37262 Mathematical Statistics

Lecture 10

Markov Chains

- Recall the definition of a Markov Chain.

- A sequence of random variables $\{X_0, X_1, X_2, ...\}$ is a Markov Chain on the set of states S if

  $$P(X_{n+1} = j \,|\, X_n = i, X_{n-1} = i_{n-1}, X_{n-2} = i_{n-2}, ... X_0 = i_0) = P(X_{n+1} = j \,|\, X_n = i)$$

  for all possible n and all possible $j, i, i_{n-1}, i_{n-2}, ..., i_0 \in S$.

- In plain language, this simplified to a sequence of observations such that the probabilities of all possible future states depended only on knowledge of the present state and not on any previous history as well.

Markov Chains: Equilibrium Distributions

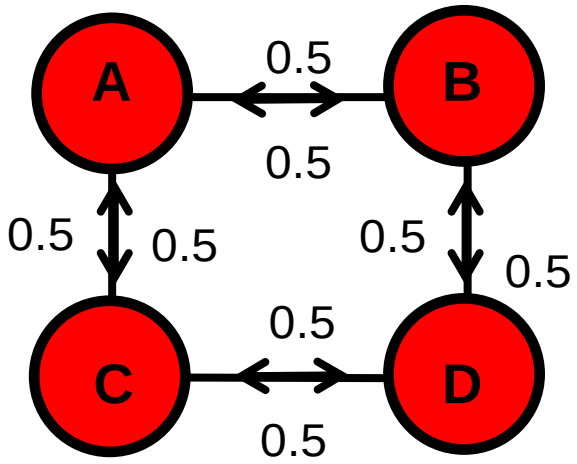
- Many Markov Chains (including all finite Markov Chains) will have an equilibrium distribution. That is, as the number of moves $\to \infty$, any further observations can be regarded as realisations of a random variable with a fixed distribution.

- This may be a single value (for example in a Snakes and Ladders game, the equilibrium distribution is simply being in the Finish square with probability 1) but, in general, will not be.

- In some cases, these distributions can be obtained visually without calculation.

Markov Chains: Equilibrium Distributions

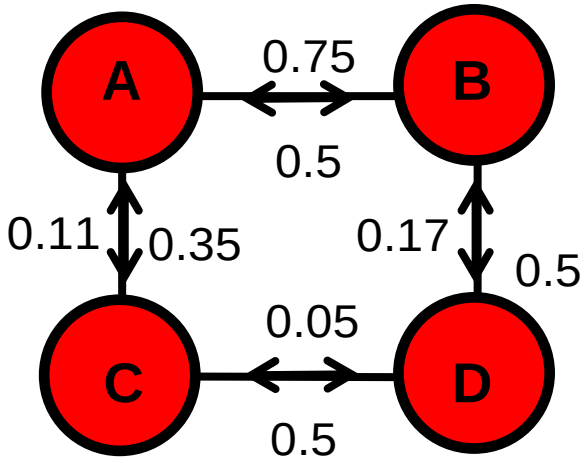
- Consider a Markov Chain on states A, B, C and D such that, if in a given position, the system moves to another state in the next with probability equal to the weight of the arrow in the diagram.



- In this case, it should be easy to see that the equilibrium distribution is that eventually the system is equally likely to be in each of the four states.

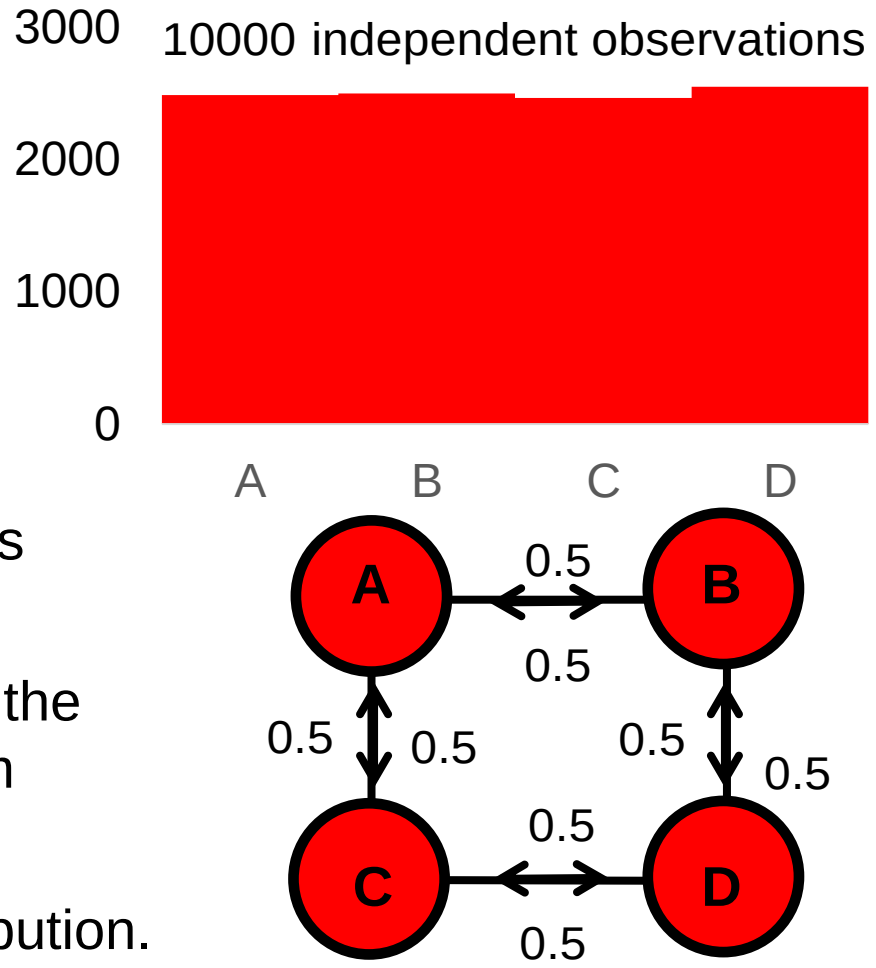
Markov Chains: Equilibrium Distributions

- Other systems might have equilibrium distributions which perhaps not as easy to identify.
- For example consider the (similar) system on ABCD.
- The equilibrium distribution for this can be found through eigenvector-eigenvalue methods, but it is certainly more problematic to obtain than the simple symmetric case on the previous slide.



