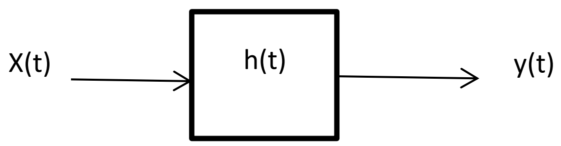
System representation

3.1 Time domain (Impulse response):

- The system is represented by its impulse response, and the relation between the input and the output of the system is as follows:

  y(t) = x(t) * h(t)



- Matlab performs only discrete convolution:

  Z = x*y

  Z[n] = x[n] * y[n]

  $= \sum x[k]y[n-k]$

  To implement continuous convolution, we must approximate it into discrete one:

  Z(t) = x(t) * y(t)

  $= \int_{-\infty}^{\infty} x(\tau)y(t-\tau)d\tau$

  $\cong \sum x(\tau)y(n-\tau)\Delta\tau$

  $\cong Ts$. discrete convolution (x,y)

3.2 Matlab function of convolution:

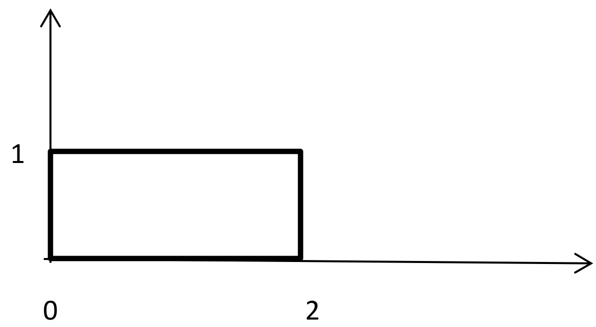
For continuous signals,

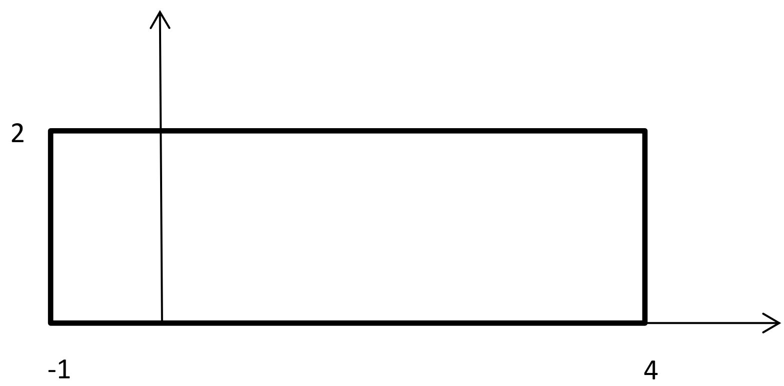
Z = $\frac{1}{fs}$ conv(x,y)

- Start time for the output of the convolution (z) = start time of (x) + start time of (y)
- End time for the output of the convolution (z) = End time of (x) + End time of (y)
- Length of the output (z) = length of (x) + length of (y) - 1

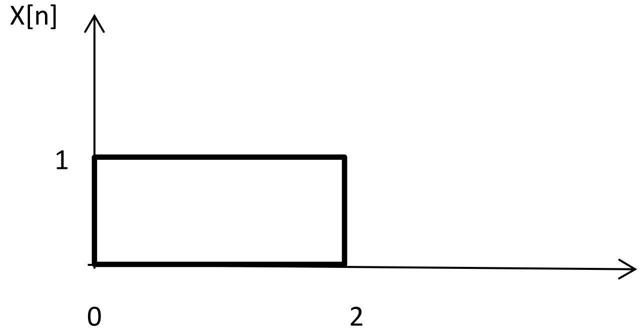
3.5 Exercises

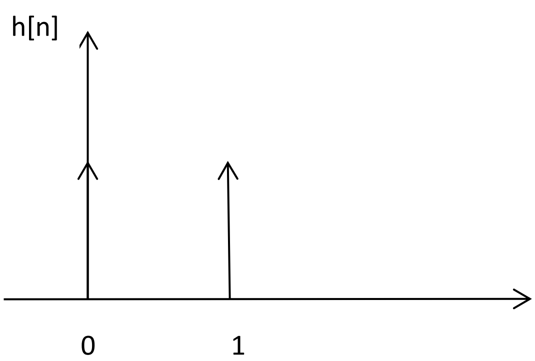
1. For sampling frequency, fs 1000, find the output of the convolution between the two rectangular signals:





