Signal Processing First

Lab 04: Numerical Evaluation of Fourier Series

Pre-Lab and Warm-Up: You should read at least the Pre-Lab and Warm-up sections of this lab assignment and go over all exercises in the Pre-Lab section before going to your assigned lab session.

Verification: The Warm-up section of each lab must be completed **during your assigned Lab time** and the steps marked *Instructor Verification* must also be signed off **during the lab time**. One of the laboratory instructors must verify the appropriate steps by signing on the **Instructor Verification** line. When you have completed a step that requires verification, simply demonstrate the step to the TA or instructor. Turn in the completed verification sheet to your TA when you leave the lab.

1 Introduction & Objective

The goal of the laboratory project is to show how to calculate the Fourier Series coefficients, $\{a_k\}$, without doing the integrals by hand. Instead, we will use MATLAB's numerical integration capability to evaluate the integrals numerically. Another approach would be to use a symbolic algebra package such as *Mathematica* or Maple to derive formulas for the Fourier Series coefficients.

1.1 Background: Fourier Series Analysis and Synthesis

The Fourier Series representation applies to periodic signals. The Fourier synthesis equation for a periodic signal $x(t) = x(t + T_0)$ is

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

where $\omega_0 = 2\pi/T_0$ is the *fundamental* frequency. To determine the Fourier series coefficients from a time-domain formula for the signal over one period, we must evaluate the *analysis* integral for every integer value of k:

$$a_k = \frac{1}{T_0} \int_{-T_0/2}^{T_0/2} x(t) e^{-jk\omega_0 t} dt \tag{2}$$

where $T_0 = 2\pi/\omega_0$ is the *fundamental* period. If necessary, we can evaluate the analysis integral over any period; in (2) the choice $[0, T_0]$ is sometimes a convenient one, but integrating over the interval $[-1/2\ T_0, 1/2\ T_0]$ would also give exactly the same answer.

2 Pre-Lab

In this lab, we will use a feature of MATLAB that allows you to evaluate integrals by numerical approximation. MATLAB has several numerical integration functions, such as `quad()` or `quad8()` that are very powerful methods for evaluating integrals of functions specified by formulas. The "formula" for the integrand must be written as a MATLAB function. Then the numerical integration M-file will adaptively figure

out the best way to approximate the integral with a sum. We can use these functions to evaluate Fourier Series integrals and then we will plot the resulting Fourier Series coefficients $\{a_k\}$ to display the spectrum.

2.1 Numerical Integration

The basic idea of numerical integration is to approximate an integral with a sum, and the simplest way to do this is with the Riemann sum:

$$\int_a^b f(t)dt \approx \sum_{n=1}^{L} f(t_n)\left(\frac{b-a}{L}\right)$$

where the "sampling points" t_n are $t_n = a + (n - \frac{1}{2})\Delta$, and $\Delta = \frac{b-a}{L}$. The parameter Δ is actually the width of subintervals defined by breaking the interval $[a, b]$ into L equal sized subintervals. The sampling points are taken to be the *mid-points* of those subintervals.[1]

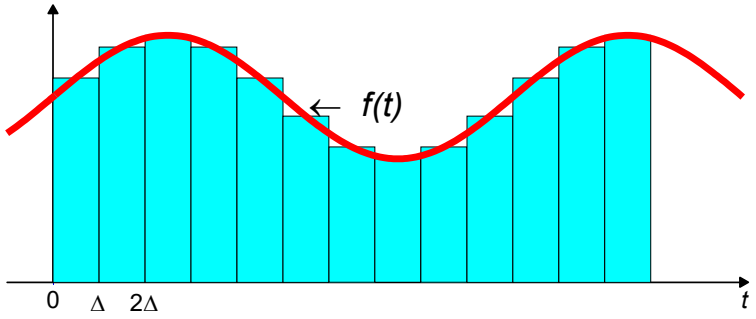

Figure 1: Approximating the area under $\mathbf{f(t)}$ with the area of many narrow triangles.

The theory of Riemann integration (in calculus) tells us that taking the limit as L $\rightarrow \infty$ will make the sum on the right converge to the integral on the left in (3). Furthermore, if we think of the integral as calculating area, then the Riemann sum is approximating the area with the sum of areas of many rectangles whose width is Δ and whose height is $\mathbf{f(t_n)}$. We could get a better approximation to the area if we used trapezoids instead of rectangles to compute the area under the curve $\mathbf{f(t)}$. Carrying this idea to the next step, we can use polynomial approximations to $\mathbf{f(t)}$ to get even more accurate area approximations. This is what the MATLAB function `quadl` do. Consult `help quadl` for more information.

