Computational Physics - Problem Sheet 5

Christian Gorjaew (354259)
Julius Meyer-Ohlendorf (355733)

1 Building a Molecular Dynamics Engine

Our molecular dynamics engine class (`MDEngine`) can be found in the file `cp_ex05_engine.py` which as well contains analysis functions outside of the defined class.

The class can in principle perform simulations that equilibrate in the NVT ensemble and use the NVE ensemble for production by employing the minimum image convention in $d = 1, 2, 3$ dimensions ("in principle" because we implemented it as general as possible but did not test the 1D and 3D compatibility thoroughly). The potential is kept variable such that a user specific potential function and a function calculating the acceleration on a particle can be passed to the class during initialization. The positions and velocities are stored as `numpy ndarrays` of shape (N, d) where N is the number of particles and d is the dimension of the system. This structure is kept throughout the whole class, i.e, accelerations, masses, etc. are kept in a compatible format.

In the following, we briefly explain our implementation of the required points that are specified on the exercise sheet:

- `_init_velocity` initializes the velocities according to the Maxwell-Boltzmann distribution. For each direction, directions are drawn from a normal probability distribution with mean $\mu = 0$ and standard deviation $\sigma = \sqrt{k_B T/m_i}$. The velocities are shifted such that the total momentum $\vec{P} = \sum_{i=0}^{N-1} m_i \vec{v}_i$ is zero.

- `velocity_verlet_step` executes one velocity Verlet step.

- `lennard_jones` (defined outside the class) calculates the potential energy potential energy resulting from two particles at position $\vec{r}_i$ and $\vec{r}_{j\neq i}$.

- `acc_lennard_jones` (defined outside the class) calculates the acceleration of particle i at $\vec{r}_i$ due to all particles $j \neq i$ at positions $\vec{r}_j$. In particular, it uses the force

$$\vec{F}_i = \sum_{j\neq i} 24\epsilon \frac{1}{r_{ij}^2}\left[2\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right](\vec{r}_i - \vec{r}_j) \tag{1}$$

  where $r_{ij} = |\vec{r}_i - \vec{r}_j|$. The function takes $\vec{r}_i$ as `numpy array` of shape $(d,)$ and the vectors of the other particles $\vec{r}_j$ as array of shape (M,d), in our case $M = N - 1$. The input for $\vec{r}_j$ is generated inside the class employing the minimum image convention.

- The minimum image convention is implemented in `find_nearest_neighbor` (assisted by the output of `_gen_cell_steps`). The method finds the images of all particles j closest to the particle i under consideration. For each particle j, the positions of its images inside the centered cell that contains particle i and the adjacent cells are calculated (9 positions in 2D and 27 in 3D). This is done by assigning a twofold tuple (k,l) to each of the relevant cells in the case of a 2D system where $k, l \in \{-1, 0, 1\}$. Thereby, (0,0) describes the central cell containing the actual particle i. For example, the cell right next to the central cell would have tuple (1,0). All combinations of tuples are calculated during the

initialization of a class instance by the method `_gen_cell_steps` for the 3D case (i.e., 27 combinations of tuples (k,l,m)). The generated tuples are ordered in such a way that taking, for example, the first nine ($3^{d=2}$) tuples using only the first $d = 2$ entries yields the correct arrangement for the 2D case. The position of particle j is calculated in all of the considered cells. The distances between these positions and the position of particle i are calculated and compared. The image position yielding the minimum distance is stored as the position vector considered for the determination of the force on i due to j. The routine does this successively for all particles j and outputs a $(N-1,\, d)$ dimensional array containing the determined positions.

- `run_nvt` and `run_nve` execute NTV equilibration and NVE production runs for a specified number of iterations. The former performs velocity rescaling every 10th step by multiplying all velocities with the common factor

$$\lambda = \sqrt{\frac{dk_B(N-1)T}{2E_{kin}}}. \tag{2}$$

  The instantaneous temperature, the kinetic, potential and total energy of the system are calculated and saved to disk in every iteration. Positions and velocities are saved every 10th step. For storage, files in the `hdf5` format are used[1]. It allows relatively fast storage to disk while also being easily resizable for followup runs.

