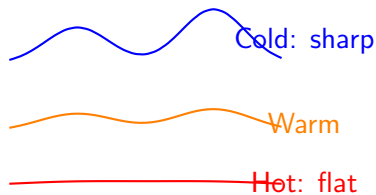
Parallel Tempering MCMC

The Problem and Solution

The Problem:

- Standard MCMC gets stuck in local modes
- Can't explore separated peaks
- Dilemma: small steps (stuck) vs. large steps (rejected)



The Solution: Temperature Ladder

- Run N chains targeting π^{γ_n}
- $0 < \gamma_1 < \cdots < \gamma_N = 1$
- Hot ($\gamma \approx 0$): Explores freely
- Cold ($\gamma = 1$): Exploits peaks

Key Insight Different temperatures see the same distribution differently - hot chains explore, cold chains exploit

The Temperature Mechanism

Key Idea: Tempered Distributions

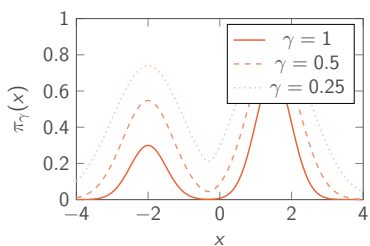
Define a family of distributions indexed by inverse temperature $0 < \gamma_1 < \gamma_2 < \ldots < \gamma_N = 1$:

$$\pi_{\gamma_n}(x) \propto \pi(x)^{\gamma_n}$$

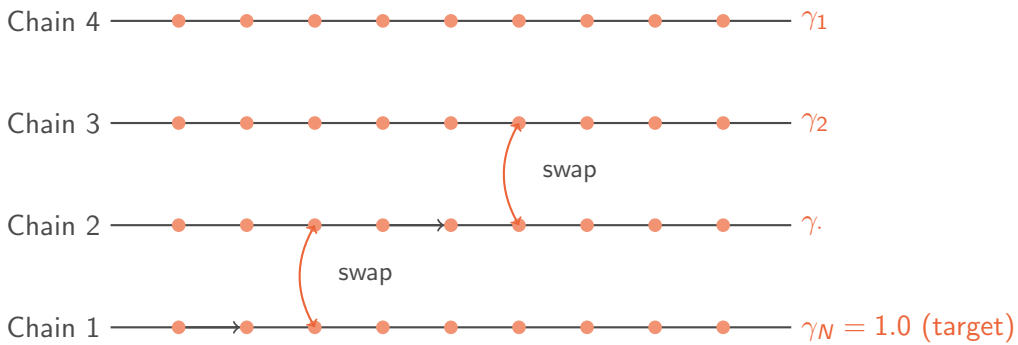
where $n = 1, \ldots, N$ and $\pi(x)$ is our target distribution.

Properties:

- $\gamma_N = 1$: Original target distribution
- $\gamma_1 \approx 0$: Uniform, explores broadly



Parallel Chains Architecture



Two Types of Moves

1. **Within-chain updates**: Standard MCMC at each temperature
2. **Between-chain swaps**: Exchange states between adjacent temperatures

Parallel Tempering Prerequisites

Just for reference

- **Target Distribution**: $\pi(x)$
- **Proposal Distribution**: For each tempered chain $q(x'|x)$ - could potentially depend on temperature
- **Initialization**: $x_n^{(0)}$ for $n = 1, \ldots, N$
- **Standard MCMC Step**: Any MCMC kernel (e.g., RWM, MALA)
- **Number of Chains**: N
- **Number of Samples per Chain**: T
- **Temperature Schedule**: $\{\gamma_n\}_{n=1}^N$ with $\gamma_N = 1$

Parallel Tempering Algorithm

Just for reference

Algorithm 1 Parallel Tempering MCMC

for $t = 1$ **to** T **do**
  for all $n \in \{1, \ldots, N\}$ **in parallel do**
    Sample $x_n^{(t)}$ using a standard MCMC step targeting π^{γ_n}
  end for
  $k \sim \text{Uniform}\{1, \ldots, N-1\}$
  $\alpha_{\text{swap}} = \min\left\{1, \left(\frac{\pi(x_{k+1}^{(t)})}{\pi(x_k^{(t)})}\right)^{\gamma_k - \gamma_{k+1}}\right\}$
  Swap $(x_k^{(t)}, x_{k+1}^{(t)})$ with probability α_{swap}
end for
return $\{x_N^{(t)}\}_{t=1}^{T}$

Swap Acceptance Ratio

Propose swap: Exchange states $x_{k_1} \leftrightarrow x_{k_2}$ between chains k_1 and k_2

$$\alpha_{\text{swap}} = \frac{P(\text{proposed state})}{P(\text{current state})} = \min\left\{1, \frac{\pi^{\gamma_{k_1}}(x_{k_2}) \cdot \pi^{\gamma_{k_2}}(x_{k_1})}{\pi^{\gamma_{k_1}}(x_{k_1}) \cdot \pi^{\gamma_{k_2}}(x_{k_2})}\right\}$$
$$= \min\left\{1, \left(\frac{\pi(x_{k_2})}{\pi(x_{k_1})}\right)^{\gamma_{k_1} - \gamma_{k_2}}\right\}$$

This mimics the Metropolis-Hastings acceptance ratio for swapping states between two chains. $\alpha = \min(1, [\text{Prob of proposed state}]/[\text{Prob of current state}])$

How swapping works

Convergence of Parallel Tempering

The algorithm has joint distribution: $\prod_{i=1}^{N} \pi(x_i)^{\gamma_i}$

Two Types of Moves Preserve This Distribution

Within-chain moves:
By standard MCMC theory, these moves satisfy detailed balance w.r.t. π_i Therefore preserve the marginal distribution for each chain.

Between-chain swaps:
This satisfies detailed balance for the joint distribution:
$\pi_{\text{before}} \times P(\text{swap } i \leftrightarrow j) = \pi_{\text{after}} \times P(\text{reverse swap } j \leftrightarrow i)$
The swap acceptance probability rate is specifically chosen so that the flow of probability mass from configuration "before" to "after" exactly equals the reverse flow, thereby maintaining the joint equilibrium distribution joint

Irreducibility Argument

The combined system is irreducible because:

- Hot chains (small γ) can easily explore the entire space
- Swaps allow states to "flow" between temperatures
- Any state reachable in the hot chain can eventually reach the cold chain

Choosing Parallel Tempering Hyperparameters

- Briefly mention that PT requires N times more computation than standard MCMC.
- Number of chains (N): Typically 4-20 chains balance computational cost with mixing efficiency. More chains provide better exploration but increase computational burden proportionally.
- How to set the temperature schedule? (geometric spacing is common). The hottest chain should be close to uniform.
- Swap frequency: Swaps are often attempted every 1-10 MCMC iterations. Too frequent swaps waste computation on rejection, while too infrequent swaps reduce the benefit of parallel tempering.
- Rule of thumb: Aim for adjacent-chain swap acceptance rates around 20-40%. If too low, add more intermediate temperatures; if too high ($>60\%$), you may have redundant chains that could be removed to save computation.