Exercises NMSM 2025-2026

Lorenzo Rizzi - 2156773

December 12, 2025

Contents

Exercise 1: Lecture 7: Off-lattice Monte Carlo simulations

1.1 The Hard Sphere model

The Hard Sphere model is a paradigmatic model in Soft Matter (despite the nam- ing would not suggest so). The interaction energy between two hard spheres is zero if the separation distance is larger than the diameter of the particle σ and infinite otherwise. Set $\sigma = 1$ as the unit of length. Using the Hard Sphere model, one can simplify the Metropolis acceptance rule as follows:

- reject every displacement that brings any two particles closer than σ call it an overlap. You can assign a very large energy to the configuration if an overlap is present, thus overlap can be spotted more easily.
- accept every displacement that keeps every pair of particles at a distance larger than σ or that removes an overlap.

An acceptable state for a Hard Sphere system has zero energy and is homogeneously distributed in the simulation box. Consider a system of $N = 200$ Hard Spheres in three dimensions. Starting from random conformations, simulate the system at different number density $\rho = \frac{N}{V} : (i)\rho = 0.05\sigma^{-3}(ii)\rho = 0.4\sigma^{-3}$ and $(iii)\rho = 0.62\sigma^{-3}$. Use a simple cubic box and pay attention to set a suitable maximum displacement. Check that the energy reaches zero (equilibration) before collecting data. Compute the radial distribution function $g_2(r)$, discussed in class and describe what changes between the three different states.

Resolution

The model we are implementing is based on the hard-sphere nature of the constituent molecules. Formally, we can write the Hamiltonian of the system as:

$$\mathcal{H} = T + V = \sum_i \frac{1}{2} m v_i^2 + U$$

where T represents the standard kinetic energy and U is the total potential energy of the system, composed of pair-wise interactions $U = \sum_{i \neq j} U_{ij}$. Specifically, the hard-sphere condition is realized by imposing[1]:

$$U_{ij} = \begin{cases} \infty & \text{if } |\vec{r}_i - \vec{r}_j| < 1 \\ 0 & \text{if } |\vec{r}_i - \vec{r}_j| > 1 \end{cases} \tag{1.1}$$

Thus, the molecules interact only to avoid spatial overlaps but are otherwise free to move without experiencing any forces. In the NVT ensemble, the probability distribution function in phase space

[1] Note that we have rescaled the spatial lengths by choosing $\sigma = 1$, where σ is the molecular diameter. In this report, all lengths will therefore be dimensionless

$(\vec{r}_i, \vec{p}_i)$ is given by the Boltzmann formula:

$$p(\{\vec{r}, \vec{p}\}) = \frac{1}{Z} \exp(-\beta \mathcal{H}(\{\vec{r}, \vec{p}\}))$$

However, in this specific context, we are interested only in the marginalized configurational probability distribution, i.e., the probability of observing a specific configuration in coordinate space:

$$p(\{\vec{r}\}) = \int d\vec{p}_1 d\vec{p}_2 \dots d\vec{p}_n \, p(\{\vec{r}, \vec{p}\})$$

This expression factorizes conveniently due to the fact that the potential energy depends only on the configuration $\{\vec{r}_i\}$ and the kinetic energy only on the momenta $\{\vec{p}_i\}$:

$$p(\{\vec{r}\}) = p(\vec{r}_1, \vec{r}_2, \dots \vec{r}_n) = \frac{1}{Z_{conf}} \exp(-\beta U) = \frac{1}{Z_{conf}} \exp\left(-\beta \sum_{i \neq j} U_{ij}\right)$$

This is the probability distribution we want to sample from. In the hard-sphere model, this further simplifies to:

$$p(\{\vec{r}\}) = \begin{cases} const \neq 0 & \text{if no overlap occurs} \\ 0 & \text{if even one overlap occurs} \end{cases} \tag{1.2}$$

This implies that all *acceptable* states (i.e. configurations with no overlaps) are equiprobable. The Metropolis algorithm that will be implemented in the next section is designed to generate samples drawn precisely from this probability distribution. Before proceeding with the implementation of the algorithm, a few remarks:

- The derivation of the marginalized configurational probability was carried out in the canonical ensemble (NVT). However, due to the specific form of the potential, there is effectively no distinction in this context between the microcanonical (NVE) and the canonical ensembles. Indeed, the dependence on the thermal parameter β vanishes because the potential is either zero or infinite, rendering the Boltzmann factor independent of temperature for all allowed configurations.
- We construct Markov chains designed to sample from the PDF in Eq.1.2. It is important to note that this type of MC simulation is inherently static, meaning that we are sampling only the configurational component of the full distribution. Consequently, our MCMC algorithm yields configurations sampled with the correct probability, but no information regarding velocities can be inferred from these purely spatial samples. We can, therefore, compute only static or structural properties (such as the radial distribution function) that depend solely on the configurational subspace.

1.1.1 The algorithm

This section briefly summarizes the implementation of the algorithm in C++. A base structure `Particle` was defined to store a d-dimensional tuple of coordinates. The core class of the codebase is structured as follows:

```
1    template<int Dim>
2    class NDMolDyn {
3        std::vector<Particle> particles;
4        std::vector<std::vector<int>> verletLists;
5        std::vector<int> headOfChain;
6        std::vector<Particle> backupPositions;
7    };
```

where:

- The vector `particles` contains all N constituent particles, where N is defined by the input parameters.
- The matrix `verletLists` implements the **Verlet list strategy**. Specifically, the element `verletLists[i]` is a list containing the indices of particles that are sufficiently close to the i-th particle. Constructing these lists is an effective method to accelerate the force calculation. In fact, when checking if a particle i overlaps with any other particles, we only need to iterate through its Verlet list, whose size is typically much smaller than N. This eliminates the need to check all possible particle pairs.
- The process of generating the Verlet lists itself can be computationally intensive, potentially introducing a new $O(N^2)$ bottleneck. For this reason, the vector `headOfChain` implements the *cell list strategy*, which converts the costly Verlet list construction from an $O(N^2)$ operation to a highly efficient $O(N)$ algorithm. The vector `backupPositions` is used to evaluate whether it is necessary to rebuild the Verlet lists (rebuilding is avoided at each iteration thanks to the presence of a safety parameter, `skin`). The linked-list structure needed for the cell list approach is not contained in a separate vector L, but is included directly into the `Particle` structure for implementation simplicity:

```
    template<int Dim>
    struct Particle{
        std::array<double, Dim> position;
        int next = -1;    // -1 means nullptr
    };

```

The MC algorithm goes as follows[2]:

```
#define vecd std::array<double, Dim>

//Run a single step of MC (thus N proposals)
template<int Dim>
void NDMolDyn<Dim>::singleMCStep(){
    // First of all, select a random particle
    int particle_idx = rand() % m_particles.size();
    vecd old_pos = m_particles[particle_idx].position;
    vecd new_pos;
    for(int k = 0; k < Dim; ++k){
        double delta = (static_cast<double>(rand()) / RAND_MAX) * 2.0 *
    max_displacement - max_displacement;
        new_pos[k] = old_pos[k] + delta;
        if(new_pos[k] >= m_L){ //PBC
            new_pos[k] -= m_L;
        } else if(new_pos[k] < 0.0){
            new_pos[k] += m_L;
        }
    }
    // Now that we have the new_pos, verify whether there is an overlap
    for(auto it = m_verletLists[particle_idx].begin(); it != m_verletLists[
    particle_idx].end(); ++it){
        vecd pos_j = m_particles[*it].position;
        double d2 = dist_sq_pbc(new_pos, pos_j);
        if(d2 < 1.0){ //Overlap detected! Return and don't do anything
```

[2]This is just a minimal example, the full code is longer because there are additional feature (acceptance rate counter, ...)

```
            return;
        }
    }
    // If we reach this point, we can accept the move
    m_particles[particle_idx].position = new_pos;
    // Finally, check whether we need to rebuild the verlet lists
    double trigger = distance_square_pbc(m_backupParticles[particle_idx], new_pos);
    if (trigger > m_skin/2 * m_skin/2) {
        // Rebuild the verlet lists
        buildVerletLists();
    }
}
```

The simulation can be run in an arbitrary number of dimension, but we'll use $d = 2$ or $d = 3$ for the sake of simplicity. The parameters that we can modify can be summarized: When the function

ρ	Density
N	Number of particles
r_{cut}	Cut-off radius, Verlet's technique
r_{skin}	Skin, Verlet's technique
M	Number of cell per dimension
$MCSteps$	Number of MC steps to perform (1 MCStep = N proposals)

`buildVerletLists()` is called, the program will start filling the object `std::vector<std::vector<int>> verletLists`. In particular, the list `verletLists[i]` will consist of all particles whose position r_j is such that:

$$|\vec{r}_j - \vec{r}_i| < r_{cut} + r_{skin}$$

The parameter r_{cut} should be chosen as to represent a characteristic length over which the interaction dies out. In this specific case, since the pair-wise hard-sphere interaction is non zero only when $|\delta r| < 1$ (diameter of the molecules), we can safely set $r_{cut} = 1$. The skin parameter is a performance parameter, meaning that it sets how frequently one has to update the Verlet lists (the less, the better). In this code, I've used to rule according to which when the total displacement of a molecules (with respect to the position occupied the last time `buildVerletList()` was called) is larger than half of the skin parameter, then the Verlet lists must be rebuilt from scratch. A good choice for $r_{skin} = 0.5$ (but this depends on the specific run). In order for the algorithm to work, a few sanity check are in order: When the function `buildVerletLists()` is called, the program will start populating the structure `std::vector<std::vector<int>> verletLists`. In particular, the list `verletLists[i]` will consist of all particles whose position $\vec{r}_j$ is such that:

$$|\vec{r}_j - \vec{r}_i| < r_{cut} + r_{skin}$$

The parameter r_{cut} should be chosen to represent a characteristic length over which the interaction is non-negligible. In this specific case, since the pair-wise hard-sphere interaction is non-zero only when $|\delta r| < 1$ (the molecular diameter), we can safely set $r_{cut} = 1$. The skin parameter is a performance optimization, meaning that it sets how frequently one has to update the Verlet lists (the less frequent, the better). In this code, the following rule is applied: when the total displacement of a molecule (with respect to the position occupied the last time `buildVerletList()` was called) is larger than half of the skin parameter, then the Verlet lists must be rebuilt from scratch. A good choice for $r_{skin} = 0.5$ (but this depends on the specific run). In order for the algorithm to work, a few sanity checks are in order:

- The Verlet lists are built upon the cell lists, meaning that the neighbours of a particle are selected by only checking adjacent cells. If the linear size of the cells is too small, then we might be losing

ρ	Max Displacement	Skin	M	Rebuilding	Acceptance rate
0.05	0.4	1	5	14 %	0.92 ± 0.02
0.4	0.35	0.8	4	10 %	0.48 ± 0.05
0.62	0.2	0.5	4	4%	0.38 ± 0.06

Table 1.1: *A list of the parameters governing the Verlet and cell list techniques that were implemented to enhance code performance (`max_displacement, skin, M`). The total number of MC steps and r_{cut} were held constant at* $10,000$ *and* 1*, respectively. In addition, two diagnostic parameters were monitored to ensure the Markov chain explored the space correctly and that the acceleration technique was effective: the acceptance rate and the rebuilding factor. The acceptance rates for the higher densities ($\rho = 0.4$ and $\rho = 0.62$) can be considered satisfactory, whereas the rate observed for $\rho = 0.05$ is rather high. While in ordinary MC simulations this might suggest the algorithm is exploring the space too slowly, this is not the case here: the low number density results in extremely rare interactions.*

essential interactions. Specifically, we require:

$$M < \left\lfloor \frac{L}{r_{cut} + r_{skin}} \right\rfloor \tag{1.3}$$

- At each MC step, N particles are randomly selected and a spatial move is proposed according to a parameter `max_displacement`. In order for this move not to break the Verlet list inner working, we request:

$$\text{max\_displacement} < \frac{1}{2} d_{skin} \tag{1.4}$$

The program will automatically perform those sanity check at the beginning of the simulation and signal to the user whether a violation has been detected.

