Sampling with Markov chain Monte Carlo

Feng Li

feng.li@cufe.edu.cn

School of Statistics and Mathematics
Central University of Finance and Economics

Revised on April 22, 2020

Today we are going to learn...

1. Markov Chains
2. Metropolis Algorithm
3. Metropolis-Hastings
4. Multiple variables

Markov Chains

- The goal of today's lecture is to learn about the **Metropolis Hastings algorithm**
- The Metropolis Hastings algorithm allows us to simulate from any distribution as long as we have the kernel of the density of the distribution.
- To understand the Metropolis Hastings algorithm, we must learn a little bit about **Markov chains**

Basic Probability Rules

- **Law of conditional probability**

$$\Pr(A = a, B = b) = \Pr(A = a|B = b)\Pr(B = b) \tag{1}$$

- More general conditional probability

$$\Pr(A = a, B = b|C = c) = \Pr(A = a|B = b, C = c) \times \Pr(B = b|C = c) \tag{2}$$

Basic Probability Rules

- **Marginalizing** (for a discrete variable)

$$\Pr(A = a) = \sum_b \Pr(A = a, B = b) \tag{3}$$

- More general

$$\Pr(A = a|C = c) = \sum_b \Pr(A = a, B = b|C = c) \tag{4}$$

Independence

- Two variables are **independent** if

$$\Pr(A = a, B = b) = \Pr(A = a)\Pr(B = b) \quad \forall a, b \tag{5}$$

- Dividing both sides by Pr(B=b) gives

$$\Pr(A = a|B = b) = \Pr(A = a) \quad \forall a, b \tag{6}$$

Conditional Independence

- Two variables A and B are **Conditionally Independent** if

$$\Pr(A = a, B = b|C = c) = \Pr(A = a|C = c) \times \Pr(B = b|C = c) \quad \forall a, b, c \tag{7}$$

- Dividing both sides by $\Pr(B = b|C = c)$ gives

$$\Pr(A = a|B = b, C = c) = \Pr(A = a|C = c) \quad \forall a, b, c \tag{8}$$

A simple game

- Player A and Player B play a game. The probability that Player A wins each game is 0.6 and the probability that Player B wins each game is 0.4.
- They play the game N times.
- Each game is **independent**.
- Let
  - $X_i = 0$ if Player A wins game i
  - $X_i = 1$ if Player B wins game i
- Also assume there is an initial Game called Game 0 (X_0)

Some simple questions

- What is the probability that Player A wins Game 1 ($(X_1 = 0)$) if
  - If $X_0 = 0$ (Player A wins Game 0)
  - If $X_0 = 1$ (Player B wins Game 0)
- What is the probability that Player A wins Game 2 ($(X_2 = 0)$) if
  - If $X_0 = 0$ (Player A wins Game 0)
  - If $X_0 = 1$ (Player B wins Game 0)
- Since each game is independent all answers are 0.6.

A different game: A Markov chain

- Now assume that both players have a better chance of winning Game $i+1$ if they already won Game i.

$$\Pr(X_{i+1} = 0 | X_i = 0) = 0.8 \tag{9}$$
$$\Pr(X_{i+1} = 1 | X_i = 1) = 0.7 \tag{10}$$

- Assume nothing other than game i has a direct effect on Game $i+1$.
- This is called the **Markov Property**. Mathematically

$$\Pr(X_{i+1} | X_i, X_{i-1}, \ldots, X_1, X_0) = \Pr(X_{i+1} | X_i) \tag{11}$$

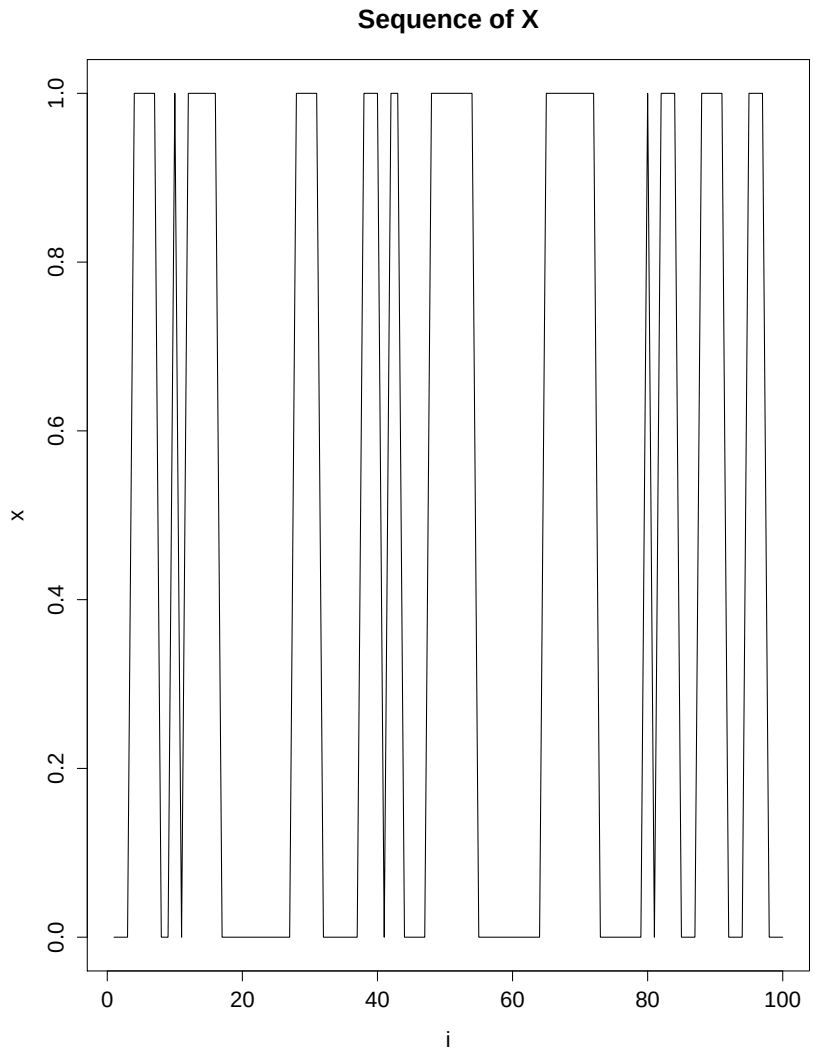
Markov Property

- Another way to define the Markov property is to notice that X_{i+1} and $X_{i-1}, \ldots, X_0$ are **independent** conditional on X_i
- This may be a model for the stock market, all the valuable information about tomorrow's stock price is contained in today's price.
- This is related to the **Efficient Market Hypothesis**, a popular theory in finance.
- Now back to the simple game.

