LAMMPS Tutorial - Brownian Dynamics Simulation

Introduction to BD and LAMMPS

February 5, 2026

UNIVERSITY OF TURKU

Outline

1. Recap: Understanding Brownian Dynamics Simulation

2. Hands On: Performing Your First Brownian Dynamics (BD) Simulation

3. Hands On: Beyond Brownian Dynamics; Capturing Motion at the Scale of Relaxation Time

4. Continuum Models (Voluntary Tasks) and Conclusion

Recap: Understanding Brownian Dynamics Simulation

What is Brownian Dynamics?

Key idea

Brownian Dynamics (BD) models the motion of particles in a solvent **without explicitly simulating solvent molecules**.

- Thermal motion arises from **random collisions**
- Solvent effects are replaced by:
  - Friction (dissipation)
  - Random forces (noise)
- Ideal for:
  - Soft matter
  - Polymers
  - Biomolecules

Brownian Dynamics: Kicking each other without realizing the same.

Passive Motion

Passive particles do not possess self-propulsion. Their motion arises from thermal fluctuations of the surrounding fluid and viscous dissipation.

Langevin Equation

The dynamics of a passive Brownian particle of mass m is governed by

$$m\frac{d\mathbf{v}}{dt} = -\gamma\mathbf{v} + \mathbf{F}_{\text{ext}} + \boldsymbol{\xi}(t),$$

where

- γ is the friction coefficient,
- $\mathbf{F}_{\text{ext}}$ is an external deterministic force,
- $\boldsymbol{\xi}(t)$ is Gaussian white noise with

$$\langle \boldsymbol{\xi}(t) \rangle = 0, \quad \langle \boldsymbol{\xi}(t)\boldsymbol{\xi}(t') \rangle = 2\gamma k_B T\, \delta(t - t').$$

Brownian Dynamics Limit

Brownian dynamics corresponds to the **overdamped limit**:

$$m\frac{d\mathbf{v}}{dt} \ll \gamma\mathbf{v}$$

or equivalently,

$$\mathrm{Re} \ll 1, \quad t \gg \tau,$$

where

$$\tau = \frac{m}{\gamma}$$

is the **momentum relaxation time**, and

$$\mathrm{Re} = \frac{\rho v L}{\eta}$$

is the **Reynolds number**.

Brownian Dynamics Limit

In this regime, inertia is negligible and the Langevin equation reduces to

$$\gamma \frac{d\mathbf{r}}{dt} = \mathbf{F}_{\text{ext}} + \boldsymbol{\xi}(t),$$

where $\boldsymbol{\xi}(t)$ represents thermal noise, also known as Gaussian white noize.

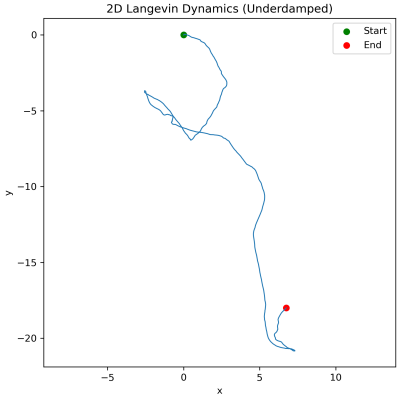


Figure 1: Underdamped motion trajectory.

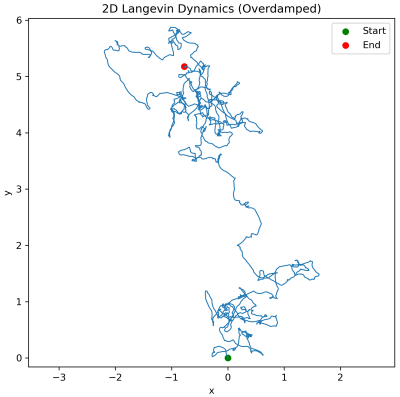


Figure 2: Overdamped motion trajectory.

