Worksheet 2

Jens Jakschik, Fabio Oelschläger

December 4, 2020

Source code via Github
`https://github.com/bliepp/Simulation-Methods-in-Physics-Exercises`

1 Exercise 2 - Statistical Mechanics

1.1 Exercise 2.1 - Small tasks

The first task here was to consider a system A with two subsystem A_1 and A_2. A_1 has 10^{31} possible configurations, while A_2 has 10^{28} possible configurations. The number of configurations when combining multiple systems is multiplicative, meaning the total number of configurations for A is:

$$\Omega = \Omega_1 \cdot \Omega_2 = 10^{31} \cdot 10^{28} = 10^{59} \tag{1}$$

Then, in addition the entropies S, S_1, and S_2 should be calculated. The entropy for a given number of possible configurations is calculated with

$$S = k_\mathrm{B} \ln \Omega, \tag{2}$$

resulting in the following entropies for the system and subsystems:

$$S_1 = k_\mathrm{B} \ln 10^{31} = 9.85510 \cdot 10^{-22}\mathrm{J/K} \tag{3}$$
$$S_2 = k_\mathrm{B} \ln 10^{28} = 8.90138 \cdot 10^{-22}\mathrm{J/K} \tag{4}$$
$$S = k_\mathrm{B} \ln 10^{59} = 1.87565 \cdot 10^{-21}\mathrm{J/K}. \tag{5}$$

This shows, that while the number of possible configurations is multiplicative, the entropy is additive, as $S = S_1 + S_2$.

The second task was to calculate the increase in available configurations when 1 m^3 of neon at 1 atm and 298 K is allowed to expand by 1% at constant temperature. The number of possible states can be calculated using the partition function from statistical mechanics. For this, some assumptions have to be taken. First, we assume neon as an ideal gas. Then, as the neon is at room temperature, we assume the atmosphere around it as a heat storage. The partition function is then calculated as following.

$$Z(V,N,T) = \int \frac{\mathrm{d}\xi}{h^{3N}N!} \exp(-\beta H(p,q)) \tag{6}$$

Another assumption that is taken is, that there is no gravity acting on the cube. Thus, the Hamiltonian of the system reduces to:

$$H(p) = H_0(p) = \sum_{i=0}^{N} \frac{p_\mathrm{i}^2}{2m}. \tag{7}$$

With this, the number of possible configurations/states can be calculated.

$$\begin{aligned} Z(V,N,T) &= \int \frac{\mathrm{d}\xi}{h^{3N}N!} \exp(-\beta H(p)) && (8)\\ &= \frac{1}{h^{3N}N!} \int \mathrm{d}p\mathrm{d}q^3 \, \exp\left(-\beta \sum_{i=1}^{3N} \frac{p_\mathrm{i}^2}{2m}\right) && (9)\\ &= \frac{L^{3N}}{h^{3N}N!} \prod_{i=1}^{3N} \int \mathrm{d}p \, \exp\left(-\beta \frac{p_\mathrm{i}^2}{2m}\right) && (10)\\ &= \frac{V^N}{h^{3N}N!} \sqrt{\frac{2\pi m}{\beta}}^{3N} && (11) \end{aligned}$$

Here, it can be seen that the number of possible configurations is changed by a factor of 1.01^N if the volume is increased by 1%.

The third task was to calculate the increase in possible configurations when 100 kJ is added to a system consisting of 1 mol of particles at 400K. 1 mol means that the system consists of $6.022 \cdot 10^{23}$ particles. Modifying the Hamiltonian to $H(p) = H_0(p) + \Delta E$ changes equation (8) to equation (12).

$$\begin{aligned} Z(V,N,T) &= \int \frac{\mathrm{d}\xi}{h^{3N}N!} \exp -\beta(H_0(p) + \Delta E) && (12)\\ &= \exp -\beta\Delta E \int \frac{\mathrm{d}\xi}{h^{3N}N!} \exp\left(-\beta H_0(p)\right) && (13)\\ &= \exp\left(-\beta\Delta E\right) \frac{V^N}{h^{3N}N!} \sqrt{\frac{2\pi m}{\beta}}^{3N} && (14) \end{aligned}$$

With a temperature of $T = 400\,\mathrm{K}$ the inverse temperature becomes $\beta = \frac{1}{k_\mathrm{B} \cdot 400\,\mathrm{K}}$ defining the factor f to

$$f = \exp\left(-\frac{\Delta E}{k_\mathrm{B} T}\right) = \exp\left(-\frac{100\,\mathrm{kJ}}{k_\mathrm{B} \cdot 400\,\mathrm{K}}\right) = \exp\left(-\frac{250\,\mathrm{J\,K^{-1}}}{k_\mathrm{B}}\right) \approx 0 \tag{15}$$