Simulating from a Markov chain

- Now let's simulate a sequence $X_1, X_2, \ldots, X_{100}$ from the Markov chain.
- Initialize at $x_0 = 0$. Then inside a loop
- Code the following using *if*.
  - if $X_i = 0$ then $X_{i+1} = \begin{cases} 0 & \text{with probability } 0.8 \\ 1 & \text{with probability } 0.2 \end{cases}$
  - if $X_i = 1$ then $X_{i+1} = \begin{cases} 0 & \text{with probability } 0.3 \\ 1 & \text{with probability } 0.7 \end{cases}$
- Try it

Markov chain



Simple questions again

- What is the probability that Player A wins the first game (i.e $(X_1 = 0)$) if
  - If $X_0 = 0$ (Player A wins initial game)
  - If $X_0 = 1$ (Player B wins initial game)
- The answers are 0.8 and 0.3.
- What is the probability that Player A wins the second game $(X_2 = 0)$ if
  - If $X_0 = 0$ (Player A wins initial game)
  - If $X_0 = 1$ (Player B wins initial game)

Solution

- Let $X_0 = 0$. Then $\Pr(X_2 = 0|X_0 = 0)$

$$
\begin{aligned}
&= \sum_{x_1=0,1} \Pr(X_2 = 0, X_1 = x_1|X_0 = 0) \\
&= \sum_{x_1=0,1} \Pr(X_2 = 0|X_1 = x_1, X_0 = 0)\Pr(X_1 = x_1|X_0 = 0) \\
&= \sum_{x_1=0,1} \Pr(X_2 = 0|X_1 = x_1)\Pr(X_1 = x_1|X_0 = 0) \\
&= 0.8 \times 0.8 + 0.3 \times 0.2 \\
&= 0.7
\end{aligned}
$$

- What if $X_0 = 1$?

Recursion

- Notice that the distribution of X_i depends on X_0
- The sequence is no longer independent.
- How could you compute $\Pr(X_n = 0|X_0 = 0)$ when $n = 3$, when $n = 5$, when $n = 100$?
- This is hard, but the Markov Property does make things simpler
- We can use a recursion to compute the probability that Player A wins any game.

Recursion

Note that $\Pr(X_i = 0|X_0 = 0)$

$$
\begin{aligned}
&= \sum_{x_{i-1}} \Pr(X_i = 0, X_{i-1} = x_{i-1}|X_0 = 0) \\
&= \sum_{x_{i-1}} \Pr(X_i = 0|X_{i-1} = x_{i-1}, X_0 = 0)\Pr(X_{i-1} = x_{i-1}|X_0 = 0) \\
&= \sum_{x_{i-1}} \Pr(X_i = 0|X_{i-1} = x_{i-1})\Pr(X_{i-1} = x_{i-1}|X_0 = 0)
\end{aligned}
$$

We already applied this formula when $i = 2$. We can continue for $i = 3, 4, 5, \ldots, n$

Recursion

$$\Pr(X_i = 0|X_0 = 0) = \sum_{x_{i-1}} \Pr(X_i = 0|X_{i-1} = x_{i-1})\Pr(X_{i-1} = x_{i-1}|X_0 = 0)$$

- Start with $\Pr(X_1 = 0|X_0 = 0)$
- Get $\Pr(X_1 = 1|X_0 = 0)$
- Use these in formula with $i = 2$
- Get $\Pr(X_2 = 0|X_0 = 0)$
- Get $\Pr(X_2 = 1|X_0 = 0)$
- Use these in formula with $i = 3$
- Get $\Pr(X_3 = 0|X_0 = 0)$
- $\vdots \quad \vdots \quad \vdots \quad \vdots$

Matrix Form

It is much easier to do this calculation in matrix form (especially when X is not binary). Let P be the transition matrix

	$X_i = 0$	$X_i = 1$
$X_{i-1} = 0$	$\Pr(X_i = 0 \mid X_{i-1} = 0)$	$\Pr(X_i = 1 \mid X_{i-1} = 0)$
$X_{i-1} = 1$	$\Pr(X_i = 0 \mid X_{i-1} = 1)$	$\Pr(X_i = 1 \mid X_{i-1} = 1)$

Matrix Form

In our example:

	$X_i = 0$	$X_i = 1$
$X_{i-1} = 0$	0.8	0.2
$X_{i-1} = 1$	0.3	0.7

$$P = \begin{pmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{pmatrix} \tag{12}$$

Matrix Form

Let π_i be a 1×2 row vector which denotes the probabilities of each player winning Game i conditional on the initial Game

$$\pi_i = (\Pr(X_i = 0|X_0), \Pr(X_i = 1|X_0)) \tag{13}$$

In our example if $X_0 = 0$

$$\pi_1 = (0.8, 0.2) \tag{14}$$

In our example if $X_0 = 1$

$$\pi_1 = (0.3, 0.7) \tag{15}$$

Recursion in Matrix form

- The recursion formula is

$$\pi_i = \pi_{i-1} P \tag{16}$$

  Therefore

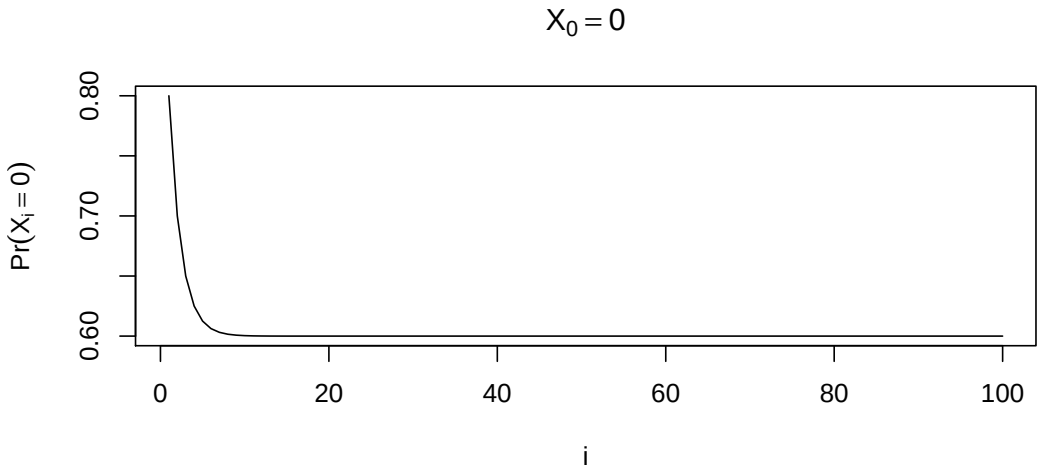
$$\pi_n = \pi_1 P \times P \times \ldots \times P \tag{17}$$

