Lab 10

**Discrete Time Fourier Transform (DTFT)**

**OBJECTIVES**

• To study and implement the discrete time Fourier transform (DTFT).

• To learn system analysis using discrete time Fourier transform (DTFT).

**INTRODUCTION**

The discrete-time Fourier transform, DTFT, provides the frequency-domain representation for a discrete-time signal. The DTFT for discrete-time signals is the counterpart to the continuous-time Fourier transform for continuous-time signals.

The discrete-time Fourier transform (DTFT), *X*(Ω), of a discrete time signal *x*[*n*] is defined as

In general *X*(Ω) is a complex function of the real variable Ω and can be expressed in the rectangular form

where and are, respectively, the real and imaginary parts of 𝑋(Ω), and are real functions of Ω. 𝑋(Ω ) can alternately be expressed in the polar form

The quantity |𝑋(Ω)| is called the magnitude function and the quantity ∠𝑋(Ω) is called the phase function

The DTFT *X*(Ω) is a periodic continuous function in Ω with a period 2*π*. The DTFT satisfies several useful properties that are often utilized in several applications.

**Prelab Exercises**

For the filter impulse response *h*[*n*] = 0.25 *δ* [*n*] + 0.5 *δ* [*n*-1] + 0.25 *δ* [*n*-2] find the following:

1. The DTFT of *h*[*n*]. Simplify your answer to the form .
2. Plot the magnitude |*H*(Ω)| and the phase versus Ω for the range [-3π to 3π].
3. Find the output *y*[*n*] of the filter for the input *x*[*n*] = 2 cos(0.2π*n*).

**lab Exercises**

**Task 1: Using the DTFT Definition to Find *H*(Ω)**

1. Use the DTFT definition to calculate *H*(Ω) of the impulse response *h*[*n*] for the two filters shown below

*h*1[*n*] = 0.25 *δ* [*n*] + 0.5 *δ* [*n*-1] + 0.25 *δ* [*n*-2]

*h*2[*n*] = -0.25 *δ* [*n*] + 0.5 *δ* [*n*-1] - 0.25 *δ* [*n*-2]

1. For both filters, plot the magnitude |*H*(Ω)| and the phase vs frequency Ω for the range [-3π to 3π].
2. Do you see periodicity in the spectra? If so, what is the period?
3. What type of filters are *h*1[*n*] and *h*2[*n*], in term of lowpass, highpass, bandpass, or notch filter?
4. Let us define the cutoff frequency as the frequency at which the magnitude |*H*(Ωc)| drops or rises to 50% of its maximum value. What are the cutoff frequencies for the two filters? Express these cutoff frequencies in term of Hertz if the sampling rate is 8000 Hz. What will the cutoff frequencies be if the sampling rate is 12,000 Hz.

**Task 2: Finding *H*(Ω) by Testing the System with Sinusoidal Inputs.**

We will find the frequency response *H*1(Ω) of the filter *h*1[*n*] by applying a sinusoidal input of different frequencies to the filter and record the maximum amplitudes of the sinusoidal outputs. The magnitude |*H*1(Ω)| will be the amplitude of the output signal divided by the amplitude of the input signal, *A*o/*A*i. The six frequencies of the sinusoidal inputs are

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Ω | 0 | 0.2 π | 0.4π | 0.6π | 0.8π | π |

Your first step is to create the input signal *x*[*n*] in Matlab for *n* = 0 … 50. For example, The following code creates a sinusoidal signal *x*[*n*] =10cos( 0.2π*n*)

>> omg = 0.2\*pi;

>> n = 0:1:50;

>> x = 10\*cos(omg\*n);

1. Filter each of the six sinusoidal signals of different frequencies with the filter

*h*1[*n*] = 0.25 *δ*[*n*] + 0.5 *δ*[*n*-1] + 0.25 *δ*[*n*-2],

using the standard MATLAB function “**filter( )**”.