1.2 Exercise 2.2 - Thermodynamic Variables in the Canonical Ensemble

Goal of this exercise was to show that all thermodynamic properties, namely internal energy U, pressure p, and entropy S can be derived from the Helmholtz free energy

$$F = -k_\mathrm{B} T \ln\left[Z(N, V, T)\right]. \tag{16}$$

Using the fundamental thermodynamic relation for a constant number of particles, the Helmholtz free energy can also be written as

$$\mathrm{d}F = -S\mathrm{d}T - P\mathrm{d}V. \tag{17}$$

Here S is the entropy and P the pressure, already showing the relation between the Helmholtz free energy and there thermodynamic properties. By taking the appropriate derivations, the following equations can be derived.

$$S = -\left(\frac{\partial F}{\partial T}\right)\Big|_{V,N} = -k_\mathrm{B} \ln Z - k_\mathrm{B} T \frac{1}{Z} \tag{18}$$

$$P = -\left(\frac{\partial F}{\partial V}\right)\Big|_{T,N} = -\frac{k_\mathrm{B} T}{Z} \tag{19}$$

The internal energy is related to the free energy with:

$$U = \partial_\beta(\beta \cdot F). \tag{20}$$

Therefore, the internal energy is:

$$U = \partial_\beta\left(-\frac{1}{k_\mathrm{B} T} \cdot k_\mathrm{B} T \ln Z\right) = -\partial_\beta \ln Z \tag{21}$$

1.3 Exercise 2.3 - Ideal Gas

In this exercise, the basis is the partition function which was already calculated in exercise 2.1.

$$Z(V, N, T) = \frac{V^N}{\lambda^{3N} N!} \tag{22}$$

The Helmholtz free energy is derived using:

$$F(V,N,T) = -\frac{1}{\beta}\ln Z(V,N,T) = -\frac{1}{\beta}\ln\frac{V^N}{\lambda^{3N}N!} \tag{23}$$

$$= -\frac{1}{\beta}\cdot[N\cdot\ln V - 3N\cdot\ln\lambda - \ln N!] \tag{24}$$

$$= -\frac{N}{\beta}[\ln V - 3\cdot\ln\lambda - \ln N + 1] \tag{25}$$

$$= \frac{N}{\beta}[-\ln V + 3\ln\lambda + \ln N - 1] \tag{26}$$

The pressure in the system is thus:

$$P = -\left(\frac{\partial F}{\partial V}\right)\Big|_{T,N} = -\frac{N}{\beta}V \tag{27}$$

And the internal energy:

$$U = -\partial_\beta \ln Z = -\frac{1}{Z}\cdot\partial_\beta Z = -\frac{\lambda^{3N}N!}{V^N}\cdot\frac{V^N}{N!h^{3N}}\frac{3N}{2}\sqrt{\frac{2\pi m}{\beta}}^{3N}\frac{-1}{\beta} \tag{28}$$

$$= \frac{3N}{2}\frac{1}{\beta} \tag{29}$$

2 Exercise 3 - Molecular Dynamics: Lennard-Jones Fluid

2.1 Exercise 3.2 - Reduced Units

The task for this exercise was to program functions for the Lennard-Jones potential and the resulting Lennard-Jones force. The Lennard-Jones potential was realised with the following code.

```
1 eps = 1
2 sig = 1
3 def lj_potential(r_ij: np.ndarray) -> float:
4     r = scipy.linalg.norm(r_ij)
5     return 4*eps*((sig/r)**12 - (sig/r)**6)
```

The resulting force out of the Lennard-Jones potential was calculated by taking the derivation of the potential. This resulted in the following function.

```
def lj_force(r_ij: np.ndarray) -> np.ndarray:
    r = scipy.linalg.norm(r_ij)
    return 24*eps*(2*(sig/r)**12 - (sig/r)**6) * r_ij/(r*r)
```

The Lennard-Jones potential and force should then be simulated for a one-dimensional problem. Here, the potential and force should be calculated for a thousand steps for x in [0.85,2.5]. This resulted in the expected shape of the Lennard-Jones potential and the resulting force.

