interactions in the latter are inelastic. The lost energy goes into the internal degrees of freedom of a grain and ultimately is dissipated. From the point of view of the motion of the granular particles, the energy is lost. Experimentalists have studied models of granular material composed of small steel balls or glass beads using sophisticated imaging techniques that can track the motion of individual particles. There have also been many complementary computer simulation studies.

What are some of the interesting properties of granular matter? Because the interactions are inelastic, granular particles will ultimately come to rest unless there is an external source of energy, usually a vibrating wall or gravity (for example, the fall of particles through a funnel). When granular particles come to rest, they can form a granular solid that is different than molecular solids. One difference is that there frequently exists a complex network of force lines within the solid. In addition, unlike ordinary liquids, the pressure does not increase with depth because the walls of the container help support the grains. As a consequence, sand flowing out of an aperture flows at a constant rate independent of the height of the sand above the aperture. For this reason sand is used in hour glasses. Another interesting property is that under some conditions, the large grains in a mixture of large and small grains can move to the top while the container is being vibrated—the "Brazil nut" effect. Under other conditions, the large grains might move to the bottom. What happens depends on the size and density of the large grains compared to the small grains (see Sanders et al.).

It is also known that there is a critical angle for the slope of a sand pile, above which the sand pile is unstable. This slope is called the angle of repose. These and many other effects have been studied using theoretical, computational, and experimental techniques.

The first step in simulating granular matter is to determine the effective force law between particles. For granular gases the details of the force do not influence the qualitative results, as long as the force is purely repulsive and short range, and there is some mechanism for dissipating energy. Common examples of force laws are spring-like forces with stiff spring constants and hard disks with inelastic collisions. For simplicity, we will consider the Lennard–Jones potential with a cut off at $r_c = 2^{1/6}$ so that the force is always repulsive. To remove energy during a collision, we will introduce a viscous damping force given by

$$f_{ij} = -\gamma(\mathbf{v}_{ij} \cdot \mathbf{r}_{ij})\frac{\mathbf{r}_{ij}}{r_{ii}^2}, \tag{8.59}$$

where the viscous damping coefficient $\gamma$ equals 100 in reduced units. A more realistic force model necessary for granular flow problems is given in Hirchfeld et al.

(a) Modify class `LJParticles` so that the cutoff is at $2^{1/6}$. Is the total energy conserved? Include a viscous damping force as in (8.59) and plot the kinetic energy per particle versus time. We will define the kinetic temperature to be the mean kinetic energy per particle. Why does this definition of temperature not have the same significance as the temperature in molecular systems in which the energy is conserved? Choose $N = 64$, $L = 20$, and $\Delta t = 0.001$. Begin with a random configuration and initial kinetic temperature equal to 10. How long does it take for the kinetic temperature to decrease to 10% of its initial value? Describe the spatial distribution of the particles at this time.

(b) Compute the mean kinetic temperature versus time averaged over three runs. What functional form describes your results for the mean kinetic temperature at long times?



(c) To prevent "granular collapse" where the particles ultimately come to rest, we need to add energy to the system. The simplest way of doing so is to give random kicks to randomly selected particles. You can use the same algorithm we used to set the initial velocities in `LJParticles`:

```
int i = (int)(N*math.random();  // selects random particle
// use to generate Gaussian distribution
double r = Math.random();
double a = -Math.log(r);
double theta = 2.0*Math.PI*Math.random();
// assign velocities according to Maxwell-Boltzmann distribution
// using Box-Muller method
state[4*i+1] = Math.sqrt(2.0*desiredKE*a)*Math.cos(theta);    // vx
state[4*i+3] = Math.sqrt(2.0*desiredKE*a)*Math.sin(theta);    // vy
```

(The Box–Mueller method is described in Section 11.5.) Assume that at each time step one particle is chosen at random and receives a random kick. Adjust `desiredKE` so that the mean kinetic energy per particle remains roughly constant at about 5.0. Compute the velocity distribution function for each component of the velocity. Compare this distribution on the same plot to the Gaussian distribution:

$$p(v_x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(v_x - \langle v_x \rangle)^2/2\sigma^2}, \tag{8.60}$$

where $\sigma^2 = \langle v_x^2 \rangle - \langle v_x \rangle^2$. Is the velocity distribution function of the system a Gaussian? If not, give a physical explanation for the difference. ■

## APPENDIX 8A: READING AND SAVING CONFIGURATIONS

For most of the problems in this chapter, qualitative results can be obtained fairly quickly. However, in research applications, the time for running a simulation is likely to be much longer than a few minutes, and runs that require days or even months are not uncommon. In such cases it is important to be able to save the intermediate configurations to prevent the potential loss of data in the case of a computer crash or power failure.

Also, in many cases it is easier to save the configurations periodically and then use a separate program to analyze the configurations and compute the quantities of interest. In addition, if we wish to compute averages as a function of a parameter, such as the temperature, it is convenient to make small changes in the temperature and use the last configuration from the previous run as the initial configuration for the simulation at the new temperature.

The standard Java API has methods for reading and writing files. The usual way of saving a configuration is to use these methods to simply write all the positions and velocities as numbers into a file. Additional simulation parameters and information about the configuration would be saved using a custom format. Although this approach is the traditional one for data storage, the use of a custom format means that you might not remember the format later, and sharing data between programs and other users becomes more difficult.

An alternative is to use a more structured and widely shared format for storing data. The Open Source Physics library has support for the Extensive Markup Language (XML). The XML format offers a number of advantages for computational physics: clear markup of