Listing 9.4 A program that displays a Fourier series.

```
package org.opensourcephysics.sip.ch09;
import org.opensourcephysics.controls.*;
import org.opensourcephysics.display.FunctionDrawer;
import org.opensourcephysics.frames.DisplayFrame;

public class SynthesizeApp extends AbstractCalculation {
   DisplayFrame frame = new DisplayFrame("x", "f(x)",
        "Fourier Synthesis");

   public void calculate() {
      double xmin = control.getDouble("xmin");
      double xmax = control.getDouble("xmax");
      int N = control.getInt("N");
      double period = control.getDouble("period");
      double[] sinCoefficients =
           (double[]) control.getObject("sin coefficients");
      double[] cosCoefficients =
           (double[]) control.getObject("cos coefficients");
      FunctionDrawer functionDrawer = new FunctionDrawer(new Synthesize
           (period, 0, cosCoefficients, sinCoefficients));
      functionDrawer.initialize(xmin, xmax, N, false);
      frame.clearDrawables();          // remove old function drawer
      frame.addDrawable(functionDrawer); // add new function drawer
   }

   public void reset() {
      control.setValue("xmin", -1);
      control.setValue("xmax", 1);
      control.setValue("N", 300);
      control.setValue("period", 1);
      control.setValue("sin coefficients", new double[]
           {1.0, 0, 1.0/3.0, 0, 1.0/5.0, 0, 0});
      control.setValue("cos coefficients", new double[]
           {0, 0, 0, 0, 0, 0, 0});
      calculate();
   }

   public static void main(String[] args) {
      CalculationControl.createApp(new SynthesizeApp());
   }
}
```

Problem 9.9 Fourier synthesis

(a) The process of approximating a function by adding together a fundamental frequency and harmonics of various amplitudes is *Fourier synthesis*. The `SynthesizeApp` class shows how a sum of harmonic functions can represent an arbitrary periodic function. Consider the series

$$f(t) = \frac{2}{\pi}\left(\sin t + \frac{1}{3}\sin 3t + \frac{1}{5}\sin 5t + \cdots\right). \tag{9.28}$$

Describe the nature of the plot of $f(t)$ when only the first three terms in (9.28) are retained. Increase the number of terms until you are satisfied that (9.28) represents

the desired function with sufficient accuracy. What function is represented by the infinite series?

(b) Modify `SynthesizeApp` so that you can create an initial state with an arbitrary number of terms. (Do not use the control to input each coefficient. Input only the total number of terms.) Consider the series (9.28) with at least 32 terms. For what values of t does the finite sum most faithfully represent the exact function? For what values of t does it not? Why is it necessary to include a large number of terms to represent $f(t)$ where it has sharp edges? The small oscillations that increase in amplitude as a sharp edge is approached are known as the *Gibbs phenomenon*.

(c) Modify `SynthesizeApp` and determine the function that is represented by the Fourier series with coefficients $a_k = 0$ and $b_k = (2/k\pi)(-1)^{k-1}$ for $k = 1, 2, 3, \ldots$. Approximately how many terms in the series are required? ■

So far we have considered how a sum of sines and cosines can approximate a known periodic function. More typically, we generate a time series consisting of N data points $f(t_i)$, where $t_i = 0, \Delta, 2\Delta, \ldots, (N-1)\Delta$. The Fourier series approximation to this data assumes that the data repeats itself with a period T given by $T = N\Delta$. Our goal is to determine the Fourier coefficients a_k and b_k. We will see that these coefficients contain important physical information.

Because we know only a finite number of data points $f_i \equiv f(t_i)$, it is possible to find only a finite set of Fourier coefficients. For a given value of Δ, what is the largest frequency component we can extract? In the following, we give a plausibility argument that suggests that the maximum frequency we can analyze is given by

$$\omega_Q \equiv \frac{\pi}{\Delta} \quad \text{or} \quad f_Q \equiv \frac{1}{2\Delta} \quad \text{(Nyquist frequency)}, \tag{9.29}$$

where f_Q is the Nyquist frequency.

One way to understand (9.29) is to analyze a sine wave, $\sin \omega t$. If we choose $\Delta = \pi/\omega$, that is, sample at the Nyquist frequency, we would sample two points per cycle. If we sample at or below this frequency, we sample during the sine's positive and negative displacement. If we sample above this frequency, we would sample during the positive displacement of one crest and the positive displacement of another crest, thereby completely missing the negative displacement. In Problem 9.10 we explore sampling at frequencies higher than the Nyquist frequency.

Problem 9.10 Nyquist demonstration

Run `OscillatorsApp` with 16 particles in normal mode 27. What is the nature of the motion that you observe? Because the particles sample the analytic function $f(x)$ which sets the initial position at intervals Δ that are larger than $1/(2\lambda)$, the particle positions cannot accurately represent the high spatial frequencies. What mode number corresponds to sampling at the Nyquist frequency? ■

The Nyquist frequency is very important in signal processing because the sampling theorem due to Nyquist and Shannon states that a continuous function is completely determined by N sampling points if the signal has passed though a filter with a cutoff frequency less than f_Q. This theorem is easy to understand. The filter will take out all the high frequency modes and allow only N modes of the signal to pass through. With N sampling points we can solve for the amplitudes of each mode using (9.35) (also see Section 9.6).