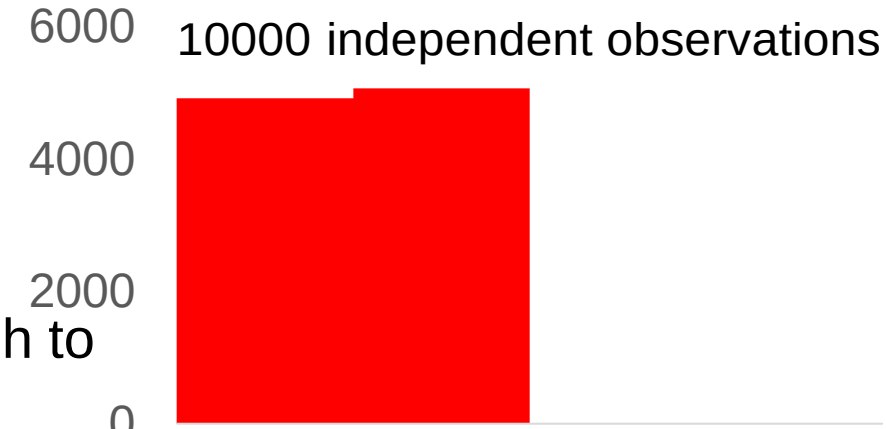
Equilibrium Distributions Through Simulation

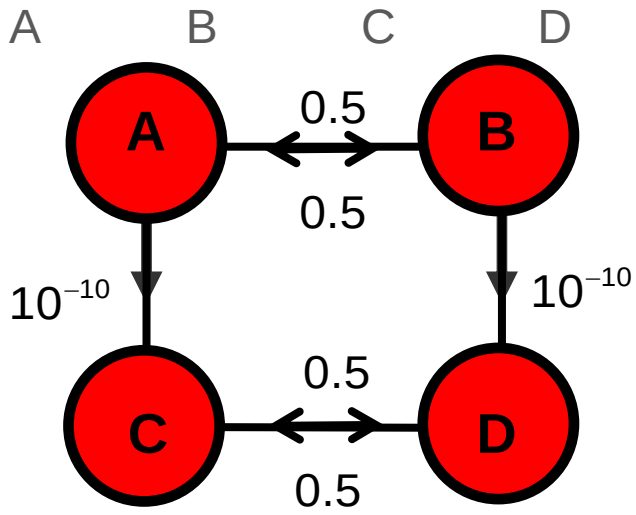
- Another alternative for finding the equilibrium distribution of a system is through simulation.
- By definition, the equilibrium distribution is the distribution of possible states for the system as the number of allowed moves $\rightarrow \infty$.
- We can (under certain conditions) simply run the Markov Chain for a large number of moves and build up an idea of its possible states in future.
- If we pick one large value for *n*, then observe the position of the chain, we have generated one realisation from its equilibrium distribution.
- Repeating this multiple times builds up a picture of the distribution.



Equilibrium Distributions Through Simulation

- We previously said that this method works under certain conditions.
- What problems could arise?
- The system could simply not have been running long enough to overcome its initial conditions and truly reach its equilibrium distribution.
- For example, consider the Markov Chain here.