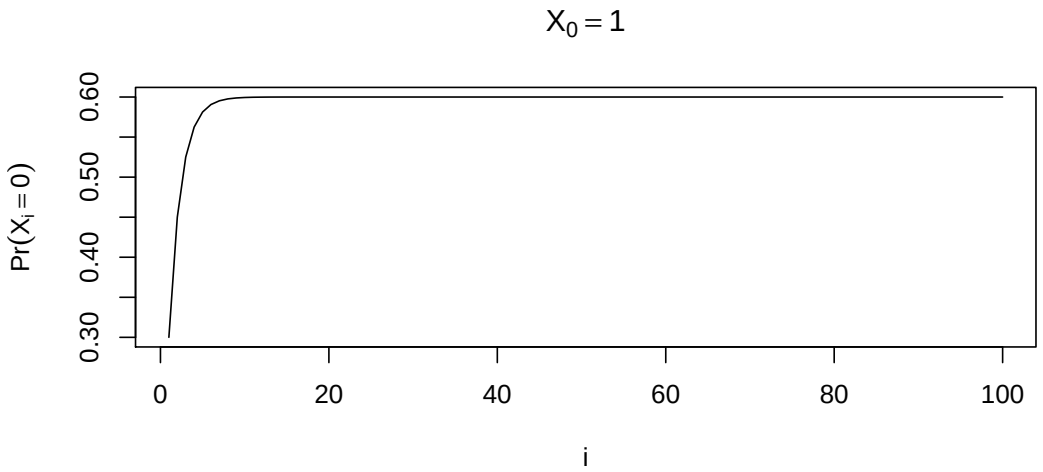
- Now code this up in R.
- What is $Pr(X_n = 0 | X_0 = 0)$ when
  - $n = 3$
  - $n = 5$
  - $n = 100$?
- Do the same when $X_0 = 1$

Convergence?

- For $n = 3$ and $n = 5$, the starting point made a big difference.
- For $n = 100$ it did not make a big difference.
- Could this Markov chain be converging to something?
- Now write code to keep the values of π_i for $i = 1, 2, \ldots, 100$.
- Then plot the values of π_{i1} against i

Convergence





More Questions

- What is $\Pr(X_{100} = 0 | X_0 = 0)$?
- What is $\Pr(X_{100} = 0 | X_0 = 1)$?
- What is $\Pr(X_{1000} = 0 | X_0 = 0)$?
- What is $\Pr(X_{1000} = 0 | X_0 = 1)$?
- The answer to all of these is 0.6.
- The X do not converge. They keep changing from 0 to 1. The Markov chain however converges to a **stationary distribution**.

Simulation with a Markov chain

- Go back to your code for generating a Markov chain and generate a chain with $n = 110000$
- Exclude the first 10000 values of X_i and keep the remaining 100000 values.
- How many $X_i = 0$? How many $X_i = 1$
- We have discovered a new way to simulate from a distribution with $Pr(X_i = 0) = 0.6$ and $Pr(X_i = 1) = 0.4$

Markov Chains

- Sometimes two different Markov chains converge to the same stationary distribution. See what happens when

$$P = \begin{pmatrix} 0.9 & 0.1 \\ 0.15 & 0.85 \end{pmatrix} \tag{18}$$

- Sometimes Markov chains do not converge to a stationary distribution at all.
- Some Markov chains can get stuck in an **absorbing state**. For example what would the simple example look like if $Pr(X_{i+1} = 0|X_i = 0) = 1$?
- Markov chains can be defined on continuous support as well, X_i can be continuous.

Some important points

- This is a very complicated way to generate from a simple distribution.
- For the binary example the direct method would be better.
- However for other examples, either the direct method or accept/reject algorithm do not work.
- In these cases we can construct a Markov chain that has a stationary distribution that is our **target distribution**.
- All we need is the kernel of the density function, and an algorithm called the **Metropolis Algorithm**

Normalizing Constant and Kernel

What are the normalizing constant and kernel of the Beta density?

$$\text{Beta}(x; a, b) = \frac{\Gamma(a+b)}{(\Gamma(a)\Gamma(b))} x^{a-1}(1-x)^{b-1} \tag{19}$$

The Metropolis algorithm

- The Metropolis algorithm was developed in a 1953 paper by Metropolis, Rosenbluth, Rosenbluth, Teller and Teller.
- The aim is to simulate $\mathrm{x} \sim \mathrm{p}(\mathrm{x})$ where $\mathrm{p}(\mathrm{x})$ is called the **target density**.
- We will need a **proposal density** $\mathrm{q}(\mathrm{x}^{[\mathrm{old}]} \to \mathrm{x}^{[new]})$
- For example one choice of q is

$$\mathrm{x}^{[new]} \sim \mathrm{N}(\mathrm{x}^{[\mathrm{old}]}, 1) \tag{20}$$

- This is called a **Random Walk proposal**

Symmetric proposal

- An important property of q in the Metropolis algorithm is symmetry of the proposal

$$q(x^{[old]} \to x^{[new]}) = q(x^{[new]} \to x^{[old]}) \tag{21}$$

- Later we will not need this assumption
- Can you confirm this is true for $x^{[new]} \sim N(x^{[old]}, 1)$?
- Can you simulate from this random walk (use $x_0 = 0$ as a starting value)?

Proof of symmetry of random walk

The proposal

$$q(x^{[old]} \to x^{[new]}) = (2\pi)^{-1/2} exp\left\{-\frac{1}{2}\left(x^{[new]} - x^{[old]}\right)^2\right\} \tag{22}$$

$$= (2\pi)^{-1/2} exp\left\{-\frac{1}{2}\left[-1\left(x^{[new]} - x^{[old]}\right)\right]^2\right\} \tag{23}$$

$$= (2\pi)^{-1/2} exp\left\{-\frac{1}{2}\left(x^{[old]} - x^{[new]}\right)^2\right\} \tag{24}$$

$$= q(x^{[new]} \to x^{[old]}) \tag{25}$$

Accept and reject

- By itself the random walk will not converge to anything.
- To make sure this Markov chain converges to our target, we need to include the following.
- At step $i+1$ set $x^{[old]} = x^{[i]}$.
- Generate $x^{[new]} \sim N(x^{[old]}, 1)$ and compute

$$\alpha = \min\left(1, \frac{p(x^{[new]})}{p(x^{[old]})}\right) \tag{26}$$

- Then
  - Set $x^{[i+1]}$ to $x^{[new]}$ with probability α (accept)
  - Set $x^{[i+1]}$ to $x^{[old]}$ with probability $1 - \alpha$ (reject)

Code it up

- Use the Metropolis algorithm with a random walk proposal to simulate a sample from the standard t distribution with 5 df.
- The target density is