  The latter method does basically the same except for an initial shift of the velocities to zero total momentum, lack of velocity rescaling and minor differences in the construction of the `hdf5` files.

  Both methods, in combination with the method `load_frame`, allow to continue from the last step of a previous run that was stored to disk.

2 Let the Atoms Dance

2.1 Units

The values for ϵ, σ and k_B are obtained as follows:

$$\epsilon = \frac{1.65}{1.66}\,\frac{10^{27+20}}{10^{24+21}}\,\frac{\mathrm{u\AA}^2}{\mathrm{ps}^2} \approx 99.4\,\frac{\mathrm{u\AA}^2}{\mathrm{ps}^2} \tag{3}$$

$$\sigma = 3.4\,\frac{10^{10}}{10^{10}}\,\mathrm{\AA} = 3.4\,\mathrm{\AA} \tag{4}$$

$$k_B = \frac{1.38}{1.66}\,\frac{10^{27+20}}{10^{24+23}}\,\frac{\mathrm{u\AA}^2}{\mathrm{ps}^2 K} \approx 0.83\,\frac{\mathrm{u\AA}^2}{\mathrm{ps}^2 K} \tag{5}$$

The length of the simulation cell l_{cell} is calculated as $l_{\mathrm{cell,i}} = \sqrt{\frac{100}{\rho_i}}$:

$$l_{\mathrm{cell},1} \approx 37.8\ \mathrm{\AA} \tag{6}$$

$$l_{\mathrm{cell},2} \approx 31.6\ \mathrm{\AA} \tag{7}$$

[1] `http://docs.h5py.org/en/stable/index.html`

2.2 Analysis

The analysis scripts can be found in the files `cp_ex05_dancing_rho1.py` and `cp_ex05_dancing_rho2.py`.
The analysis functions are implemented in the file `cp_ex05_engine.py`.

Initial note: Our code has either a bug that we were not able to resolve or there is a further physical condition that is unknown to us and needs to be implemented. It leads to the problem that the analysed system never properly equilibrate and diverge when the velocity rescaling is omitted in the NVE production run. In the following, we still try to describe our (faulty) results.

2.2.1 Density ρ_1

We started with the initial square lattice configuration, which can be seen in Figure 1.

