CSC 412:
Probabilistic Learning and Reasoning
Week 5 : Sampling with Markov Chain Monte Carlo

Denys Linkov

University of Toronto

Overview

We continue with sampling algorithms.

- Markov chains
- Markov chain Monte Carlo
  - Metropolis-Hastings
  - Gibbs sampling
  - Hamiltonian Monte-Carlo

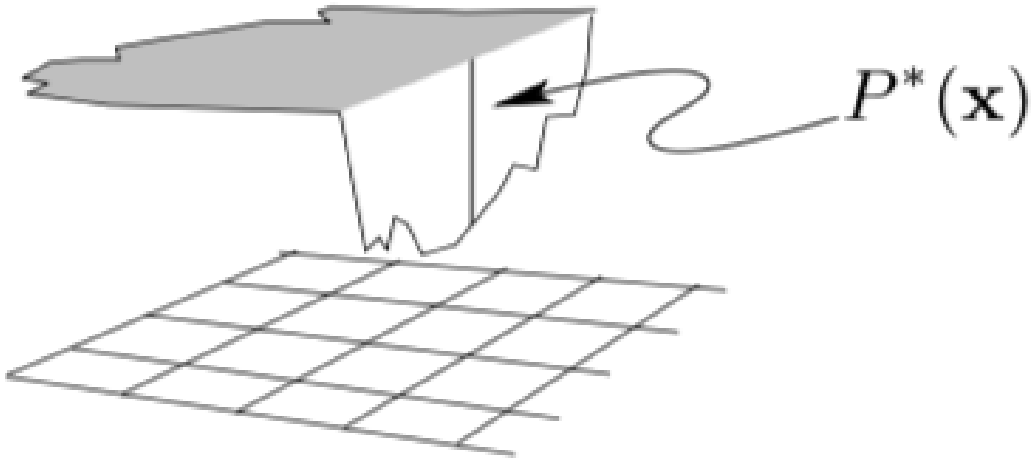
Announcements:

- Assignment 2 is released
- Please check your typesetting for A2

Day 2 of your internship, you are back on the lake

Imagine the tasks of drawing random water samples from a lake and finding the average plankton concentration. Let

- $\tilde{p}(\mathbf{x})$ = the depth of the lake at $\mathbf{x} = (x, y)$
- $\phi(\mathbf{x})$ = the plankton concentration as a function of $\mathbf{x}$
- Z = the volume of the lake = $\int \tilde{p}(\mathbf{x}) d\mathbf{x}$



What did we try last week?

Which problems are we trying to solve?

You can take the boat to any desired location $\mathbf{x}$ on the lake, and can measure the depth, $\tilde{p}(\mathbf{x})$, and plankton concentration, $\phi(\mathbf{x})$, at that point. Therefore,

- **Problem 1** is to draw water samples at random such that each sample is equally likely to come from any point within the lake.
- **Problem 2** is to find the average plankton concentration.

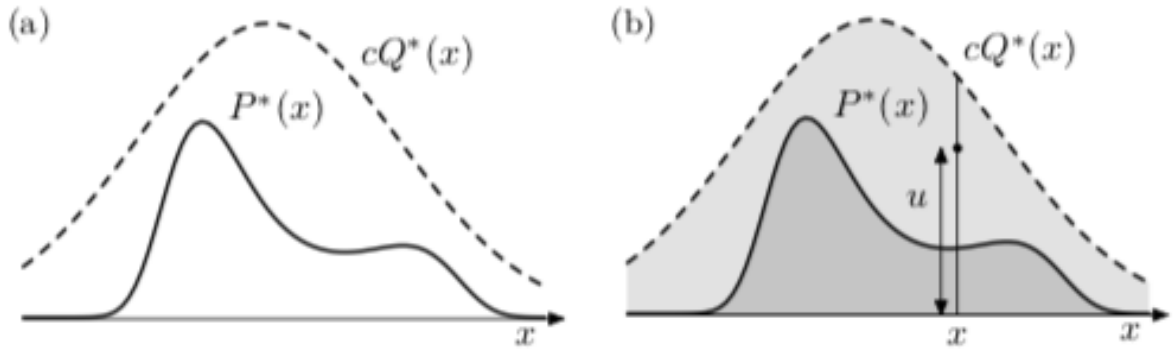
A slice through a lake that includes some canyons.

- To correctly estimate Φ, our method must implicitly discover the canyons and find their volume relative to the rest of the lake.

Using another distribution - Rejection sampling

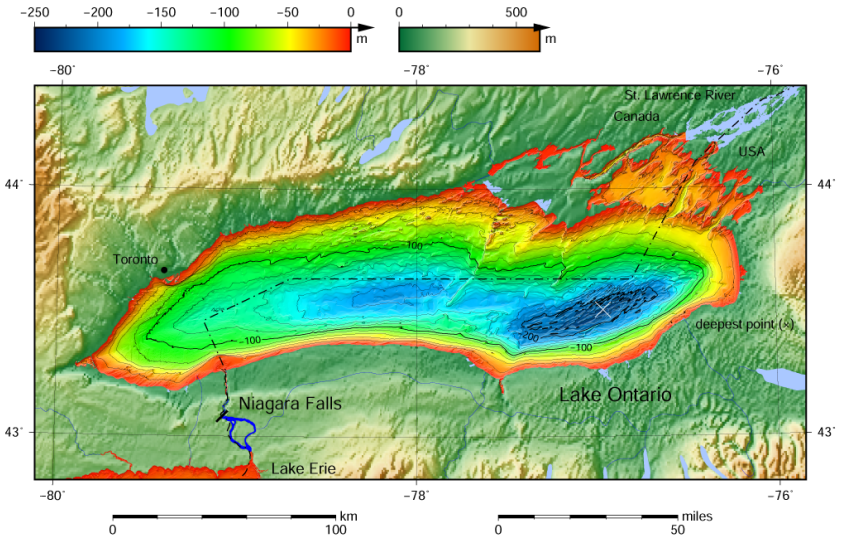
- We want expectations under $p(x) = \tilde{p}(x)/Z_p$ which is a very complicated one-dimensional density.
- Assume that we have a simpler proposal density $q(x)$ which we can evaluate (within a multiplicative factor Z_q, as before), and from which we can generate samples, i.e. $\tilde{q}(x) = Z_q \cdot q(x)$.
- Further assume that we know the value of a constant c such that

$$c\tilde{q}(x) > \tilde{p}(x) \quad \forall x$$



Using another distribution - Rejection sampling

- You call your friend who is near a puddle and use its depth $q(x)$ to help run rejection sampling.
- If the random number between 0 and $c * q(x)$ is less than $\tilde{p}(x)$, we accept
- If not ... we pour the water back into the lake



What if we see a bunch of plankton?

