Module 6: Introduction to Metropolis

Rebecca C. Steorts

```r
knitr::opts_chunk$set(cache=TRUE)
```

Agenda

- Motivation
- Markov chain Monte Carlo (MCMC)
- Hard Discs in a Box Example
- Metropolis Algorithm
- Example Applied to Normal-Normal
- Practice Excerise (Hoff 10.3)

Intro to Markov chain Monte Carlo (MCMC)

Goal: sample from $f(x)$, or approximate $E_f[h(X)]$.

Recall that $f(x)$ is very complicated and hard to sample from.

How to deal with this?

1. What's a simple way?
2. What are two other ways?
3. What happens in high dimensions?

High dimensional spaces

- In low dimensions, IS and RS works pretty well.
- But in high dimensions, a proposal $g(x)$ that worked in 2-D, often doesn't mean that it will work in any dimension.
- Why? It's hard to capture high dimensional spaces!

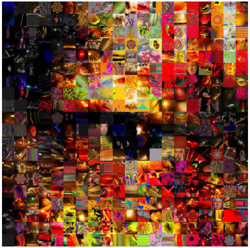

Figure 1: A high dimensional space (many images).

We turn to Markov chain Monte Carlo (MCMC).

Intuition

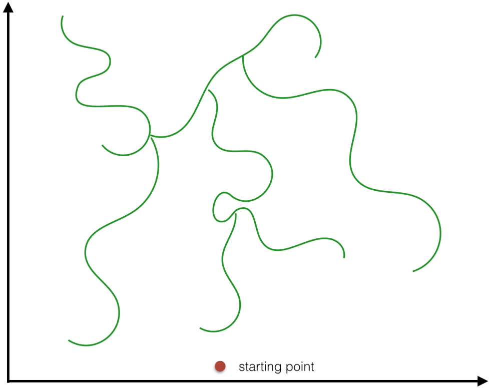

Figure 2: Example of a Markov chain and red starting point

Intuition

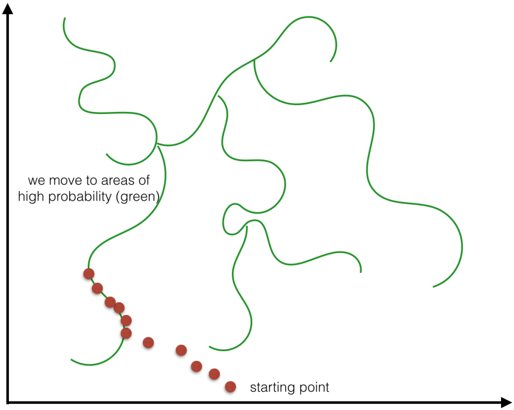

Figure 3: Example of a Markov chain and moving from the starting point to a high probability region.

What is Markov Chain Monte Carlo

- Markov Chain – where we go next only depends on our last state (the Markov property).
- Monte Carlo – just simulating data.

The Markov property

Suppose that we have just visited states $x_1, \ldots, x_{n-1}$. The Markov property says the following:

$$P(X_n = x_n \mid X_1 = x_1, \ldots, X_{n-1} = x_{n-1}) = P(X_n = x_n \mid X_{n-1} = x_{n-1}).$$

Why MCMC?

(a) the region of high probability tends to be "connected"

- That is, we can get from one point to another without going through a low-probability region, and

(b) we tend to be interested in the expectations of functions that are relatively smooth and have lots of "symmetries"

- That is, one only needs to evaluate them at a small number of representative points in order to get the general picture.

Advantages/Disadvantages of MCMC:

Advantages:

- applicable even when we can't directly draw samples
- works for complicated distributions in high-dimensional spaces, even when we don't know where the regions of high probability are
- relatively easy to implement
- fairly reliable

Disadvantages:

- slower than simple Monte Carlo or importance sampling (i.e., requires more samples for the same level of accuracy)
- can be very difficult to assess accuracy and evaluate convergence, even empirically

Hard Discs in a Box Example

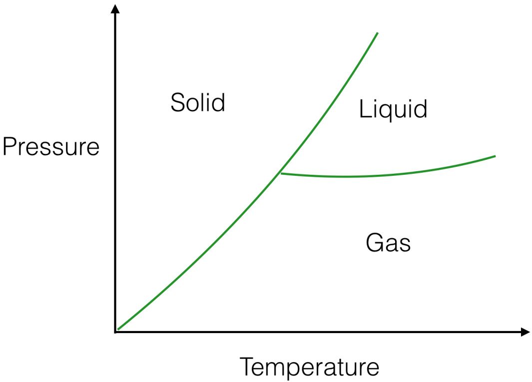

Figure 4: Example of a phase diagram in chemistry.

