Lecture 1: Molecular Dynamics

L. Filion and M. Dijkstra

Contents

Chapter 1

Introduction to Molecular Dynamics

1.1 Molecular Dynamics

An important tool for studying many-body systems is Molecular Dynamics simulations. In Molecular Dynamics simulations, the equations of motions are numerically integrated creating trajectories through phase space that are similar to the trajectories followed in an experiment. More specifically, in Molecular Dynamics simulations, Newton's equations of motion are integrated numerically for typically $N = 100 - 100000$ particles, starting from some initial configuration. Similar to an experiment, after equilibrium is reached, we can measure time averages $\overline{\mathcal{A}}$ of a corresponding microscopic function $\mathcal{A}(\Gamma)$ over a certain time interval $t_0 \leq t \leq t_0 + \tau$ of a phase trajectory:

$$\overline{\mathcal{A}} = \lim_{\tau \to \infty} \frac{1}{\tau} \int_{t_0}^{t_0+\tau} dt \mathcal{A}\Big(\Gamma(t)\Big). \tag{1.1}$$

In contrast to Monte Carlo simulations, which only yield thermodynamic ensemble averages, Molecular Dynamics simulations can provide us with not only the equilibrium properties, but also the transport properties of the system. A Molecular Dynamics simulation is generally constructed as follows:

1. Start with a configuration Γ_o, i.e., select initial positions and velocities.

2. Calculate the forces on all particles. For a given pair of particles the α-component of the force ($\alpha = x, y, z$) is given by $f_\alpha(r) = -\partial\phi(r)/\partial r_\alpha = -(r_\alpha/r)(\partial\phi(r)/\partial r)$ with $\phi(r)$ the pair potential and $r = \sqrt{r_x^2 + r_y^2 + r_z^2}$ is the distance between two particles.

3. Integrate Newton's equations of motion to obtain the new positions and velocities.

4. Repeat step 2 and 3

In the next few sections, we discuss these quantities in detail.

1.2 Initialization

A Molecular Dynamics simulation is started from an initial configuration. We select first initial positions and velocities for every particle in our system. The positions of the particles are chosen such that the associated Boltzmann weight is non-zero, i.e., no hard-core overlaps. The particles can be placed at random in the simulation box, or they can be placed on lattice sites. We then assign a velocity to each particle, which can either be picked randomly in an interval $[-1; 1]$ for

```
subroutine init              Initialization of MD program

sumv=0
sumv2=0
do i=1,npart
   x(i)=latticepos(i)        place the particle on a lattice position
   v(i)=2×(ranf()-0.5)       give random velocity
   sumv=sumv+v(i)            total momentum
   sumv2=sumv2+v(i)**2       kinetic energy
enddo
sumv=sumv/npart
sumv2=sumv2/npart
fs=sqrt(3*temp/sumv2)        we assume a three-dimensional system
do i=1,npart                 scale velocities to desired temperature
   v(i)=(v(i)-sumv)*fs       total momentum is zero
enddo
return
end
```

Table 1.1: Pseudo computer code for an initialization of a Molecular Dynamics simulation. Function `latticepos` gives the coordinates of lattice position i and `ranf()` gives a random number uniformly distributed between $[0:1]$. We do not start with a Maxwell-Boltzmann velocity distribution, but after equilibration this distribution will be reached very quickly.

each component v_α, where α denotes the x, y, and z-direction, or can be taken from a Maxwellian velocity distribution. Subsequently, the velocities are shifted such that the total momentum is zero and the velocities are scaled such that the mean kinetic energy E_{kin} matches the desired temperature T:

$$E_{kin} = \sum_{i=1}^{N} \frac{m v_i^2}{2} = \frac{3 N k_B T}{2} \tag{1.2}$$

where $v_i^2 = v_{x,i}^2 + v_{y,i}^2 + v_{z,i}^2$. As $T \propto \sum_{i=1}^{N} v_i^2$, the instantaneous temperature $T(t)$ can be adjusted to match the desired temperature T by scaling all velocities with a factor $\sqrt{T/T(t)}$.

1.3 Force Calculation

In the next step of a Molecular Dynamics simulation, we determine the force acting on every particle. If we assume a system interacting with pairwise additive interactions, we calculate the contribution to the force on particle i due to the presence of all the other particles. If we consider only the interaction between a particle and the nearest image of another particle, we must evaluate $N(N-1)/2$ pairs of particles, where N is the number of particles in our simulation box. Hence, we first compute the distance between each pair of particles i and j. We use periodic boundary conditions and compute the nearest image distance. Moreover, it is often convenient to use a cut-off at a distance r_{cut}, where r_{cut} is less than half the diameter of the periodic box. We then calculate the force for a given pair between particle i and j as a result of the corresponding pair interaction.

