UNIVERSITY COLLEGE DUBLIN

SCHOOL OF ELECTRICAL & ELECTRONIC ENG.

EEEN30110 SIGNALS AND SYSTEMS

Experiment 3SS3

Fourier Transform and System Theory

1. Objective

To investigate applications of the Fourier Transform and filtering.

2. Background Information:

The fundamental value of the Fourier transform in *System Theory* is that it permits a very significant simplification of our description of the action of a linear time-invariant system. The Fourier transform of the output is equal to the transfer function multiplied by the Fourier transform of the input, i.e.

$$G(j\omega) = H(j\omega)F(j\omega).$$

The fundamental value of the Fourier transform to *Signal Theory* is that we may interpret the Fourier transform of a signal in terms of the frequency content of the signal, i.e. the signal contains significant frequency content at or close to the frequency ω_1 if and only if the Fourier transform of the signal evaluated at this frequency, namely $F(j\omega_1)$, is decidedly non-zero.

Combining these two features of the Fourier transform we find that if the input of a linear time-invariant system has very little frequency content at (or close to) the frequency ω_1 then the same is true of the output. If on the other hand the input of the system has significant frequency content at (or close to) the frequency ω_1 then the same *may* be true of the output. Precisely, the same *is* true of the output unless the transfer function evaluated at this frequency $H(j\omega_1)$ is very nearly zero, that is to say the modulus of this number, $|H(j\omega_1)|$, is very small. This permits us to interpret the function $|H(j\omega)|$, the so-called magnitude response of the system, as describing the frequency response of the system, i.e. how it reacts to different frequencies and specifically which frequencies it suppresses (or stops) and which frequencies it allows through (or passes) and possibly which frequencies it even amplifies (or at which it resonates).

Electrical/electronic systems which are specifically designed such that they pass certain frequencies and stop others are called electrical filters (or just filters). In this laboratory we will investigate the action of a particular filter called the *twin tee*. The circuit is shown in Figure 1. In this figure resistance R2 equals R/2 and capacitance C2 equals 2C. The output is the voltage v_{out} (not shown) between the output pin of the op amp and earth, otherwise simply known as the output voltage of the op amp.

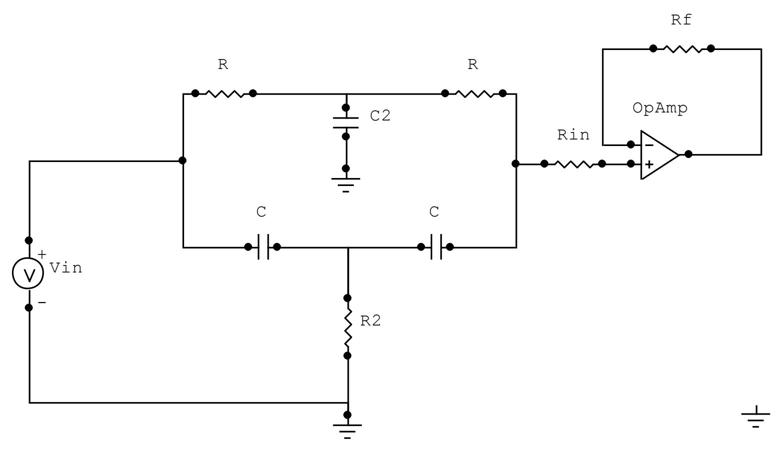

Figure 1: Twin Tee Filter

The resulting system with input v_{in} and output equal to the output voltage v_{out} of the op amp can be shown to have transfer function:

$$H(j\omega) = \frac{K\left((j\omega)^2 + \left(\frac{1}{RC}\right)^2\right)}{(j\omega)^2 + \left(\frac{4}{RC}\right)(j\omega) + \left(\frac{1}{RC}\right)^2}$$

for suitable gain K. The purpose of the Twin Tee filter is to block frequencies in a relatively narrow range of frequencies around a particular frequency. In other words $|H(j\omega)|$ is to be very low for frequencies close to a particular frequency ω_1 and is to be reasonably high and reasonably constant for all other frequencies.

