Lab Assignment 9
Fourier Methods

ACS II
Spring 2020

Assigned: March 5, 2020
Due: March 12, 2020

1 Fourier Series

In this lab, we will use the Fourier Series to approximate a set of functions, and then we'll use the Fourier transform to do the same. We'll need to compute the Fourier coefficients, which will require us to compute some integrals.

The Fourier series is a representation of a periodic function $f(t)$ on the domain $[-\pi, \pi]$, and is given by

$$f(t) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n \cos(nt) + \sum_{n=1}^{\infty} b_n \sin(nt),$$

where

$$a_0 = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\ \mathrm{d}t,$$

$$a_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(t)\cos(nt)\ \mathrm{d}t,$$

$$b_n = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin(nt)\ \mathrm{d}t.$$

Using complex variables, the formulation of the Fourier series coefficients can be somewhat simplified. Using Euler's identity, we rewrite the above series approximation as

$$f(t) = \sum_{n=-\infty}^{+\infty} A_n e^{int}, \tag{1}$$

where

$$A_n = \frac{1}{2\pi}\int_{-\pi}^{\pi} f(t)e^{-int}\ \mathrm{d}t. \tag{2}$$

If $f(t)$ is periodic over $[-L, L]$ instead of $[-\pi, \pi]$ we can use the following change of variables to transform the interval of integration to $[-L, L]$:

$$t \to \frac{\pi t'}{L}, \qquad \mathrm{d}t \to \frac{\pi}{L}\mathrm{d}t'.$$

1.1

Consider the function:

$$f(t) \;=\; t$$

on the interval $t \in [-\pi, \pi]$. Plot the exact solution (this can be done in Matlab/Python), along with the following Fourier series approximations:

- $\sum_{n=-1}^{1} A_n e^{int}$
- $\sum_{n=-2}^{2} A_n e^{int}$
- $\sum_{n=-4}^{4} A_n e^{int}$
- $\sum_{n=-8}^{8} A_n e^{int}$
- $\sum_{n=-16}^{16} A_n e^{int}$

Where A_n is computed according to (2). You can compute the integrals by hand using integration by parts or by using software. In either case please include a compact formula for A_n (*not* a numeric approximation). Remember to look at the $n = 0$ case separately. Comment on the error as we include more terms.

1.2

Repeat this exercise for the function:

$$f(t) \;=\; t^2$$

on the interval $t \in [-\pi, \pi]$.

1.3

And again for the periodic square wave function:

$$f(t) = \begin{cases} -A & -P/2 < t < 0 \\ A & 0 < t < P/2 \end{cases}$$

for $A = 10$ and $P = 10$.

2 Fourier Transform

A function $f(t)$ and its Fourier transform $F(k)$ are related as follows

$$F(k) = \mathcal{F}[f](k) = \int_{-\infty}^{\infty} f(t)e^{-2\pi ikt}\,\mathrm{d}t, \qquad f(t) = \mathcal{F}^{-1}[F] = \int_{\infty}^{\infty} F(k)e^{2\pi ikt}\,\mathrm{d}k.$$

The Fourier transform expresses $f(t)$ in terms of the orthogonal basis functions $e^{2\pi ikt}$, where $k \in \mathbb{R}$. The variable t is often called the physical variable, while k is called the frequency variable. The Fourier transform is actually the limit of the Fourier series of h as $L \to \infty$.

Computationally we may want to evaluate the Fourier transform of a function whose value we know only at a discrete set of points. This known as the discrete Fourier transform (DFT). Suppose we have some underlying resolution Δ and we have the data:

$$f_n = f(n\Delta), \qquad n = 0, \dots, N-1.$$

In other words we know the value of some function $f(t)$ at N equally spaced locations. Since we have N function values, we can only get N frequencies. Using quadrature to evaluate the continuous transform, we get the discrete Fourier transform:

$$F_k = \sum_{n=0}^{N-1} f_n e^{2\pi ikn/N},$$

and the inverse DFT:

$$f_n = \frac{1}{N}\sum_{k=0}^{N-1} F_k e^{-2\pi ikn/N}.$$

This can be expressed as a matrix multiplication:

$$\mathbf{F} = A\mathbf{f}, \qquad \mathbf{f} = B\mathbf{F},$$

where

$$A_{\ell m} = e^{2\pi i\ell m/N}, \qquad B_{\ell m} = \frac{1}{N}e^{-2\pi i\ell m/N}.$$