1.1.2 Qualitative analyis

We set $N = 200$ particles and investigate the behavior of the system in $3D$ across three different number densities: $\rho = 0.62, \rho = 0.4$, and $\rho = 0.05$. The simulation executes the MCMC algorithm and records the coordinates of all particles at specified intervals, dumping them into a `traj.xyz` file. This output is particularly useful when analyzed with `OVITO`, a free software package capable of reading `.xyz` trajectory files and rendering the simulation in $3D$.[3] The resulting configurations for the three defined densities are visualized in Fig.1.1: the results qualitatively confirm that higher concentration leads to denser molecular packing. Table.1.1 presents the parameters defining the MC chain along with key diagnostic quantities. These diagnostics include the acceptance rate and the rebuild probability (which quantifies the frequency of the computationally expensive Verlet list updates). With the obtained MC chains for the spatial configurations, we can analyze the structural properties of the system, starting with the *radial distribution function* $g(r)$. For this purpose, `OVITO` was used to calculate the instantaneous, frame-by-frame radial distributions. These instantaneous distributions were subsequently averaged (the Python notebook) to yield a smoother, statistically reliable curve. The resulting time-averaged $g(r)$ curves are shown in the right panel of Fig.1.2. For all simulated densities, $g(r)$ vanishes when $r < 1$. This confirms the expected behavior and serves as a successful sanity check for the algorithm. As spatial overlap is strictly prohibited, no pair of particles can stay closer than the molecular diameter ($r_\star = 1$), and $g(r)$ is precisely zero within this exclusion zone.

[3]The native $2D$ rendering option was implemented using the SFML library; however, due to the complexity of real-time $3D$ rendering, we relied on an external third-party software.

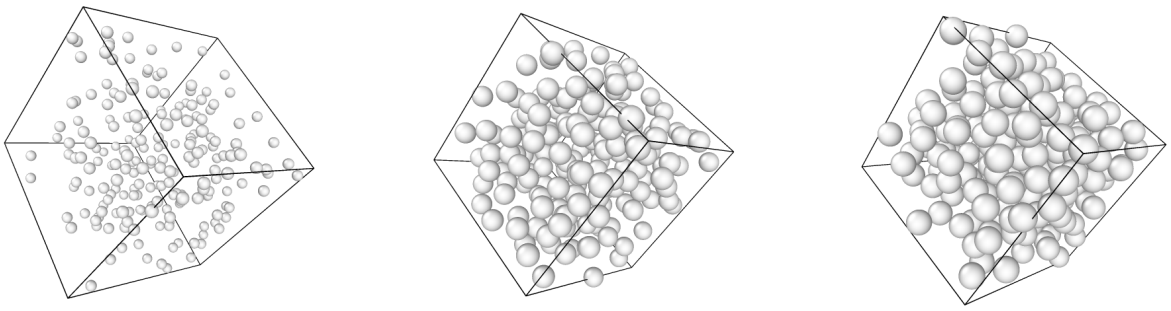

Figure 1.1: *3D rendering (under periodic boundary condition) performed by OVITO starting from three MC chains. From the left to the right,* $\rho = 0.05, \rho = 0.4, \rho = 0.62$ *and* N *is kept constant at* $N = 200$*. The spatial lengths are all rescaled in such a way that* $\sigma = 1$*, where* σ *is the molecules diameter.*

The behaviour for $r > 1$ suggests the formation of at least two phases (a gaseous phase and a condensed one). When $\rho = 0.05$, $g(r)$ remains close to the asymptotic line $g(r) = 1$, which is characteristic of an ideal gas. In other words, at sufficiently low densities, the local density is relatively homogeneous and comparable to the global density $\rho = N/V$. A slight enhancement is observed at $r \approx 1$, which decays back to 1 as $r \to \infty$; this indicates that the local density in the immediate vicinity of a molecule is slightly higher than the typical (uncorrelated) density of an ideal gas. This effect is a direct consequence of the excluded volume interaction. Consider a particle fixed at $r = 0$: in an ideal gas system, a second particle can arbitrarily occupy any point $\vec{r}$ in space independent of the existence of the first particle at $\vec{r} = 0$. Consequently, there would be a finite probability $p(V_\sigma)$ of finding the second particle within the region V_σ, defined as a sphere of radius σ centered at the origin. Getting back to a system with excluded volume interactions, this probability $p(V_\sigma)$ must necessarily be zero, since the potential energy of an overlapping configuration is infinite. Consequently, this probability mass $p(V_\sigma)$ is effectively "redistributed" into the accessible region of space immediately surrounding the sphere V_σ.

When $\rho = 0.4$ or $\rho = 0.62$, the radial distribution curves exhibit more complex features. The peak around $r = 1$ becomes significantly more pronounced, increasing in height as ρ increases. Furthermore, the curve displays the characteristic behavior of oscillations exponentially damped that asymptotically decay to the value 1 as $r \to \infty$. This pattern provides us with crucial information regarding the geometric structure of the system. The alternating peaks and valleys correspond to successive *coordination shells*. The first peak represents the nearest neighbors forming a "cage" around the reference particle, while the subsequent peaks represent layers of secondary neighbors. The first valley implies a region of particle depletion: this is expected, as the presence of a tightly packed first shell prohibits other particles from occupying the same volume due to steric exclusion. The damping of these oscillations indicates that the positional correlation is eventually lost at long distances.

Clearly enough, the larger the density ρ, the more pronounced those oscillations are (since the geometric packing become more and more regular).

1.2 The Patchy Hard Sphere model

1.2.1 Implementation

To implement the patchy interaction described by the Kern-Frenkel potential, the `Particle` structure was extended to include a `Patch` object, which essentially represents a three-dimensional unit vector. Although each particle is decorated with two patches, their diametrically opposite arrangement allows for a significant simplification: for practical purposes, tracking a single unit vector is sufficient to define

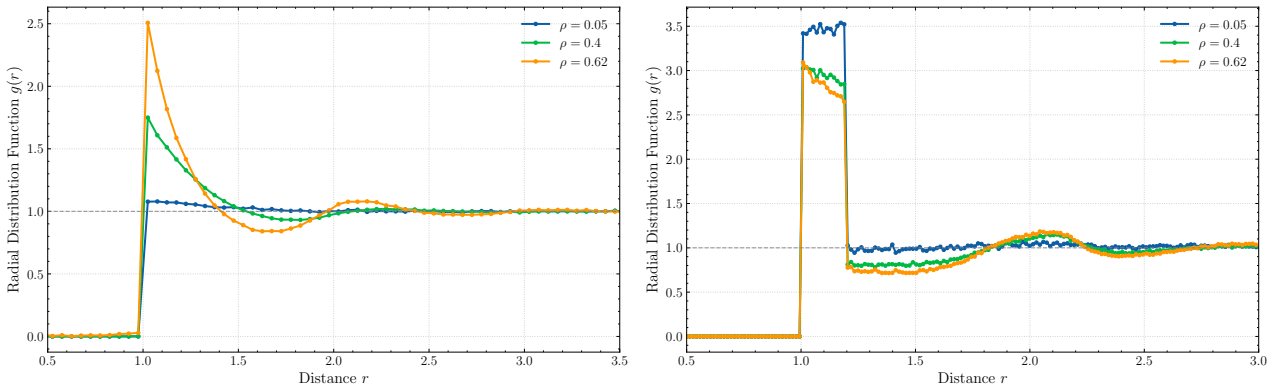

Figure 1.2: *(Left panel) Radial distribution function for the hard-sphere model as a function of (rescaled) distance r at different values of ρ (but $N = 200$). In all cases, $g(r) = 0$ when $r < 1$, thanks to the excluded volume, and the correlation dies out when r gets larger (Right panel) Radial distribution function for the Van der Waals model (isotropic patchy hard sphere where $\theta_{max} = \frac{\pi}{2}$). Now a clear correlation appears for all three densities in the region $1 < r < 1.2$.*

the orientation. To evaluate the pairwise interaction energy, the function `getEnergy()` was developed:

```
double NDMolDyn<3>::getEnergy(vecd particle_pos, Patch particle_patch, int
    particle_idx) const {
    double energy = 0.0;
    for(auto it = m_verletLists[particle_idx].begin(); it != m_verletLists[
    particle_idx].end(); ++it){
        vecd pos_j = m_particles[*it].position;
        double d2 = dist_sq_pbc(particle_pos, pos_j);
        // If within interaction range (we've already checked for overlaps)
        if (d2 < (1.0 + 0.2)*(1.0 + 0.2)) {
            double cos_angle_1 = computeAngle(particle_patch, particle_pos, pos_j, d2
    );
            double cos_angle_2 = computeAngle(m_particles[*it].getPatch(), pos_j,
    particle_pos, d2);
            if (std::abs(cos_angle_1) > m_cosThetaMax && std::abs(cos_angle_2) >
    m_cosThetaMax) {
                energy += -1.0; //Dimensionless energy, assume epsilon_SW = 1
            }
        }
    }
    return energy;
}
```

1.2.2 Extreme cases

Before considering a reasonable set of values for θ_{max}, let's explore the extreme cases, where $\theta_{max} = 0$ and $\theta_{max} = \frac{\pi}{2}$. In the first case, the molecules needs to be completely and perfectly aligned to interact; this is impossible ('cause of the machine error), so when $\theta_{max} = 0$ we fallback to the previous case where no interactions (apart from the repulsive volume) occurs. However, when $\theta_{max} = \frac{\pi}{2}$ then particles will always interact as long as their distance is smaller than $1 + \delta = 1 + 0.2$ (but larger than 1). This is the case where no patchy-like interactions occur and only a potential well around the molecules is considered (a Van der Walls gas!)