- We can use previous concentrations to better understand where we can find more plankton
- Maybe we should add some sequential dependency?

Sequential data

We considered methods in which the generated samples are *i.i.d*:

- We generated T samples

$$x_{1:T} = \{x_1, ..., x_T\}.$$

- But each sample was independent from each other

$$x_t \sim p(x) \text{ i.i.d.}$$

- This lecture, we will generate samples that are dependent.

Sequential data

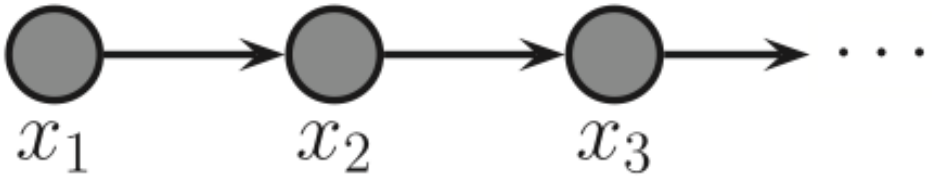
This also comes up when modelling the data. We generally assume data was i.i.d, however this may be a poor assumption:

- **Sequential data** are common:
  - time-series modelling (e.g. stock prices, speech, video analysis)
  - ordered data (e.g. textual data, gene sequences)
- Recall the general joint factorization via the chain rule

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{t-1}, ..., x_1) \quad \text{where } p(x_1|x_0) = p(x_1).$$

- This quickly becomes intractable for high-dimensional data. Each factor requires exponentially many parameters to specify.

Markov chains



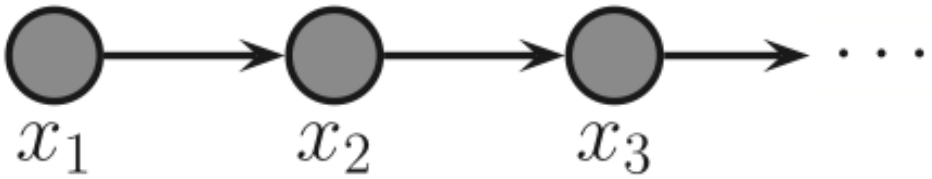
- We make the simplifying **first-order Markov chain** assumption:

$$p(x_t|x_{1:t-1}) = p(x_t|x_{t-1})$$

- This assumption greatly simplifies the factors in the joint distribution

$$p(x_{1:T}) = \prod_{t=1}^{T} p(x_t|x_{t-1})$$

Stationary Markov chains



Further assumptions may be useful:

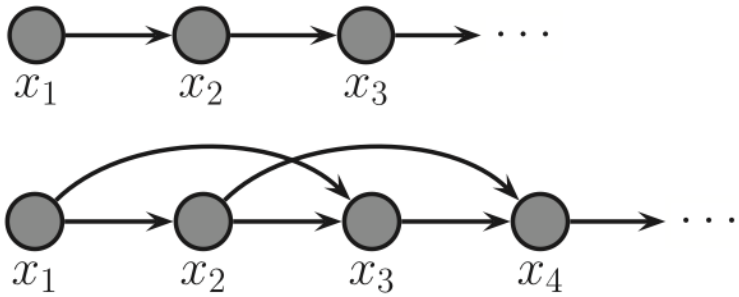
- **Stationary (homogeneous) Markov chain**: the distribution generating the data does not change through time

$$p(x_{t+1} = y | x_t = x) = p(x_{t+2} = y | x_{t+1} = x) \quad \text{for all } t.$$

- **Non-stationary Markov chain**: the transition probabilities $p(x_{t+1} = y | x_t = x)$ depend on the time t.

Here we only consider stationary Markov chains.

Higher-order Markov chains



In some cases, the first-order assumption may be restrictive (such as when modeling natural language, where long-term dependencies occur often). We can generalize to high-order dependence trivially

- Second order:

$$p(x_t|x_{1:t-1}) = p(x_t|x_{t-1}, x_{t-2})$$

- m-th-order

$$p(x_t|x_{1:t-1}) = p(x_t|x_{t-1:t-m})$$

Transition matrix

- When x_t is discrete (e.g. $x_t \in \{1, ..., K\}$), the conditional distribution $p(x_t|x_{t-1})$ can be written as a $K \times K$ matrix.
- We call this the **transition** (or stochastic) **matrix** A:

$$A_{ij} = p(x_t = j|x_{t-1} = i), \qquad A \in \mathbb{R}^{K \times K}.$$

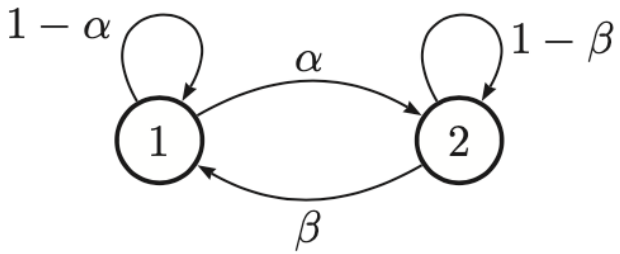
- Note that

$$\begin{aligned} p(x_t = j) &= \sum_i p(x_t = j|x_{t-1} = i)p(x_{t-1} = i), \\ &= \sum_i A_{ij} p(x_{t-1} = i). \end{aligned}$$

- Each row of the matrix sums to one, $\sum_j A_{ij} = 1$.

State transition diagram

- The transition matrix A: $A_{ij} = p(x_t = j | x_{t-1} = i)$ is the probability of going from state i to state j.



- We can visualize Markov chains via a directed graph, where nodes represent states and arrows represent legal transitions, i.e., non-zero elements of A.

