APiE Exercise – Molecular Dynamics (MD) for Solids 1D

Exercise 1 (3pt)

Goal is to set up a linear 1D (one-dimensional) chain. First, generalize your linear spring ODE-program to 2 "particles" or atoms connected by one spring (see APiE-script).

Recommendation: Preparation for future programming in 2D:
Using the linear spring model, implement in your solver the interaction force using the normal vector $\hat{n} = (\vec{x}_i - \vec{x}_j)/|\vec{x}_i - \vec{x}_j|$, and the departure from the equilibrium position length $\delta = |\vec{x}_i - \vec{x}_j| - x_e$. Take care that the sign is correct.

Implement the force calculation in a function that receives the two particles and returns the force (scalar in 1D, vector in 2D). Then establish for each particle a loop over all particles it has a spring-connection with (this will be relevant below for the linear chain and later for 2D) and sum up all forces acting on a particle. For each particle pairi (i,j), the forces acting on i by j and reverse are related by $f_{i \leftarrow j} = -f_{j \leftarrow i}$.

Note: Make sure that you program modular. Separate variable definition, input, output, force-calculation and integration clearly as different modules – or functions.

Display the motion of the pair of particles for some time and also display the total energy and the kinetic and potential fractions.

Exercise 2 (4pt)

Generalize the program to N particles and implement:

(a) a linear chain with 11 particles, see Fig. 1

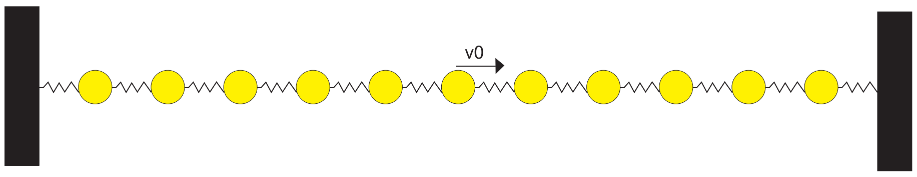

Figure 1: Linear Chain

The first and the last particle are connected to a fixed wall. The distance between the particles is x_e and this is equal to the equilibrium length of the springs. The central particle gets an initial velocity v_0, all the other particles have initial velocity equal to zero.

Display the motion of the particles (in a graph).

(b) Write a function for the force-calculation using the method from above, such that the force calculation appears only once per particle-pair in the program. For this implement a loop over particle pairs.

Exercise 3 (3pt)

Visualize the movement of the particles in a movie. Let the color of each particle give a measure of the speed of the particle.

Exercise 4 (voluntary fast – 2D is subject of the exercise MDSolids2D)

Generalize the pair of particles to 2D (voluntary extra – 2 extra points: 3D). a square-system with N=10x10 particles, see Fig. 2

Assign to the top particle on the right side an initial vertical velocity v_0. All the other particles have initial velocity equal to zero, while their pair-wise separations are all equal to x_e.

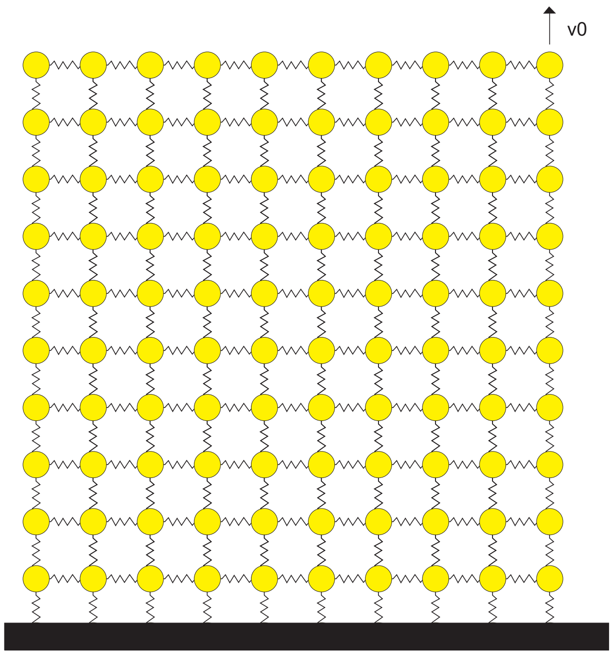

Figure 2: Square Lattice

Hints:
Make sure that you split the force-calculation and the integration (Verlet or Runge-Kutta) loops. Make sure that you set forces to zero at each new time.

Here an example algorithm (but better also see the script):

```
Loop 1: integration loop over time ($t_i = t_{i-1} + dt$)

  • reset ALL forces ($f_x[.] = 0$)
  • Loop 2: over all particles (i<$N_{max}$)
      – Loop 3: over all contact partners (j<i)
          * distance    dist=sqrt((x[i]-x[j])*(x[i]-x[j]))
          * normal      $n_x$=(x[i]-x[j])/dist
          * overlap     delta=rad[i]+rad[j]-(x[i]-x[j])*$n_x$
          * contact: if delta>0
              · relative velocity ($v_{rel}$=-($v_x$[i]-$v_x$[j])*$n_x$)
              · interaction force ($f_x$[i]=(k*delta+v*$v_{rel}$)*$n_x$)
              · partner interaction ($f_x$[j]=-(k*delta+v*$v_{rel}$)*$n_x$)
      – end Loop 3
  • temporary store position (xtmp=x[i])
  • integrate (x[i]=2*xtmp-$s_x$[i]+$f_x$[i]/m[i]*dt*dt)
  • save old position ($s_x$[i]=xtmp)
  • end Loop 2
  • increase time (t=t+dt)

end Loop 1

end program
```