The problem is that matrix multiplication is an $O(N^2)$ operation. If N is large this is very slow. The fast Fourier transform (FFT) is an algorithm to perform a DFT in $O(N \log N)$ operations. This algorithm is not straightforward to implement, however there are several implementations freely available.

For this section, we will use a software package to compute the DFT and inverse DFT of a function using the FFT. We will use software from the FFTPACK package, which is available from the NETLIB repository, or the FFTW package. Both softwares are available in either C or Fortran (actually, there is also a Java implementation for FFTPACK) so you can choose what language you'd like to use. For this lab, we will need the Fourier transform functions for real-valued functions. For a given value of sample points you'll have to use the initialization routine to prepare the allocated memory, and then the forward- and backward- transform routines.

One of the difficulties of working with these packages is that different implementations use different conventions for indexing and scaling. For example, instead of indexing data points as $n \in [0, N-1]$, many implementations index as $n \in [-N/2+1, N/2]$ for even N , or $n \in [-(N-1)/2, (N-1)/2]$ if N is odd. In addition, some packages omit the $1/N$ factor when performing the inverse DFT. ***Make sure to read the documentation for whatever FFT library you use!***

When plotting the Fourier spectrum of a function, we plot the absolute value of the Fourier coefficients versus the indices of the coefficients. The Fourier coefficients are complex, so the absolute value may be computed by

$$|a+ib| = \sqrt{a^2+b^2}.$$

Additionally, the absolute values of the Fourier coefficients are symmetric, so when we plot the Fourier spectrum we need only to plot the first half of the coefficients (for example coefficient numbers $0 \ldots N/2-1$, depending on how they are indexed). You will need to examine the output of the FFTPACK\FFTW routine to discover how to obtain these values.

Using the appropriate software, compute the discrete Fourier transform of the function

$$f(t) = t^2 \cos(t)$$

on the interval $t \in [-\pi, \pi]$. Use the inverse Fourier transform to shift back to the physical domain. Generate a plot of the original function and the discrete Fourier approximations for $n = 16, 32, 64$ on a single set of axes. Do you notice any difference between these plots? Also plot the Fourier spectrum for the case where $n = 64$. The results from these tests should indicate whether you're using the FFTPACK routines successfully.

Consider the function:

$$f(t) = \sin(2\pi(25)t) + \sin(2\pi(80)t) + \sin(2\pi(125)t) + \sin(2\pi(240)t) + \sin(2\pi(315)t)$$

on the interval $t \in [0, 1]$. Compute the discrete Fourier transform of this function for $n = 2^{10}$ sample points. Plot the frequency spectrum. Next, add a vector of white noise to the function data (this noise should have a mean of zero and a standard deviation of 2). Compute the discrete Fourier transform of this "noisy signal," and plot the frequency spectrum. Comment on the differences between the frequency spectrum of the clean and noisy signals. Also plot the clean and noisy functions together on the same set of axes.

Don't forget to include a README file with your submission containing compilation instructions!

Submission and Grading

To get credit for this assignment, you must submit the following information to the lab instructor and course instructor by 11:59pm, March 12, 2020:

- your source code files
- Your report as a single file in pdf format, including results from your work and relevant discussion of your observations, results, and conclusions.

This information must be received by 11:59pm, March 12, 2020. Upload the required documents to Canvas. As stated in the course syllabus, late assignment submissions will be subject to a 10% point penalty per 24 hours past the due date at time of submission, to a maximum reduction of 50%, according to the formula:

*[final score] = [raw score] - min(0.5, 0.1 * [# of days past due]) * [maximum score]*