Lab assignment #6: Solution

Nico Grisouard (Instructor), Mohamed Shaaban (TA & marker),
Mikhail Schee (TA)

Due Friday, 23 October 2020, 5 pm

Lab Questions

1. **Modeling space garbage (see Newman Exercise 8.8; 30% of the lab mark)**

   See `L06-407-2020-Q1.py` for the code. I show the plot on fig. 1.

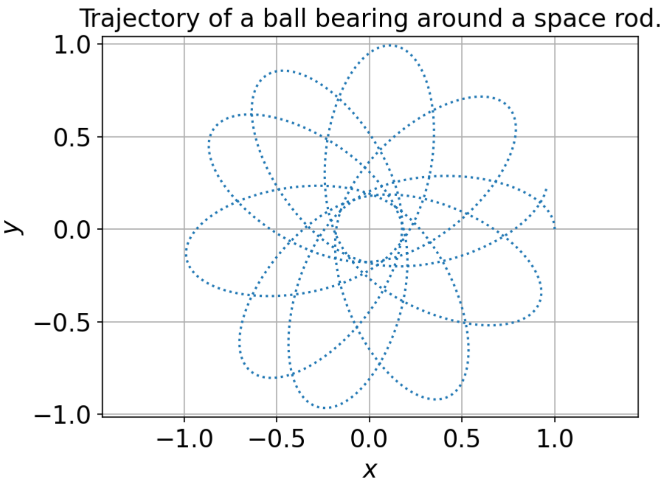

Figure 1: Trajectory of the space garbage.

2. **Molecular dynamics simulation, Part 1 (30% of the lab mark):**

   See `L06-407-2020-Q23.py` for a script that will work for all questions. Set `case = 'i'`, `'ii'` or `'iii'` for Q2b. Set `case = 'iv'` for Q3a and b, and `case = 'v'` for Q3c.

   (a) (not requested) If the potential is $V(x, y)$, the corresponding acceleration felt by molecule 1 due to molecule 2 is

$$\vec{a}_{2\to 1} = \frac{1}{m}\vec{F} = -\frac{1}{m}\vec{\nabla}V = \frac{4\epsilon}{m}\left(\frac{12\sigma^{12}}{r^{14}} - \frac{6\sigma^6}{r^8}\right)\begin{bmatrix} x_1 - x_2 \\ y_1 - y_2 \end{bmatrix}, \tag{1}$$

with $r = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. And for the acceleration felt by 1 due to 2, you just have to flip the 1 and 2 indices.

Sign check: if 2 is at $(x_2, y_2) = (0,0)$, and 1 is at $(x_1, y_1) = (X, 0)$, with X sufficiently small that $12\sigma^{12}/r^{14} - 6\sigma^6/r^8 \approx 12\sigma^{12}/r^{14} > 0$, then $a_y = 0$ and $a_x > 0$. It works.

Later on, when coding, we will use $\sigma = 1$, $\epsilon = 1$ and $m = 1$.

(b) "Pseudocode" for a program that updates the position of the 2 molecules using the Verlet method:

- Define `dt=0.01` and `Nt=100`,
- Define a switch to investigate either cases i, ii or iii.
- `IF` statements to initialize the two positions in one numpy array `rs`,
- initialize 2 empty numpy arrays for positions and velocities, of shapes `(2, Nt, 2)` (2 coordinates, x and y, `Nt` time steps, 2 molecules.)
- do a RK2 to get the first velocity (at 1/2 time step, i.e, after the initial velocity) and second position.
- fill in the first time step of position array with initial positions.
- define the right-hand-side for velocity (acceleration):
  - input is the 2×2 position array at the current time step $((\vec{r}_0(t), \vec{r}_1(t))$,
  - LOOP over molecules: compute the vector from molecule n to molecule $1 - n$, then the distance r, then the acceleration;
  - return the acceleration.
- LOOP a Verlet over `Nt` time steps (no $v(t+h)$ at this point):
  - iterate the positions $\vec{r}_i(t+h)$ using the previous velocities $\vec{v}_i(t + h/2)$, $i = 0$ or 1 the molecule index.
  - compute $k = dt\,\vec{a}_i(\vec{x}_i(t))$,
  - iterate $\vec{v}_i(t + 3h/2) = \vec{v}_i(t + h/2) + k$.
- plot $x_1(t)$, $x_2(t)$ (actually not requested in question),
- plot $y_1(x_1)$, $y_2(x_2)$.

In case (i) (see fig. 2), the molecules oscillate, they are trapped in each other's potential well.

