Metropolis Algorithm (Continuous State Space)

Instructor: Dr. Wei Li Scribe: Kyle Beiter

Nov 11th, 2021

Metropolis Algorithm in Continuous Space

We consider a continuous state space, from which we would like to sample a target density $f(\cdot)$ or a kernel $\tilde{f}$. Our goal is to find a transition function $p(\cdot|\cdot)$ to satisfy the 'detailed balance conditions', i.e: for a proposed x^*,

$$p(x^*|x)f(x) = p(x|x^*)f(x^*).$$

If we can find such a p, then the long run distribution of the ergodic markov chain with transition distribution $p(\cdot|\cdot)$ is $f(\cdot)$.

Method

Choose an arbitrary transition density $q(\cdot|\cdot)$, without requring that the detailed balance conditions hold. Then use an acceptance probability on proposed samples as

$$\alpha(x^*|x) = \min\left(1, \frac{q(x|x^*)f(x^*)}{q(x^*|x)f(x)}\right).$$

The goal of this is to regulate the flow of probabilities between the current state and the proposed state. So given that our current state is x then the probability of a particular state being proposed and accepted is $\alpha(x^*|x)q(x^*|x)$. Similarly, we can calculate the probability that any proposal is accepted as $a(x) := \int \alpha(x^*|x)p(x^*|x)dx^*$. A move is not accepted with probability $r(x) := 1 - a(x)$, in which case we set $x^* = x$. The (effective) transition density is then

$$p(x^*|x) = \alpha(x^*|x)q(x^*|x) + r(x)\delta_x(x^*)$$

where $\delta_x(x^*) = 1$ if $x^* = x$ and 0 otherwise. Then since we are in continuous state space, this can be generalized to

$$\int_A p(x^*|x)dx^* = p(A|x) = \int_A \alpha(x^*|x)q(x^*|x)dx^* + r(x)\delta_A(x)$$

where $\delta_A(x) = 1$ if $x \in A$ and 0 otherwise.

We can show that the (effective) transition density indeed satisfy the detailed balance condition.

Proof:

Algorithm

1. Given a target $f(\cdot)$ and a chosen $q(\cdot|\cdot)$, generate x^* from $q(\cdot|x^{(s)})$
2. Compute $\alpha(x^*|x^{(s)}) = \min(1, \frac{q(x^{(s)}|x^*)f(x^*)}{q(x^*|x^{(s)})f(x^{(s)})})$
3. Generate $u \sim \text{unif}(0,1)$
   i. if $u \leq \alpha(x^*|x^{(s)})$, set $x^{(s+1)} = x^*$
   ii. if $u > \alpha(x^*|x^{(s)})$, set $x^{(s+1)} = x^{(s)}$

Remark

A necessary condition for the algorithm to work is $\text{supp}(f) \subseteq \text{supp}(q(\cdot|x))$ for all $x \in \text{supp}(f)$. A minimal necessary condition would be $\text{supp}(f) \subseteq \cup_{x\in\text{supp}(f)}[\text{supp}(q(\cdot|x))]$

Choice of Transition Density

1. Classical Choice (Random Walk): Choose $q(x^*|x) = \tilde{q}(x^* - x)$ where $\tilde{q}$ is symmetric about 0. For example $x^* = x^{(s)} + \epsilon^{(s)}$, where $\epsilon^{(s)} \sim \text{unif}(-1,1)$ or $\epsilon^{(s)} \sim N(0,\tau^2)$. Then we would have $x^* \sim \text{unif}(x^{(s)} - 1, x^{(s)} + 1)$, or $x^* \sim N(x^{(s)}, \tau^2)$ respectively. Then due to the symmetry about 0, $q(x^*|x) = q(x|x^*)$, so the acceptance probability simplifies to $\alpha(x^*|x) = \min(1, \frac{f(x^*)}{f(x)})$.
2. Independent Metropolis Hastings: Choose $x^* \sim q(\cdot) = q(\cdot|x)$ independent of the current value $x^{(s)}$. For example $q \sim N(0,1)$ throughout.

Practicalities

1. **Burn-in**
   i. Run algorithm until some iteration B for which Markov Chain appears to reach stationarity.
   ii. Run algorithm B' more times and keep $\{x^{B+1}, \ldots, x^{B+B'}\}$
   iii. Use empirical distribution of $\{x^{B+1}, \ldots, x^{B+B'}\}$ to approximate the density f.
2. **Thinning**: After burning to discard first B observations, then thin the chain by including every kth draw.
3. **Trace Plot**: The plot of the values generated from the metropolis algorithm against the iteration number. Want to check:
   a. The chain doesn't get stuck in a certain area of space.
   b. The chain can move relatively fast. Between 30%-60% acceptance rate is ideal.
   c. The chain moves at an appropriate speed. Proposals are not too bold (standard deviation of p is too high), however they are large enough to be consequential (standard deviation of p is not too small)
4. **"ACF" Autocorrelation Function**: Calculate the "lag j sample covariance" as

$$\hat{\gamma}_j = \frac{1}{T}\sum_{t=j+1}^{T}(x^{(t)} - \bar{x})(x^{(t-j)} - \bar{x})$$

and the "lag j sample autocorrelation" as $\hat{\rho}_j = \frac{\hat{\gamma}_j}{\hat{\gamma}_0}$. Then plotting autocorrelation $\hat{\rho}_j$ v.s. the lag index (j), relatively high values would indicate a high degree of correlation between draws and therefore, slow mixing.