TWEAG

Multimodal MCMC with Replica Exchange and TensorFlow Probability

www.tweag.io

Your hosts

Simeon (presentation)

- background in computational biology
- Data Scientist at Tweag since 2019
- lives in Paris

Tweag I/O

Software innovation lab and consultancy based in Paris with employees all around the world and a strong focus on open-source software.

We specialize in

- software engineering, with a focus on functional programming
- DevOps, with a focus on reproducible software systems and builds
- data science

Key industries: finance, biotech, automotive

Need help with your project? Want to work with us?

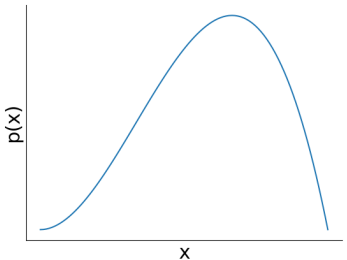
www.tweag.io

hello@tweag.io

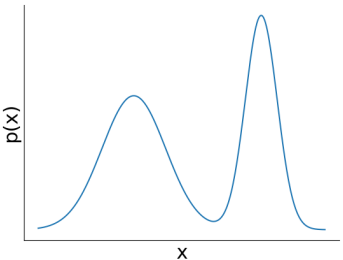
What you’re in for

Multimodality

Unimodal distribution:



Bimodal distribution:



Multimodality: common occurences

mixture models

For GMM parameterized by mixture weight θ and μ_1, μ_2, σ:

$$p(\theta, \mu_1, \mu_2, \sigma|y) = p(1-\theta, \mu_2, \mu_1, \sigma|y)$$

models invariant under reflections

e.g., biomolecular structure determination:

$$L(\mathbf{x}_1, \ldots \mathbf{x}_N) = p(D|\mathbf{x}_1, \ldots \mathbf{x}_N) = L(\ldots, |\mathbf{x}_i - \mathbf{x}_j| \ldots)$$

latent variable models

e.g., probabilistic PCA: symmetry w.r.t. rotation of latent variable coordinates

Refresher: Metropolis-Hastings

Markov chain

Random process with

$$p(x_{i+1}|x_i, x_{i-1}, \ldots, x_1) = p(x_{i+1}|x_i)$$

$\rightarrow$ a Markov chain has no "memory"

In some conditions: converges to a unique invariant distribution $\pi(x)$

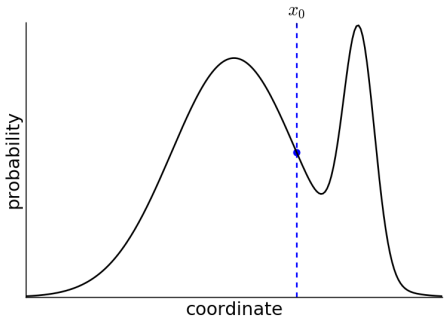
Metropolis-Hastings algorithm

Construct Markov chain with invariant distribution $\pi(x) = p(x)$:

1. starting at state, x_i, propose a new state x^*_{i+1} from $q(x^*_{i+1}|x_i)$
2. calculate acceptance probability p_{acc}
3. draw $u \sim \mathcal{U}(0, 1)$
4. if $u < p_{\text{acc}}$: $x_{i+1} = x^*_{i+1}$, else $x_{i+1} = x_i$

Metropolis (-Hastings)

Initialize with any state x_0



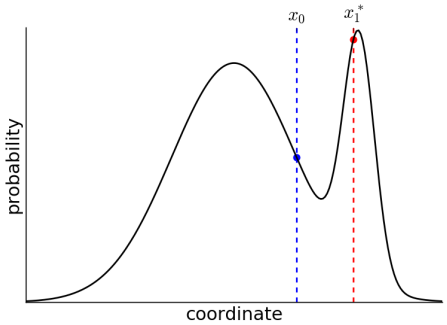
Sequence of states:
(x_0)

Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

Metropolis (-Hastings)

Initial state: x_0

1. calculate a proposal state x_1^* by randomly perturbing x_0



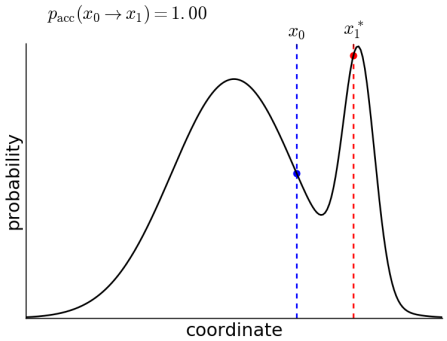
Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

Metropolis (-Hastings)