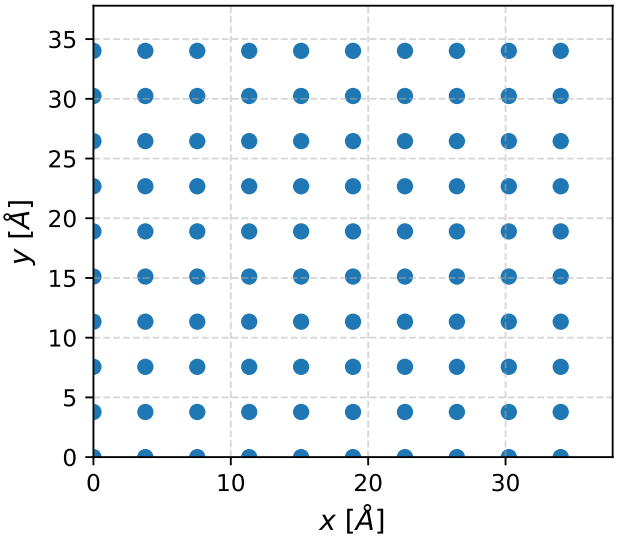
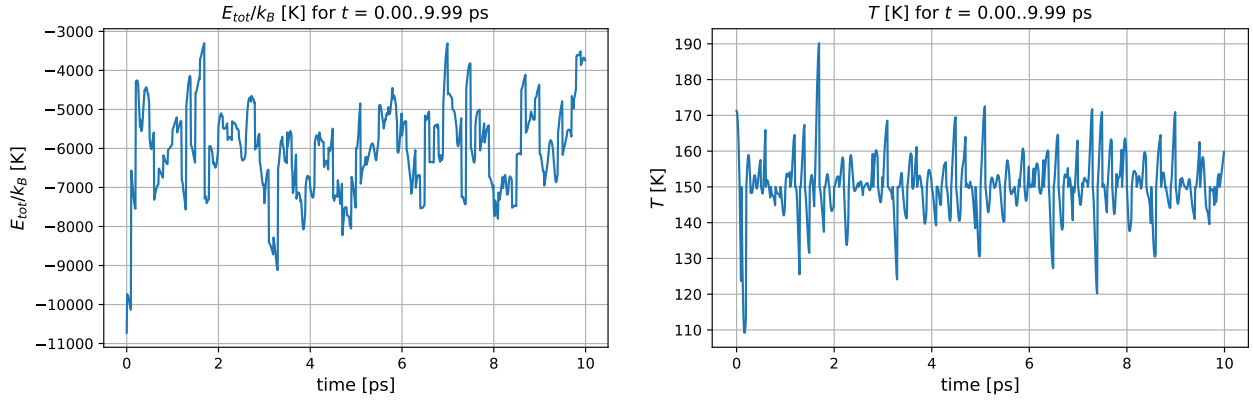


Figure 1: Square lattice configuration for initial atomic positions within the simulation cell with $l_{\text{cell},1} = 37.8\,\text{Å}$.

Using the initial atomic configuration we performed a NVT equilibration run for 1000 steps with a time step of $\tau = 0.01\,\text{ps}$ and the target temperature $T_{\text{target}} = 150\ K$. Both the total energy E_{tot} divided by the Boltzmann constant k_B and the temperature T as a function of the simulation time are shown in Figure 2.



(a) (b)

Figure 2: NVT equilibration run with 1000 steps using a time step of $\tau = 0.01\,\text{ps}$. (a) Total energy divided by the Boltzmann constant as function of the simulation time. (b) Temperature as a function of the simulation time

The scaled total energy $\frac{E_{\text{tot}}}{k_B}$ and T take some value at time $t = 0\,\text{ps}$ and the system then starts to evolve. For the first time the velocity is rescaled after 10 time steps pulling the system towards the target temperature $T_{\text{target}} = 150\,\text{K}$. Due to the periodic rescaling the scaled total energy and the temperature also show a periodic behaviour. Already after a simulation time of $t = 1\,\text{ps}$, the value of the rescaled energy starts to fluctuate around $-6000\,\text{K}$ and one would expect that it would converge towards a specific value eventually. The temperature starts to fluctuate around its target value after the first velocity rescaling operation. As opposed to the expectations, the fluctuations of the scaled total energy and the temperature do not really decrease towards the end of the equilibration run and the system does not converge properly. In order to check if the system needed more time to equilibrate, we simulated an additional 500 time steps, which did not lead to any improvement.
Figure 3 shows a snapshot of the atomic configuration at the last step of the NVT equilibration run. The configuration does not reveal any visible ordered or structured pattern.