- This is a **state transition diagram**.
- The weights associated with the arcs are the probabilities.
- The transition matrix for the 2-state chain shown above is given by

$$A = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$$

Chapman-Kolmogorov equations

- The n-step transition matrix $A(n)$ is defined as

$$A_{ij}(n) = p(x_{t+n} = j | x_t = i)$$

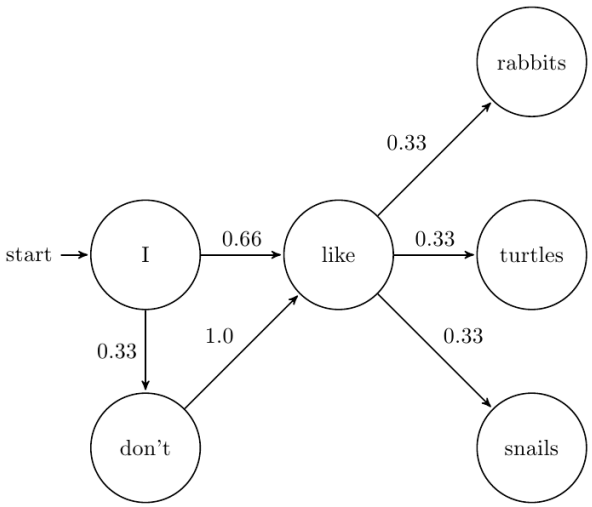
  which is the probability of getting from i to j in exactly n steps.
- Notice that $A(1) = A$.
- **Chapman-Kolmogorov** equations state that

$$A_{ij}(m+n) = \sum_{k=1}^{K} A_{ik}(m) A_{kj}(n) \quad \text{equivalently} \quad A(m+n) = A(m)A(n)$$

  the probability of getting from i to j in $m+n$ steps is just the probability of getting from i to k in m steps, and then from k to j in n steps, summed up over all k.
- So $A(n) = A \times A(n-1) = A \times A \times A(n-2) = \cdots = A^n$.

Application: Markov Language Models

- We could use Markov chains as language models, which are distributions over sequences of words.
- State space is all words and x_t denotes the t-th word in a sentence.
- We may use a first-order Markov model.



- To estimate A and π, note that the probability of any particular sentence of length T is:

$$p(x_{1:T}|\theta) = \pi(x_1)A(x_1, x_2)\cdots A(x_{T-1}, x_T)$$
$$= \prod_{j=1}^{K} \pi_j^{1[x_1=j]} \prod_{t=2}^{T} \prod_{j=1}^{K} \prod_{k=1}^{K} A_{jk}^{1[x_t=k, x_{t-1}=j]}$$

where $\pi(x_1)$ is the probability of the sentence starting with word x_1.

The estimating equations have a natural form.

Application: Markov Language Models

- We use MLE to estimate A from data $\mathcal{D} = \{x^{(1)}, ..., x^{(N)}\}$.
- Likelihood of any particular sentence $x^{(i)}$ of length T_i

$$p(x^{(i)}|\theta) = \prod_{j=1}^{K} \pi_j^{1[x_1^{(i)}=j]} \prod_{t=2}^{T_i} \prod_{j=1}^{K} \prod_{k=1}^{K} A_{jk}^{1[x_t^{(i)}=k, x_{t-1}^{(i)}=j]}$$

- Log-likelihood of $\mathcal{D}$ (all sentences treated as independent)

$$\log p(\mathcal{D}|\theta) = \sum_{i=1}^{N} \log p(x^{(i)}|\theta) = \sum_j N_j^1 \log \pi_j + \sum_j \sum_k N_{jk} \log A_{jk}$$

where we define the counts

$$N_j^1 = \sum_{i=1}^{N} 1[x_1^{(i)} = j], \qquad N_{jk} = \sum_{i=1}^{N} \sum_{t=1}^{T_i - 1} 1[x_t^{(i)} = j, x_{t+1}^{(i)} = k].$$

- The MLE is given as $\hat{\pi}_j = \frac{N_j^1}{\sum_j N_j^1}$ $\hat{A}_{jk} = \frac{N_{jk}}{\sum_k N_{jk}}$.

Stationary distribution of a Markov chain

- We are often interested in the long term distribution over states, which is known as the **stationary distribution** of the chain.
- Let A be the transition matrix, e.g. $p(x_{t+1} = j | x_t = i) = A_{ij}$ and $\pi_t(j) = p(x_t = j)$ be the probability of being in state j at time t.
- The initial distribution is given by $\pi_0 \in \mathbb{R}^K$ and

$$\pi_1(j) = \sum_{i=1}^{K} p(x_1 = j | x_0 = i)\pi_0(i) = \sum_{i=1}^{K} A_{ij}\pi_0(i) = \sum_{i=1}^{K} (A^\top)_{ji}\pi_0(i).$$

- Using the vector notation $\pi_1 = A^\top \pi_0$ and more generally

$$\pi_t = A^\top \pi_{t-1} = A^\top A^\top \pi_{t-2} = \cdots = (A^\top)^t \pi_0.$$

- Do this infinitely many steps, the distribution of x_t may converge

$$\pi = A^\top \pi.$$

then we have reached the stationary distribution (aka the invariant distribution) of the Markov chain.

Stationary distribution

A bit of linear algebra:

- We can find the stationary distribution of a Markov chain by solving the eigenvector equation

$$A^\top v = v \quad \text{and set} \quad \pi = v.$$

  v is the eigenvector of $A^\top$ with eigenvalue 1.

- Need to normalize!
- Since $A\mathbf{1} = \mathbf{1}$ (row sums are 1), 1 is an eigenvalue of A with eigenvector $\mathbf{1}$. A and $A^\top$ have the same eigenvalues. It follows that 1 is also the eigenvalue of $A^\top$.
- The stationary distribution may not be unique.

Example: Stationary distribution

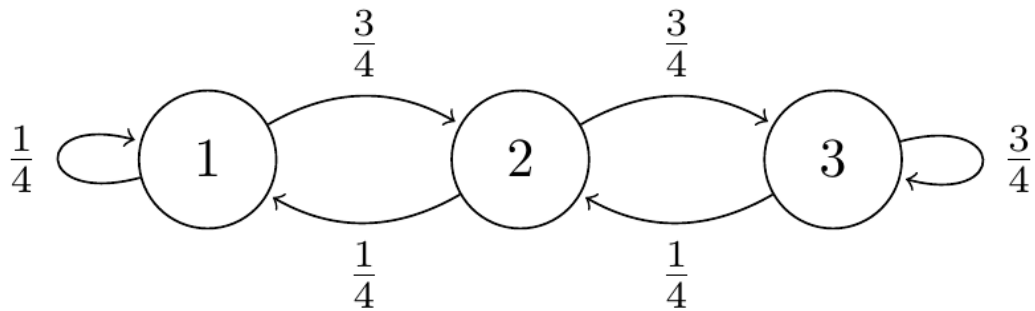
Assume you are trying to figure out where your friend will be at a given point in time. They never respond to messages.

