Middle East Technical University

Department of

Electrical and Electronics Engineering

EE430

Digital Signal Processing

Term Project – Part I

Durmuş Umutcan Uğuz

Contact Info: umutcanuguz@gmail.com
+90 539 764 0717

SPECTROGRAM VIEWER

Introduction

In this report, it is aimed to represent the basic work of the preliminary part of the term project. Since it is only the preliminary part and it includes a basic part, this report does not go into the full detail of the concepts used in this part. More detailed report will already be given when the whole project is complete.

Spectrogram

Spectrograms enable us to investigate the frequency representation of a time signal more accurately. Since most of the signals are not stationary, investigating them just by taking their DFT is not accurate. A more accurate method is to investigate them by small time intervals. On the x-axis of the spectrogram one can see each time interval. Therefore, vertical line at that specific x value is actually DFT of a much smaller part of the original signal. By this method, we can observe the frequency components closely at each time.

MATLAB Function

MATLAB function I have written to implement a spectrogram is represented within this report. Construction of the function is quite simple and does not require a detailed explanation to someone who is familiar with MATLAB environment. Therefore, only necessary small explanations within the code will be given as comments. Moreover, the inputs will be specified hereby.

There are four inputs which will be given to the function by the user.

soundFile: the audio signal which will be investigated.

winLength: the length of the window, representing the length of any small time length by which we investigate the whole signal.

winShift: the shift between each consecutive small time interval.

windowType: the type of the window which will be used during the multiplications. For this input there are predetermined options. These options are represented below.

'rectwin'
'hann'
'hamming'
'gausswin'
'blackman'
'chebwin'
'flattopwin'
'kaiser'
'taylorwin'
'tukeywin

Window Types

Windowing is an important part of the spectrogram. Different window types may have advantages and disadvantages in some cases. The further analysis of windowing is not within the scope of this part. Different types of windowing may be investigated further. However, as it is not required here, I will simply use Gaussian type windows. These windows have α constant determining the spread of the function. It is set to 2.5 in MATLAB, if anything is not specified.

Example Outputs of Function

In order to get some outputs I have used a sample from lecture notes. The sample "scholars.wav" is taken from the ODTUClass which was uploaded by the instructors. Now, I represent some outputs from function. The definitions and the command line for the function are given.

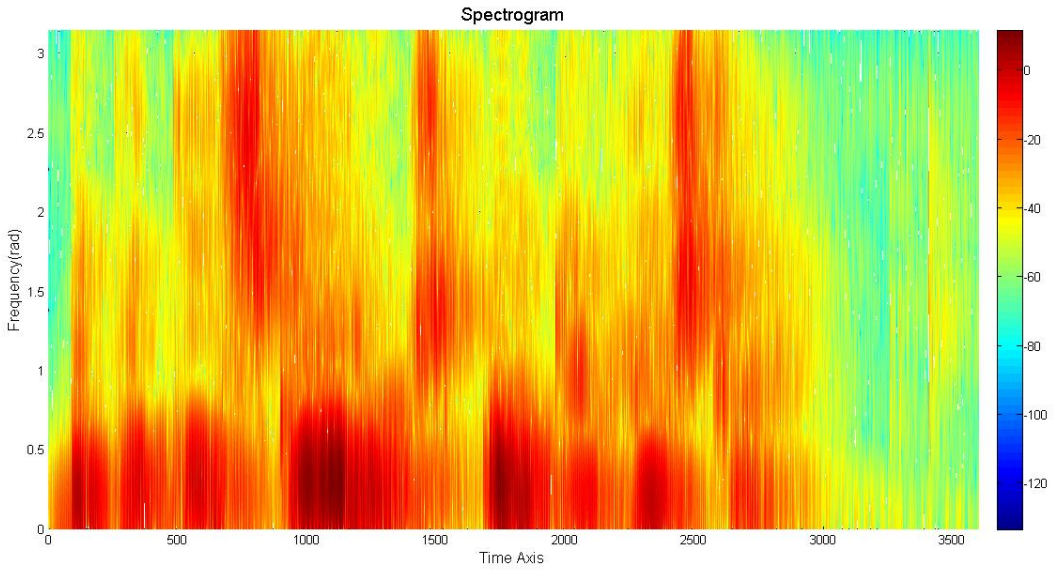

Figure 1 -- projeDSPpart1(scholars,25,10,'gausswin')