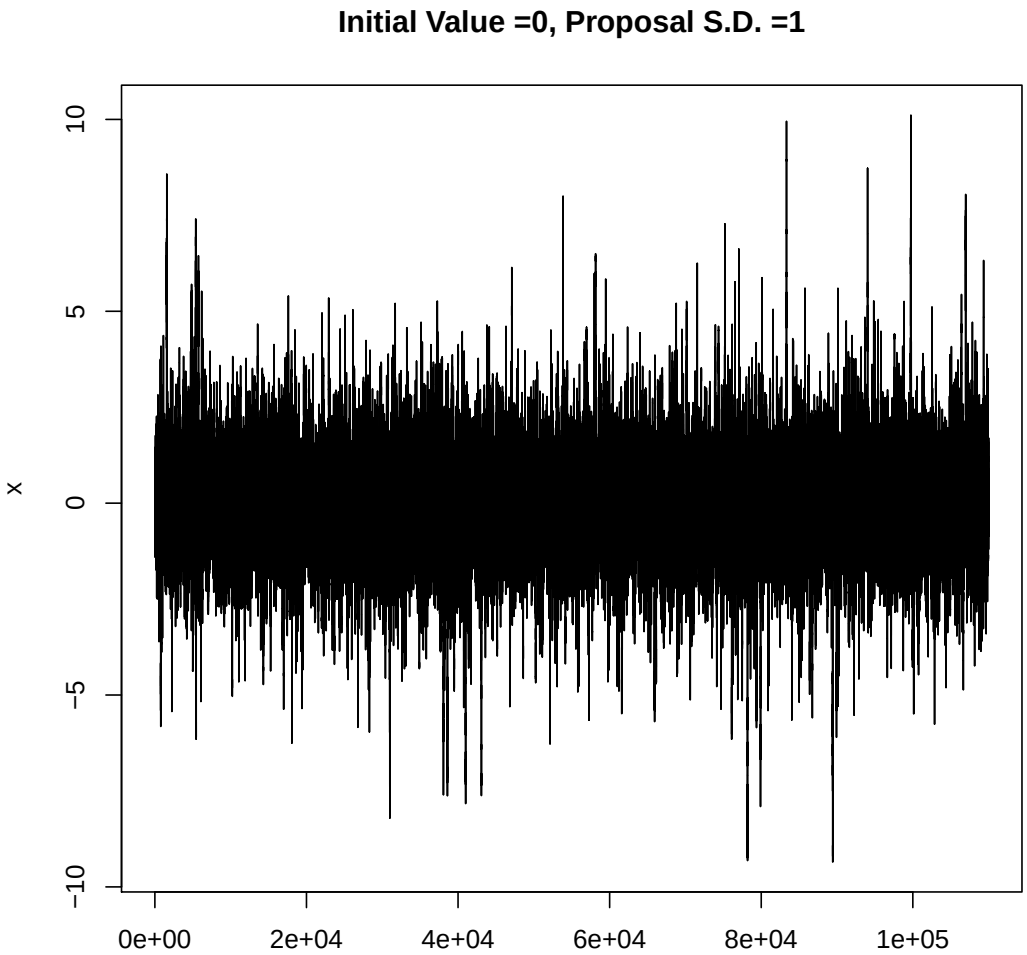
$$p(x) = \left[1 + \frac{x^2}{5}\right]^{-3} \tag{27}$$

- Simulate a Markov chain with 110000 iterates using the random walk as a proposal.
- The first 10000 iterates are the **burn-in** and will be left out because the Markov chain may not have converged yet.
- Use $x_0 = 0$ as a starting value

Some diagnostics - Convergence

- There a few diagnostics we can use to investigate the behaviour of the chain
- One is a trace plot (including burn in), which is simply a line plot of the iterates.
- Plot this for your Markov chain
- Another diagnostic is the **Geweke diagnostic** which can be found in the R package *coda*.
- The Geweke diagnostic tests the equality of the means of two different parts of the chain (excluding burn-in). The test statistic has a standard normal distribution.
- Rejecting this test is evidence that the chain has not converged

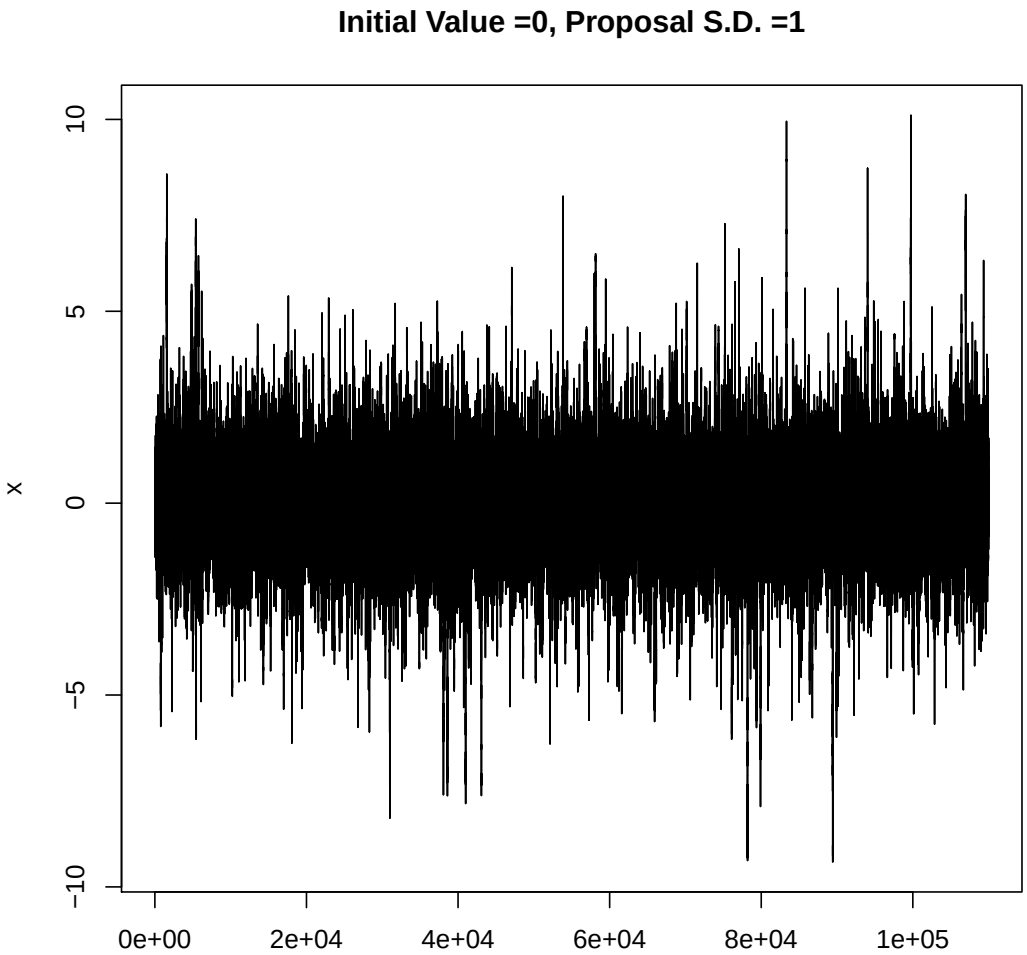
Trace Plot


Initial Value =0, Proposal S.D. =1
x
10
5
0
−5
−10
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

