Event-driven molecular dynamics

Goal: Learn simulating particles that interact via hard-core potentials.

So far in our MD simulations, we considered particles interacting via a spatially continuous potential. In contrast, event-driven MD can be used to simulate particles interacting via hard-core potentials by simply considering binary collisions (no lasting contacts). More concretely, assuming no external forces, the particle trajectories can be analytically calculated, i.e. *they follow a straight line between collisions*[1].

The time t_{ij} between the latest and the next collisions between a pair of particles i and j can be calculated as follows

$$|\vec{r}_{ij}(t_0) + \vec{v}_{ij}(t_0)t_{ij}| = \sigma_i + \sigma_j, \tag{1}$$

where $\vec{r}_{ij}$ is the relative position vector between particles i and j, $\vec{v}_{ij}$ is the relative velocity, σ_i the radii of the hard spheres and t_0 is the time step at which the latest collision occurred. To get the "global" collision time, $t_C := \min_{ij}(t_{ij})$, naïvely *one needs to compute the collision time between all particle pairs.*

Task 1: *Assuming that all hard-spheres have identical masses $m_i = m$ and radii $\sigma_i = \sigma$, implement the event-driven dynamics in 1- and 2-dimensions (assume also perfect-slip collisions between hard-disks).*

Hint: Note that, only the neighbouring hard-rods interact with each other in 1-dimension. There you can set the width (radius) of the rods to $\sigma_i = 0$.

Task 2: *Investigate different scenarios:*

- *Simulate a 1-dimensional row of N beads in a box with restitution coefficient $e = 1$, and validate your implementation conserves the total energy.*
- *Consider a 1-dimensional row of N beads with $e < 1$ hitting a resting wall. Compute the effective restitution coefficient $e_{\text{eff}} := \sqrt{E_f/E_i}$, where E_i and E_f are the initial and final total kinetic energies of the system, respectively. Vary N at fixed e. Determine $N = N_c$, at which e_{eff} practically vanishes, in other words, the cluster of beads ceases to re-bounce.*

Task 3 (optional): *Improve the efficiency of your code by storing the events for each particle in a priority queue (see lectures). While checking all particle pairs requires $\mathcal{O}(N^2)$ operations, generating events only for the colliding particles and inserting them in the queue has a complexity*[2] *of only $\mathcal{O}(N \log N)$.*

Hint: Only the interacting particles need to be advanced, but keep in mind that the collision can invalidate the previously predicted events. You can either remove them manually from the queue in time $\mathcal{O}(N)$ or, more efficiently, keep a counter of each particle's collisions and identify an event as invalid only when resolving it.

Solution. Task 1: 2-d system in a box. We assume identical radii $\sigma_i = \sigma$ for the disks and consider two types of collisions:

- Particle-wall collisions: For a given hard-disk, given the position $\vec{r}$ and the velocity $\vec{v}$, we take the smallest of the collision times with the four walls:

$$t_{1\alpha} = \frac{L - \sigma - r_\alpha}{v_\alpha}, \qquad t_{2\alpha} = \frac{\sigma - r_\alpha}{v_\alpha}, \tag{S.1}$$

[1]In fact, note that, the event driven algorithm is *exact* if the velocities and the collision times are calculated with *infinite* precision. However, computers truncate floating point numbers at a finite precision. This is particularly important in dimensions > 1 because there the negative curvature of the surface of the sphere leads to *chaos* in the system, *i.e.* there can be the vast differences in the final states despite having only marginal differences in the initial conditions. Therefore, the event-driven MD simulations would be valid only for a limited number of time steps despite the numbers calculated by the algorithm have millions of digits.

[2]Intuitively, the bound is even tighter in 1-dimension: methods for an insertion time of $\mathcal{O}(1)$, *i.e.* $\mathcal{O}(N)$ in total, have been proposed. See, *e.g.*, `https://arxiv.org/pdf/physics/0606226.pdf`.

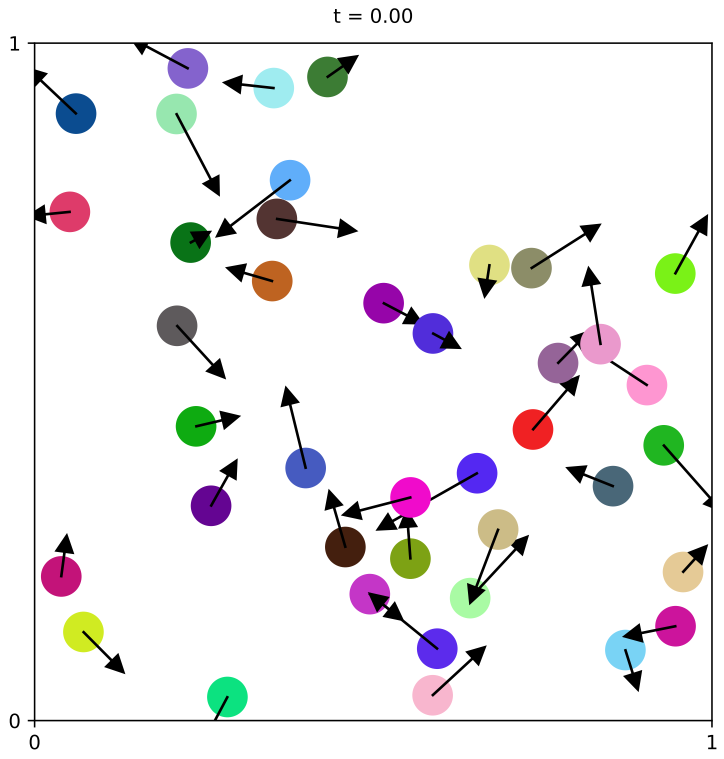

Figure 1: A snapshot from event driven simulation of hard-disks with $\sigma = 0.03$, $N = 40$ and $L = 1$. The arrows denote the velocity vectors.

where $\alpha = x, y$ denotes the spatial component of the vectors.

