Foundations of Multiscale Modelling: Kinetics. Homework 5. Green-Kubo relations and Aggregation-Fragmentation Kinetics. Due 23:59 10 March.

March 1, 2022

1. Green-Kubo relations* (2 points)

Compute (i) the non-diagonal part of the stress tensor σ_{xz} and (ii) the shear viscosity coefficient η using the Green-Kubo relations. You can use the LAMMPS script for liquid Ar (from HW3). Calculate the stress auto-correlation function $\langle \sigma_{xz}(0)\sigma_{xz}(t)\rangle$, where $\langle \dots \rangle$ is the time average. The viscosity can be calculated as

$$\eta = \frac{1}{k_{\mathrm{B}}TV}\int_0^\infty \langle \sigma_{xz}(0)\sigma_{xz}(t)\rangle \, dt.$$

where the stress tensor can be calculated by

$$\sigma_{\alpha\beta}(t) = -\sum_{k=1}^{N} m_k(v_k^\alpha - u^\alpha)(v_k^\beta - u^\beta) - \frac{1}{2}\sum_{k=1}^{N}\sum_{j\neq k} r_{kj}^\alpha F_{kj}^\beta, \qquad \alpha,\beta = x,y,z.$$

where m_k is the mass and v_k^α is the velocity component of kth atom, u^α is the average velocity, $r_{jk}^\alpha = r_j^\alpha - r_k^\alpha$, r_k^α is the position of the kth atom, and F_{kj}^β is the component of the force applied on atom k by atom j.

Hint: You can calculate and save the components σ_{xy}, σ_{xz} and σ_{yz} of stress tensor summed over atoms at different time steps via commands:

```
compute 1 all stress/atom NULL
compute 2 all reduce sum c_1[4] c_1[5] c_1[6]
fix...print..."$(c_2[1]) $(c_2[2]) $(c_2[3])" file stress.txt title "Sxy Sxz Syz"
```

2. Aggregation and Fragmentation Kinetics (6 points)

Consider a polydisperse system with discrete distribution of masses of particles. There is a smallest particle of mass m_1 (monomer) present in the system. The masses of other particles are $m_k = km_1$, where k $= 1, 2, \ldots, N$ is an integer number, N is the number of different species in the system. The amount of particles of mass m_k is n_k, the full amount of particles is $N = \sum_k n_k$. The evolution of number densities n_k of particles is described by the system of Smoluchowski equations:

$$\frac{\mathrm{d}n_1}{\mathrm{d}t} = -n_1 \sum_{j\geq 1} n_j + n_1 \sum_{j\geq 2} \lambda j n_j + \frac{1}{2} \sum_{i,j\geq 2} \lambda(i+j) n_i n_j, \tag{1}$$

$$\frac{\mathrm{d}n_k}{\mathrm{d}t} = \frac{1}{2} \sum_{i+j=k} n_i n_j - \sum_{i\geq 1} (1+\lambda) n_i n_k, \qquad k \geq 2 \tag{2}$$

Here the aggregation rates coeffcients $K_{ij} = 1$ and fragmentation rates are $F_{ij} = \lambda K_{ij} = \lambda$, where $0 < \lambda < 1$. We assume that the fragmentation events are complete, that is only monomers appear at the collisions: $X_i(k) = k\delta_{i1}$. Also, we assume the monodisperse initial conditions: $n_k(t=0) = \delta_{k1}$.

(a) Find the analytical steady-state (i.e., $\mathrm{d}n_k/\mathrm{d}t = 0$) solution of system (1) and (2) using the generating functions approach (**2 points**).

Hint 1. Use the Lectures Notes.

Hint 2. Use the relation:

$$(1-z)^{1/2} = 1 - \sum_k \frac{z^k}{2\sqrt{\pi}} \frac{\Gamma(k-1/2)}{\Gamma(k+1)}. \tag{3}$$

Hint 3. Take into account that at steady-state $n_1 = \lambda/(1+\lambda)$.

(b) Solve the system of Smoluchowski equations (1) and (2) numerically using the Euler method. Apply the fast solver scheme (in Python, you can use *fftconvolve* from *scipy* package). Take $\lambda = 0.3$ and $\lambda = 0.1$, plot both analytical versus numerical solutions and compare them (**4 points**).

Hint 4. Plot the results in $\log\log$ scale.

3. Gillespie algorithm (6 points)

Implement the Gillespie algorithm in order to solve system (1) and (2). Take $\lambda = 0.1, 0.3$ and present the outcome of the Monte Carlo solution on the same plot.

Description of the algorithm

The algorithm may be implemented in the following way. We assume, that at the initial time moment only the monomer units are present in the system: $n_1 = N = 100000$, $n_k = 0$ for $k > 1$. Two possible events can occur during the simulation:

1. Aggregation of particles i and j with formation of particle of size $i+j$:

$$\ldots, n_i, \ldots, n_j, \ldots, n_{i+j}, \ldots \qquad \rightarrow \qquad \ldots, n_i - 1, \ldots, n_j - 1, \ldots, n_{i+j} + 1, \ldots$$

The rate of the aggregation event is $p(A_{ij}) = n_i n_j$.

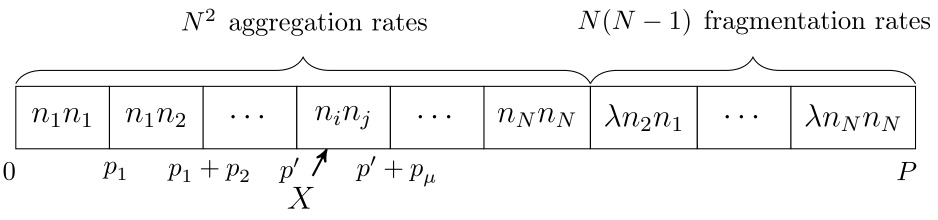

Figure 1: The random number X between 0 and P is generated. The sequence of aggregation and fragmentation rates $p(A_{ij}) = n_i n_j$ and $p(F_{ij}) = \lambda n_i n_j$ is produced and the event is choosen according to the position of X in this sequence.

2. Fragmentation of particles:

a) Disruptive collisions of particles of size $i \geq 2$ and $j \geq 2$:

$$n_1, \ldots, n_i, \ldots, n_j, \ldots \qquad \rightarrow \qquad n_1 + i + j, \ldots, n_i - 1, \ldots, n_j - 1, \ldots$$

b) Disruptive collisions of particles of size $i \geq 2$ and monomers, $j = 1$:

$$n_1, \ldots, n_i, \ldots \qquad \rightarrow \qquad n_1 + i, \ldots, n_i - 1, \ldots$$

The rate of a fragmentation event is $p(F_{ij}) = \lambda n_i n_j$.

The total rate of all possible events has the form:

$$P = \frac{1}{2} \sum_{i,j} n_i n_j + \lambda \sum_{i \geq 2} n_i \sum_{k \geq 1} n_k.$$

The time lag between two consequent events is: $\tau = -\frac{1}{P} \ln X_1$, where X_1 is a uniformly distributed random number between 0 and 1. The next event, which could be either an aggregation or a fragmentation event, is defined in the following way. The sequence of events with probabilities $p(A_{ij})$ and $p(F_{ij})$ is produced and a random number X from 0 to P is generated. According to its place in the sequence of events either aggregation or fragmentation of particles of masses m_i and m_j takes place (see Fig. 1).

Hint: C/C++ or Fortran implementations are recommended for a faster performance of the algorithm.