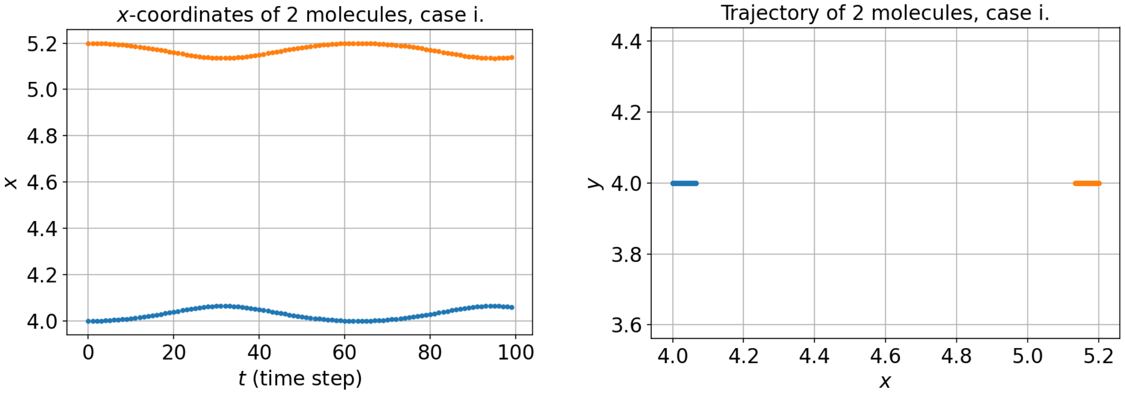

Figure 2: For Q2b, Case i. Left: x coordinates over time *(not requested)*; right: trajectories.

In case (ii) (see fig. 3), the molecules start too close from each other, they are ejected from each other's well and go ballistic.

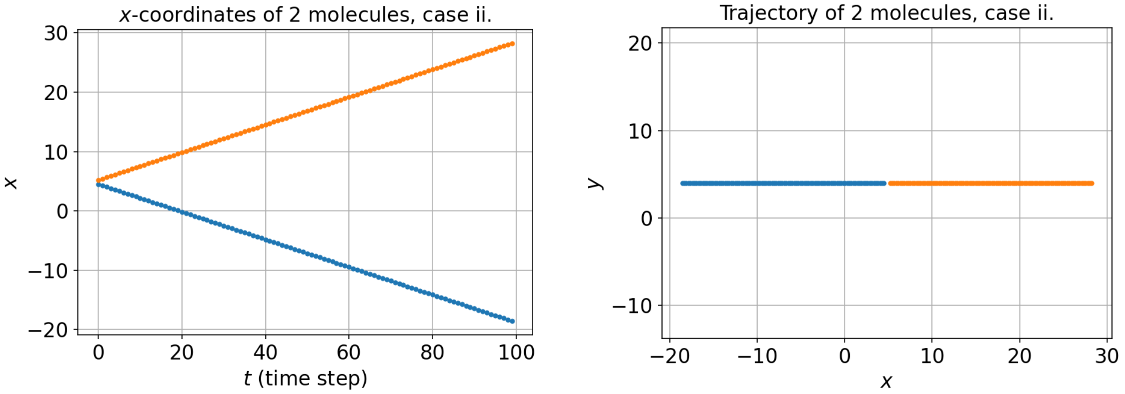

Figure 3: For Q2b, Case ii. Left: x coordinates over time *(not requested)*; right: trajectories.

In case (iii) (see fig. 3), the molecules start far apart from each other that they are attracted to each other.

(c) If we make a plot of the Lennard-Jones potential, there is a minimum around (but not exactly) $r = 1.2$ for $\sigma = 1$, which is the separation of the particles for this case. They start with zero velocity (no initial KE), and because they are not exactly at the bottom of each other's wells, they start with a little PE. PE converts to KE, then to PE again, and the cycle of conversion creates oscillations.