2. Find the output of the system h[n] for the input x[n], and find the transfer function.





ECE 460 – Introduction to Communication Systems

MATLAB Tutorial #1

Evaluating Exponential Fourier Series

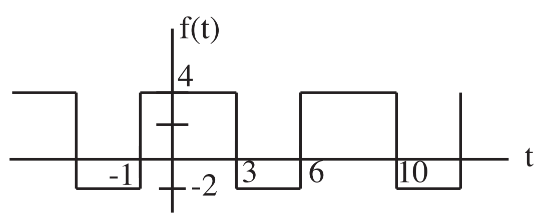
The homework assignments in this course contain problems that must be completed using MATLAB. A minimal knowledge of MATLAB is required to get started. Advanced MATLAB features will be introduced in tutorials posted on the homework web page. Students who have not used MATLAB before should go to the following web page:

http://www.mathworks.com/academia/student_center/tutorials/launchpad.html

View the "Getting Started" and "MATLAB Examples" videos for an overview. Then proceed to the Interactive MATLAB Tutorials and view "Navigating the MATLAB Desktop" and "MATLAB Fundamentals." This should give you enough information to proceed with the remainder of this tutorial. You may need to refer to the other Interactive MATLAB Tutorials if unfamiliar commands are used. Additionally, MATLAB has extensive online help and documentation.

Finding the Exponential Fourier Series Coefficients

For the waveform, f(t), shown below:



we can write the exponential Fourier Series expansion as:

$$f(t) = \sum_{n=-\infty}^{\infty} F_n e^{jn\omega_0 t}$$

where

$$F_n = \frac{1}{T}\int_T f(t)e^{-jn\omega_0 t}dt = \frac{1}{7}\left[\int_{-1}^{3} 4e^{-jn2\pi t/7}dt - \int_{3}^{6} 2e^{-jn2\pi t/7}dt\right]$$

$$= \ldots = \frac{1}{jn\pi}\left[-3e^{-jn6\pi/7} + 2e^{jn2\pi/7} - e^{-jn12\pi/7}\right]; \quad n \neq 0$$

and

$$F_0 = \frac{1}{T}\int_T f(t)dt = \frac{1}{7}\left[\int_{-1}^{3} 4dt - \int_{3}^{6} 2dt\right] = \frac{1}{7}\left[16 - 6\right] = \frac{10}{7}$$

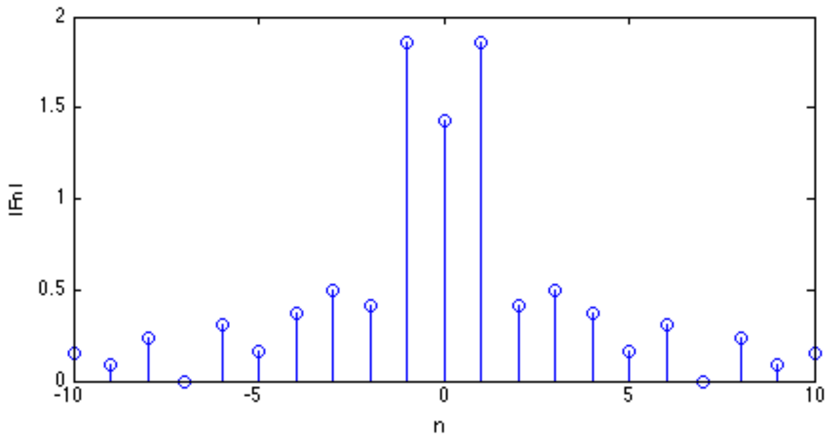
These coefficients may be simplified further but since we will be using MATLAB to evaluate them that is not necessary.

<u>Plotting the Amplitude and Phase Spectrum</u>

In order to plot the amplitude and phase spectrum of this pulse train we need to evaluate the values of F_n. Due to the divide by zero condition that occurred above, we need to handle the case for n = 0 separately. There are a number of methods for handling this. One method is to evaluate the positive, negative and zero values of n separately. In this case we will plot the spectrum for $-10 \leq n \leq 10$ using the following MATLAB commands.

```
>> clear
>> nneg=-10:-1;
>> npos=1:10;
>> Fnneg=(1./(j*nneg*pi)).*(-3*exp(-j*nneg*6*pi/7) ...
+2*exp(j*nneg*2*pi/7)+exp(-j*nneg*12*pi/7));
>> Fnpos=(1./(j*npos*pi)).*(-3*exp(-j*npos*6*pi/7) ...
+2*exp(j*npos*2*pi/7)+exp(-j*npos*12*pi/7));
>> F0=10/7;
>> n=[nneg, 0, npos];
>> Fn=[Fnneg, F0, Fnpos];
>> stem(n, abs(Fn))
```