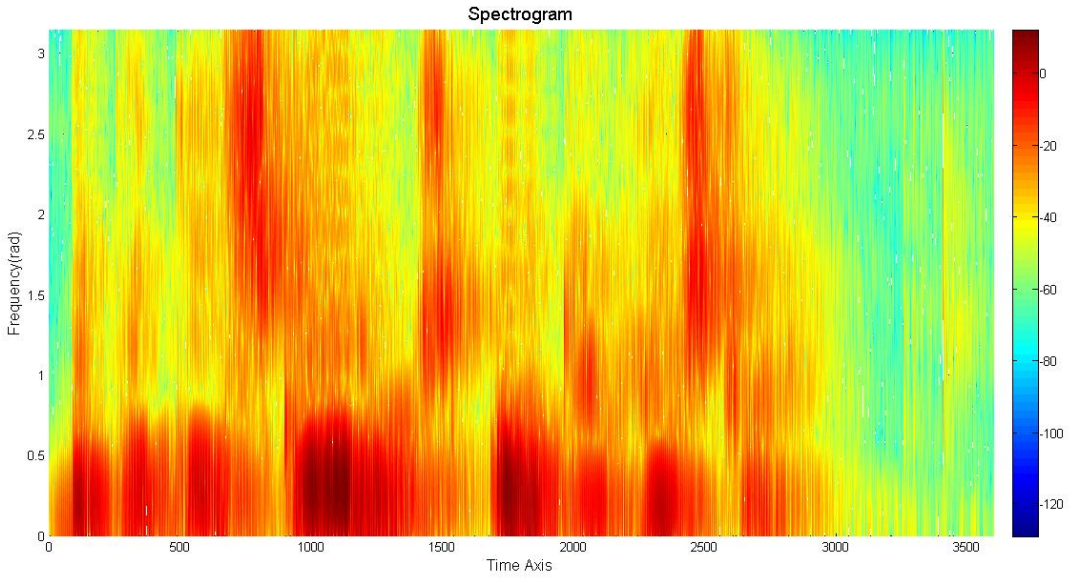

Figure 2 -- projeDSPpart1(scholars,25,10,'hamming')

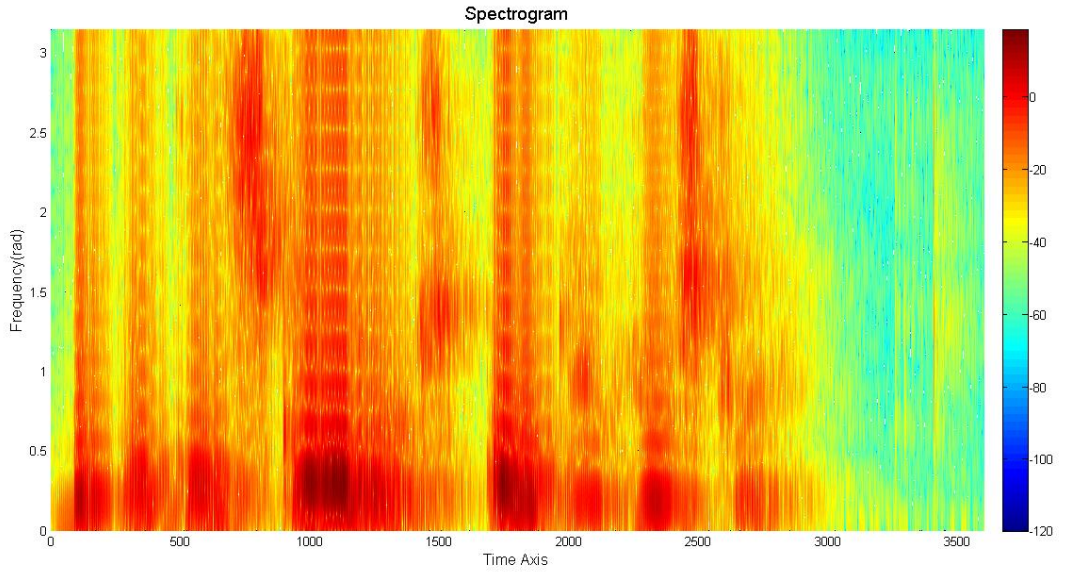

Figure 3 -- projeDSPpart1(scholars,25,10,'rectwin')

The function itself is represented below.

```
function projeDSPpart1(soundFile,winLength,winShift,windowType)

    %%% check if it is necessary to take transpose
    if(size(soundFile,1) > 5)
        soundFile = soundFile';
    end

    win = 0;     % initial blank window

    %%% check for the length errors
    if( length(soundFile) < winLength )
    soundFile = [soundFile zeros(1,winLength-length(x))];
    end

    if( winShift > winLength )
    disp(' Window Shift length cannot be greater than the length of the
window ');
    end



    %%% Check for the window type selected from the user
    if( strcmp(windowType,'rectwin')) %#ok<*STCMP>
        win=ones(1,winLength);
    end
    if( strcmp(windowType,'hann')) %#ok<*STCMP>
        win=hann(winLength)';
    end
    if( strcmp(windowType,'hamming'))
        win=hamming(winLength)';
    end
    if( strcmp(windowType,'gausswin'))
        win=gausswin(winLength)';
    end
    if( strcmp(windowType,'blackman'))
        win=blackman(winLength)';
    end
    if( strcmp(windowType,'chebwin'))
        win=chebwin(winLength)';
    end
    if( strcmp(windowType,'flattopwin'))
        win=flattopwin(winLength)';
    end
    if( strcmp(windowType,'kaiser'))
        win=kaiser(winLength)';
    end
    if( strcmp(windowType,'taylorwin'))
        win=taylorwin(winLength)';
    end
    if( strcmp(windowType,'tukeywin'))
        win=tukeywin(winLength)';
    end
    if( win == 0 )
        disp(' Please enter a valid window type ');
    end
```

```
        %%% unimportant initials
        %shorttft = zeros(3,winLength*100);
        ini = 1;    %%% initial point of the window
        i=1;        %%% counter of vertical lines

        %%% fill the matrix representing STFTs
    while( ini + winLength -1 <= length(soundFile) )

        shorttft(i,:) = abs(fft(soundFile( ini : ini+winLength-1
).*win,winLength*100));
        ini = ini + winShift;
        i = i+1 ;

    end

    %%% define the axis names
    time = 0 : size(shorttft,1)-1;
    freq = linspace(0,pi,winLength*50);

    %%% take dB
shorttftDB=(20*log10(shorttft))';
figure
    %%% plot a surface plot
surf(time,freq,shorttftDB(1:winLength*50,:))
shading interp
    %%% change the view angle
view(2)
xlabel('Time Axis','Fontsize',12)
ylabel('Frequency(rad)','Fontsize',12)
title('Spectrogram','Fontsize',14)
colorbar
ylim([0 pi]);
xlim([0 size(shorttft,1)]);
```