Passive Motion and Brownian Dynamics

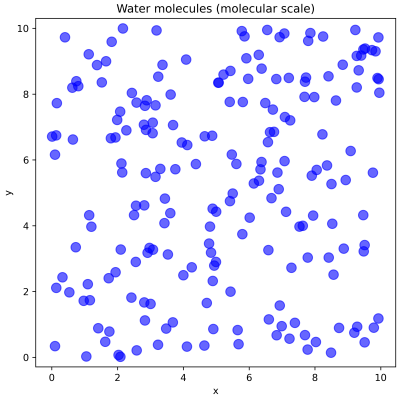


Figure 3: At smaller length scale.

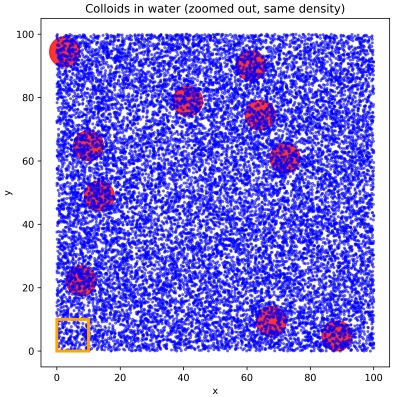


Figure 4: At larger length scale.

Passive Motion and Brownian Dynamics

Key idea

Brownian dynamics can be extended to the active matter system. Majority of the real life system falls in this category.

The basic idea can be summarized just in two numbers.

- **Péclet number:** Measures the ratio of the energy due to the active inertial forces to the random kicks in the system. It depends on the degree of propulsion (velocity) and mass of the particle. It is related to the Brownian motion.
- **Reynolds number:** It measures the ratio of the inertial forces to the forces due to viscosity. It is related to the Hydrodynamics.

Péclet Number (Pe)

Physical meaning

The **Péclet number** quantifies the competition between **directed motion** and **thermal diffusion**.

$$\mathrm{Pe} = \frac{vL}{D}$$

- v — characteristic self-propulsion or drift velocity
- L — characteristic length scale (e.g., particle size)
- D — translational diffusion coefficient

- $\mathrm{Pe} \ll 1$: motion dominated by Brownian diffusion
- $\mathrm{Pe} \gg 1$: motion dominated by active or driven propulsion

Reynolds Number (Re)

Physical meaning

The **Reynolds number** compares **inertial forces** to **viscous forces** in a fluid.

$$\mathrm{Re} = \frac{\rho v L}{\eta}$$

- ρ — fluid density
- v — characteristic velocity
- L — characteristic length scale
- η — dynamic viscosity

- $\mathrm{Re} \ll 1$: overdamped, Stokes flow (typical for colloids, bacteria).
- $\mathrm{Re} \gg 1$: inertia-dominated, turbulent or inertial flow, e.g., swimmers, airplanes, cyclists, go-kart dynamics.

Hands On: Performing Your First Brownian Dynamics (BD) Simulation

Patchy Particles

Description

Patchy particles are coarse-grained models in which a central core is decorated with a finite number of attractive sites (*patches*) that interact directionally rather than isotropically.

In this exercise, we construct a patchy particle system, define its anisotropic interactions, and simulate its Brownian dynamics in LAMMPS to explore how directional bonding drives self-assembly, chain formation, and aggregation.

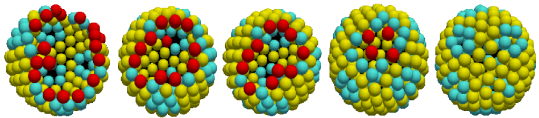


Figure 5: Patchy subunits are self-assembling to form closed shells.

Simulating Patchy Particle Model System

Objective

We will create a patchy particle system, define its interactions, and simulate its Brownian dynamics in LAMMPS. This exercise demonstrates how directional interactions (patches) affect the motion and self-assembly of particles. This relates, e.g., to protein systems or gel forming systems.

Let's Do!

LAMMPS Manual is always the best reference and the primary source of information regarding commands, variables, and their behavior.

Patchy particle self-assembly: Folder Location and Code Access

Source Files

Click on the link and use the folder named **`Hands_on/patchy_particle_model`**. Follow the instructions shown on the screen or provided in the supplied PDF file.