Many materials have phase diagrams that look like the picture above.

Hard Discs in a Box Example

To understand this phenoma, a theoretical model was proposed: Metropolis, Rosenbluth, Rosenbluth, and Teller, 1953

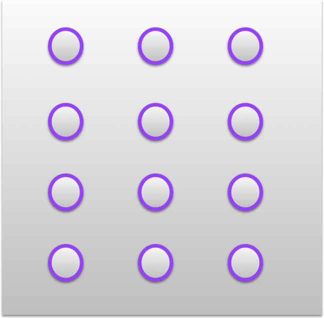

Figure 5: Example of N molecules (hard discs) bouncing around in a box.

Called hard discs because the molecules cannot overlap.

Hard Discs in a Box Example

Have an $X = (u, v)$ coordinate for each molecule.[1]

The total dimension of the space is $\mathbb{R}^{2N}$.

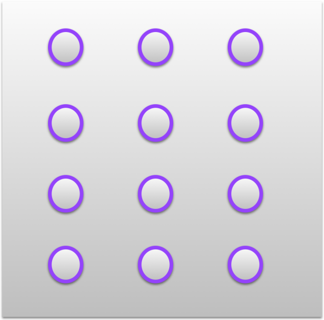

Figure 6: Example of N molecules (hard discs) bouncing around in a box.

[1] For the rest of the lecture, X will denote the two coordinate vectors of the molecule.

Hard Discs in a Box Example

$X \sim f(x)$ (Boltzman distribution).

Goal: compute $E_f[h(x)]$.

Since X is high dimensional, they proposed "clever moves" of the molecules.

Hard Discs in a Box Example

Metropolis algorithm: For iterations $i = 1, \ldots, n$, do:

1. Consider a molecule and a box around the molecule.
2. Uniformly draw a point in the box.
3. According to a “rule", you accept or reject the point.
4. If it's accepted, you move the molecule.

Uniformly = pick a point at random with equal probability to all other points in the box

Example of one iteration of algorithm

Consider a molecule and a box around the molecule.

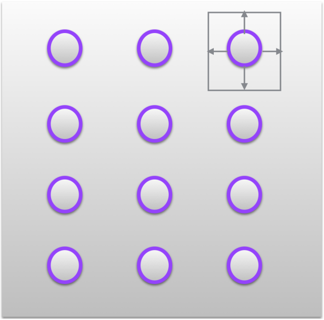

Figure 7: This illustrates step 1 of the algorithm.

Example of one iteration of algorithm

Uniformly draw a point in the box.

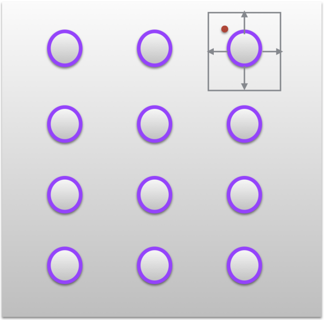

Figure 8: This illustrates step 2 of the algorithm.

Example of one iteration of algorithm

According to a “rule", you accept or reject the point.

Here, it was accepted, so we move the molecule.

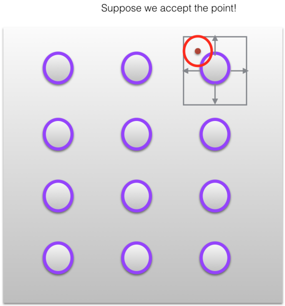

Figure 9: This illustrates step 3 and 4 of the algorithm.

Example of one iteration of algorithm

Here, we show one entire iteration of the algorithm.