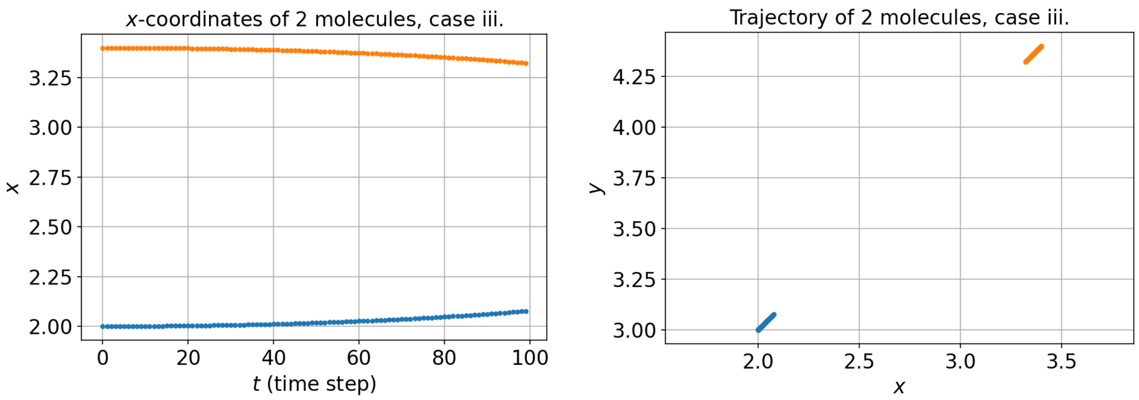

Figure 4: For Q2b, Case iii. Left: x coordinates over time *(not requested)*; right: trajectories.

3. **Molecular dynamics simulation, Part 2 (40% of the lab mark):**

   Reminder: in `lab06-407-2020-Q23.py`, set `case = 'iv'` for Q3a and b, and `case = 'v'` for Q3c .

   (a) "Pseudocode", updated from the previous question (changes highlighted):

   - Define `dt=0.01` and `Nt=1000`,
   - `IF` statements to initialize the 16 positions in one numpy array `rs`,
   - initialize 2 empty numpy arrays for positions and velocities, of shapes `(2, Nt, 16)` (2 coordinates, x and y, `Nt` time steps, 16 molecules.)
   - fill in the first time step of position array with initial positions.
   - do a RK2 to get the first velocity (at 1/2 time step, i.e, after the initial velocity) and second position.
   - define the right-hand-side for velocity (acceleration):
     - input is the 2× 16 position array at the current time step ($(\vec{r}_0(t), \dots \vec{r}_{15}(t)$),
     - LOOP over molecules;
       - ∗ Initialize acceleration as 2 × 16 numpy array of zeros,
       - ∗ within a LOOP over *all other molecules,*
       - ∗ compute the vector from molecule n to every other molecule i ,
       - ∗ then the distance r_{ni},
       - ∗ then add the result to the acceleration;

  - return the acceleration.

- LOOP a Verlet over `Nt` time steps (with $v(t+h)$ this time!):
- and the rest is the same from Q2b.

Trajectories are plotted on fig. 5. We can see that the particles initially start with too much packing: the particles split into 4 clusters of 4 particles each. Each cluster has a ballistic trajectory away from the others, and each particle within a cluster is in the collective potential well of the cluster.

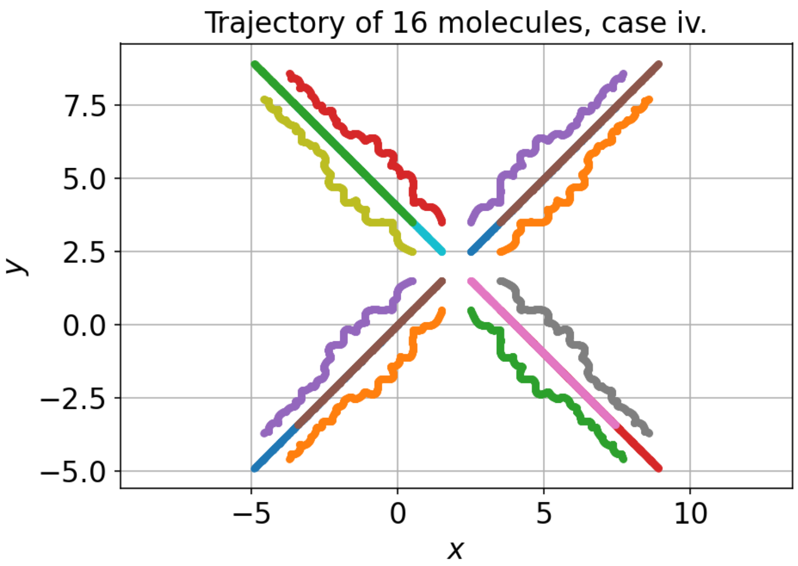

Figure 5: For Q3a. Trajectories of 16 particles in an infinite domain.

(b) To compute the energy, we need to add the intermediate velocity in the Verlet algorithm. I did not store it in a separate array, but instead defined length-`Np` arrays for PE and KE. The KE is updated in the time loop with the intermediate velocity, and is computed directly as the sum of all the individual KEs. The PE is also updated in the time loop, and is computed as $\sum_i V_i(r_i)/2$, where $V_i(r_i)$ is the potential energy of a couple of molecules (hence the division by two to keep the count "per molecule").