Patchy Particle Self-Assembly: Color and Notation Scheme

- All code snippets are displayed using the standard `monospace` font.
- **`Black text;`** denotes LAMMPS commands.
- **`Cyan text;`** denotes user-defined variables (e.g., strings, numbers, and other parameters chosen for clarity and ease of interpretation).
- **`Red text;`** denotes restricted or sensitive variables and should be modified with caution.

Patchy Particles Self-assembly Step 1: Creating the Simulation Box

Defining the Box

- Box dimensions: $30 \times 30 \times 30$ LJ units.
- Periodic boundary conditions in all directions.
- Atom style: hybrid (`sphere molecular`) to allow for both core and patch atoms.

```
region box block 0 30.0 0 30.0 0 30.0
create_box 2 box
```

Patchy Particles Self-assembly Step 2: Creating Atoms

Molecule Template

The molecule template (`patchy_molecule.mol`) defines a central core (type 1) surrounded by patches (type 2), forming a rigid, directional particle.

Patchy Particles self-assembly Extra step: Creating Molecules

- Create 1000 molecules randomly inside the box.
- Avoid overlap using `overlap` and `maxtry` options.

```
molecule patchy_part patchy_molecule.mol
create_atoms 0 random 1000 87910 NULL mol patchy_part 454756 overlap 1.5
....maxtry 50
```

Molecule Template (`patchy_molecule.mol`)

A single patchy particle is represented as a rigid molecule with one core atom and four peripheral patch atoms arranged tetrahedrally.

```
5 atoms
Coords
1 1.0 1.0 1.0 # must be specified
for all the atoms
...
Types
1 1 # must be specified for all
the atoms
...
```

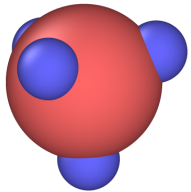


Figure 6: Patchy particles with tetrahedral arrangement.

Patchy Particles Self-assembly Extra step: Description

Atom Types and Functionality

- **Atom 1:** Central core (type 1), have finite mass, interacts via Lennard-Jones potential.
- **Atoms 2-5:** Peripheral patches (type 4), nearly massless, interact via directional cosine-squared potential.
- Tetrahedral geometry ensures patches create specific bonding orientations.
- In LAMMPS, the template is loaded using the `molecule` command, and multiple molecules are placed randomly in the simulation box.
- The given patchy particle design allows us to observe directional bonding and cluster formation in a Brownian dynamics simulation model using a relatively much simple yet powerful approach.

Patchy Particles Self-assembly Step 3: Defining Particles' attributes

Mass and Size

```
set type 1 mass 1.0
set type 2 mass 0.000001
set type 1 diameter 1.0
set type 2 diameter 0.0
```

Patchy Particles Self-assembly Step 4: Grouping particle together

Core, Patches and Molecule

```
group core type 1
group patch type 2
group rigid_molecule type 1 2
```

Patchy Particles Self-assembly Step 5: Defining Particle Interactions

Pair Potentials

- Core-core: Lennard-Jones (LJ) potential.
- Patch-patch: Cosine squared (Gaussian-like) potential for directional binding.
- Core-patch: no interaction (**none**) to prevent artificial bonding.

```
pair_coeff 1 1 lj/cut 0.01 1.3 2.0
pair_coeff 1 2 none
pair_coeff 2 2 cosine/squared 8 0.3 0.35
```

Functional Forms

Lennard-Jones (shifted at cutoff):

$$U_{\mathrm{LJ}}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] - 4\epsilon\left[\left(\frac{\sigma}{r_{\mathrm{cut}}}\right)^{12} - \left(\frac{\sigma}{r_{\mathrm{cut}}}\right)^{6}\right], \quad r < r_{\mathrm{cut}}$$

Cosine-squared (patch-patch):

