LECTURE #12
STAT MECH CORE

October 29, 2018

Goal: to study phenomena such as phase transitions for systems of N interacting particles when N is large.

Specifically, today we will focus on developing numerical Monte Carlo (MC) methods to study the XY model.

Main reference:
Newman & Barkema, "Monte Carlo Methods in Statistical Physics"

Outline for today:

- Expectation values of observables
- Classical MC methods
  - ↳ Importance sampling
  - ↳ Markov chain MC
  - ↳ Metropolis algorithm, Single-spin-update algorithm
- Data analysis practices: equilibration, measurement correlations

Expectation values of observables in classical stat. mech.

In the canonical ensemble, our system is in thermal equilibrium with a heat bath at fixed T.

The expectation value of a quantity Q is

$$\langle Q \rangle = \sum_\mu Q_\mu \rho_\mu$$

$$\rho_\mu = \frac{1}{Z} e^{-\beta E_\mu}$$

$$Z = \sum_\mu e^{-\beta E_\mu}$$

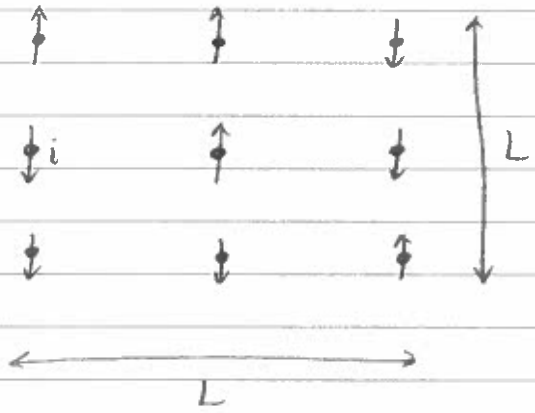
where:

- $\beta = \frac{1}{k_B T}$ (we set $k_B = 1$)
- μ represents a microstate of the system
- E_μ is the energy of state μ

We would like to study the behaviour of expectation values in the limit where the # of particles $N \to \infty$.

For example, let's consider how to evaluate the sums over μ directly for a simple nearest-neighbour Ising model:

$$H = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j \qquad \text{with } \sigma_i = \pm 1 \text{ on each site } i$$



$N = L^d$

exact solutions are available in $d=1$ and 2

The total # of accessible states is

$$2^N = 2^{(L^d)}$$

For $N = 25$, the fastest algorithms for computing the sums $\sum_\mu$ require around a minute. Since this time scales as 2^N:

N	time
25	~ 1 min.
30	~ 30 min.
35	~ 20 hours
40	~ 20 days
50	~ 60 years
80	~ 7×10^{10} years ← longer than the age of the universe!

We see that if we try to evaluate Σ_μ directly, we are limited to $L = 7$ or 8 in $d = 2$ for the Ising model.
Would be even worse for the XY model!
Recall: $H_{XY} = -J \sum_{\langle ij \rangle} \cos(\theta_i - \theta_j)$ $\quad 0 \leq \theta_i < 2\pi$

Classical MC Methods

↳ addresses the exponential complexity by considering only M states
$\mu_1, \mu_2, \ldots, \mu_M$
selected at random

For example, if we choose the M states μ_m $(1 \leq m \leq M)$ from all possible states with equal probability ("flat sampling"), then we can estimate $\langle Q \rangle$ from:

$$\langle Q \rangle \approx Q_M = \frac{\sum_{m=1}^{M} Q_{\mu_m} e^{-\beta E_{\mu_m}}}{\sum_{m=1}^{M} e^{-\beta E_{\mu_m}}}$$

At high T ($\beta \to 0$)	At low T ($\beta \to \infty$)
Works well!	Fails!
	Only a small subset of states have high probability in the original Σ_μ and it is very unlikely that we choose them!

More generally, if the states μ_m are taken from a probability distribution W_μ, then the estimator for $\langle Q \rangle$ is:

$$Q_M = \frac{\sum_{m=1}^{M} Q_{\mu_m} W_{\mu_m}^{-1} e^{-\beta E_{\mu_m}}}{\sum_{m=1}^{M} W_{\mu_m}^{-1} e^{-\beta E_{\mu_m}}}$$

Importance Sampling

We would like to sample the more "important" (highly-probable) states more frequently.

In particular, if we sample from the Boltzmann distribution such that $W_\mu = \rho_\mu = e^{-\beta E_\mu}/Z$, then

$$Q_M = \frac{1}{M} \sum_{m=1}^{M} Q_{\mu_m}$$

But how do we choose states according to ρ_μ?