We fix $\theta_{max} = \frac{\pi}{2}$ and evaluate the radial distribution function at $k_B T = 0.8$ for three different

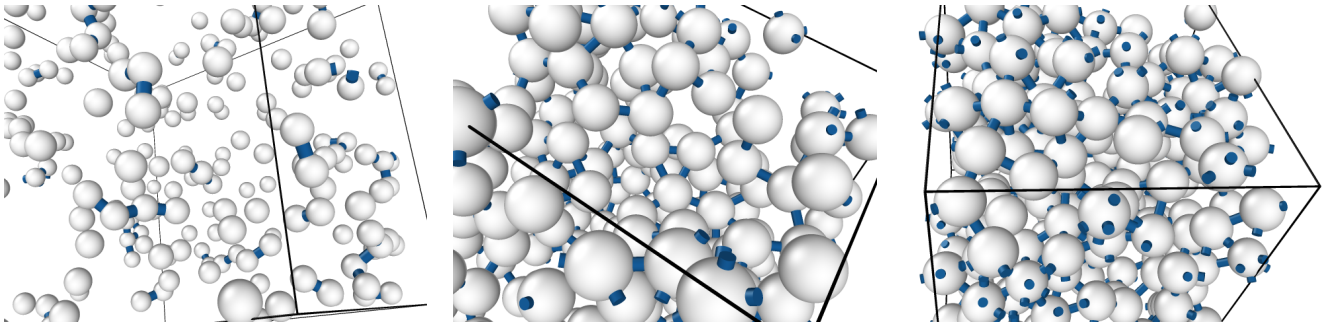

Figure 1.3: *3D rendering (under periodic boundary condition) performed by OVITO starting from three MC chains (isotropic potential well + excluded volume). From the left to the right,* $\rho = 0.05, \rho = 0.4, \rho = 0.62$ *and* N *is kept constant at* $N = 200$*. The blue bonds connect pair of molecule whose distance is less than* 1.2 *(thus interacting particles).*

densities[4]. The results are presented in the right panel of Fig.1.2. Once again, for $r < 1$, the radial distribution vanishes due to the excluded volume interaction. However, particles are now subject to a short-range attractive potential well (energy gain of -1) which favors configurations where particles are directly surrounded by other particles. Indeed, Fig.1.2 confirms the presence of a region $1 < r < 1.2$ where the local density significantly exceeds the bulk density, suggesting the formation of small molecular clusters. Beyond the interaction range ($r > 1.2$), we recover the typical oscillatory behavior for high densities or the flat asymptotic profile for low densities.

In Fig.1.3, we present three simulation snapshots rendered with OVITO, corresponding to the configurations analyzed in the right panel of Fig.1.2, where an isotropic potential well is included. Particles separated by a distance less than 1.2 are visually connected by a bond, representing a direct (negative) contribution to the potential energy. The case of $\rho = 0.05$ is particularly interesting: although the system remains in a gaseous phase, the presence of a short-range attractive potential renders small molecular clusters thermodynamically stable (and energetically favorable). Consequently, we observe the formation of pairs (dimers), and with rapidly decreasing probability, triplets (trimers) or higher-order structures.

One might argue that the formation of these structures is merely coincidental, potentially occurring even in the absence of a potential well (these apparent bonds could simply result from random spatial fluctuations where particles momentarily approach each other at a distance smaller than 1.2)[5]. To verify that the emergence of these structures is effectively driven by the short-range interaction, we quantify the number of bonds in both cases (i.e., $\epsilon = 0$ and $\epsilon = -1$), defining a bond between two particles whenever their mutual distance is less than 1.2.

I've run a simulation with $10k$ MC steps under those conditions and we averaged the number of bonds b for the last 1000 steps. When $\epsilon = 0$, we measure $b = 16 \pm 4$, whereas when $\epsilon = -1$ we obtain $b = 52 \pm 5$. This is enough evidence to prove that the short range potential is responsible for the formation of those very small structures (only a few particles wide).

Energy and condensation An alternative approach to investigating bond formation involves the analysis of the system's potential energy. Indeed, a direct relationship exists such that $U = Nb$, where b denotes the number of bonds. The evolution of the potential energy as a function of MC steps is

[4]Energy is rendered dimensionless by normalizing to the potential well depth, that is by setting $\epsilon_{SW} = -1$.

[5]Bonds in the case where no potential well exists doesn't make much sense, since they do not contribute to the energy of the system. The point I'm trying to prove here is that, when the potential is switched on, the fact that a certain amount of molecules tend to cluster together in small structure cannot be simply explained by pure luck, i.e. particles that eventually come closer.

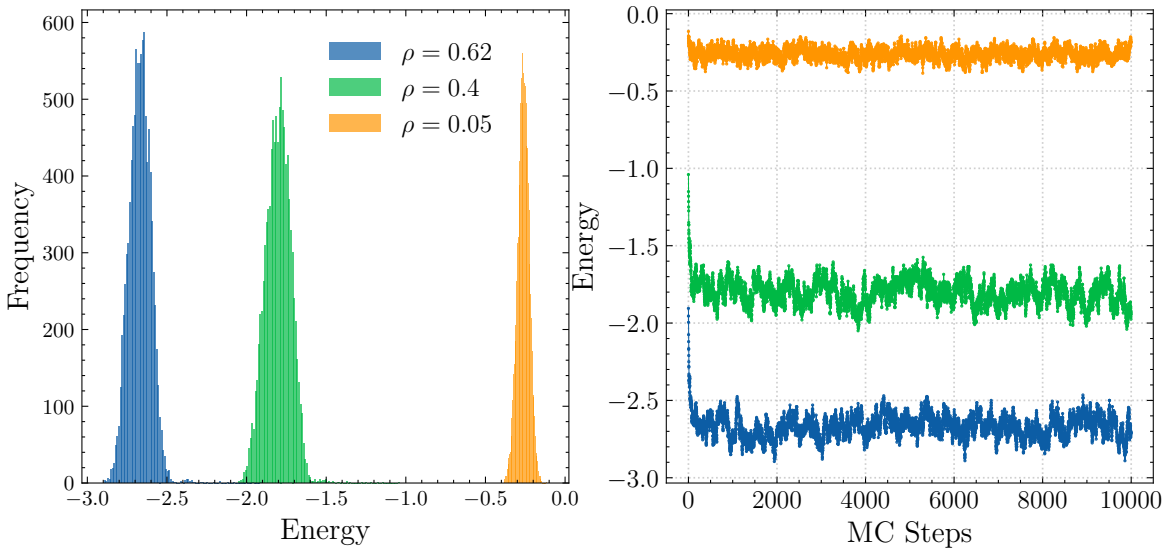

Figure 1.4: *Trace plots (on the left) and relative histograms (on the right) for three MC chains sampling from the isotropic potential well system at fixed temperature* $k_BT = 0.8$ *for three different values of density* ρ*. Number of particles* $N = 200$ *and* 10000 *MC steps. Both the trace plots and the histograms display the usual behaviour expected from a Markov chain-based algorithm (an initial transient + a thermalized phase).*

illustrated in Fig.1.4, alongside a histogram of the same observable at a temperature of $k_BT = 0.8$. As expected, the MCMC algorithm begins sampling configurations from the correct equilibrium distribution only after a transient period, which depends on the chosen initial state. Following this initial **burn-in** phase, the trace plots clearly demonstrate that the Monte Carlo chains have converged to the limiting distribution and are sampling effectively. The distribution of the samples is approximately Gaussian, a result consistent with standard expectations in statistical mechanics that serves as a validity check. To obtain a Monte Carlo estimate of the thermal average $\langle U \rangle$, we discard the initial samples (burn-in) and compute the mean over the remaining trajectory. Next, we can compute the average energy $\langle U \rangle$ for different values of ρ and T (Fig.1.5).

At sufficiently high temperatures (specifically, where "high" means much larger than the depth of the potential well), the effect of the short-range potential becomes essentially negligible, as particles have sufficient thermal energy to freely enter and exit the potential well. Consequently, these high-temperature systems effectively behave as systems governed purely by excluded volume interactions. The energy curve should thus relax to an asymptotic value U_{free} that can be interpreted as the energy that the system would have if the probability distribution function on the configuration space was the one associated to the same system with no potential well. When the density is low enough ($\rho = 0.05$), then particles are usually far away from each other and we expect:

$$U_{free} = \lim_{T \to \infty} U_{well}(T) = 0$$

However, when the density grows, the asymptotic energy U_{free} becomes increasingly negative. This is simply due to the fact that, when the system is high-density, there will be a moderate probability that a pair of particles gets closer than 1.2, thus creating a bond and adding a quanta $\epsilon = -1$ to the energetic budget. However, this does not imply that particles are actively driven together by the short-range attraction, as the energetic influence of this potential is essentially negligible when the temperature is sufficiently high. Instead, the observed asymptotic energy reflects the fact that particles are forced into

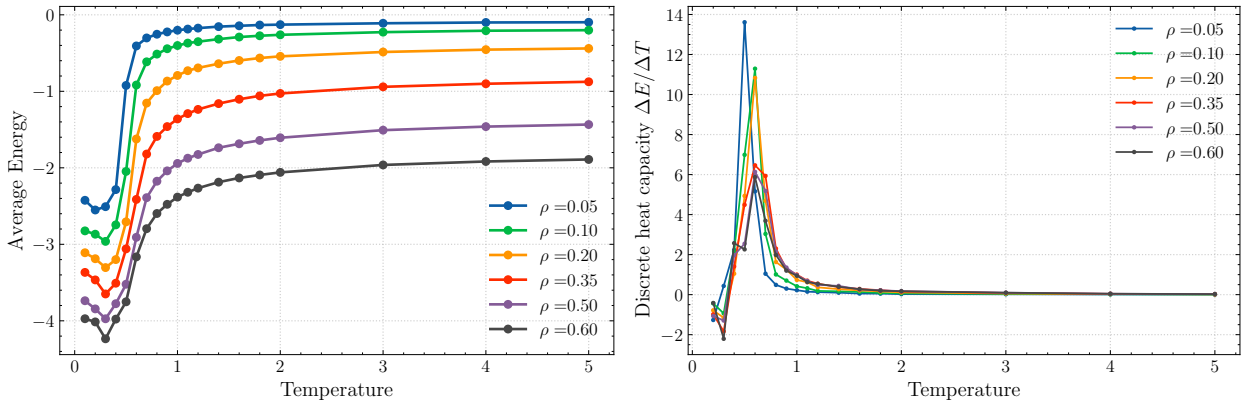

Figure 1.5: *On the left, curves representing the average potential energy $\langle U \rangle$ as a function of the temperature for various densities. Below a critical value T_c, it's thermodynamically more favorable for the system to condense and create dense pockets of particles (droplets) which minimizes the energy. This will, of course, decrease the entropy as well. But for low temperature, it's the energetic term that really contributes to the free energy. On the right, heat capacity plot (obtained by dividing the discrete ΔE by the interval ΔT). For all densities, it seems like there is a peak around a critical value T_c, a behavior reminiscent of phase transition*

close proximity purely by geometric packing constraints (excluded volume), rather than being directed by attractive forces. The difference between $U_{free} = \lim_{T\to\infty} U_{well}(T)$ and $U_{well}(T)$ is precisely the reason why the count of bonds is different for the two systems ($\epsilon = 0$ and $\epsilon = -1$).

