ECE 300
Signals and Systems
Homework 5

Due Date: Thursday, April 9, 2009 *at the beginning of class*

Problems

1. For the following system models, determine if the model represents a BIBO stable system. If the system is not BIBO stable, give an input $x(t)$ that demonstrates this.

a) $y(t) = \int_{-\infty}^{t} (x(\lambda) - 5) d\lambda$ b) $y(t) = \cos\left(\frac{1}{x(t)}\right)$

c) $y(t) = e^{-|x(t)|}$ d) $y(t) = x(t) + y(t)x(t)$

2. For LTI systems with the following impulse responses, determine if the system is BIBO stable.

a) $h(t) = e^{-t}u(t)$ b) $h(t) = u(t)$ c) $h(t) = u(t) - u(t-10)$ d) $h(t) = \delta(t-1)$

e) $h(t) = \sin(t)u(t)$ f) $h(t) = e^{-t^2}u(t)$ (hint: use your answer to **a)**

3. In this problem we will determine the trigonometric Fourier Series for a full wave rectified signal.

a. Using Euler's identity, show that $\sin(\alpha)\cos(\beta) = \frac{1}{2}\sin(\alpha+\beta) - \frac{1}{2}\sin(\beta-\alpha)$ and $\sin(\alpha)\sin(\beta) = \frac{1}{2}\cos(\alpha-\beta) - \frac{1}{2}\cos(\alpha+\beta)$

b. Show that for $x(t) = V_m \sin\left(\frac{\omega_0}{2}t\right) 0 \le t \le T_0$, the trigonometric Fourier series coefficients are given by

$$a_0 = \frac{2V_m}{\pi}$$

$$a_k = \frac{V_m}{\pi}\frac{1}{0.25 - k^2}$$

$$b_k = 0$$

4. (**Matlab/Prelab Problem**) Read **Appendix A**, then

Download **Fourier_Sine_Series.m** from the class website.

a) If you type (in Matlab's command line) Fourier_Sine_Series(5) you should get a plot like that shown in Figure 1. As you increase the number of terms in the Fourier series, you should get a better match to the function. Run the code for N=100 and turn in your plot.

b) Copy **Fourier_Sine_Series.m** to a file named **Trig_Fourier_Series.m** and implement a full trigonometric Fourier series representation. This means you will have to compute the average value a_o and the a_k, and then use these values in the final estimate.

c) Using the code you wrote in part **c**, find the trigonometric Fourier series representation for the following functions (defined over a single period)

Half-wave rectifier ($V_m = 100\sqrt{2}\ volts,\ f_0 = 60\ Hz$)

$$x_1(t) = \begin{cases} V_m \sin(2\pi f_0 t) & 0 \le t < \dfrac{T_0}{2} \\ 0 & \dfrac{T_0}{2} \le t < T_0 \end{cases}$$

Full-wave rectifier ($V_m = 100\sqrt{2}\ volts,\ f_0 = 60\ Hz$)

$$x(t) = V_m \,|\sin(2\pi f_0 t)\,|\ 0 \le t \le T_0$$

(Note that this is not the form of the function we used in problem 3, but this form gives a better idea of what a full-wave rectifier does)

Use N = 11 and turn in your plots for each of these functions. *Also, turn in your Matlab program for one of these.*