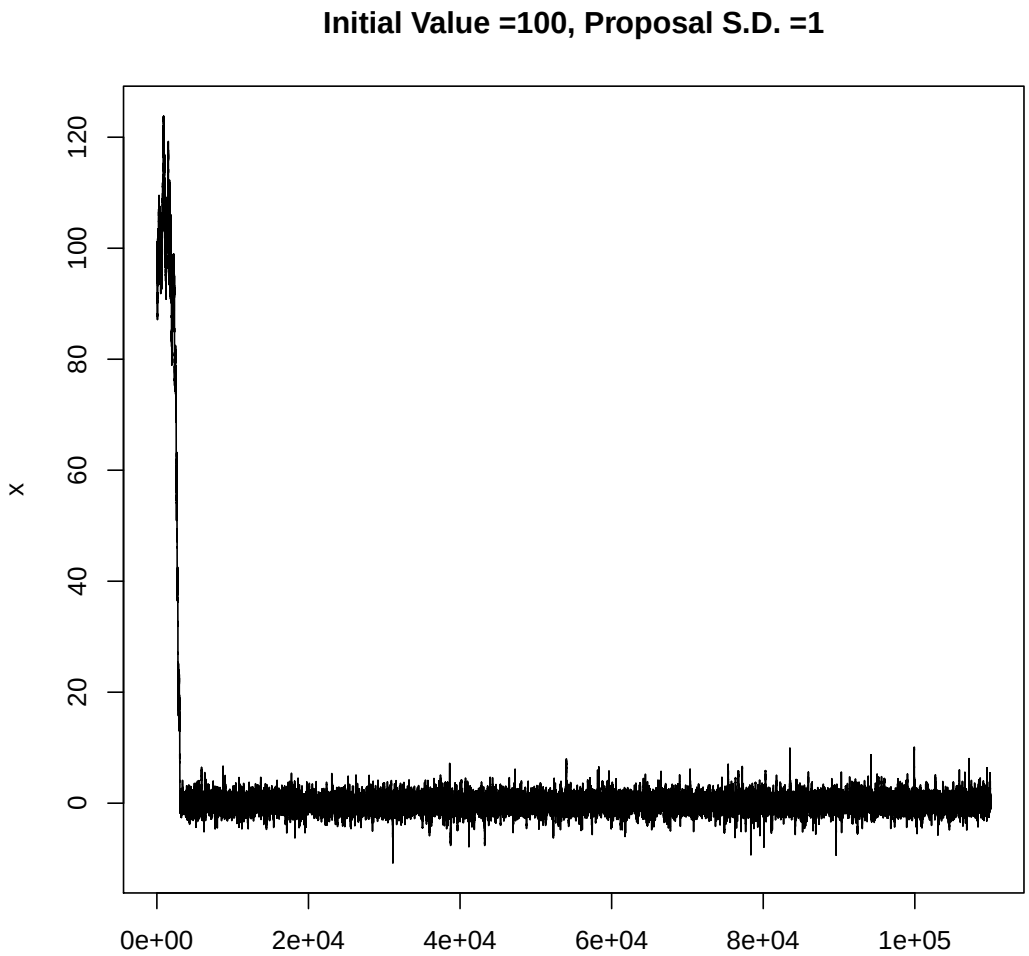
The effect of starting value

- Now rerun the code with a starting value of $X_0 = 100$
- Does the chain still converge?
- Does it converge quicker or slower?

Trace Plot


Initial Value =0, Proposal S.D. =1
x
10
5
0
−5
−10
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

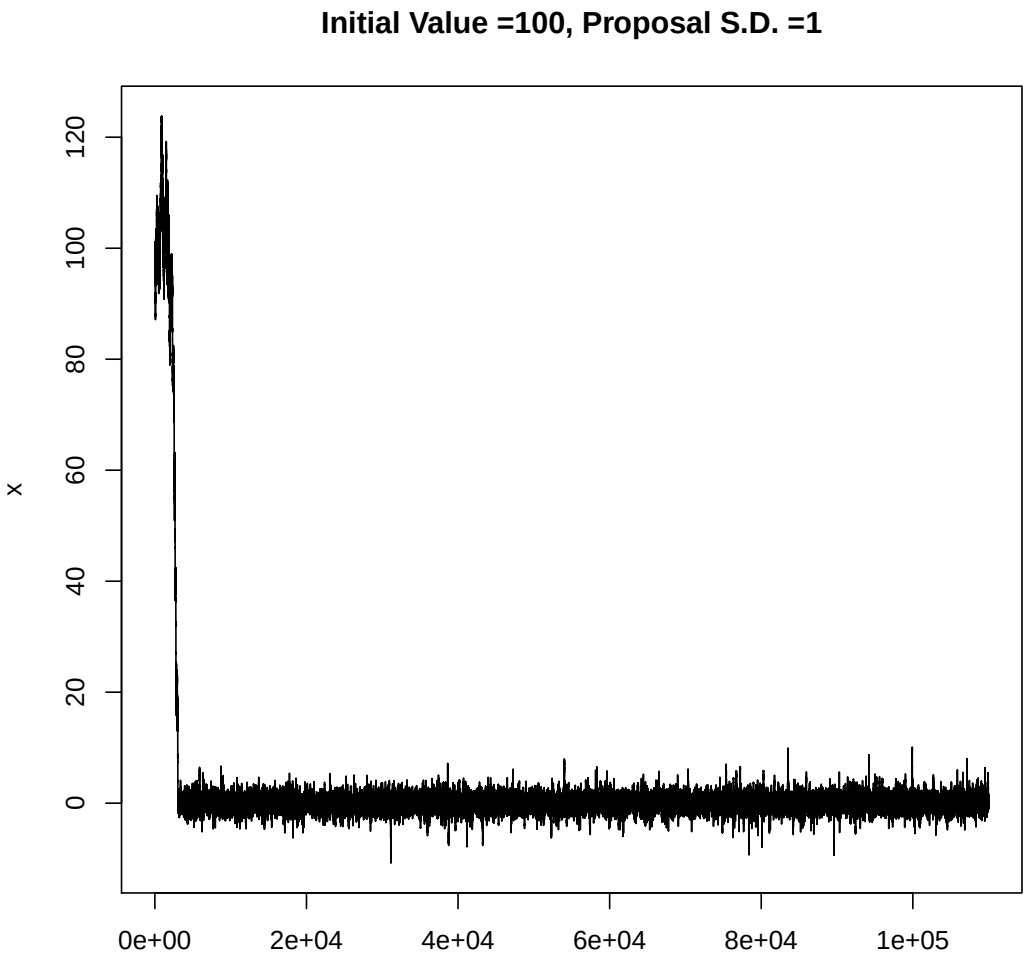
Trace Plot


Initial Value =100, Proposal S.D. =1
x
0
20
40
60
80
100
120
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