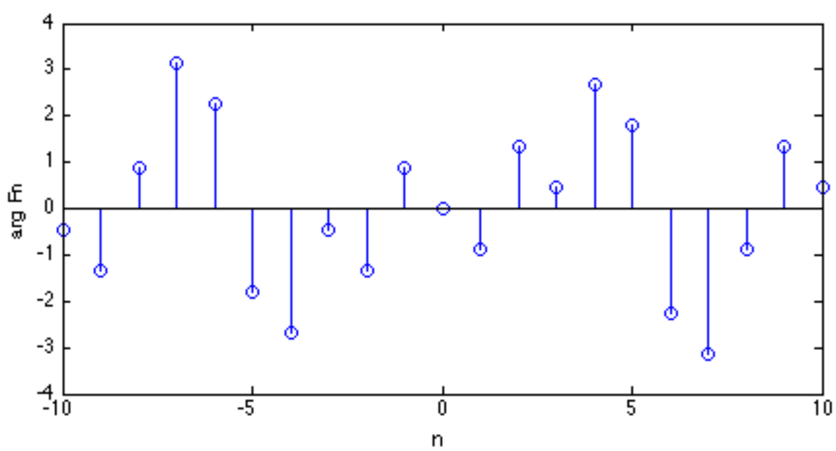
which results in the following amplitude spectrum:



The phase spectrum can then be found using:

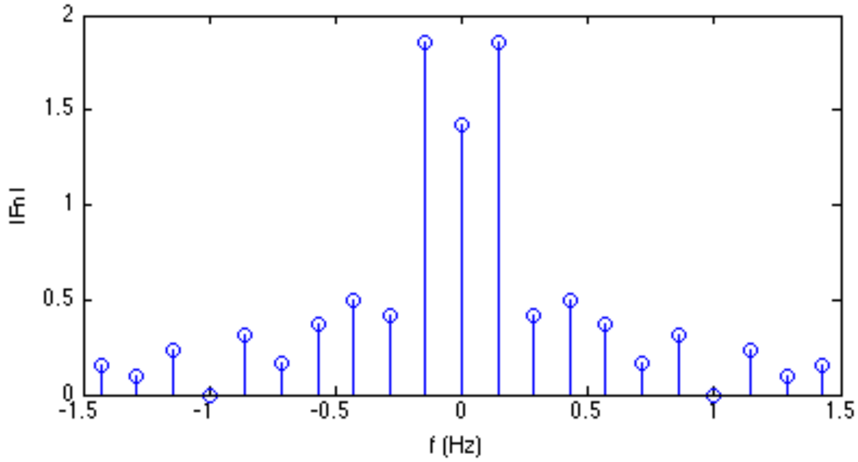
```
>> stem(n,angle(Fn))
```

which results in the following phase spectrum:



In this example, the fundamental frequency, $f_0 = 1/T = 1/7$. Thus, the amplitude spectrum can also be plotted versus the frequency in Hz with:

```
>> stem(n*(1/7),abs(Fn))
```

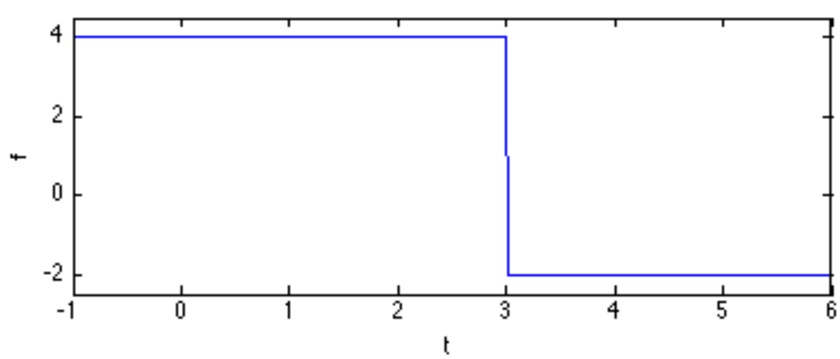


Plotting Functions with Discontinuities

In order to observe the accuracy of a truncated Fourier Series we need to plot it along with the exact function. There are many methods for plotting the exact function f(t). The following method makes use of logical operators.

```
>> t=-1:.01:6;
>> fexact=4*(t<=3)-2*(t>=3);
>> plot(t,fexact)
```

and results in the plot:



