5. Computer Simulation Methods

A. General Considerations

We have just seen that successful analytical theories for simple liquids like Ar- gon can be constructed by means of modern perturbation theories of liquids. However, these calculations become more difficult and tedious when one moves to consider more complicated atomic and molecular fluids. Moreover, computers have gotten much faster since the 1970’s, so people today are more likely to turn to computer simulations of fluids.

a1. Choice of Method

There are basically three methods used for liquids and complex fluids:

**Molecular dynamics** (MD) – One integrates Newton’s equations for a collection of particles, keeping track of their positions and velocities at each time step. Thermo properties are computed as time averages over dynamical trajectories.

**Brownian dynamics** (BD) – Similar to MD, except that stochastic force and frictional drag terms are included. Usually used for colloids or polymers when solvent can be treated as a continuum.

**Monte Carlo** (MC) – A method for sampling the configurational integral by means of a “pseudo” dynamical trajectory.
<table>
<thead>
<tr>
<th>System</th>
<th>Technique</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>atomic or molecular</td>
<td>MC (statics)</td>
<td>MC faster</td>
</tr>
<tr>
<td>liquid</td>
<td>MD (dynamics)</td>
<td>for statics</td>
</tr>
<tr>
<td>polymer solution or melt</td>
<td>MC (statics)</td>
<td>coarse-grained</td>
</tr>
<tr>
<td></td>
<td>BD/MD (dynamics)</td>
<td>models often used</td>
</tr>
<tr>
<td>colloids</td>
<td>MC (statics)</td>
<td>solvent is</td>
</tr>
<tr>
<td></td>
<td>BD (dynamics)</td>
<td>continuum</td>
</tr>
</tbody>
</table>

### a2. Periodic BCs and Potential Truncation

Unfortunately, we can only afford to run fairly small simulations with $N \lesssim 10^3 \sim 10^6$ for atoms, much less for polymers! Different ensembles can be used; in fixed $N,V$ we choose $N,V$ to establish some $\rho$. It is also best to choose $N$ consistent with any known crystal structure, e.g., fcc argon: $N = 4n^3$, $n = 1, 2, 3, \ldots$.

To minimize “finite size” effects in the simulation cell (mostly surface efforts), it is typical to impose periodic BCs:

When it comes to computing forces, one also uses the “minimum image convention”:

1. translate a cell of the same size (L) and shape to be centered around the molecule of interest.

2. Sum the forces with the $N-1$ molecules in the translated cell. These are the closest periodic images of these molecules.

Frequently, one also truncates the potential at some cutoff distance $r_c$ (e.g., $2.5\sigma$). It is crucial that

$$r_c \leq L/2$$

for consistency with the minimum image convention as shown in the attached figure:
B. The MD Method

The basic idea here is to solve Newton’s equations of motion:

\[ m \ddot{\mathbf{r}}_i = \mathbf{F}_i \quad i = 1, 2, \ldots, N \]

forward in time, starting from some set of initial positions \( \{ \mathbf{r}_i \} \) and velocities \( \{ \dot{\mathbf{r}}_i \} \). Many finite difference algorithms are available; the simplest and most popular is the Verlet algorithm. Start with:

\[ \mathbf{r}_i(t \pm \delta t) = \mathbf{r}_i(t) \pm \delta \mathbf{r}(t) + \frac{\delta t^2}{2!} \ddot{\mathbf{r}}_i(t) \pm \ldots \]

Adding the + and the −:

1. \[ \mathbf{r}_i(t + \delta t) = -\mathbf{r}_i(t - \delta t) + 2\mathbf{r}_i(t) + \frac{\delta t^2}{2!} \ddot{\mathbf{r}}_i(t) + \theta(\delta t^4) \]

Subtracting the + and the −:

2. \[ \dot{\mathbf{r}}_i(t) = \frac{1}{2\delta t}[\mathbf{r}_i(t + \delta t) - \mathbf{r}_i(t - \delta t)] + \theta(\delta t^2) \]

We can thus iterate these to get a series of new positions and velocities at equally-spaced time increments \( \delta t \), known as the “timestep”.