Note that we have implicitly assumed here that the potential is continuous. If the potential is discontinuous, then another version of MD must be used - event driven molecular dynamics.

```
subroutine force(f,en)                calculation of the forces

en=0                                  set total energy to zero
do i=1,npart
   f(i)=0                             set forces to zero
enddo
do i=1,npart-1                        consider all particle pairs
   do j=i+1,npart
      xr=x(i)-x(j)                    distance between $i$ and $j$
      xr=xr-box*nint(xr/box)          periodic boundary conditions
      r2=xr**2
      if(r2.lt.rc2) then              rc2=rc*rc, where rc is the cut-off distance
        r2i=1/r2
        r6i=r2i**3
        ff=48*r2i*r6i(r6i-0.5)
        f(i)=f(i)+ff*xr               update force for particle i
        f(j)=f(j)-ff*xr               update force for particle j
        en=en+4*r6i*(r6i-1)-ecut      update total potential energy
      endif
   enddo
enddo
return
end
```

Table 1.2: Pseudo computer code for the calculation of the forces in a Molecular Dynamics simulation. `ecut` denotes the value of the potential at $r = r_{\mathrm{cut}}$. For the Lennard-Jones potential `ecut` is given by $4(1/r_{\mathrm{cut}}^{12} - 1/r_{\mathrm{cut}}^{6})$.

We will not delve into that further here.

1.4 Integration of the Equations of Motion

For the integration of Newton's equations of motion, we employ the Verlet algorithm, which is based on a Taylor expansion of the coordinate of a particle at time $t + \Delta t$ and $t - \Delta t$ about time t:

$$r(t + \Delta t) = r(t) + v(t)\Delta t + \frac{f(t)}{2m}\Delta t^2 + \frac{\partial^3 r}{\partial t^3}\frac{\Delta t^3}{3!} + \mathcal{O}(\Delta t^4). \tag{1.3}$$

and

$$r(t - \Delta t) = r(t) - v(t)\Delta t + \frac{f(t)}{2m}\Delta t^2 - \frac{\partial^3 r}{\partial t^3}\frac{\Delta t^3}{3!} + \mathcal{O}(\Delta t^4). \tag{1.4}$$

Adding these two equations and subtracting $r(t - \Delta t)$ on both sides gives us

$$r(t + \Delta t) = 2r(t) - r(t - \Delta t) + \frac{f(t)}{m}\Delta t^2 + \mathcal{O}(\Delta t^4). \tag{1.5}$$

Note that the new position is accurate to order Δt^4. The Verlet algorithm does not use the velocity to compute the new position, but the velocity can be derived from the trajectory, which is only accurate to order Δt^2:

$$v(t) = \frac{r(t + \Delta t) - r(t - \Delta t)}{2\Delta t} + \mathcal{O}(\Delta t^2) \tag{1.6}$$

The velocities can be used to compute the kinetic energy and, thereby, the instantaneous temperature. It is important to note that Eqs. 1.5 and 1.6 conserve

- the total linear momentum of the system, i.e.

$$\frac{d}{dt}\sum_{i=1}^{N} m_i v_i(t) = 0 \tag{1.7}$$

- the total energy of the system, which is the sum of the potential energy

$$E_{pot} = \sum_{i<j} u_{ij}(\mathbf{r}_{ij}) \tag{1.8}$$

  and the kinetic energy E_{kin} (see Eq. 1.2).

1.5 Other Integration Algorithms

In a Molecular Dynamics simulation it is essential to have a good algorithm to integrate Newton's equation of motions. Different integration algorithms exist and a good algorithm should satisfy several criteria. First, the algorithm should be fast, and require little memory. Second, it should be sufficiently accurate for large time steps as larger time steps results in fewer evaluations of the force and a speed up of the simulation. Third, it should satisfy energy and momentum conservation, and it should be time reversible (see Ref. [3]). The Verlet algorithm satisfies these criteria. Below, we discuss a few alternatives for this algorithm.

The Euler algorithm is simply based on a Taylor expansion of the particle coordinates truncated beyond the term in Δt^2 and reads

$$\begin{aligned} r(t + \Delta t) &= r(t) + v(t)\Delta t + \frac{f(t)}{2m}\Delta t^2 && (1.9) \\ v(t + \Delta t) &= v(t) + \frac{f(t)}{m}\Delta t && (1.10) \end{aligned}$$

```
subroutine integrate(f,en)           Integration of equations of motion

sumv=0
sumv2=0
do i=1,npart
   xx=2*x(i)-xm(i)+delt**2*f(i)      Verlet algorithm (Eq. 1.5)
   vi=(xx-xm(i))/(2*delt)
   sumv2=sumv2+vi**2
   xm(i)=x(i)                        update positions
   x(i)-xx
enddo
temp=sumv2/(3*npart)                 temperature
etot=(en+0.5*sumv2)/npart            total energy per particle
return
end
```

Table 1.3: Pseudo computer code for the integration of the equations of motion in a Molecular Dynamics simulation.