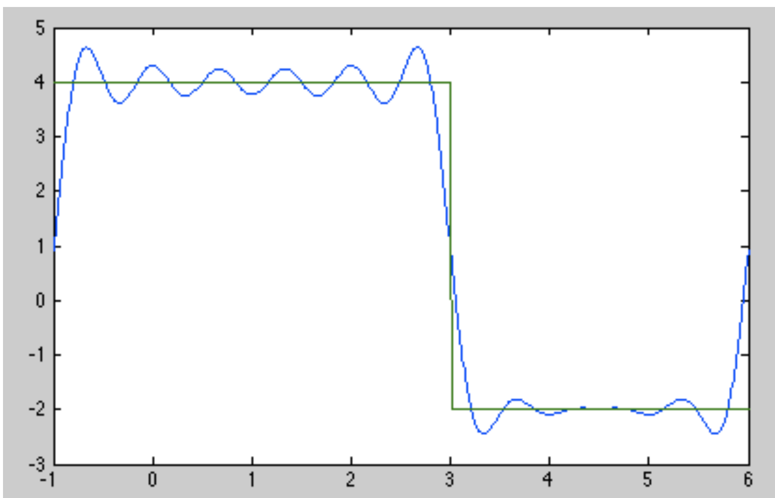
Plotting the Truncated Fourier Series

We can use the truncated exponential Fourier series as an approximation to the function, f(t). Recall that we must always use a symmetric range of n values ($-n_0 \le n \le n_0$) to obtain a real function. For $n_0 = 3$:

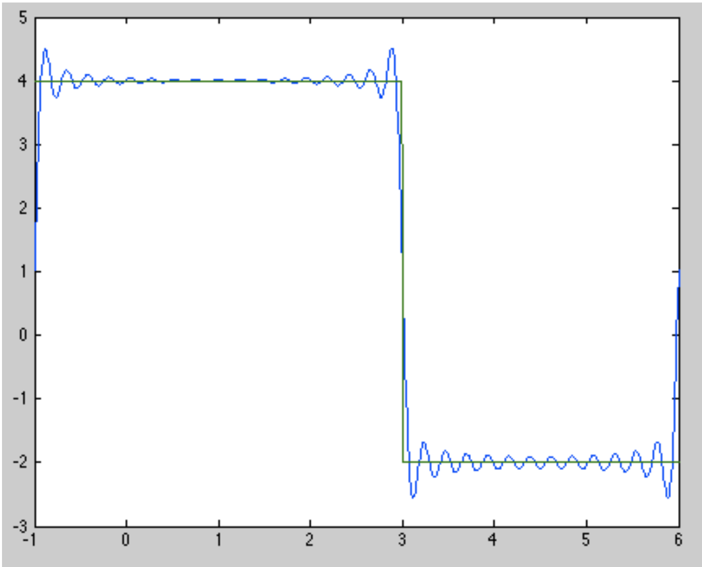
```
>> clear
>> nneg=-3:-1;
>> npos=1:3;
```

```
>> n=nneg;
>> Fnneg=(1./(j*n*pi)).*(-3*exp(-j*n*6*pi/7) …
        +2*exp(j*n*2*pi/7)+exp(-j*n*12*pi/7));
>> n=npos;
>> Fnpos=(1./(j*n*pi)).*(-3*exp(-j*n*6*pi/7) …
     +2*exp(j*n*2*pi/7)+exp(-j*n*12*pi/7));
>> F0=10/7;
>> n=[nneg,0,npos];
>> Fn=[Fnneg,F0,Fnpos];
>> k=0;
>> for t=-1:.01:6
k=k+1;
fapprox(k)=sum(Fn.*(exp(j*n*2*pi*t/7)));
end
>> t=-1:.01:6;
>> fexact=4*(t<=3)-2*(t>=3);
>> plot(t,fapprox,t,fexact)
```

Note that the first 10 lines calculate the F_n values as shown in a previous section. This is followed by a for loop that evaluates the series summation for each value of t. The values are placed in a vector fapprox. The following curves result from this MATLAB code:



Using only seven terms the truncated Fourier series makes a reasonable approximation to the pulse train. Note that as more terms are added the approximation improves. For $n_o = 30$:


5
4
3
2
1
0
-1
-2
-3
-1
0
1
2
3
4
5
6

Signal Processing in Frequency Domain

1 Introduction

In this lesson, we will explain how we transform a signal from time domain to frequency domain and vice versa, and how to carry out frequency domain operations. We will start by studying the FFT and IFFT functions, and the auxiliary FFT shift function. We will then consider two examples of operations in the frequency domain.

