Fourier Series

Let (a, b) be an interval of finite length. Among functions defined on $a < x < b$, there are some of finite length—i.e., ones for which the integral

$$\|f\|^2 \;=\; \int_a^b |f(x)|^2\, dx,$$

exists and is finite, and some for which the integral in (1) is infinite. The functions defined on (a, b)[1] for which (1) is finite form a vector space called $L^2(a, b)$. Inside of $L^2(a, b)$ resides the subspace of all continuous functions defined on $[a, b]$.

In what follows, we restrict our attention to intervals of the form $(-\ell, \ell)$—i.e., ones centered on 0. In particular, we start in the context of $(-\pi, \pi)$. Consider, for some fixed n, the set of functions

$$T_n(\ell) \;=\; \left\{1, \cos\left(\frac{\pi x}{\ell}\right), \sin\left(\frac{\pi x}{\ell}\right), \cos\left(\frac{\pi x}{\ell}\right), \sin\left(\frac{2\pi x}{\ell}\right), \ldots, \cos\left(\frac{n\pi x}{\ell}\right), \sin\left(\frac{n\pi x}{\ell}\right)\right\}. \qquad (1)$$

Here are some claims about the set $T_n(\ell)$, which hold regardless of the choices of $n = 1, 2, \ldots$, or $\ell > 0$.

Theorem 1:

(i) The functions in $T_n(\ell)$ are mutually orthogonal under the $L^2(-\ell, \ell)$ inner product. In particular, the set $T_n(\pi)$ is a set of mutually orthogonal functions under the $L^2(-\pi, \pi)$ inner product.

(ii) With just one exception, the *squared length*, in $L^2(-\ell, \ell)$, of the functions in T_n is ℓ. The one exception is the constant function, with $\|1\|^2 = 2\ell$.

(iii) The nearest function in span(T_n), to a given function f defined on (a, b) is

$$p(x) \;=\; \frac{a_0}{2} + \sum_{j=1}^{n}\left[a_j \cos\left(\frac{j\pi x}{\ell}\right) + b_j \sin\left(\frac{j\pi x}{\ell}\right)\right], \qquad (2)$$

where the coefficients are given by

$$a_j \;:=\; \frac{1}{\ell}\int_{-\ell}^{\ell} f(x)\cos\left(\frac{j\pi x}{\ell}\right) dx \qquad \text{and} \qquad b_j \;:=\; \frac{1}{\ell}\int_{-\ell}^{\ell} f(x)\sin\left(\frac{j\pi x}{\ell}\right) dx\,. \qquad (3)$$

[1] Actually, it is possible to loosen this domain slightly, demanding only that f be defined for *almost every* x in (a, b), while still requiring the integral (1) be finite. This distinction between "every x in (a, b)" and "almost every x in (a, b)" is beyond the scope of this class and will not be mentioned again. To learn more, you might investigate the fascinating area of mathematical analysis known as *measure theory*.

Proof: Parts (i) and (ii) can be demonstrated by carrying out the integrals involved, and using trig. identities when needed. I will not do these here, but say only that you want to handle multiple instances at once by calculating

$$\int_{-\ell}^{\ell} \cos\left(\frac{m\pi x}{\ell}\right)\cos\left(\frac{k\pi x}{\ell}\right)\,dx, \quad \int_{-\ell}^{\ell} \sin\left(\frac{m\pi x}{\ell}\right)\sin\left(\frac{k\pi x}{\ell}\right)\,dx, \quad \text{and} \quad \int_{-\ell}^{\ell} \cos\left(\frac{m\pi x}{\ell}\right)\sin\left(\frac{k\pi x}{\ell}\right)\,dx$$

without giving specific values to m and k.

