Lecture 8 - Monte Carlo Sampling

Reminder: goal of stat mech to calculate average properties

$$\langle O \rangle = \int d\vec{x}\, O(\vec{x})\, P(\vec{x})$$

$\vec{x} = \{x_1 \ldots x_{3N}, p_1 \cdots p_{3N}\}$ or

$= \{x_1 \ldots x_{3N}\}$ if O doesn't depend on $\vec{p}$ and

$$P(\vec{x}, \vec{p}) = P(\vec{x})\, P(\vec{p})$$

eg, @ const NVT, usually

$$\mathcal{H}(\vec{x}, \vec{p}) = \sum_{i=1}^{3N} p_i^2/2m_i + U(\vec{x})$$

$$P(\vec{x}) = \int d\vec{p}\; P(\vec{x}, \vec{p}) = \frac{e^{-\beta U(\vec{x})}}{\int dx\, e^{-\beta U(x)}}$$

But, even if know $u(x)$, cannot compute $\langle O \rangle$ directly for most problems, high dimension

Idea: produce representative set of configurations $\{\vec{x}_t\}$ s.t. x_t distributed according to target distribution

More specifically, have correct relative weights. So eg for Boltzmann distribution

$$P(\vec{x}_j)/P(\vec{x}_i) = \frac{e^{-\beta u(\vec{x}_j)}}{e^{-\beta u(\vec{x}_i)}} = e^{-\beta[u(\vec{x}_j) - u(\vec{x}_i)]}$$

$$= e^{-\beta \Delta u_{ij}}$$

How can we do this? ① Going to generate a "Markov Chain"

This means a set of cfgs

$$\vec{x}_1 \rightarrow \vec{x}_2 \rightarrow \vec{x}_3 \longrightarrow \vec{x}_T$$

made by some rule where

$P(\vec{x}_t \rightarrow \vec{x}_{t+1})$ only depends on what $\vec{x}_t$ is (not $\vec{x}_1, \vec{x}_2 \ldots \vec{x}_{t-2}$)

② It can be proven that if our rule satisfies "detailed balance" then sampling will <u>converge</u>

"detailed balance" - each microscopic step follows equilibrium rule

$$P(x_t)\, P(x_t \rightarrow x_{t+1}) = P(x_{t+1})\, P(x_{t+1} \rightarrow x_t)$$

We do this by choosing the rule $P(\vec{x}_t \to \vec{x}_{t+1})$

This is actually made of two parts!

$$P(x_t \to x_{t+1}) = P_{gen}(x_t \to x_{t+1}) \cdot P_{acc}(x_t \to x_{t+1})$$

Plugging into D.B. (switch to x, y)

$$P(x) P_{gen}(x \to y) P_{acc}(x \to y) = P(y) P_{gen}(y \to x) P_{acc}(y \to x)$$

$$\Rightarrow P_{acc}(x \to y) = \underbrace{\frac{P(y)}{P(x)} \frac{P_{gen}(y \to x)}{P_{gen}(x \to y)}}_{r(x \to y) \text{ "rate"}} \cdot P_{acc}(y \to x)$$

How can we get this:

Metropolis Rule (1953)

$$P_{acc}(x \to y) = \min[1, r(x \to y)]$$

Why does this work?

either $r(x \to y) > 1$ or (a)

$r(y \to x) > 1$ lets try both (b)

a: $r(y \to x) = \frac{1}{r(x \to y)} < 1$

$$\frac{P_{acc}(x \to y)}{P_{acc}(y \to x)} = \frac{1}{[1/r(x \to y)]} = r(x \to y)$$

b: $r(y \to x) = 1$

$$\frac{P_{acc}(x \to y)}{P_{acc}(y \to x)} > r(y \to x)$$

Algorithm: start at x_t

(1) Propose x_{t+1} w/ prob $P_{gen}(x_t \to x_{t+1})$

(2) gen random # $r \in [0,1)$

if $r < P_{acc}(x_t \to x_{t+1})$:

move to x_{t+1}

if not:

duplicate x_t ie $x_{t+1} = x_t$

(3) back to 1

Let's look at $r(x \to y)$

$$r(x \to y) = \frac{P_{gen}(y \to x)}{P_{gen}(x \to y)} \cdot \frac{P(y)}{P(x)}$$