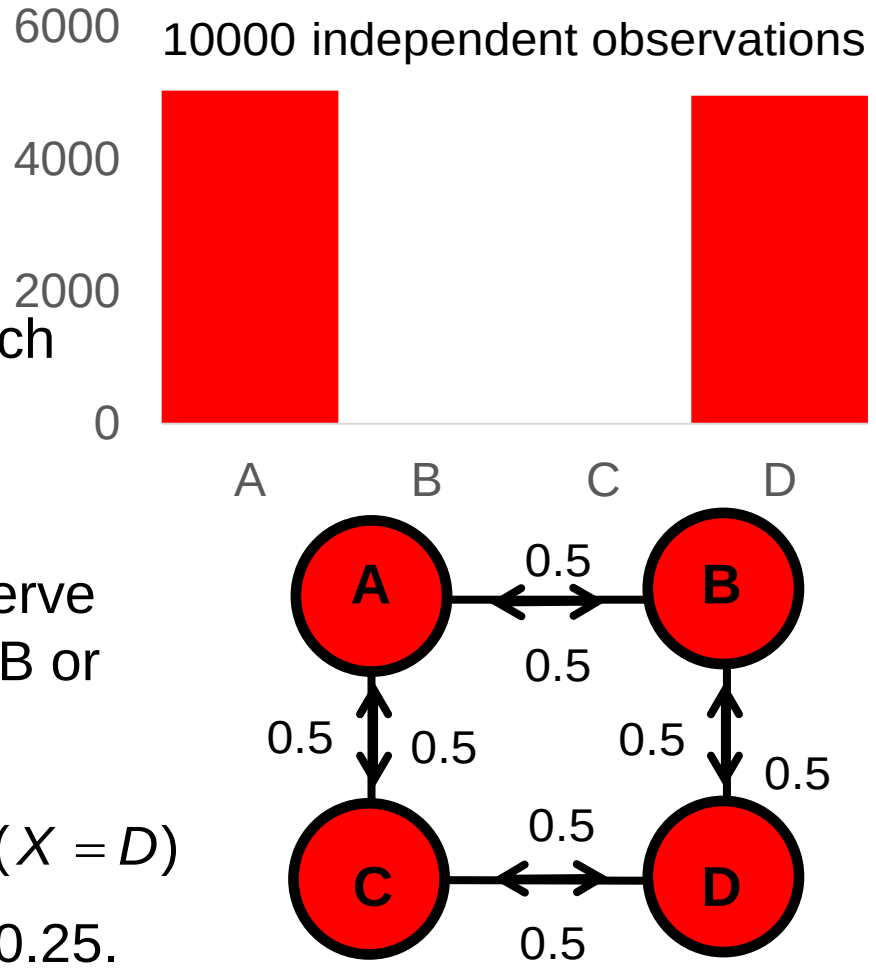




- Beginning at time 0 in state A, running for only a few thousand moves would very likely still find it in state A or state B.
- The true equilibrium destruction is $P(X = C) = 0.5 = P(X = D)$.

Equilibrium Distributions Through Simulation

- We previously said that this method works under certain conditions.
- What problems could arise?
- Consider running the Markov Chain here for 1000 moves each time, starting in state A.
- What is the chance that we observe the system in state B?
- This system is periodic, with period 2 since we can only observe state A or D after an even number of moves and only states B or C after an odd number.
- Observing after 1000 moves would imply $P(X = A) = 0.5 = P(X = D)$ when, in fact, $P(X = A) = P(X = B) = P(X = C) = P(X = D) = 0.25$.



Ergodic and Aperiodic Markov Chains

- A Markov Chain is said to be **ergodic** if every state communicates with every other.
- That is, when leaving any state, it is possible (but not necessarily in a single move) to reach any other.
- A state of a Markov Chain is **periodic** with period *d* if and only if it cannot be left and returned to unless the number of moves taken is divisible by *d*.
- If all states of the chain have period 1, then the chain is said to be **aperiodic**.
- In general, if we are to simulate a realisation from the equilibrium distribution of a chain by running the chain for a long time and picking an observation, then we want the chain to be both ergodic and aperiodic.
- The time to avoid being biased by initial conditions ("**burn in**" period) varies depending on the chain.

Markov Chain Monte Carlo (MCMC)

- The **Markov Chain Monte Carlo (MCMC)** method is one of the most powerful and commonly used tools in machine learning and statistics.
- It builds on both the idea of obtaining an equilibrium distribution for a Markov Chain through repeated simulation and of generating realisations of a variable through Monte Carlo like approaches.
- If we have to simulate samples from a “difficult” distribution, MCMC can offer an alternative approach.
- The basic idea is to construct a Markov Chain whose equilibrium distribution is equal to the problematic distribution and simulating the chain until equilibrium.
- Like ordinary Monte Carlo, the exact distribution of the samples isn’t important, but the limit is.

MCMC: Metropolis-Hastings Algorithm

- The Metropolis-Hastings algorithm is one of the most common implementations of MCMC.
- Originally developed by nuclear physicists working at the Los Alamos National Laboratory in the 1950s.
- Originally named after Nicholas Metropolis (one of five authors on the original paper.)
- The Metropolis algorithm was later refined/improved by and co-named after statistician W.K. Hastings.

Nicholas Metropolis

(1915-1999)

MCMC: Metropolis-Hastings Algorithm

- The algorithm relies on being able to sample from a (hopefully similar) and simpler distribution than the more difficult one, $P(x)$, required. This distribution $g(x)$ is the **proposal distribution**.

- The algorithm works by selecting possible moves from the proposal distribution, but only implementing them if they are accepted according to some rule. This ensures that the behaviour of the chain resembles samples from the required distribution.

- If the chain is currently in state x and the proposal distribution proposes a move to state x_p, then this proposed move is only accepted with probability $\min\left\{1, \left(\frac{P(x_p)}{P(x)}\frac{g(x|x_p)}{g(x_p|x)}\right)\right\}$.

MCMC: Metropolis-Hastings Algorithm

- A proposed move is only accepted with probability $\min\left\{1, \left(\frac{P(x_p)}{P(x)}\frac{g(x|x_p)}{g(x_p|x)}\right)\right\}$.

- In other words, if the proposal distribution selects a move which is more likely than the current state, then the chain moves to it.

- If the chain selects a less likely move, it moves to it with a probability <1.

MCMC: Metropolis-Hastings Algorithm

- The algorithm works as follows:

1) Select a starting state x at random.

2) Propose a possible move by simulating from $g(x_p|x)$.

3) Generate a realisation u of $U[0,1]$.

4) If $u < \min\left\{1, \left(\frac{P(x_p)}{P(x)}\frac{g(x|x_p)}{g(x_p|x)}\right)\right\}$, then accept the move and update the position of the system from x to x_p. If not, reject the move and return to step 2.

5) Record the move if accepted and return to step 2.

MCMC: Metropolis-Hastings Algorithm

- The exact behaviour of the chain can vary quite a lot depending on the choice of proposal distribution $g(x)$.
- For example, a poor choice of $g(x)$ might lead to proposing a lot of moves which are rejected. This would lead to extremely slow running and convergence.
- The original Metropolis algorithm required the proposal distribution to be symmetric i.e. $g(x|x_p) = g(x_p|x)$. This simplifies the acceptance probability to $\frac{P(x_p)}{P(x)}$.
- In the extreme (but obviously not very useful) case when $g(x) = P(x)$, we can see that we have the simple Markov Chain random walk rule that all proposed moves are accepted.