The most expensive part of an MD simulation is the force calculation. Let’s see how this works for the LJ potential:

\[ \mathbf{F}_i = \sum_{j \neq i} \mathbf{F}_{ij}(r_{ij}) \text{, where} \]

\[ \mathbf{F}_{ij} = -\frac{\partial}{\partial r_i} u(r_{ij}) = -\frac{\partial}{\partial r_{ij}} u(|\mathbf{r}_{ij}|) \]

Now, consider \( \frac{\partial}{\partial \mathbf{x}} u(|\mathbf{u}|) \), \( \mathbf{x} = (x_1, x_2, x_3) \)

\[ \frac{\partial}{\partial x_1} u(x) = \frac{\partial x}{\partial x_1} \frac{\partial}{\partial x} u = \frac{\partial x}{\partial x_i} u'(x) \]

\[ x^2 = \sum_i x_i^2, \quad 2x \frac{\partial x}{\partial x_i} = 2x_i \]

\[ \Rightarrow \frac{\partial}{\partial x_i} u(x) = \frac{x_i}{x} u'(x) \quad \text{[or \( \nabla u(|\mathbf{x}|) = \mathbf{x} u'(x) \)]} \]

Thus,

\[ \mathbf{F}_{ij} = -\frac{1}{r_{ij}} \left( \frac{d}{dr_{ij}} \right) r_{ij} = -\frac{w(t_{ij})}{r_{ij}^3} r_{ij} \]
where \( w(r) = r \frac{\partial u}{\partial r} \) is the so-called “pair virial function”. For the LJ 6–12:

\[
F_{ij} = \frac{24\epsilon}{r_{ij}^6}[2(\sigma/r_{ij})^6]r_{ij}
\]

**Lecture 8**

**MD summary**

\[
\dot{\mathbf{r}}_i = \frac{1}{m} \mathbf{F}_i = \frac{1}{m} \sum_{j \neq i} \mathbf{F}_{ij}(r_{ij})
\]

\[
\mathbf{F}_{ij} = -\frac{1}{r_{ij}} \frac{\partial u}{\partial r_{ij}} \mathbf{r}_{ij} = \frac{24\epsilon}{r_{ij}^6}[2(\sigma/r_{ij})^6 - (\sigma/r_{ij})^8] \mathbf{r}_{ij} \quad \text{for LJ 6–12}
\]

**Verlet**

\[
\mathbf{r}_i(t + \delta t) = -\mathbf{r}_i(t - \delta t) + 2\mathbf{r}_i(t) + \frac{\delta t^2}{m} \sum_{j \neq i} \mathbf{F}_{ij} + \theta(\delta t^4)
\]

\[
\mathbf{\ddot{r}}_i(t) = \frac{1}{2m} [\mathbf{r}_i(t + \delta t) - \mathbf{r}_i(t - \delta t)] + \theta(\delta t^2)
\]

Introduce dimensionless variables \( \mathbf{r}^*_i = \mathbf{r}_{ij}/\sigma \) etc.:  

\[
\mathbf{r}^*_i(t + \delta t) = -\mathbf{r}^*_i(t - \delta t) + 2\mathbf{r}^*_i(t) + 24\delta t^2 \left( \frac{m\sigma^2}{\epsilon} \right) \sum_{j \neq i} [2(\mathbf{r}^*_i)^{-14} - (\mathbf{r}^*_i)^{-8}] \mathbf{r}^*_i
\]

Now, define

\[
\tau = \left( \frac{m\sigma^2}{\epsilon} \right)^{1/2} \quad \text{“L-J 6–12 time constant”}
\]

For argon,

\[
\begin{align*}
\sigma &\approx 0.34 \text{ nm}, \\
\epsilon/k_B &= 120 K \\
m &= 6.7 \times 10^{-23} g
\end{align*}
\]

\( \Rightarrow \tau \sim 1.7 \times 10^{-12} \) s “picosecond” timescale. Finally, non-dimensionalize the time and velocity according to

\[
t^* = t/\tau, \quad \mathbf{v}^* = \mathbf{r} \cdot t/\sigma.
\]

Now,
1. \[
    r_i^*(t^* + \delta t^*) = 2r_i^*(t^*) - r_i^*(t^* - \delta t^*) \\
    + 24(\delta t^*)^2 \sum_{j \neq i} [2(r_{ij}^*)^{-8}] r_{ij}^*
    \]

2. \[
    v_i^*(t^*) = \frac{1}{2\delta t^*}[r_i^*(t^* + \delta t^*) - r_i(t^* - \delta t^*)]
    \]

For accuracy, we need \(\delta t^* \ll 1\), typically \(\delta t^* \approx 0.01\).