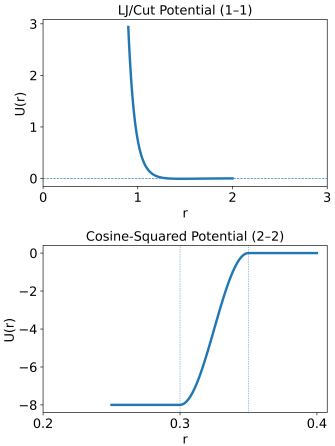
$$U_{\mathrm{CS}}(r) = \begin{cases} -\epsilon, & r < \sigma \\ -\epsilon\cos^2\left(\frac{\pi(r-\sigma)}{2(r_{\mathrm{cut}}-\sigma)}\right), & \sigma \le r < r_{\mathrm{cut}} \\ 0, & r \ge r_{\mathrm{cut}} \end{cases}$$

Patchy Particles Self-assembly Step 5: Defining Particle Interactions

Pair Potentials

- Core-core: Lennard-Jones (LJ) potential.
- Patch-patch: Cosine squared potential for directional binding.
- Core-patch: no interaction (**none**) to prevent artificial bonding.

```
pair_coeff 1 1 lj/cut 0.01 1.3 2.0
pair_coeff 1 2 none
pair_coeff 2 2 cosine/squared 8 0.3 0.35
```



Patchy Particles Self-assembly Step 6: Grouping Particles and Optimization

Groups

- `core`: central atoms
- `patch`: peripheral patch atoms
- `rigid_molecule`: all atoms in the molecule

Neighbor List Optimization

Exclude intra-molecular neighbors from force calculations to speed up simulation:

```
neigh_modify exclude molecule/intra rigid_molecule every 1 delay 0
...check no
```

Patchy Particles Self-assembly Step 7: Thermostat and Integration

Langevin Thermostat

Applied to core particles to simulate Brownian dynamics at constant temperature:

```
fix thermo_stat core langevin 1.0 1.0 0.1 428984 omega yes
```

Integration of Molecules

Use rigid-body integration for molecules:

```
fix rigid_thermo rigid_molecule rigid/small molecule
```

Patchy Particles Self-assembly Step 8: Data Collection and Thermodynamic Quantities

Dump Particle Positions

```
dump 1 all custom 10 simulation_data.lammpstrj id type x y z mol
```

Compute Temperature and Kinetic Energy (Core)

```
compute kinetic_core core ke
fix kinetic_output core ave/time 100 1 100 c_kinetic_core
...file kinetic.dat mode scalar
compute temp_core core temp/sphere
fix temp_output core
...ave/time 100 1 100 c_temp_core file temperature.dat mode scalar
```

Patchy Particles Self-assembly Step 9: Running the Simulation

Timestep and Runtime

```
timestep = 0.005
```

Terminal Thermo Output

```
thermo 100
thermo_style custom step temp ke pe press c_kinetic_core c_temp_core
```

Run Command

```
run 100000
```

Patchy Particles Self-assembly Step 10: Analysis of Results

Visualizing the System

- Load `simulation_data.lammpstrj` in OVITO or any visualization software.
- Observe motion of patchy particles and clustering due to directional interactions.

Patchy Particles Self-assembly: Thermodynamic Observables

- Check temperature and kinetic energy in logs.
- Copy paste the potential energy section in a .txt file and run the gunplot command to plot the result and see the kinetics and equilibration of the system.

Hands On: Beyond Brownian Dynamics; Capturing Motion at the Scale of Relaxation Time

From Brownian Dynamics to Hydrodynamics

Motivation

Brownian dynamics accurately captures overdamped motion at longer times (in compare to relaxation time of fluid), however, there are many dynamical and transport properties such as viscosity, which requires resolving momentum transport and velocity correlations over finite relaxation times. This is relevant for many products, such as cosmetics, paints, some food products, etc.

What Changes?

- Momentum is no longer relaxed instantaneously to thermal noise.
- Velocity gradients develop under external driving.
- Transport coefficients emerge from stress–velocity relations.

Viscosity through Non-Equilibrium Shearing

Objective

We compute the shear viscosity of a Lennard-Jones fluid using a non-equilibrium molecular dynamics (NEMD) approach.

Key Idea

- Impose a steady shear flow by moving confining walls.
- Measure the resulting velocity gradient.
- Extract viscosity from the momentum flux (pressure tensor).

Viscosity through Non-Equilibrium Shearing

Let's Do!

LAMMPS Manual is always the best reference and the primary source of information regarding commands, variables, and their behavior.