MCMC: Metropolis-Hastings Algorithm

- The Markov Chain Monte Carlo method allows us to generate samples from a distribution $\pi(x)$ by constructing a Markov Chain whose equilibrium distribution is $\pi(x)$ and sampling from its long-run distribution instead.

- We start by constructing a Markov Chain which is reversible. That is, for every two states x and x_p, the probability of moving from x to x_p is equal to the probability of moving from x_p to x.

- This gives $\pi(x)\pi(x_p|x) = \pi(x_p)\pi(x|x_p)$

- However, if we sample from a separate proposal distribution $g(x)$ and only accept moves according to some rule, we have that $\pi(x_p|x) = g(x_p|x)A(x_p|x)$ where $A(x_p|x)$ is the probability that a proposed move from x_p to x is accepted.

MCMC: Metropolis-Hastings Algorithm

- Substituting $\pi(x_p|x) = g(x_p|x)A(x_p|x)$ into $\pi(x)\pi(x_p|x) = \pi(x_p)\pi(x|x_p)$ gives

  $$\pi(x)g(x_p|x)A(x_p|x) = \pi(x_p)\pi(x|x_p) = \pi(x_p)g(x|x_p)A(x|x_p)$$

- Rearranging gives $\dfrac{A(x_p|x)}{A(x|x_p)} = \dfrac{\pi(x_p)g(x|x_p)}{\pi(x)g(x_p|x)}$

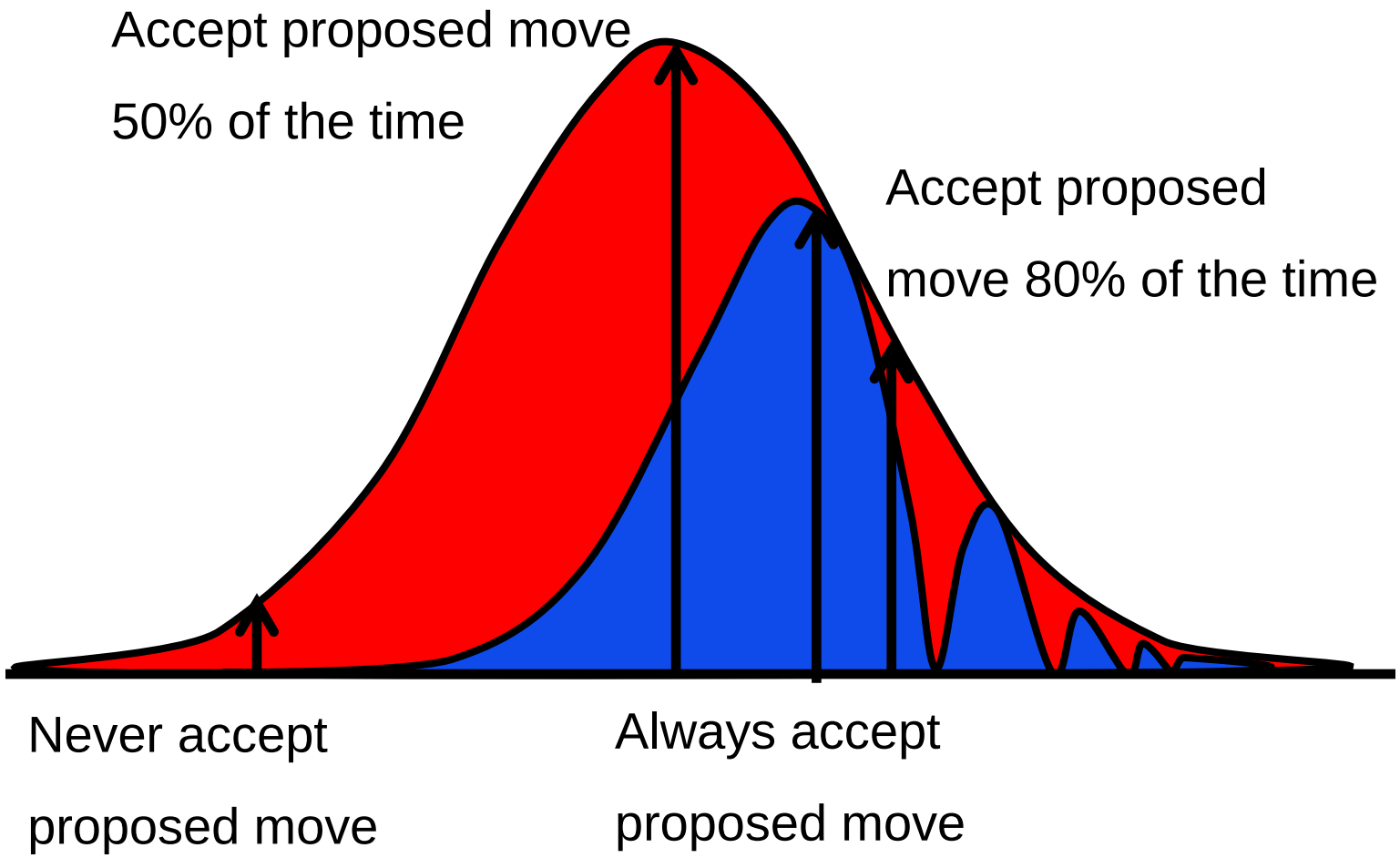
- Since *A* has to be a probability (i.e. cannot be larger than 1 or smaller than 0), we can choose

  $A(x_p|x) = \min\left\{1, \dfrac{\pi(x_p)g(x|x_p)}{\pi(x)g(x_p|x)}\right\}$ as our acceptance probability.