Recall that Newton’s equations conserve energy, so as described, this is a microcanonical algorithm. We can check accuracy by the extent of energy conservation.

Now, how to get thermo properties?

\[
    E = K + U
\]

and

\[
    \langle K \rangle = \frac{1}{2} \sum_i m v_i^2 = \frac{3}{2} N k_B T
\]

Measuring energies in units of \(\epsilon\), \(T^* = k_B T / \epsilon\)

\[
    \langle K^* \rangle = \frac{1}{2} \sum_i (v_i^*)^2 = \frac{3}{2} N T^*
\]

or \[T^* = \left( \frac{1}{3N} \sum_i (v_i^*)^2 \right)\] use to calculate temperature.

Now

\[
    U^* = \sum_{i < j} u(r_{ij}^*)
\]

\[
    E^* = \langle E^* \rangle = \langle K^* \rangle + \langle U^* \rangle
\]

\[
    E^* = \frac{3}{2} N T^* + \sum_{i < j} 4[(r_{ij}^*)^{-12} - (r_{ij}^*)^{-6}]
\]

should remain constant.

From the “virial equation”:

\[
    \frac{\partial \rho}{\partial \rho} = 1 - \frac{1}{6} \beta \rho \int dr g(r) r \frac{du}{dr_{12}}
    = 1 - \frac{1}{6} \beta \rho \frac{1}{N} \int dr_1 \int dr_2 \rho^{(2)}(r_1, r_2) r_{12} \frac{du}{dr_{12}}
\]

but \(\rho^{(2)} = \sum_{i \neq j} \delta(r_1 - r_i) \delta(r_2 - r_j)\)

\[
    \frac{\partial \rho}{\partial \rho} = 1 - \frac{1}{6} \beta \rho \left( \sum_{i < j} \sum_{j \neq i} w(r_{ij}) \right)
\]

where \(w\) is the virial function \(r \frac{du}{dr} \equiv w(r)\).

In dimensionless units: \(w^* = w / \epsilon\)
\[ Z^* \equiv \frac{P^*}{T^* \rho^*} = 1 - \frac{1}{3} \frac{1}{NT^*} \left( \sum_{i<j} \sum w^*(r_{ij}^*) \right) \]

\[ w^*(r^*) = -24[2(r^*)^{-12} - (r^*)^{-6}] \] for EOS

Let’s try this out in a Mathematica notebook — “demo2.nb”

Thus, we have seen how a basic NVE MD algorithm and simulation works. We leave the subject with a few comments:

1. The MD equations are stiff at (close approach of particles) and numerically unstable. Particle trajectories are not computed accurately (e.g., different \( \delta t \to \) trajectories exponentially different in time) and chaos ensues. However, this is what we want in order to sample constant \( E \) phase space! All that matters is that we conserve \( E \) accurately along the overall trajectory.

2. Potential truncation and neighbor lists can reduce the force calculation work from \( \frac{1}{2} N(N-1) \sim N^2 \) to \( \theta(N) \) for short-ranged potentials.

3. Long-ranged potentials (e.g., Coulomb \( \sim \frac{1}{r} \)) are difficult. Ewald sums and other specialized techniques have been developed.

4. Other ensembles can also be simulated, e.g., methods for NVT and NPT MD have been devised. You can read about these in the Allen & Tildesley reference.

Lecture 9

MC Method

Sample configuration states \( i \leftrightarrow r^N(i) \) from Boltzmann distribution \( P(i) = e^{-\beta U(i)}/\sum e^{-\beta U(i)} \) “importance sampling” \( \Rightarrow \) aves \( \langle f(r^N) \rangle = \frac{1}{N} \sum_{i=1}^{L} f(i) \).