Initial state: x_0

1. calculate a proposal state x_1^* by randomly perturbing x_0
2. calculate acceptance probability

$$p_{\text{acc}} = \min\left(1, \frac{p(x_1^*)}{p(x_0)}\right)$$



Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

Metropolis (-Hastings)

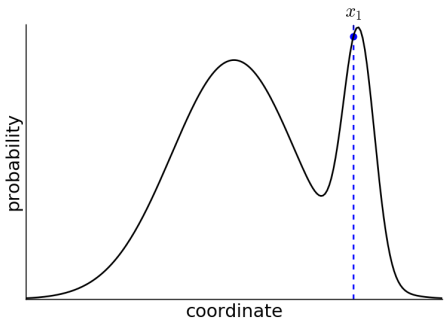
Initial state: x_0

1. calculate a proposal state x_1^* by randomly perturbing x_0
2. calculate acceptance probability

$$p_{\text{acc}} = \min\left(1, \frac{p(x_1^*)}{p(x_0)}\right)$$

3. with probability p_{acc}, accept proposal state x_1^* as the next state x_1, else copy x_0

Sequence of states:
(x_0, x_1)

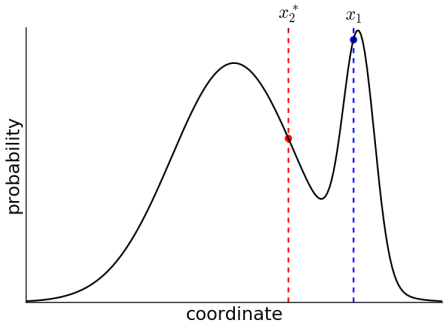


Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

Metropolis (-Hastings)

Current state: x_1

1. calculate a proposal state x_2^* by randomly perturbing x_1



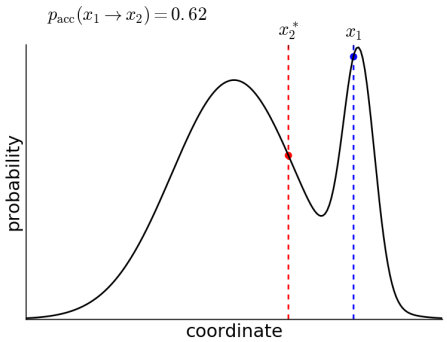
Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

Metropolis (-Hastings)

Current state: x_1

1. calculate a proposal state x_2^* by randomly perturbing x_1
2. calculate acceptance probability

$$p_{\text{acc}} = \min\left(1, \frac{p(x_2^*)}{p(x_1)}\right)$$



Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

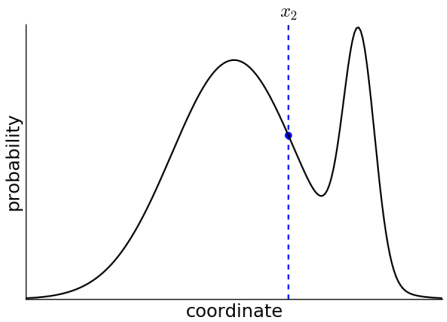
Metropolis (-Hastings)

Current state: x_1

1. calculate a proposal state x_2^* by randomly perturbing x_1
2. calculate acceptance probability

$$p_{\text{acc}} = \min\left(1, \frac{p(x_2^*)}{p(x_1)}\right)$$

3. with probability p_{acc}, accept proposal state x_2^* as the next state x_2, else copy x_1



Sequence of states:
(x_0, x_1, x_2)

Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

Metropolis (-Hastings)

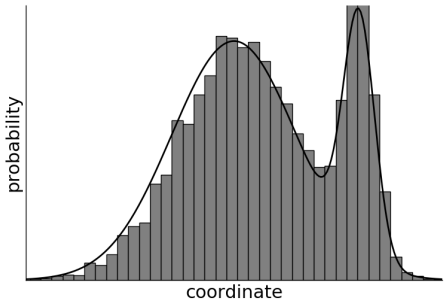
Current state: x_1

1. calculate a proposal state x_2^* by randomly perturbing x_1
2. calculate acceptance probability

$$p_{\text{acc}} = \min\left(1, \frac{p(x_2^*)}{p(x_1)}\right)$$

3. with probability p_{acc}, accept proposal state x_2^* as the next state x_2, else copy x_1

Sequence of states:
$(x_0, x_1, x_2, \ldots, x_n)$

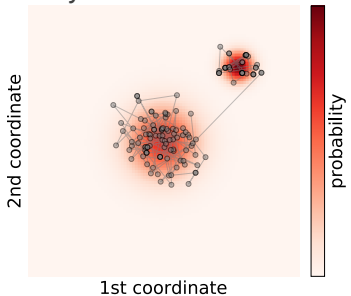


Metropolis et al., J. Chem. Phys (1953); Hastings, Biometrika (1970)