Part (iii) is worth investigating, as it flows directly out of our work in Chapter 4. Any $p \in \text{span}(T_n(\ell))$ is a linear combination of functions in $T_n(\ell)$:

$$\begin{aligned} p(x) &= A_0 \cdot 1 + a_1 \cos\left(\frac{\pi x}{\ell}\right) + \cdots + a_n \cos\left(\frac{n\pi x}{\ell}\right) + b_1 \sin\left(\frac{\pi x}{\ell}\right) + \cdots + b_n \sin\left(\frac{n\pi x}{\ell}\right) \\ &= A_0 c_0(x) + a_1 c_1(x) + a_2 c_2(x) + \cdots + a_n c_n(x) + b_1 s_1(x) + b_2 s_2(x) + \cdots + b_n s_n(x), \end{aligned}$$

where I have abbreviated $c_0(x) = 1$, $c_j(x) = \cos(j\pi x/\ell)$ and $s_j(x) = \sin(j\pi x/\ell)$, for $j = 1, \ldots, n$. On the heels of our work for projections in function space, the closest $p \in \text{span}(T_n(\ell))$ to f is found by solving the normal equations

$$\begin{bmatrix} \langle f, c_0\rangle \\ \langle f, c_1\rangle \\ \vdots \\ \langle f, c_n\rangle \\ \langle f, s_1\rangle \\ \vdots \\ \langle f, s_n\rangle \end{bmatrix} = \begin{bmatrix} \langle c_0, c_0\rangle & \langle c_0, c_1\rangle & \cdots & \langle c_0, c_n\rangle & \langle c_0, s_1\rangle & \cdots & \langle c_0, s_n\rangle \\ \langle c_1, c_0\rangle & \langle c_1, c_1\rangle & \cdots & \langle c_1, c_n\rangle & \langle c_1, s_1\rangle & \cdots & \langle c_1, s_n\rangle \\ \vdots & \vdots & \ddots & & \vdots & & \vdots \\ \langle c_n, c_0\rangle & \langle c_n, c_1\rangle & \cdots & \langle c_n, c_n\rangle & \langle c_n, s_1\rangle & \cdots & \langle c_n, s_n\rangle \\ \langle s_1, c_0\rangle & \langle s_1, c_1\rangle & \cdots & \langle s_1, c_n\rangle & \langle s_1, s_1\rangle & \cdots & \langle s_1, s_n\rangle \\ \vdots & \vdots & & & \vdots & \ddots & \vdots \\ \langle s_n, c_0\rangle & \langle s_n, c_1\rangle & \cdots & \langle s_n, c_n\rangle & \langle s_n, s_1\rangle & \cdots & \langle s_n, s_n\rangle \end{bmatrix} \begin{bmatrix} A_0 \\ a_1 \\ \vdots \\ a_n \\ b_1 \\ \vdots \\ b_n \end{bmatrix}$$

$$= \begin{bmatrix} 2\ell & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & \ell & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots & & \vdots \\ 0 & 0 & \cdots & \ell & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & \ell & \cdots & 0 \\ \vdots & \vdots & & & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & \ell \end{bmatrix} \begin{bmatrix} A_0 \\ a_1 \\ \vdots \\ a_n \\ b_1 \\ \vdots \\ b_n \end{bmatrix},$$

where the orthogonality of functions in $T_n(\ell)$ leads to the diagonal matrix above, whose diagonal entries are nonzero, making it nonsingular. We invert to find expressions for the coefficients:

$$\begin{bmatrix} A_0 \\ a_1 \\ \vdots \\ a_n \\ b_1 \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} 1/(2\ell) & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 1/\ell & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots & & \vdots \\ 0 & 0 & \cdots & 1/\ell & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1/\ell & \cdots & 0 \\ \vdots & \vdots & & & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 1/\ell \end{bmatrix} \begin{bmatrix} \langle f, c_0\rangle \\ \langle f, c_1\rangle \\ \vdots \\ \langle f, c_n\rangle \\ \langle f, s_1\rangle \\ \vdots \\ \langle f, s_n\rangle \end{bmatrix} = \begin{bmatrix} \frac{1}{2\ell}\langle f, c_0\rangle \\ \frac{1}{\ell}\langle f, c_1\rangle \\ \vdots \\ \frac{1}{\ell}\langle f, c_n\rangle \\ \frac{1}{\ell}\langle f, s_1\rangle \\ \vdots \\ \frac{1}{\ell}\langle f, s_n\rangle \end{bmatrix}.$$

Reading the expressions at the two ends, the expected formulas for $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$ are there. The only strange one is the formula for A_0, the coefficient for $c_0(x) = 1$:

$$A_0 = \frac{1}{2\ell}\langle f, c_0\rangle.$$

If we define $a_0 = 2A_0$, then

$$a_0 = \frac{1}{\ell}\langle f, c_0\rangle,$$

and now $(a_0/2)$ is the coefficient of $c_0(x)$, in accordance with formulas (2) and (3). □

