# Computational Biophysics

Pranay Vure

January 21, 2026

## 1 The Boltzmann Distribution

The Boltzmann distribution describes the probability $P(x)$ that a system will be in a specific state $x$ (conformation) as a function of that state's potential energy $U(x)$ and the temperature $T$.

$$P(x) = \frac{1}{Z} e^{-U(x)/k_B T} \tag{1}$$

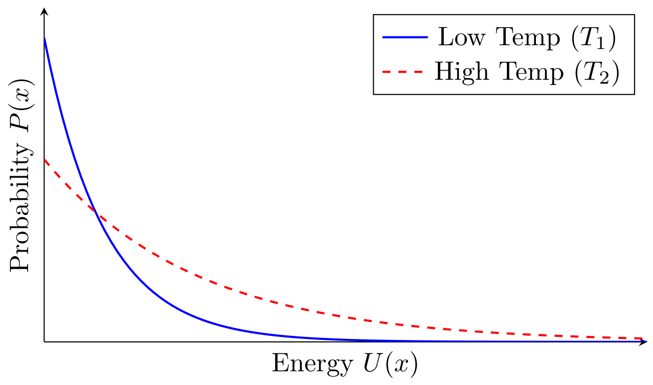

Figure 1: **The Boltzmann Distribution.** Lower energy states are exponentially more probable.

### Variable Definitions

- $x$: The state of the system (e.g., coordinates of atoms in a protein).
- $U(x)$: The potential energy of state $x$.
- $k_B$: The Boltzmann constant.
- $T$: The temperature in Kelvin.
- $k_B T$: The thermal energy scale.
- $Z$: The Partition Function (normalization constant).

### The Partition Function ($Z$)

To ensure that the sum of all probabilities equals 1 (100%), we divide by $Z$, which is the sum of the Boltzmann factors for all possible states:

$$Z = \sum_{\text{all } x} e^{-U(x)/k_B T} \tag{2}$$

## 2 Energy vs. Probability: Entropy as an Important Factor

While $P(x)$ depends on Potential Energy $U(x)$, observable probability is often described by the Gibbs Free Energy $G$, which accounts for Entropy ($S$).

$$G = H - TS \tag{3}$$

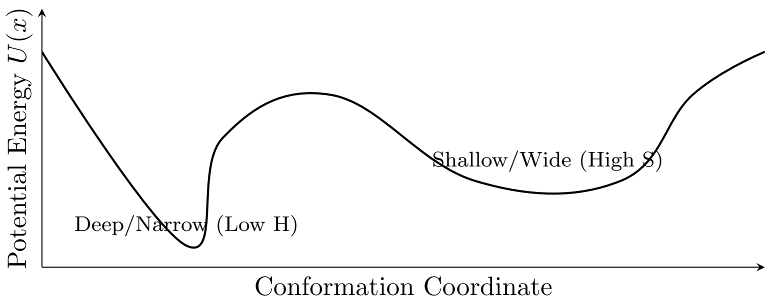

Figure 2: **Enthalpy vs. Entropy.** A wide basin (right) often dominates at high temperatures due to higher entropy, even if it has higher energy.

- **Deep Basins (Low Enthalpy $H$):** Favorable due to strong interactions (low $U$).
- **Wide Basins (High Entropy $S$):** Favorable due to the high number of available microstates.

**Key Insight:** A state with higher Potential Energy $U(x)$ can be *more probable* than a lower energy state if the basin is significantly wider (higher entropy), especially at high temperatures. However, a state with **lower Free Energy (G)** is *always* more probable, as G accounts for both potential energy and entropy.

## 3 Molecular Dynamics (MD)

Molecular Dynamics is **deterministic**. It simulates the actual physical time evolution of the system by solving Newton's laws of motion.

### 1. Newton's Equation of Motion

For every atom $i$, the force $F_i$ is derived from the gradient (slope) of the potential energy landscape $U(x)$:

$$F_i = -\nabla U(x_i) = m_i a_i \tag{4}$$

### 2. Time Integration

MD moves the system forward in small time steps ($\Delta t$). Unlike Monte Carlo, MD includes **momentum**. The position at the next time step is often calculated using the Velocity Verlet algorithm:

$$x(t+\Delta t) = x(t) + v(t)\Delta t + \frac{F(t)}{2m}\Delta t^2 \tag{5}$$

***Physical Takeaway:*** *MD produces a "movie" of the molecule. It is used when we need to know kinetics (how fast something happens) or the pathway of a reaction.*

## 4 Monte Carlo Simulation (MCMC)

Since calculating $Z$ explicitly is computationally impossible for complex systems, we use Markov Chain Monte Carlo (MCMC) to sample the landscape stochastically without calculating forces.

