We have now a working code. What we need next is to set proper initial conditions for an equilibrium scenario, i.e. we need to simulate the system at a given density and temperature. We start with the positions. Our choice will be motivated by the following two considerations:

1. We do not want the particles to start too close to each other, otherwise the r^{-12}-repulsive potential can lead to huge contributions to the potential energy. Hence, putting particles on some sort of regular grid is sensible.

2. We want to simulate different phases of Argon. That is we want to be able to fix the density and the temperature of the system.

It is known that Argon forms a face-centered cubic lattice, see Fig. 1.

As we use periodic boundary conditions and as we want a regular arrangement of atoms in the solid phase, we best choose a total number of particles that is compatible with such a lattice. In short, we initially place the particles on a face-centered cubic lattice. Each repeating unit consists of 4 atoms. If we choose a total volume containing $3 \times 3 \times 3$ such units, we need $N = 108$ atoms. This turns out to be a reasonable size for a simulation (recall that at every time-step will require the computation of N^2 pairs of interactions). It is then straightforward to set the desired particle density ρ by choosing the appropriate volume of the simulation box.

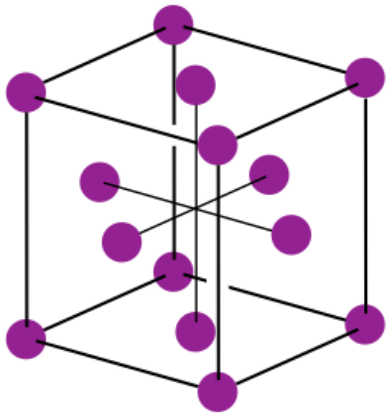

Figure 1: A unit cell of the face-centered cubic lattice. The full lattice is obtained by extending this unit in all directions.

Finally, we also need to set the temperature. The initial temperature should be chosen such that we get a Maxwell velocity distribution for that

given temperature. This is done by choosing the velocities components in x, y and z to be Gaussian distributed, e.g.

$$p\left(v_x\right) \sim e^{-mv_x^2/(2k_BT)}. \tag{1}$$

Question for you: What does this look like using our choice of dimensionless units?

At first it could seem relatively straightforward to set the temperature by drawing velocities from the Maxwell distribution. Unfortunately however, the situation is more complex: when we start the simulation (after having drawn the velocities randomly) the system is not yet equilibrated. In fact, it equilibrates by exchanging energy between its kinetic and potential form. It is almost impossible to predict in advance what will happen exactly. It could take forever for the system to reach a stable equilibrium. We need to help the system get closer to the equilibrium before fully "releasing" it.

The solution is to first let the system equilibrate for a while (a few steps) and then force the kinetic energy to have a value corresponding to the desired temperature. We do this by rescaling the velocities

$$\mathbf{v}_i \to \lambda \mathbf{v}_i \tag{2}$$

with the same parameter λ for all particles.

The value of λ can be derived from the equipartition theorem. Specifically, we know that our "target" kinetic energy is given by

$$E_{\text{kin}}^{\text{target}} = (N-1)\,\frac{3}{2}k_BT. \tag{3}$$

The reason for using $N-1$ instead of N is that the total momentum in the system, $\sum_i mv_i$, is conserved, leaving thus only $N-1$ independent degrees of freedom. In order for our current kinetic energy $E_{\text{kin}}^{\text{current}} = (1/2)\sum_i mv_i^2$ to have the right value $E_{\text{kin}}^{\text{target}}$, λ must be chosen such that $E_{\text{kin}}^{\text{target}} = \lambda^2 E_{\text{kin}}^{\text{current}}$. From this follows

$$\lambda = \sqrt{\frac{(N-1)\,3k_BT}{\sum_i mv_i^2}}. \tag{4}$$

The equilibration and rescaling needs to be repeated a few times until the temperature has converged.

Questions for you: the formulas above are not in dimensionless units yet. For the simulation they need to be converted accordingly.

Fourth milestone:

Implement the initial conditions such that you can simulate the system at any desired density starting from the FCC lattice.

Implement the velocity relaxation method to reach the desired temperature.

You should be able to start a simulation that takes as input a density and a temperature only (for the 108 particles setup described above). The code should then generate the distribution of particle positions, draw velocities and then apply the procedure to reach equilibrium.