An interesting behavior can be observed for low temperatures: once T reaches a certain critical value, the average potential energy abruptly drops in a way that closely resembles a phase transition (enhanced by the peak of the heat capacity $\frac{\Delta E}{\Delta T}$). The low-temperature system is dominated by configurations with relatively high potential energy (in absolute value), meaning spatial configurations with lots of inter-particles bonds (since energy and bonds are connected). Put in other words, the gaseous phase has condensed and has formed droplets of dense liquid (Fig.1.6).

1.2.3 Anisotropic patchy hard spheres

Let us now consider a case where $\theta_{max} < \frac{\pi}{2}$. When setting $\rho = 0.02$ and $k_BT = 0.8$, the system does not display any meaningful behavior (other than the entropy-rich gaseous phase). Conversely, we expect that by decreasing the solid angle where the interaction potential is active, the probability of two particles successfully interacting is significantly reduced compared to the isotropic case ($\theta_{max} = \frac{\pi}{2}$). This reduction in effective attraction is analogous to lowering the critical freezing temperature of the system. Indeed, Fig.1.7 confirms this intuition: for $\theta_{max} = 60°$ or $20°$, a temperature of $k_BT = 0.8$ is too hot to observe any condensation phenomena. Hence, let us simulate a system where $\rho = 0.05$, $k_BT = 0.01$, $\theta_{max} = \frac{\pi}{3}$ or $\theta_{max} = 0.35$ (for 100k MC steps). Snapshots from OVITO rendering are plotted in Fig.1.8 At this low temperature, the system tends to condense and form dense clusters of aligned particles. However, the structure of these clusters is now substantially different: previously, when the potential well was isotropic, the clusters were approximately spherical; now, however, they become increasingly **anisotropic and elongated** as the interaction cone narrows. Setting $\theta_{max} = 0.35$, for example, ensures the steric constraint such that the interaction cone of the potential is tight enough to accommodate only one particle per patch. Consequently, Fig.1.7 clearly demonstrates the formation of long linear chains of particles. When $\theta_{max} = \frac{\pi}{3}$, the cone is large enough to support at least 3 particles, meaning that the chains are much wider and flexible.

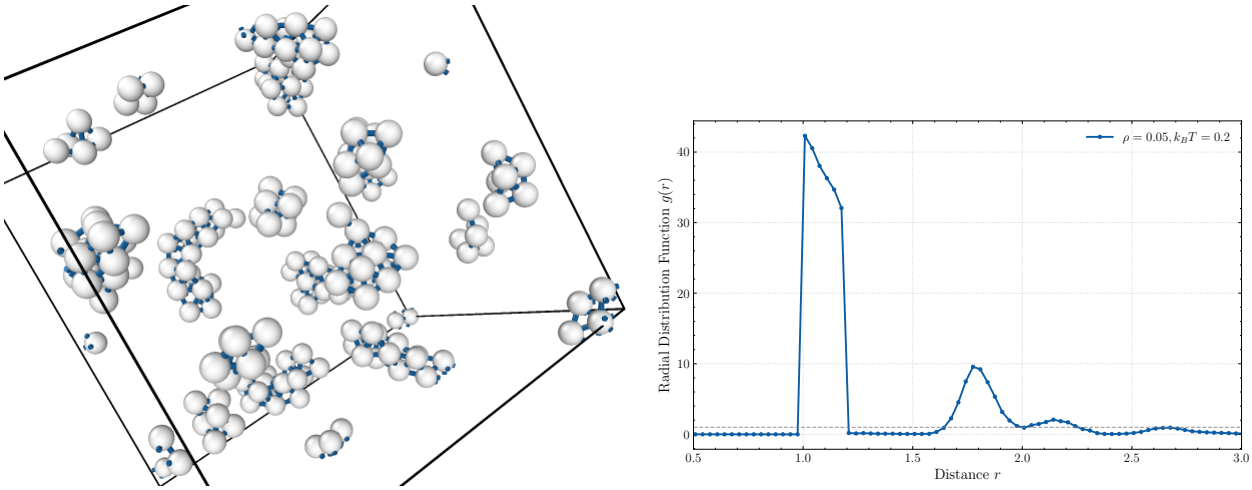

Figure 1.6: *On the right, a snapshot from OVITO rendering for a potential-well system when* $\rho = 0.05$ *and* $k_BT = 0.2$*. What was before a gas of molecules randomly interacting becomes now a system where particles condense and create highly dense clusters with a large amount of internal bonds (thus decreasing the energy). On the left, the radial distribution function for the same configuration*

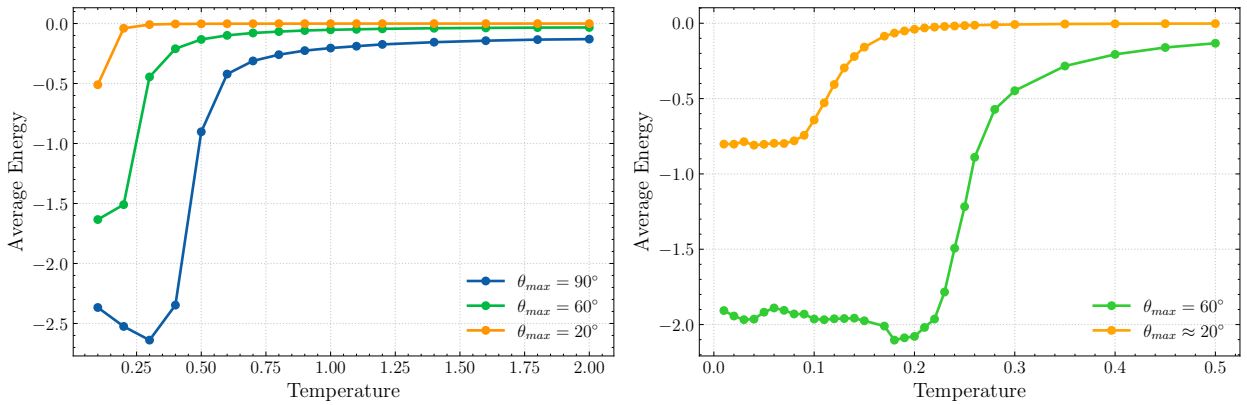

Figure 1.7: *(Left) Average potential energy vs temperature at fixed density* $\rho = 0.05$ *for the patchy hard sphere model for different values of* θ_{max}*. When* $\theta_{max} = 90°$*, then we essentially fallback to the isotropic case discussed in the previous paragraph. As* θ_{max} *approaches* 0*, the system effectively behaves as if no potential well exists and the energy curve flattens out to* 0*. The freezing point (critical* T_c*) becomes closer and closer to* 0 *as* θ_{max} *approaches* 0*. (Right) Focus on the region of very low temperature*

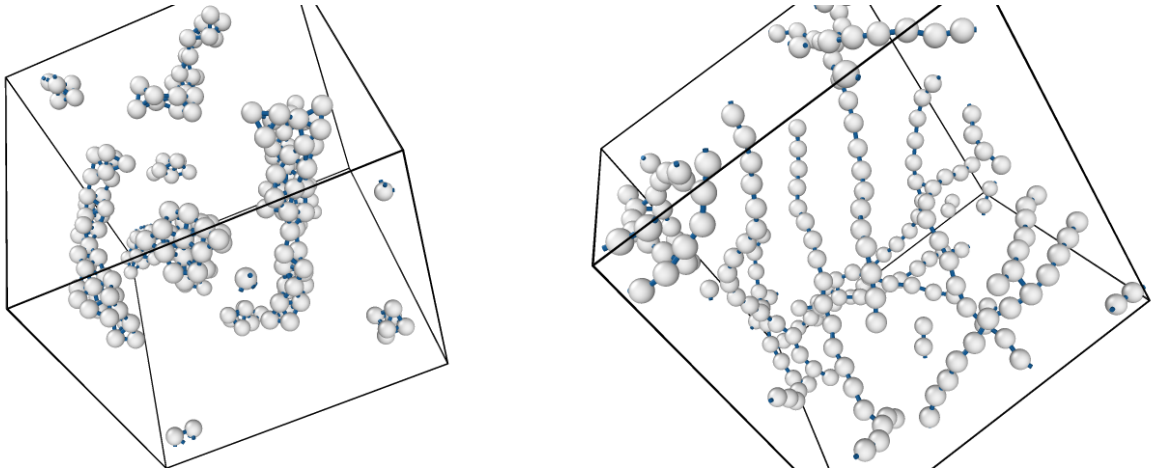

Figure 1.8: *Snapshots from the 3D OVITO render. Parameters of the patchy hard-spheres model:* $\rho = 0.05$, $k_BT = 0.01$, $\theta_{max} = \frac{\pi}{6}$ *(right)* ,$\theta_{max} = 0.35$ *(left) after about* $800k$ *MC steps.*

Bonds and energy With the help of OVITO, let's evaluate the average number of bonds in the system per each one of the run in Fig.1.8. In particular, labeling as "wide" the case where $\theta_{max} = \frac{\pi}{3}$ and as "narrow" when $\theta_{max} = 0.35$:

$$\begin{aligned} b_{wid} &= 470 \pm 1 \\ b_{nar} &= 200 \pm 1 \end{aligned} \tag{1.5}$$

In Fig.1.9 we illustrate the energy of the system as a function of MC steps (it took a lot of MC steps to reach thermal equilibrium and we were not even completely there yet. However, 10^6 MC steps implies approximately 100 GB worth of data to be stored in the memory, so I couldn't proceed any further.). The average energy per particle u, taken considering only the last 10^5 MC iteration is:

$$\begin{aligned} u_{wid} &= -0.955 \pm 0 \\ u_{nar} &= -2.272 \pm 0.002 \end{aligned} \tag{1.6}$$

Note that here it is not strictly true that $u = \frac{b}{N}$, since some OVITO counts as bonds all pair of particles whose distance is less then 1.2, even though they might not be orientated in such a way that they really interact. Still, these observations allow us to extrapolate interesting properties based on the structural measure.

The average number of bonds per particle is approximately ≈ 4.7 (wide patch) or ≈ 2 (narrow patch), which is compatible with the chain structures observed in Fig.1.8. In fact, when $\theta_{max} = 0.35$, almost all particles possess one bond per patch (thus two bonds in total), confirming the formation of long linear chains. Conversely, when $\theta_{max} = \frac{\pi}{3}$, each particle has on average 4.7 bonds, equating to 2.35 bonds per patch. This is still acceptable, since the geometric maximum for such an angle θ_{max} allows for at most 3 bonds per patch. It is possible that, were the system allowed to thermalize further (by increasing the number of MC steps), we would observe a more regular structure with a constant number of bonds per particle.

