

SIGNALS AND SYSTEMS

ECE1004 PROJECT REPORT

NON STATIONARY SIGNAL ANALYSIS USING SHORT TIME FOURIER TRANSFORM
AND WAVELET TRANSFORM

PROJECT MEMBERS:

- 1)AYUSHMAN MOOKHERJEE -18BEC0888
- 2)AKSHAR KOMMAJOSYULA-18BEC0863
- 3)VINEET RAJPAL- 18BEC0908
- 4)VARNIKA SINGH – 18BEC0784

Overview:

- ▶ Throughout the project, our basic motive is to analyse non stationary signals using Short Time Fourier Transform(STFT) and wavelet transform by analysing their power spectrum respectively. The analysis started by understanding the basic concept of STFT and wavelet transform.

Short Time Fourier Transform:

- ▶ The short-time Fourier transform (STFT), is a Time – frequency representation (TFR) used to determine the local sinusoidal frequency and temporal content of a signal as it changes over time.
- ▶ In practice, the procedure for computing STFTs is to divide a longer time signal into shorter segments of equal length and then compute the Fourier transform separately on each shorter segment. Mathematically this is performed by multiplying the signal with the window function and then taking its fourier transform.
- ▶ This reveals the Fourier spectrum on each shorter segment. One then usually plots the changing spectra as a function of time.

Continuous Wavelet Transform:

- ▶ The continuous wavelet transform (CWT) is a generalization of the STFT.
- ▶ Allows for the analysis of non-stationary signals at multiple scales.
- ▶ Similar to the STFT, the CWT makes use of an analysis window to extract signal segments;
- ▶ Window is called a wavelet (or mother wavelet).
- ▶ Unlike the STFT, the analysis window or wavelet is not only convolved (translated), but dilated and contracted depending on the scale of activity under study.
- ▶ Wavelet dilation increases the CWT's sensitivity to long time-scale events
- ▶ Wavelet contraction increases its sensitivity to short time-scale events.

Plan Of Action:

We take three signals for analysis

1. Self-generated signal

The self-generated signal is a sinusoid of two different frequencies mixed with transient at self-decided points and with noise signal(gaussian white noise).

$$x = \cos(2\pi \cdot 150 \cdot t)(t \geq 0.1 \ \& \ t < 0.3) + \sin(2\pi \cdot 200 \cdot t)(t > 0.7);$$

2. Real life example- signal of quadratic chirp

3. Real life example- signal of Kobe's earth quake 1995

The graphs for CWT and STFT for all the three signals will be plotted and comparisons will be made which will help to realise the strengths and weakness, similarities and dissimilarities between wavelet transform and STFT; And to analyse which one is a more powerful tool. All these would be done with the help of **MATLAB** codes.

MATLAB Codes:

TRANSIENT SIGNAL GENERATION :

```
rng default;
dt = 0.001;
t = 0:dt:1-dt;
x = cos(2*pi*150*t).*(t>=0.1 & t<0.3)+sin(2*pi*200*t).*(t>0.7);
figure;
plot(t.*1000,x);
title("Base signal");
xlabel('Milliseconds');
ylabel('Amplitude');
figure;
x([222 800]) = x([222 800 ])+[-4 4];
plot(t.*1000,x);
title("signal with transient");
xlabel('Milliseconds');
ylabel('Amplitude');
figure;
Noise = 0.05*randn(size(t));
x = x+Noise;
plot(t.*1000,x);
title("signal with transient and noise");
xlabel('Milliseconds');
ylabel('Amplitude');
figure;
plot(t(184:264).*1000,x(184:264));
ylabel('Amplitude')
title('Transients')
figure;
plot(t(760:840).*1000,x(760:840));
ylabel('Amplitude')
xlabel('Milliseconds')
```

TRANSIENT SIGNAL STFT:

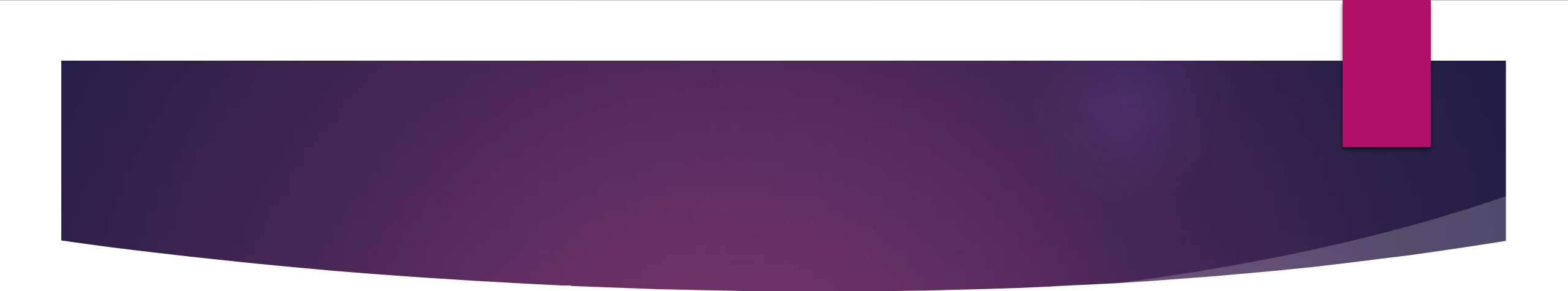
```
rng default;
dt = 0.001;
t = 0:dt:1-dt;
x = cos(2*pi*150*t).*(t>=0.1 &
t<0.3)+sin(2*pi*200*t).*(t>0.7);
x([222 800]) = x([222 800 ])+[-4 4];
Noise = 0.05*randn(size(t));
x = x+Noise;
[S,F,T] = spectrogram(x,50,48,128,1000);
helperCWTTimeFreqPlot(((abs(S)).^2),T,F,'surf','spectrogram of
the transient','milliSeconds','Hz')

function
helperCWTTimeFreqPlot(cfs,time,freq,PlotType,varargin)
params = parseinputs(varargin{:});
if strcmpi(PlotType,'surf',1)
    args = {time,freq,cfs};
    surf(args{:},'edgecolor','none');
    view(0,90);
    axis tight;
    shading interp; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
end
```

```
elseif strcmpi(PlotType,'contour')
    contour(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end

elseif strcmpi(PlotType,'contourf')
    contourf(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end

elseif strncmpi(PlotType,'image',1)
    imagesc(time,freq,abs(cfs).^2);
    colormap(parula(128));
    AX = gca;
```

```

        h = colorbar;
        h.Label.String = 'Power';
        if isempty(params.xlab) && isempty(params.ylab)
            xlabel('Time'); ylabel('Hz');
        else
            xlabel(params.xlab); ylabel(params.ylab);
        end
    end

    if ~isempty(params.PlotTitle)
        title(params.PlotTitle);
    end
end

%-----
-----
function params = parseinputs(varargin)

    params.PlotTitle = [];
    params.xlab = [];
    params.ylab = [];
    params.threshold = -Inf;

    if isempty(varargin)
        return;
    end

    Len = length(varargin);
    if (Len==1)
        params.PlotTitle = varargin{1};
    end

    if (Len == 3)
        params.PlotTitle = varargin{1};
        params.xlab = varargin{2};
        params.ylab = varargin{3};
    end
end
end