Viscosity: Color and Notation Scheme

- All code snippets are displayed using the standard `monospace` font.
- **`Black text;`** denotes LAMMPS commands.
- `Cyan text;` denotes user-defined variables (e.g., strings, numbers, and other parameters chosen for clarity and ease of interpretation).
- `Red text;` denotes restricted or sensitive variables and should be modified with caution.

Viscosity: Folder Location and Code Access

Source Files

Click on the link and use the folder named **`Hands_on/viscosity_through_shearing`**. Follow the instructions shown on the screen or provided in the supplied PDF file.

Viscosity Step 1: System Geometry and Confinement

Simulation Box

- A rectangular box extended along the shear-gradient direction.
- Periodic boundaries in all directions.
- Two solid walls define the shear geometry.

```
region box block -3 23 0 20 0 20
create_box 3 box
```

Viscosity Step 2: Defining Wall and Fluid Regions

Wall Construction

- Lower and upper regions act as rigid walls.
- Wall atoms are frozen during the simulation.

```
region lowersec block INF 0 INF INF INF INF
region uppersec block 20 INF INF INF INF INF
```

Viscosity Step 3: Filling up particles

Particle filling

- Filling randomly can lead to the overlap of particles and finally leading the LAMMPS to blow up.
- Easiest way to fill the particles is assign them on a lattice.
- LAMMPS while assigning particles does not care about the periodic boundary condtion.

```
lattice fcc 0.3 region uppersec
create_atoms 1 box
mass * 1.0
```

Viscosity Step 4: Defining Wall and Fluid Regions

Group Definition

- Upper and lower wall atoms are grouped based on geometric regions.
- Wall atoms are combined into a single group.
- Fluid atoms are defined as all remaining particles.

```
group upper region uppersec
group lower region lowersec
group wall union upper lower
group fluid subtract all wall
set group upper type 2
set group lower type 3
```

Viscosity Step 5: Interatomic Interactions

Lennard-Jones Fluid

- Standard LJ interactions are used for all atom pairs.
- Potential is shifted to ensure continuity at the cutoff.

```
pair_style lj/cut 3.0
pair_coeff * * 1.0 1.0 3.5
pair_modify shift yes
```

Viscosity Step 6: Assigning velocity

Random velocity assignment only to fluid layers

```
velocity fluid create $T 12345 mom yes rot yes dist gaussian
```

Viscosity Step 7: Imposing Shear Flow

Driving the System

- The upper wall is assigned a constant velocity.
- The lower wall remains stationary.
- This creates a velocity gradient in the fluid.

```
velocity upper set 0.0 $srate 0.0 sum no
fix frozenwall wall setforce 0.0 0.0 0.0
```

Viscosity Step 8: Computing temperature only along X-Z axis

Temperature

```
compute bias fluid temp/partial 1 0 1 velocity upper set 0.0 $srate 0.0 sum no

fix frozenwall wall setforce 0.0 0.0 0.0
```

Viscosity Step 9: Thermostatting Under Shear

Langevin Thermostat

- Applied only to the fluid atoms.
- Maintains temperature while allowing momentum transport.

```
fix thermostat fluid langevin 1.1 1.1 0.1 933888
fix integrator all nve
```

Streaming Velocity Removal and Running for Equilibration

```
fix_modify thermostat temp bias
```

Viscosity Step 10: Showing data and running for equilibration

Log file output

```
thermo 100
compute fluid_temp fluid /ramp vy 0 $srate x 1 19
thermo_style custom step temp press c_bias c_fluid_temp
dump 1 all custom 5000 simulation_data.lammpstrj id type x y z
run 1000000
```

Viscosity Step 11: Measuring Velocity Profiles

Velocity Gradient

- Fluid is binned along the shear-gradient direction.
- Average velocity in each bin is computed.

```
compute chunk_1 fluid chunk/atom bin/1d x lower 0.05 units reduced
compute chunkvel fluid vcm/chunk c_chunk_1
fix 1 all ave/time 100 10 1000 c_chunkvel[*] file velocity_profile.txt
```