How to do sampling? Generate “Markov chain” of states from

\[
\frac{\partial}{\partial r} P(i, t) = \sum_{i \neq 1} W_{ij} P(j, t) - \sum_{i \neq 1} W_{ji} P(i, t)
\]

for proper s. state distribution:

\[ W_{ij} P(j) = W_{ji} P(i) \quad \text{“microscopic reversibility”} \]
C. The Monte Carlo Method

At its heart, the Monte Carlo Method is a method for evaluating integrals. To illustrate, imagine that we wanted to do a 1-D integral

\[ I = \int_{a}^{b} dx f(x). \]

Let \( \rho(x) \) be any probability distribution function, normalized such that \( \int_{a}^{b} dx \rho(x) = 1 \). Then,

\[ \left[ \frac{f(x)}{\rho(x)} \right] = \left\langle \frac{f(x)}{\rho(x)} \right\rangle_{\rho} \]

Now, suppose we choose a total of \( M \) random numbers \( x_i, i = 1, 2, \ldots, M \) from the distribution \( \rho(x) \), where \( a \leq x_i \leq b \). Then

\[ I \approx 1 + \sum_{i=1}^{M} f(x_i) \rho(x_i) + \theta(M^{-1/2}) \]

The simplest choice of \( \rho(x) \) is a uniform distribution \( \rho(x) = 1/(b-a) \):

\[ I \approx \frac{b-a}{M} \sum_{i=1}^{M} f(x_i) + \theta(M^{-1/2}) \]

which gives a simple scheme for evaluating integrals. This “Monte Carlo” scheme is poor for low-dimensional integrals. The errors are

\[ \text{error (MC)} \sim \theta(M^{-1/2}) \]

while a “deterministic” quadrature method, e.g., trapezoidal rule, has errors that are much smaller:

\[ \text{error (TR)} \sim \theta(M^{-2}) \]

Consider now evaluating the canonical configuration partition function

\[ Q_c = \int dx^N e^{-\beta U(r^N)}. \]

This is a 3N-dimensional integral. If we used a deterministic method with say \( M \sim 10 \) points in each dimension, this would require \( 10^{3N} \) function evaluations, which is enormous! Instead, we might try the MC method, using a uniform distribution of points in \( r^N \) space:

\[ Q_c \approx V^N M \sum_{i=1}^{M} e^{-\beta U(r^N(i))} \]
This requires the generation of $M \cdot 3N$ random numbers, $r^N(i)$, and $M$ function evaluations, which clearly scales much better with $N$!

The problem with the method just described is that we give equal weight to configuration points $r^N(i)$ where $e^{-\beta U}$ is nearly zero (e.g., a configuration where particles overlap), as to points where $e^{-\beta U}$ is $\theta(1)$. Thus, the method is inefficient because we spend a lot of time generating configurations that contribute little to $Q_c$.

We can make the scheme more efficient by using “importance sampling”.

Suppose we wanted to calculate the ensemble average (in NVT) of some property $f(r^N)$:

$$\langle f(r^N) \rangle = \int d_{r^N} e^{-\beta U(r^N)} Q_c f(r^N) = \int d_{r^N} P(r^N) \left( \frac{P(r^N) f(r^N)}{\rho(r^N)} \right) \approx \frac{1}{M} \sum_{i=1}^{M} \frac{P(i) f(i)}{\rho(i)}$$

as before. Now, suppose that our random configurations $r^N(i)$ are chosen not from a uniform distribution, but from one with the canonical pdf:

$$\rho(r^N) = P(r^N).$$

Then,

$$\langle f(r^N) \rangle \approx \frac{1}{M} \sum_{i=1}^{M} f(r^N(i)).$$

This choice gives a simple formula for calculating averages and is the original choice by Metropolis and coworkers. The drawback of this “importance sampling” is that we need a scheme for sampling the $M$ configurations $r^N(i)$ from $\rho = P$.

The method for sampling from the pdf $P(r^N)$ is to invent a “fictitious” stochastic dynamics that has as its steady state, $P(r^N)$. To simplify the notation, let “$i$” stand for a particular 3N-configuration space point $r^N(i)$. Then,

$$P(i) = \frac{e^{-\beta U(i)}}{\sum_i e^{-\beta U(i)}}$$

is the PDF we would like to sample. Consider the following stochastic dynamics, known as a “master equation”:

$$\frac{\partial}{\partial t} P(i, t) = \sum_{j(i) \neq i} W_{ij} P(j, t) - \sum_{j(i) \neq i} W_{ji} P(i, t)$$