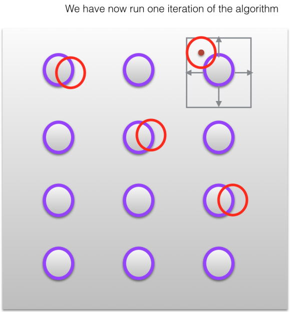

Figure 10: This illustrates one iteration of the algorithm.

After running many iterations n (not N), we have an approximation for $E_f(h(X))$, which is $\frac{1}{n}\sum_i h(X_i)$.

We will talk about the details later of why this is a "good approximation."

Some food for thought

We just covered the Metropolis algorithm (1953 paper).

- We did not cover the exact procedure for accepting or rejecting (to come).
- Are the X_i's independent?
- Our approximation holds by The Ergodic Theorem for those that want to learn more about it.
- The ergodic theorem says: "if we start at a point x_o and we keeping moving around in our high dimensional space, then we are guaranteed to eventually reach all points in that space with probability 1."

Metropolis Algorithm

Setup: Assume pmf π on $\mathcal{X}$ (countable).

Have $f : \mathcal{X} \to \mathbb{R}$.

Goal:

a) sample/approximate from π

b) approximate $E_\pi[f(x)], X \sim \pi$.

The assumption is that π and or $f(X)$ are complicated!

Why things work!

Big idea and why it works: we apply the ergodic theorem.

“If we take samples $X = (X_0, X_1, \ldots,)$ then by the ergodic theorem, they will eventually reach π, which is known as the stationary distribution (the true pmf)."

Metropolis Algorithm

The approach is to apply the ergodic theorem.

1. If we run the Markov chain long enough, then the last state is approximately from π.
2. Under some regularity conditions,

$$\frac{1}{n}\sum_{i=1}^{n} f(X_i) \xrightarrow{a.s} E_\pi[f(x)].$$

Terminology

1. Proposal matrix = stochastic matrix.

   Let
   $$Q = (Q_{ab} : a, b \in \mathcal{X}).$$

   Note: I will use $Q_{ab} = Q(a, b)$ at times.

2. Note:
   $$\pi(x) = \tilde{\pi}(x)/z, z > 0.$$

   What is known and unknown above? (Think back to rejection sampling)

Metropolis Algorithm

- Choose a symmetric proposal matrix Q. So, $Q_{ab} = Q_{ba}$.
- Initialize $x_o \in X$.
- for $i \in 0, 1, 2, \ldots, n-1$:
  - Sample proposal x from $Q(x_i, x)$ if x is discrete, otherwise, $p(x \mid x_i)$.
  - Sample r from Uniform$(0, 1)$.
  - If
    $$r < \frac{\tilde{\pi}(x)}{\tilde{\pi}(x_i)},$$
    accept and $x_{i+1} = x$.
  - Otherwise, reject and $x_{i+1} = x_i$.

Output: $x_o, x_1, \ldots, x_n$

Comment: r is the rule for uniformly drawing a point in the box (slide 18).

You do not need to know the general proof of this.

Metropolis within a Bayesian setting

Goal: We want to sample from

$$p(\theta \mid y) = \frac{f(y \mid \theta)\pi(\theta)}{m(y)}.$$

Typically, we don't know $m(y)$.

The notation is a bit more complicated, but the set up is the same.

We'll approach it a bit differently, but the idea is exactly the same.

Building a Metropolis sampler

We know $\pi(\theta)$ and $f(y \mid \theta)$, so we can can draw samples from these.

Our notation here will be that we assume parameter values $\theta_1, \theta_2, \ldots, \theta_s$ which are drawn from $\pi(\theta)$.

We assume a new parameter value comes in that is θ^*.

Building a Metropolis sampler

Similar to before we assume a symmetric proposal distribution, which we call $J(\theta^* \mid \theta^{(s)})$.