I show the plot of the energies in fig. 6. We see that initially, the system packs a lot of potential energy, some of which being permanently converted into KE: these are the clusters, moving away from each other in a ballistic fashion. There are still a lot of fluctuations: there are the molecules in each cluster moving with respect to each others.

The total energy looks very constant, but fig 6 is not precise enough to quantify

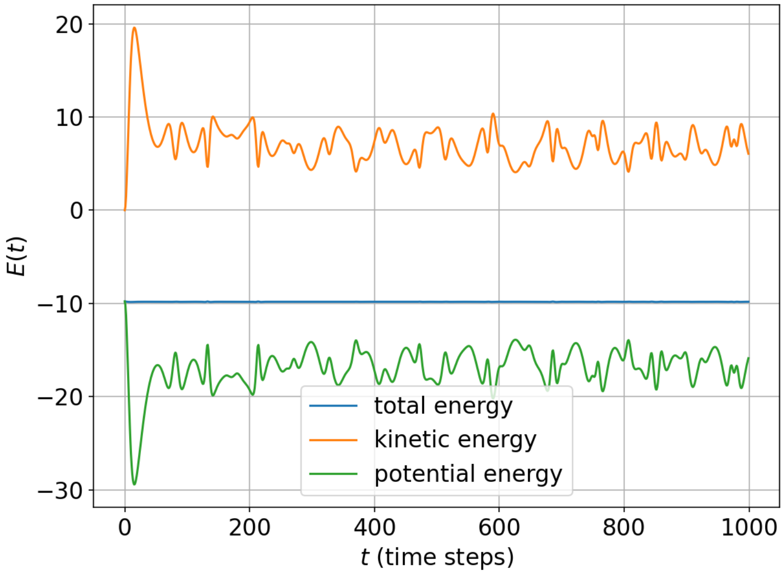

Figure 6: For Qb.

how much. I therefore computed the ratio $[\max(E(t)) - \min(E(t))]/\text{mean}(E(t))$, which is ≈ -0.00603, i.e., 0.6%.

(c) I show plots of the trajectories in fig. 7. We can see that for about half of the duration, the particles don't seem to move too much. This is because even though the same packing as before would have the molecules move, the symmetry of the arrangement means that the forces cancel two-by-two. Eventually, perhaps due to accumulating round-off or error method, or perhaps due to our domain being not strictly periodic, the molecules start to move around, but stay organized in a lattice.

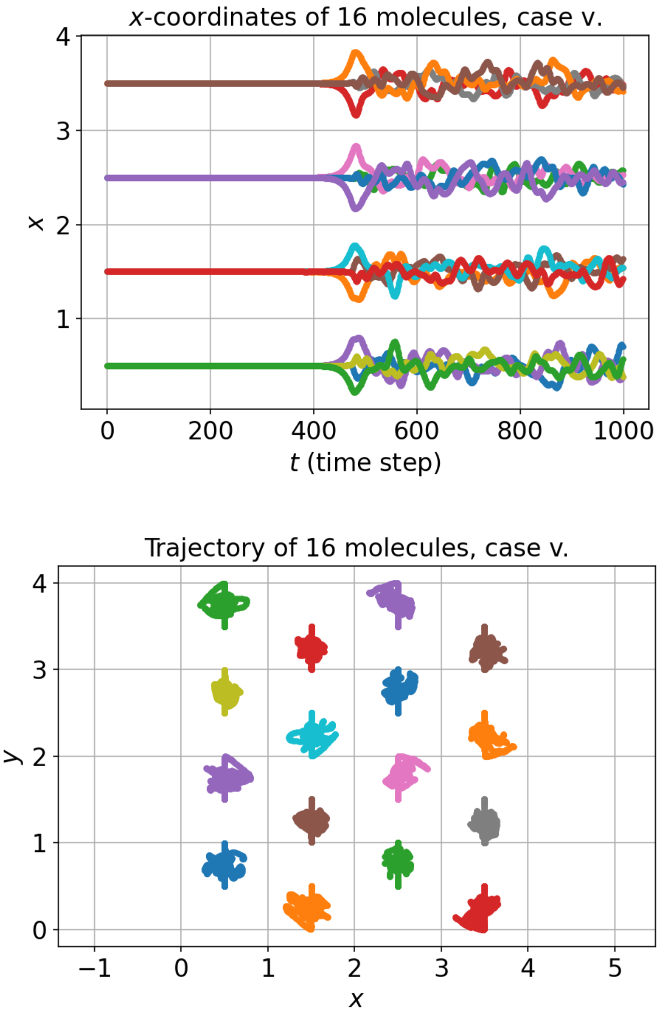

Figure 7: For Q3c. Left: x coordinates over time *(not requested)*; right: trajectories.