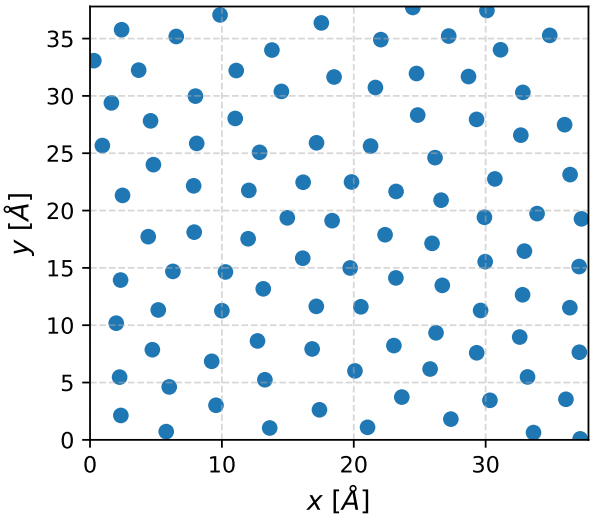
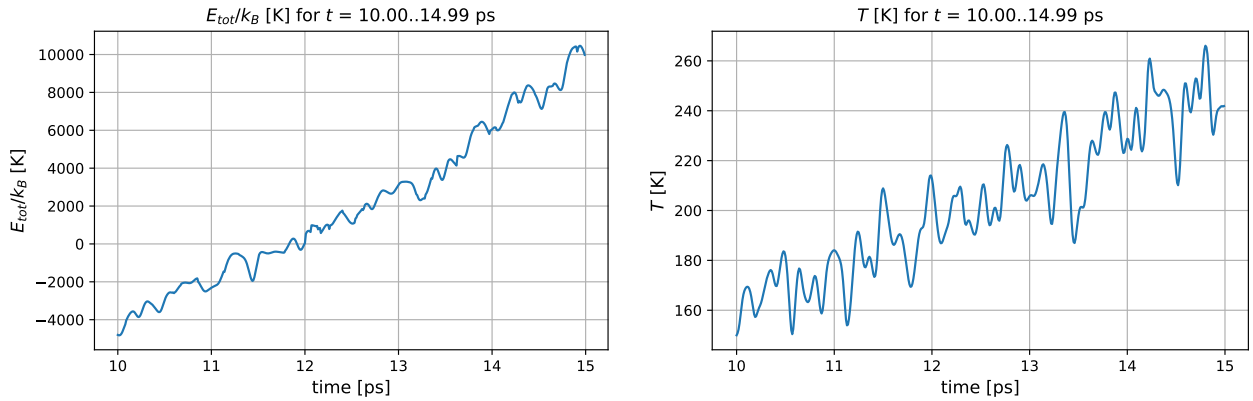


Figure 3: Atomic configuration after 1000 steps of the NVT equilibration run with $l_{\text{cell},1} = 37.8\,\text{Å}$. The coordinates were wrapped back into the initial cell.

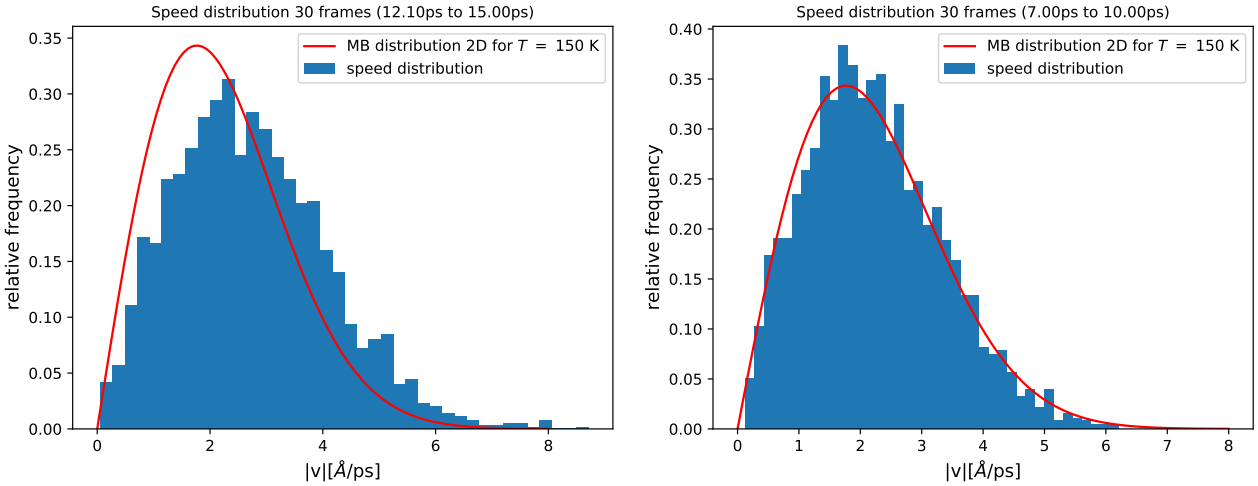
We then performed a NVE production run for 500 steps. The resulting behavior of the scaled total energy and the temperature is displayed in Figure 4.