Several remarks are in order:

- The interval that serves as the domain above, for a given $\ell > 0$, is $(-\ell, \ell)$. That is an interval in which the functions in $T_n(\ell)$ are mutually orthogonal, but it is not the only one. These same functions are mutually orthogonal under the $L^2(0, 2\ell)$ inner product:

$$\langle g_1, g_2\rangle = \int_0^{2\ell} g_1(x)g_2(x)\,dx.$$

  In fact, given any interval (a, b) with $b-a = 2\ell$, the functions of $T_n(\ell)$ are mutually orthogonal under the $L^2(a, b)$ inner product. In Section 10.5, Strang's discussion focus on functions defined on $[0, 2\pi]$, whereas following my paradigm we would look at the domain $[-\pi, \pi]$.

- The most important case, the case that Fourier himself investigated in the early 1800s, is when $\ell = \pi$. In that case, the orthogonal basis is

$$T_n(\pi) = \{1, \cos(x), \cos(2x), \ldots, \cos(nx), \sin(x), \sin(2x), \ldots, \sin(nx)\}.$$

- In mathematics, the word *series* refers to an infinite sum. True Fourier series is no exception. The orthogonality of the functions in $T_n(\ell)$ is not lost as n grows. Letting $n \to \infty$, the sum (2) becomes the infinite series

$$\frac{a_0}{2} + \sum_{n=1}^{\infty}\left[a_n \cos\left(\frac{n\pi x}{\ell}\right) + b_n \sin\left(\frac{n\pi x}{\ell}\right)\right], \tag{4}$$

  with the values of the coefficients a_n, b_n the same as before, given in Equation (3). One of the great open questions that dogged mathematics for perhaps 150 years following the introduction of Fourier series to find under what circumstances does the series (4) *equal* the function f that generated it, within the confines of the interval $(-\ell, \ell)$. That mystery has been cleared up, with one answer provided by the next theorem.

Theorem 2: Suppose f is a piecewise $\mathscr{C}^1$ function on $[-\ell, \ell]$, and define $\overline{f}(x)$ to be the sum (4) at x. Then for each $x \in (-\ell, \ell)$,

$$\overline{f}(x) = \frac{1}{2}\left[\lim_{t\to x^-} f(t) + \lim_{t\to x^+} f(t)\right],$$

while

$$\overline{f}(-\ell) = \overline{f}(\ell) = \frac{1}{2}\left[\lim_{t\to -\ell^+} f(t) + \lim_{t\to \ell^-} f(t)\right].$$

That is, the series (4) converges to the average of the left- and right-hand limiting values of f at each point, and equals f at every (interior) point of continuity in $(-\ell, \ell)$.

Computational Fourier Series

In what follows, I use a couple of scripts, `fourierCoeffs.m` and `fsApprox.m`, which can be downloaded from the website https://calvin.edu/~scofield/courses/m355/materials/octave/.

Some tips about defining functions and making plots in Octave

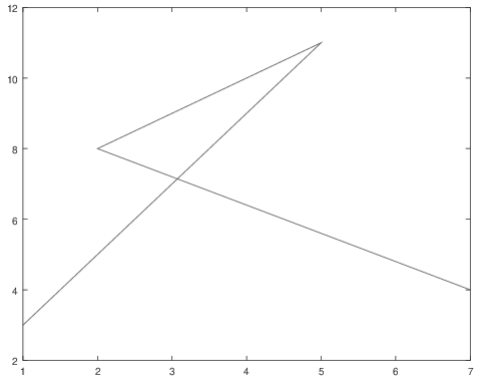
In Octave, a command like

```
octave:1> cos([1  2  3])
ans =

   0.54030  -0.41615  -0.98999
```

shows performs something like parallel evaluation of built-in mathematical functions when multiple inputs (stored in a vector) are given.