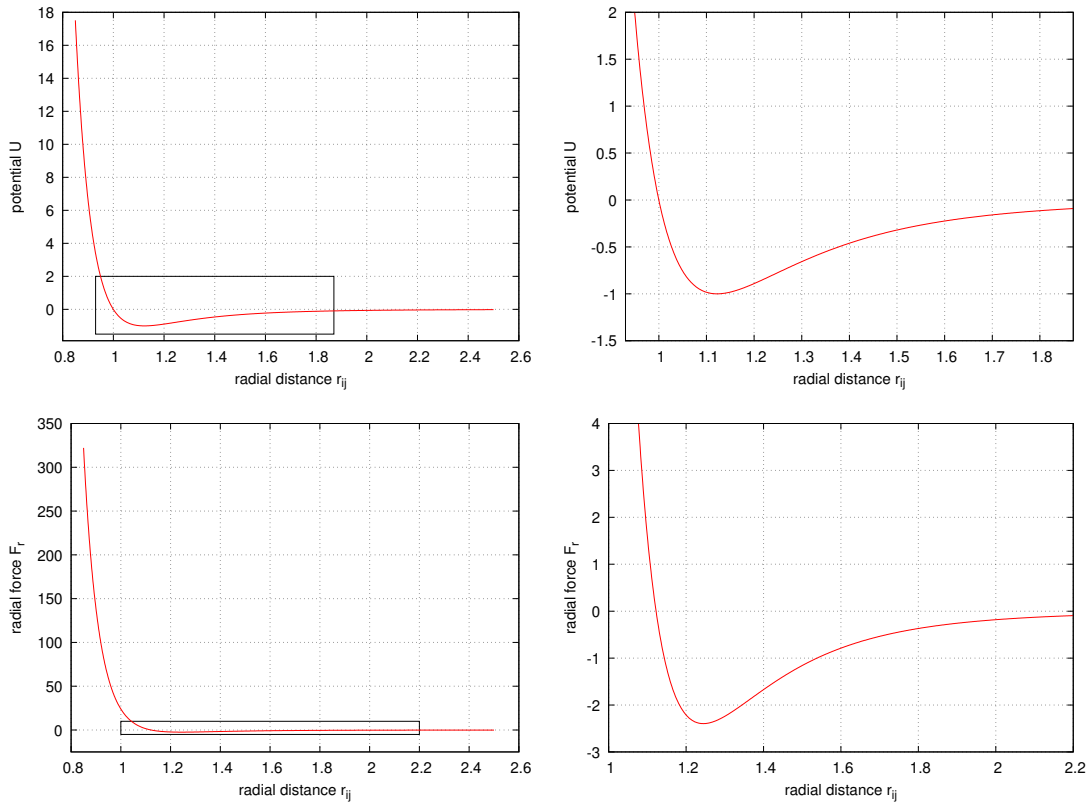

Figure 1: Overview (left) and detailed view (right) of the Lennard-Jones potential (top) and its matching force (bottom).

2.2 Exercise 3.3 - Lennard-Jones Billiard

For this exercise, a simulation for the interaction between particles interacting via the Lennard-Jones potential should be done. This was done in the form of Lennard-Jones billiard. The code for this was already provided by the exercise and the results of the

simulations were then visualized using VMD (Visual Molecular Dynamics).

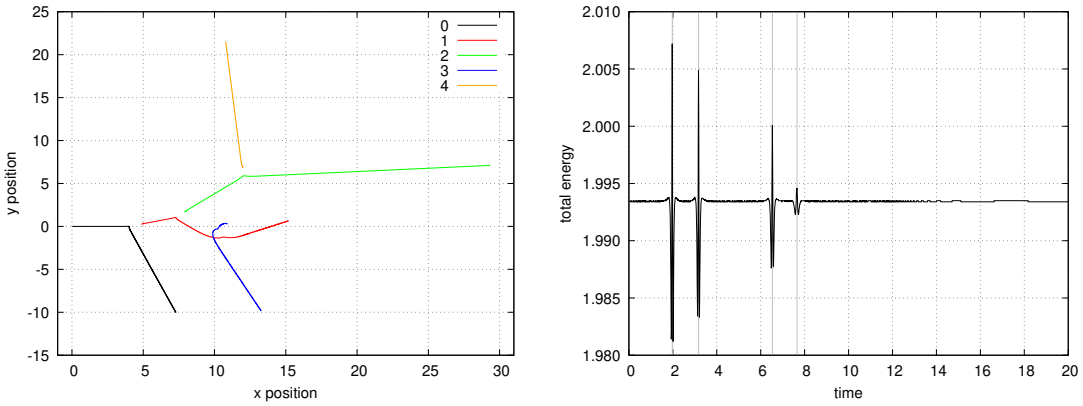

Figure 2: The trajectories of the billard balls (left) and the total energy over time (right).