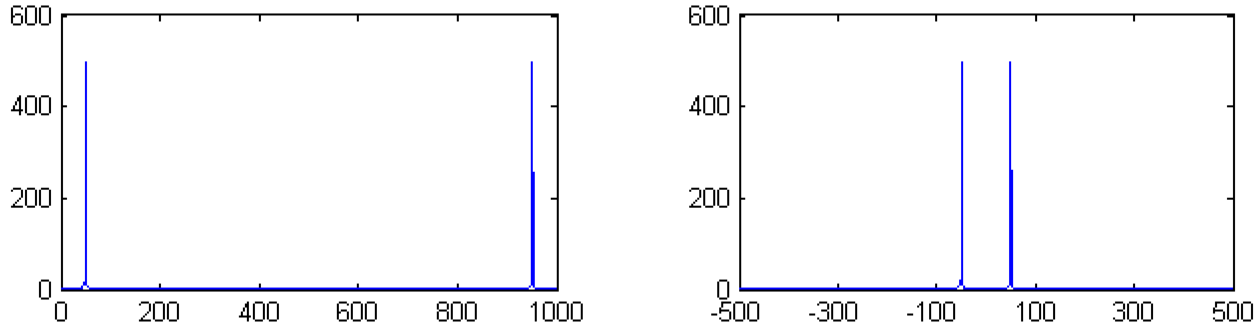
2 FFT and FFT-Related Functions

The Fast Fourier Transform (FFT) is a fast algorithm for computing the discrete Fourier transform of a discrete time limited signal. The following syntax shows how the FFT function is used.

```
>>X=fft(x);
```

In this instruction, x is the time domain signal, and X is its corresponding FFT result. The length of X is the same as the length of x.

There are several steps that are typically carried out after fft is invoked. The first typical step is invoking ‘fftshift’, a function that divides X into two halves, and swaps the position of the first and second half. This operation is usually required because the zero frequency location in the vector returned by FFT is not in the middle as we require, but at the start of the vector. After invoking the ‘fftshift’ function, the DC component is brought to the middle of the vector. This makes it easier in many cases to visualize the positive and negative frequency halves. The following two figures illustrate the effect of ‘fftshift’. The spectrum in figure 4.1 (a) was generated without invoking fftshift. The spectrum in figure 4.1 (b) was generated using fftshift.



(b)

Figure 4.1

Note that the spectra plotted in figure 4.1 are for a cosine wave of frequency 50 Hz. The second plot is more illustrative of the positive and negative frequency halves.

We also invoke the ‘abs’ (absolute) function to return the magnitude or the ‘angle’ function to return the phase of each element in the complex spectrum X as shown below.

```
>>Xmagnitude=abs(X);
>>Xphase=angle(X);
```

The instructions we have written so far are responsible for generating the data for the y-coordinates of the plot points. Now, we wish to generate the data for the frequency-coordinates of the plot points. The vector of frequency coordinates is called the “frequency base vector”. It may be generated using the following instruction:

```
>>Fvec=linspace(-fs/2,fs/2,Ns);
```

In the previous instruction, fs is the sample rate of the signal, and Ns is its number of samples. To sum up, let us consider the following example, in which we plot the magnitude and phase spectra of a signal comprising of the sum of two sinusoids.

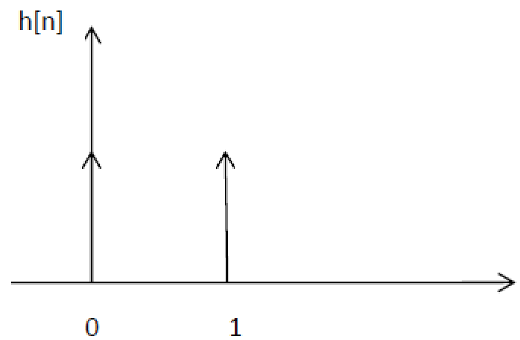
```
>>t=linspace(0,5,1000);
>>y=0.5*cos(60*pi*t)+0.3*sin(80*pi*t);
>>Y=fftshift(fft(y));
>>Ymag=abs(Y); Yphase=angle(Y);
>>Fvec=linspace(-100,100,1000);
>>figure; plot(Fvec,Ymag); title('Magnitude Spectrum');
>>figure; plot(Fvec,Yphase); title('Phase Spectrum');
```

The last FFT-related function we will consider is the inverse FFT (ifft) function.

```
>>x=ifft(X);
```

ASSIGNMENT

1) Find the output of convolution h(t) with x(t)



2) Consider an aperiodic function f(t)=e^-2t * u(t).

a) Plot f(t).

b) Express the Fourier Transform of f(t).

c) Calculate and plot the magnitude of the frequency spectra of f(t). This will be the magnitude of Fourier transform for f(t).

d) Calculate and plot the phase of the frequency spectra of f(t).

e) Calculate the power for f(t).