Viscosity Step 12: Measuring Shear Stress

Momentum Flux

- The off-diagonal pressure tensor component P_{xy} represents momentum transport.
- Time averaging improves statistical accuracy.

```
variable pxy equal pxy
fix pxy_ave all ave/time 100 10 1000 v_pxy file pxy.dat
```

Viscosity Step 13: Running for Production

Production Run

```
run 1000
```

Viscosity: Constitutive Relation

Newtonian Shear Viscosity

For a Newtonian fluid, the shear stress is linearly related to the velocity gradient:

$$\sigma_{xy} = \eta \frac{\partial v_y}{\partial x}$$

Relation to the Pressure Tensor

In molecular dynamics, LAMMPS outputs the *pressure tensor*:

$$P_{xy} = -\sigma_{xy}$$

Combining both relations gives

$$\eta = -\frac{\langle P_{xy} \rangle}{\partial v_y / \partial x}$$

where $\langle \cdot \rangle$ denotes a time and ensemble average.

Viscosity: Microscopic Origin of P_{xy}

Pressure Tensor in LAMMPS

LAMMPS computes the off-diagonal pressure component as

$$P_{xy} = \frac{1}{V}\left(\sum_i m_i v_{ix} v_{iy} + \sum_{i<j} r_{ij,x} f_{ij,y}\right)$$

Practical Implementation

- The first term is the kinetic contribution.
- The second term is the configurational (virial) contribution.
- $\langle P_{xy} \rangle$ is obtained by time-averaging `pxy.dat`.
- The velocity gradient $\partial v_y / \partial x$ is extracted from the velocity profile.

Viscosity Step 14: Analysis and Visualization

Post-Processing

- Run the provided Python script to compute viscosity.
- Inspect velocity profiles and linear fits.
- Verify steady-state behavior of P_{xy}.

Physical Interpretation

This approach explicitly connects microscopic momentum transfer to macroscopic viscosity through non-equilibrium dynamics.

Continuum Models (Voluntary Tasks) and Conclusion

Continuum Model: Fokker-Planck Equation; An example

Description

The Fokker-Planck equation describes the time evolution of the probability density function of a particle's position and velocity under stochastic forces, such as Brownian motion. It provides a continuum description of the dynamics, complementing discrete particle simulations like Langevin dynamics.

$$\frac{\partial P(\mathbf{r},t)}{\partial t} = -\nabla \cdot (\mathbf{v}P) + D\nabla^2 P$$

where $P(\mathbf{r},t)$ is the probability density, $\mathbf{v}$ is the drift velocity, and D is the diffusion coefficient.

Fokker-Planck Equation: Continuum Model

Objective

Observe the evolution of the density distribution in the continuum model using the Fokker-Planck equation.

1. Navigate to the folder **`langevin_dynamics_vs_fokker_planck_equation`**.
2. Run the Python script `evolution_focker_planck.py`.
3. Open the generated GIF file and observe the evolution of the probability distribution of particles under Brownian dynamics as predicted by the Fokker-Planck formalism.

Langevin Dynamics: Discrete Particle Model

Objective

Compare the discrete particle dynamics from Langevin simulations with the continuum Fokker-Planck model.

1. Run your LAMMPS input file `langevin.in`.
2. Load the output file `simulation_data.lammpstrj` in OVITO.
3. Apply `Add Modification` → `Histogram` and select `position_X`.
4. Set the histogram range between `0-20` for the x-axis.
5. Observe the histogram evolution and compare it with the Fokker-Planck probability distribution.

Conclusion

Key Takeaways. You

- successfully set up a patchy particle system in LAMMPS to study self-assembly,
- computed thermodynamic quantities using Brownian dynamics simulations,
- studied viscosity in overdamped systems and non-equilibrium shearing,
- visualized and compared discrete particle dynamics with continuum Fokker-Planck predictions.

Thank you!

Learn by Example

Congratulations you have been armed to tackle the real world problems!

——— May the force be with you ! ———

Learn by Example

Welcome to Questions and Discussion

.... ????????

Feel free to ask, or send me an email at address vavarm@utu.fi or vikkivarma16@gmail.com