(a) (b)

Figure 4: NVE production run with 500 steps using a time step of $\tau = 0.01\,\text{ps}$. (a) Total energy divided by the Boltzmann constant as function of the simulation time. (b) Temperature as a function of the simulation time

As already mentioned in the initial note the system starts to diverge. The scaled total energy grows from a negative to a positive value describing an unbounded system. Additionally, the temperature grows continuously. It therefore does not really make sense to extract physical properties from this faulty production run. However, we still sampled the speed distribution from the last 30 frames (corresponds to 300 time steps) of this NVE run. We also show the speed distribution from the last 30 frames of the NVT equilibration run (c.f Figure 5).



(a) (b)

Figure 5: Speed distribution of the last 30 frames (300 time steps) and theoretical 2D Maxwell-Boltzmann distribution in red. (a) For NVE production run. (b) For NVT equlibration run.

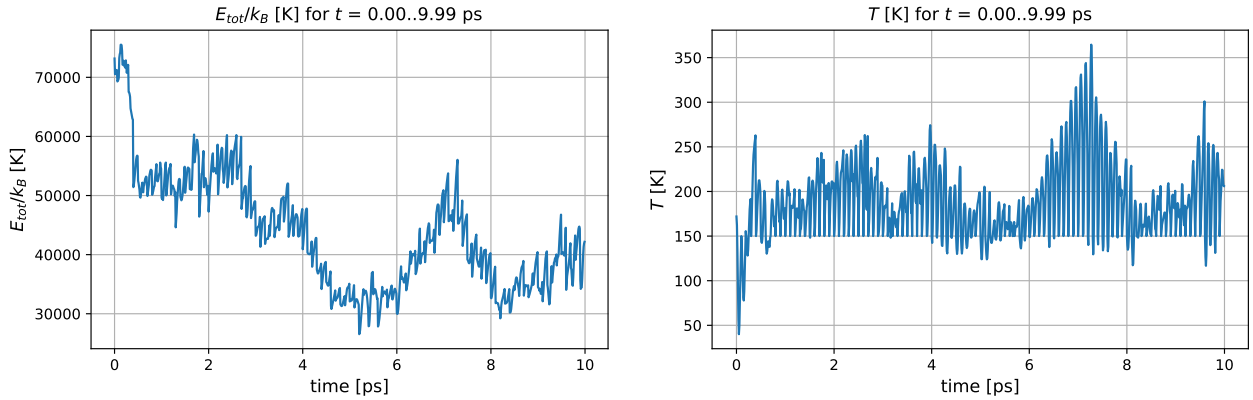
The distribution of the NVE run does not really follow a Maxwell distribution as it is sampled from a faulty run. In the case of the NVT run, the the speed distribution follows a Maxwell

distribution quite well. This is not very surprising as velocity rescaling is applied in this run. But due to this very reason the data also does not correspond to a real physical scenario and therefore the result is not very meaningful.

2.2.2 Density ρ_2

In general the procedure for the second investigated density is the same. The same parameters as described above were used. The initial configuration qualitatively looks the same as Figure 1 and differs only in the smaller spacing of the atoms and a consequently smaller cell. In Figure 6, temperature and total energy vs. time can be seen for the equilibration run. Here, the temperature fluctuates wildly and shows a bias towards higher values. The rescaling prevents divergence. Similarly, the total energy does not saturate at a specific value. Also, it takes on positive values indicating a unbounded system.