Octave does allow one to plot functions in a manner similar to graphing calculators—where you give a formula and limits on the viewing window. But Octave's `plot()` command was written for situations in which one has the coordinates of points stored in vectors or matrices.

```
octave:2> xs = [1 5 2 7];
octave:3> ys = [3 11 8 4];
octave:4> plot(xs, ys)
```



You see that lines are added by default, connecting consecutive points. There are variants you should try out, which alter this behavior in various ways. Try them out, after creating like-dimensioned vectors `xs` and `ys` as above.

```
octave:10> plot(xs, ys, 'k.', "markersize", 16)      # k stands for black, the . indicates points only
octave:14> plot(xs, ys, 'r-.')          # r stands for red, -. is dash-dot pattern
```

While built-in functions can evaluate a full vector at a time, user-defined functions must be implemented appropriately to do it. Nothing special is required to implement the very simple function $f(x) = 3x$, besides what was demonstrated in class:

```
octave:21> f = @(x) 3*x
f =

@(x) 3 * x

octave:22> f([3 -1 8])      # vector evaluation goes smoothly
ans =

    9   -3   24
```

But things get a bit interesting with a function like $f(x) = (x^2 + 1)/\sqrt{x}$.

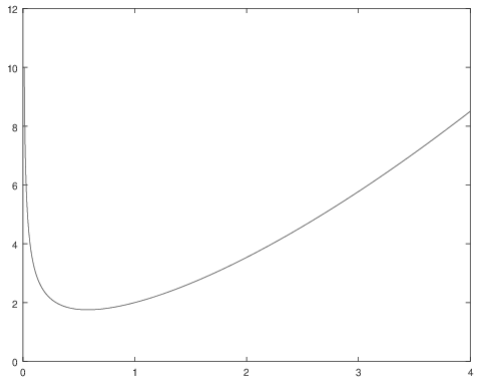
```
octave:23> f = @(x) (x^2 + 1) / sqrt(x)      # Naive implementation
f =

@(x) (x ^ 2 + 1) / sqrt (x)

octave:24> f(2)
ans = 3.5355
octave:25> f([1 2 3])
error: for x^A, A must be a square matrix. Use .^ for elementwise power.
error: called from
    @<anonymous> at line 1 column 20
```

The function definition was sufficient to evaluate f at a single number, 2. But it balked at evaluating a vector. The issue is that, when we multiply the vector `x` with itself to get $\mathbf{x}^2$, we want it to do the multiplication coordinate-wise, something that isn't naturally done when multiplying matrices. And division of matrices/vectors is not natural, let alone performed coordinate-wise. To indicate that these operations should be done coordinate-wise, we add a . (dot) in front of the operation:

```
octave:25> f = @(x) (x.^2 + 1) ./ sqrt(x)
f =

@(x) (x .^ 2 + 1) ./ sqrt (x)

octave:26> f([1 2 3])
ans =

   2.0000   3.5355   5.7735
```

We can now evaluate the function on an entire vector of inputs in one step:

```
octave:29> xs = 0:.01:4;
octave:30> plot(xs, f(xs))
```



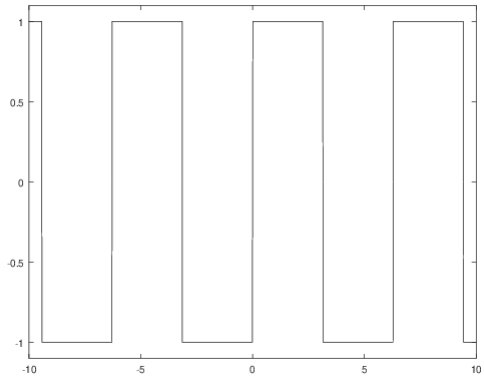
Using the Octave Fourier series scripts

In Section 10.5, Strang mentions a "square wave" function, a function defined on $[-\pi, \pi)$ in this manner:

$$f(x) = \begin{cases} -1, & \text{if } -\pi \leqslant x < 0, \\ 1, & \text{if } 0 \leqslant x < \pi. \end{cases}$$

One can use a combination of the `sin()` and `sign()` functions in Octave to define this f:

```
octave:34> f = @(x) sign(sin(x));
octave:35> xs=-10:.01:10;
octave:36> plot(xs, f(xs))      # deceptive vertical lines
octave:37> plot(xs, f(xs), 'b.')    # truer graph
octave:40> axis([-10 10 -1.1 1.1])      # adjusts viewing window
```