However, it suffers from a tremendous energy drift and the Euler algorithm should be avoided at any time. Another algorithm is the so-called Leap Frog algorithm. It evaluates the velocities at half-integer time steps and uses these velocities to compute new positions. The velocities at half-integer time steps are defined as follows

$$v(t-\Delta t/2) = \frac{r(t)-r(t-\Delta t)}{\Delta t} \tag{1.11}$$

$$v(t+\Delta t/2) = \frac{r(t+\Delta t)-r(t)}{\Delta t} \tag{1.12}$$

The latter equation yields an expression for the new positions, based on the old positions and velocities

$$r(t+\Delta t) = r(t) + v(t+\Delta t/2)\Delta t \tag{1.13}$$

Combining Eqs. 1.11 and 1.12 and the Verlet algorithm (1.5), we arrive at the following update for the velocities

$$v(t+\Delta t/2) = v(t-\Delta t/2) + \Delta t\frac{f(t)}{m} \tag{1.14}$$

It is important to note that the kinetic and potential energy are not defined at the same time, and hence we cannot compute the total energy directly in the Leap Frog scheme. In the velocity Verlet algorithm, the positions and velocities are computed at equal times. The update of the coordinates in this algorithm is based on a Taylor expansion for the coordinates

$$r(t+\Delta t) = r(t) + v(t)\Delta t + \frac{f(t)}{2m}\Delta t^2 \tag{1.15}$$

while the update of the velocities is

$$v(t+\Delta t) = v(t) + \frac{f(t+\Delta t)+f(t)}{2m}\Delta t \tag{1.16}$$

In this algorithm, we can only compute the new velocities after we have computed the new coordinates and from these the new forces. This scheme is, however, equivalent to the original Verlet algorithm.

```
program md                    Molecular Dynamics algorithm

call init                     select initial positions and velocities
t=0.0                         time t = 0
do while (t.lt.tmax)          for t < tmax
   call force(f,en)           calculate the forces on all particles
   call integrate(f,en)       integrate equations of motion
   t=t+deltat                 update the time
   call sample                sample the time averages
enddo
end
```

Table 1.4: Pseudo computer code for a Molecular Dynamics simulation.

1.6 The Andersen Thermostat

In the canonical MD scheme of Andersen the system is coupled to a heat bath that imposes the desired temperature. The coupling to a heat bath is represented by stochastic forces that act occasionally on randomly selected particles. These stochastic collisions with the heat bath can be considered as Monte Carlo moves that transport the system from one constant-energy shell to another. Between stochastic collisions, the system evolves at constant energy according to the normal Newton's equations of motion. The stochastic collisions ensure that all accessible constant-energy shells are visited according to their Boltzmann weight. A constant-temperature simulation consists of the following steps:

1. Start with an initial set of positions and momenta $\{\mathbf{r}^N(0), \mathbf{p}^N(0)\}$ and integrate the equations of motion for a time Δt.

2. A number of particles is selected to undergo a collision with the heat bath. The probability that a particle is selected in a time step of length Δt is $\nu\Delta t$, where ν is the frequency of stochastic collisions which determines the coupling strength to the heat bath.

3. If particle i has been selected to undergo a collision, its new velocity will be drawn from a Maxwell-Boltzmann distribution corresponding to the desired temperature T. All other particles are unaffected.

1.7 The Nosé Hoover Thermostat

Here we want to address whether we can design a thermostat that is not stochastic in nature. We begin by considering the Lagrangian for our system. Following Nosé, we add an extra variable to the Lagrangian s:

$$\mathcal{L}_{NOSE} = \sum_{i=1}^{N} \frac{m_i}{2} s^2 \dot{\mathbf{r}}_\mathbf{i}^2 - U(\mathbf{r}^N) + \frac{Q}{2}\dot{s}^2 - \frac{L}{\beta}\ln s$$

where Q is an "effective" mass associated with s, and L is a parameter which will be fixed during the derivation. As we will show in the rest of this section, we can use this extra variable to impose a constant-temperature constraint on our molecular dynamics simualtion.

```
program mdAndersen                        MD at constant temperature

call init(temp)                           initialization
call force(f,en)                          determine the forces
t=0
do while (t.lt.tmax)                      MD loop
   call integrate(1,f,en,temp)            first part of the equations of motion
   call force(f,en)
   call integrate(2,f,en,temp)            second part of the equations of motion
   t=t+dt
   call sample                            sample averages
enddo
end


subroutine integrate(switch,f,en,temp)    integrate equations of motion
if(switch.eq.1) then                      first step velocity Verlet
   do i=1,npart
      x(i)=x(i)+dt*v(i)+dt*dt*f(i)/2      update positions
      v(i)=v(i)+dt*f(i)/2                 update velocity
   enddo
elseif (switch.eq.2) then                 second step velocity Verlet
   tempa=0
   do i=1,npart
      v(i)=v(i)+dt*f(i)/2                 second update velocity
      tempa=tempa+v(i)**2
   enddo
   tempa=tempa/(s*npart)                  instantaneous temperature
   sigma=sqrt(temp)                       Andersen heat bath
   do i=1,npart
      if(ranf().lt.nu*dt) then            test for collision with bath
          v(i)=gauss(sigma)               give particle Gaussian velocity
      endif
   enddo
endif
return
end
```

Table 1.5: Pseudo computer code for a Molecular Dynamics simulation using the velocity Verlet algorithm with the Andersen thermostat.