<u>Choose</u> P_{gen} &

$P(x)$ is target distr

So far Metropolis

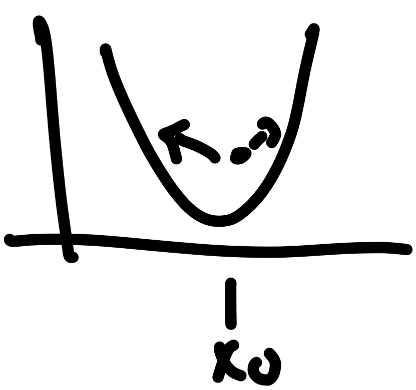
$$P(y)/P(x) = e^{-\beta \Delta U_{xy}}$$

What about this P_{gen}

Usually choose P_{gen} to be

Symmetric st $\dfrac{P_{gen}(x \to y)}{P_{gen}(y \to x)}$ cancel

Eg: Suppose we want to sample 1 particle in HO $U(x) = \frac{1}{2}kx^2$



Rule could be

$$x_{t+1} = x_t + \xi r_2$$

$r_2 \in (-1, 1)$ uniform random

ξ is max step size

Reverse move equally likely

In this case Metropolis rule simplifies to $P_{acc} = \min[1, e^{-\beta \Delta u}]$

Accept always if energy goes down
Accept sometimes if energy goes up

Tune move rule to get accept prob ~ 0.25–0.5
trade off between efficiency and exploration, eg size of ε
bigger move, more likely to increase $\mathcal{E}$ more

Reminder: even if initial cfg not good, will eventually sample from target P (if ergodic)

Do we have to use metropolis?
There are other rules:
Eg: Glauber rule

$$P_{acc}(x \to y) = \frac{1}{2}\left[1 - \tanh\left(\frac{\beta \Delta E_{xy}}{2}\right)\right]$$

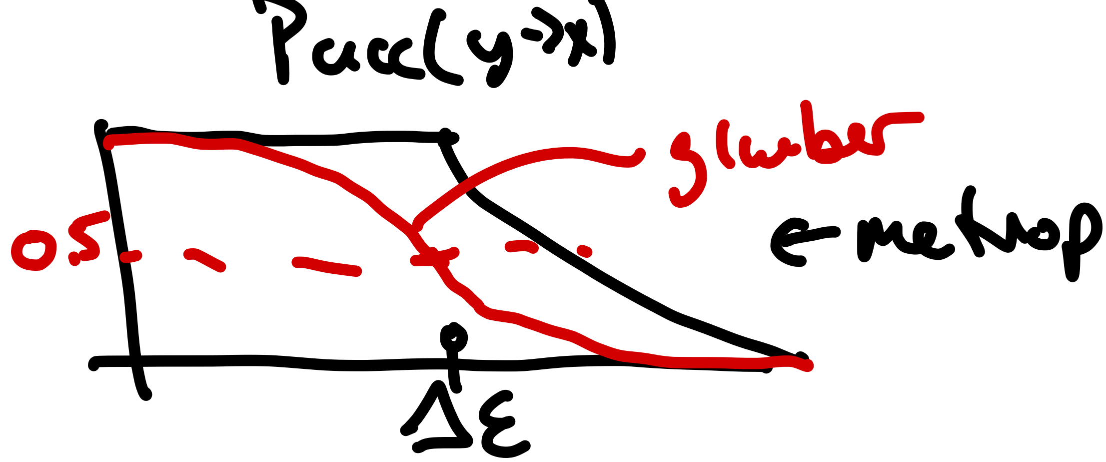
$$= \frac{e^{-\beta \Delta E_{xy}/2}}{e^{\beta \Delta E_{xy}/2} + e^{-\beta \Delta E_{xy}/2}}$$

why?

$$\Delta E_{xy} = E_y - E_x$$

$$\Delta E_{yx} = E_x - E_y = -\Delta E_{xy}$$

So
$$\frac{P_{acc}(x \to y)}{P_{acc}(y \to x)} = e^{-\beta \Delta E_{xy}} \quad \checkmark$$



Why MC & why Not:

1. Easy
2. can choose smart MC moves which sample space much faster

Why not:

1. Not real dynamics (if you care)
   Only static properties right (usually)
2. Usually only small moves accepted
   Eg in liquid, move 1 MC not all at once as in MD