>> h = [0.25 0.5 0.25];

>> FiltSig = filter(h, 1, x);

>> plot(FiltSig);

1. Calculate the gain *G*, |*H*1(Ω)|, by dividing the amplitude of the output signal by the amplitude of input signal. You can find the peak amplitude of the output signal by plotting the output signal, *FiltSig*. Repeat the above steps to calculate the gain for the six sinusoidal inputs with the six different frequencies mentioned above. You will get six data points for the gain at the six different frequencies, see table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Frequency (Ω) | Input Amplitude (*A*i) | Output Amplitude (*A*o) | Gain |*H*1(Ω)| = *A*o / *A*i |
| 0 | 10 |  |  |
| 0.2π | 10 |  |  |
| 0.4π | 10 |  |  |
| 0.6π | 10 |  |  |
| 0.8π | 10 |  |  |
| π | 10 |  |  |

1. Plot the magnitude |*H*1(Ω )| vs Ω using the six data points calculated in the above task. Is this a lowpass, highpass, bandpass, or a notch filter?
2. Repeat the above steps for the filter *h*2[*n*]= -0.25 *δ*[*n*] + 0.5 *δ*[*n*-1] - 0.25 *δ*[*n*-2].

|  |  |  |  |
| --- | --- | --- | --- |
| Frequency (Ω) | Input Amplitude (*A*i) | Output Amplitude (*A*o) | Gain |*H*2(Ω)| = *A*o / *A*i |
| 0 | 10 |  |  |
| 0.2π | 10 |  |  |
| 0.4π | 10 |  |  |
| 0.6π | 10 |  |  |
| 0.8π | 10 |  |  |
| π | 10 |  |  |

1. Compare the plots of |*H*(Ω )| for the two filters in this task with the corresponding plots of the two filters in task 1.

**Task 3: Listening to Audio Materials Filtered by *H*(Ω).**

1. Download the wave file *sentence.wav* from the course website to the current directory of MATLAB. The wave file has a speech signal. Use *audioread* function to load the file into Matlab. You can name the downloaded speech signal as OrgSig.

>> [OrgSig, fs] = audioread(‘sentence.wav’);

1. Resample the speech signal from its original sampling rate 44,100 to 8000 samples/sec using the resampling function in Matlab**,**

>> ResSig = resample(OrgSig, 8000, fs);

The resampling function in Matlab will first filter out all frequencies above 4 kHz to avoid aliasing and then down-sample the signal to 8000 Hz (down-sampling with decimation). Play the original signal and the resampled signal.

>> sound(OrgSig, fs);



>> sound(ResSig, 8000);

Do you hear a difference in the audio quality between the original sentence “OrgSign” and the resampled sentence “ResSig”?

Why do you need to resample the input signal from 44100 to 8000 samples/sec.

1. Filter the speech signal ResSig with *h*1[*n*] as shown below

>>FiltSig = filter([0.25 0.5 0.25], 1, ResSig);

Play the audio output of the filter “FiltSig”. Is there a difference in audio quality between the input signal to the filter “ResSig” and the output signal “FiltSig”? Describe any difference on the sound quality and the reason for this difference. Plot both “ResSig” and “FiltSig” on the same plot as follow.

>> plot(ResSig);

>> hold on;

>> plot(FiltSig);

Can you explain the impact of the filter on the signal in the time domain?

What is the cutoff frequency in Hz of the digital filter *h*1[*n*] for the sampling rate 8000 of the audio signal?

1. Use the spectrogram function in Matlab to plot the spectrum of the input signal to the filter “*ResSig*” and the output signal of the filter “*FiltSig*”. Compare the spectrum of the input and output signals.

>> spectrogram(ResSig,256,64,8000,'yaxis');

>> figure;

>> spectrogram(FiltSig,256,64,8000,'yaxis');

The spectrogram calculates and plots the discrete Fourier transform (DFT) for time segments of the signal. It will use the first 256 samples and calculates the 256 discrete Fourier transform, and then move to the sample located at 256-64 which is at 192 and use the next 256 samples [192 to 448] to calculate the DFT, and so forth, to the end of the signal. The *x*-axis represent time, the *y*-axis represents frequency. The color represents magnitude, where blue represents week magnitude and yellow strong magnitude.

1. Repeat steps 3 to 4 for the filter *h*2[*n*].

**LAB REPORT FORMAT**

Please use D2L to submit the lab report electronically. In case there is a technical issue with D2L, please email the lab report to the course instructor before the cutoff time. The lab report should include: (1) Title Page; (2) Introduction; (3) Results (Include programs, answers, and figures to all steps in Lab Procedure); (4) Conclusion/Discussion.