```

TRANSIENT CWT :

```
rng default;
dt = 0.001;
t = 0:dt:1-dt;
x = cos(2*pi*150*t).*(t>=0.1 &
t<0.3)+sin(2*pi*200*t).*(t>0.7);
x([222 800]) = x([222 800])+[-4 4];
Noise = 0.05*randn(size(t));
x = x+Noise;
[cfs,f] = cwt(x,1/dt,'amor');
helperCWTTimeFreqPlot(real(cfs),t,f,'surf','real part of CWT
of transient signal','milliseconds','Hz')
figure;
helperCWTTimeFreqPlot(imag(cfs),t,f,'surf','imaginary part of
CWT of transient signal','milliseconds','Hz')
figure;
helperCWTTimeFreqPlot((abs(cfs)).^2),t,f,'surf','power
spectrum of CWT of transient signal','milliseconds','Hz')
function
helperCWTTimeFreqPlot(cfs,time,freq,PlotType,varargin)
params = parseinputs(varargin{:});
```

```
if strncmpi(PlotType,'surf',1)
    args = {time,freq,cfs};
    surf(args{:},'edgecolor','none');
    view(0,90);
    axis tight;
    shading interp; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end

elseif strcmpi(PlotType,'contour')
    contour(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
```

```

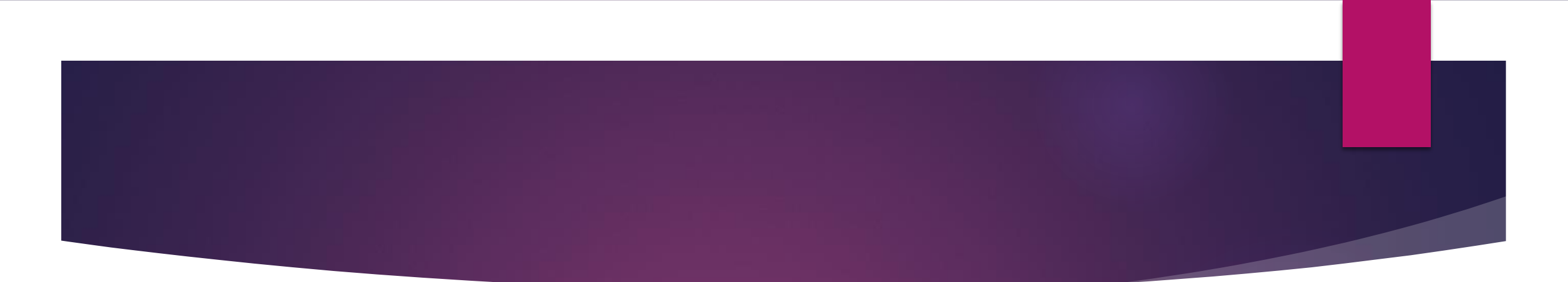
elseif strcmpi(PlotType,'contourf')
    contourf(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end

elseif strncmpi(PlotType,'image',1)
    imagesc(time,freq,abs(cfs).^2);
    colormap(parula(128));
    AX = gca;
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
end

if ~isempty(params.PlotTitle)
    title(params.PlotTitle);
end
end

%-----
-----
function params = parseinputs(varargin)

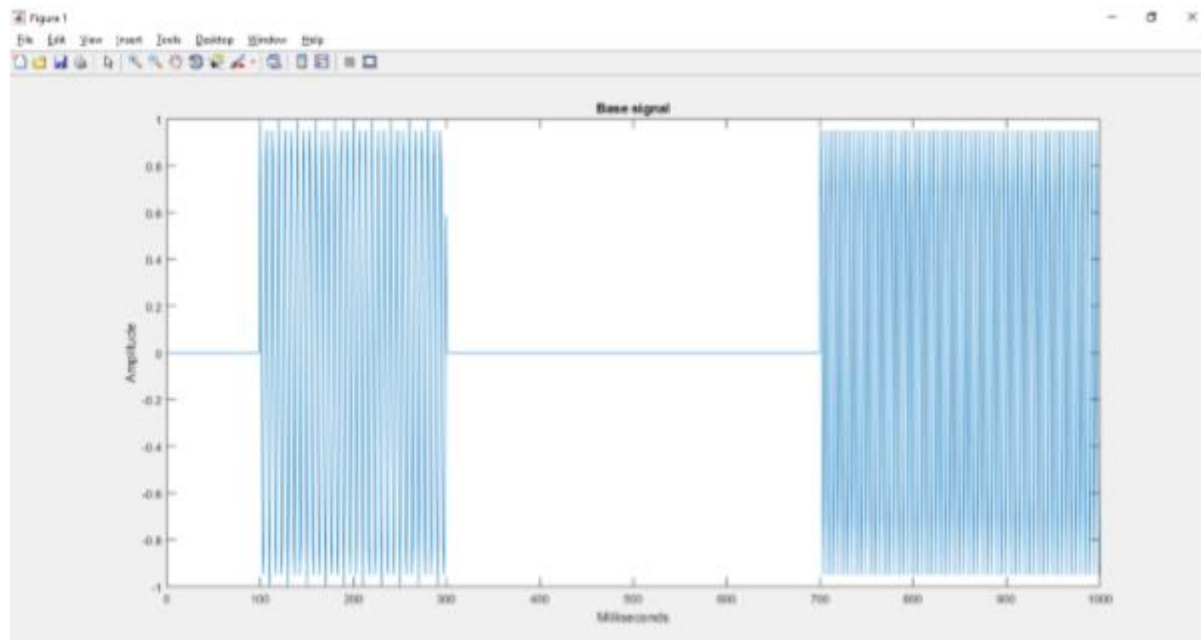
```