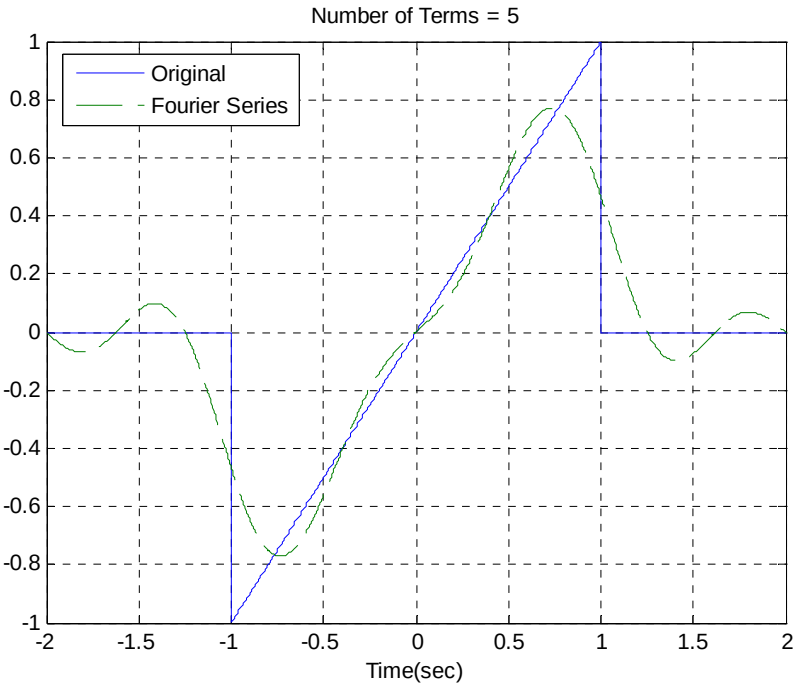

Figure 1. Trigonometric Fourier series for problem 3.

5. (**Matlab/Prelab Problem**) Read **Appendix B** and then do the following:

a) Copy the file **Trigonometric_Fourier_Series.m** to file **Complex_Fourier_Series.m**.

b) Modify **Complex_Fourier_Series.m** so it computes the average value c_o

c) Modify **Complex_Fourier_Series.m** so it ***directly*** computes c_k for $k=1$ to $k=N$. You are **not** to use the trigonometric Fourier series coefficients for this.

d) Modify **Complex_Fourier_Series.m** so it also computes the Fourier series estimate using the formula

$$x(t) \approx c_o + \sum_{k=1}^{N} 2\,|\,c_k\,|\cos(k\omega_o t + \measuredangle c_k)$$

You will probably need to use the Matlab functions **abs** and **angle** for this.

e) Using the code you wrote in part **d,** find the complex Fourier series representation for the following functions (defined over a single period)

$$f_1(t) = e^{-t}u(t) \quad 0 \le t < 3$$

$$f_2(t) = \begin{cases} t & 0 \le t < 2 \\ 3 & 2 \le t < 3 \\ 0 & 3 \le t < 4 \end{cases}$$

$$f_3(t) = \begin{cases} 0 & -2 \le t < -1 \\ 1 & -1 \le t < 2 \\ 3 & 2 \le t < 3 \\ 0 & 3 \le t < 4 \end{cases}$$

Turn in your code. *Be sure to modify your program so any unnecessary code is eliminated (not just commented out). Note that the values of* ***low*** *and* ***high*** *will be different for each of these functions!*

Appendix A

In DE II you learned about representing periodic functions with a Fourier series using trigonometric functions (sines and cosines). In this appendix we will determine how to determine a Fourier sine series series for an odd periodic function using our knowledge of numerical integration in Matlab. The only new thing we will need is the idea of a loop.

Trigonometric Fourier Series If $x(t)$ is a periodic function with fundamental period T, then we can represent $x(t)$ as a Fourier series

$$x(t) = a_0 + \sum_{k=1}^{\infty} a_k \cos(k\omega_o t) + \sum_{k=1}^{\infty} b_k \sin(k\omega_o t)$$

where $\omega_o = \frac{2\pi}{T}$ is the fundamental period, a_o is the average (or DC, i.e. zero frequency) value, and

$$a_o = \frac{1}{T}\int_T x(t)dt$$

$$a_k = \frac{2}{T}\int_T x(t)\cos(k\omega_o t)dt$$

$$b_k = \frac{2}{T}\int_T x(t)\sin(k\omega_o t)dt$$

Even and Odd Functions Recall that a function $x(t)$ is **even** if $x(t) = x(-t)$ (it is symmetric about the y-axis) and is **odd** if $x(-t) = -x(t)$ (it is antisymmetric about the y-axis). If we know in advance that functions are even or odd, we can determine that some of the Fourier series coefficients are zero. Specifically,

If $x(t)$ is even all of the b_k are zero.