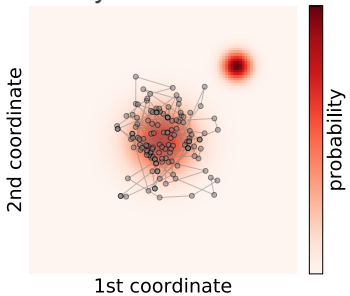
Multimodality: hard to sample

Markov chain can get stuck in modes:

Lucky us:



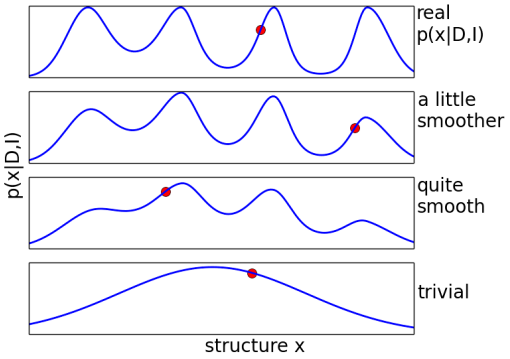
Unlucky us:



In higher dimensions: long time until all modes are discovered, if ever

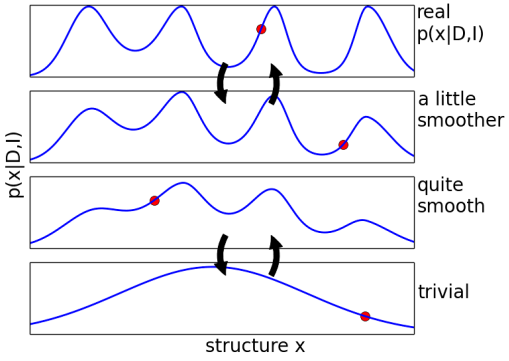
Replica Exchange

Simulate “flatter” versions of probability distribution...



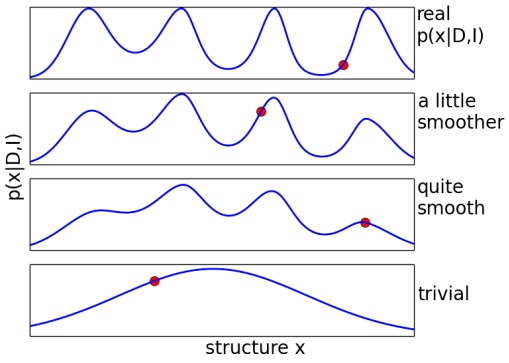
Replica Exchange

Simulate “flatter” versions of probability distribution and exchange states between Markov chains



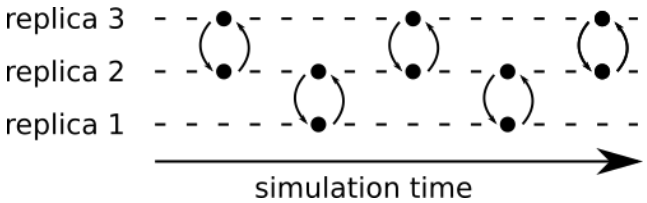
Replica Exchange

Simulate “flatter” versions of probability distribution and exchange states between Markov chains



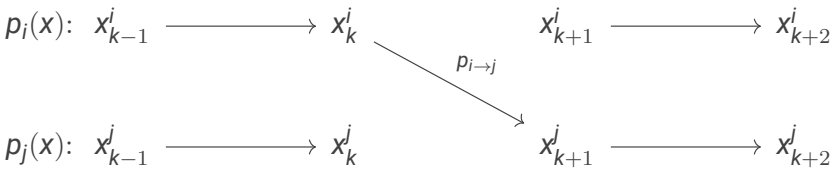
Replica Exchange: choosing swap partners

Choose swap partners such that all replicas are connected, for example:



Replica Exchange: acceptance criterion (informal motivation)

Plain exchanges disturb equilibrium distributions
$\rightarrow$ correct with acceptance criterion

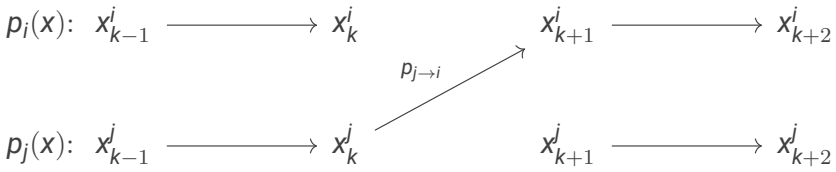


Probability of accepting x^i_k as $(k+1)$st state of $p_j(x)$ chain:

$$p_{i\to j} = \frac{p_j(x^i_k)}{p_j(x^j_k)}$$

Replica Exchange: acceptance criterion (informal motivation)

