CHAPTER 6 - PERIODIC BOUNDARIES

In considering an ideal gas, we didn't worry about whether our numerical results depend upon the size of the container because the particles are pointlike. However, if the particles of the system interact with each other, even if through hard-core repulsion only, we must be concerned with effects arising from the finite system size. One way of reducing finite size effects is the use of periodic boundaries, in which a given sample is surrounded by replicas of itself. In this chapter, we introduce periodic boundary conditions and apply them to a non-ideal gas.

6.1 Non-ideal gas

The ideal gas law describes most gases fairly well at very low density, where the intermolecular separation is large. However, the law is less accurate at moderate density, for which intermolecular interactions are more common. In this section, we describe the deviations from ideal gas behavior expected at low density. With the aim of simulating a non-ideal gas, different computational approaches to minimizing boundary effects on small systems are presented in Secs. 6.2-6.4. One of the boundary schemes is implemented in the project of Sec. 6.5, which is a simulation of a simple, but not ideal, gas of hard spheres.

Virial expansion

The ideal gas law can be trivially rewritten as

$$P / k_B T = N / V. \qquad (6.1)$$

Although this equation is well-behaved at very low density, the relationship between P, V and T works less well at medium density. One way of parametrizing the deviation from ideal gas behavior at medium density is to expand Eq. (6.1) as a power series in the density:

$$P / k_B T = (N / V) + B_2(T)\,(N / V)^2 + B_3(T)\,(N / V)^3 + ... \qquad (6.2)$$

This equation is called the virial expansion, and the temperature-dependent terms are refered to as the virial coefficients. Specifically, $B_2(T)$ is the second virial coefficient and is the leading-order correction to the ideal gas law. Because it is more common to express the density in terms of moles per unit volume, rather than the number of molecules N per unit volume, the second virial coefficient in molar densities is equal to $N_o B_2$, where N_o is Avogadro's number.

Eq. (6.2) is not the only way of parametrizing deviations from ideal gas behavior. One approach due to van der Waals, which we develop in more detail below, recognizes that the non-zero size of molecules reduces the volume available for molecular motion. Van der Waals parametrized this reduction by writing a gas law of the form

$$P(V - Nv_{av}) = Nk_B T, \qquad (6.3)$$

where v_{av} is the average volume reduction per molecule associated with the molecular size. Van der Waals considered other effects as well, that are included in a more general form of Eq. (6.3) called the van der Waals equation.

The second virial coefficient can be expressed in terms of the parameter v_{av} by rearranging Eq. (6.3):

$$\begin{aligned} P / k_B T &= N / (V - Nv_{av}) \\ &= (N / V) / \ (1 - Nv_{av} / V) \\ &\quad (N / V) \bullet \ [1 + (N / V)v_{av}]. \qquad (6.4) \end{aligned}$$

Comparing with Eq. (6.2), we expect

$$B_2 = v_{av}. \qquad (6.5)$$

Second virial coefficient

In Eq. (6.3), the parameter v_{av} is the reduction in apparent volume per particle. The following argument provides an expression for v_{av} in terms of the molecular volume v_o. Consider the situation in which all molecules are spherical with diameter a, such that the molecular volume is

$$\begin{aligned} v_o &= (4\pi/3)(a/2)^3 \\ &= (\pi/6)a^3. \qquad (6.6) \end{aligned}$$