You know their transition matrix and it's defined as

- A_1 is the library
- A_2 is home
- A_3 is the blue food truck

$$A = \begin{pmatrix} \frac{1}{4} & \frac{3}{4} & 0 \\ \frac{1}{4} & 0 & \frac{3}{4} \\ 0 & \frac{1}{4} & \frac{3}{4} \end{pmatrix}$$

You want to make the best guess where they will be.



Equations and Normalizing Condition

Since we are looking for the eigan vector for $\lambda = 1$, we will look for $\pi = \pi A$ We need to solve $\pi = \pi A$:

$$\pi_1 = \frac{1}{4}\pi_1 + \frac{1}{4}\pi_2$$
$$\pi_2 = \frac{3}{4}\pi_1 + \frac{1}{4}\pi_3$$
$$\pi_3 = \frac{3}{4}\pi_2 + \frac{3}{4}\pi_3$$

Normalizing condition:

$$\pi_1 + \pi_2 + \pi_3 = 1$$

Solving for π_2 and π_3 in terms of π_1

From the first equation:

$$\frac{3}{4}\pi_1 = \frac{1}{4}\pi_2 \implies \pi_2 = 3\pi_1$$

From the third equation:

$$\frac{1}{4}\pi_3 = \frac{3}{4}\pi_2 \implies \pi_3 = 3\pi_2$$

Substituting $\pi_2 = 3\pi_1$, we get $\pi_3 = 9\pi_1$.
Now we substitute to find the normalizing condition.

$$\pi_1 + 3\pi_1 + 9\pi_1 = 1$$

$$13\pi_1 = 1 \implies \pi_1 = \frac{1}{13}$$

The Stationary Distribution

Now we can find π_2 and π_3:

$$\pi_2 = 3\pi_1 = \frac{3}{13}$$

$$\pi_3 = 9\pi_1 = \frac{9}{13}$$

Therefore, the stationary distribution is:

$$\pi = \left(\frac{1}{13}, \frac{3}{13}, \frac{9}{13}\right)$$

Detailed balance equations

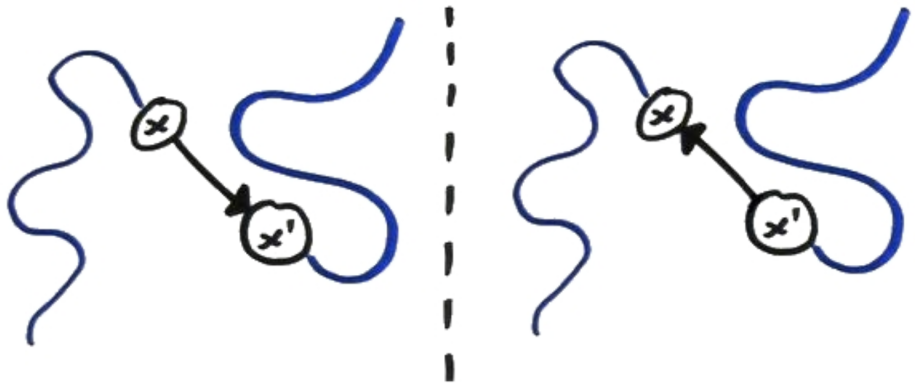
Markov Chain is called:

- **irreducible** if we can get from any state to any other state.
- **regular** if A^n has positive entries for some n.
- **time reversible** if there exists a distribution π such that

$$\pi_i A_{ij} = \pi_j A_{ji} \qquad \text{for all } i, j.$$

This is called the **detailed balance equations**.

Detailed balance means $\rightarrow x \rightarrow x'$ and $\rightarrow x' \rightarrow x$ are equally probable:



Detailed balance equations

Detailed balance equations (DB): $\pi_i A_{ij} = \pi_j A_{ji}$ for all i, j.

Theorem

If a Markov chain with transition matrix A satisfies detailed balance wrt distribution π, then π is a stationary distribution.

Proof: Show that $A^\top \pi = \pi$ or, in other words, that

$$\sum_{i=1}^{K} \pi_i A_{ij} = \pi_j \qquad \text{for all } j.$$

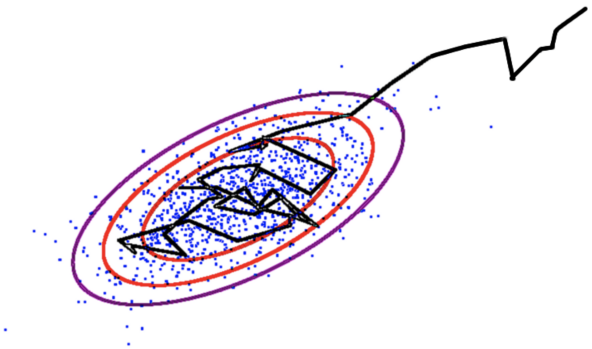
Indeed, for every $j = 1, \ldots, K$, we have

$$\sum_{i=1}^{K} \pi_i A_{ij} \overset{\text{(DB)}}{=} \sum_{i=1}^{K} \pi_j A_{ji} = \pi_j \sum_{i=1}^{K} A_{ji} = \pi_j.$$

Metropolis Algorithm (first encounter with MCMC)

- Now that you know about markov chains, can we do a random walk on our $p(x)$ to sample effectively?
- We can just move around using information from $p(x)$ and random sampling.
- Do we even need our friend at the puddle $q(x)$?
- Yes, just ask them to be a random number generator.

Markov Chain Monte Carlo (MCMC)



- In contrast to rejection sampling, where the accepted points $\{x^{(t)}\}$ are independent, MCMC methods generate a dependent sequence.
- Each sample $x^{(t)}$ has a probability distribution that depends on the previous value, $x^{(t-1)}$.
- MCMC methods need to be run for a time in order to generate samples that are from the target distribution p.

