Fourier Series

EE 224: Signals and Systems I

1 Overview

In this lab you will experience the power of Fourier series analysis and synthesis.

2 Learning Objectives

By the end of this lab, students will be able to:

1. Use MATLAB to approximate Fourier series coefficients of a sampled signal.
2. Use MATLAB to synthesize a signal from its Fourier series coefficients.
3. Describe the difference in sound made by two musical instruments in terms of their Fourier series coefficients.
4. Create your own synthesized instrument using Fourier series (bonus).

3 Pre-Lab Reading

3.1 Fourier Analysis

Begin by reviewing the Fourier analysis formula. Given a periodic signal $x(t)$ with period T, $x(t)$ can be represented by

$$x(t) = \sum_{k=-\infty}^{\infty} a_k e^{jk\omega_0 t} \tag{1}$$

where $\omega_0 = \frac{2\pi}{T}$ and

$$a_k = \frac{1}{T}\int_0^T x(t)e^{-jk\omega_0 t}dt \qquad \text{for all } k \tag{2}$$

Although an infinite number of harmonics may be required for a general signal, in most situations, a finite number of them provides a practically good approximation.

If the signal $x(t)$ is not given as a mathematical function but we have a sampled recorded version of it, the integrals to compute the coefficients (a_k) cannot be evaluated precisely.

Although there are more efficient methods to perform the Fourier analysis directly on the sampled signals (e.g., the Fast Fourier Transform (FFT)), we will use a simple method to approximately evaluate those integrals: the Riemann approximation. Assuming that the signal $x(t)$ is reasonably smooth (Riemann integrable) and that the sampling time T_s is an integer fraction of the period T, i.e., $T = NT_S$ of samples in a period (this is not really necessary but simplifies our code), then:

$$a_k = \frac{1}{T}\int_0^T x(t)e^{-jk\omega_0 t}dt \simeq \frac{T_s}{T}\sum_{n=0}^{T/T_s} x[T_s n]e^{-jk\omega_0 T_s n}. \tag{3}$$

The approximation gets better and better as the number of samples goes to infinity (meaning T_s goes to zero). In other words, fixing the sampling time limits the quality of the approximation, especially for larger values of k. This is because T_s becomes too large with respect to $k\omega_0$, the frequency of the kth harmonic.

Find the Fourier series coefficients a_0, a_1, a_2, and a_3 for the square wave given below (assume the signal repeats with period $T = 0.04$). Use the integral in (3) and include your answer in your report.

$$x(t) = \begin{cases} 1 & 0 \le t < 0.02 \\ 0 & 0.02 \le t < 0.04 \end{cases}$$

3.2 Fourier Synthesis

The synthesis formula allows one to generate periodic signals from the linear combination of harmonic complex exponentials. Here we assume that the number of non-zero coefficients a_k is finite. Then

$$\tilde{x}(t) = \sum_k a_k e^{jk\omega_0 t}$$

In particular,

$$\tilde{x}[T_s n] = \sum_k a_k e^{jk\omega_0 T_s n}$$

4 Lab Exercises

4.1 Fourier Series Analysis Function

Write a Matlab function "findFS" which takes as inputs x_T, a vector of samples of $x(t)$ covering one period of $x(t)$; k, the index of the desired coefficient; the period T; and the sampling time T_s; and produces as output the approximate a_k coefficient according to (3). Note that you can easily take advantage of vectorization by considering the product of the row vector x_T and the column vector $e^{-jk\omega_0 T_s n}$. A template for the function is provided.

4.2 Fourier Series Coefficients of a Square Wave

Let's construct a test signal. Define in Matlab

`x_T=[ones(1000,1);zeros(1000,1)];`

This is one period of a square wave signal. Let the fundamental period be $T = 0.04$ sec.

a) Calculate the sampling time T_s and explain your answer. (Hint: $T_s = T/2000$)

b) Use the function you wrote in the previous exercise to compute a_0, a_1, a_2, and a_3.

c) Find a_{-1}, a_{-2}, and a_{-3}. How do these relate to a_1, a_2, and a_3?

d) Verify the coefficients you have computed approximate reasonably well the actual coefficients you obtained for the prelab.

4.3 Fourier Series Synthesis

Write a Matlab function "synthFS" which has the following inputs: a column vector $C = [a_0; a_1; \ldots; a_m]$ of coefficients where m is the largest integer corresponding to a non-zero coefficient, the sampling time T_s, and the fundamental period T. The output should be $\tilde{x}_T(T_s n)$, the vector of $N = T/T_s$ (N an integer) samples corresponding to one period of $\tilde{x}$. Note that your program should verify that T/T_s is an integer. You will also need to extend C to include the complex conjugate coefficients corresponding to the negative ks. Finally, you may want to generate an array `F` whose columns are the vectors $e^{jk\omega_0 T_s n}$. $\tilde{x}_T$ is then computed as `F*CC`, where `CC` is a vector containing all the coefficients from $-m$ to m. A template for this function is provided.

4.4 Spectra of a Trumpet and Whistle

Load the data file `trumpet_whistle.mat`, which can be found on Canvas. It contains vectors `trumpet` and `whistle`. Both signals are sampled at $f_0 = 44100$ Hz, and they represent one period of synthetic trumpet and whistle tones generated by a toy electric keyboard.

a) Compute the period of the two signals.

b) For each signal, compute and report the first 9 harmonic coefficients, i.e., a_ks with $k = -9, \ldots, 9$ using the function "findFS" you have developed.

c) For each signal, plot the magnitude and phase spectra of the frequency components from part b. Briefly comment on their main differences. You may use the Matlab function `stem` to plot the spectrum.

d) For each signal, synthesize an approximation by using the periods in part a, the coefficients in part b, the sampling time $T_s = 1/f_0$, and your function "synthFS."

e) For each signal, plot on the same plot the given signal and its synthesized approximation. (You may use `spectraplot.m` found on Canvas. Make sure the axis labels are accurate.) Comment on the quality of the approximation in both cases.

f) For each signal and its approximation, generate a new signal by repeating them for 1000 periods (e.g., using the built in function `repmat`). Use the function `sound` in Matlab and the sampling frequency f_0 to hear the sound you have generated. Can you hear the difference between the originals and the approximations? Comment. (Note you may need to scale the signals to have magnitude 1, see `help sound`.)

Note that the trumpet sound has a much richer spectrum and corresponds to a more complex sound. While the whistle sound essentially does not have any harmonic components above the ninth and does not have any even harmonic components, the trumpet sound has important harmonics above the ninth; this can be argued from the error in the approximation.

4.5 Bonus

Synthesize your own "instrument" by repeating steps d-f of the last subsection. Instead of using the a_k that you calculated for the trumpet and whistle, come up with your own coefficients any way you want. (You don't have to limit yourself to nine coefficients.) Upload your sound and describe it in your report.

4.6 Report Checklist

Be sure the following are included in your report.

1. Section 4.1: include your function in the appendix
2. Section 4.2: provide answers for a-d
3. Section 4.3: include your function in the appendix
4. Section 4.4: provide answers for a-c, e, and f
5. Section 4.5: (bonus) describe the sound you created and explain how you created it