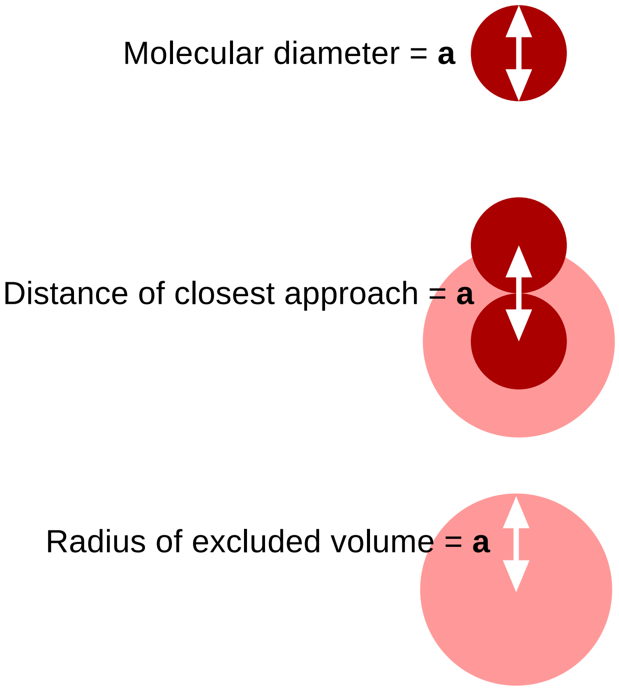

Molecular diameter = **a**
Distance of closest approach = **a**
Radius of excluded volume = **a**

Fig. 6.1 Two hard-sphere molecules cannot be separated by a center-to-center distance of less than *a*. Thus, the volume excluded to a pair of molecules is $v_{ex} = (4\pi/3)a^3$.

As illustrated in Fig. 6.1, the center-to-center distance between two molecules cannot be less than *a*, meaning that a given molecule is excluded from a volume v_{ex} around another molecule of

$$v_{ex} = (4\pi/3)a^3 = 8v_o. \tag{6.7}$$

That is, the volume reduction associated with each *pair* of molecules is $8v_o$. However, the average volume reduction *per molecule* is not $8v_o$.

Consider two molecules, *A* and *B*. The volume reduction experienced by either member of the pair is v_{ex}. But the total volume reduction of the pair must also be v_{ex}, and not $2v_{ex}$, since the excluded volume is the same region of space for both molecules. In other words, one must be careful not to double-count the same region of space in evaluating the excluded volume. Thus, the excluded volume *per molecule* is

$$v_{av} = v_{ex} / 2 = 4v_o. \tag{6.8}$$

Substituting for v_{av} and v_o, the excluded volume argument predicts that

$$B_2 = (2\pi/3)a^3. \tag{6.9}$$

[Once again, see the caveat following Eq. (6.2) about B_2 in molar units]. This prediction is tested in Project 6 by simulating a gas of hard spheres under pressure.

6.2 Periodic systems

Most macroscopic systems contain a very large number of particles which is beyond the size of system that is achievable or necessary in a computer simulation. Usually, the physical behavior of a system is apparent in a simulation with less than 10^5 particles; indeed, many properties can be obtained fairly accurately using just 10^3 particles in a three-dimensional system. In this section, we discuss periodic boundary conditions (*PBCs*), which is one of several techniques employed in the simulation of small systems.

A typical macroscopic system in three dimensions has perhaps 10^{21} molecules or more. On the order of $(10^{21})^{2/3} \sim 10^{14}$ of these molecules may be at the surface, or boundary, of the system. Although 10^{14} is a large number of molecules, they represent but a tiny fraction - $10^{14} / 10^{21} \sim 10^{-7}$ - of the total number of molecules in the system. Surface effects are usually not observable if the system has such a tiny fraction of its molecules at the surface. However, there are physical systems for which surface effects are very important. For example, even the largest atomic nuclei have only about 250 nucleons, of which a large fraction reside on the nuclear surface at any given time.

Consider a large cubical system, itself made of small cubical objects with sides of unit length. If there are 1000 small cubes in total, packed tightly together, then each surface is covered by 100 small cubes. Thus, there are about 500-600 small cubes at the surface (488 to be precise). In a typical three-dimensional simulation involving 10^3 particles, perhaps half of the particles will lie at or near the boundary of the system.

What does it matter that some particles are at the boundary while others are in the interior? In what way are particles at the boundary different from particles in the interior? Suppose that the particles are subject to short range forces of the square-well form discussed in Chap. 4, and reproduced in Fig. 6.2.