- This is the acceptance probability for the Metropolis-Hastings algorithm.

MCMC: Metropolis-Hastings Algorithm

- Consider the example of sampling from a “difficult” distribution (blue shading) through MCMC methods using the “easier” proposal distribution (red shading.)

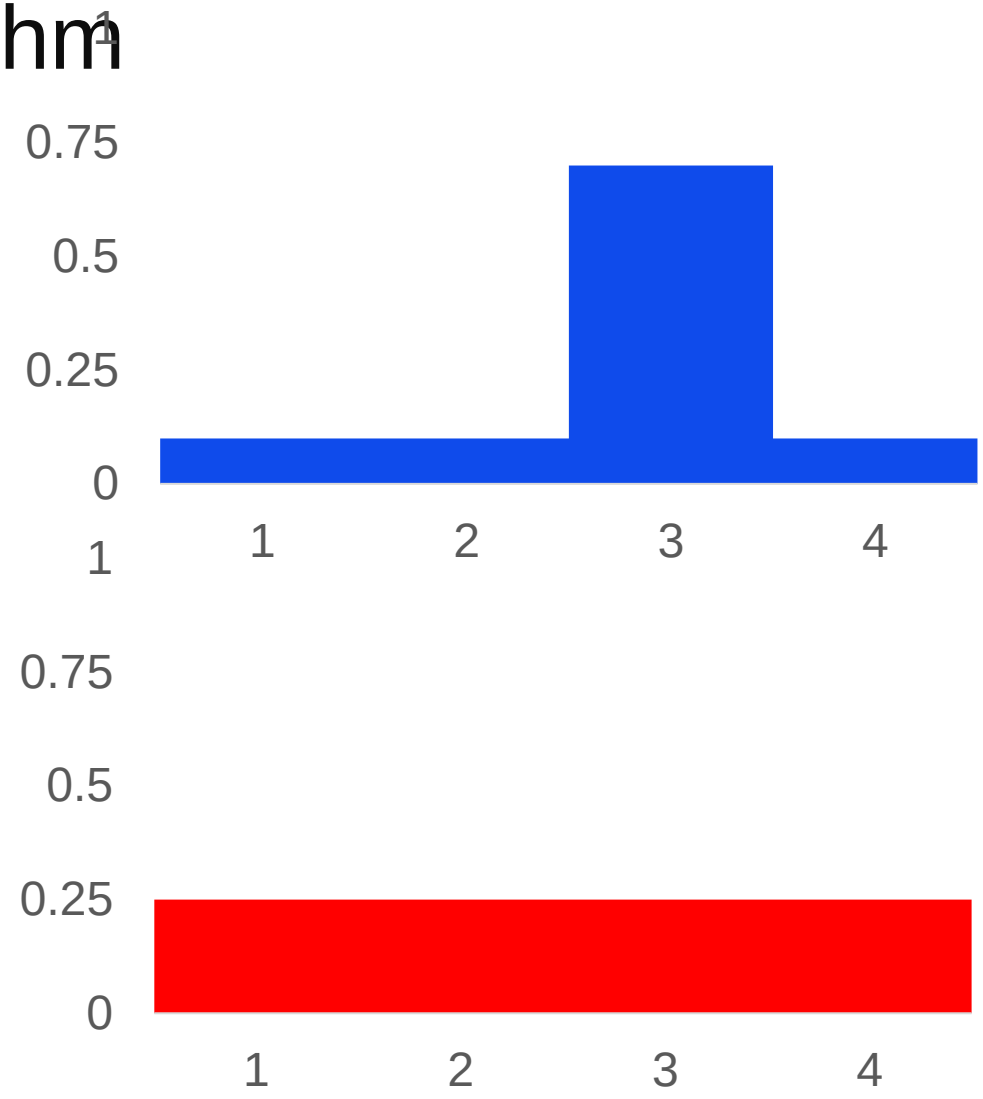


MCMC: Metropolis-Hastings Algorithm

- Consider the example of sampling from the discrete distribution with probability mass function

$$P(X=k)=\begin{cases}0.1 & k\in\{1,2,4\}\\ 0.7 & k=3\\ 0 & \text{otherwise}\end{cases}$$ using

proposal distribution $$P(Y=k)=\begin{cases}0.25 & k\in\{1,2,3,4\}\\ 0 & \text{otherwise}\end{cases}$$



- At each step, the proposal distribution suggests moving the chain to one of the 4 possible states with all 4 equally likely.

- Not every proposed move is accepted, however.