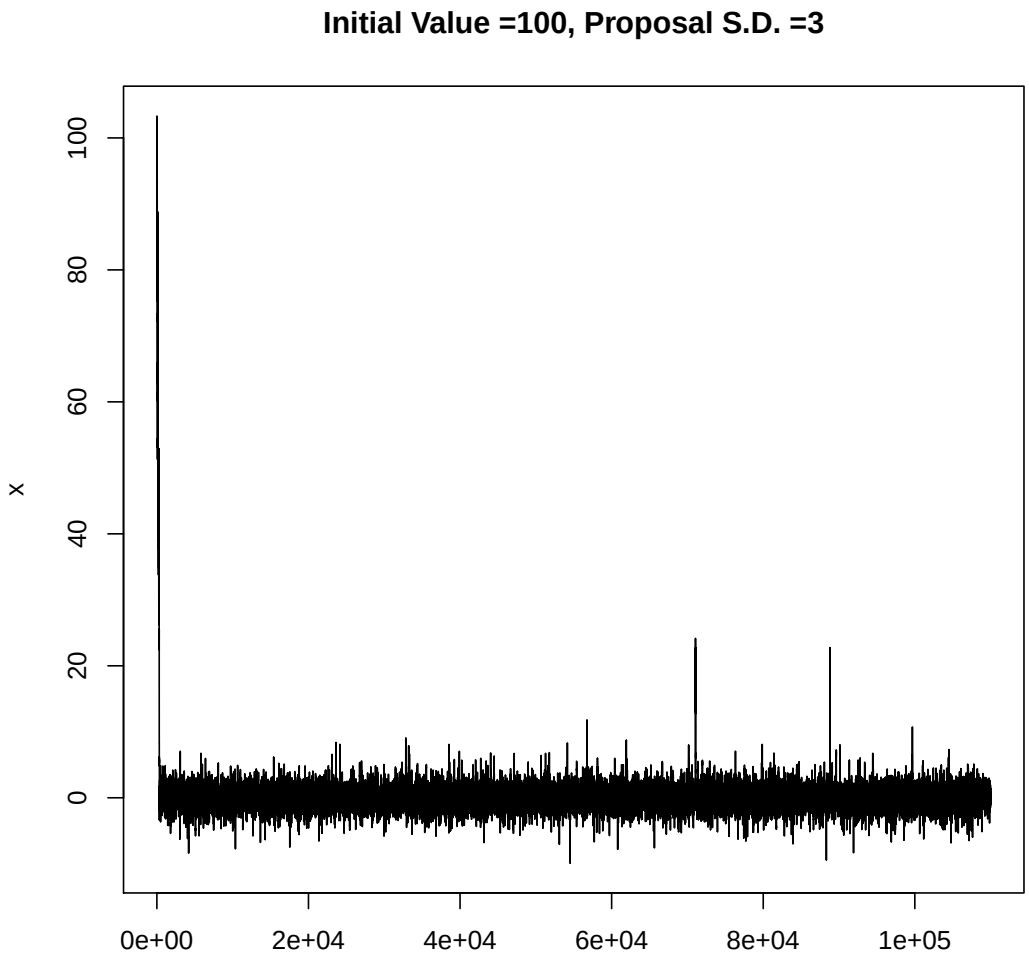
The effect of proposal variance

- Keep the starting value of $X_0 = 100$
- No change the standard deviation of the proposal to 3.
- Does the chain still converge?
- Does it converge quicker or slower?

Trace Plot


Initial Value =100, Proposal S.D. =1
x
0
20
40
60
80
100
120
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

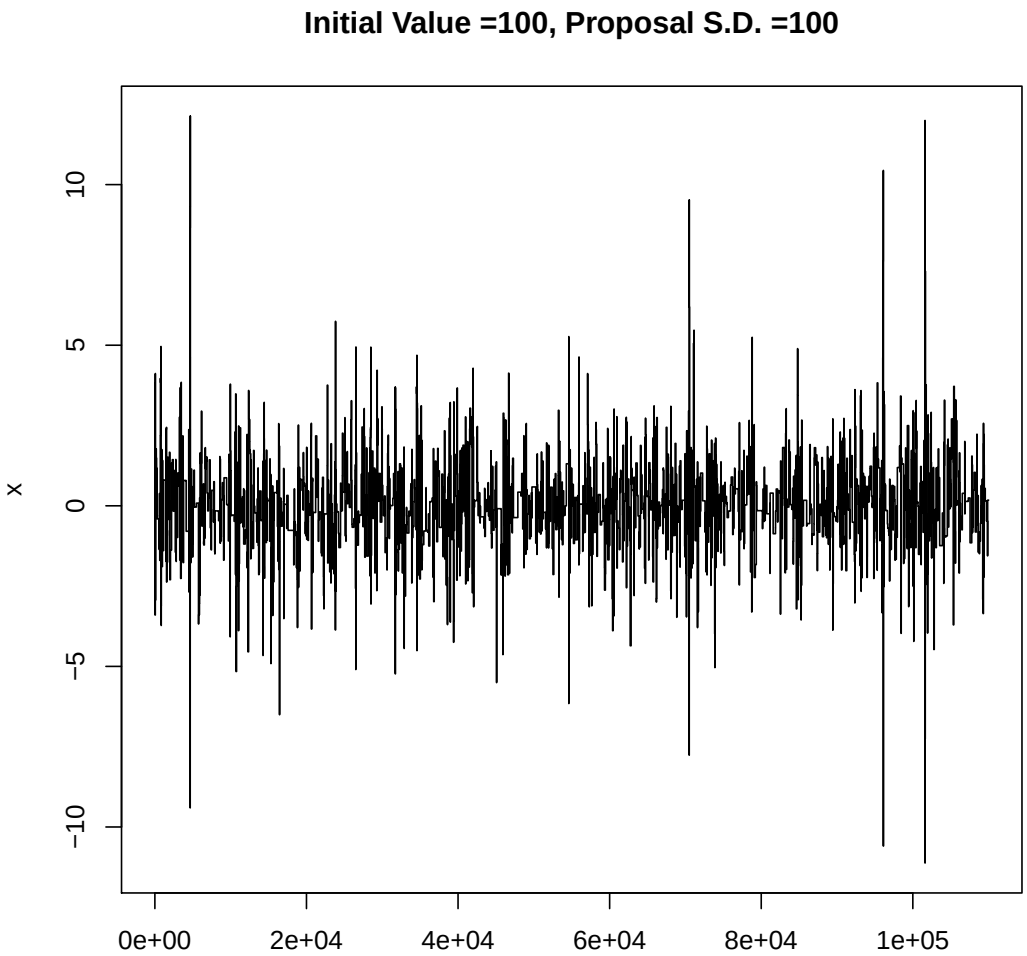
Trace Plot


Initial Value =100, Proposal S.D. =3
x
0
20
40
60
80
100
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

Huge proposal variance

- Maybe you think the best strategy is to choose a huge standard deviation.
- Try to use a proposal standard deviation of 100. Plot a trace plot of the chain.
- The plot is rejecting many iterates. This must be inefficient
- Change your code to compute the percentage of times a new iterate is accepted (excluding burn in).
- Use $\chi_0 =$ as an initial value. What is the acceptance rate when the proposal standard deviation is 1? What is the acceptance rate when the proposal standard deviation is 100?