“gain” “loss”
where $P(i, t)$ is the probability of observing state “$i$” at time $t$. $W_{ij}$ is a matrix that gives the transition probabilities per unit time of a state change from $j$ to $i$. Because of the probabilistic nature of the entities in this equation:

$$\sum_i P(i, t) = 1$$

and the RHS is consistent with this. However, we have considerable freedom in selecting the $W_{ij}$ and, hence, defining a dynamical model. In particular, we would like this dynamics to have the Boltzmann $P(i)$ as its steady-state solution. For consistency, this requires for $t \to \infty$:

$$\frac{\partial}{\partial t} P(i, t) = 0 = \sum_{j \neq i} W_{ij} P(j) - \sum_{j \neq i} W_{ji} P(i).$$

One way, but not the only way, for the RHS to be zero is to demand “microscopic reversibility”:

$$W_{ij} P(j) = W_{ji} P(i).$$

This gives a constraint on the ratios of the corresponding off-diagonal elements of $W$:

$$\frac{W_{ij}}{W_{ji}} = \frac{P(i)}{P(j)} = e^{-\beta U(i)} e^{-\beta U(j)} = e^{-\beta |U(i) - U(j)|} \equiv e^{-\beta \Delta_{ij}}, \quad \Delta_{ij} \equiv U(i) - U(j)$$

A suitable model is thus:

$$W_{ij} = \alpha \begin{cases} 1, & \Delta_{ij} < 0 \\ e^{-\beta \Delta_{ij}}, & \Delta_{ij} > 0 \end{cases}$$

where $\alpha$ is some “rate” of making transitions.

We now have rules for making transitions between states (configurations in a fluid), such that the states are Boltzmann distributed. The Metropolis MC Method is one implementation of these rules:

1. Choose a particle (from among the $N$) at random. Call this particle “$k$”. State “$i$” is the state where this particle is at its original position, $r_k(i)$.

2. General a random, “trial” displacement of particle $k$, $\delta r = (\delta x, \delta y, \delta z)$, where $\delta x, \delta y, \delta z$ are selected from a uniform distribution over $(-\delta r_{max}, \delta r_{max})$. $\delta r$ is thus uniformly distributed in a cube of side $2\delta r_{max}$ about $r_k(i)$.

3. Calculate $\Delta \equiv \Delta_{i+1,i} = U(i + 1) - U(i)$, where $U(i + 1)$ is the pot energy if particle $k$ were displaced to $r_k(i + 1) = r_k(i) + \delta r$. Note that this only
requires summing \( \theta(N-1) \) terms:

\[
\Delta = \sum_{\ell \neq k} [U(|r_{\ell k}(i+1)|) - U(|r_{\ell k}(i)|)]
\]

4. If \( \Delta < 0 \) (downhill in energy): \( \rightarrow \) accept the move with probability 1.

If \( \Delta > 0 \) (uphill in energy): \( \rightarrow \) accept the move with probability \( 3^{-\beta \Delta} \).

To do the latter, choose a random number \( x \) over \((0,1)\). If \( x < e^{-\beta \Delta} \), accept; otherwise reject. If accepted, then \( r_k(i+1) = r_k(i) + \delta r \). If move is rejected, then \( r_k(i+1) = r_k(i) \).

5. Repeat steps 1-4 to generate a “Markov chain” of \( M \) configurational states. Averages from:

\[
\langle f(r^N) \rangle \approx \frac{1}{M} \sum_{i=1}^{M} f(r^N(i))
\]

Some comments about the Metropolis method:

i) \( \delta r_{\text{max}} \) is typically chosen and adjusted so that roughly 50% of the trial moves are rejected.

ii) Often people move particles \textit{sequentially}, rather than randomly. This is faster since less random number generation. \( N \) trial moves (1 per particle) \( \equiv \) “MC cycle”. Work is comparable to one MD time step.

iii) More rapid equilibration can be achieved by \textit{multi-particle} moves rather than single-particle moves. This is particularly advantageous on a \textit{parallel computer}!

iv) MC is easily adapted to NPT and \( \mu V T \), \( \mu P T \) ensembles.

Let’s now try this out on the LJ 6–12 fluid via “demo3.nb”:

---

**Lecture 11**

Recap:

- MF theories can be unified via Landau expansions.
• Give “chemical” crit. exponents independent of $d$.
• Notion of universality.