Following standard classical mechanics definitions, we can determine the momenta associated with the above Lagrangian variables. Specifically, we find

$$\begin{aligned}\mathbf{p_i} &\equiv \frac{\partial \mathcal{L}_{NOSE}}{\partial \dot{\mathbf{r}}_i} = m_i s^2 \dot{\mathbf{r}}_\mathbf{i} \\ p_s &\equiv \frac{\partial \mathcal{L}_{NOSE}}{\partial \dot{s}} = Q\dot{s}\end{aligned}$$

In statistical physics we usually work with Hamiltonian of a system and not the Lagrangian. Hence, next we determine the Hamiltonian associated with these new "momenta" variables. This is given by

$$\mathcal{H}_{NOSE} = \sum_{i=1}^{N} \frac{{\mathbf{p}_i}^2}{2m_i s^2} + U(\mathbf{r}^N) + \frac{p_s^2}{2Q} + \frac{L}{\beta}\ln s$$

where we have used the expression $\mathcal{H} = \sum_i p_i q_i - \mathcal{L}$ which relates a Hamiltonian to a Lagrangian. With the Hamiltonian, we can determine the statistical physics quantities associated with this systems. Specifically, the associated microcanonical partition function is given by

$$Q_{NOSE}(E,V,N) = \frac{c}{N!}\int \mathrm{d}p_s \mathrm{d}s \mathrm{d}\mathbf{p}^N \mathrm{d}\mathbf{r}^N \delta(E - \mathcal{H}_{NOSE})$$

where c is simply a constant.

Let

$$\mathbf{p}' = \frac{\mathbf{p}}{s} \tag{1.17}$$

and

$$\mathcal{H}(\mathbf{p}', \mathbf{r}) = \sum_{i=1}^{N} \frac{\mathbf{p}_i'^2}{2m_i} + U(\mathbf{r}^N).$$

Making these substitutions to our Hamiltonian, we obtain

$$\begin{aligned}Q_{NOSE} &= \frac{c}{N!}\int \mathrm{d}p_s \mathrm{d}s \mathrm{d}\mathbf{p}^N \mathrm{d}\mathbf{r}^N \delta(E - \mathcal{H}_{NOSE}) \\ &= \frac{c}{N!}\int \mathrm{d}\mathbf{p}'^N \mathrm{d}\mathbf{r}^N \mathrm{d}p_s \mathrm{d}s \;\; s^{3N}\delta\left(\sum_{i=1}^{N}\frac{\mathbf{p}_i'^N}{2m_i} + U(\mathbf{r}^N) + \frac{p_s^2}{2Q} + \frac{L}{\beta}\ln s - E\right) \\ &= \frac{c}{N!}\int \mathrm{d}\mathbf{p}'^N \mathrm{d}\mathbf{r}^N \mathrm{d}p_s \mathrm{d}s \;\; \frac{\beta s^{3N+1}}{L}\delta\left(s - \exp\left(-\beta\frac{\mathcal{H}(\mathbf{p}',\mathbf{r}) + p_s^2/(2Q) - E}{L}\right)\right) \\ &= \frac{C}{N!}\int d\mathbf{p}'^N d\mathbf{r}^N e^{-\beta\frac{3N+1}{L}\mathcal{H}(\mathbf{p}',\mathbf{r})}\end{aligned}$$

where we have used the fact that if $f(x)$ is a function with roots at x_i, then

$$\delta\left(f(x)\right) = \sum_i \frac{\delta(x - x_i)}{|f'(x_i)|}.$$

Note that the expression we now have for Q_{NOSE} no longer depends on s or p_s, but still contains L.

Using the partition function Q_{NOSE} we can determine the average value of a measurable A associated with this contructed system. We find:

$$\langle A(\mathbf{p}', \mathbf{r})\rangle = \frac{\int d\mathbf{p}'^N d\mathbf{r}^N A(\mathbf{p}',\mathbf{r}) e^{-\beta\frac{3N+1}{L}\mathcal{H}(\mathbf{p}',\mathbf{r})}}{\int d\mathbf{p}'^N d\mathbf{r}^N e^{-\beta\frac{3N+1}{L}\mathcal{H}(\mathbf{p}',\mathbf{r})}}$$

If we make the choice that $L = 3N + 1$ (which we are free to do as L was simply a parameter in our original Lagrangian), this simplifies to the expression we would have for a canonical ensemble with variables $\mathbf{r}$ and $\mathbf{p}'$:

$$\left\langle A(\mathbf{p}', \mathbf{r}) \right\rangle \;=\; \left\langle A(\mathbf{p}', \mathbf{r}) \right\rangle_{\text{canonical}} \quad \text{with Hamiltonian } \mathcal{H}(\mathbf{p}', \mathbf{r})$$

One way to think about our current system is that we now have "real" and "virtual" variables defined such that

real	virtual
$\mathbf{r}'$	$\mathbf{r}$
$\mathbf{p}'$	$\mathbf{p}/s$
s'	s
$\Delta t'$	$\Delta t/s$

Examining these variable in detail, we see immediately one worrying point: if we were to measure at equal time intervals in Δt, then the intervals are not equally spaced in "real" time, meaning that if we were to simulate the Nosé Hamiltonian, "real" time would fluctuate during simulation! This is clearly undesirable, and leads to the immediate question: "How do we sample at equal intervals in "real" time?"

