Metropolis Algorithm

Gustavo Ramirez

January 21, 2017

A clear explanation to the Metropolis algorithm is developed here.

The Metropolis algorithm is a way of sampling a p.d.f. $f(y)$ through the use of a Markov chain. For the evolution of the chain, an auxiliar function is used, called the *proposal function* $q(y)$. This proposal function, as is suggested by its name, proposes the next state of the chain for each iteration of the algorithm; therefore $y_i = q(y_{i-1})$. The whole set of values generated using $q(y)$ are distributed according to a p.d.f., let us say $f_q(y)$.
There is also another auxiliar p.d.f., the *prior distribution* $\pi(y)$, which is selected generally such that $y_0 = \pi(0)$. Just as for $q(y)$, the whole set of possible values generated with $\pi(y)$ are distributed according to a p.d.f., let us call it $f_\pi(y)$.
It is important to note that there may be different possible functions (or in this case, implementations) $q(y)$ generating the same p.d.f. $f_q(y)$, and the same applies for $\pi(y)$ and $f_\pi(y)$; but what is important is the form of the p.d.f.s $f_q(y)$ and $f_\pi(y)$.
Finally, the fundamental function to be sampled is called the *target function* $f(y)$.
The algorithm is generally described through a series of steps. More descriptive features of both $f_q(y)$ and $f_\pi(y)$ will be given after the steps of the algorithm are listed. The term *acceptance probability*, refering to how good is to take each new step in the chain proposed by the proposal function, will be explained as well after those steps.

1. generate y_0 using $\pi(y)$, set it as "current state" y and add it to $\{y_i\}$
2. keep in the following loop until necessary:
   a) generate a proposal state y^* using $q(y)$, and set it as "current state" y
   b) calculate the acceptance probability:
   $$\alpha = \min\left(1, \frac{f(y^*)}{f(y)}\right) \tag{0.1}$$
   c) generate a random number u, uniform over the interval $[0,1]$
   d) if $u \leq \alpha$, set y^* as "current state" y and add it to $\{y_i\}$; if not, go back to step 2-(a)

In step 2, "until necessary" means until a desired accuracy (e.g. variance) is reached, until a number of steps is reached, etc.
Usually, when implementing the Metropolis algorithm, the form of $f(y)$ is fixed/given, and $f_q(y)$ and $f_\pi(y)$ are chosen such that the the output of the sampling, i.e. $\{y_i\}$, meets certain characteristics. One of the nice characteristics of Markov chains, is that after a large number of steps, the system converges towards a fixed point, i.e. towards a fixed p.d.f.; in the case

under consideration, 1-dim integrals and 1-dim p.d.f.s, this means that after a large number of steps, $\{y_i\}$ is distributed according to $f(y)$. Here is where the term *burn-in* comes in; burn-in is that initial subset of $\{y_i\}$ which doesn't resemble that much to the target distribution $f(y)$.

If the burn-in is represented as $\{y_i^{bi}\}$, then the set used in the Monte Carlo integration is $\{y_i'\} = \{y_i\} - \{y_i^{bi}\}$; i.e. the summation in equation 0.1 is performed over the set $\{y_i'\}$. How is the burn-in subset obtained? Usually $\{y_i^{bi}\} = \{y_0, y_1, ..., y_{49}\}$.