- What does symmetry mean here? $J(\theta_a \mid \theta_b) = J(\theta_b \mid \theta_a)$.
- That is, the probability of proposing $\theta^* = \theta_a$ given that $\theta^{(s)} = \theta_b$ is equal to the probability of proposing $\theta^* = \theta_b$ given that $\theta^{(s)} = \theta_a$.
- Symmetric proposals include:

$$J(\theta^* \mid \theta^{(s)}) = \text{Uniform}(\theta^{(s)} - \delta, \theta^{(s)} + \delta)$$

and

$$J(\theta^* \mid \theta^{(s)}) = \text{Normal}(\theta^{(s)}, \delta^2).$$

Metropolis algorithm

The Metropolis algorithm proceeds as follows:

1. Sample $\theta^* \sim J(\theta \mid \theta^{(s)})$.
2. Compute the acceptance rule (r):

$$r = \frac{p(\theta^*|y)}{p(\theta^{(s)}|y)} = \frac{p(y \mid \theta^*)p(\theta^*)}{p(y \mid \theta^{(s)})p(\theta^{(s)})}.$$

3. Let

$$\theta^{(s+1)} = \begin{cases} \theta^* & \text{with prob min(r,1)} \\ \theta^{(s)} & \text{otherwise.} \end{cases}$$

Remark: Step 3 can be accomplished by sampling $u \sim \text{Uniform}(0, 1)$ and setting $\theta^{(s+1)} = \theta^*$ if $u < r$ and setting $\theta^{(s+1)} = \theta^{(s)}$ otherwise.

Exercise: Convince yourselves that step 3 is the same as the remark!

A Toy Example of Metropolis

Let's test out the Metropolis algorithm for the conjugate Normal-Normal model with a known variance situation.

$$
\begin{aligned}
X_1, \ldots, X_n \mid \theta &\overset{iid}{\sim} \text{Normal}(\theta, \sigma^2) \\
\theta &\sim \text{Normal}(\mu, \tau^2).
\end{aligned}
$$

Recall that the posterior of $\theta \mid X_1, \ldots, X_n$ is $\text{Normal}(\mu_n, \tau_n^2)$, where

$$
\mu_n = \bar{x}\frac{n/\sigma^2}{n/\sigma^2 + 1/\tau^2} + \mu\frac{1/\tau^2}{n/\sigma^2 + 1/\tau^2}
$$

and

$$
\tau_n^2 = \frac{1}{n/\sigma^2 + 1/\tau^2}.
$$

Toy Example

In this example: $\sigma^2 = 1$, $\tau^2 = 10$, $\mu = 5$, $n = 5$, and

$$x = (9.37, 10.18, 9.16, 11.60, 10.33).$$

For these data, $\mu_n = 10.03$ and $\tau_n^2 = 0.20$.

Note: this is a toy example for illustration.

Toy example

We need to compute the acceptance rule r.

$$
\begin{aligned}
r &= \frac{p(\theta^*|x)}{p(\theta^{(s)}|x)} && (1)\\
&= \frac{p(x|\theta^*)p(\theta^*)}{p(x|\theta^{(s)})p(\theta^{(s)})} && (2)\\
&= \left(\frac{\prod_i \text{dnorm}(x_i, \theta^*, \sigma)}{\prod_i \text{dnorm}(x_i, \theta^{(s)}, \sigma)}\right)\left(\frac{\text{dnorm}(\theta^*, \mu, \tau)}{\text{dnorm}(\theta^{(s)}, \mu, \tau)}\right) && (3)
\end{aligned}
$$

Toy example

In many cases, computing the rule r directly can be numerically unstable, however, this can be modified by taking $\log r$.

This results in

$$\log r = \sum_i \left[\log \mathrm{dnorm}(x_i, \theta^*, \sigma) - \log \mathrm{dnorm}(x_i, \theta^{(s)}, \sigma)\right]$$
$$+ [\log \mathrm{dnorm}(\theta^*, \mu, \tau)] - \log \mathrm{dnorm}(\theta^{(s)}, \mu, \tau).$$

Then a proposal is accepted if $\log u < \log r$, where u is sampled from the Uniform(0,1).

Toy example

We generate 500 iterations of the Metropolis algorithm starting at $\theta^{(0)} = 0$ and using a normal proposal distribution, where

$$\theta^{(s+1)} \sim \text{Normal}(\theta^{(s)}, 1).$$

We will then generate 10,000 iterations since 500 will not be sufficient.

Figure~12 shows a traceplot for this run as well as a histogram for the Metropolis algorithm compared with a draw from the true normal density.