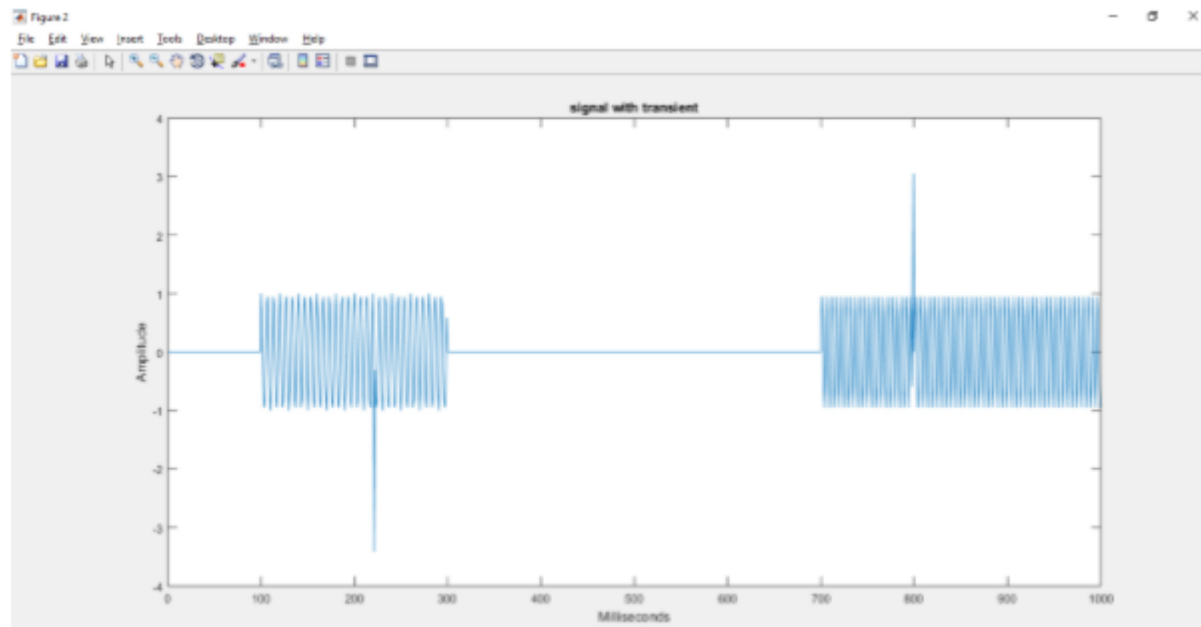
```
params.PlotTitle = [];  
params.xlab = [];  
params.ylab = [];  
params.threshold = -Inf;  
  
if isempty(varargin)  
    return;  
end  
  
Len = length(varargin);  
if (Len==1)  
    params.PlotTitle = varargin{1};  
end  
  
    if (Len == 3)  
        params.PlotTitle = varargin{1};  
        params.xlab = varargin{2};  
        params.ylab = varargin{3};  
    end  
end
```

Output Graphs:

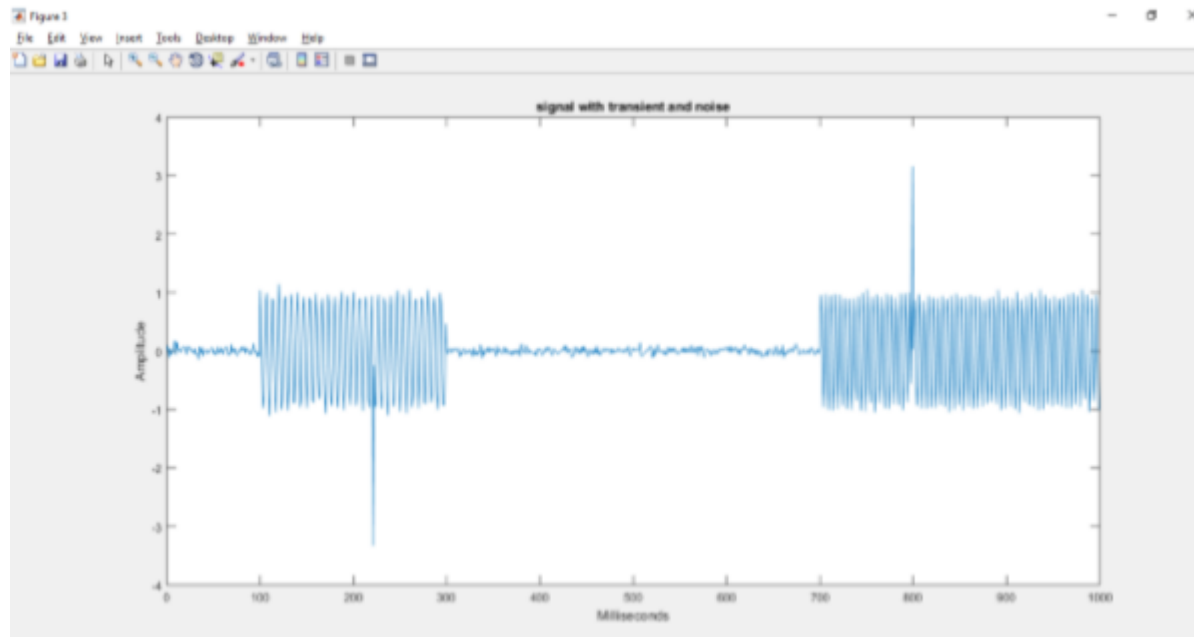
- Base signal generation:



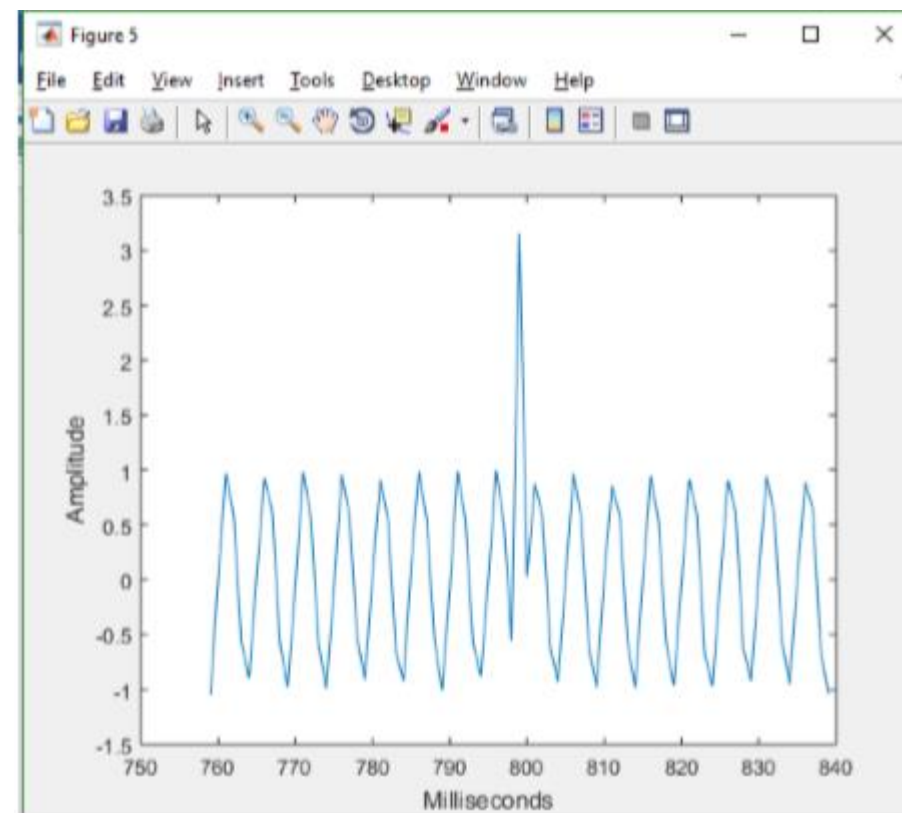
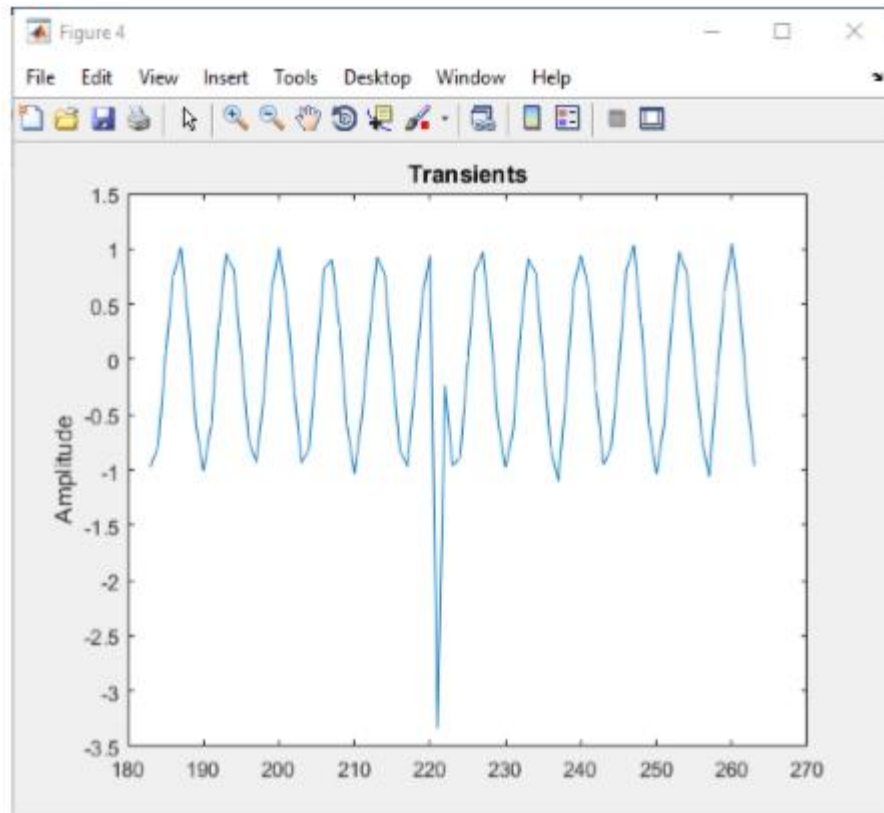
► Signal with transient



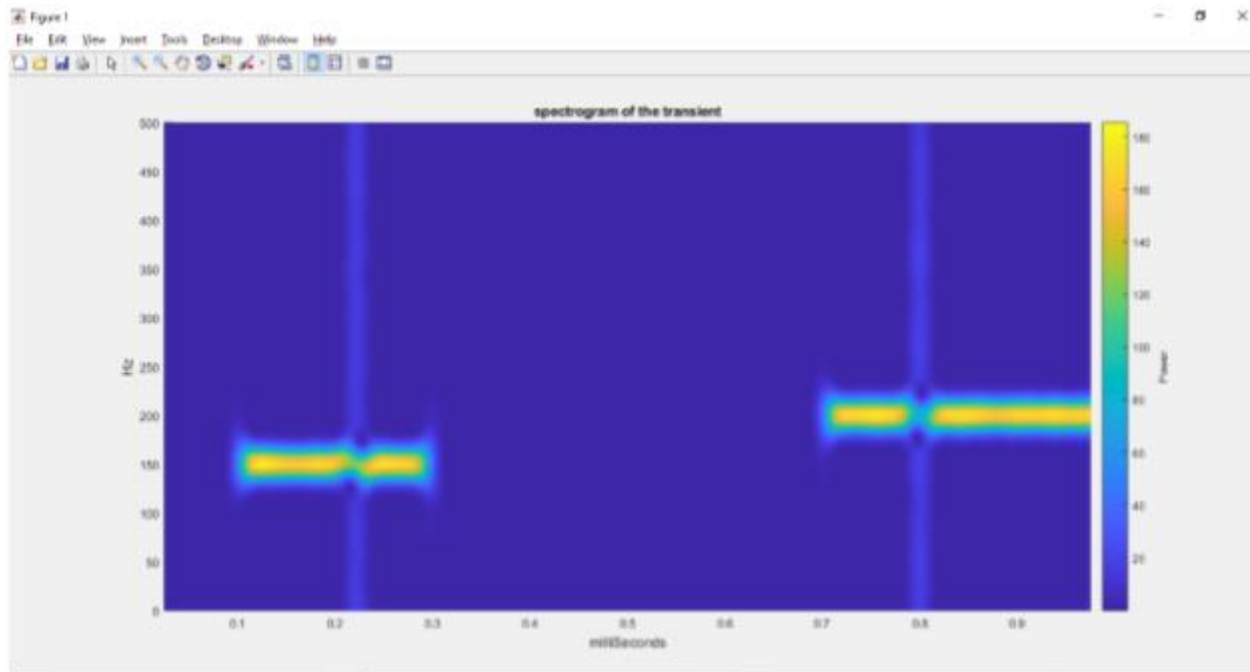
- Signal with transient and noise:



► Transients:

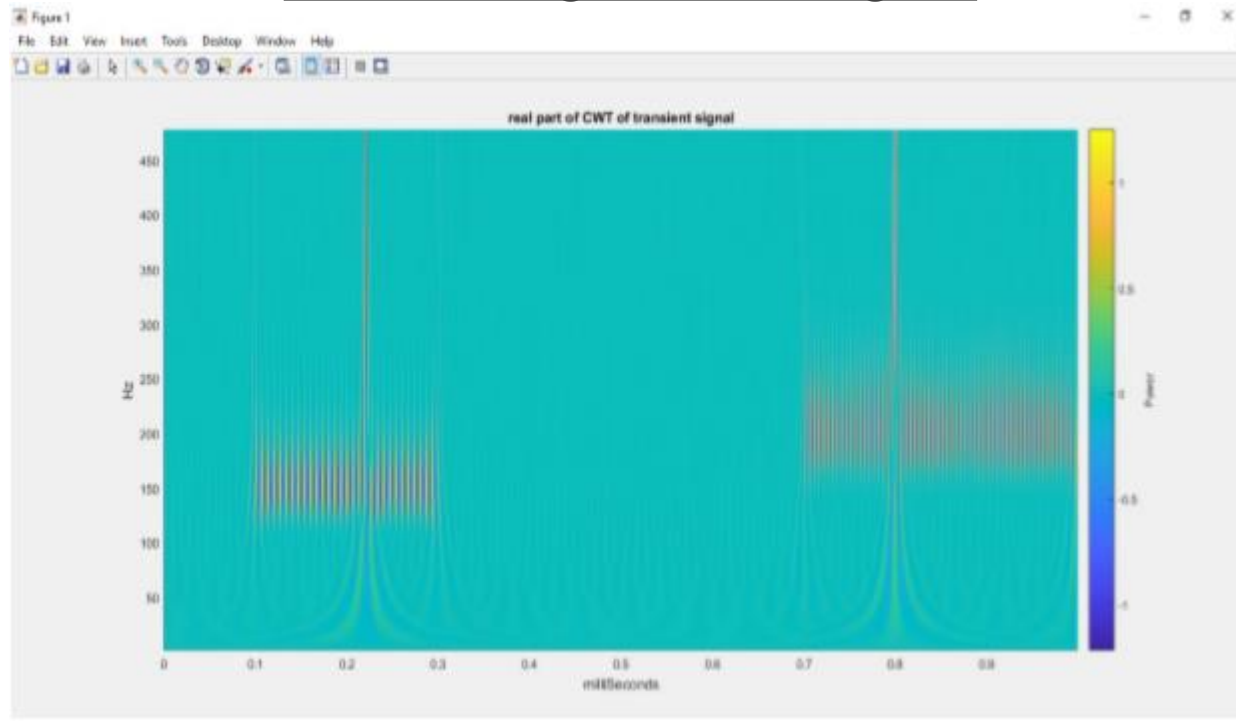


► STFT of the generated signal:

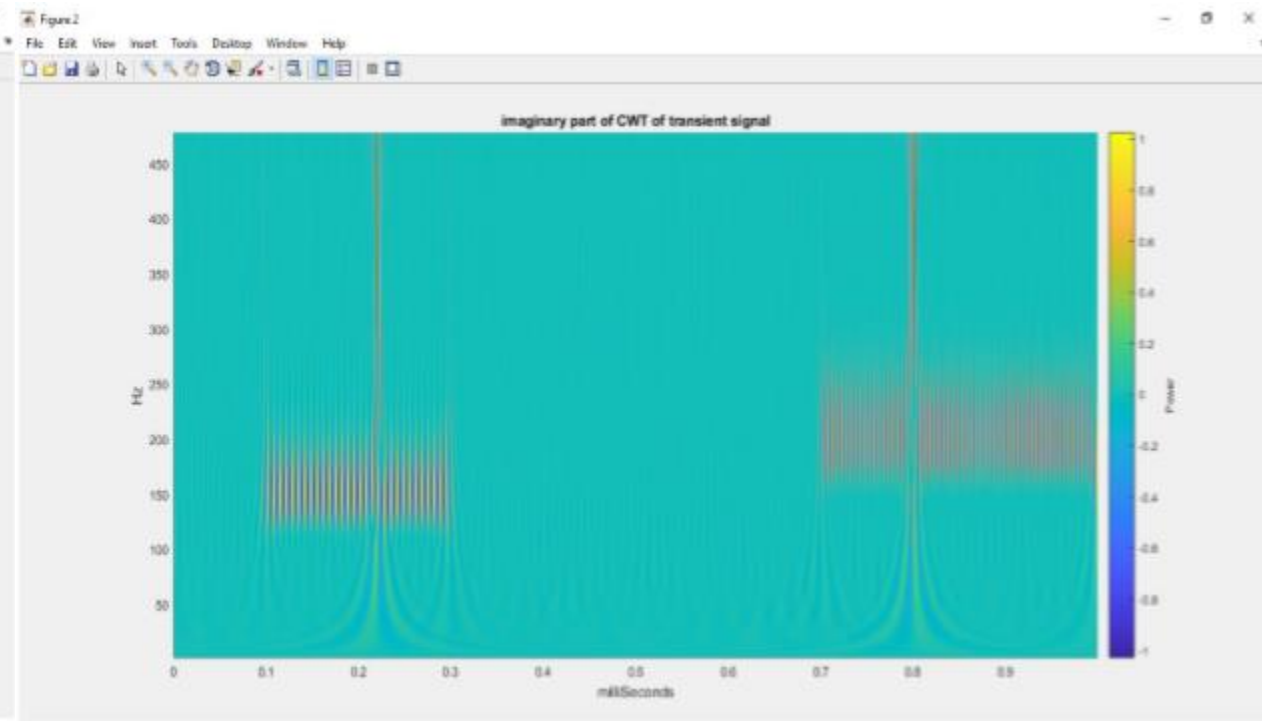


Spectrogram of the transient

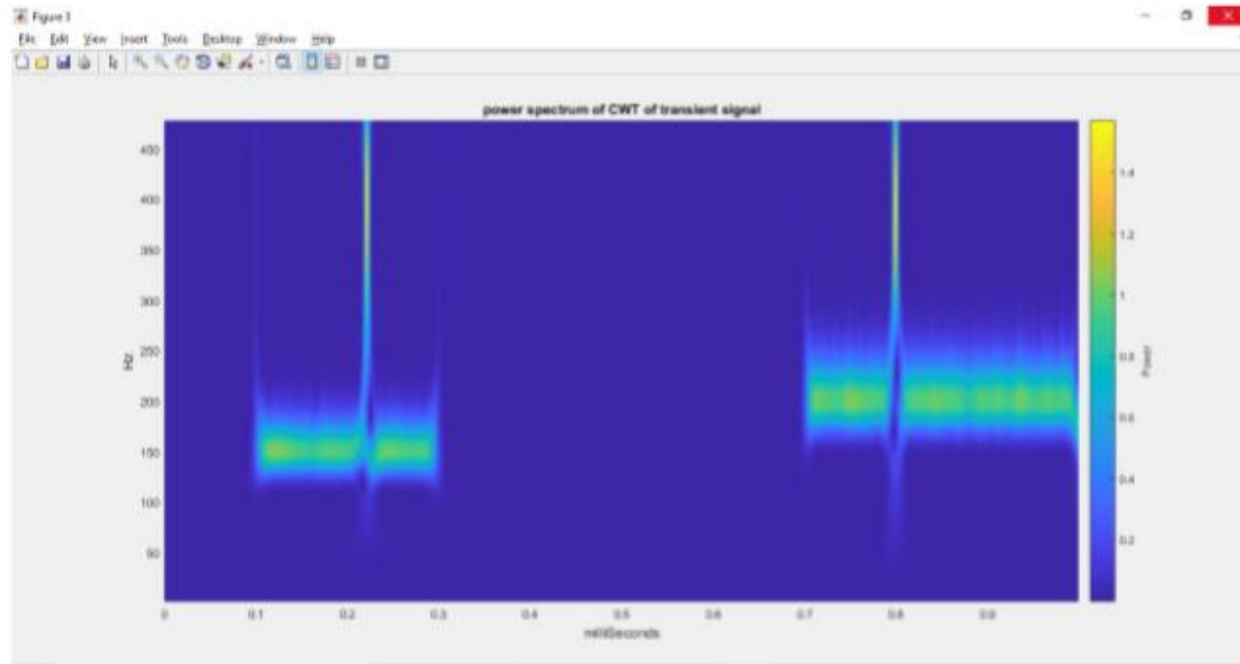
► CWT of the generated signal:



Real part of CWT of transient signal



Imaginary part of CWT of transient signal



Power spectrum of CWT of transient signal

Matlab Code:

CHIRP SIGNAL STFT AND CWT:

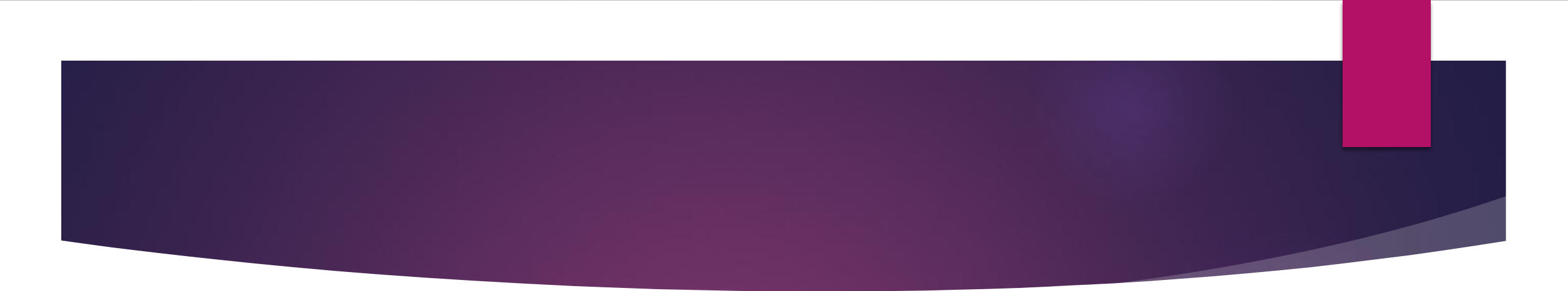
```
load quadchirp;
fs = 1000;
[S,F,T] = spectrogram(quadchirp,100,98,128,fs);
subplot(1,2,1)
helperCWTTimeFreqPlot(S,T,F,'surf','STFT of Quadratic
Chirp','Seconds','Hz')
[cfs,f] = cwt(quadchirp,fs,'WaveletParameters',[14,200]);
subplot(1,2,2)
helperCWTTimeFreqPlot(cfs,tquad,f,'surf','CWT of Quadratic
Chirp','Seconds','Hz')
function
helperCWTTimeFreqPlot(cfs,time,freq,PlotType,varargin)
params = parseinputs(varargin{:});

| if strncmpi(PlotType,'surf',1)
    args = {time,freq,abs(cfs).^2};
    surf(args{:},'edgecolor','none');
    view(0,90);
    axis tight;
    shading interp; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
end
```

```

elseif strcmpi(PlotType,'contour')
    contour(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
elseif strcmpi(PlotType,'contourf')
    contourf(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
elseif strncmpi(PlotType,'image',1)
    imagesc(time,freq,abs(cfs).^2);
    colormap(parula(128));
    AX = gca;
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
end
end