The file AudioData.wav on the Blackboard page for the module is a wav file. It contains sample data of a speech signal to which I have added a jamming signal of significantly greater amplitude. The object in this laboratory is to suppress the jamming signal sufficiently so that the original becomes audible (assuming you have the facilities in your version of Matlab to hear it). You should be able to download the file either to your Matlab work space or to some other location on your machine. You may then import the file into Matlab by employing the Matlab command **wavread**. Specifically

```
>> [y,fs,bits]=wavread('AudioData');
```

will import the file, provided you placed it in your Matlab workspace. Otherwise you will have to specify the path which wavread must employ to find the file. The resulting vector y contains the actual data samples. The number fs reports the sampling frequency, i.e. the samples are uniformly spaced with time gaps between samples of 1/fs sec. To listen to the audio data you may employ the command **sound**. Specifically

```
>> sound(y,fs)
```

As the machines in this room have no external speakers you will need to plug in a set of headphones. Any headphones will do. Take care, particularly if employing insertion-style headphones. Loud signals may cause damage. Test the volume with headphones at a distance from your ears first. Unfortunately, the delivery of Matlab over the network has affected the audio facilities in Matlab and you may find that they do not function on your machine.

The Matlab command **lsim** can be employed to find the response of a linear time-invariant system of a given transfer function to a specified input signal, where that signal is specified by samples and the corresponding sample times. Specifically, if the system variable associated with the transfer function H has been created using the **tf** command as usual, then

```
>>  [yout,xout]=lsim(H,u,t);
```

returns in the vector yout the samples of the response of the system at the times specified by the input vector t, where the vector u is the vector of samples of the input evaluated at these same times t. For a number of reasons this is not the best way of using Matlab for the problem at hand, but it will do.

As a preliminary optional exercise to prepare you for the problem below generate a vector comprising 50000 uniformly-spaced samples of a sinusoidal signal of amplitude 0.5 and with any audible frequency you desire. The sampling frequency f_s , which determines the time gaps between the samples, will need to be considerably more than twice the signal frequency in practice. Let it be about 25 times this frequency. Test your headphones by playing the resulting signal using the **sound** command, but take note of the safety warning above. The sound command may not work on your machine. Plot the signal to see, at least, that it does indeed look approximately sinusoidal. Estimate the frequency of the signal from this plot. By employing the command **fft** plot the magnitude spectrum of the signal and confirm that it has the expected shape. From the magnitude plot make a second estimate of the frequency of the signal. To do this you must note that the **fft** calculates the approximate Fourier transform at the frequencies nf_s/N Hz, where f_s is the sampling frequency (in Hz) and N is the number of data points or samples. To properly investigate both the time and the frequency domain plots you will also need to use the *zoom in* feature in the Matlab figure created by the plot command. The default setting for *zoom in* is “unconstrained zoom”, which will not give you the best results. In the **Tools>Options** menu of the figure window you are given the option to select “horizontal zoom”, which is the more appropriate setting on this occasion. In this setting you may zoom in without changing the vertical axis.

3. Problems

1. The Twin Tee filter completely blocks a particular positive frequency. Find an expression for this frequency in terms of the parameters K, R and C of the transfer function.

2. Import the file AudioData.wav into Matlab. What is the sampling frequency?

3. The Audio file has been corrupted by the addition of a relatively large amplitude jamming sinusoid of a certain frequency. If possible confirm this by playing the corrupted file. You should hear a tone. Plot the corrupted file to see what it looks like in the time domain. By determining the spectrum of the resulting corrupted signal estimate the frequency of the jamming signal.

4. Design a Twin Tee filter (i.e. specify its transfer function) so that the jamming sinusoidal frequency should be blocked. You will have to experiment a little with gain value K so that you can hear the signal. Do *not* make this gain too large, the resulting signal may be too loud to be safely played over insertion-style headphones. For now set it to one.

5. Plot the frequency response of the resulting Twin Tee filter and confirm that it should be expected to block the undesired jamming frequency but pass most other frequencies with essentially constant gain.

6. Using Matlab's **lsim** command find the output of the Twin Tee filter for input equal to the corrupted data. If possible listen to the resulting output signal, increasing the gain in increments until it is audible. Although the elimination of the jamming signal is far from perfect the underlying, original audio file should now be recognisable. Finally plot the magnitude spectrum of the output of the Twin Tee and, by comparing it with the magnitude spectrum of the input, explain why the audio file is now (or should now be) recognisable.