MCMC: Metropolis-Hastings Algorithm

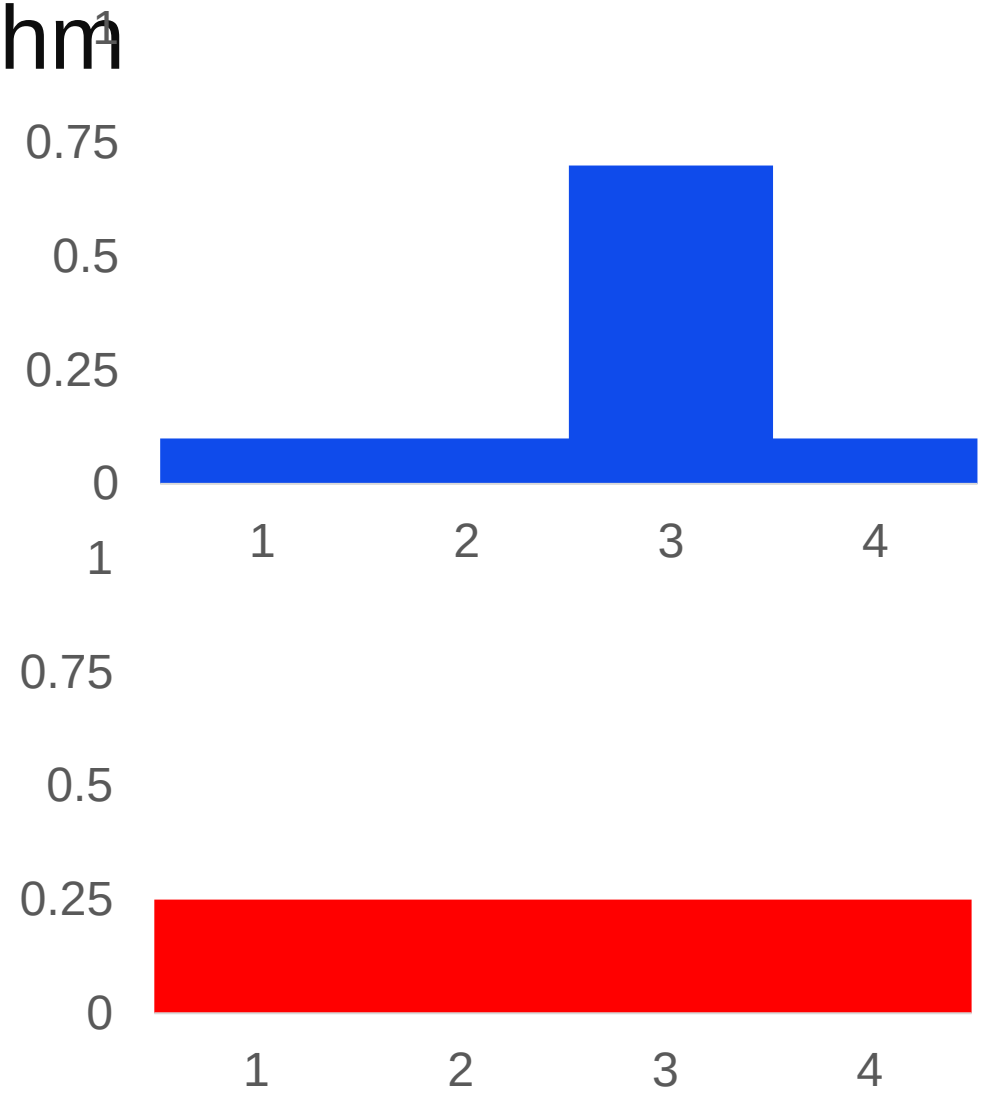
- If the chain is in state 1, all proposed moves are accepted, since whatever the proposed move x_p is

$$\frac{P(x_p)}{P(1)} = \begin{cases} \frac{0.1}{0.1} & x_p \in \{1,2,4\} \\ \frac{0.7}{0.1} & x_p = 3 \end{cases}$$

and hence

$$\min\left\{1, \left(\frac{P(x_p)}{P(x)}\frac{g(x|x_p)}{g(x_p|x)}\right)\right\} = 1.$$

- Note, here all possible moves are proposed with probability 0.25 hence $\frac{g(x|x_p)}{g(x_p|x)} = \frac{0.25}{0.25} = 1.$