We also investigated the temperature and total energy for much more than 1000 steps (more than 3000) which did not show any effect in terms of proper equilibration.



(a) (b)

Figure 6: NVT equilibration run with 1000 steps using a time step of $\tau = 0.01\,\text{ps}$. (a) Total energy divided by the Boltzmann constant as function of the simulation time. (b) Temperature as a function of the simulation time.

Surprisingly, the atomic configuration after the last executed equilibration step shows a hexagonal pattern (Figure 7).

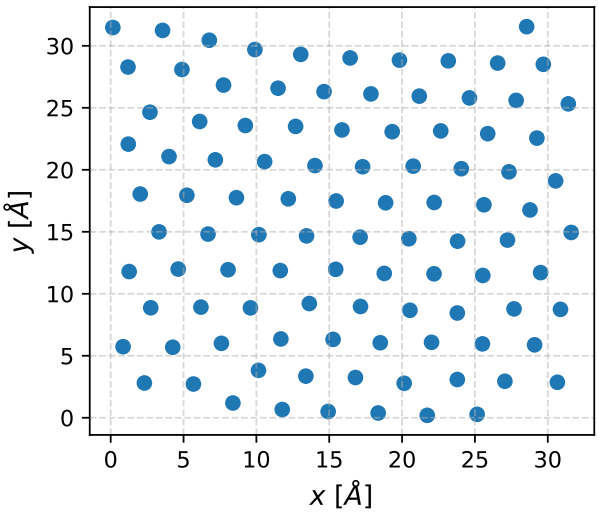
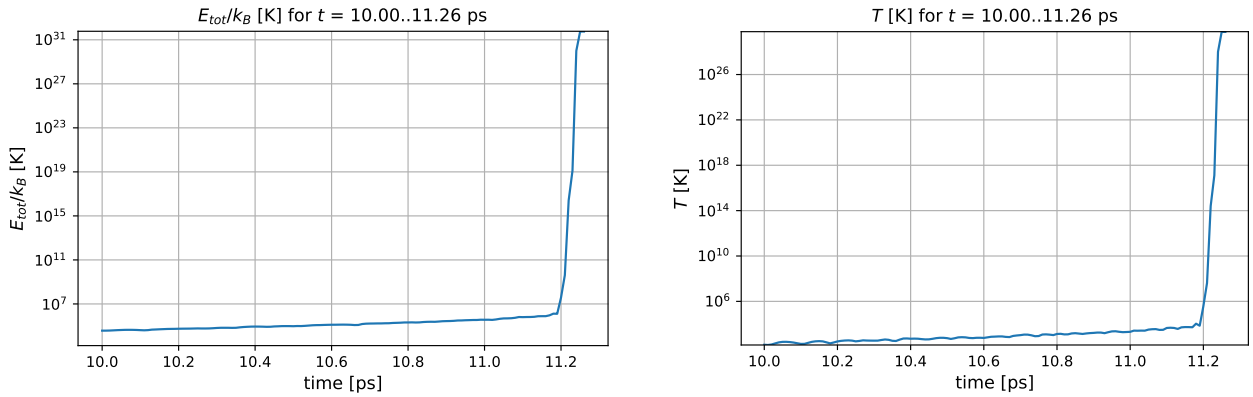


Figure 7: Atomic configuration after 1000 steps of the NVT equilibration run for ρ_2 with $l_{\text{cell},2} = 31.6\,\text{Å}$. The coordinates were wrapped back into the initial cell.

We still performed a production run after the previous 1000 equilibration steps. The system diverged and after approximately 120 steps the run stopped due to machine limitations. The divergent temperature and energy vs. time can be seen in Figure 8. The configuration at the iteration before failure is shown in the Appendix in Figure 11 where the positions were not wrapped back to the initial cell.



