise.ch3.Raw')'; Noise.ch4.Mean  =
 mean(Noise.ch4.Raw')';
% Noise.t = Time_domain;
% Noise.f = Freq_domain;

TimeRange = Time_domain;
FreqRange = Freq_domain;

NUMS = 5:5:55;
NUMS = [NUMS 13 51];
NUMS=sort(NUMS);
```

```
NUMS2 = {'00', '00', '50', '00', '00', '00', '00', '00', '00', ...
    '00', '00', '70', '00'}';

S = [];
for j = 1:length(NUMS);
Data = [];
for i = 1:5;
    if NUMS(j) == 20 && i ==3;

    else
    FileName =
 strcat('freq_00',figureTag(NUMS(j)),'_',NUMS2(j,1),'_',int2str(i),'.mat');
    FileName=char(FileName{1});
    load(FileName,'-mat');

    Data.ch(1).Raw(:,i) = Time_chan_1;
    Data.ch(1).Cor(:,i) = ManFilt(Time_chan_1, dt);
    %
    Data.ch(2).Raw(:,i) = Time_chan_2;
    Data.ch(2).Cor(:,i) = ManFilt(Time_chan_2, dt);
    %
    FreqRange = asd(Data.ch(2).Cor(:,i), dt, [], ave);
    %
    Data.ch(3).Raw(:,i) = Time_chan_3;
    Data.ch(3).Cor(:,i) = ManFilt(Time_chan_3, dt);
    %
    Data.ch(4).Raw(:,i) = Time_chan_4;
    Data.ch(4).Cor(:,i) = ManFilt(Time_chan_4, dt);
    %
    % get Tf for channel
 2==============================================
    [~, Data.ch(2).TfrH1(:,i)] = tfest(Data.ch(2).Raw(:,i), ...
        Data.ch(3).Raw(:,i), dt, [], Options1);
    [~, Data.ch(2).TfcH1(:,i)] = tfest(Data.ch(2).Cor(:,i), ...
        Data.ch(3).Cor(:,i), dt, [], Options1);
    %
    [~, Data.ch(2).TfrH2(:,i)] = tfest(Data.ch(2).Raw(:,i), ...
        Data.ch(3).Raw(:,i), dt, [], Options2);
    [~, Data.ch(2).TfcH2(:,i)] = tfest(Data.ch(2).Cor(:,i), ...
        Data.ch(3).Cor(:,i), dt, [], Options2);

    % get Tf for channel
 4==============================================
    [~, Data.ch(4).TfrH1(:,i)] = tfest(Data.ch(4).Raw(:,i), ...
        Data.ch(3).Raw(:,i), dt, [], Options1);
    [~, Data.ch(4).TfcH1(:,i)] = tfest(Data.ch(4).Cor(:,i), ...
        Data.ch(3).Cor(:,i), dt, [], Options1);
    %
    [~, Data.ch(4).TfrH2(:,i)] = tfest(Data.ch(4).Raw(:,i), ...
        Data.ch(3).Raw(:,i), dt, [], Options2);
    [~, Data.ch(4).TfcH2(:,i)] = tfest(Data.ch(4).Cor(:,i), ...
        Data.ch(3).Cor(:,i), dt, [], Options2);
```

```
    end
end

S(j).Data = Data;
%S(j).obj  = SignalAnalysis( Data, Noise.t);
S(j).num = NUMS(j);

end

%Find index of frequency values needed to constuct transfer function
Q=[];
for k = 1:length(Freq);
    B=(FreqRange>(Freq(k)-0.15) & FreqRange<(Freq(k)+0.15));
    Qi=find(B==1);
    Q = [Q; Qi];
end

Ext=[];
%Extract values from the PSD and CRSD vectors
for l = 1:length(NUMS);
    Data=[];
    for p = 1:5;
        %Channel 2 ~output~
        Data.ch(2).TfrH1(:,p) = S(l).Data.ch(2).TfrH1(Q(l),p);
        Data.ch(2).TfcH1(:,p) = S(l).Data.ch(2).TfcH1(Q(l),p);
        Data.ch(2).TfrH2(:,p) = S(l).Data.ch(2).TfrH1(Q(l),p);
        Data.ch(2).TfcH2(:,p) = S(l).Data.ch(2).TfcH1(Q(l),p);

        %Channel 4 ~output~
        Data.ch(4).TfrH1(:,p) = S(l).Data.ch(4).TfrH1(Q(l),p);
        Data.ch(4).TfcH1(:,p) = S(l).Data.ch(4).TfcH1(Q(l),p);
        Data.ch(4).TfrH2(:,p) = S(l).Data.ch(4).TfrH1(Q(l),p);
        Data.ch(4).TfcH2(:,p) = S(l).Data.ch(4).TfcH1(Q(l),p);
    end
    Ext(l).Data = Data;
    Ext(l).num = NUMS(l);

    Tf2rH1(l, :) = Ext(l).Data.ch(2).TfrH1;
    Tf2cH1(l, :) = Ext(l).Data.ch(2).TfcH1;
    Tf2rH2(l, :) = Ext(l).Data.ch(2).TfrH2;
    Tf2cH2(l, :) = Ext(l).Data.ch(2).TfcH2;

    Tf4rH1(l, :) = Ext(l).Data.ch(4).TfrH1;
    Tf4cH1(l, :) = Ext(l).Data.ch(4).TfcH1;
    Tf4rH2(l, :) = Ext(l).Data.ch(4).TfrH2;
    Tf4cH2(l, :) = Ext(l).Data.ch(4).TfcH2;

end

Tf2rH1(:, 6) = mean(Tf2rH1')';
Tf2cH1(:, 6) = mean(Tf2cH1')';
Tf2rH2(:, 6) = mean(Tf2rH2')';
Tf2cH2(:, 6) = mean(Tf2cH2')';
%
```

```
Tf4rH1(:, 6) = mean(Tf4rH1')';
Tf4cH1(:, 6) = mean(Tf4cH1')';
Tf4rH2(:, 6) = mean(Tf4rH2')';
Tf4cH2(:, 6) = mean(Tf4cH2')';

%Get Coherence
COH2r = abs(Tf2rH1(:, 6)./Tf2rH2(:, 6));
COH4r = abs(Tf4rH1(:, 6)./Tf4rH2(:, 6));

clearvars -
except S ave dt ave Freq FreqRange Q Tf2rH1 Tf2cH1 TimeRange ...
    NUMS NUMS2 df Random Options Ext Tf2rH1 Tf2cH1 Tf2rH2 Tf2cH2 ...
    Tf4rH1 Tf4cH1 Tf4rH2 Tf4cH2 COH2r COH4r
```

Published with MATLAB® R2015b