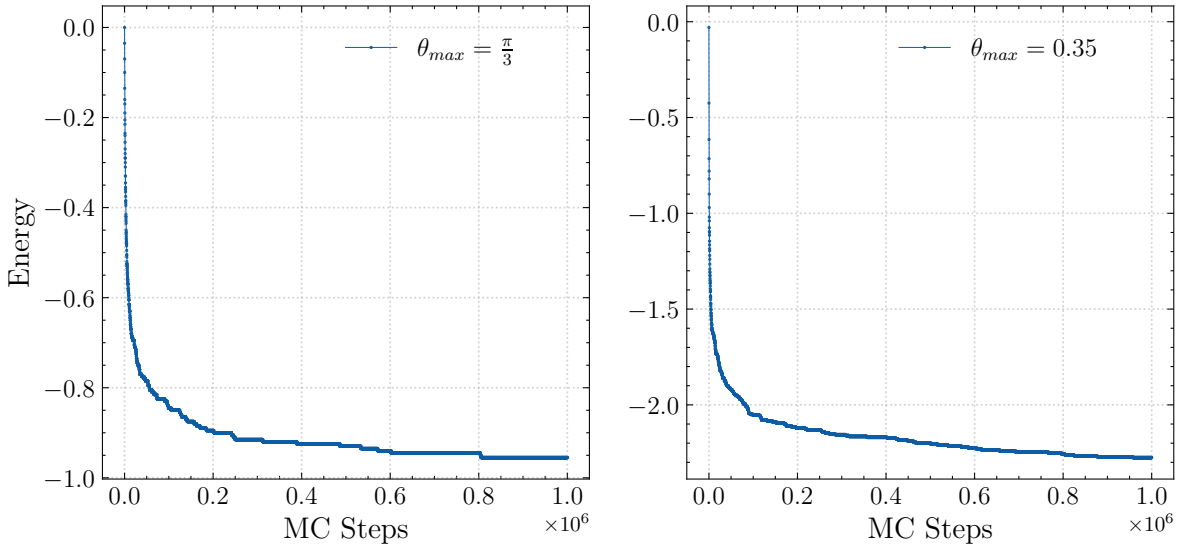

Figure 1.9: *ddd*

Algorithms for Equations of Motion integration

1.3 Harmonic oscillator

Implement the Velocity Verlet algorithm. Consider the harmonic oscillator:

$$\begin{aligned} \dot{q} &= p \\ \dot{p} &= -\omega^2 q \end{aligned} \tag{1.7}$$

Check the energy conservation, plotting $(E(t) - E_0)/E_0$ vs t, E_0 being the initial total energy and the discrepancy with the analytical solution as a function of time. Verify that the Velocity Verlet is stable (i.e. it follows the analytical solution) for $\omega \Delta t < 2$ checking a few values of ω.

Resolution

The system in Eq.1.7 can be quite easily integrated with standard analysis; the solution, given a set of initial conditions (q_0, p_0) reads:

$$\begin{aligned} q(t) &= q_0 \cos(\omega t) + \frac{p_0}{\omega} \sin(\omega t) \\ p(t) &= p_0 \cos(\omega t) - q_0 \omega \sin(\omega t) \end{aligned} \tag{1.8}$$

For the sake of simplicity, we are going to simulate trajectories that started in $t = 0$ at $q_0 = 1, p_0 = \omega$, so that the equations of motion can be simplified:

$$\begin{aligned} q(t) &= \cos(\omega t) \\ p(t) &= -\omega \sin(\omega t) \end{aligned} \tag{1.9}$$

The Hamiltonian for such a system is (remember $m = k = 1$):

$$\mathcal{H} = \frac{1}{2}\omega^2 q^2 + \frac{1}{2}p^2 \tag{1.10}$$

and $\mathcal{H}$ is conserved because this is a Hamiltonian system.

A generic EOM integrator would compute a sequence of states $(q_0 = 1, p_0 = 0; q_1, p_1; q_2, p_2, ..., q_n, p_n)$, at discrete time-steps Δt. In particular, the Velocity Verlet works by updating positions q_i and momenta p_i according to:

$$\begin{aligned} q(t + \Delta t) &= q(t) + \dot{q}(t)\Delta t + \frac{1}{2}F(q(t))\Delta t^2 \\ p(t + \Delta t) &= p(t) + \frac{1}{2}(F(q(t)) - F(q(t + \Delta t)))\Delta t \end{aligned} \tag{1.11}$$

where $F(q(t))$ is the force acting at time t (which should be a function of positions only). It's quite easy to show that the error that one makes when using Velocity Verlet is, at each time steps, of the order of $O(\Delta t^3)$ (local error). However, to fully integrate the equations of motion on a finite interval $[0, T]$, we have to repeat the local algorithm $O(\frac{1}{\Delta t})$, so that errors can build up and the final global error should be on the order of $O(\Delta t^2)$.

A nice property of Velocity Verlet, despite it being a 2nd order integrator, is that it displays symplectic properties. This implies that, although the integrator proceeds in finite timesteps, it always conserves a quantity in phase-space (called the shadow Hamiltonian $\mathcal{H}'$). This is a desirable property, especially in physics where most of the ODEs possess a Hamiltonian structure and the conserved quantity (the Hamiltonian itself) has a direct and fulfilling interpretation, i.e. the total mechanical energy of the system. Other integration schemes (for example, Runge-Kutta methods) are higher-order integrators, hence the error they make is usually smaller, but they fail to correctly preserve the energy over long time windows.

A minimal snapshot of the implemented code for the Velocity Verlet (VV) algorithm is shown here:

```
    def velocity_verlet_step(q, p, dt, a, omega):
        p += 0.5 * a * dt       # Update velocity by half step
        q += p * dt             # Update position by full step
        a_t_dt = -omega**2 * q  # Compute acceleration at new position
        new_a = a_t_dt          # Cache the force, so it only has to be computed once
        p += 0.5 * a_t_dt * dt  # Update velocity by another half step
        return q, p, new_a

    def run(omega, dt, T, q=1, p=0):
        steps = int(T / dt)
        a_t = -omega**2 * q         # Initial acceleration
        q_traj = np.zeros(steps)
        p_traj = np.zeros(steps)
        for i in range(steps):
        q, p, a_t = velocity_verlet_step(q, p, dt, a_t, pmega)
        q_traj[i] = q
        p_traj[i] = p
        return q_traj, p_traj
```

The harmonic oscillator integrated trajectories are displayed in phase space in Fig.1.10 and in a timeplot in Fig.1.11 having set $\omega = 1$. As expected, the trajectory in phase space closely resembles a circle centered in the origin with radius $\omega = 1$. Furthermore, the oscillatory nature of the solutions is nicely reproduced by the algorithm.

As stated before, the VV algorithm is supposed to be symplectic, i.e. it has to conserve a phase-space observable which is allegedly close enough to the actual Hamiltonian of the problem. To verify whether this property truly holds in the our specific case, we study the percentage variation of the empirical energy $E_n = \frac{1}{2}\omega^2 q_n^2 + \frac{1}{2}p_n^2$ compared to the true actual energy $\mathcal{H} = \frac{1}{2}\omega^2 = E_0$ throughout time in Fig.1.12. The characteristic percentage deviation from the actual value E_0 and oscillates back and forth in time around a typical value -1.25×10^{-5} with a period approximately half of the natural period of the coordinates trajectory. In particular, the integrator always underestimate the true energy of the system, but within a reasonable bounded interval. This is a good sign and confirm the symplectic nature of the algorithm, meaning that no energetic drift will appear during the numerical integration of the equations of motion. Another way to verify whether the VV has correctly integrated Eq.1.7 is to simply plot the variation with respect to the analytical form of $q(t), p(t)$: this is done in Fig.1.13[6]. The variation for both q and p is of the order of 1%, hence we can be satisfied with the algorithm's output.

[6]If the coordinate q is of order 1, then p is usually of order ω, so we normalize $p(t)/\omega$ to get realistic results.

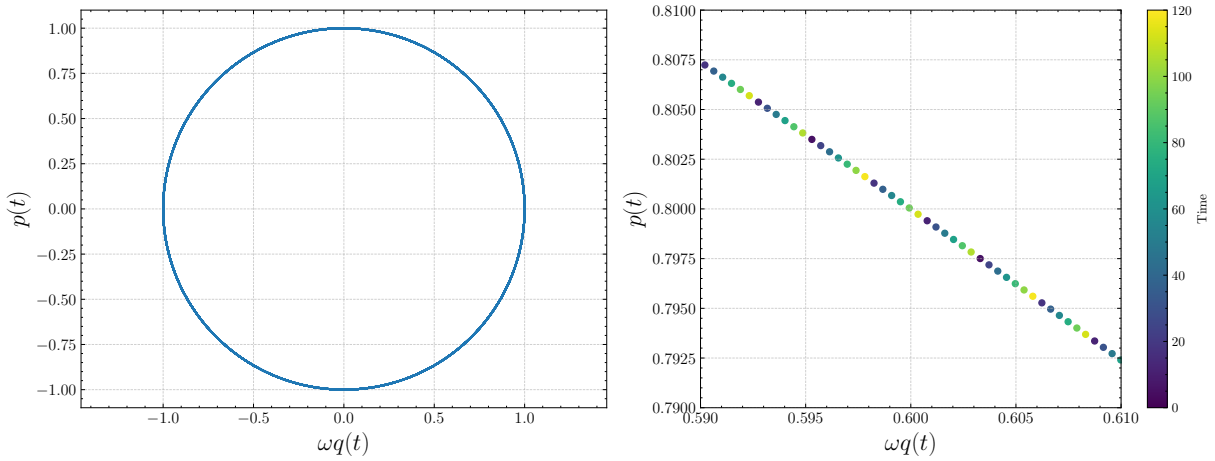

Figure 1.10: *On the right, phase space representation of the simulated trajectories when* $\Delta t = 0.01$ *integrating from* $t = 0$ *up to* $T = 120$*. As one would expect, the trajectory. On the left, a close up view of a smaller portion of the plane region, where the granularity of the algorithm becomes clearer (and the arrow of time is color coded.)*

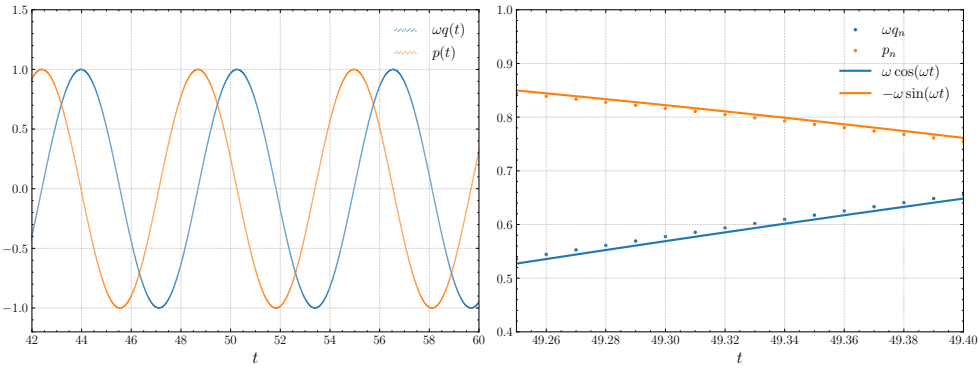

Figure 1.11: *Velocity Verlet integrated curves for* $\omega q(t)$ *and* $p(t)$ *(on the left, a closer and zoomed view to better see the discrete points). Overall, the algorithm has found a seemingly convincing trajectory reproducing the typical oscillations of the physical problem. It looks like a small bias is present in all steps, but this will be better studied later on*