(a) (b)

Figure 8: NVE run with approximately 120 steps using a time step of $\tau = 0.01\,\text{ps}$. (a) Total energy divided by the Boltzmann constant as function of the simulation time. (b) Temperature as a function of the simulation time. A strong divergence is seen which is most likely caused by a faulty implementation of the simulation (see initial note).

Thus, the following analysis is performed using the data generated in the equilibration run, since here the quantities stay well behaved to some extend. We are aware that due to the rescaling, this data does not correspond to a physical situation.

In Figure 9 the speed distribution can be seen. The distribution of the simulated data deviates from the theoretical prediction although the velocities used here are those after the rescaling.

A reason for this could be that although the velocities result in the correct instantaneous temperature, they do not portray a Maxwell-Boltzmann distribution.

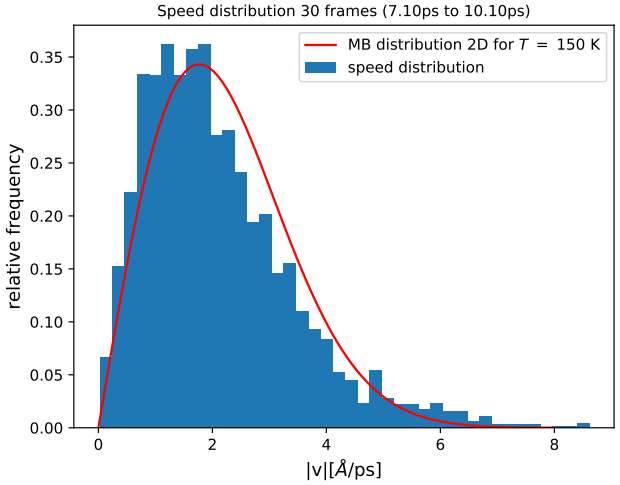


Figure 9: Speed distribution of the last 30 frames of the NVT run and theoretical 2D Maxwell-Boltzmann distribution in red.

2.2.3 Pair correlation function

We evaluated the radial pair correlation function

$$g(r_k) = \frac{2}{\rho(N-1)} \frac{\langle h(r_k) \rangle}{2\pi r_k \cdot \Delta r} \tag{8}$$

as defined in the lecture notes (lecture 9, slide 52) for the 2D case. The normalization is checked using

$$\text{Normalization} = \int_0^{r_{max}} dr 2\pi r g(r) \approx \sum_k 2\pi r_k g(r_k) \Delta r \tag{9}$$

where $r_{max} = l_{cell,i}/2$. In the present case

$$\text{Normalization} \stackrel{!}{=} l_{cell,i}^2 \tag{10}$$

We show the pair correlation function averaged over 71 frames of the equilibration run for both densities (c.f. Figure 10). The coordinates were wrapped back to the initial cell before calculating the pair correlation function. A discretization of $\Delta r = 0.1$ has been used. Although the results cannot be interpreted quantitatively, one recognizes certain features that could have been expected from the atomic configurations. The first peak for ρ_2 at $r \approx 3.5\,\text{Å}$ is much sharper and larger than for ρ_1 which is in line with the hexagonal arrangement seen in Figure 7. The configuration in Figure 3 is less structured resulting in a broader peak. The sharpness (broadness) can be observed for the follow up peaks as well. Conclusively, the distribution for ρ_1 resembles that of a liquid, the one for ρ_2 that of an solid (c.f., lecture notes Mazzarello, lecture 9, slide 54). We also note that the normalization does not take on the expected value ($l_{cell,1}^2 \approx 1429$, $l_{cell,2}^2 = 1000$). This could be either due to discretization errors or too low statistics. As a final remark, we note again that these results have to be interpreted with caution and the statistic is rather poor due to the small number of frames considered.