Originally it was also part of the exercise to slightly alter the initial positions of the particles, so that particle 2 interacts with particle 4. This is already the case. Therefore, the exercise was considered completed after the VMD simulation was performed. The image of the start of the simulation can be seen in the following figure.

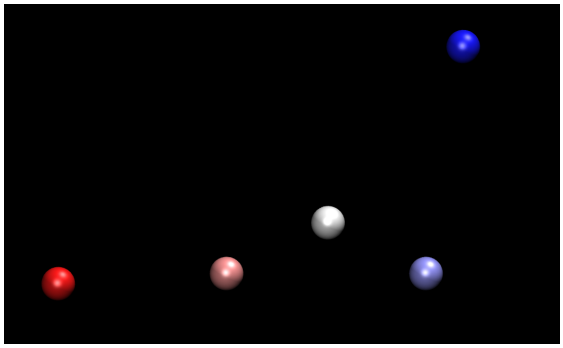

Figure 3: Rendering of the starting state of the Lennard-Jones billard

In figure 2 it can also be seen that some particles do not interact as they would for billiard with classic collision. Here, there are instead attractive forces between the particles, which can be seen in figure 2. Particle 3 and particle 1 begin to move towards to each other as soon as particle 1 enters the proximity of particle 3. This marks the difference between classical and molecular billiard, as there is also an attractive force between the particles for certain ranges.

2.3 Exercise 3.4 - Periodic Boundary Conditions

Here, periodic boundary conditions were introduced for the simulation done in section 2.2. For this to work, the potential needed to be limited to a certain range to make the computing time feasible. This was done by introducing a truncated potential.

```
def truncated_LJ(r_ij: np.ndarray):
    r = scipy.linalg.norm(r_ij)
    if r > r_cut:
        return 0
    return lj_potential(r_ij) - lj_potential(np.array([r_cut, 0]))
```

All that was done here, was to introduce a limit for the potential where it stops being computed as it is set to zero. The same rule applies for the Lennard-Jones force but without the shift. Then, the periodic boundary condition had to be implemented into the simulation done is section 2.2. For this, the functions calculating the forces acting on the molecules and the energy of the molecules had to be altered.

2.3.1 Minimum Image Convention

We are making use of the minimum image convention. It states that when looking at a periodic image size of $L > r_{\text{cutoff}}$ you only need to calculate the force a particle exerts on a single periodic image of another particle. The image used for the distance calculation is the minimum image (the closest one). Instead of transforming all particle positions transforming only the relative distance vectors like in equation (30) is sufficient.

$$\min(r_{ij}) = r_{ij} - L \cdot \text{round}(r_{ij}/L) \tag{30}$$

In equation (30) the round function gets whether the particle distance is greater than half of a simulation box or lower. An equivalent expression for this is to use a modulo function like in equation (31).

$$\begin{aligned} \min(r_{ij}) &= [(r_{ij} + L/2) \bmod L] - L/2 && (31)\\ &= [r_{ij} + L/2 - L \cdot \text{floor}((r_{ij} + L/2)/L)] - L/2 \\ &= r_{ij} - L \cdot \text{floor}((r_{ij} + L/2)/L) \\ &= r_{ij} - L \cdot \text{round}(r_{ij}/L) \qquad \square && (32) \end{aligned}$$

2.3.2 Test simulation

Running a test simulation with two particles with the start values $\underline{x}_1 = (3.9, 3)^\top$, $\underline{x}_2 = (6.1, 5)^\top$, $\underline{v}_1 = (-2, -2)^\top$ and $\underline{v}_2 = (2, 2)^\top$ results in a trajectory as shown in figure 4.

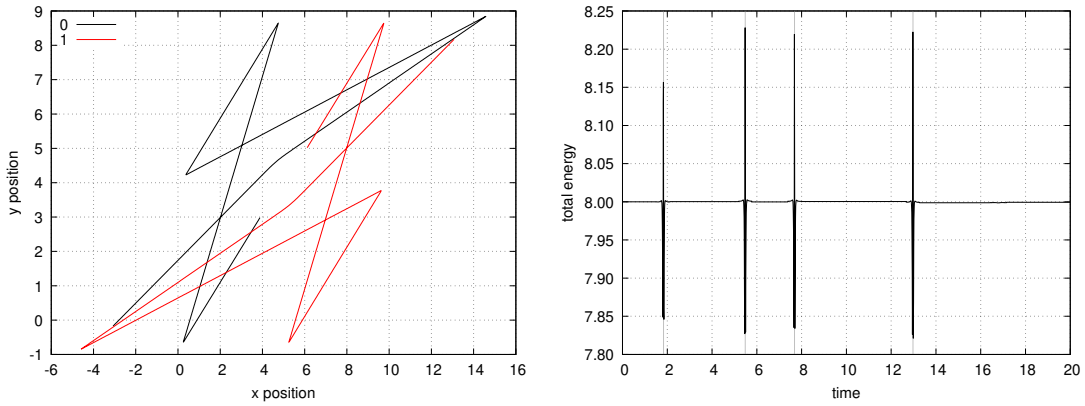

Figure 4: The trajectory of the test simulation and its corresponding energy fluctuation.