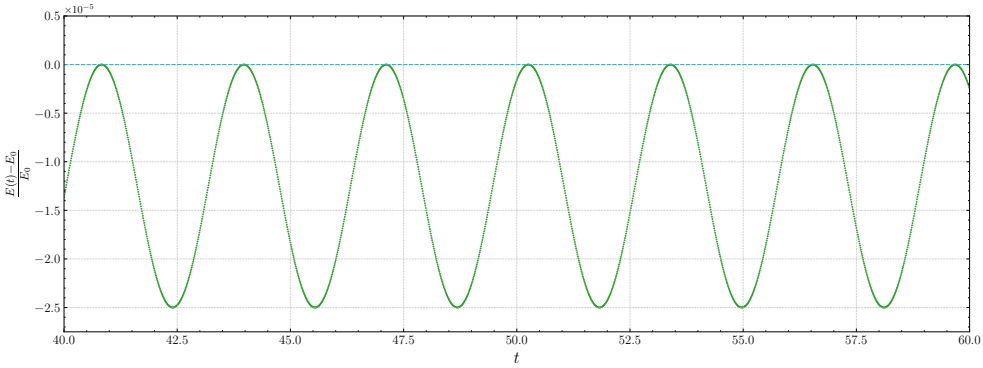

Figure 1.12: *Percentage variation of integrated energy* $E(t)$ *with respect to the constant true value* E_0*. The plot clearly shows that the integrator always underestimate the energy by an oscillating amount bounded by* $\max \Delta E \approx -2.5 \times 10^{-5} E_0$*. Overall, the symplectic nature of the VV algorithm is confirmed.*

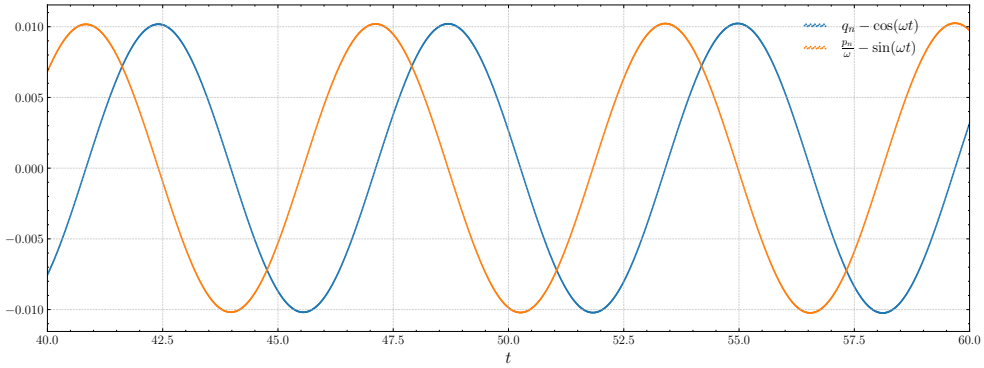

Figure 1.13: *Relative variation of $q(t), p(t)$ with respect to the true analytical curves as a function of time. Similarly to Fig.1.12, the deviation oscillates back and forth around 0 and stays bounded within $\approx 1\%$.*

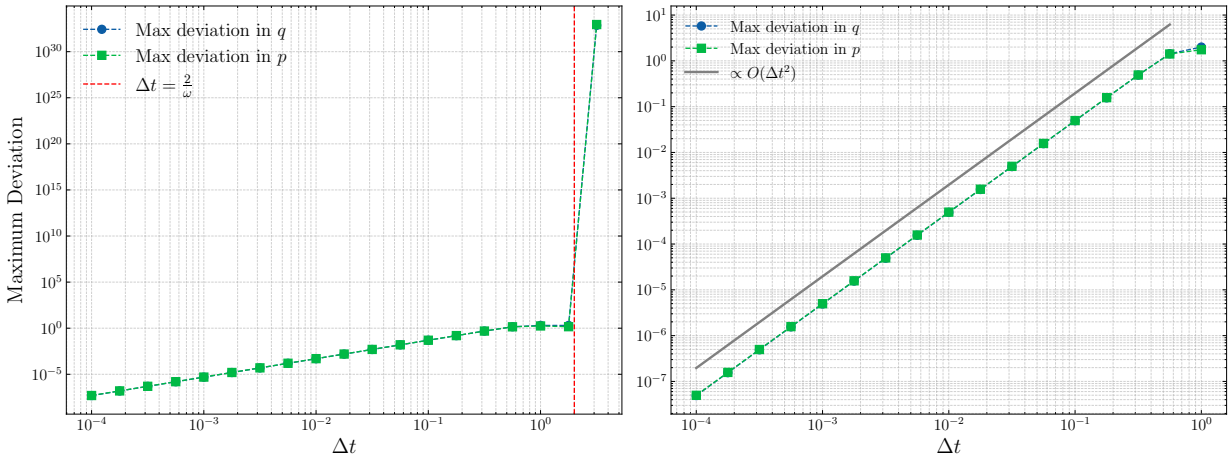

Figure 1.14: *Algorithmic errors (intended as maximum deviations of $q(t), p(t)$ from their theoretical curves) vs time-step Δt, keeping $\omega = 1$ fixed. The right plot highlights the existence of a threshold at $\Delta t = \frac{2}{\omega}$ above which the error skyrockets and the integrator completely fails. The left plot can be used to infer the scaling behavior of the error with respect to Δt, $\Delta q \propto O(\Delta t^2)$ (a typical property of a 2nd order integrator such as VV).*

Maximum deviation Based on the trajectories in Fig.1.13, we can extract the maximum deviation for both q and p (denoted as $\Delta q, \Delta p$) and interpret it as the typical error incurred by the Velocity Verlet (VV) algorithm when the time-step is set to $\Delta t = 0.01$. We expect that decreasing Δt will lead to smaller errors, although at a higher computational cost: to verify this behavior, we compute the associated error across a range of Δt values (Fig.1.14). The Verlet algorithm is found to be stable up to a threshold value $\Delta t^\star$ satisfying $\omega \Delta t^\star = 2$; beyond this limit, the algorithm becomes unstable, yielding divergent trajectories (left panel of Fig.1.14).

Furthermore, the right panel of the same figure highlights another key feature of the algorithm: the characteristic scaling of the error with respect to the time-step Δt (at fixed ω). Since VV is a second-order integrator, we expect a scaling behavior of $\Delta q, \Delta p \sim O(\Delta t^2)$, which is precisely what the numerical results confirm (at least up to the critical $\Delta t^\star$).

The same plot can be realized by keeping Δt fixed to $\Delta t = 0.01$ and letting ω vary (Fig.1.15). Again, we observe a critical threshold when $\omega = \frac{2}{\Delta t}$ above which the Python interpreter failed because

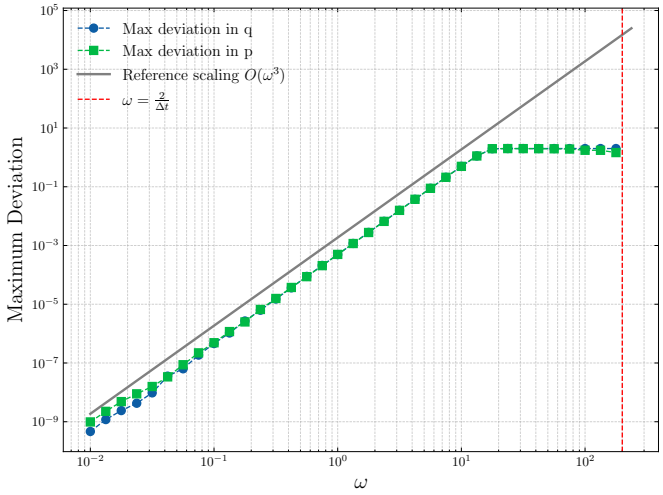

Figure 1.15: *Algorithmic errors (intended as maximum deviations of $q(t), p(t)$ from their theoretical curves) vs ω, this time keeping $\Delta t = 0.01$ fixed. The same critical value $\omega = \frac{2}{\Delta t}$ can be seen from a different perspective (no points over that value since overflows errors hindered the numerical integration). A scaling behavior $O(\omega^3)$ emerges.*

of a overflow error, suggesting the trajectory diverged and exploded to infinity. The scaling of the error is now $O(\omega^3)$.

1.4 Lennard-Jones fluid in the NVE ensemble

The purpose of this section is to simulate the dynamical evolution of a system composed of $N = 200$ interacting particles using MD techniques to infer thermodynamical quantities such as the internal energy or structural properties like the radial distribution function. The simulation is carried on using a dedicated and powerful software called LAMMPS, which enables us to instantiate and inspect highly-customizable Molecular-Dynamics simulation.

LAMMPS can be readily used in the terminal given a script file is provided to him. The script file should contain all the details that the LAMMPS MD engine will need to run the computations. An instructive section of the script that I've used is reported here for the sake of completeness:

- Step 1: Initialization. In this section, we tell LAMMPS the geometric properties of our simulation:

```
# 1) Initialization
units lj
dimension 3
atom_style atomic
boundary p p p

# 2) System definition
region simbox block 0 6.85824 0 6.85824 0 6.85824
create_box 1 simbox
create_atoms 1 random 200 34134 simbox overlap 0.3
```

This portion of the script will instruct LAMMPS to generate a $3D$ box with periodic boundary condition whose size is $L = 6.85824$. This value of L was chosen such that the final volume of the simulation $V = L^3$ will yield the target numerical density $\rho = 0.62$. The last line will make

LAMMPS generate particles at random trying the avoid overlaps within 0.3 units from the center of each molecule.[7]

- Step 2: settings This section will define the parameters associated with the pairwise potential making particles interact. In this case, we're going to use the Lennard-Jones potential:

$$V(r) = 4\epsilon\left(\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right) \tag{1.12}$$

units are rescaled in such a way that $\sigma = 1$ and $\epsilon = 1$.

```
1        # 3) Settings
2        mass 1 1.0
3        pair_style lj/cut 4
4        pair_coeff 1 1 1.0 1.0
5        pair_modify shift yes
```

The second line allows us to select the cut-off radius of the potential, i.e. the distance over which particles don't feel any type of interaction. In this report, we are going to examine what happens to the moleculars systems when $r_{cut} = 4, 3, 2.5, 2^{1/6}$. The last value was chosen because it is precisely the stable equilibrium point for the pair-wise Lennard-Jones potential, where particles feel no net force. In practice, when $r_{cut} = 2^{1/6}$ we only retain the repulsive part of the Lennard-Jones potential, reducing the problem to a simple hard-sphere simulation that we already encountered in the previous chapter.