- Collisions between hard-disks: Given $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ and $\vec{v}_{ij} = \vec{v}_i - \vec{v}_j$ for a pair of particles i and j, we can compute the times:

$$t_{ij} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \tag{S.2}$$

where $a = v_{ij}^2$, $b = 2\vec{v}_{ij} \cdot \vec{r}_{ij}$ and $c = r_{ij}^2 - (2\sigma)^2$. If $a = 0$, *i.e.* if particles do not move relative to each other, or $b < 4ac$, *i.e.* particles cross their trajectories but at different times, no collision occurs and we set the collision time $t_C = \infty$. Otherwise we choose the smallest positive solution as the collision time.

Particles are then advanced in straight lines until the earliest collision. If the collision is with a wall in direction α, the velocity is simply reflected: $v_\alpha \leftarrow -v_\alpha$ and $v_\beta \leftarrow v_\beta$ for $\beta \neq \alpha$, whereas after a pairwise collision, the velocity update is given by

$$\begin{aligned} \vec{v}_i &\leftarrow \vec{v}_i + \hat{r}_{ij}(\vec{v} \cdot \hat{r}_{ij}), \\ \vec{v}_j &\leftarrow \vec{v}_j - \hat{r}_{ij}(\vec{v} \cdot \hat{r}_{ij}) \end{aligned} \tag{S.3}$$

where $\hat{r}_{ij} := \vec{r}_{ij}/r_{ij}$. In case of inelastic collisions, the updated velocities after a pairwise collision are further scaled by the restitution constant e.

The algorithm then proceeds by iteratively determining the earliest collisions (either with a wall or between disks) and velocity updates as described. As a brief illustration, in Fig. 1 a snapshot

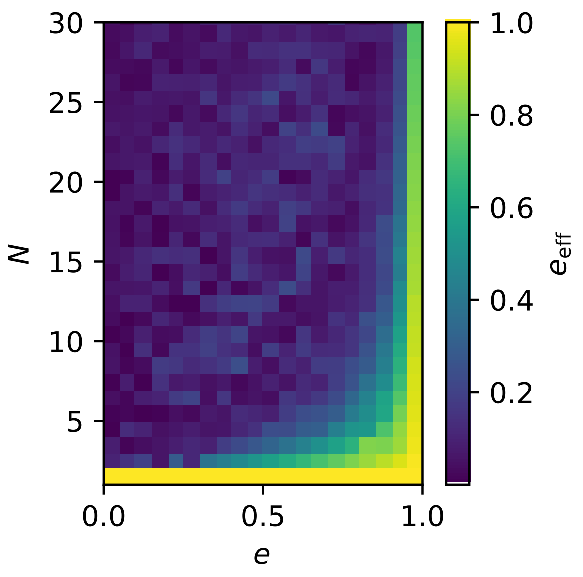

Figure 2: Phase diagram of a chain of dissipative ($e < 1$) rods thrown at a resting wall.

from a simulation with $\sigma = 0.03$, $N = 40$ and $L = 1$ is shown. Note that generating random initial conditions in a box can be non-trivial when the disks are densely packed.

Task 2: Smashing 1-dimensional rods onto a wall. Pairwise collision times are trivially determined in 1-dimension:

$$t_{ij} = \frac{r_{ij}}{v_{ij}} - \frac{2\sigma}{|v_{ij}|}, \tag{S.4}$$

with $r_{ij} = r_i - r_j$, $v_{ij} = v_i - v_j$ and the collision time is set to infinity if $t_{ij} < 0$. Likewise the collision times are

$$t_1 = -\frac{r}{v} + \frac{(L-\sigma)}{v}, \qquad t_2 = \frac{r}{|v|} - \frac{\sigma}{|v|}, \tag{S.5}$$

which respectively correspond to collisions with the right and the left walls. Note that it is useful to set a threshold and manually set $t_C \to \infty$ when the relative velocity is very close to 0 to avoid numerical instabilities.

Finally we consider experiment where a row of rods colliding inelastically are thrown at a resting elastic wall. The qualitative observation is that, when sufficiently many of the beads are launched towards the wall, the energy of the system is drastically dissipated in collisions between the rods still moving towards the wall and the ones bouncing back. Effectively, this mechanism forms a cluster that comes to a stand still near the wall.

We quantify the dependence of this behaviour on the total number of rods thrown at the wall and the restitution constant e by computing the effective restitution constant $e_{\text{eff}} = \sqrt{E_f/E_i}$. Note that we are considering a system with a wall on only one of the boundaries. The phase diagram of this system is shown Fig. 2. It suggests that e_{eff} can vanish for any $e < 1$, nevertheless, the sufficient number of particles for this grows exponentially in e.