# STAT5003

## Week 11 : Markov Chain Monte Carlo

Dr. Justin Wishart

Semester 2, 2020

THE UNIVERSITY OF SYDNEY

# Markov Chain Monte Carlo

# Markov Chain Monte Carlo

- Markov Chain Monte Carlo (MCMC) is a Monte Carlo sampling technique for generating samples from an arbitrary distribution
- The difference between MCMC and Monte Carlo simulation from last week is that it uses a Markov Chain
- Two popular implementations of MCMC are
  - Metropolis-Hastings algorithm (core by Metropolis, Rosenbluth, Rosenbluth, Teller, and Teller (1953) and generalization by Hastings (1970))
  - Gibbs samplers.

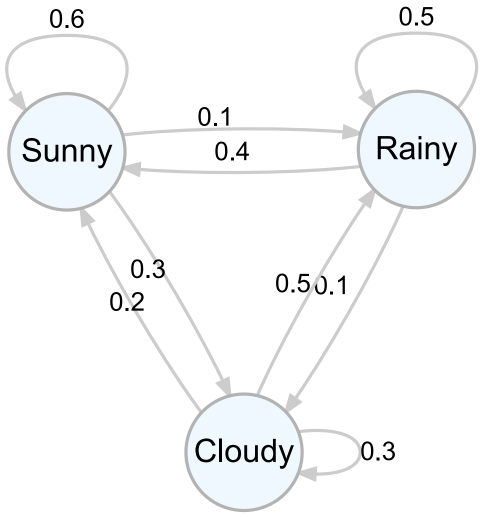
# Markov Chains

# What are Markov Chains?

- Markov chain is a stochastic process that follows the Markov property
- Markov property means that the future state of the process only depends on the current state
  - Consider a **dependent sequence** where each point only depends on the immediate past.
  - Sequence $\{X_1, X_2, \ldots, X_n\}$
  - Probabilities $P(X_n|X_{n-1}, X_{n-2}, \ldots, X_1) = P(X_n|X_{n-1})$
- Almost Memory-less system

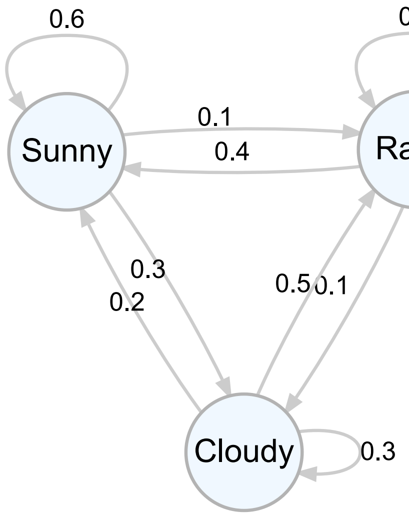
# Markov state diagrams

- Represent states as the vertices of the graph
- Edges represent the probability of moving from one state to another state
    - e.g. if it is sunny today, 10% chance of being rainy tomorrow
- Can use this state diagram to construct a sequence of states



# Transition Probability Matrix

$$P = \begin{pmatrix} 0.6 & 0.1 & 0.3 \\ 0.4 & 0.5 & 0.1 \\ 0.2 & 0.5 & 0.3 \end{pmatrix}$$



# Transition Probability Matrix

- Start with a sunny day on day 0

$$p_0 = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$$

$$p_1 = p_0 P = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0.6 & 0.1 & 0.3 \\ 0.4 & 0.5 & 0.1 \\ 0.2 & 0.5 & 0.3 \end{pmatrix}$$

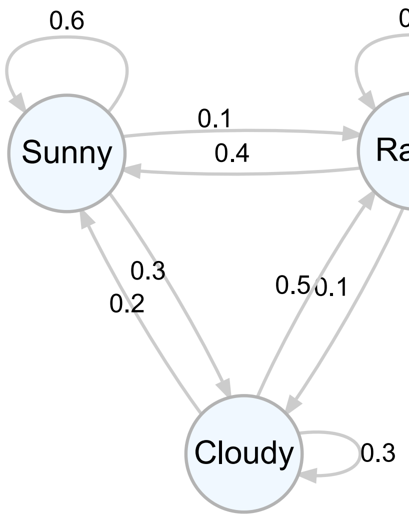
$$p_1 = \begin{pmatrix} 0.6 & 0.1 & 0.3 \end{pmatrix}$$

$$p_2 = \begin{pmatrix} 0.46 & 0.26 & 0.28 \end{pmatrix}$$

$$p_3 = \begin{pmatrix} 0.436 & 0.316 & 0.248 \end{pmatrix}$$

$$p_4 = \begin{pmatrix} 0.4376 & 0.3256 & 0.2368 \end{pmatrix}$$

- Eventually converges to an invariant distribution



# Invariant distribution

- For regular Markov chains, the probability vector $p_t$ converges to the invariant distribution $\pi$ in the limit
- Can also be represented as:

$$\pi = \pi P$$

- This is satisfied if the Markov chain is :

