Problem 9.1 Normal modes

(a) How many modes are there for a chain of 16 oscillators? Predict the initial positions of the oscillators for modes 1 and 14 and compare your prediction to the program's output.

(b) Because a normal mode is a standing wave, it can be written as the sum of right and left traveling sinusoidal waves, $f_+ = A\sin(kx - \omega t)$ and $f_- = A\sin(kx + \omega t)$, respectively. Does the phase velocity $v = \omega/k$ depend on the mode (wavelength)? In other words, how does the speed depend on the wavelength (see (9.9))?

(c) Determine the wavelength and frequency for modes with the largest and smallest wavelengths. Note that you can click-drag within the window to measure position. The time is displayed in the yellow message box. Compare your measured values with (9.9).

(d) Are negative mode numbers acceptable? Give two different listings of mode numbers that contain a complete set of modes for $N = 16$.

(e) Write a short stand-alone program to verify that the normal mode solutions (9.10) are orthonormal.

(f) Modify your program to show the evolution of the superposition of two or more normal modes. Compare this evolution to that of a single mode. ■

As seen in Problem 9.1, the number of unique nontrivial solutions is equal to the number of oscillators. Modes with mode numbers $1, 2, 3, \ldots, N$ form a complete set and all other modes are indistinguishable from them. (The number of modes is related to the Nyquist frequency and is discussed further in Section 9.3.)

The analytic solution (9.11), together with the initial conditions, represent the complete solution of the displacement of the particles. We can use a computer to calculate the sum in (9.11) and plot the time dependence of the displacements $u_j(t)$. There are many interesting extensions that are amenable to an analytic solution. What is the effect of changing the boundary conditions? What happens if the spring constants are not all equal but are chosen from a probability distribution? What happens if we vary the masses of the particles? For these cases we can follow a similar approach and look for the eigenvalues ω_n and eigenvectors $u_{j,n}$ of the matrix equation

$$\mathbf{T}\,\mathbf{u} = \omega^2\,\mathbf{u}. \tag{9.18}$$

The matrix elements $T_{i,j}$ are zero except for

$$T_{i,i} = \frac{1}{m_i}[k_{i,i+1} + k_{i,i-1}] \tag{9.19a}$$

$$T_{i,i+1} = -\frac{k_{i,i+1}}{m_i} \tag{9.19b}$$

$$T_{i,i-1} = -\frac{k_{i,i-1}}{m_i}, \tag{9.19c}$$

where $k_{i,j}$ is the spring constant between particles i and j. The solution of matrix equations such as (9.18) is a well-studied problem in linear programming, and an open source library

such as LINPAC available from NetLIB (<www.netlib.org>) or a stand-alone program such as Octave (<www.octave.org>) can be used to obtain the solutions.

9.2 ■ NUMERICAL SOLUTIONS

Because we are also interested in the effects of nonlinear forces between the particles, for which the matrix approach is inapplicable, we study the numerical solution of the equations of motion (9.1) directly.

To use the ODE interface, we need to remember that the ordering of the variables in the coupled oscillator state array is important because the implementations of some ODE solvers, such as Verlet and Euler–Richardson, make explicit assumptions about the ordering. Our standard ordering is to follow a variable by its derivative. For example, the state vector of an N oscillator chain is ordered as $\{u_0, v_0, u_1, v_1, \ldots, u_N, v_N, u_{N+1}, v_{N+1}, t\}$. Note that the state array includes variables for the chain's end points although the velocity rate corresponding to the end points is always zero. We include the time as the last variable because we will sometimes model time-dependent external forces. With this ordering, the `getRate` method is implemented as follows:

```
static final double OMEGA_SQUARED = 1; // equals k/m
public void getRate(double[] state, double[] rate) {
   for(int i = 1, N = x.length-1; i<N; i++) { // skip ends
      rate[2*i] = state[2*i+1]; // displacement rate
      rate[2*i+1] =
           -OMEGA_SQUARED*(2*state[2*i]-state[2*i-2]-state[2*i+2]);
   }
   rate[state.length-1] = 1;
}
```

Problem 9.2 Numerical solution

(a) Modify the `Oscillators` class to solve the dynamical equations of motion by implementing the `ODE` interface. Compare the numerical and the analytic solution for $N = 10$ using an algorithm that is well suited to oscillatory problems.

(b) What is the maximum deviation between the analytic and numerical solution of $u_j(t)$? How well is the total energy conserved in the numerical solution? How does the maximum deviation and the conservation of the total energy change when the time step Δt is reduced? Justify your choice of numerical algorithm. ■

Problem 9.3 Dynamics of coupled oscillators

(a) Use your program for Problem 9.2 for $N = 2$ using units such that the ratio $k/m = 1$. Choose the initial values of u_1 and u_2 so that the system is in one of its two normal modes, for example, $u_1 = u_2 = 0.5$ and set the initial velocities equal to zero. Describe the displacement of the particles. Is the motion of each particle periodic in time? To answer this question, add code that plots the displacement of each particle versus the time. Then consider the other normal mode, for example, $u_1 = 0.5$, $u_2 = -0.5$. What is the period in this case? Does the system remain in a normal mode indefinitely? Finally, choose the initial particle displacements equal to random values between -0.5 and $+0.5$. Is the motion of each particle periodic in this case?