2.2 Integrate a Simple Function with MATLAB

Suppose that we want to calculate the value of the definite integral $\int_0^5 \cos(\pi t)dt = ?$

In MATLAB this can only be done as a numerical approximation with functions such as `quadl`.

2.2.1 Numerical Integration with `quadl`

In order to use `quadl('f',a,b)` we must pass three arguments: the function definition (of the integrand), and also the lower and upper limits of integration. The limits of integration are scalar constants, so the tricky part is passing the function definition. There are two ways to do this: (1) write a separate MATLAB function to define the integrand and use the name in `quadl` , or (2) use the MATLAB function `inline()` to define the entire function via one string.

In MATLAB it seems reasonable that we might be able to define the first argument in `quadl` by writing `'cos(pi*t)'` as a string.[3] However, this won't work with `quad8` unless we use the `inline()` function.
(See Section 2.5 below.) The alternative is to write a separate M-file that sets up the integrand as a function that can be evaluated. For example, we might create an auxiliary M-file called `fcos.m` by writing the following simple function:

```
function out = fcos(t)
%FCOS   function definition of a cosine used in an integral
%       the input t can be a vector
%
out = cos(pi*t);
```

Once we have the `fcos()` function stored in the file `fcos.m`, we can perform the integration by using the **name** of the function in the call to `quadl()`:

```
quadl( 'fcos', 0, 5 )
```

You should run this example and verify that it gives the correct answer (because you can do the integral in your head). Notice that MATLAB gives an answer that is correct to the inherent numerical precision of double-precision floating point which is usually about 15 digits of accuracy.

(a) Modify the example above to do the integral of $e^{-|t|}$ from $t = -2$ to $t = 2$. This requires that you write a new auxiliary function M-file to generate "e to the minus absolute value of t."

(b) Verify by hand that the correct value of the definite integral in part (a) is 1.7293.

2.2.2 Numerical Integration with `quadl` (Version 6 only)

Starting with version 6 of MATLAB, the function `quad8` is being phased out in favor of one called `quadl`. The function calls are basically the same, except that `quadl` supports two additional ways to define the integrand. First of all, if the M-file name is used, it should be preceded with an "at" symbol, i.e., `quadl(@fcos,0,5)` for the example above. Second, if a text string is used to define the function, it no longer has to be wrapped inside of the `inline()` function, i.e., `quadl('cos(pi*t)',0,5)` works.

2.3 Inline Functions

We can eliminate the auxiliary M-file if we exploit MATLAB's capability to define the function as a string. This only works when the function is simple and can be expressed as a "one-liner." In version 5, it is necessary to use an "in-line" function definition as a wrapper around the string.

Consult help on `inline`

For the cosine example above, here is the method:

```
quadl( inline('cos(pi*t)'), 0, 5)
```

(a) Make a plot of the signal $\mathbf{f(t) = 2 - |t|}$ over the range $-1 \leq t \leq 1$ by using `inline`.

(b) Evaluate the definite integral of $\mathbf{f(t) = 2 - |t|}$ over the range $-1 \leq t \leq 1$ by using one call to `quadl` and `inline`. Verify by hand that MATLAB gave the correct answer.

3 Warmup

In the warm-up you will use `quadl` to do more complicated integrals, such as those needed in Fourier Series analysis (2) to extract the Fourier coefficients, a_k .

3.1 Piecewise Definition of a Signal

Many signals are defined by giving several cases that define pieces of the overall signal. For example, the square wave is one for part of its period and zero for the rest. There are a couple of ways of dealing with cases when writing the auxiliary M-file needed for `quadl`. The tricky part is making sure that the auxiliary function can handle a vector input. Pay careful attention to the warning in the help for `quadl` that states:

```
Q = QUADL('F',A,B) approximates the integral of F(X) from A to B...  'F'
is a string containing the name of the function.  The function must return
a vector of output values if given a vector of input values.
```

(a) Consider the problem of plotting the triangle wave of Fig. 2 In order to define the waveform so that you can plot it over *any interval*, it is necessary to figure out two things: how to plot one period of the signal, and then how to plot other periods. In this part, write an auxiliary function that will define the triangle wave over one period from $\mathbf{t = 0}$ to $\mathbf{t = 0.04}$. In other words, convert the following mathematical statement into **vectorized** MATLAB code:

$$f(t) = \begin{cases} 50t & \text{for } 0 \leq t < 0.02 \\ 2 - 50t & \text{for } 0.02 \leq t \leq 0.04 \end{cases} \tag{4}$$

