Exercise_1 Molecular Dynamics Simulation

Andrea Scaioli

May 2025

1 Introduction

Molecular dynamics (MD) simulations are a powerful method to study the evolution of molecular systems by numerically integrating Newton's equations of motion. Despite being approximate, they allow the investigation of complex systems with many degrees of freedom, such as proteins or materials, that are otherwise intractable with analytical approaches.

A critical aspect in MD simulations is the choice of the time step: it must be small enough to accurately capture the fastest motions (e.g., bond vibrations), but large enough to reach meaningful simulation times with reasonable computational cost. If the time step is too large, the numerical integration becomes unstable and the total energy of the system may diverge, leading to unphysical results. Optimal time step tuning is therefore essential to achieve both energy conservation and simulation efficiency.

One of the most widely used MD engines is `GROMACS`, an open-source software optimized for high-performance simulations of biomolecules, polymers, and other systems. It offers robust tools for trajectory analysis and energy evaluation, making it widely adopted in biophysics and materials science.

2 Methods

2.1 Gromacs

To perform a molecular dynamics (MD) simulation using `GROMACS`, three key input files are required: a topology file (`.top`), a coordinate file (`.gro`), and a parameter file (`.mdp`).

- `.top` (topology file): This file contains the definitions of the molecular system, including information about atoms, bonds, angles, dihedrals, and force field parameters. It specifies the interactions between particles and provides the necessary data for force calculations during the simulation.
- `.gro` (coordinate file): This file holds the atomic coordinates of the system at a specific time, along with box dimensions. It is used to initialize the simulation by defining the initial positions of the molecules or atoms.
- `.mdp` (molecular dynamics parameter file): This file contains the simulation parameters, such as temperature, pressure, time step, and other physical parameters governing the dynamics. It defines the simulation's conditions and control settings.

Once these files are prepared, the simulation is set up using the `grompp` command:

```
grompp -f input.mdp -c input.gro -p input.top -o output.tpr
```

This command processes the input files and generates a binary `.tpr` file, which contains all the information needed to run the simulation.

Next, the simulation is executed using the `mdrun` command:

```
mdrun -v -deffnm output
```

This command runs the MD simulation and generates output files, including the trajectory file (`output.xtc`), which contains the atomic positions at each time step of the simulation.

2.2 Simulations

In this work, I performed 12 molecular dynamics simulations across 3 different systems to investigate the influence of the time step on energy conservation and numerical stability. Each system consists of 1000 atomic dimers, where the atomic masses are respectively 2, 4, 6, and 12 atomic units. The aim was to identify, for each system, the critical time step that balances simulation stability with computational efficiency.

For every system, I determined three characteristic time steps: the smallest one that ensures energy conservation throughout the simulation, a middle one where the system begins to exhibit instability and energy drift, and a larger one at which the energy diverges rapidly, causing the simulation to fail. This allowed me to systematically evaluate how the maximum stable time step depends on particle mass and system dynamics.

To ensure meaningful results, each simulation was preceded by an equilibration phase, as the initial configurations were randomly generated and far from equilibrium. After equilibration, production runs were carried out at varying time steps to analyze their effects on total energy and overall stability. This approach highlights the importance of careful time step selection in achieving long, stable simulations at minimal computational cost.

As can be seen in the Appendix, to automate the simulations, I used a shell script that performs all necessary steps: energy minimization, equilibration (short NVT simulation), and energy extraction. For each combination of mass and time step, the script generates a tailored `.mdp` file with the correct time step, prepares the input with `grompp`, runs the simulation with `mdrun`, and extracts the relevant energy terms.

However, it is important to note that when the time step is too large and the system becomes unstable, the total energy quickly diverges. In these cases, GROMACS halts the simulation automatically due to numerical instability. To properly classify such simulations (especially at or near the divergence threshold), I monitored the runs manually.

3 Results

As expected, the results show a clear distinction between the three selected time steps for each system. In the smallest time step, total energy is well conserved throughout the simulation. At the intermediate time step, the energy begins to drift, indicating the onset of numerical instability. Finally, at the largest time step, the energy rapidly diverges, leading to a breakdown of the simulation. This confirms the importance of carefully tuning the time step to ensure both stability and efficiency. The behavior is consistently observed across all systems analyzed.