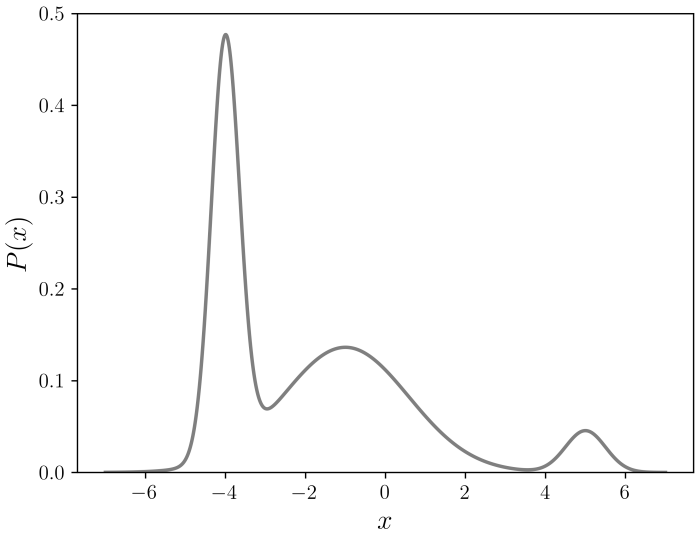
We can still do Monte Carlo estimaton for large enough T to estimate the mean of a test function ϕ:

$$\mathbb{E}_{x\sim p}[f(x)] \approx \frac{1}{T}\sum_{t=1}^{T} f(x^{(t)}).$$

(good idea to discard a bunch of initial samples)

Metropolis (first encounter with MCMC)

Importance and rejection sampling work only if the proposal density $q(x)$ is similar to $p(x)$. In high dimensions, it is hard to find one such q.



- The Metropolis algorithm instead uses our target density p which depends on the current state $x^{(t)}$.
- We then sample a new state uniformly (symetrically), based on the interval $[x^{(t-1)} - s, x^{(t-1)} + s]$
- We then compare $\tilde{p}(x')$ and $\tilde{p}(x^{(t)})$

We accept the new state with probability of

$$A(x'|x^{(t)}) = \min\left\{1, \frac{\tilde{p}(x')}{\tilde{p}(x^{(t)})}\right\}$$

Example of Metropolis

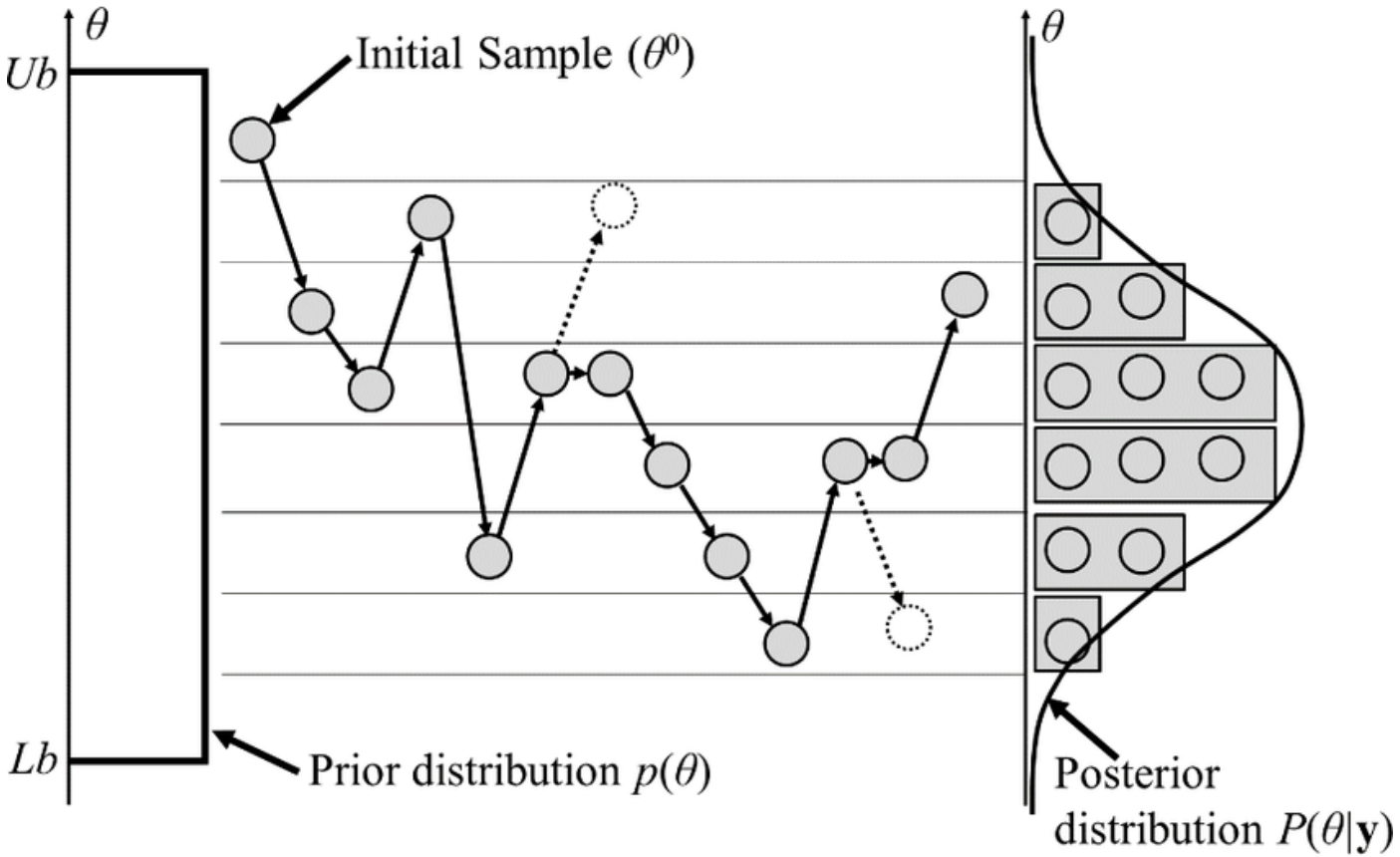
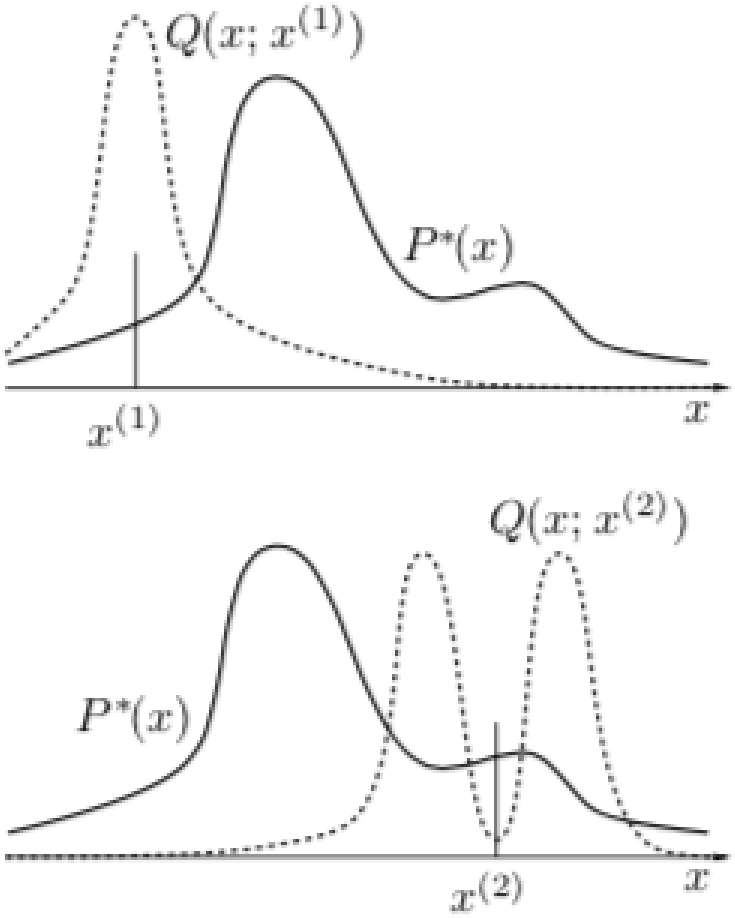