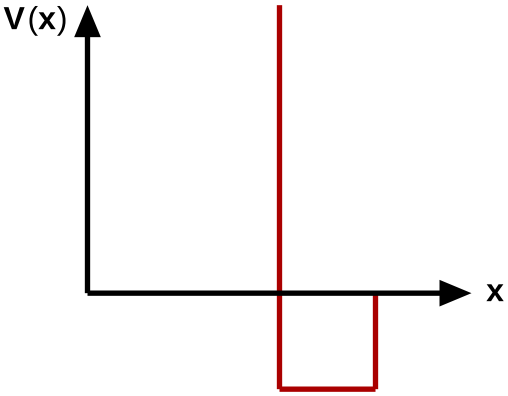


Fig. 6.2 Short-range square-well interaction potential.

If the system is dense, the mean interparticle separation lies in the attractive region of the potential. One would then expect the following:
•particles in the interior have attractive interactions with about 12 nearest neighbors
•particles on the surface have attractive interactions with about 9 nearest neighbors.
Hence, the surface particles are less deeply bound than the interior particles, and they experience an unbalanced force directed towards the interior. The unbalanced force gives the system a surface tension. The situation is illustrated for a two-dimensional system in Fig. 6.3.

Because surface effects are potentially important for simulations involving only hundreds or thousands of particles, measures have been developed to minimize the effects of boundaries. The most commonly used method involves the imposition of periodic boundary conditions, or *PBCs*, in which the finite system is imagined to be

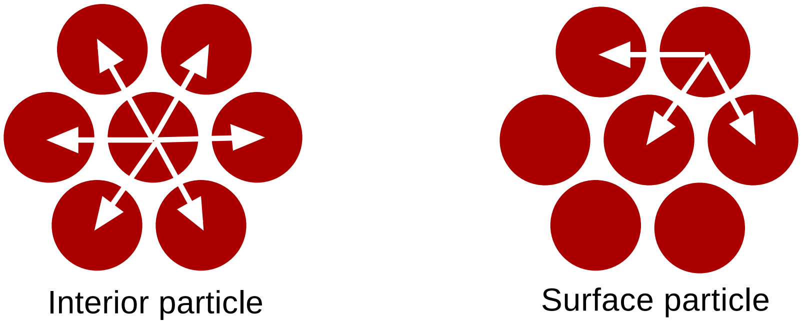


Fig. 6.3 Particles on the surface of a system are less well bound and experience unbalanced forces compared to those in the interior.

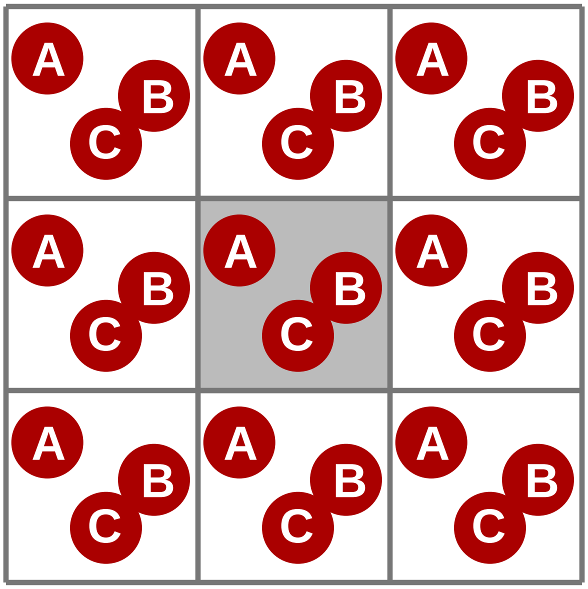


Fig. 6.4. The system or cell in the center, colored light gray, is surrounded by periodic images of itself.

surrounded by periodic replicas of itself. In principle, each particle may interact with other particles within the system, or with particles in the images. The system in Fig. 6.4 is somewhat small, in that each particle interacts with its neighbor and images of its neighbor: that is, particle A lies within interaction distance of particle B within its cell and with images of particle B in neighboring cells. In essence, periodic boundaries give the system a toroidal shape in a higher-dimensional space, as demonstrated in Fig. 6.5.