1. **Irreducible** - i.e. there is a path from every vertex to every other vertex
2. **Aperiodic** – i.e. there are no loops in the Markov chain. If this is not satisfied, then the system will oscillate

# MCMC - Metropolis-Hasting algorithm

# Metropolis-Hasting algorithm - Intuition

- Travelling politician problem
- Imagine you are a politician trying to visit all the town halls in your electorate and you want to spend time proportional to the number of voters in each town hall
- You start at a random town hall
- Choose the next town hall to visit
  - If the new town hall has more voters than your current town hall, then go there
  - If not, then go there with a probability that is equal to Number of people in new town hall / Number of people in current town hall

# Metropolis-Hastings algorithm

- Similar to the acceptance-rejection method
  - it simulates a trial state
  - accepts or rejects it according to some random mechanism
- Uses the Markov chain because each trial state depends on the previous state – almost memoryless system.
- Aim is to construct a Markov chain $X_t$ , $t = 0, 1, \ldots$ such that the limiting distribution is $f(x)$

# Metropolis-Hastings algorithm

Initialise state to $X_0$. Require as input a target pdf $f(x)$ and a proposal pdf $q(x, y)$

For $t = 0, 1, \ldots, N-1$ do:

- Draw $Y \sim q(x|X_t)$
- Calculate acceptance probability $\alpha(X_t, Y)$
- Define $\alpha(x, y) = \min\left\{\frac{f(y)q(x|y)}{f(x)q(y|x)}\right\}$
- Draw $U \sim U(0, 1)$
- if $U \leq \alpha$ then $X_{t+1} \leftarrow Y$ else $X_{t+1} \leftarrow X_t$

Return $X_1, X_2, \ldots, X_N$

# Proposal function

- If the proposal density function is symmetric,
  - $q(y|x) = q(x|y)$
  - the acceptance probability has a simpler form.
  - the MCMC algorithm is also known as a **random walk sampler**.
- One common choice of a symmetric proposal function is just the Gaussian function i.e. $q(x)\ \mathcal{N}(x_t, \sigma)$
- The choice of $\sigma$ affects how quickly the state space is explored.

# Where would you use MCMC

- One common application of MCMC is to draw from the posterior distribution in Bayesian statistical methods.

$$P(A|B) = \frac{P(B|A)}{P(B)} P(A)$$

- Posterior: The likelihood of $A$ occuring given $B$ has occurred.
- Likelihood ratio: The support $B$ provides for $A$
- Prior: The probability of $A$ before any data is gathered.

# The posterior distribiution

- Can use the Bayes rule for modelling and data.

$$P(\phi|D) = \frac{P(D|\phi)}{P(D)} P(\phi)$$

- Posterior: The likelihood of $\phi$ occuring given the data $D$.
- Likelihood ratio: The support $D$ provides for $\phi$
- Prior: The probability of $\phi$ before any data is gathered.

- Typically $P(D)$ is a difficult integral to evaluate.

$$P(D) = \int P(D|\phi)P(\phi)\,d\phi$$

# Estimating posterior with MCMC

- In the Metropolis-Hastings algorithm, we only need to calculate

$$\alpha = \frac{P(\phi'|D)}{P(\phi|D)} = \frac{P(D|\phi')P(\phi')}{P(D|\phi)P(\phi)}$$

- Since $P(D)$ doesn't depend on $\phi$, it cancels out on the right hand side of the above formula and hence it isn't included in the formula.

