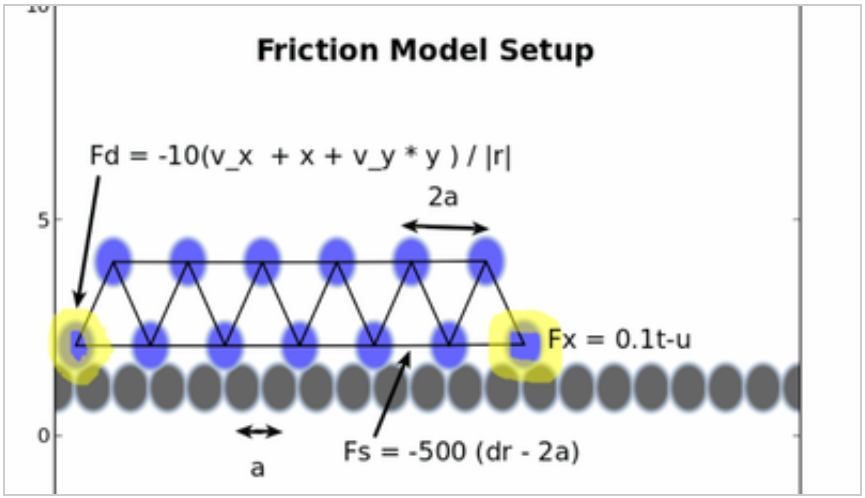
Project I: Molecular Friction

From CS 477/577 Computer Simulation & Modeling

See Moodle for source and references materials for modeling the molecular basis for friction. Hints will appear here.

This project will be due **Monday, March 26, 10:00 AM**. The format that you use should address every evaluation criteria discussed in the rubric for projects shown in the Syllabus. You will have to use Latex for writing the document.

Contents



Initial conditions and forces in the friction project. Note that Fx is scalar (in the x direction only), but Fd and Fs are **vectors**. Also know that Lennard-Jones type forces exist between all other particles, but the floor being fixed in place prevents its motion.

Overview of Modifications

There are several modifications that must
be made to a working code base developed for the molecular dynamics problems. We will spend the week detailing them, and discussing strategies for implementing them. In summary, the modifications will consist of:

- The change from an autonomous to a non-autonomous system, or one that depends explicitly on t.
- The addition of spring forces between the 13 atoms that create the 'solid'.
- The damping on the lower left atom.
- The pulling of the lower right atom.
- The elimination of forces between the atoms that create the floor.
- A new state variable that tracks the force required to pull the solid.

For the most part, I was able to accomplish these goals in a new **Force** class that I started from the Lennard Jones. However, some wider changes were required to make the system non-autonomous. My code for initialization became complex, generating both initial conditions, and data structures used in the force calculation. Below, I detail each in an effort to help you.

Computing forces

I found it useful to have index markers for 'special' atoms in this simulation. So, the container class now has an index to tell me which atom is being dragged, another to tell me which atom has damping, and a third to for the last "floor" atom's index. In this way my approach to the determining accelerations is

1. Do a full Lennard-Jones calculation.
2. Add the damping on the lower left atom.
3. Add the dragging forces on the lower right atom.
4. Add the spring forces between the atoms in the solid.
5. Zero out any accelerations that have been computed between atoms in the floor.

As you will see below, the forces between springs will require some special consideration.

Non-autonomous system

The right-hand side of the functions being integrated;

$$\frac{dx}{dt} = f(x, t)$$

is now *explicitly dependent* on t, whereas in the past it was dependant on x only.

The dependence arises from the spring being displaced from left to right according to the formula

$$F_p(t) = k(0.1t - u)$$

which is a horizontal force on the spring. u in this equation is the displacement from the initial position of the lower right atom, and the current position of the dragger is parameterized by $x(x) = 0.1t$. Note that the assignment suggests k = 1.

Changes to the code may (depending on your implementation) consist of those to the Force itself, as well as the driver that calls force, and the integration routine, which now accepts both the state x and the time t.

Initial conditions

One of the first things that you should write is the code to generate the floor, the solid, and the data structures (indices and adjacency matrix) needed for efficient calculation of the forces. There are several opportunities for mistakes, so I recommend spending some time simply getting the positions right, before worrying at all about the forces. Also spend some time making sure that your data model allows the passage of the appropriate data (indices and adjacency matrix) from the initial conditions to the force calculation. You should be able to test the code by plotting the initial positions of the atoms and lines connecting them according to the entries in the adjacency matrix.

Spring forces

The forces of the springs between the atoms in the solid are stiff springs. While it isn't very elegant, an easy way to do this calculation is to do the stiff string force between *all* particles, and then use some matrix manipulation to eliminate the forces that should not be there. To do this, you'll need a matrix that is True wherever there are no springs linking atoms. Having that in hand, you can write code like this:

```
# Determine the magnitude of the force that is due to the spring that connect the atom
# dr is returned as part of the distance matrix calculation.
# c.spring_eq_dist is the initial distance between atoms in the solid, 2a, or 2*2**(1/
# c.no_springs is a boolean adjacency matrix that is True unless a spring connects its

magnitude_s = -self.__k * (dr - p.spring_eq_dist)
magnitude_s[p.no_springs] = 0.0
```

for spring forces. You will have to add this to the Lennard-Jones magnitude before projecting the magnitude onto the unit vector between atoms, just as you did for the previous problems.

Damping

As specified in the problem, the damping force is

$$\mathbf{F_d} = -10(v_x\hat{x} + v_y\hat{y})$$

the required data should all be on hand in the force method (velocities really). This leaves you to add the damping to the correct particle, which has an index defined in the initialization.

Student Answers

- Heracles: Project_I:_Molecular_Friction
- Surt: Project I: Molecular Friction
- Medea: Project I: Molecular Friction
- Huginn: Project I: Molecular Friction
- Clytemnestra: Project I: Molecular Friction