Figure: Random walk on Normal distribution (Wikipedia)

Intuition: if we are going into are lower probability area, we should be accept with a lower probability

Metropolis-Hastings

Importance and rejection sampling work only if the proposal density $q(x)$ is similar to $p(x)$. In high dimensions, it is hard to find one such q.



- The Metropolis-Hastings algorithm instead uses a proposal density q which depends on the current state $x^{(t)}$.
- The density $q(x|x^{(t)})$ might be a simple distribution such as a Gaussian centered on the current $x^{(t)}$, but can be any density from which we can draw samples.
- In contrast to importance and rejection sampling, it is not necessary that $q(x|x^{(t)})$ looks similar to $p(x)$.

Metropolis-Hastings algorithm

As before, assume we can evaluate $\tilde{p}(x)$ for any x. Our procedure:

- A tentative new state x' is generated from the proposal density $q(x'|x^{(t)})$. We accept the new state with probability

$$A(x'|x^{(t)}) = \min\left\{1, \frac{\tilde{p}(x')q(x^{(t)}|x')}{\tilde{p}(x^{(t)})q(x'|x^{(t)})}\right\}$$

  - If accepted, set $x^{(t+1)} = x'$. Otherwise, set $x^{(t+1)} = x^{(t)}$.

- Metropolis: Simpler version when $q(x'|x) = q(x|x')$ for all x, x'.

- **Theorem:** This procedure defines a Markov chain with stationary distribution $\pi(x)$ equal to the target distribution $p(x)$.

Proof of the theorem

Recall $A(x'|x) = \min\left\{1, \frac{\tilde{p}(x')q(x|x')}{\tilde{p}(x)q(x'|x)}\right\} = \min\left\{1, \frac{p(x')q(x|x')}{p(x)q(x'|x)}\right\}$.

The resulting Markov chain has the following transition probabilities:

$$r(x'|x) = \begin{cases} q(x'|x)A(x'|x) & \text{if } x' \neq x \\ q(x|x) + \sum_{x' \neq x} q(x'|x)(1 - A(x'|x)) & \text{if } x' = x \end{cases}.$$

Show (DB) $r(x'|x)p(x) = r(x|x')p(x')$. If $x \neq x'$

$$r(x'|x)p(x) = p(x)q(x'|x)\min\left\{1, \frac{p(x')q(x|x')}{p(x)q(x'|x)}\right\} = \min\left\{p(x')q(x|x'), p(x)q(x'|x)\right\}$$

$$r(x|x')p(x') = p(x')q(x|x')\min\left\{1, \frac{p(x)q(x'|x)}{p(x')q(x|x')}\right\} = \min\left\{p(x')q(x|x'), p(x)q(x'|x)\right\}$$

Thus p is a stationary distribution of this Markov chain.

Overview for the remaining hour

- Gibbs sampling
- Hamiltonian Monte Carlo
- MCMC diagnostics

Gibbs Sampling Procedure

Suppose the vector x has been divided into d components

$$x = (x_1, ..., x_d).$$

Start with any $x^{(0)} = (x_1^{(0)}, \ldots, x_d^{(0)})$. In the t-th iteration:

- For $j = 1, \ldots, d$:
  - Sample $x_j^{(t)}$ from the conditional distribution given other components:

$$x_j^{(t)} \sim p(x_j | x_{-j}^{(t-1)})$$

    Where $x_{-j}^{(t-1)}$ represents all the components of x except for x_j at their current values:

$$x_{-j}^{(t-1)} = (x_1^{(t)}, x_2^{(t)}, ..., x_{j-1}^{(t)}, x_{j+1}^{(t-1)}, ..., x_d^{(t-1)})$$

- No accept/reject, only accept.

Example: Bivariate Gaussian

Consider a (simple) problem of sampling from the bivariate Gaussian

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \sim N_2(\mu, \Sigma), \qquad \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}.$$

We have

$$X_1|X_2 = x_2 \sim N(\mu_1 + \rho(x_2 - \mu_2), 1 - \rho^2)$$

$$X_2|X_1 = x_1 \sim N(\mu_2 + \rho(x_1 - \mu_1), 1 - \rho^2)$$

Given $X^{(0)} = (0, 0)$ we proceed iteratively for $t \geq 1$:

$$X_1^{(t)} \sim N(\mu_1 + \rho(x_2^{(t-1)} - \mu_2), 1 - \rho^2)$$

$$X_2^{(t)} \sim N(\mu_2 + \rho(x_1^{(t)} - \mu_1), 1 - \rho^2)$$

Sampling Job Offer Happiness with Gibbs

You want to sample how happy you'll be different job offers, assume your happiness is given by the function $p(h, i, s, m, c, w)$ where

- h is happiness
- i is your interest in the work
- s is income
- m is morality
- c is coworkers
- w is hours worked.