There are two ways to approach this (use either one to write your triangle wave function):

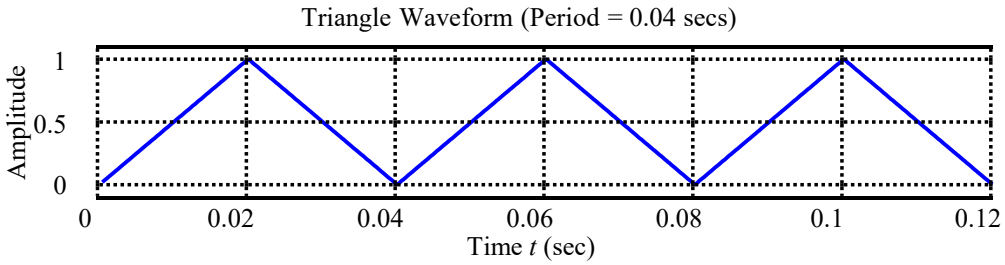

Figure 2: Triangle Wave requires a piecewise definition.

(i) Use the `find()` function to identify the subset of inputs associated with each case. For example, `jkl = find(tt>=0 & tt<=1); yy = exp(tt(jkl));` will define a signal that is e^t over the interval $0 \leq t \leq 1$.

(ii) Define one period of the triangle wave by using unit-step signals to isolate different regions of the signal. For example, you can define $f(t) = e^t[u(t) - u(t-1)]$ by writing

```
function out = myfunc(tt)
out = exp(tt).*((tt>=0)-(tt>=1));
```

The logical expression `((tt>=0)-(tt>=1))` evaluates to 1 if the condition ($0 \leq t \leq 1$) is true and it evaluates to 0 if the condition is false. Thus, multiplying by `((tt>=0)-(tt>=1))` has the effect of "switching on" the exponential function over the desired set of values of `tt`. You can use this technique to switch on and off various pieces of the function definition as required in (4). Notice that the two cases of the triangle wave define in (4) can be written as:

$$f(t) = (50t)[u(t) - u(t - 0.02)] + (2 - 50t)[u(t - 0.02) - u(t - 0.04)] \qquad (5)$$

(b) Show that you can calculate the DC value of the triangle wave by integrating over any one period of the signal. For example, you can do the integral from $t = -0.02$ to $t = +0.02$; or from $t = 0$ to $t = 0.04$. Do the integral over both ranges and compare. Remember to divide by the period. Compare your answer to the correct answer which you can obtain by hand.

3.2 Using Parameters in the Integrand

One limit of the examples above is that we cannot have any variables in the function definition other than t. However, we want to do Fourier Series calculations, so we know that we must have the parameter k and probably the period T_0 inside the integrals. If you look at the help on `quad8()` you will see that it has additional arguments that can handle these extra parameters. For example, we can construct a general cosine signal with different frequencies by doing the inline definition:

```
inline('cos(2*pi*f*t)','t','f')
```

Even in the case of `inline` we must be careful to identify the names of the different parameters in the function definition. The statement above tells us that t and f are parameters, and gives an ordering: t is the first parameter and f is the second one. This ordering is significant when `fplot` and `quad8` are used because they will treat the first parameter as the independent variable and all others as fixed parameters.

(a) When using `quadl` there are 2 input arguments that must be skipped in order to pass parameters to the function definition (consult `help quadl`). Here is an example of integrating a chirp, $A\cos(\pi\alpha t^2)$.

```
quadl(inline('A*cos(pi*alfa*t.*t)','t','A','alfa'),0,1,[],[],100,13;
```

You must have the two empty matrices in the argument list, so that the last two arguments will be used as parameters, i.e., the number 100 will be used for the amplitude A, and 13 will be used for the value of α. Notice that you must use the "point-star" operator for t^2 because `quadl()` is expecting to evaluate the inline function for a vector of t's.

(b) Follow the style of the previous two parts to make a plot of the signal:

$$x(t) = A\cos(\pi\alpha t^2)\,[u(t) - u(t-1)]$$

over the interval $-1 \leq t \leq 3$. For the parameters, use the values $A = 2.5$ and $\alpha = 10$. In addition, compute the definite integral over the same range.