It can be shown that

$$\bar{A} = \lim_{\tau' \to \infty} \frac{1}{\tau'} \int_0^{\tau'} dt' A\left(\mathbf{p}(t')/s(t'), \mathbf{r}(t')\right) = \left\langle \frac{A(\mathbf{p}', \mathbf{r})}{s} \right\rangle / \left\langle \frac{1}{s} \right\rangle$$

Rewriting these expectation values in terms of the partition function, we find

$$\left\langle \frac{A(\mathbf{p}', \mathbf{r})}{s} \right\rangle / \left\langle \frac{1}{s} \right\rangle \;=\; \frac{\frac{\int d\mathbf{p}'^N d\mathbf{r}^N A(\mathbf{p}',\mathbf{r}) \exp\left(-\beta\mathcal{H}(\mathbf{p}',\mathbf{r})3N/L\right)}{\int d\mathbf{p}'^N d\mathbf{r}^N \exp\left(-\beta\mathcal{H}(\mathbf{p}',\mathbf{r})(3N+1)/L\right)}}{\frac{\int d\mathbf{p}'^N d\mathbf{r}^N \exp\left(-\beta\mathcal{H}(\mathbf{p}',\mathbf{r})3N/L\right)}{\int d\mathbf{p}'^N d\mathbf{r}^N \exp\left(-\beta\mathcal{H}(\mathbf{p}',\mathbf{r})(3N+1)/L\right)}} \tag{1.18}$$

Now, if we have choose $L = 3N$, this reduces to the canonical average for the "real" variables.

Before we continue further a few points should be made. Firstly, the Nosé equations of motion can be written in terms of either "virtual" or "real" variables, but it is typically undesirable to simulate the "virtual" variables as in this case "real" time fluctuates during the simulation. Secondly, derivatives of the virtual variables with respect to time can be related directly to derivatives of the Nosé Hamiltonian with respect to the associated momenta. Thirdly, the "real" variables can be written in terms of the "virtual" variables. Putting this all together, we obtain:

$$\begin{aligned}
\frac{d\mathbf{r}'_i}{dt'} &= s\frac{d\mathbf{r}_i}{dt} = s\frac{\partial \mathcal{H}_{NOSE}}{\partial \mathbf{p}_i} = \frac{\mathbf{p}'_i}{m_i} \\
\frac{p'_i}{dt'} &= s\frac{d(\mathbf{p}_\mathbf{i}/s)}{dt} = \frac{d\mathbf{p}_i}{dt} - \frac{\mathbf{p}_i}{s}\frac{ds}{dt} = -\frac{\partial \mathcal{H}_{NOSE}}{\partial \mathbf{r}_i} - \frac{\mathbf{p}'_i}{s}\frac{\partial \mathcal{H}_{NOSE}}{\partial p_s} \\
&= -\frac{\partial U(\mathbf{r}'^N)}{\partial \mathbf{r}'_i} - \frac{s' p'_s}{Q}\mathbf{p}'_i \\
\frac{ds'}{dt'} &= s\frac{ds}{dt} = s\frac{\partial \mathcal{H}_{NOSE}}{\partial p_s} = \frac{(s')^2 p'_s}{Q} \\
\frac{d(s' p'_s/Q)}{dt'} &= \frac{s}{Q}\frac{dp_s}{dt} = -\frac{s}{Q}\frac{\partial \mathcal{H}_{NOSE}}{\partial s} = \left(\sum_i \frac{\mathbf{p}'^2_i}{m_i} - \frac{L}{\beta} \right) /Q.
\end{aligned} \tag{1.19}$$

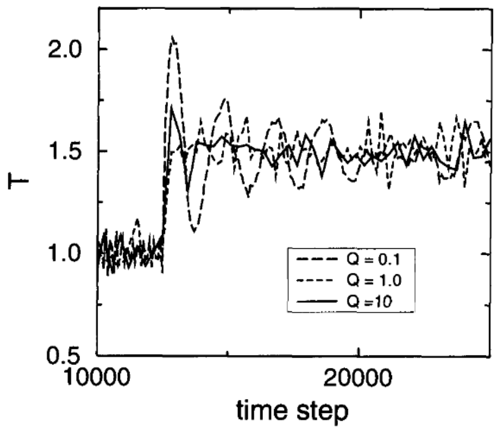

Figure 1.1: Lennard-Jones fluid. Taken from "Understanding Molecular Simulations" by Frenkel and Smit