```



```
%-----  
-----  
function params = parseinputs(varargin)  
  
    params.PlotTitle = [];  
    params.xlab = [];  
    params.ylab = [];  
    params.threshold = -Inf;  
  
    if isempty(varargin)  
        return;  
    end  
  
    Len = length(varargin);  
    if (Len==1)  
        params.PlotTitle = varargin{1};  
    end  
  
    if (Len == 3)  
        params.PlotTitle = varargin{1};  
        params.xlab = varargin{2};  
        params.ylab = varargin{3};  
    end  
  
end
```

SEISMIC SIGNAL STFT AND CWT

(KOBES EARTHQUAKE 1995)

```
load kobe
DT = 1;
t = 0:DT:(numel(kobe)*DT)-DT;
[cfs,f] = cwt(kobe,1/DT,'amor');
subplot(1,2,1)
helperCWTTimeFreqPlot(cfs,t*1e6,f./1000,'surf','CWT of
kobe',...
    'Microseconds','kHz')
[S,F,T] = spectrogram(kobe,50,48,128,1/DT);
subplot(1,2,2)
helperCWTTimeFreqPlot(S,T.*1e6,F./1e3,'surf','STFT of
kobe',...
    'Microseconds','kHz')
function
helperCWTTimeFreqPlot(cfs,time,freq,PlotType,varargin)
params = parseinputs(varargin{:});

    if strcmpi(PlotType,'surf',1)
        args = {time,freq,abs(cfs).^2};
        surf(args{:},'edgecolor','none');
        view(0,90);
        axis tight;
        shading interp; colormap(parula(128));
        h = colorbar;
        h.Label.String = 'Power';
        if isempty(params.xlab) && isempty(params.ylab)
            xlabel('Time'); ylabel('Hz');
        else
            xlabel(params.xlab); ylabel(params.ylab);
        end
    end
```



```

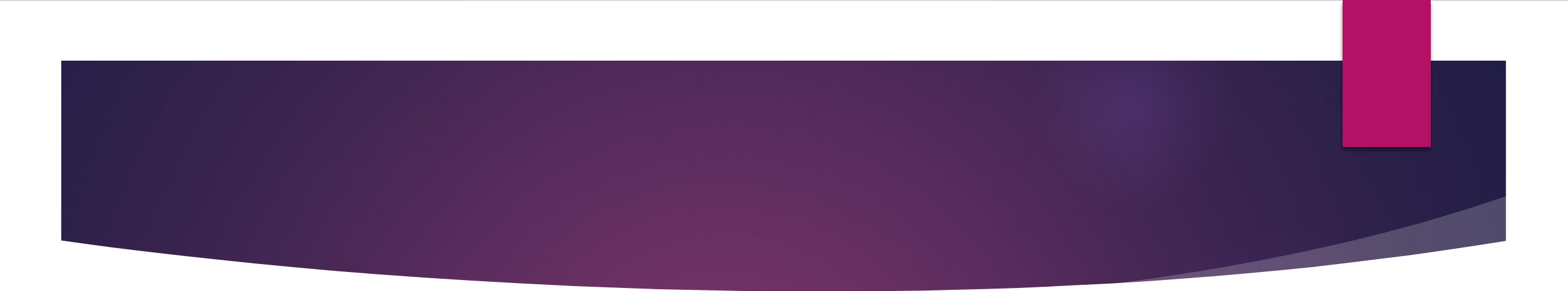
elseif strcmpi(PlotType,'contour')
    contour(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)

        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end

elseif strcmpi(PlotType,'contourf')
    contourf(time,freq,abs(cfs).^2);
    grid on; colormap(parula(128));
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end

elseif strncmpi(PlotType,'image',1)
    imagesc(time,freq,abs(cfs).^2);
    colormap(parula(128));
    AX = gca;
    h = colorbar;
    h.Label.String = 'Power';
    if isempty(params.xlab) && isempty(params.ylab)
        xlabel('Time'); ylabel('Hz');
    else
        xlabel(params.xlab); ylabel(params.ylab);
    end
end
end

```



```
if ~isempty(params.PlotTitle)
    title(params.PlotTitle);
end
```

```
end
```

```
%-----  
-----
```

```
function params = parseinputs(varargin)
```

```
    params.PlotTitle = [];  
    params.xlab = [];  
    params.ylab = [];  
    params.threshold = -Inf;
```

```
    if isempty(varargin)  
        return;  
    end
```

```
    Len = length(varargin);  
    if (Len==1)
```

```
        params.PlotTitle = varargin{1};
```

```
end
```