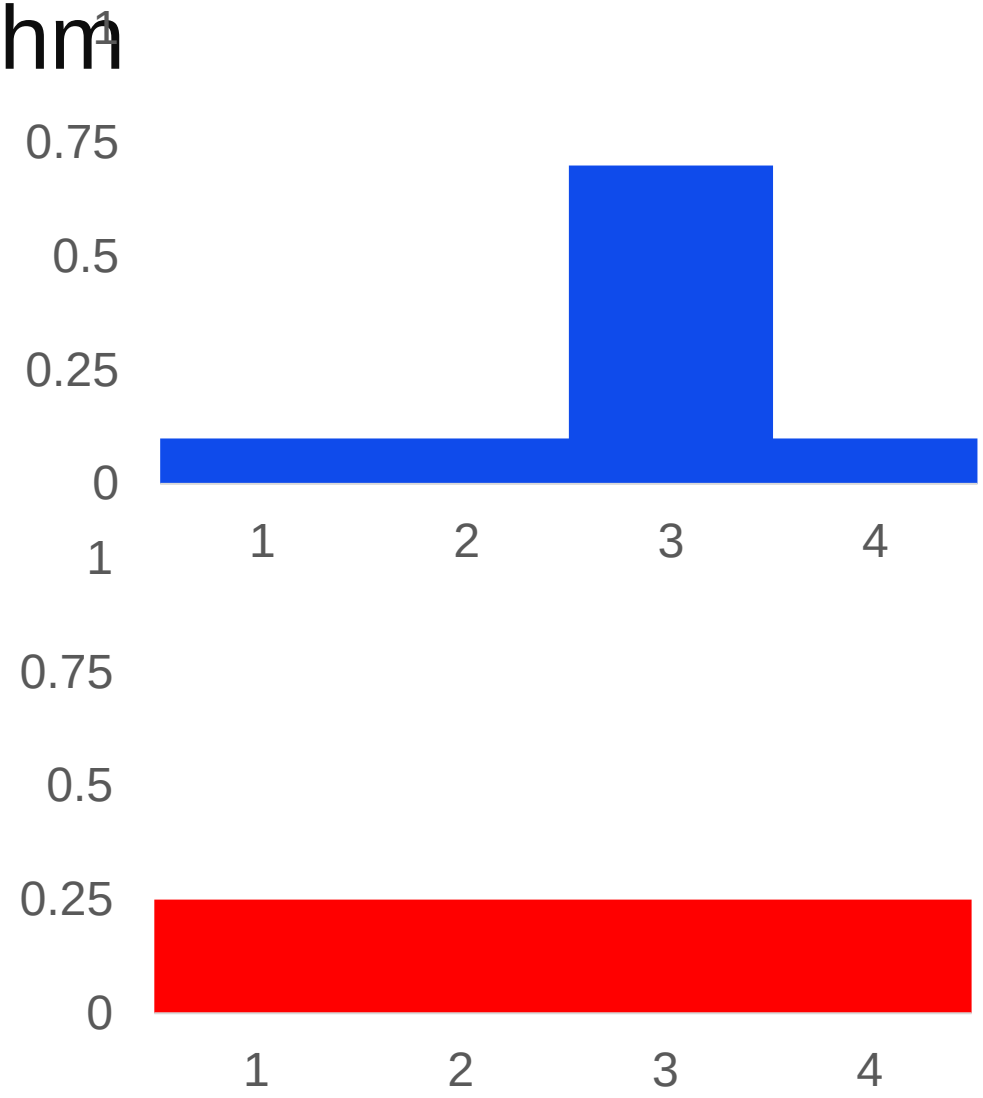
MCMC: Metropolis-Hastings Algorithm

- By the same argument, if the chain is in state 2 or 4, all proposed moves are accepted.

- If the chain is in state 3, a proposed move x_p is

accepted with probability $\frac{P(x_p)}{P(3)} = \begin{cases} \frac{0.1}{0.7} & x_p \in \{1,2,4\} \\ \frac{0.7}{0.7} & x_p = 3 \end{cases}$

so every time the chain proposes staying in state 3, it does. Every time it proposes moving to another state, only does so with probability $\frac{1}{7}$.

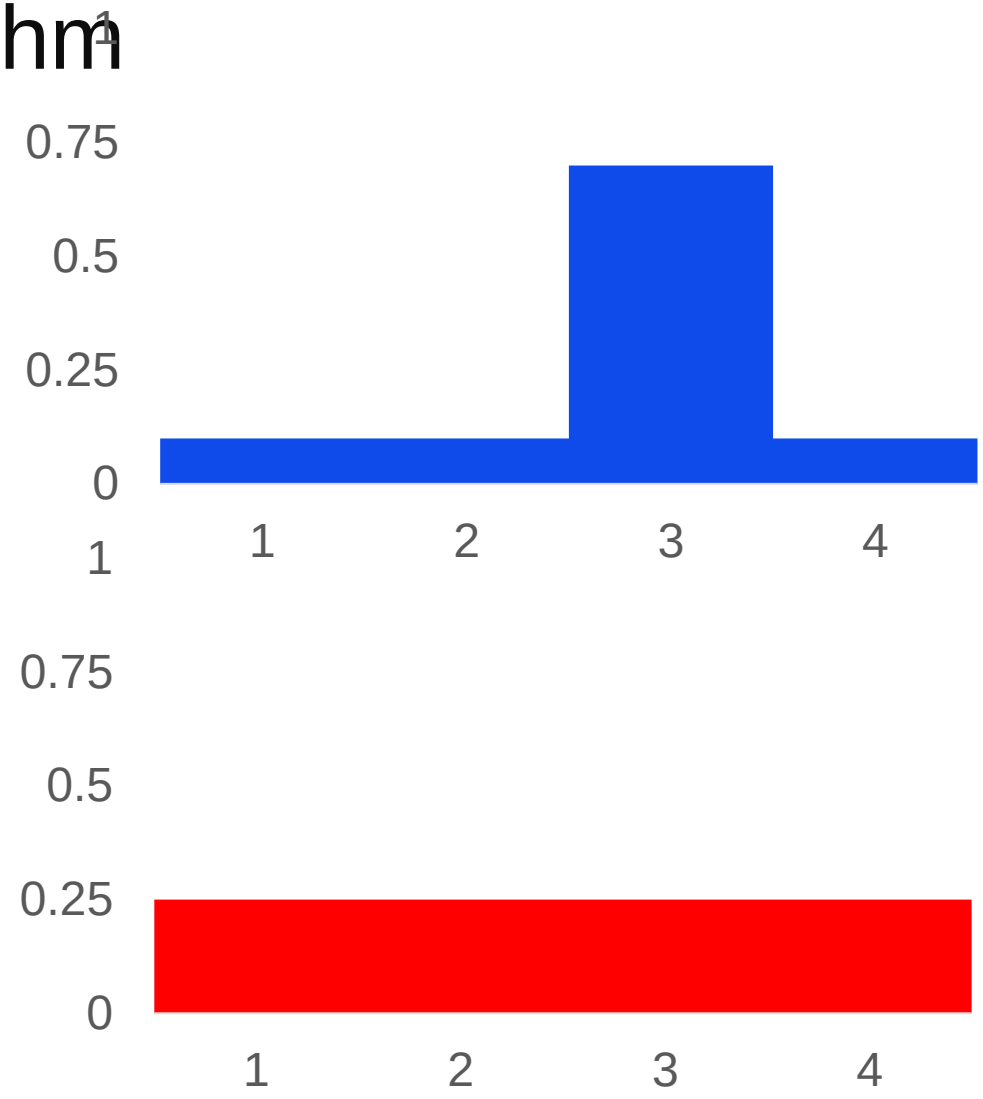


MCMC: Metropolis-Hastings Algorithm

- This gives a transition matrix for the chain

$$\begin{pmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ \frac{0.25}{7} & \frac{0.25}{7} & \frac{6.25}{7} & \frac{0.25}{7} \\ 0.25 & 0.25 & 0.25 & 0.25 \end{pmatrix}$$