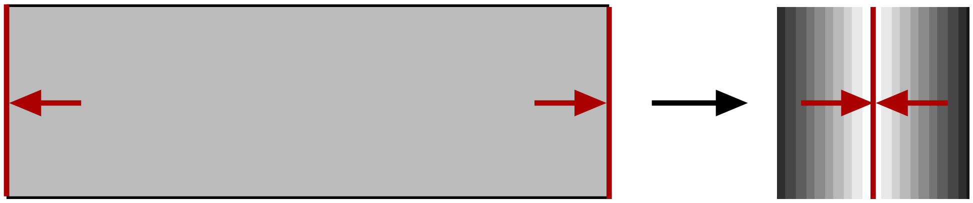


Fig. 6.5. Representation of periodic boundaries in a two-dimensional system by forming a torus in three-dimensional space.

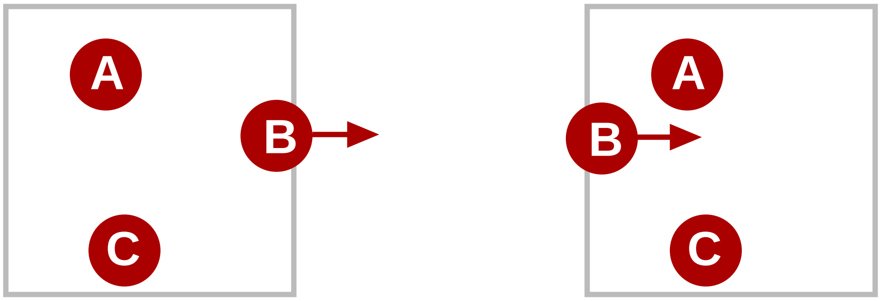


Fig. 6.6. Particle reaching the boundary of a periodic cell.

6.3 Rectangular cells

The cells shown in Figs. 6.4 and 6.5 are rectangular in shape, and are very commonly used for periodic systems. Other representations are pursued in Sec. 6.4. Figs. 6.4 and 6.5 not only illustrate how particles interact with their neighbors, they also show how particles should move through the system. Consider what happens when a particle, say B in Fig. 6.6, reaches the boundary of the cell. If the cell has a hard wall, then B will bounce off the wall, presuming that the wall isn't sticky. However, if the boundaries are periodic, then B will pass through the wall and reappear on the opposite side of the cell. Thus, the *x*, *y* or *z*-coordinates of the particle are shifted by one cell length in the appropriate direction when the particle crosses the cell wall.

There are a number of ways to implement rectangular PBCs. Often, the coordinate system is chosen such that the particles lie between -**box**/2 and +**box**/2, for a rectangular cell of length **box**. After each move, the position of a particle is checked and shifted so that the particle lies within the box:

```
hbox=0.5*box;
if(x > hbox) x=x-box;                                    (6.10)
if(x < -hbox) x=x+box;
```

The difference in the position of two particles is also subject to PBCs. Simply taking the difference between two position vectors does not necessarily yield the appropriate displacement vector. Consider the situation in Fig. 6.7. The shortest displacement between *A* and *B and their images* is not found within a cell, but rather between cells. Applying the obvious definition

$$\Delta \boldsymbol{r} = \boldsymbol{r}_\mathrm{B} - \boldsymbol{r}_\mathrm{A} \qquad (6.11)$$

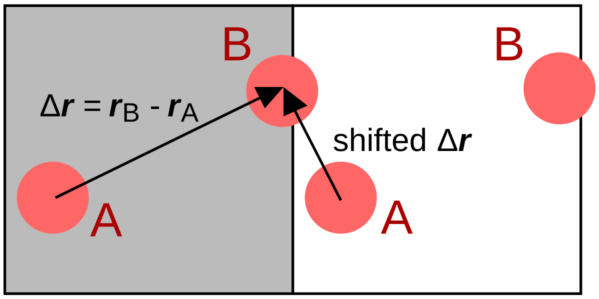

B
B
$\Delta \boldsymbol{r} = \boldsymbol{r}_B - \boldsymbol{r}_A$
shifted $\Delta \boldsymbol{r}$
A
A