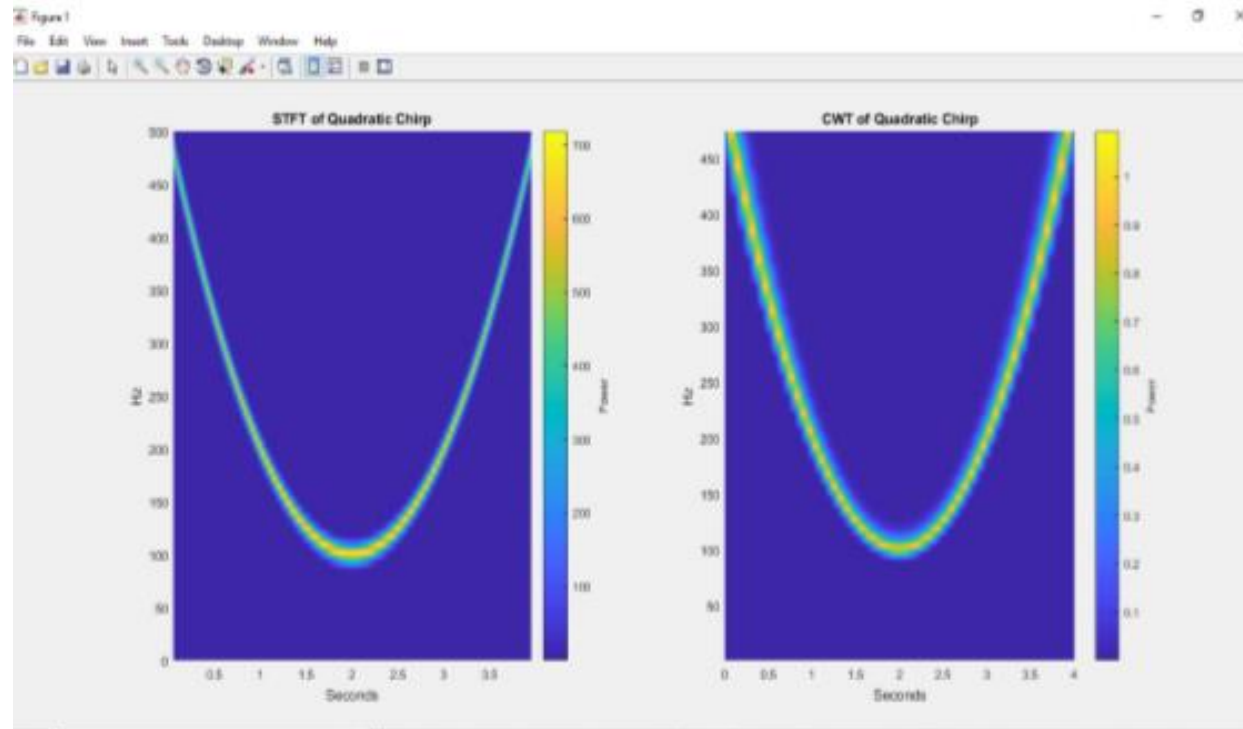
```
if (Len == 3)  
    params.PlotTitle = varargin{1};  
    params.xlab = varargin{2};  
    params.ylab = varargin{3};
```

```
end
```

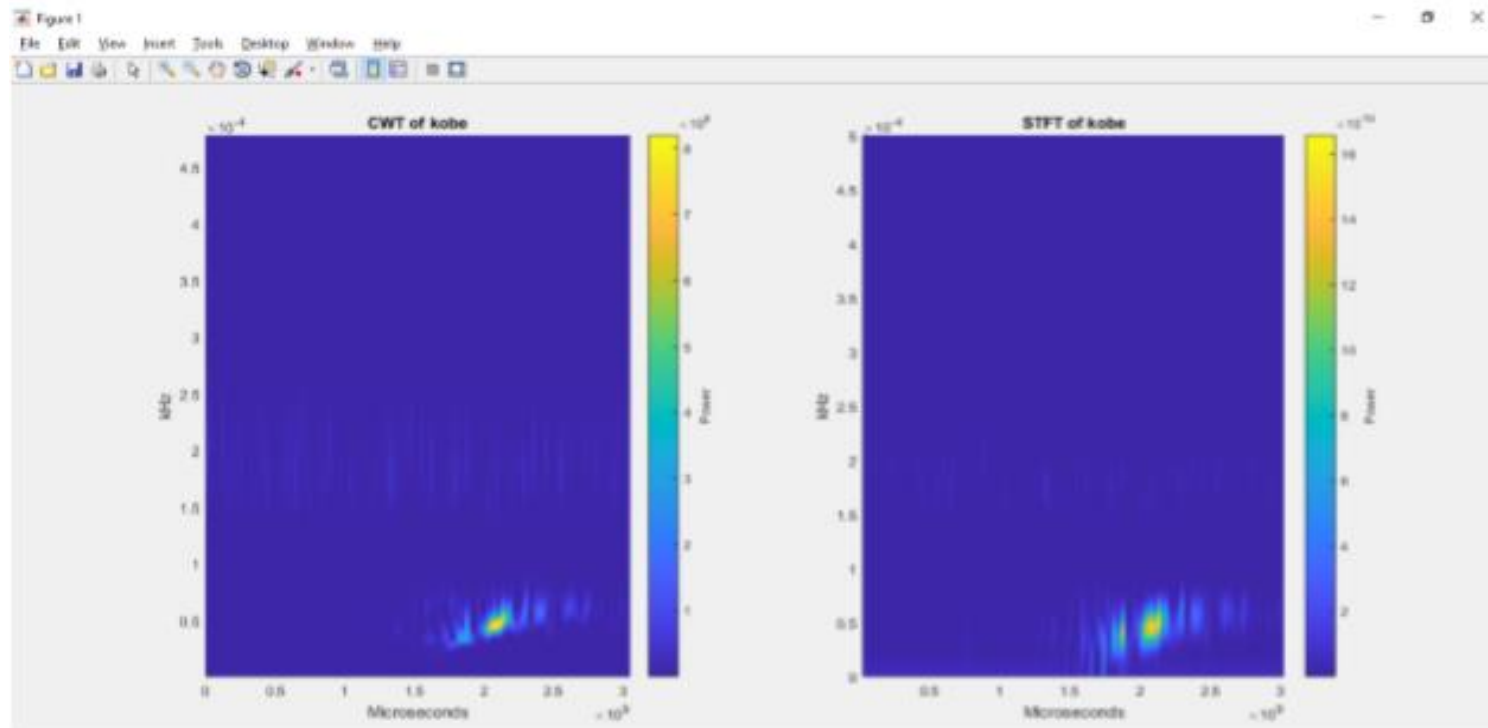
```
end
```

Output Graphs:

► CHIRP SIGNAL:



► 1995 KOBE SEISMIC SIGNAL :



Conclusion:

- ▶ It is found that STFT is easy to understand and use, while wavelet analysis has a wide application and can be used for both denoising and spectrum analysis. The salient feature as well as primary advantage of the wavelet transform is adaptive resolution. The wavelet transform gives a window that has constant relative error in the frequency domain rather than constant absolute error, at the expense of time resolution
- ▶ Unlike the TF or the STFT, the WT analyses a signal at different frequencies with different resolutions. It can provide good time resolution and relatively poor frequency resolution at high frequencies while good frequency resolution and relatively poor time resolution at low frequencies. Wavelet transform shows excellent advantages for the analysis of transient signals.