In principle, we now have equations of motion for the "real" variables. However, if we look closely, we see that s, p_s and Q only occur as

$$\xi = s' p'_s / Q$$

. Using this information, Hoover rewrote these expressions in terms of a so-called "thermodynamic friction coefficient". Following Hoover, the equations of motion become:

$$\begin{aligned}
\dot{\mathbf{r}}' &= \frac{\mathbf{p}'_i}{m_i} \\
\dot{\mathbf{p}}' &= -\frac{\partial U(\mathbf{r}'^N)}{\partial \mathbf{r}'_i} - \xi \mathbf{p}'_i \\
\dot{\xi} &= \left(\sum_i \frac{p_i'^2}{m_i} - \frac{L}{\beta} \right) / Q \\
\frac{\dot{s}'}{s'} &= \xi .
\end{aligned}$$

Interestingly, the first three equations actually make a complete set and as such, the last is not strictly necessary for simulations. However, it can be useful to simulation to ensure that simulations are running correctly.

One might ask, what is the role of the thermodynamic friction coefficient? In short, $\xi > 0$ slows down all particles and $\xi < 0$ speeds up all particles. Moreover,the time evolution of ξ is proportional to the difference between the kinetic energy of the system and the kinetic energy of a system with temperature T. Finally, Q sets how quickly ξ can change in magnitude (see Figure 1.1).

For more information on the Nosé Hoover thermostat, please see the book by Frenk and Smit [3].

Chapter 2

Diffusion

If we tag a number of particles in a specific region of a fluid, after a time this inhomogeneity that we imposed is "washed out" due to the motion of the particles (see Figure 2.1). This is diffusion.

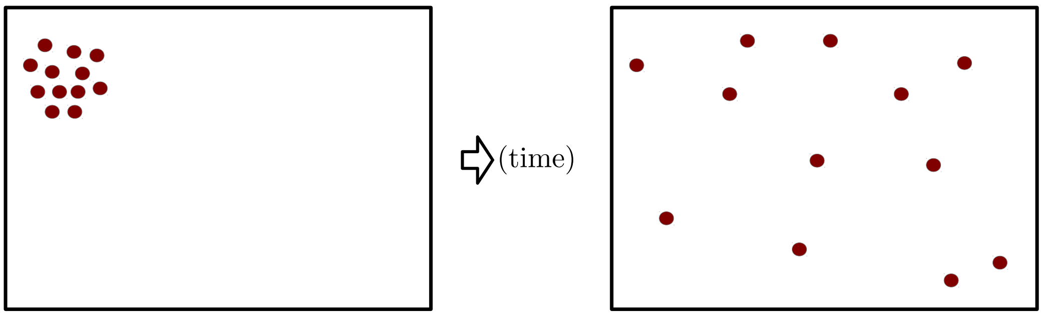

Figure 2.1: Example of diffusion; the random (diffusive) motion of the particles removes inhomogeneities hence the labeled particles grouped together on the left spread out such that eventually they are approximately evenly distributed throughout the available space. Note that there are a number of non-labeled particles in the background which are not depicted in this cartoon.

More explicitly, Fick's law says that the flux of labeled particles at position $\mathbf{r}$ and time t, which we will denote $\mathbf{j}$ is proportional to (negative) the gradient of the concentration which we denote $c(\mathbf{r}, t)$, i.e.

$$\mathbf{j}\left(\mathbf{r}, t\right) = -D\nabla c\left(\mathbf{r}, t\right), \tag{2.1}$$

where D is called the self-diffusion coefficient.

If we consider a subvolume of the system Γ, contained within a surfaced denoted S, then the total number of particles in this volume is simply

$$N_\Gamma(t) = \int_\Gamma d\mathbf{r}\; c\left(\mathbf{r}, t\right), \tag{2.2}$$

and so

$$\frac{dN_\Gamma}{dt} = \int_\Gamma d\mathbf{r}\; \frac{\partial c\left(\mathbf{r}, t\right)}{\partial t}. \tag{2.3}$$

Additionally, if we assume we are in a system where the particles themselves cannot be created nor destroyed, which is most generally the case aside from systems undergoing a chemical reaction, then the only way to change the number of particles in the volume Γ is for particles to