Fig. 6.7 Vectors in a periodic system. The displacement vector $\Delta \boldsymbol{r} = \boldsymbol{r}_B - \boldsymbol{r}_A$ evaluated only within one cell (colored gray), is not the smallest one between *A* and *B*, if one takes the images into account. The shifted $\Delta \boldsymbol{r}$ is the smallest displacement between *A* and *B*, including the image positions.

does not yield the relevant displacement vector. First, the vector is too large, having a component in the *x*-direction that is more than half of the cell length. Second, it points in the wrong direction. The value of $\Delta \boldsymbol{r}$ obtained from Eq. (6.11) must be shifted by the same procedure as is used for the position vectors themselves, if one is to obtain the shortest distance between two specific particles in a periodic system:

```
hbox=0.5*box;
dx=x2-x1;
if(dx > hbox) dx=dx-box;                                    (6.12)
if(dx < -hbox) dx=dx+box;
```

Procedure (6.12) must be applied to each Cartesian coordinate.

In some situations, it may be efficient to store a particle's coordinates as a number between -0.5 and +0.5, and apply a scale factor to generate the "physical" coordinate only when needed. We call these *reduced* coordinates because they are dimensionless. As usual, both **x** and **dx** must lie in the range -0.5 to +0.5. However, if one uses the range 0 to 1 for **x**, then a separate procedure for shifting **dx** must be used to allow -0.5 < **dx** < 0.5. Whether this reduced coordinate system is computationally useful depends upon the problem at hand: if the code involves many calculations of physical positions, then reduced coordinates may not be efficient.

On some computers, codes with **if** statements run inefficiently, and means have been developed for avoiding **if** statements with periodic boundary conditions. However, these alternatives may be slower than using **if**s in many situations, and are not necessarily recommended by the author. Check your codes!

In C (invalid if **x** is more than **box**/2 away from of a boundary):

```
boxi = 1.0 / box;
x = x - box * (int)(2.0*x*boxi);      (-box/2 ≤ x ≤ +box/2)          (6.13)
```

In FORTRAN:

```
BOXI = 1.0 / BOX
X = X - BOX*ANINT(X * BOXI)           (-BOX/2 ≤ X ≤ +BOX/2)          (6.14)
X = X - ANINT(X)                      (-1/2 ≤ X ≤ +1/2)
```

6.4 Variations on PBCs

While rectangular boundaries may be fine for simulations of fluids, they may be inappropriate for solids, since the rectangular box shape may impose an artificial symmetry on the solid. A way of reducing the effects of the boundary symmetry on the system is to allow the prism to become a parallelopiped. The situation is illustrated for two dimensions in Fig. 6.8: one corner of the parallelogram is fixed at the origin while the two adjacent corners are allowed to move. Care must be taken to ensure that the configuration space of the box shapes is sampled randomly, and this can be accomplished by moving the corners of the box in Cartesian directions.

A method that eliminates the boundaries completely, at the expense of introducing other finite size effects, is the usage of so-called *spherical boundaries*.