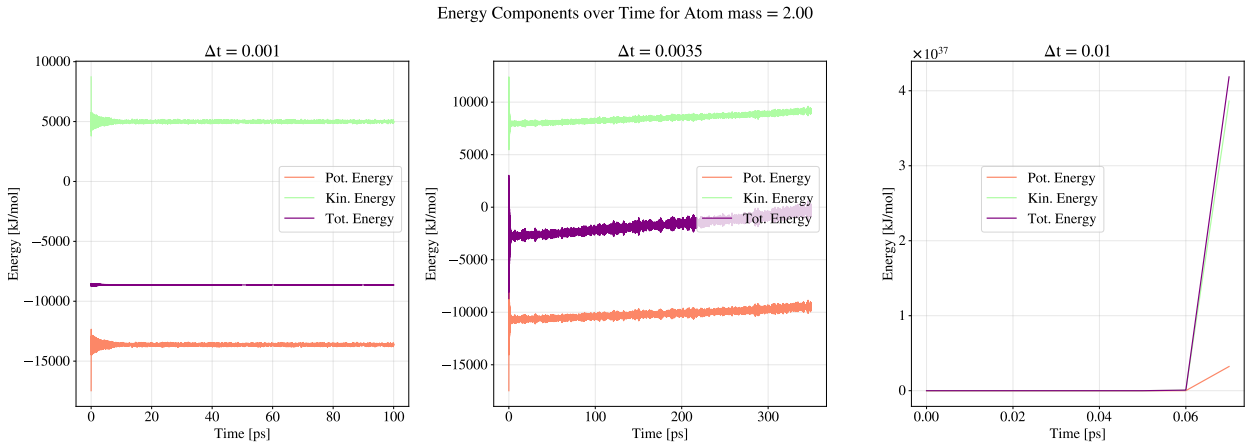

Figure 1: Total energy over time for the system with mass 2. The plot shows simulations at three time steps: stable (conserved energy), marginal (energy drift), and unstable (energy explosion).

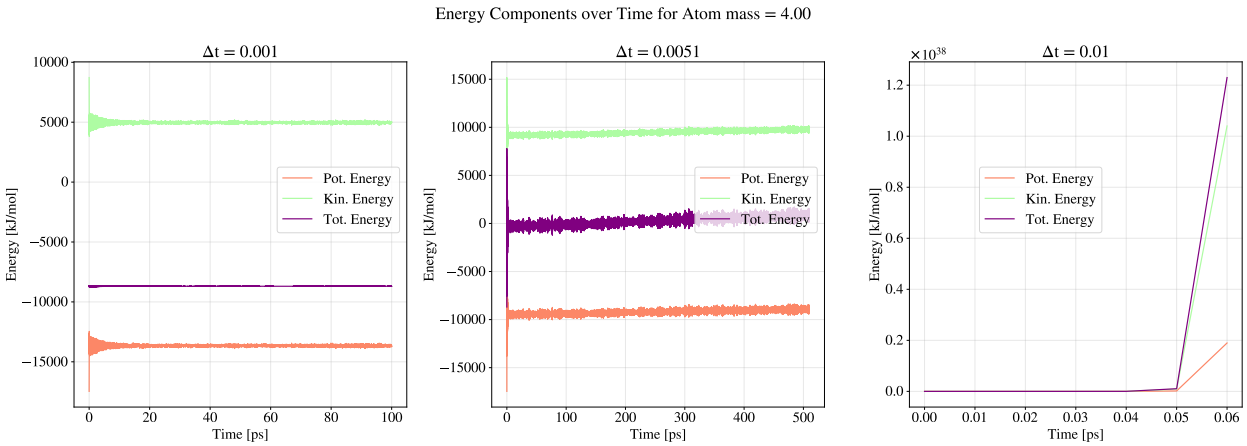

Figure 2: Total energy over time for the system with mass 4. The plot shows simulations at three time steps: stable (conserved energy), marginal (energy drift), and unstable (energy explosion).

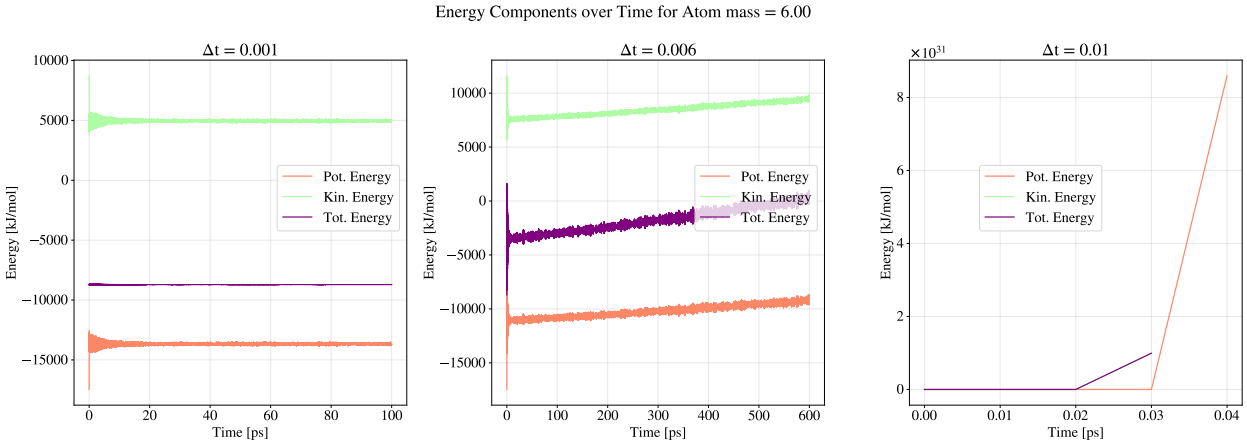

Figure 3: Total energy over time for the system with mass 6. The plot shows simulations at three time steps: stable (conserved energy), marginal (energy drift), and unstable (energy explosion).