Traceplot

What is a traceplot? A traceplot is a convergence diagnostic.

It's a plot of the parameter of interest versus the number of MCMC iterations.

What does it tell us?

1. It can tell us when we have not run our MCMC long enough.
2. It can tell us when we migth be in a situation of multimodality (we will see this in future lectures).
3. If the plots looks stable, then it tells us that we do not see anything to warrant issues with a lack of convergence of the chain.

What does it not tell us?

It cannot tell us that our MCMC has converged!

Code

```r
# setting values
set.seed(1)
s2<-1
t2<-10
mu<-5;
n<-5

# defining the data
x<-c(9.37, 10.18, 9.16, 11.60, 10.33)
# mean of the normal posterior
mu.n<-( mean(x)*n/s2 + mu/t2 )/( n/s2+1/t2)
# variance of the normal posterior
t2.n<-1/(n/s2+1/t2)
```

Code (continued)

```r
##S = total num of simulations
theta<-0 ; delta<-1 ; S<-500 ; THETA<-NULL
set.seed(1)

for(s in 1:S){
## simulating our proposal
theta.star<-rnorm(1,theta,sqrt(delta))
##taking the log of the ratio r
log.r<-( sum(dnorm(x,theta.star,sqrt(s2),log=TRUE)) +
dnorm(theta.star,mu,sqrt(t2),log=TRUE) ) -
( sum(dnorm(x,theta,sqrt(s2),log=TRUE)) +
dnorm(theta,mu,sqrt(t2),log=TRUE) )
if(log(runif(1))<log.r) {
  theta<-theta.star
}

THETA<-c(THETA,theta) ##updating THETA

}
```

Code (continued)

```
length(THETA)
```

```
## [1] 500
```

```
head(THETA)
```

```
## [1] 0.000000 1.329799 2.925080 2.925080 3.412509 3.41250
```

Code (continued)

```
## pdf
##   2
```

Code (continued)

```
## pdf
##   2
```

Traceplot and Histogram

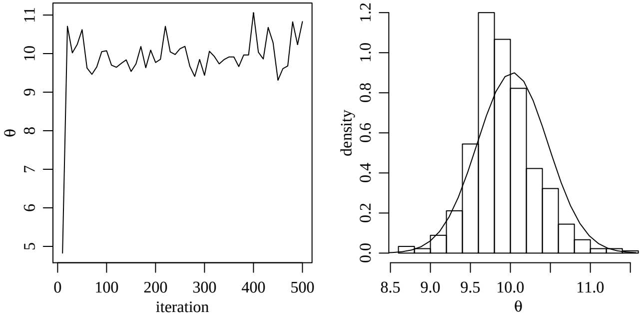

Figure 11: Left: trace plot of the Metropolis sampler. Right: Histogram versus true normal density for 500 iterations.

Traceplot and Histogram

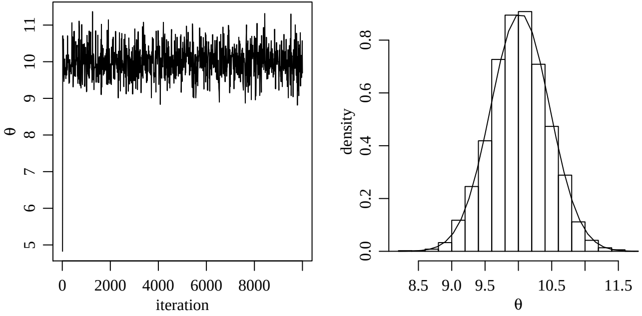

Figure 12: Left: trace plot of the Metropolis sampler. Right: Histogram versus true normal density for 10,000 iterations.

Traceplot

Given that we have looked at $n = 500$ iterations of the Metroplis sampler, does it seem that our approximation is a good one?

What about $n = 10,000$ iterations?

Questions you should be able to answer!

- What is the goal of Metropolis?
- What is known and unknown?
- What are good proposals?
- What does the ergodic theorem say in words?
- Are good proposals always easy to choose?
- When would we use Metropolis over Importance sampling and Rejection sampling?
- What is a simple diagnostic of a Markov chain?
- Are we guaranteed convergence of the Markov chain?