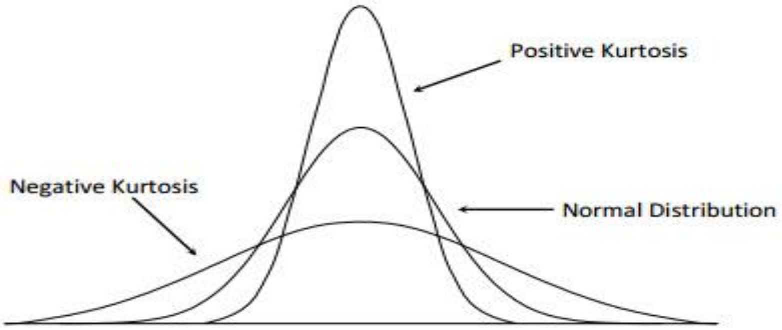
Kurtosis is calculated using the average and standard deviation

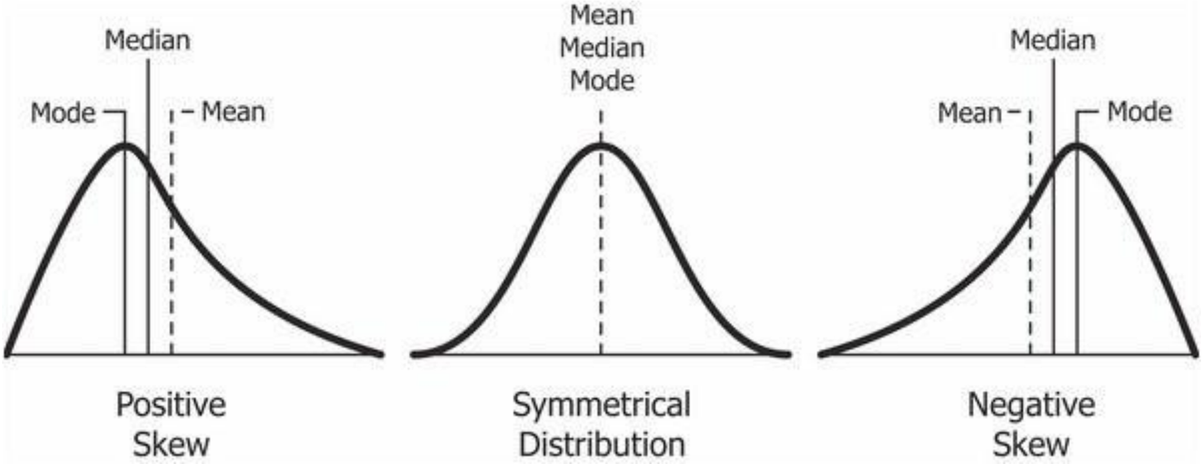


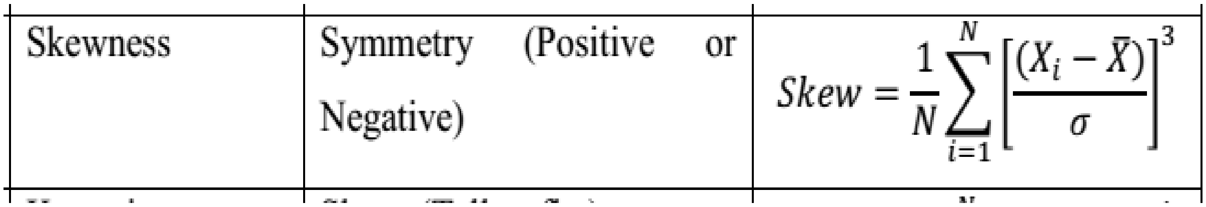
$$Kurtosis = \frac{1}{n}\sum_{i=1}^{n}(\frac{x_i - \bar{x}}{SD(x)})^4$$

```
clc
clear all
N=256*4;                % seconds=4;
numberOfchannels=23;
load('.mat');            % loading my record;



%%%checking whether .mat file is cell or matrix and if cell convert to matrix of one column%%%

for channel=1:numberOfchannels

  if(iscell(all_data))

     data=cell2mat(all_data(:,channel));     % converting each channel;


    else
       data=all_data;
    end

 data=data(1:floor(length(data)/N)*N,1);   % flooring my data;

%%%%%%reshape data to 1024 rows and 3692544/1024=3606 columns%%%%%

 new_shaped=reshape(data,N,length(data)/N);

%%%%%%loop on columns of the reshaped output from 1to its length%%%%%%

           for i=1:(length(data)/N)

%%%%%%have std on each column of the reshaped data and assign it to a matrix of one
row and columns = length of data/N %%%%%

      outPut(1,i)=kurt(new_shaped(:,i));

           end


    Kurtos(channel,:)=outPut; %output matrix for each channel
    Kurtos(isnan(Kurtos))=0;  %avoidance of any not a number (Nan) appearence
```

```
%%%%%%%%%%%%Kurtosis function implementation module%%%%%%%%%%%%%%

function y=kurt(x)      %function called kurt that takes x and return y
                        %where y is the kurtosis values
X=x;                               %to avoid mistakes of capitalization
averageofX=sum(X)/length(X);       %calculate average

stdofX=STD(x);                     %calculate standard deviation using the
                                   %previous implemented STD function

y=sum((((X-averageofX)/stdofX).^4))/length(X);   %kurtosis equation

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%STD function implementation module%%%%%%%%%%%%%%%%%%%%%%%%%


 function y=STD(x)


   averageofX=sum(x)/length(x);

   y=sqrt(sum(((x-averageofX).*(x-averageofX)))/(length(x)-1));


end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Skewness





Skewness	Symmetry (Positive or Negative)	$Skew = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{(X_i - \bar{X})}{\sigma}\right]^3$

```
%%%%%%%%%%%%%%%%%%%%%%implementation of Skewness module%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



function y=Pskew(x) %function called Pskew that takes x and return y
                    %where y is the kurtosis values

X=x;                %to avoid mistakes of capitalization

averageofX=sum(X)/length(X);   %calculating average

stdofX=STD(x);                 %calling STD module

y=sum((((X-averageofX)/stdofX).^3))/length(X); %calculating skewness

end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%applying on data%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%




clc
clear all
N=256*4;                % seconds=4;
numberOfchannels=23;
load('.mat');            % loading my record;



%%%checking whether .mat file is cell or matrix and if cell convert to matrix of one column%%%

for channel=1:numberOfchannels

  if(iscell(all_data))

     data=cell2mat(all_data(:,channel));     % converting each channel;


    else
       data=all_data;
    end

 data=data(1:floor(length(data)/N)*N,1);   % flooring my data;

%%%%%%reshape data to 1024 rows and 3692544/1024=3606 columns to get windows%%%%%

 new_shaped=reshape(data,N,length(data)/N);

%%%%%%loop on columns of the reshaped output from 1to its length%%%%%%

           for i=1:(length(data)/N)

%%%%%%have std on each column of the reshaped data and assign it to a matrix of one
row and columns = length of data/N %%%%%

      outPut(1,i)=Pskew(new_shaped(:,i));

           end


   Skewness (channel,:)=outPut; %output matrix for each channel
   Skewness(isnan(Skewness))=0;  %avoidance of any not a number (Nan) appearance
```