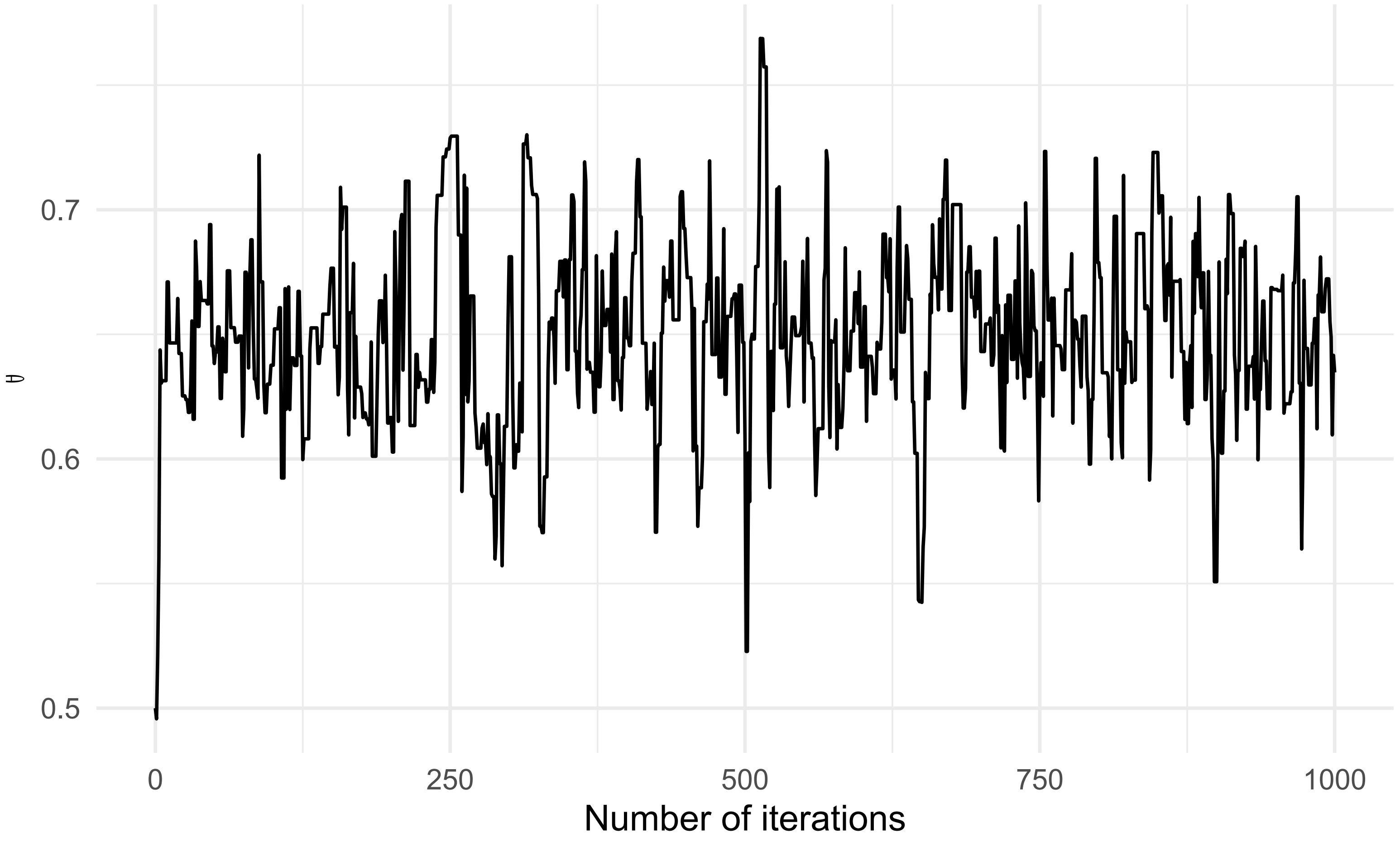
# Example

Observe a series of coin flips

H, T, H, H, T, H, T, H, H, T, H, H, H, T, H, H, H, T, H, H, ...

> Can you estimate the $P(Head)$ of this coin?
>
> Assume you don't know anything about this coin and it could be biased!

# Estimate of $P(Head)$ using MCMC



# MCMC Practical considerations

- The samples at the start of the MCMC chain, before the algorithm converges to the true distribution are known as the burn-in period.
    - It should be discarded
- The samples generated by MCMC are correlated since they are from a Markov chain.
    - Previously, many practitioners advocated **thinning** the samples by taking say every $k^{\text{th}}$ sample.
    - This was done for a few reasons historically
        - Reduce correlations and compute standard errors more easily
        - Less space needed to store the chain.

# References

Hastings, W. K. (1970). "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1, pp. 97-109. ISSN: 0006-3444. DOI: 10.1093/biomet/57.1.97. eprint: https://academic.oup.com/biomet/article-pdf/57/1/97/23940249/57-1-97.pdf. URL: https://doi.org/10.1093/biomet/57.1.97.

Metropolis, N, A. W. Rosenbluth, M. N. Rosenbluth, et al. (1953). "Equation of State Calculations by Fast Computing Machines". In: *The Journal of Chemical Physics* 21.6, pp. 1087-1092. DOI: 10.1063/1.1699114. eprint: https://doi.org/10.1063/1.1699114. URL: https://doi.org/10.1063/1.1699114.