2.4 Exercise 3.5 - Lennard-Jones Fluid

In most situations it makes more sense to set the initial particle positions on a lattice and give the particles random initial velocities instead setting everything up by hand. This is rather simple with a given number of particles per dimension and agiven density.

```
x = np.empty((2, N_PART))
for p in range(N_PART):
    i = p % N_PER_SIDE
    j = p // N_PER_SIDE
    rel = np.array([i/N_PER_SIDE, j/N_PER_SIDE])
    x[:, p] = rel*BOX
```

By executing simulations of different particle sizes gives an estimate of the scaling behaviour of this algorithm. As shown in figure 5 calculation time over the number of particles gives a linear function in double logarithmic space. This means the scaling behaviour is of the type ax^b. Using a linear regression the coefficients a and b can be determined to be $a = 2.61 \cdot 10^{-3}$ and $b = 1.93 \approx 2$.

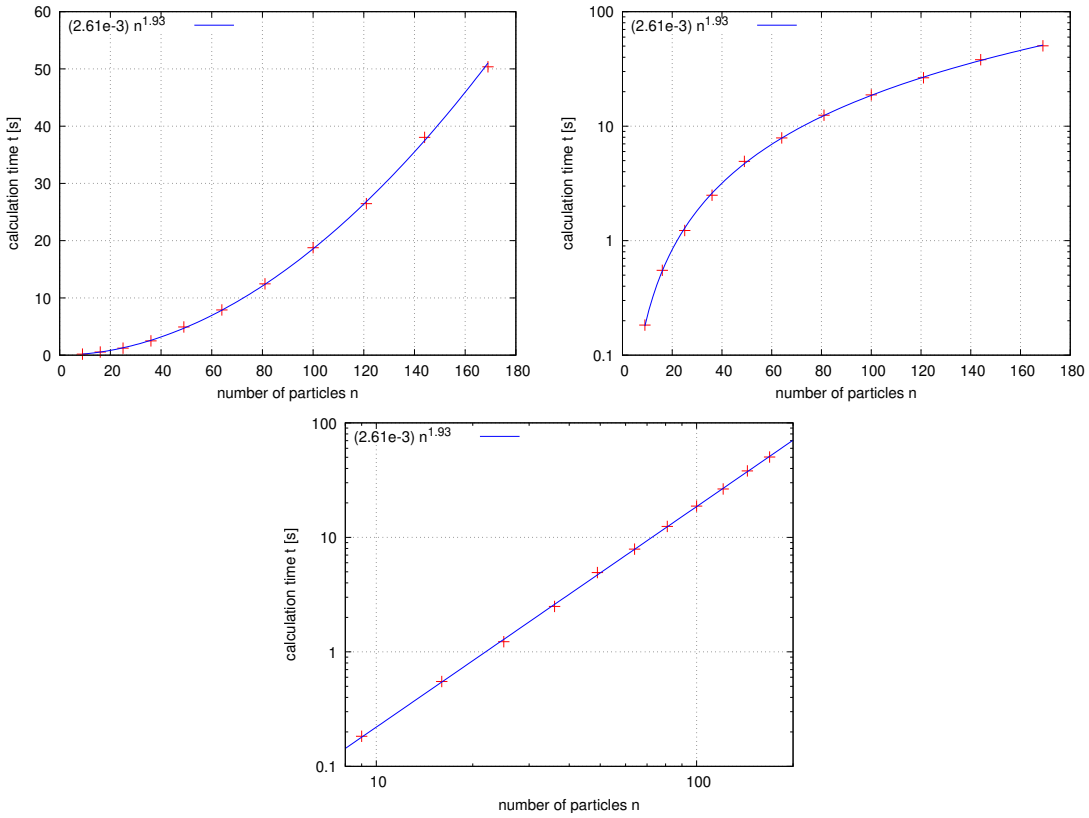

Figure 5: The scaling behaviour of the simulation algorithm.