The purpose of the script called `fourierCoeffs.m` is to calculate the coefficients a_n, b_n for the projection into span$(T_n(\ell))$, the linear combination of Equation (2). We display the contents of the `help` for this function:

```
octave:46> help fourierCoeffs
'fourierCoeffs' is a function from the file /Users/scofield/Sites/courses/m355/materials/octave/fourierCoeffs.m

 function [a, b] = fourierCoeffs(f, k, r_end = pi)

   This routine computes the classical Fourier series coefficients
   for f up to order k on the interval [-r_end, r_end].

   INPUTS:
     f       a function handle, pointing to the function to approximate
     k       the number of sine terms to include
     r_end   the right endpoint for the spatial domain; defaults to pi

   OUTPUTS:
      a      the coefficients of the cosine terms
      b      the coefficients of the sine terms
```

If we want to project the square wave f into span$(T_9(\pi))$, we need the values of coefficients $a_0, a_1, \ldots, a_9$ and $b_1, \ldots, b_9$.

```
octave:49> fourierCoeffs(f, 9);
octave:50> a'
ans =

   0   0   0   0   0   0   0   0   0   0

octave:51> b'
ans =

 Columns 1 through 7:
```

```
  1.2732e+00   1.0900e−16   4.2441e−01   7.9132e−17   2.5465e−01   3.5715e−16   1.8189e−01

Columns 8 and 9:

 −5.6485e−16  1.4147e−01
```

The values $a_0, \ldots, a_9$ appear sequentially in the vector `a`, and similarly $b_1, \ldots, b_9$ are in `b`. Despite the fact that Strang works on $[0, 2\pi]$ instead of $[-\pi, \pi]$ the particular f involved should result in the same Fourier coefficients. Indeed, if we take any coefficient smaller than 10^{-10} in our output to be zero, then our best approximation to f in $\text{span}(T_9(\pi))$ is

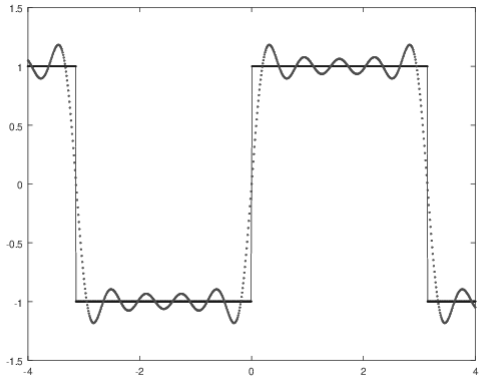
$$p(x) \;=\; 1.2732 \sin x + 0.42441 \sin(3x) + 0.25465 \sin(5x) 0.18189 \sin(7x) + 0.14147 \sin(9x),$$

which matches the coefficients Strang gives in Equation (8) on p. 492, repeated here:

$$\frac{4}{\pi}\left[\frac{\sin x}{1} + \frac{\sin(3x)}{3} + \frac{\sin(5x)}{5} + \cdots\right] \;=\; \frac{4}{\pi}\sin x + \frac{4}{3\pi}\sin(3x) + \frac{4}{5\pi}\sin(5x) + \cdots.$$

The other script, `fsApprox.m`, allows you to evaluate the projected function $p(x)$ at various x-values. It does the job of computing the Fourier coefficients allowing you to bypass that step (but it still uses `fourierCoeffs.m` in the process, so that script must be available). Let's use it and compare how similar (or not) the projection p and the square wave f are:

```
octave:54> xs = −4:.01:4;
octave:55> ys = fsApprox(f, xs, 9);
octave:56> plot(xs, f(xs), "b.−")
octave:57> hold on    # allows addition of another plot
octave:58> plot(xs, ys, "r.")
octave:59> hold off
```



Anything in $\text{span}(T_n(\ell))$ is, by the very nature of the basis functions, (2ℓ)-periodic. Of course, the square wave function above was also (2ℓ)-periodic. The result of projecting a *non-periodic* f into $\text{span}(T_n(\ell))$ may be predictable, but is worth seeing, anyway. Consider the function e^x on the interval $[-1, 1]$. We plot it on the larger interval $[-2, 2]$, along with its projection into $\text{span}(T_{20}(1))$:

```
octave:68> xs = −2:.01:2;
octave:69> plot(xs, exp(xs))
octave:70> hold on
octave:71> plot(xs, fsApprox(@(x) exp(x), xs, 20, 1), 'r')
octave:72> hold off
```