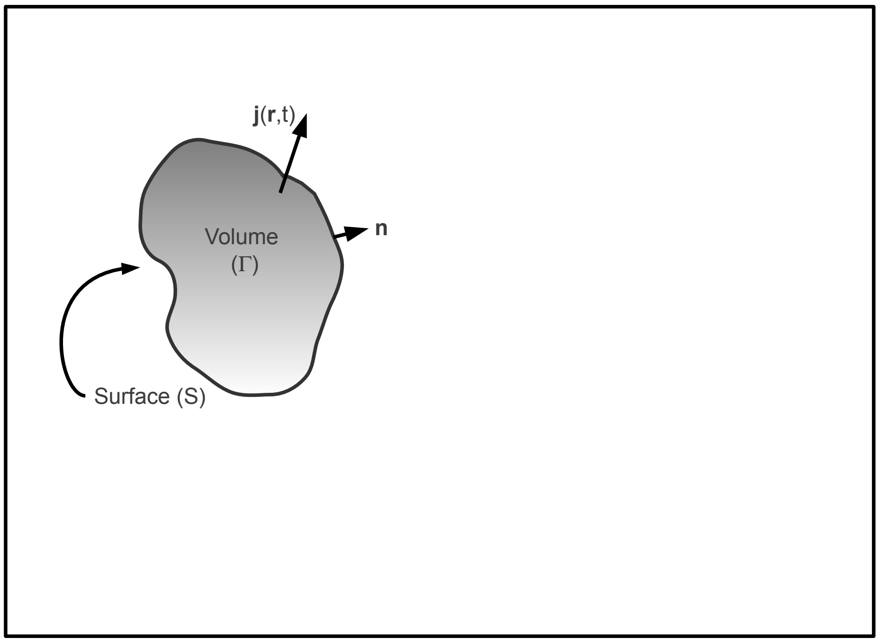

Figure 2.2: The region contained within the surface S is a subvolume (Γ) of a total volume V. The only way to change the number of particles in Γ is for particles to flow in or out of this volume.

flow into or out of this volume (see Figure 2.2). The flux along the normal to the surface $\mathbf{n}$ is simply $\mathbf{j}\left(\mathbf{r},t\right)\cdot\mathbf{n}$, so the net amount flux flowing through the total surface S per unit time is

$$\int_S dS\,\mathbf{j}\left(\mathbf{r},t\right)\cdot\mathbf{n}. \tag{2.4}$$

The normal in Figure 2.2 is pointing out of the surface, thus the flux as depicted here relates to the particles *leaving* the region Γ. Hence the change in the number of particles contained within the surface S is

$$\frac{dN_\Gamma}{dt} = -\int_S dS\,\mathbf{j}\left(\mathbf{r},t\right)\cdot\mathbf{n} = -\int_\Gamma d\mathbf{r}\,\nabla\cdot\mathbf{j}\left(\mathbf{r},t\right), \tag{2.5}$$

where the last "equals" follows from the divergence theorem.

Putting Equations 2.3 and 2.5 together results in

$$\int_\Gamma d\mathbf{r}\left(\frac{\partial c\left(\mathbf{r},t\right)}{\partial t} + \nabla\cdot\mathbf{j}\left(\mathbf{r},t\right)\right) = 0 \tag{2.6}$$

However, since this expression must be valid for all choices of Γ, the integrand itself must to identically zero. The resulting relation is referred to as the "equation of continuity":

$$\frac{\partial c\left(\mathbf{r},t\right)}{\partial t} = -\nabla\cdot\mathbf{j}\left(\mathbf{r},t\right). \tag{2.7}$$

Combining the equation of continuity (Equation 2.7) together with Fick's Law (Equation 2.1), we obtain

$$\frac{\partial c\left(\mathbf{r},t\right)}{\partial t} - D\nabla^2 c\left(\mathbf{r},t\right) = 0 \tag{2.8}$$

In the following we will determine the time dependence of the mean square width in the concentration. The mean-square width defined as

$$\left\langle r^2(t)\right\rangle = \frac{\int d\mathbf{r}\; r^2 c(\mathbf{r},t)}{\int d\mathbf{r}\; c(\mathbf{r},t)} \tag{2.9}$$

gives a measure of the spread in the concentration. For simplicity, we will define a normalized concentration $\tilde{c}$ such that

$$\tilde{c}(\mathbf{r},t) = \frac{c(\mathbf{r},t)}{\int d\mathbf{r}\; c(\mathbf{r},t)} \tag{2.10}$$

so that the mean square displacement is simply

$$\left\langle r^2(t)\right\rangle = \int d\mathbf{r}\; r^2 \tilde{c}(\mathbf{r},t). \tag{2.11}$$

Additionally, we will restrict this discussion to one dimension and leave the two and three dimensional cases for an exercise. In one dimension the mean square width reduces to

$$\left\langle x^2(t)\right\rangle = \int_{-\infty}^{\infty} dx\; x^2 \tilde{c}(x,t). \tag{2.12}$$

Taking the derivative with respect to time we obtain

$$\frac{d}{dt}\left\langle x^2(t)\right\rangle = \frac{\partial}{\partial t}\int_{-\infty}^{\infty} dx\; x^2 \tilde{c}(x,t) = \int_{-\infty}^{\infty} dx\; x^2 \frac{\partial \tilde{c}(x,t)}{\partial t} \tag{2.13}$$

Using the one dimensional version of Equation 2.8 we obtain

$$\frac{d}{dt}\left\langle x^2(t)\right\rangle = D\int_{-\infty}^{\infty} dx\; x^2 \frac{\partial^2 \tilde{c}(x,t)}{\partial x^2}. \tag{2.14}$$