The last line `pair_modify shift yes` is crucial: if we simply cut the LJ potential for $r < r_{cut}$ and assume $V(r) = 0$ for $r > r_{cut}$, then we'd introduce a steep potential discontinuity in r_c (in general $V_{LJ}(r_{cut} \neq 0)$). This translates in an impulsive non-conservative force that will break the energy conservation condition. To make sure this doesn't happen, we have to add a constant energy term to the LJ potential to ensure continuity in $r = r_{cut}$, which is what we do with that last line in the script. A more detailed discussion of this problem is discussed and illustrated in Fig.1.16

- Step 3: Minimization During the generation phase, LAMMPS will simply place molecules in a random fashion, uniformly within the volume L^3. However, especially when the numerical density is large, there's a high chance that two particles are created very close one to each other, experiencing extremely large and potentially unstable forces. To avoid this problem, we will let LAMMPS run a pre-processing phase whose main goal is to find a spatial configuration that minimizes the (potential) energy:

```
1        minimize 1.0e-6 1.0e-6 1000 10000
```

Note that this is not yet the numerical integration section

- Step 4: Thermalization As of now, we have only prepared the state trying to find a non-overlapping spatial configuration where the potential energy is stable. We still haven't dealt with the kinetic part of the system, as the velocities of the particles were left untouched. Since we want to run a energy-conserving simulations where $T = 1$, we should sample the particles' velocities according to the correct Maxwell-Boltzmann distribution (ensuring no net drift). An alternative way to initialize the particles momenta is to introduce with LAMMPS a temporary thermostat that will bring the temperature of the system up to the desired value:

[7] Spatial units are rescaled in terms of molecular sizes, as usual in MD simulation

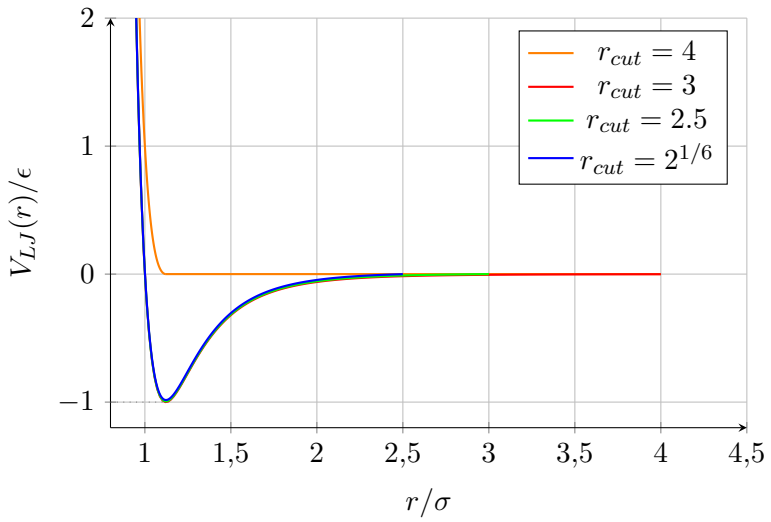

Figure 1.16: *Lennard-Jones potential cut at different distances r_{cut}. Note that cutting the potential at small r will introduce a discontinuity in the $V(r)$, i.e. a delta-shaped instantaneous force which has no physical relevance and can inject spurious energy absorption into the system. To counterbalance this problem, one should just add a constant term $+V_{LJ}(r_{cut})$. This procedure has little effect on $r_{cut} = 4, 3, 2.5$ where $V(r_{cut}) \approx 0$ but must be implemented on $r_{cut} = 2^{1/6}$ where $V_{LJ}(2^{1/6}) = -1$. The resulting potential is also called Weeks-Chandler-Andersen (WSA) potential in the scientific community*

```
fix equil all langevin 1.0 1.0 0.1 99999
fix integrate all nve
thermo 100
run 5000
# Now we can turn off the thermostat
unfix equil
unfix integrate
```

At the end of this stage, we can remove the thermostat. By now, all velocities should be initialized in such a way that $T = 1$.

- Verlet integration We have correctly initialized the spatial configurations and the velocities of the system: we can now run the MD simulations selecting the timestep Δt and the total integration time:

```
# 5) Finally, run the MD simulation
reset_timestep 0
fix mynve all nve
timestep 0.005
run 1000000
```

Note that the line `fix mynve all nve` will use Verlet algorithm under the hood, so we expect energy to be conserved.

1.4.1 Energy conservation

Let E_0 be the initial energy of the system:

$$E_0 = K_0 + V_0 = \sum_i \frac{1}{2} p_i^2(t=0) + \sum_{i \neq j} V_{LJ}(r_{ij}(t=0))$$

r_{cut}	Average $\bar{T}$	St Dev σ_T
4	1.038	0.0320
3.5	1.0464	0.0320
2.5	1.0700	0.0323
1.122	1.0462	0.0282

r_{cut}	Average $\bar{T}$	St Dev σ_T
4	0.9685	0.0453
3.5	0.9775	0.0447
2.5	1.0413	0.0394
1.122	1.0493	0.0204

Table 1.2: *Average temperature and associated standard deviation at different values of r_{cut} when $\rho = 0.62$ (right) and $\rho = 0.4$ (left).*

where the $p_i(t = 0)$ were sampled from the Maxwell-Boltzmann at temperature $T = 1$ thanks to the thermostat phase in LAMMPS and the configuration was chosen at random while keeping sufficient distance between the molecules. The time evolution of the total energy is reported in Fig.1.17 for $\rho = 0.62$ and $\rho = 0.4$. Along with the relative energy variation, Fig.1.12 also displays the behavior of the temperature T as a function of time.

Overall, the VV integrator has produced valid trajectories and the energy has approximately stayed constant: in the worst scenario (when $r_{cut} = 1.12 \approx 2^{1/6}$), the relative variation with respect to E_0 was about $\approx 0.1\%$, which is a solid result and an usually acceptable value when performing MD simulations. Furthermore, one has to consider that the simulation was run until 10^6 steps were reached: other more sophisticated integration schemes (e.g. Runge-Kutta methods) would have probably better approximated the system during the initial steps but would have yielded a noticeable and unacceptable energy drift over the course of 1 million time-steps.

Let us now analyze individual results. When $r_{cut} = 4$, the energy-conservation plot is surprisingly nice for both $\rho = 0.62$ and $\rho = 0.4$: small energy fluctuations of the order of $10^{-4}E_0$ can still be appreciated, but no overall energy drift is detected. When $\rho = 0.62$ and $r_{cut} = 3.0$, we observe an interesting phenomena: the system experiences a small positive drift during the first half of the simulation3 and then stabilizes around a fixed plateau (up to some small fluctuations). This is both a good and a bad news: during the first steps, the integrator has injected a small and unphysical amount of energy into the system, breaking energy conservation. However, this trend eventually came to an end and the integrator regained its ability to maintain the energy stationary. Strictly speaking, to obtain stable and robust results, we should have only considered the system in its stationary phase (even though here the energy is different than the one we started with); however, the drift in energy is well below the acceptable threshold, so we included all the time evolution in the analysis to come. The same reasoning goes for the most unstable configurations with $r_{cut} = 2.5$ and $r_{cut} = 2^{1/6}$, where a stationary phase (meaning no drift, just small fluctuations) is harder to find and multiple drifting phases coexist. Still, the overall energy variation is well bounded within $\approx 0.1\%$, hence we can proceed with our analysis.

Another interesting aspect to study is the time evolution of the temperature T which can (and should) vary over time, since we are in a microcanonical ensemble, not in a canonical one. Overall, we expect a temperature fluctuating around the fixed value $\langle T \rangle = 1$, and now the entity of the fluctuations is macroscopic ($\approx 10/20\%$) and has relevant physical properties. Fig.1.17 partially agrees with this theoretical analysis in all cases, but there are small deviations from the expected behavior of $T(t)$. For example, computing the average T over all MD samples, we always observe a slightly different value with respect to $\langle T \rangle = 1$, suggesting a leak (or an injection) of kinetic energy from (into) the system (Table 1.2). This is, however, to be expected: from the analysis of the energy $E(t)$ over time, we observed small inconsistencies due to the imperfect nature of the algorithm. Those spurious and unphysical energy leaks/injections have direct effect on the kinetic energy, and, consequently, on the temperature T. As long as we don't measure huge deviations from $\langle T \rangle = 1$, we can safely use those results to build robust analysis.

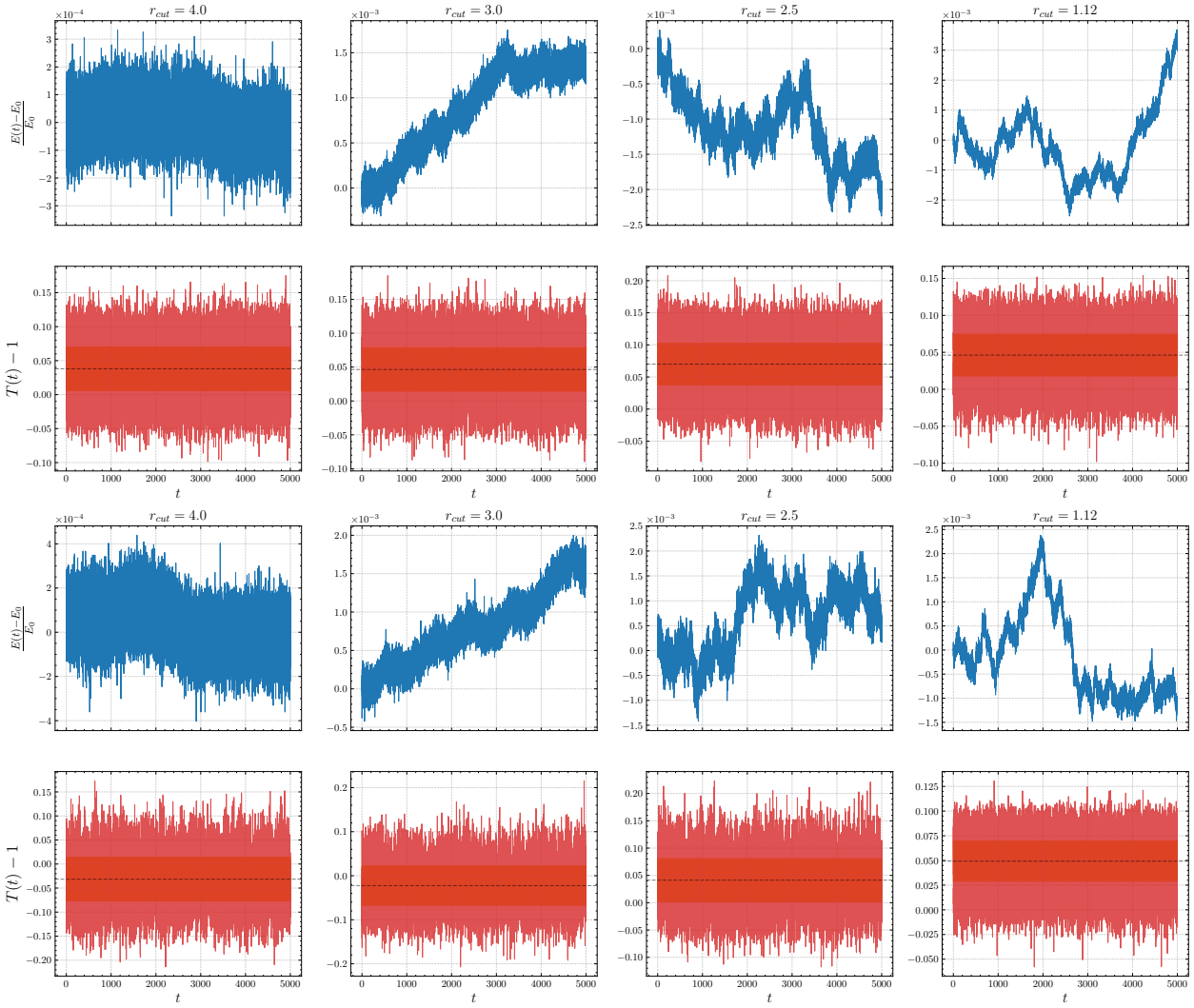

Figure 1.17: *Energy and temperature plots at different cut-off radius r_{cut} when $\rho = 0.62$ (first two rows) and $\rho = 0.4$ (last two rows). The time-step was set to $\Delta t = 0.005$ and the number of steps is 10^6. Overall, the energy deviations and the average temperature are well behaved and bounded. In the temperatures plots, the average $\bar{T}$ is represented with a dashed line and the standard deviation with a solid band.*