Markov Chain Monte Carlo

Idea: Given state μ_m, move to the next state μ_{m+1} according to the transition probability $T(\mu_m \to \mu_{m+1})$, where

- $T(\mu_m \to \mu_{m+1})$ depends only on states μ_m and μ_{m+1} (not on $\mu_1, \mu_2, \ldots, \mu_{m-1}$)
- the transition probabilities don't depend on time
- $\sum_\nu T(\mu \to \nu) = 1$

The resulting set $\{\mu_1, \mu_2, \ldots, \mu_M\}$ is called a Markov chain of states.

In order to reach states from the target distribution W_μ, we must design an algorithm that satisfies two conditions:

(1) Ergodicity: Given two states μ and ν, it must be possible to transition from μ to ν (with non-zero probability) through some chain of intermediate states

$$\mu \to \ldots \to \nu$$

(2) Detailed balance (DB): the rates of transition into and out of any state μ must be equal (otherwise W_μ would change with time):

$$\underbrace{\sum_\nu W_\nu T(\nu \to \mu)}_{\text{rate of transition into } \mu} - \underbrace{\sum_\nu W_\mu T(\mu \to \nu)}_{\text{rate of transition out of } \mu} = 0$$

One way to satisfy this condition is:

$$\boxed{W_\mu T(\mu \to \nu) = W_\nu T(\nu \to \mu)} \quad \text{DB}$$

So when $W_\mu = e^{-\beta E_\mu}/Z$:

$$\frac{T(\mu \to \nu)}{T(\nu \to \mu)} = e^{-\beta(E_\nu - E_\mu)}$$

We have freedom in how we choose the transition probabilities as long as this ratio stays fixed.

Let's introduce the notation:

$$T(\mu \to \nu) \equiv \underbrace{g(\mu \to \nu)}_{\text{selection probability}} \underbrace{A(\mu \to \nu)}_{\text{acceptance probability}}$$

- $g(\mu \to \nu)$: probability of proposing a move to state ν given initial state μ
- $A(\mu \to \nu)$: probability of accepting the proposed move $\mu \to \nu$

We want the acceptance probabilities to be as close to 1 as possible so that we sample as many different states as possible.

From DB: $$\frac{A(\mu \to \nu)}{A(\nu \to \mu)} = \frac{g(\nu \to \mu) W_\nu}{g(\mu \to \nu) W_\mu}$$

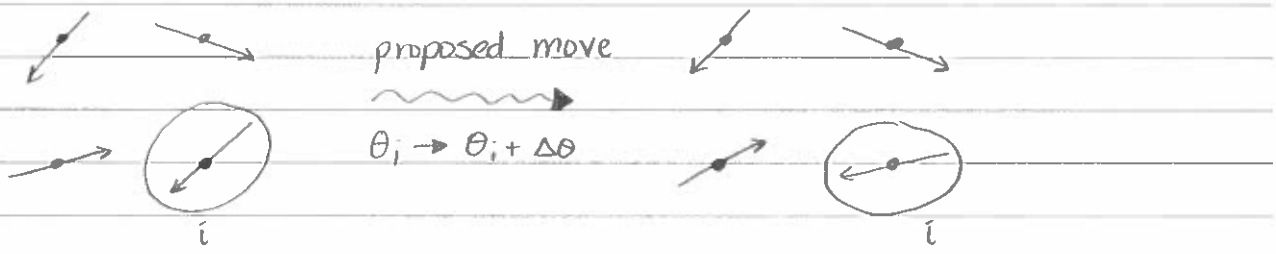
Metropolis Algorithm:

Ensure that the acceptance probability is equal to 1 for either $\mu \to \nu$ or $\nu \to \mu$:

$$A(\mu \to \nu) = \min\left(1, \frac{g(\nu \to \mu)}{g(\mu \to \nu)} \frac{W_\nu}{W_\mu}\right)$$

Single-Spin-Update Algorithm for the XY Model

Given a state μ_m of N spins (rotors) at step m of the Markov chain, propose a new state μ_{m+1} that differs from μ_m by a single spin rotation of the spin at some randomly chosen lattice site i.



where $\Delta\theta$ is chosen at random from a uniform distribution.