Now we initialize our initial values.

$$[h, i, s, m, c, w] = [0, 0, 0, 0, 0, 0]$$

And iterate by sampling from some conditional distribution p_c

$$h_1 \sim p_c(h_0, i_0, s_0, m_0, c_0, w_0)$$

Sampling Job Offer Happiness with Gibbs

And iterate by sampling from some condition distribution p_c

- $h_1 \sim p_c(h_0, i_0, s_0, m_0, c_0, w_0)$
- $i_1 \sim p_c(h_1, i_0, s_0, m_0, c_0, w_0)$
- $s_1 \sim p_c(h_1, i_1, s_0, m_0, c_0, w_0)$
- $m_1 \sim p_c(h_1, i_1, s_1, m_0, c_0, w_0)$
- $c_1 \sim p_c(h_1, i_1, s_1, m_1, c_0, w_0)$
- $w_1 \sim p_c(h_1, i_1, s_1, m_1, c_1, w_0)$

Is this the best way to sample for this type of problem?
Probably not, conditional probability isn't easier to sample from, random walk might not be intuitive approach.

Example: Bivariate Gaussian

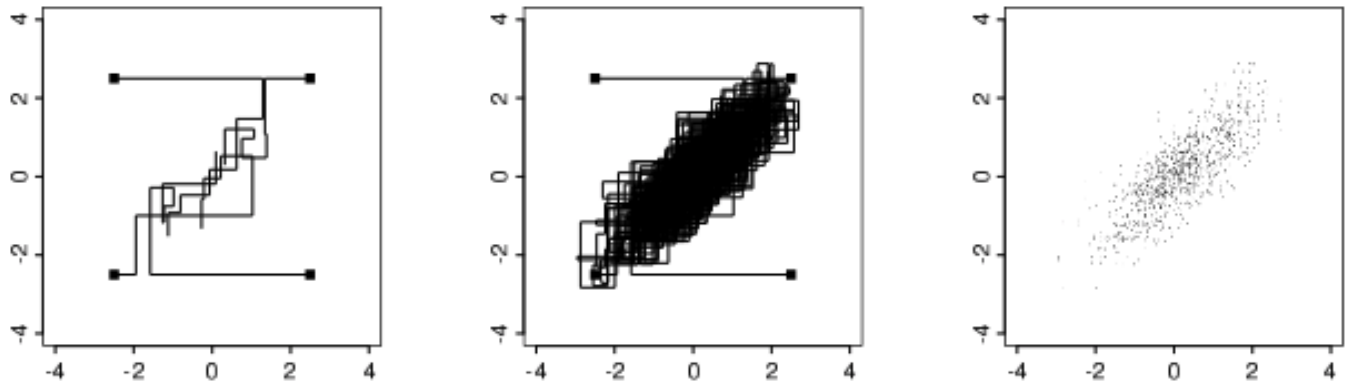


Figure 11.2 *Four independent sequences of the Gibbs sampler for a bivariate normal distribution with correlation $\rho = 0.8$, with overdispersed starting points indicated by solid squares. (a) First 10 iterations, showing the componentwise updating of the Gibbs iterations. (b) After 500 iterations, the sequences have reached approximate convergence. Figure (c) shows the points from the second halves of the sequences, representing a set of correlated draws from the target distribution.*

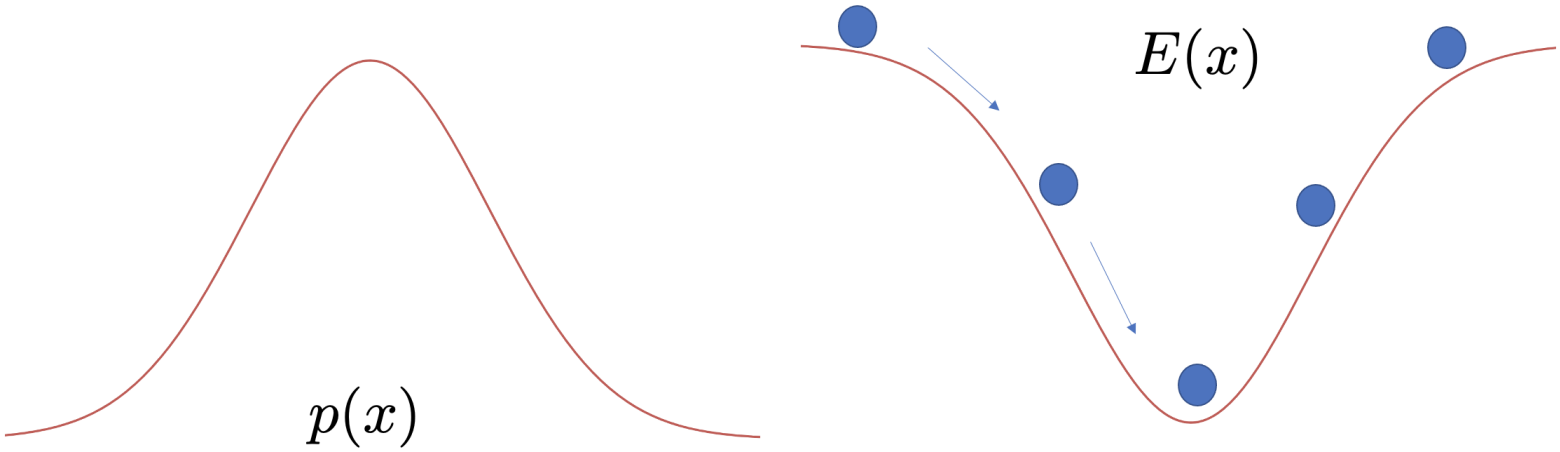
1

(The real power of Gibbs approach comes in situations when the distribution is hard but full-conditionals are simple, e.g. Ising)

[1] From "Bayesian Data Analysis Third edition" by Gelman, Carlin, Stern, Dunson, Vehtari, Rubin

Hamiltonian Monte Carlo

- This is essentially a Metropolis-Hastings algorithm with a specialized proposal mechanism.
- Algorithm uses a physical analogy to make proposals.
- Given the position x, the potential energy is $E(x)$



- Construct a distribution

$$p(x) \propto e^{-E(x)}, \quad \text{with} \quad E(x) = -\log(\tilde{p}(x))$$

where $\tilde{p}(x)$ is the unnormalized density we can evaluate.

Hamiltonian Monte Carlo

- Introduce **momentum** v carrying the kinetic energy

$$K(v) \;=\; \tfrac{1}{2}\|v\|^2 \;=\; \tfrac{1}{2}v^\top v.$$

- Total energy or **Hamiltonian**:

$$H(x, v) = E(x) + K(v).$$

- Energy is preserved:
  - Frictionless ball rolling $(x, v) \to (x', v')$
  - $H(x, v) = H(x', v')$.