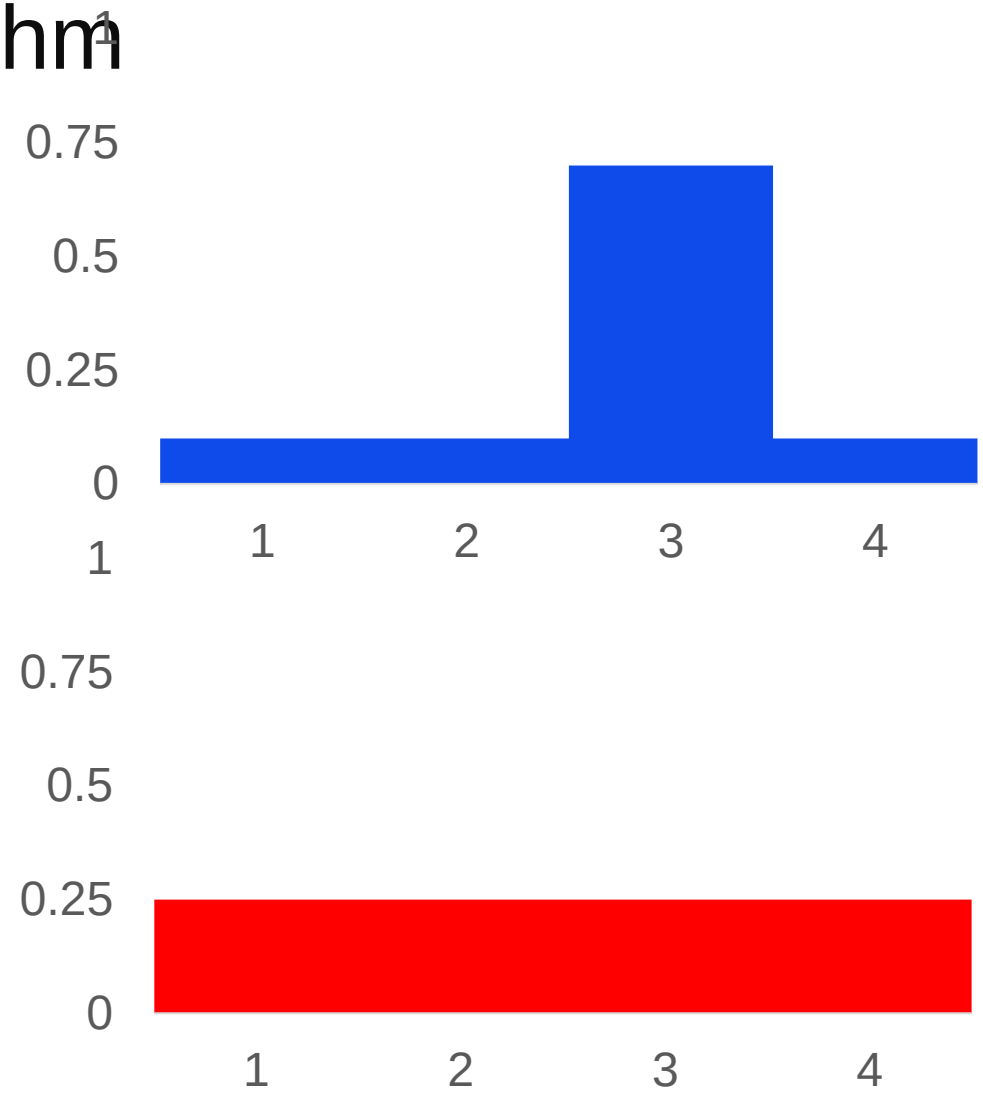
- We can verify that this gives the correct equilibrium distribution (i.e. the probability mass function of X) for the chain by eigenvector-eigenvalue methods.



MCMC: Metropolis-Hastings Algorithm

- $\begin{pmatrix} 0.1 & 0.1 & 0.7 & 0.1 \end{pmatrix} \begin{pmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ \frac{0.25}{7} & \frac{0.25}{7} & \frac{6.25}{7} & \frac{0.25}{7} \\ 0.25 & 0.25 & 0.25 & 0.25 \end{pmatrix}$

$= \begin{pmatrix} 0.1 & 0.1 & 0.7 & 0.1 \end{pmatrix}$



Bayes' Theorem

- In some contexts, we might know the conditional probability of one event given another but, in fact, we require the exact opposite.

- For example, with disease screening, we might know how likely a scan is to pick up a disease if the patient has it, but we really want to know the other way round – if the scan has "seen" the disease, how likely is the patient to be sick? (Or similarly, how likely is the "all clear" to be false hope?)

- $P(A \cap B) = P(A|B) \times P(B) = P(B|A) \times P(A)$ so $P(A|B) = \dfrac{P(B|A) \times P(A)}{P(B)}$.

- This is known as **Bayes' Theorem**.

Rev. Thomas Bayes

(1701-1761)

Bayes' Theorem

- The continuous equivalent of this is $f(\theta|x) = \frac{f(x|\theta)f(\theta)}{f(x)}$.
- This, however, requires the calculation of the denominator, which may well be difficult to do analytically, $f(x) = \int_{\Theta} f(x,\theta)d\theta$.
- Using MCMC we can, however, still obtain simulated draws from this distribution without first working out this integral.
- The acceptance probability relies only on a ratio of probabilities and not the probabilities themselves, so we can ignore the (possibly problematic) denominator, since it will cancel out. The ratio of two fractions with equal denominators is simply equal to the ratio of the numerators.
- This simplification allows MCMC to be widely used in Bayesian statistics for machine learning.