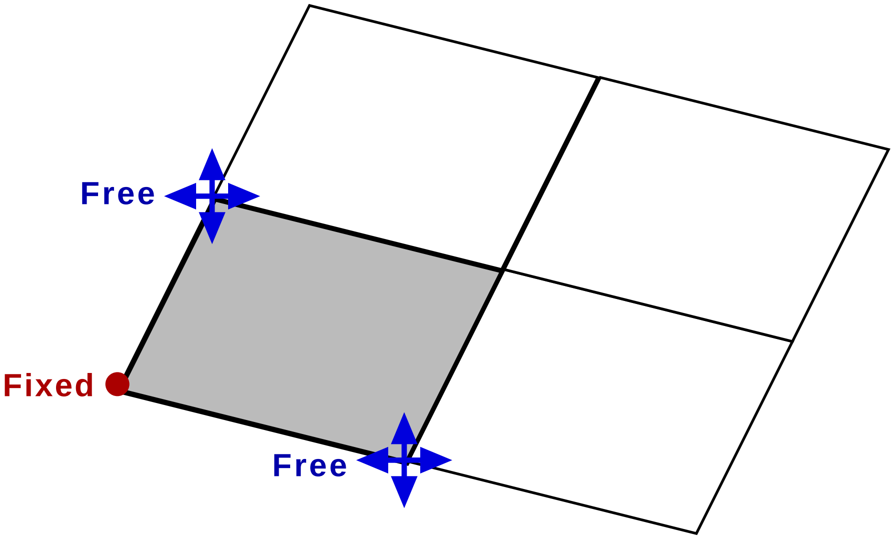


Fig. 6.8. Periodic, non-rectangular boundaries for a two-dimensional system.

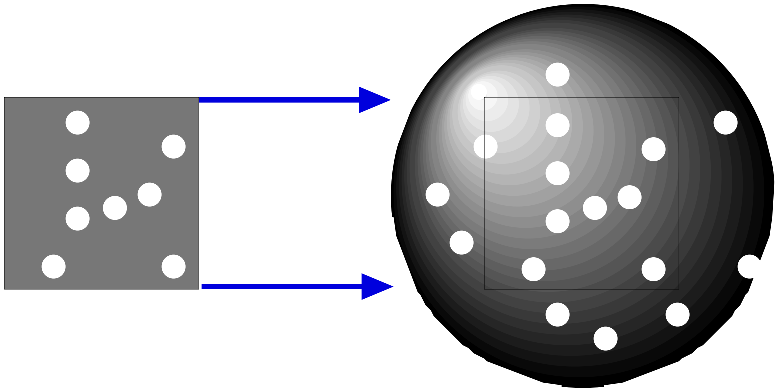


Fig. 6.9. Spherical boundary conditions, in which a two-dimensional system is placed onto the surface of a sphere.

Consider the two-dimensional planar system shown on the left-hand side of Fig. 6.9. With spherical boundary conditions, the particles of the system are placed on the edge-less surface of a sphere. The measurement of distances along the surface does not have the same geometry as it would on a plane, even if geodesics or "great circles" are used. However, this measurement issue becomes less important as the sphere is made larger in radius.

6.5 Project 6 - Hard sphere gas

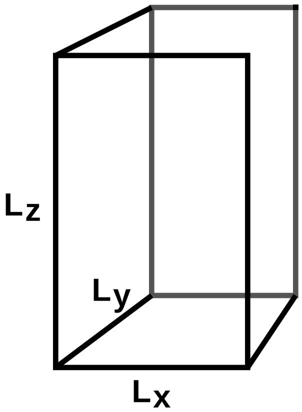
All of the coding elements that have been developed in the previous Chaps. 3-5 for Monte Carlo are brought together in Project 6 for the non-ideal gas. This project uses periodic boundary conditions and neighbor lists to simulate a system of particles which have hard-core geometry and which are subject to an external pressure. Once again, the author repeats the mantra that it is important to build codes from simple structures with compact notation that is easy to debug.

Physical system

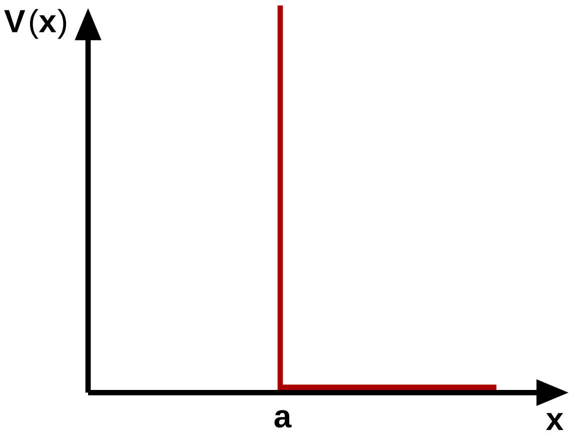
The system is a set of particles placed in a rectangular prism, as in Project 5. However, we now impose periodic boundaries on the system, so that a particle may pass by the boundary and appear through the opposite face of the cell: it is not stopped at the cell boundary, as in Project 6. The sides of the cells fluctuate at fixed pressure. All particles interact through a hard core potential, which causes the cell volume at a given pressure to increase above the corresponding ideal-gas volume.