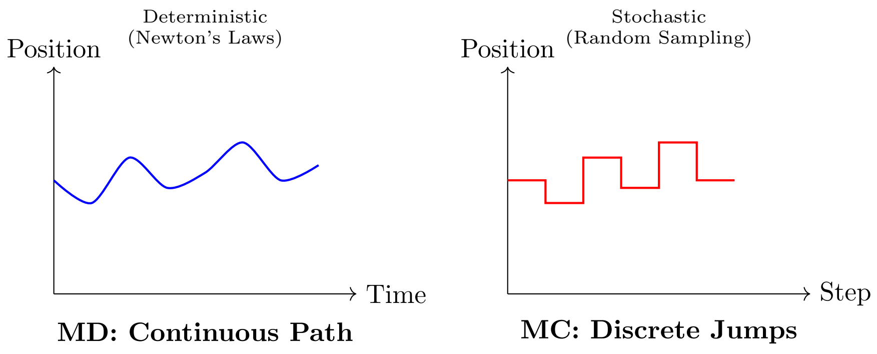

Figure 3: **Trajectory Comparison.** MD (left) produces a smooth, time-continuous trajectory useful for kinetics. MC (right) produces a series of discrete states useful for calculating equilibrium averages.

### 1. The Proposal Step (The "Walk")

The system evolves by proposing a small random change to the current coordinates:

$$x_{\text{new}} = x_{\text{old}} + \delta \cdot \xi \tag{6}$$

- $x_{\text{old}}$: Current state.
- $\delta$: Maximum step size.
- $\xi$: A random number drawn from a distribution (e.g., uniform on $[-1, 1]$).

### 2. Stationarity

Over a sufficiently long simulation ($N$ steps), the average of an observable converges to the equilibrium ensemble average:

$$\langle x \rangle = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{7}$$

***Physical Takeaway:*** *This equation is the bridge between the computer simulation and the real world. It relies on the **Ergodic Hypothesis**: if we run the simulation long enough, the simple average of our random steps becomes identical to the true physical average of the system (as if we measured it in a lab).*

## 5 The Metropolis Algorithm

To decide whether to accept or reject a proposed move in Monte Carlo (from state $i$ to state $j$), we use the Metropolis criterion. This ensures the simulation obeys the Boltzmann distribution.

The acceptance probability $A(i \to j)$ is defined as:

$$A(i \to j) = \min\left(1, \frac{P(j)}{P(i)}\right) \tag{8}$$

Substituting the Boltzmann distribution $P(x) \propto e^{-U(x)/k_B T}$:

$$\frac{P(j)}{P(i)} = \frac{\frac{1}{Z} e^{-U(j)/k_B T}}{\frac{1}{Z} e^{-U(i)/k_B T}} = e^{-(U(j)-U(i))/k_B T} \tag{9}$$

Letting $\Delta U = U(j) - U(i)$, the final algorithm is:

$$A(i \to j) = \min\left(1, e^{-\frac{\Delta U}{k_B T}}\right) \tag{10}$$

### Interpretation

1. **If $\Delta U < 0$:** The new state is lower energy. The exponential term is $> 1$. We accept the move with probability 1.
2. **If $\Delta U > 0$:** The new state is higher energy. We accept the move with probability $e^{-\Delta U/k_B T}$. (This allows the system to escape local energy minima).

***Physical Takeaway:*** *The Metropolis criterion acts as a "smart filter." It takes the random noise of the Proposal Step and sculpts it into the correct physical distribution. It ensures that while the system prefers low energy, it is still allowed to visit high-energy states occasionally, exactly as nature does.*

## 6 Lattice Models: The HP Model

For large systems or long timescales, atomistic detail is too expensive. We use simplified **Coarse-Grained Models** like the 2D HP Lattice Model.

### Model Rules

1. **Beads:** Amino acids are simplified to two types: **H** (Hydrophobic) and **P** (Polar).
2. **Lattice:** The chain lives on a discrete grid (integers only).

3. **Energy Function:** Energy decreases only when hydrophobic beads cluster together to avoid water (topological contacts).

$$E = -\epsilon \sum_{\text{H-H contacts}} 1 \tag{11}$$

(Where a contact is a non-bonded pair of H beads occupying adjacent lattice sites).

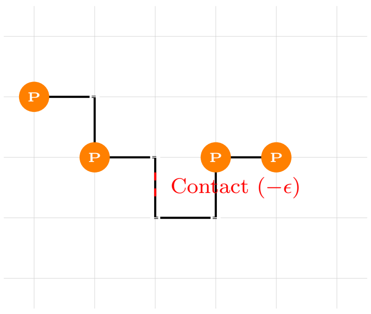

Figure 4: **2D HP Lattice Model.** Blue squares are Hydrophobic (H), Orange circles are Polar (P). The red dashed line represents a favorable non-bonded contact between H beads that lowers the energy.

## 7 Advanced Sampling: Pivot Moves

In lattice simulations, moving one bead at a time often leads to "kinetic traps" where the protein cannot move without breaking the chain. To solve this, we use **Global Moves**.

### The Pivot Algorithm

Instead of wiggling a single atom, we select a random bead as a "pivot" point and rotate the entire rest of the chain around it (e.g., by 90° or 180°).

- **Efficiency:** A single pivot move can radically change the global structure, allowing the system to explore the landscape much faster than local moves (known as critical slowing down).
- **Self-Avoidance:** After a rotation, we must check if the new chain overlaps with itself. If it does, the move is rejected immediately ($P = 0$).