Integrating by parts twice we obtain

$$\frac{d}{dt}\left\langle x^2(t)\right\rangle = D\left(x^2 \frac{\partial \tilde{c}(x,t)}{\partial x}\Bigg|_{-\infty}^{\infty} - 2x\tilde{c}(x,t)\Bigg|_{-\infty}^{\infty} + 2\int_{-\infty}^{\infty} dx\; \tilde{c}(x,t)\right) \tag{2.15}$$

However, since the diffusing particles cannot reach infinity in finite time, we have that $\tilde{c}(-\infty,t) = \tilde{c}(\infty,t) = 0$ and $\partial\tilde{c}(-\infty,t)/\partial x = \partial\tilde{c}(\infty,t)/\partial x = 0$. Thus

$$\frac{d}{dt}\left\langle x^2(t)\right\rangle = 2D \tag{2.16}$$

where we have used the fact that $\tilde{c}(x,t)$ is normalized to preform the last integral. Hence,

$$\left\langle x^2(t)\right\rangle = 2Dt. \tag{2.17}$$

To derive the same equation in higher dimensions (e.g. d = 3), recall that $r^2 = x^2 + y^2 + z^2$. Using this, we obtain $\left\langle r^2(t)\right\rangle = 2dDt$, where d is the dimension of the system.

Note that the mean-square width also corresponds to the average behaviour of a particle. Hence, the conditional probability distribution that a particle is located at position $\mathbf{r}$ at time t undergoing diffusive motion follows the same expression.

2.1 Green-Kubo Relation

There is another way to think of the average $\left\langle \mathbf{r}^2\left(t\right)\right\rangle$: it is simply the average-squared distance traveled by the labeled particles in time t. Specifically,

$$\left\langle \mathbf{r}^2\left(t\right)\right\rangle = \frac{1}{N}\sum_{i}^{N}\Delta\mathbf{r_i}\left(t\right)^2 \tag{2.18}$$

where N is the number of labeled particles and $\Delta\mathbf{r_i}\left(t\right) = \mathbf{r_i}\left(t\right) - \mathbf{r_i}\left(0\right)$ with $\mathbf{r_i}\left(0\right)$ the position of particle i at time $t = 0$. Since the displacement of a single particle is simply

$$\Delta\mathbf{r_i}\left(t\right) = \int_0^t dt'\mathbf{v_i}\left(t'\right) \tag{2.19}$$

where $\mathbf{v_i}$ is the velocity of the particle. Hence

$$\left\langle \mathbf{r}^2\left(t\right)\right\rangle = \int_0^t dt' \int_0^t dt'' \ \left\langle \mathbf{v}\left(t'\right)\cdot\mathbf{v}\left(t''\right)\right\rangle . \tag{2.20}$$

The time derivative can then be written

$$\frac{d}{dt}\left\langle \mathbf{r}^2\left(t\right)\right\rangle = 2\int_0^t dt'\left\langle \mathbf{v}\left(0\right)\cdot\mathbf{v}\left(t\right)\right\rangle . \tag{2.21}$$

To show the above equality, we used the fact that $\langle A(t)B(t')\rangle = \langle A(t+t'')B(t'+t'')\rangle$ and that $\langle A(t)B(t')\rangle = f(t-t') = f(t'-t)$.

Using $\left\langle r^2\left(t\right)\right\rangle = 2dDt$ we finally obtain

$$D = \frac{1}{d}\int_0^\infty dt \ \left\langle \mathbf{v}\left(0\right)\cdot\mathbf{v}\left(t\right)\right\rangle ; \tag{2.22}$$

hence, we have an expression linking the self-diffusion coefficient to the *velocity autocorrelation function* $\left\langle \mathbf{v}\left(t'\right)\cdot\mathbf{v}\left(t''\right)\right\rangle$. Equation 2.22 is an example of a Green-Kubo formula since it links the diffusion coefficient which is a transport coefficient to the integral of an autocorrelation function.

Bibliography

[1] Allen, M.P.; Tildesley, D.J., *Computer Simulation of Liquids* Clarendon Press; Oxford, 1987.

[2] Landau, D.P.; Binder, K., *A Guide to Monte Carlo Simulations in Statistical Physics* Cambridge University Press; Cambridge, 2000.

[3] Frenkel, D.; Smit, B., *Understanding Molecular Simulation: from Algorithms to Applications*, 2nd ed. Academic Press; San Diego, 2002.

[4] Rapaport, D.C., *The art of molecular dynamics simulation*, 2nd ed. Cambridge University Press; Cambridge, 2004.

[5] Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller, A.N.; Teller, E. *J. Chem. Phys.* **1953**, *21*, 1087–1092.

[6] Manousiouthakis, V.I.; Deem, M.W. *J. Chem. Phys.* **1999**, *110*, 2753–2756.

[7] Panagiotopoulos, A.Z. *Mol. Phys.* **1987**, *61*, 813–826.

[8] Panagiotopoulos, A.Z. *Mol. Phys.* **1987**, *62*, 701.

[9] Polson, J.M.; Trizac, E.; Pronk, S.; Frenkel, D. *J. Chem. Phys.* **2000**, *112*, 5339–5342.