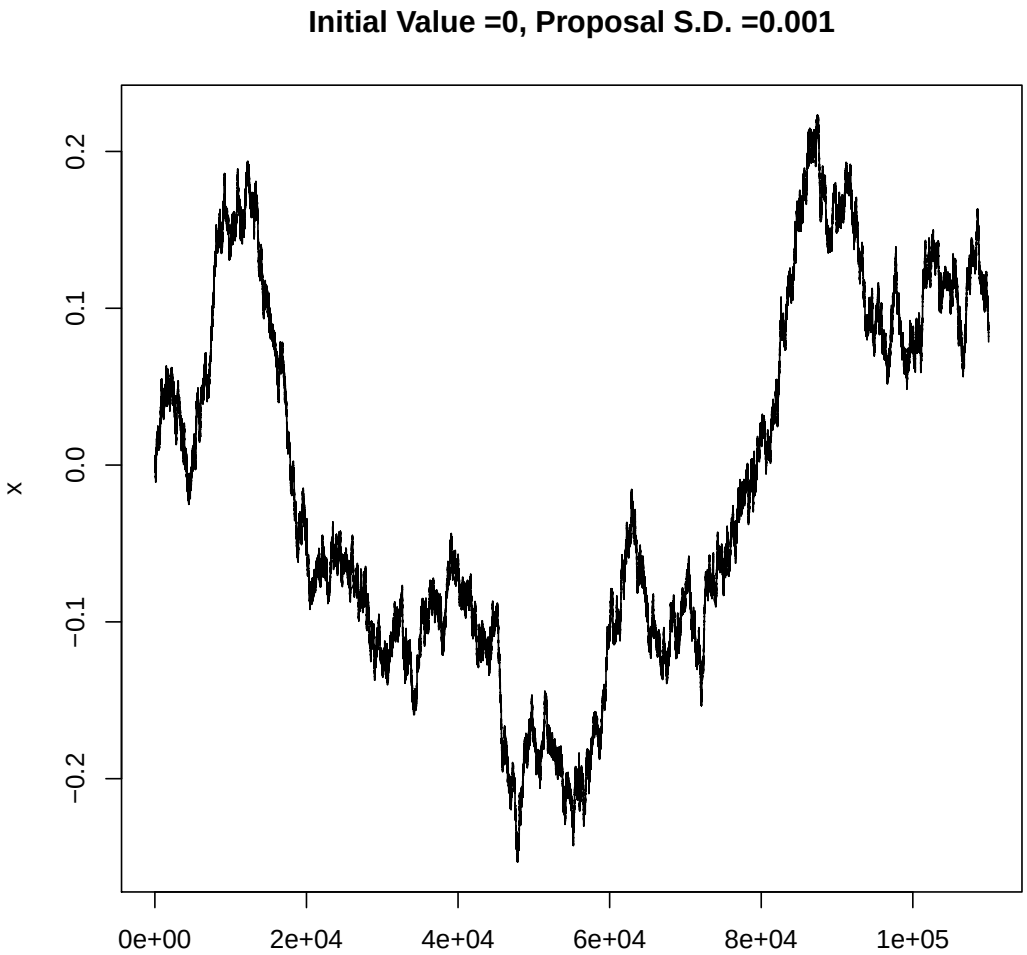
Trace Plot


Initial Value =100, Proposal S.D. =100
x
10
5
0
−5
−10
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

Acceptance Rate

- If the proposal variance is too high
  - Values will be proposed that are too far into the tails of the stationary distribution
  - The Metropolis algorithm will mostly reject these values.
  - The sample will still come from the correct target distribution but this is a very inefficient way to sample.
- What happens if a very small proposal variance is used.
- Try a proposal variance of 0.001. What is the acceptance rate?

Trace Plot


Initial Value =0, Proposal S.D. =0.001
x
0.2
0.1
0.0
−0.1
−0.2
0e+00
2e+04
4e+04
6e+04
8e+04
1e+05

Acceptance Rate

- The acceptance rate is almost 1.
- However is this a good proposal?
- The jumps made by this proposal are too small, and do not sample enough iterates from the tails of the distribution.
- If it runs long enough the Markov chain will provide a sample from the target distribution. However, it is very inefficient.

Random Walk proposal

- For a random walk proposal it is not good to have an acceptance rate that is too high or too low.
- What exactly is *too high* and *too low*?
- It depends on many things including the target and proposal.
- A rough rule is to aim for an acceptance rate between 20% and 70%
- If your acceptance rate is outside this range the proposal variance can be doubled or halved
- There are better (but more complicated) ways to do this.

Monte Carlo Error

- Now that there is a sample. $X^{[1]}, X^{[2]}, \ldots, X^{[M]} \sim p(x)$. What can it be used for?
- We can estimate the expected value $E(X)$
- This can be done by taking:

$$E(X) \approx \frac{1}{M} \sum_{i=1}^{M} X^{[i]}$$

- Note we use $\approx$ instead of $=$. There is some error since we are estimating $E(X)$ based on a sample.
- Luckily we can make this smaller by generating a bigger sample.
- We call this Monte Carlo error.

Measuring Monte Carlo Error

- One way to measure Monte Carlo Error is the variance of the sample mean.

$$\begin{aligned}\mathrm{Var}\left(\frac{1}{M}\sum_{i=1}^{M} X^{[i]}\right) &= \frac{1}{M^2}\mathrm{Var}\left(\sum_{i=1}^{M} X^{[i]}\right) \\ &= \frac{1}{M^2}\sum_{i=1}^{M}\mathrm{Var}(X^{[i]}) \\ &= \frac{\mathrm{Var}(X)}{M}\end{aligned}$$

- A sample from a Markov chain is **correlated**

Measuring Monte Carlo Error

- One way to measure Monte Carlo Error is the variance of the sample mean.

$$\begin{aligned}\mathrm{Var}\left(\frac{1}{M}\sum_{i=1}^{M} X^{[i]}\right) &= \frac{1}{M^2}\mathrm{Var}\left(\sum_{i=1}^{M} X^{[i]}\right) \\ &= \frac{1}{M^2}\sum_{i=1}^{M}\mathrm{Var}(X^{[i]}) \\ &= \frac{\mathrm{Var}(X)}{M}\end{aligned}$$

- A sample from a Markov chain is **correlated**

Measuring Monte Carlo Error

- One way to measure Monte Carlo Error is the variance of the sample mean.

$$\begin{aligned}
\mathrm{Var}\left(\frac{1}{M}\sum_{i=1}^{M} X^{[i]}\right) &= \frac{1}{M^2}\mathrm{Var}\left(\sum_{i=1}^{M} X^{[i]}\right) \\
&= \frac{1}{M^2}\sum_{i=1}^{M}\mathrm{Var}(X^{[i]}) + \frac{2}{M^2}\sum_{i=1}^{M}\sum_{j>i}\mathrm{cov}(X^{[i]}, X^{[j]}) \\
&= \frac{\mathrm{Var}(X)}{M} + \frac{2}{M^2}\sum_{i=1}^{M}\sum_{j>i}\mathrm{cov}(X^{[i]}, X^{[j]})
\end{aligned}$$

- A sample from a Markov chain is **correlated**

Monte Carlo efficiency

- It is better to have lower correlation in the Markov chain.
- The efficiency of the chain can be measured using the **effective sample size**.
- The effective sample size can be computed using the function *effectiveSize* in the R Package coda.
- Obtain an effective sample size for your sample (excluding burn in) where
  - Proposal S.D. $=1$
  - Proposal S.D. $=5$
- My answers were about 6000 and 18000 and yours should be close to that

Interpret Effective Sample Size

- What does it mean to say a Monte Carlo with a sample size of 100000 has an **effective sample size (ESS)** of just 6000?
- The sampling error of a correlated sample of 100000 is equal to the sampling error of an *independent sample* of 6000.
- Mathematically

$$\mathrm{Var}\left(\frac{1}{M}\sum_{i=1}^{M} X^{[i]}\right) = \frac{\mathrm{Var}(X)}{M} + \frac{2}{M^2}\sum_{i=1}^{M}\sum_{j>i}^{M} \mathrm{cov}(X^{[i]}, X^{[j]}) = \frac{\mathrm{Var}(X)}{M_{eff}}$$

- It is a useful diagnostic for comparing two different proposal variances. A higher ESS implies a more efficient scheme.

