The MCMC principle - general idea

- A *Markov Chain Monte Carlo (MCMC) method* for simulation from a distribution $\pi$ is any method producing an ergodic Markov chain $(X_n)$ whose stationary distribution is $\pi$.
- For an arbitrary starting value $x^{(0)}$, a chain $(X_n)$ is generated using a transition kernel with stationary distribution $\pi$, which ensure the convergence in distribution of $X_n$ to a random variable from $\pi$.
- In order for the Markov chain to converge to the target (stationary or equilibrium) distribution $\pi$, it must be irreducible, aperiodic and positive recurrent.
- The main problem is how to construct a suitable chain; we will see some of the most used algorithms (namely, Metropolis-Hastings and Gibbs sampling) and also look into methods to assess whether the chain has reached its stationary distribution.
- MCMC was invented in 1953 but was not widely applied until statisticians became aware of it around 1990. Then problems that had previously been intractable suddenly became tractable.
Markov Chain Monte Carlo Methods

- Metropolis-Hastings algorithm
- Gibbs sampler
- Convergence criteria

This material is partially covered in Hoff’s book, chapters 6 & 10.
Metropolis-Hastings Algorithm

1. Start with $\theta^{(0)}$ (for example, generated from the prior)
2. At iteration $t$ generate $\theta^* \sim q(\theta^* | \theta^{(t-1)})$ proposal distribution
3. Take

$$
\theta(t) = \begin{cases} 
\theta^* & \text{with probability } \rho(\theta^{(t-1)}, \theta^*) \\
\theta^{(t-1)} & \text{with probability } 1 - \rho(\theta^{(t-1)}, \theta^*) 
\end{cases}
$$

where

$$
\rho(\theta^{(t-1)}, \theta^*) = \min \left\{ \frac{p(\theta^* | x)}{p(\theta^{(t-1)} | x)} \frac{q(\theta^{(t-1)} | \theta^*)}{q(\theta^* | \theta^{(t-1)})}, 1 \right\}
$$

[The update at each iteration is accomplished by (a) sampling $u \sim Uniform(0; 1)$ and (b) setting $\theta(t) = \theta^*$ if $u < \rho(\theta^{(t-1)}, \theta^*)$ or $\theta(t) = \theta^{(t-1)}$ otherwise]
Remarks

- The ratio $\rho(\theta^{(t-1)}, \theta^*)$ is called the MH acceptance ratio.
- A chain constructed via the MH algorithm is Markov, since $\theta^{(t)}$ depends only on $\theta^{(t-1)}$.
- Limit Behavior: With minimal regularity conditions on both $\pi$ and the proposal $q$ it can be proved that $p(\theta|x)$ is the limiting (stationary) distribution of the chain $\theta^{(t)}$ produced by the MH algorithm.
- If $q(z|y) = q(y|z)$, i.e., the proposal is symmetric, the method is generally known as Metropolis algorithm. In such a case, the acceptance is driven only by the ratio of the two density functions

$$\frac{p(\theta^*|x)}{p(\theta^{(t-1)}|x)}$$
Other remarks

- The MH algorithm depends only on the ratios

\[ \frac{p(\theta^* | x)}{p(\theta^{t-1} | x)} \quad \text{and} \quad \frac{q(\theta^{t-1} | \theta^*)}{q(\theta^* | \theta^{t-1})}. \]

Hence, it’s independent of normalizing constants. (However, those normalizing constants must be, in turn, also independent of the conditioning variables).

- In addition, a good proposal distribution would satisfy the following:
  - it should be easy to sample from it
  - it should be easy to compute the acceptance ratio
  - each proposal should go a reasonable distance in the parameter space, otherwise the random walk moves too slowly
  - the proposals are not rejected too frequently
Integral Approximation

Once the chain has reached convergence to the stationary distribution, $\pi$, we can use the realizations of the chain to approximate quantities of interest such as

$$E \left[ h(\theta|x) \right] \approx \frac{1}{n} \sum_{t=1}^{T} h(\theta^{(t)}) \quad (\text{Ergodic Theorem})$$

Of course, the approximation depends on whether or not the chain has reached convergence (we’ll discuss this point later, for now we just assume that’s the case).

Although the MH algorithm is valid for any $q$ satisfying the mild conditions we have seen before, the choice of the proposal greatly affect the efficiency of the algorithm. Hence, now we proceed discussing some default choices.
Independent Metropolis Hastings

The IMH considers a proposal \( q \) which is independent of the current state of the chain. Then, the MH proceeds as follows:

1. Given \( \theta^{(t-1)} \),
2. Generate \( \theta^* \sim q(\theta^*) \)
3. Take

\[
\theta^{(t)} = \begin{cases} 
\theta^* \text{ with probability } \min\left\{ \frac{p(\theta^*|x)}{p(\theta^{(t-1)}|x)} \frac{q(\theta^{(t-1)})}{q(\theta^*)}, 1 \right\} \\
\theta^{(t-1)} \text{ otherwise}
\end{cases}
\]

- Although the \( \theta \)'s are generated independently, the resulting sample is not i.i.d. Consider, for example, the probability of acceptance, which depends on \( \theta^{(t-1)} \).
Example – Mixture of two normals