- Ideal Hamiltonian dynamics are reversible: reverse v and the ball will return to its start point! $\quad (x', -v') \to (x, -v)$

Hamiltonian Monte Carlo

- The joint distribution:
  - $p(x, v) \propto e^{-E(x)} e^{-K(v)} = e^{-E(x)-K(v)} = e^{-H(x,v)}$
  - Momentum is Gaussian, and independent of the position.
- MCMC procedure
  - Sample the momentum from the standard Gaussian.
  - Simulate Hamiltonian dynamics, flip sign of the momentum
    - Hamiltonian dynamics is reversible.
    - Energy is constant $p(x, v) = p(x', v') = p(x', -v')$.
- How to simulate Hamiltonian dynamics? Take:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{\partial H}{\partial v} = \frac{\partial K}{\partial v}$$
$$\frac{\mathrm{d}v}{\mathrm{d}t} = -\frac{\partial H}{\partial x} = -\frac{\partial E}{\partial x}$$

(Indeed: $\frac{\mathrm{d}H}{\mathrm{d}t} = \sum_i \frac{\partial E}{\partial x_i}\frac{\mathrm{d}x_i}{\mathrm{d}t} + \sum_i \frac{\partial H}{\partial v_i}\frac{\mathrm{d}v_i}{\mathrm{d}t}$ will be zero)

Leap-frog integrator

- A numerical approximation:

$$\begin{aligned} v(t+\tfrac{\epsilon}{2}) &= v(t) + \frac{\epsilon}{2}\frac{\mathrm{d}v}{\mathrm{d}t}(t) \;=\; v(t) - \frac{\epsilon}{2}\frac{\partial E}{\partial x}(x(t)) \\ x(t+\epsilon) &= x(t) + \epsilon\frac{\mathrm{d}x}{\mathrm{d}t}(t) \;=\; x(t) + \epsilon\frac{\partial K}{\partial v}(v(t+\tfrac{\epsilon}{2})) \\ v(t+\epsilon) &= v(t+\tfrac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E}{\partial x}(x(t+\epsilon)) \end{aligned}$$

  (Slightly more accurate than the standard Euler's method)
- We do a fixed number of leap-frog steps.
- Dynamics are still deterministic (and reversible)

HMC algorithm

The HMC algorithm (run until it mixes):

- Current position: $(x^{(t-1)}, v^{(t-1)}))$
- Sample momentum: $v^{(t)} \sim \mathcal{N}(0, I)$.
- Start at $(x, v) = (x^{(t-1)}, v^{(t)})$ and run Leapfrog integrator for L steps and reach (x', v')
- Accept new state $(x', -v')$ with probability:

$$\min\left\{1, \frac{\exp(H(x^{(t-1)}, v^{(t-1)}))}{\exp(H(x', v'))}\right\}$$

- Low energy points are favored.

MCMC Inference

- Sample from unnormalized posterior.
- Estimate statistics from simulated values of x:
  - mean
  - median
  - quantiles
- **Posterior predictive distribution** of unobserved outcomes can be obtained by further simulation conditional on drawn values of x.
- All of this however requires some care, as MCMC is not without problems.

MCMC diagnostics

- How do we know we have ran the algorithm long enough?
- What if we started very far from where our distribution is?
- Since there is correlation between each item of the chain (autocorrelation), what is the "effective" number of samples?

Good Ideas for MCMC

Some obvious things to consider:

- Parallel computation is cheap - we can run multiple chains in parallel starting at different points
- We should discard some initial samples - **burn-in phase**.
- We should examine how well the chain is "mixed".

(No need to memorize any of the formulas below)

R hat

- Start with m chains each of length n, $(x_{ij})_{ij} \in \mathbb{R}^{n \times m}$.
  - this will be already after a fixed burn-in phase.
- The **between sequence variance** B is:

$$B = \frac{n}{m-1} \sum_{j=1}^{m} (\bar{x}_{.j} - \bar{x}_{..})^2,$$

where:

$$\bar{x}_{.j} = \frac{1}{n} \sum_{i=1}^{n} x_{ij} \quad \text{and} \quad \bar{x}_{..} = \frac{1}{m} \sum_{j=1}^{m} \bar{x}_{.j} = \frac{1}{mn} \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij}$$

(individual chain means, total mean)

R hat

- The **within sequence variance** W is:

$$W = \frac{1}{m} \sum_{j=1}^{m} s_j^2$$

where:

$$s_j^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_{ij} - \bar{x}_{.j})^2$$

- **Idea**: If one or more chain has not mixed well, the variance of all the chains combined together should be higher than that of individual chains.

R hat

- Next we compute the average variance:

$$\widehat{\text{var}}^{+}(x) = \frac{n-1}{n} W + \frac{1}{n} B$$

- Finally define **R-hat** coefficient:

$$\hat{R} = \sqrt{\frac{\widehat{\text{var}}^{+}(x)}{W}}$$

- If chains have not mixed well, R-hat is larger than 1.
- **Split-$\widehat{\mathbf{R}}$**: Split each chain into the first and second halves. This can detect non-stationarity within a single chain.

Effective Sample Size

- If $x_1, \ldots, x_n$ are i.i.d. with variance σ^2 then $\text{var}(\bar{x}_n) = \frac{\sigma^2}{n}$.
- In general, without assuming independence

$$\text{var}(\bar{x}) = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \text{cov}(x_i, x_j) = \frac{\sigma^2}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \text{corr}(x_i, x_j)$$

  so $\frac{n^2}{\sum_{i=1}^{n} \sum_{j=1}^{n} \text{corr}(x_i, x_j)}$ measures "effective sample size".
- Similarly, we define the **effective sample size** to be:

$$n_{\text{eff}} = \frac{mn}{1 + 2\sum_{t=1}^{\infty} \rho_t}$$

  where $\rho_t = \text{corr}(x_0, x_t)$ are unknown, so we also estimate them.

Diagnostics Summary

- Once $\hat{R}$ is near 1, and $\hat{n}_{\text{eff}}$ is more than 10 per chain **for all scalar estimands** we collect the mn simulations, (excluding the burn-in).
- We can then draw inference based on our samples. However:
  - Even if the iterative simulations appear to have converged, passed all tests etc. It may still be far from convergence!
- When we declare "convergence" - we mean that all chains appear stationary and well mixed.
- Next week: HMMs and Variational Inference