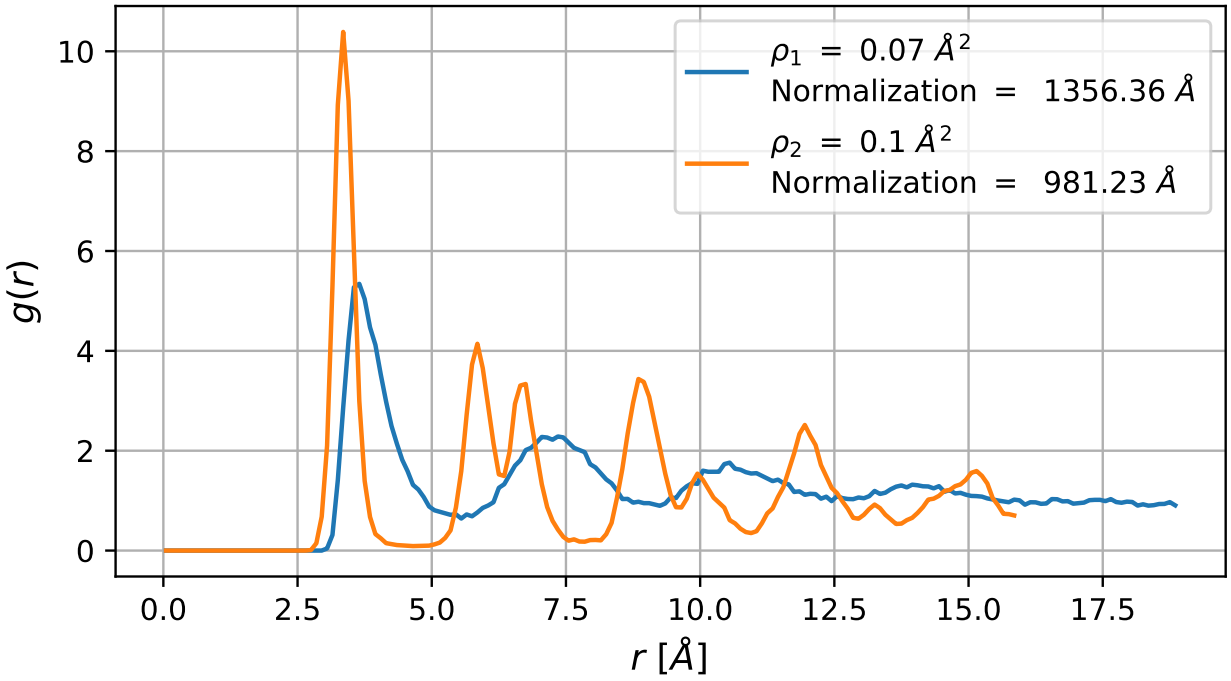


Figure 10: Radial pair correlation function $g(r)$ vs. distance r between pairs of atoms for the 71 equilibration frames for both investigated densities.

Appendix

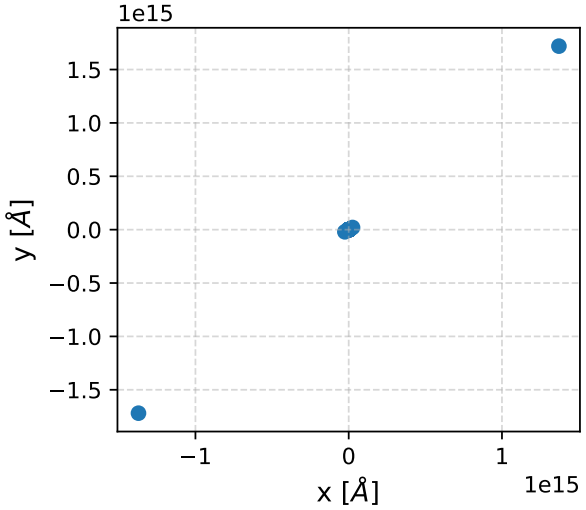


Figure 11: Atomic configuration after approximately 120 steps of the NVE production run for ρ_2. The position are far off the initial position.