Simulation parameters

A rectangular prism is set up in the same way as in Project 5, with sides of fluctuating length L_x, L_y, and L_z.



The particles are assigned a hard core radius *a*/2 through a potential energy function of the form



The hard core diameter *a* introduces a length scale into the system, unlike Project 5 where the only length scale was provided by the pressure. Thus, the pressure does not determine the length unit, but is an independent parameter, upon which the volume depends.

Code

1. Start with a copy of your code from Project 5, using notation and structure that is simple, transparent and efficient.

2. The particles in Project 5 were non-interacting, and did not pass through the prism walls. Make the boundaries periodic, allowing a particle to pass by a wall and appear through the opposite face of the cell. If there are too few particles in the system, then one of the box lengths may become so small that a particle begins to interact with its own periodic image. To avoid this situation, do not let a box side be less than 2*a*.

3. For each particle **i**, introduce a neighbor list which contains the labels of all particles within a distance of 2.5*a* of **i**. See warnings in Chap. 5 about the declared size of a neighbor list. Make sure that your calculation of interparticle displacements $\Delta\mathbf{r}$ respects the periodic boundaries, as in Eq. (6.13). Do not update the neighbor list after every sweep through the system, but update it only as often as is required by the particle position step size **ds/a** (see Chap. 4).

4. Introduce the hard core geometry of the particles. You may have to revise your particle initialization routine to make sure that the particles do not overlap in the initial configuration of the code.

5. Choose a variety of pressures to test out your code.

Analysis

1. Run two system sizes, choosing the number of particles *N* to be 64 and 216. Compare your results for these two *N*'s to check for finite system size effects. We may to pool the class data if the codes demand too much machine time.

2. Allow the system to relax, and then find the expectations $<V>$ and $<V^2>$ of the volume$V$ for two values of the pressure: βPa^3 = 0.05 and 0.1. The ensemble should have 300 configurations, each separated by 100*N* sweeps.

3. Extract the second virial coefficient using Eq. (6.2), and compare it with the theoretical expression in Eq. (6.9).

4. Determine the volume compression modulus from the fluctuation expression Eq. (5.28).

5. To gauge the accuracy of your results, analyse two or more 150-configuration subsets of your ensemble.

Report

Your report should include the following items:
•a brief description of the physical system that you are simulating
•an analytical calculation of the second virial coefficient for your system
•an outline of your code
•the data analysis that gives the virial coefficient and compression modulus
•a discussion of finite system size effects by comparing N = 64 and 216.
•an estimate of the accuracy of your results
•a copy of your code.

Demonstration code

The introduction of a hard-core interaction between particles will increase the volume of the system above the ideal-gas value, an effect that can be seen in the demonstration code for a two-dimensional system. You can choose among several pressures in starting the code, and then change the pressure while the code is running by clicking on the + or - boxes in the upper right-hand corner of the window. Each click will change $\beta \Pi a^2$ by 1/10 (Π since the problem is two-dimensional).

The physical size of the system adjusts as you change the pressure. However, the way the system is drawn on the screen is to fix the ideal gas value for the length scale $(N/\beta\Pi)^{1/2}$ at a fixed number of pixels (the 1/2 power arises because the system is two-dimensional). The ideal-gas length is indicated by the red bar above the black box representing the system. Changing the pressure *does* change the ideal-gas length, but *does not* change the size of the bar as it is drawn on the screen. In other words, the scale to which the image is drawn is pressure-dependent.

At low pressure, the system is dilute and close to an ideal gas. At high pressure, the system is physically much smaller, but so is the ideal gas length; however, the system is larger *relative* to the ideal-gas length. The change in the drawing scale can be seen by the size of the particles, which have a constant absolute length, but appear to change size with the pressure.