Plain exchanges disturb equilibrium distributions
$\rightarrow$ correct with acceptance criterion



Probability of accepting x^i_k as $(k+1)$st state of $p_j(x)$ chain:

$$p_{i\to j} = \frac{p_j(x^i_k)}{p_j(x^j_k)}$$

Probability of accepting x^j_k as $(k+1)$st state of $p_i(x)$ chain:

$$p_{j\to i} = \frac{p_i(x^j_k)}{p_i(x^i_k)}$$

Replica Exchange: acceptance criterion (informal motivation)

Multiply $p_{i\to j}$ and $p_{j\to i}$:

General acceptance criterion

$$p_{\text{acc}} = \min\left\{1, \frac{p_i(x^j)}{p_i(x^i)} \times \frac{p_j(x^i)}{p_j(x^j)}\right\}$$

Common case with $p_i(x) \propto \mathrm{e}^{-\beta_i E(x)}$:

$$p_{\text{acc}} = \min\left\{1, \mathrm{e}^{(\beta_i-\beta_j)(E(x^i)-E(x^j))}\right\}$$

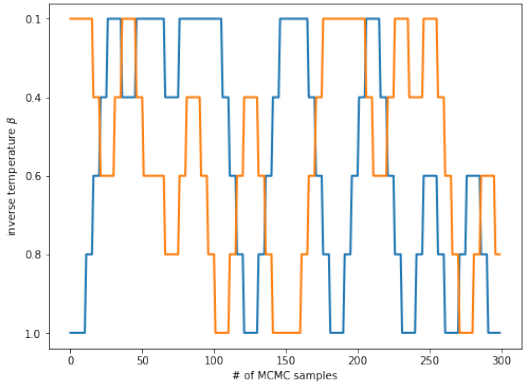
Physics terms

β: "inverse temperature"
$E(x) = -\log p(x)$: "energy"

The more different β_i and β_j, the lower p_{acc}!

Replica Exchange: life of a state

Single states perform a random walk on a temperature ladder:



Diffusion process: time to traverse ladder scales with $\sqrt{\#\text{ replicas}}$

Replica Exchange: TF Probability implementation

Application: Bayesian determination of chromatin structures

Application: Bayesian determination of chromatin structures

Waste recycling: multiple histogram reweighting

Model comparison

Evidence

$p(D) = \int \mathrm{d}x\, p(D|x)p(x)$:

normalization constant (long story...)

In our case:

$$
\begin{aligned}
p(k=1) &= \int_0^1 \mathrm{d}b\, L(k=1|b)p(b) \\
&= \int_0^1 \mathrm{d}b\ \ \mathrm{Bernoulli}(k;b) \times \mathrm{Beta}(k;\alpha=2,\beta=2)|_{k=1} \\
&= \int_0^1 \mathrm{d}b\ \ b^k(1-b)^{k-1}\frac{b(1-b)}{\frac{\Gamma(2)\Gamma(2)}{\Gamma(4)}}\Bigg|_{k=1} \\
&\vdots \\
&= \frac{1}{2}
\end{aligned}
$$

Evidence

$p(D) = \int \mathrm{d}x\, p(D|x)p(x)$:

normalization constant (long story...)

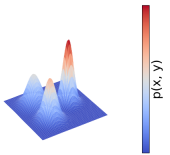
In our case:

$$
\begin{aligned}
p(k=1) &= \int_0^1 \mathrm{d}b\; L(k=1|b)p(b) \\
&= \int_0^1 \mathrm{d}b\;\; \mathrm{Bernoulli}(k;b) \times \mathrm{Beta}(k;\alpha=2,\beta=2)|_{k=1} \\
&= \int_0^1 \mathrm{d}b\;\; b^k(1-b) \left. \frac{b(1-b)}{\frac{\Gamma(2)\Gamma(2)}{\Gamma(4)}} \right|_{k=1} \\
&\;\;\vdots \\
&= \frac{1}{2}
\end{aligned}
$$

YIKES

Real-world Bayesian data analysis

In real problems: non-standard, difficult posterior distributions



Probabilistic programming libraries

Allow to

- programmatically define a statistical model
- sample from arbitrary posterior distributions
- run quality checks

Examples:

- PyMC3
- Stan
- TensorFlow Probability
- ...

Interlude: real-world issues

In reality, probability distributions often

- of non-standard form
- are multidimensional
- have highly correlated random variables
- are known only up to a normalization constant

Consequences:

- analytical evaluation of expectation values is impossible
- naïve sampling approaches are inefficient (curse of dimensionality)