Because of the large number of coefficients $c_{n,m}$, the discrete two-dimensional Fourier transform is best implemented using the FFT algorithm. The FFT2DCalculationApp program shows how to compute a two-dimensional FFT using the FFT2DFrame utility class.

Listing 9.9 The FFT2DCalculationApp program computes the two-dimensional fast Fourier transform of a function and shows the resulting coefficients using a grid plot.

```
package org.opensourcephysics.sip.ch09;
import org.opensourcephysics.controls.*;
import org.opensourcephysics.frames.FFT2DFrame;

public class FFT2DCalculationApp extends AbstractCalculation {
   FFT2DFrame frame = new FFT2DFrame("kx", "ky", "2D FFT");

   public void calculate() {
      int xMode = control.getInt("x mode");
      int yMode = control.getInt("y mode");
      double xmin = control.getDouble("xmin");
      double xmax = control.getDouble("xmax");
      int nx = control.getInt("Nx");
      double ymin = control.getDouble("ymin");
      double ymax = control.getDouble("ymax");
      int ny = control.getInt("Ny");
      // data stored in row-major format
      double[] zdata = new double[2*nx*ny];
      double y = 0, yDelta = 2*Math.PI/ny;
      for(int iy = 0;iy<ny;iy++) { // loop over rows in array
         // offset to beginning of a row; each row is nx long
         int offset = 2*iy*nx;
         double x = 0, xDelta = 2*Math.PI/nx;
         for(int ix = 0;ix<nx;ix++) {
            // z function is e^(i*xmode*x)e^(i*ymode*y)
            zdata[offset+2*ix] =
                  Math.cos(xMode*x)*Math.cos(yMode*y)
                  -Math.sin(xMode*x)*Math.sin(yMode*y); //real part
            zdata[offset+2*ix+1] =
                  Math.sin(xMode*x)*Math.cos(yMode*y)
                  +Math.cos(xMode*x)*Math.sin(yMode*y); // imaginary
                      part
            x += xDelta;
         }
         y += yDelta;
      }
      frame.doFFT(zdata, nx, xmin, xmax, ymin, ymax);
   }

   public void reset() {
      control.setValue("x mode", 0);
      control.setValue("y mode", 1);
      control.setValue("xmin", 0);
      control.setValue("xmax", "2*pi");
      control.setValue("ymin", 0);
      control.setValue("ymax", "2*pi");
      control.setValue("Nx", 16);
      control.setValue("Ny", 16);
   }
```



```
   public static void main(String[] args) {
      CalculationControl.createApp(new FFT2DCalculationApp());
   }
}
```

The FFT2DFrame is based on FFT routines contributed to the GNU Scientific Library (GSL) by Brian Gough and adapted to Java by Bruce Miller at NIST. We initialize our data array to conform to the GSL API using a one-dimensional array such that rows follow sequentially. This ordering is known as row-major format. Because the input function is assumed to be complex, the array has dimension $2N_xN_y$, where N_x and N_y are the number of grid points in the x and y direction, respectively. The FFT2DFrame object transforms and displays the data when the doFFT method is invoked.

Exercise 9.16 Two-dimensional FFT
Write a program to transform a two-dimensional Gaussian using the FFT2DFrame class. Note that color is used to represent the complex phase. What happens if the Gaussian is not centered on the grid? ■

9.5 ■ FOURIER INTEGRALS

Fourier analysis can be extended to approximate waveforms that do not repeat themselves by converting the Fourier sum over discrete frequency components to an integral. The *Fourier integral* transforms a continuous function of time $f(t)$ into a continuous function of frequency $g(\omega)$ as follows:

$$g(\omega) = \int_{-\infty}^{\infty} f(t)e^{i\omega t}\,dt. \tag{9.44}$$

The inverse transformation reverses this process:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} g(\omega)e^{i\omega t}\,d\omega. \tag{9.45}$$

Equations (9.44) and (9.45) are known as a Fourier transform pair.

Because we need to store functions such as $f(t)$ and $g(\omega)$ at a discrete number of points, Fourier integrals are usually approximated using Fourier series with a large number of terms and a large sampling time. In other words, the time over which the signal is measured is large compared to the period of interest. Exercise 9.17 illustrates how a Fourier series for a pulse train approaches a continuous frequency spectrum as the period approaches infinity.

Exercise 9.17 Fourier integral
Write a program to plot the frequency spectrum of the waveforms shown in Figure 9.2. Use the FFT algorithm. How does the frequency spectrum change as the waveform becomes less periodic? How does the finite time interval affect the result? ■