Non-Symmetric proposal

- In 1970, Hastings proposed an extension to the Metropolis Hastings algorithm.
- This allows for the case when

$$q(x^{[old]} \rightarrow x^{[new]}) \neq q(x^{[new]} \rightarrow x^{[old]}) \tag{28}$$

- The only thing that changes is the acceptance probability

$$\alpha = \min\left(1, \frac{p(x^{[new]})q(x^{[new]} \rightarrow x^{[old]})}{p(x^{[old]})q(x^{[old]} \rightarrow x^{[new]})}\right) \tag{29}$$

- This is called the **Metropolis-Hastings algorithm**

An interesting proposal

- Suppose we use the proposal:

$$x^{new} \sim N(0, \sqrt{(5/3)}) \tag{30}$$

- What is $q(x^{[old]} \rightarrow x^{[new]})$?
- It is $q(x^{[new]})$ where $q(.)$ is the density of a $N(0, \sqrt{(5/3)})$.
- Is this symmetric?
- No, since generally $q(x^{[new]}) \neq q(x^{[old]})$

Metropolis Hastings

- Code this where $p(.)$ is the standard t density with 5 d.f, and $q(.)$ is normal with mean 0 and standard deviation $5/3$.
- Inside a loop
  - Generate $x^{[new]} \sim N(0, \sqrt{(5/3)})$
  - Set $x^{old} = x^{[i]}$ and compute

$$\alpha = \min\left(1, \frac{p(x^{[new]})q(x^{[old]})}{p(x^{[old]})q(x^{[new]})}\right) \tag{31}$$

  - Set $x^{[i+1]}$ to $x^{[new]}$ with probability α (accept)
  - Set $x^{[i+1]}$ to $x^{[old]}$ with probability $1 - \alpha$ (reject)
- Try it

Comparison

- The Effective Sample Size of this proposal is about 43000 much higher than the best random walk proposal.
- Why does it work so well?
- The standard t distribution with 5 df has a mean of 0 and a standard deviation of $\sqrt{(5/3)}$
- So the $\mathrm{N}(0, \sqrt{(5/3)})$ is a good approximation to the standard student t with 5 df.

Laplace Approximation

Using a Taylor expansion of $\ln p(x)$ around the point a

$$\ln p(x) \approx \ln p(a) + \left.\frac{\partial \ln p(x)}{\partial x}\right|_{x=a} (x-a) + \frac{1}{2} \left.\frac{\partial^2 \ln p(x)}{\partial x^2}\right|_{x=a} (x-a)^2$$

Let a be the point that maximises $\ln p(x)$ and let

$$b = -\left(\left.\frac{\partial^2 \ln p(x)}{\partial x^2}\right|_{x=a} \right)^{-1} \tag{32}$$

The approximation is

$$\ln p(x) \approx \ln p(a) - \frac{1}{2b}(x-a)^2$$

Taking exponential of both sides

$$p(x) \approx k \times exp\left[-\frac{(x-a)^2}{2b}\right]$$

Any distribution can be approximated by a normal distribution with mean a and variance b where a and b values can be found numerically if needed.

Exercise: Generating from skew normal

- The density of the skew normal is

$$p(x) = 2\phi(x)\Phi(\delta x) \tag{33}$$

  where $\phi(x)$ is the density of the standard normal $\Phi(x)$ is the distribution of the standard normal.
- Using a combination of *optim*, *dnorm* and *pnorm* find the Laplace approximation of the skew normal when $\delta = 3$
- Use it to generate a sample from the skew normal distribution using the Metropolis Hastings algorithm.

Indirect method v Metropolis Hastings

- Some similarities are:
  - Both require a proposal
  - Both are more efficient when the proposal is a good approximation to the target.
  - Both involve some form of accepting/rejecting
- Some differences are:
  - Indirect method produces an independent sample, MH samples are correlated.
  - Indirect method requires $p(x)/q(x)$ to be finite for all x.
  - MH works better when x is a (high-dimensional vector).
- Why?

Multiple variables

- Suppose we now want to sample from a bivariate distribution $\mathrm{p}(\mathrm{x}, z)$
- The ideas involved in this section work for more than two variables.
- It is possible to do a 2-dimensional random walk proposal. However as the number of variables goes up the acceptance rate becomes lower.
- Also the Laplace approximation does not work as well in high dimensions.
- Indirect methods of simulation suffer from the same problem.
- We need a way to break the problem down.

Method of composition

- Markov chain methods, allow us to break multivariate distributions down.
- If it is easy to generate from $p(x)$ then the best way is **Method of composition**. Generate
  - $x^{[i]} \sim p(x)$
  - $z^{[i]} \sim p(z|x = x^{[i]})$
- Sometimes $p(x)$ is difficult to get

Gibbs Sampler

- If it is easy to simulate from the conditional distribution $f(x|z)$ then that can be used as a proposal
- What is the acceptance ratio?

$$\begin{aligned}\alpha &= \left(1, \frac{p(x^{new}, z)p(x^{old}|z)}{p(x^{old}, z)p(x^{new}|z)}\right) \\ &= \left(1, \frac{p(x^{new}|z)p(z)p(x^{old}|z)}{p(x^{old}|z)p(z)p(x^{new}|z)}\right) \\ &= 1\end{aligned}$$

Gibbs Sampler

- This gives the Gibbs Sampler
  - Generate $x^{[i+1]} \sim p(x^{[i+1]}|z^{[i]})$
  - Generate $z^{[i+1]} \sim p(z^{[i+1]}|x^{[i+1]})$
  - Repeat
- x and z can be swapped around.
- It works for more than two variables.
- Always make sure the conditioning variables are at the current state.

Metropolis within Gibbs

- Even if the individual conditional distributions are not easy to simulate from, Metropolis Hastings can be used *within* each Gibbs step.
- This works very well because it breaks down a multivariate problem into smaller univariate problems.
- We will practice some of these algorithms in the context of *Bayesian Inference*

Summary

- You should be familiar with a *Markov chain*
- You should understand this can have a *stationary distribution*
- You should have a basic understanding of the Metropolis Hastings and the special cases
  - Random Walk Metropolis
  - Laplace approximation
  - Gibbs Sampler