The selection probabilities are:

$$g(\mu \to \nu) = \begin{cases} \text{const.} & \text{if } \mu \text{ and } \nu \text{ differ by a single spin} \\ 0 & \text{otherwise} \end{cases}$$

Using the Metropolis algorithm:

$$A(\mu \to \nu) = \begin{cases} e^{-\beta(E_\nu - E_\mu)} & \text{if } E_\nu > E_\mu \\ 1 & \text{otherwise} \end{cases}$$

Single-spin-update Metropolis algorithm for XY:

1) Generate a random state μ_1 $(m = 1)$

REPEAT

2) Choose a site i of the lattice at random $(1 \leq i \leq N)$

3) Choose a random angle $\Delta\theta \in [-\pi, \pi)$ from a uniform distribution.

4) Calculate the energy difference ΔE associated with the update $\theta_i \rightarrow \theta_i + \Delta\theta$

5) Generate a random number $r \in [0, 1)$ from a uniform distribution.

6) If $\Delta E \leq 0$ OR $r < e^{-\beta \Delta E}$, accept the move to get μ_{m+1}.
Otherwise, reject the move $(\mu_{m+1} = \mu_m)$

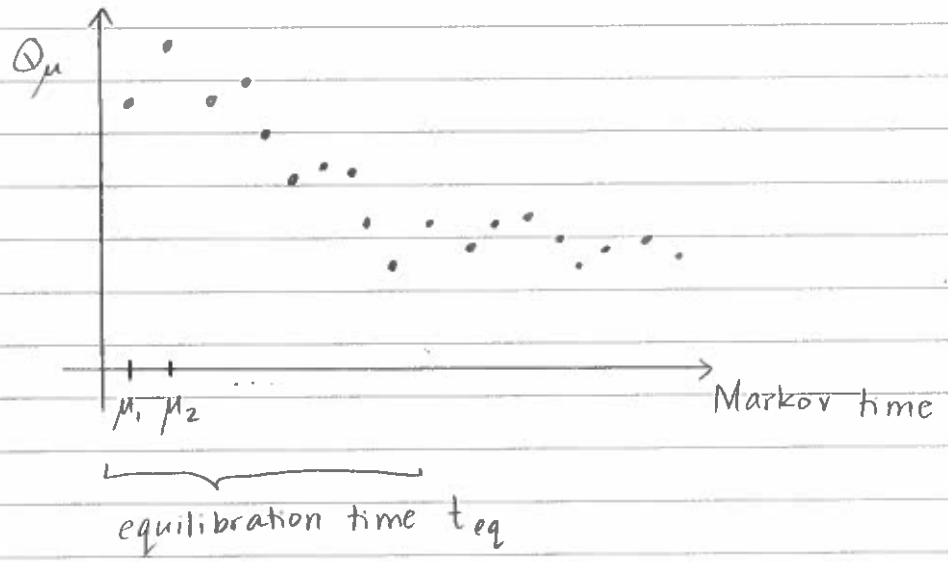
Equilibration

Recall that we can start our sampling from any state μ_1. However, we would like to (eventually) generate samples from the distribution $W_\mu = e^{-\beta E_\mu}/Z$.

In many cases, μ_1 and the subsequent $\mu_2, \mu_3, \ldots$ might have very low ρ_{μ_m} such that the probability of finding the system in these early states μ does not follow the desired distribution W_μ.

(For discussion of why we eventually approach the correct distribution, see Section 2.2.3 of Newman & Barkema.)

For example, if we plot some quantity Q_μ as a function of Markov time, we might find:



We don't want to use $\mu_1, \mu_2, \ldots, \mu_{t_{eq}}$ when we calculate our estimators:

$$Q_M = \frac{1}{M} \sum_{m=1}^{M} Q_{\mu_{t_{eq}+m}}$$

Measurement correlations

When calculating Q_M, we should ideally use sampled states μ_m that are statistically independent.

The Markov time required to get independent samples (the "autocorrelation time") depends on:

- the model
- the algorithm
- the temperature

(For explanation of how to calculate the autocorrelation time, see Section 3.3.1 of Newman & Barkema.)

For single-spin updates, μ_m and μ_{m+1} will be highly correlated. In practice, it can take a long time to get independent samples. As a result, we usually do a "sweep" of $\mathcal{O}(N)$ proposed updates before performing a measurement of Q_{μ_m} such that μ_m and μ_{m+1} are actually separated by $\mathcal{O}(N)$ single-spin updates.

Note: more efficient "cluster" algorithms that update more than one spin at a time

- ↳ Swendsen and Wang (1987)
- ↳ Wolff (1989)