Suppose we have observed data $y_1, \ldots, y_n$ from the following mixture distribution

$$\delta N(7, 0.5^2) + (1 - \delta) N(10, 0.5^2)$$

We may want to obtain inference on the $\delta$. We assume a $U(0, 1)$ prior on $\delta$. We implement a MH scheme, with an independent proposal. In particular, we compare the results for 2 different choices:

$$\delta \sim Unif(0, 1) \quad \text{and} \quad \delta \sim Beta(2, 10).$$

(see code in R – pay attention on how the MH is implemented)
Model formulation: \( Y|\delta \sim \delta \mathcal{N}(7, 0.5^2) + (1 - \delta) \mathcal{N}(10, 0.5^2) \)

Likelihood: \( f(y|\delta) = \prod_{i=1}^{n} \left[ \delta \mathcal{N}(y_i; 7, 0.5^2) + (1 - \delta) \mathcal{N}(y_i; 10, 0.5^2) \right] \)

Prior: \( \pi(\delta) = \text{Beta}(1, 1) \)

Posterior: \( p(\delta|y) \propto f(y|\delta)\pi(\delta) = f(y|\delta) \)

Metropolis-Hasting algorithm:

- Proposal: \( q(\delta) = \text{Beta}(1, 1) \) or \( q(\delta) = \text{Beta}(2, 10) \)
- Hasting ratio: \( r = \frac{p(\delta^*|y)q(\delta^{\text{old}})q(\delta^*)}{p(\delta^{\text{old}}|y)q(\delta^*)} = \frac{f(y|\delta^*)q(\delta^{\text{old}})}{f(y|\delta^{\text{old}})q(\delta^*)} \).
Data inspection

```r
y = scan("mixture.dat", skip=1)
# Plot of the data
hist(y, xlab="y", ylab="Density", main="Mixture data",
    freq=FALSE, breaks=20, col="gray")
lines(density(y, bw=0.5), lwd=2, col="red")
```
Define density of Normal mixture

# density function for the MH ratio
dmixtnorm = function(x, a, m1, m2, s1, s2) {
  # x=data; a=alpha;
  # m1=mean of first component; m2=mean of 2nd comp
  # s1=SD of first component; s2=SD of second comp
  a*dnorm(x, m1, s1) + (1-a)*dnorm(x, m2, s2)
}

m1 = 7; s1 = 0.5; m2 = 10; s2 = 0.5
# case 1: Uniform\([0,1]\)=Beta\((1,1)\) proposal

set.seed(100)

T = 100000        # number of MCMC iterations
nburn = 50000     # burn-in period
alpha = rep(NA, T) # storage for sampled values

alpha[1] = 0.5    # initial value
accept = 0        # acceptance count
## Implementation of Metropolis-Hastings algorithm

```r
for(t in 2:T) {
    alpha.prop = runif(1)  ## proposed value

    ## MH accept ratio -- on log-scale for num stability
    logR = sum(log(dmixtnorm(y, alpha.prop, m1, m2, s1, s2))) -
    sum(log(dmixtnorm(y, alpha[t-1], m1, m2, s1, s2)))
    R = exp(logR)

    u <- runif(1)  ## unif var to determine acceptance
    if(u < R) {  ## accept the new value
        alpha[t] = alpha.prop
        accept = accept + 1  ## update count
    } else {
        alpha[t] = alpha[t-1]
    }
}

accept.rate = accept/T
```
Figure: Left: Uniform(0,1) proposal distribution; Right: Beta(2,10)
Random Walk Metropolis Hastings (RWMH)

- Intuitively, the chain might be more efficient if we consider proposals that take into account the value previously simulated to generate the following value; that is, if we consider a local exploration of the neighborhood of the current value of the Markov Chain.

- A first choice is to simulate $\theta^*$ as

$$\theta^* = \theta^{(t-1)} + \varepsilon,$$

where $\varepsilon \sim g$ independently of $\theta^{(t-1)}$. Hence, now

$$q(\theta^* | \theta^{(t-1)}) = g(\theta^* - \theta^{(t-1)})$$

**Choices of g**: Most common choices are:
- Uniform distributions on spheres centered at the origin
- Scaled normal distribution
- Scaled Student’s t distribution
Summary so far

- The idea of Markov chain simulation is to simulate a random walk in the space of $\theta$, which converges to a **stationary distribution** (also called **equilibrium distribution** or **target distribution**).
- In Markov chain simulations, the samples are drawn sequentially, with the distribution of the sampled draws depending on the last value drawn.
- The key is that the approximate distributions are improved at each step in the simulation, in the sense of converging to the target distribution.
The standard practice in Markov Chain Monte Carlo approximation is to:

1. run the algorithm until some iteration $B$ for which it looks like the Markov chain has achieved stationarity;
2. run the algorithm $S$ more times, generating $\{\theta^{(B+1)}, \ldots, \theta^{(B+S)}\}$;
3. discard $\{\theta^{(1)}, \ldots, \theta^{(B)}\}$ and use the empirical distribution of $\{\theta^{(B+1)}, \ldots, \theta^{(B+S)}\}$ to approximate $p(\theta|y)$.

The iterations $\{\theta^{(1)}, \ldots, \theta^{(B)}\}$ are called the burn-in period, in which the Markov chain moves from its initial value to a region of the parameter space that has high posterior probability.