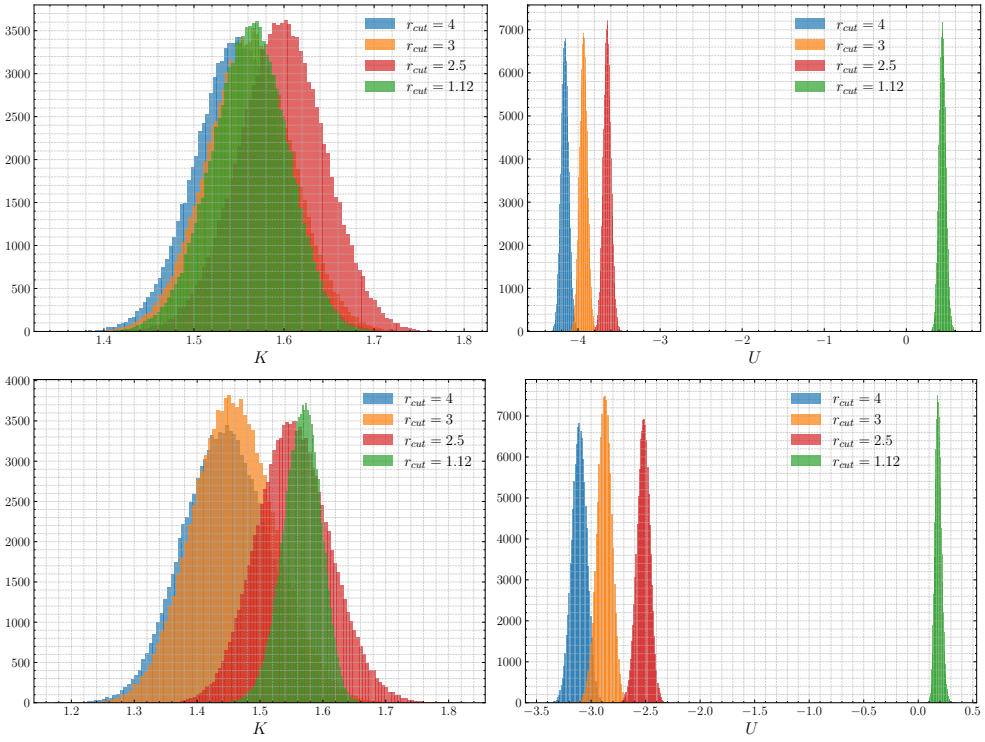

Figure 1.18: *Histograms for the kinetic energy per particle (on the right) and the potential energy per particle (on the left) at various cut-off distances when* $\rho = 0.62$ *(top band)* and $\rho = 0.4$ (bottom band)

1.4.2 Potential and kinetic energy

Provided our simulations have faithfully conserved the total energy $E(t)$ within reasonable bounds, we can start analyzing other physical properties of interest like the potential and kinetic energy. The resulting histograms for those observables (normalized per number of particles) are reported in Fig.1.18. Let us first derive how the kinetic energy histogram should look like. By invoking the equipartition theorem, we can write:

$$\langle K \rangle = \frac{3}{2} N k_B \langle T \rangle \tag{1.13}$$

for monoatomic gases. Note this is absolutely independent of the potential shape, even more so from the cut-off radius. Since we work with $k_B = 1$ and $T = 1$, we should see a kinetic energy per particle $K = 1.5$ *independent of* r_{cut}. The kinetic energy histograms in Fig. 1.18 are compatible with this computation and closely resemble gaussian distributions, but their average values is slightly different from the expected $K = 1.5$, especially for $\rho = 0.62$ when all the 4 simulations tend to overestimate the correct value of K. I believe this is again a matter of numerical instability of the integrator and a direct consequence of the total temperature drift that was commented in the previous subsection: in fact, one can easily see that the worst results for the kinetic energy (that is, the farthest from $K = 1.5$) were obtained when $\rho = 0.62$ and $r_{cut} = 2.5$, the same set of parameters that yielded the worst temperature average in Table 1.2. The average kinetic energies are, however, perfectly compatible with the average measured temperature, so that those two anomalies are intimately linked.

Now let us analyze the behavior of the potential energy. When $r_{cut} = 4, 3, 2.5$ the attractive well of the Lennard-Jones potential is kept (Fig.1.16) and allows for the formation of persistent inter-molecules bonds, resulting in negative potential energies per particles. The largest r_{cut} is, the more we keep of the attractive portion of the potential and the final average potential energy per particle gets more

and mode negative (until an asymptotic value is reached for $r_{cut} = \infty$). However, if we truncate the potential at $r_{cut} = 2^{1/6}$, we completely remove the attractive property of the Lennard-Jones potential and the molecules simply behave by repelling each other if they get too close one to another. This type of truncation, also called *WCA potential*, is a continuous version of the hard sphere model where particles don't interact with each other apart from the exclusion volume repulsion. The potential energy now becomes positive (but this is simply due to the constant added term we already discussed) and, most importantly, close to 0, precisely what we would expect from a hard sphere model

Observing Fig.1.18, the average potential energy approaches 0 but does not vanish completely. This occurs because the WCA potential is steep yet differentiable; consequently, particles with sufficiently high kinetic energy can temporarily penetrate the distance $r < 1$, resulting in a significant gain in potential energy. This effect becomes much more evident as the system density increases, since the crowdedness and the reduced free volume forces particles to interact more frequently with their neighbors.

1.4.3 Radial distribution function

Finally, we analyze the radial distribution function $g(r)$ of the system, computed via molecular dynamics simulations using LAMMPS. The results are plotted in Fig.1.19. By setting the cutoff radius to $r_{cut} = 4$, we essentially capture the full range of the interaction; consequently, we observe the typical oscillatory behavior characteristic of the Lennard-Jones potential, where peaks alternate with minima. This structural ordering indicates that the fluid possesses significant short-range order. The first sharp peak corresponds to the nearest-neighbor shell, located approximately at the distance where the Lennard-Jones potential reaches its minimum (around $r \approx 2^{1/6}\sigma$). This implies that particles tend to cluster at this energetically favorable distance. Subsequent peaks represent further coordination shells (next-nearest neighbors). However, as r increases, these oscillations progressively vanishes and $g(r)$ converges to 1. This decay implies the absence of long-range order, confirming that the system is indeed in a fluid phase where spatial correlations vanish over long distances but are fundamental at small scales. On the contrary, when $r_{cut} = 2^{1/6}$ the system essentially behaves as a hard sphere model: hence, the radial correlation completely vanishes when $r > 2^{1/6}$ since particles only interact thanks to the excluded volume. Note that the curve for $r_{cut} = 2^{1/6}$ is in agreement with the one in Fig.1.2

For intermediate cut-off radius, the radial distribution curves are similar to the one obtained for $r_{cut} = 4$, since we are still retaining most of the initial Lennard-Jones potential. This is particularly evident for $\rho = 0.62$ but becomes less noticeable when $\rho = 0.4$, where the three different curves (the one for $r_{cut} = 4, 3, 2.5$) start to become diverge one from another. This is because when $\rho = 0.6$, the typical distance between molecules is $\frac{1}{\rho} \approx 1.6$. It's highly improbable the a pair of particle is found at a distance much larger than 1.6, hence the effect of a cutoff is negligible (as long as $r_{cut} >> \frac{1}{\rho}$). When $\rho = 0.4$, the typical distance is now 2.5, hence with our choice of cut-off radius we should start to observe some discrepancies in the three curves. As expected, the larger the value of r_{cut} the higher the first peak gets, since larger r_{cut} implies more attraction between particles (and more short-range structure)

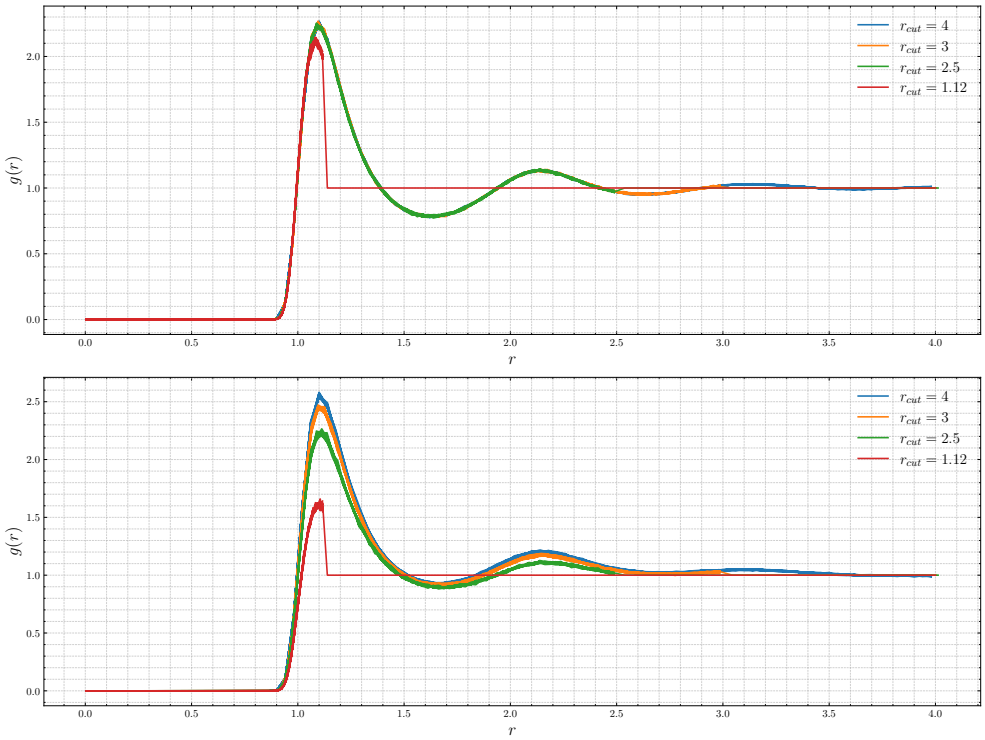

Figure 1.19: *Radial distribution function $g(r)$ for the Lennard-Jones model at different cut-off radius when $\rho = 0.62$ (above) and $\rho = 0.4$ (below)*