If $x(t)$ is odd all of the a_k (including a_o) are zero.

For Loops Let's assume we want to generate the coefficients $b_k = \frac{k}{1+k^2}$ for $k = 1$ to $k = 10$. One way of doing this in Matlab is by using the following **for** loop

```
for k=1:10
  b(k) = k/(1+k^2);
end;
```

In this loop, the variable k first takes the value of 1 until the end is reached, then the value 2, all the way up to k = 10. In this loop we are assigning the coefficients to the array b, hence b(1) = b_1, b(2) = b_2, etc.

Similarly, if we wanted to generate the coefficients $a_k = \frac{1}{k+1}$ and $b_k = \frac{2}{k^2+1}$ for $k=1$ to $k=5$, we could do this using for loops as follows:

```
for k=1:5
  a(k) = 1/(k+1);
  b(k) = 2/(k^2+1);
end;
```

Note that Matlab requires the indices in an array to start at 1, and that for loops should usually be avoided in Matlab if possible since they are usually less efficient (Matlab is designed for vectorized operations).

Fourier Sine Series Now we want to generate the Fourier series for the periodic function

$$x(t) = \begin{cases} 0 & -2 \le t < -1 \\ t & -1 \le t < 1 \\ 0 & 1 \le t < 2 \end{cases}$$

This is clearly an odd function, so we will only need to generate a Fourier sine series. The following program will generate the required Fourier sine series:

```
%
%  This routine implements a trigonometric Fourier Sine Series
%
%  Inputs: N is the number of terms to use in the series
%
function Fourier_Sine_Series(N)
%
%  one period of the function goes from low to high
%
  low = -2;
  high = 2;
%
%  the difference between low and high is one period
%
  T = high-low;
  w0 = 2*pi/T;
%
%  the periodic function...
%
  x = @(t) 0.*(t<-1) + t.*((-1<=t)&(t<1)) + 0.*(t>=1);
%
%  find b(1) to b(N)
%
  for k = 1:N
      arg = @(t) x(t).*sin(k*w0*t);
      b(k) = (2/T)*quadl(arg,low,high);
  end;
%
%  determine a time vector over one period
%
    t = linspace(low,high,1000);
%
%  find the Fourier series representation
%
  est = 0;
  for k=1:N
      est = est + b(k)*sin(k*w0*t);
  end;
%
%  plot the results
%
  plot(t,x(t),'-',t,est,'--'); grid; xlabel('Time(sec)');
  legend('Original','Fourier Series','Location','NorthWest');
  title(['Number of Terms = ', num2str(N)]);
```

Appendix B

In the majority of this course we will be using the complex (or exponential) form of the Fourier series, since it is really easier to do various mathematical things with it once you get used to it.

Exponential Fourier Series If $x(t)$ is a periodic function with fundamental period T, then we can represent $x(t)$ as a Fourier series

$$x(t) = \sum_{k=-\infty}^{\infty} c_k e^{jk\omega_o t}$$

where $\omega_o = \frac{2\pi}{T}$ is the fundamental period, c_o is the average (or DC, i.e. zero frequency) value, and

$$c_o = \frac{1}{T}\int_0^T x(t)dt$$

$$c_k = \frac{1}{T}\int_0^T x(t)e^{-jk\omega_o t}dt$$

If $x(t)$ is a real function, then we have the relationships $|c_k| = |c_{-k}|$ (the magnitude is even) and $\measuredangle c_{-k} = -\measuredangle c_k$ (the phase is odd). Using these relationships we can then write

$$x(t) = c_o + \sum_{k=1}^{\infty} 2|c_k|\cos(k\omega_o t + \measuredangle c_k)$$

This is usually a much easier form to deal with, since it lends itself easily to thinking of a phasor representation of $x(t)$. This will be particularly useful when we starting filtering periodic signals