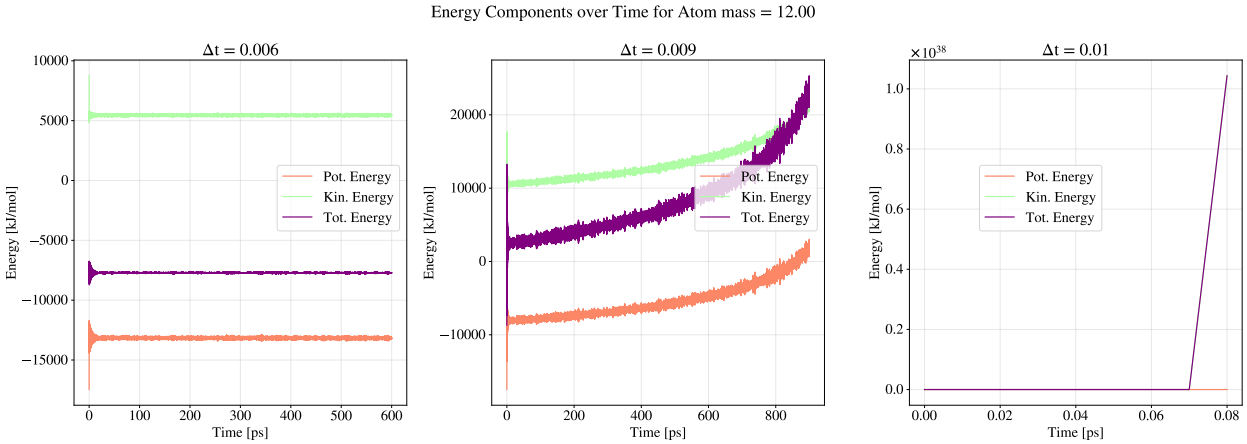

Figure 4: Total energy over time for the system with mass 12. The plot shows simulations at three time steps: stable (conserved energy), marginal (energy drift), and unstable (energy explosion).

4 Conclusion

Based on the simulations, the optimal time step for each system was determined as follows: 0.003 ps for the system with atoms of mass 2, 0.004 ps for the system with atoms of mass 4, 0.005 ps for the system with atoms of mass 6, and 0.008 ps for the system with atoms of mass 12. These values were selected to ensure energy conservation while maintaining simulation stability.

It is noteworthy that there is a clear correlation between the mass of the atoms and the optimal time step. Heavier systems, as expected, require larger time steps due to their slower dynamics. With increased mass, the atoms move more slowly, and their positions and velocities change less within a given time step. This results in the possibility of using larger time steps without compromising the accuracy or stability of the simulation.

This observation highlights the importance of considering the system's properties, such as atomic mass, when selecting an appropriate time step to balance computational efficiency and physical accuracy in molecular dynamics simulations.

5 Appendix

Simulation script:

```
#!/bin/bash

# Exit on error
set -e

# --- Input arguments ---
TOPO_NUM="$1"      # e.g., 2 → topol_2.top
STEP_NUM="$2"      # e.g., 1 → intro_0.001.mdp

# --- Usage check ---
if [[ -z "$TOPO_NUM" || -z "$STEP_NUM" ]]; then
  echo "Usage: $0 <TOPO_NUM> <STEP_NUM>"
  echo "Example: $0 2 0.001"
  exit 1
fi

# Summary of the simulation before running:
echo "Running simulation with:"
echo "- Topology: topol_${TOPO_NUM}.top"
echo "- Timestep: ${STEP_NUM} ps"

# Create intro_STEP_NUM file:
cp intro.mdp intro_${STEP_NUM}.mdp
sed -i "s/^\([[:space:]]*dt[[:space:]]*=[[:space:]]*\)[^[:space:]]*/\1${STEP_NUM}/" intro_${STEP_NUM}.m

# Step 1: Energy minimization
gmx grompp -f em.mdp -p topol_${TOPO_NUM}.top -c intro.gro -o em_${TOPO_NUM}.tpr
gmx mdrun -v -deffnm em_${TOPO_NUM}

# Step 2: NVT/short simulation
gmx grompp -f intro_${STEP_NUM}.mdp -p topol_${TOPO_NUM}.top -c em_${TOPO_NUM}.gro -o intro_${STEP_NUM}
gmx mdrun -v -deffnm intro_${STEP_NUM}_${TOPO_NUM}

# Step 3: Energy extraction (example: kinetic, potential, total)
```

```
printf "4\n5\n6\n" | gmx energy -f intro_${STEP_NUM}_${TOPO_NUM}.edr -o intro_${STEP_NUM}_${TOPO_NUM}.x

# Step 4: Visualize the results
xmgrace -nxy intro_${STEP_NUM}_${TOPO_NUM}.xvg
```