Advanced Sampling Algorithms

1 Markov Chain Monte Carlo (MCMC) (vs) Exact Sampling

Markov Chain approach - On a continuous state space, Markov Chain theory starts with a transition kernel $\mathrm{P}(x, A)$ where $x \in R^d$ and $A \in B$ where B is the Borel sigma field on R^d i.e., formed by operations of countable union, countable intersection, and relative complement. Since, the transitions of a particular x is a distribution function, $\mathrm{P}(x, R^d) = 1$. An x also permits a transition to itself. The major concern here is to estimate the existence and convergence of the iterations of the transition kernel to an invariant distribution π.

Markov Chain Monte Carlo (MCMC) - MCMC methods however turn the theory around. Given the invariant density, π, without the transition kernel we need methods to find and utilize a transition kernel P whose n^{th} $(n \to \infty)$ iteration leads to the invariant density. This is done in order to generate samples from the invariant distribution. The process is generated at an arbitrary x and iterated a large number of times. After these large number of iterations the distribution of the observations generated from the simulation is approximately equal to the target distribution.

Metropolis-Hastings [1] - The problem now is to find an appropriate transition kernel that will converge to the desired distribution π eventually. Metropolis-Hastings devised a method to generate the transition matrix by introducing the concept of Markov Chains into acceptance-rejection (A-R) sampling. A-R sampling generates samples from a target density $\pi(x) = f(x)/K$. K is the normalizing constant for the unnormalized density $f(x)$. If $h(x)$ is some known density such that $f(x) \leq ch(x)$, it can generate candidates Z for a random sample from $\pi(x)$. Along with a random number generator from $U[0, 1]$, which generates u, Z is accepted as a sample from $\pi(x)$ if $u \leq \frac{f(Z)}{cH(Z)}$ and rejected otherwise. The selection of c is crucial to reduce the number of rejections. Markov Chain is introduced here to check the reversibility condition that is mandatory for attaining a stationary distribution.

Exact Sampling - The problem with all MCMC methods is this: a given algorithm can be guaranteed to produce samples from the target density π asymptotically, 'once the chain has converged to the equilibrium distribution'. But if one runs the Markov chain for too short a time T, then the samples will come from some other distribution P^T. For how long must the Markov

chain be run before it has 'converged'? Propp and Wilson (1996) allows one, for certain chains, to answer this very question (how long to run the Markov Chain?); furthermore Propp and Wilson show how to obtain 'exact' samples from the target distribution π.

Propp-Wilson [2] - 'Coupling from the past' as the algorithm is called, works on three basic concepts:

- Coalescence of Coupled Markov Chains: If several Markov chains starting from different initial conditions share a single random-number generator, then their trajectories in state space may coalesce; and having, coalesced, will not separate again.
- Coupling from the Past: We can obtain exact samples by sampling from a time T_0 in the past, up to the present. If coalescence has occurred, the present sample is an unbiased sample from the invariant distribution; if not, we restart the simulation from a time T_0 further into the past, reusing the same random numbers.
- Monotonicity: For some Markov chains, it may be possible to detect coalescence of all trajectories without simulating all those trajectories. Ordering the states and finding two boundary states, the trajectories of the rest of the states are confined within the two boundary state trajectories which never cross.

2 Markov Chain Monte Carlo (MCMC)

In MCMC algorithms, we want to obtain samples from a target distribution $\pi(.)$. For this purpose, we need to find and use a transition kernel P whose nth iterate converges to $\pi(.)$ for large n.

Moreover, P has to satisfy the reversibility condition w.r.t $\pi(.)$ in order to prove that $\pi(.)$ is the stationary distribution of P.

Before to explain how Metropolis Hasting Algorithm works, we should recall some definitions and theorems.

Definition 1: Let $(X_0, X_1, ...)$ be a Markov chain with state space $S = s_1, ..., s_k$ and transition matrix P. A probability distribution π on S is said to **reversible** for the chain (or for the transition matrix P) if for all $x, y \in \{1, ..k\}$ we have:

$$\pi(x)p(x, y) > \pi(y)p(y, x) \tag{1}$$

Theorem 1: Let $(X_0, X_1, ...)$ be a Markov chain with state space $S = s_1, ..., s_k$ and transition matrix P. If π is a reversible distribution for the chain, then it is also a stationary distribution for the chain.

2.1 Metropolis Hasting Algorithm (MH)

Input: A target distribution $\pi(.)$

Output: Set of samples

Algorithm 1

```
1: procedure MH
2:     Initialize X_0 and t = 0
3:     loop
4:         Sample Y from the proposal distribution q(.|X_t)
5:         Sample a Uniform(0,1) random variable U
6:         if U ≤ r(X_t, Y) then
7:             X_{t+1} = Y
8:         else
9:             set X_{t+1} = X_t
10:        end if
11:        Increment t
12:    end loop
13: end procedure
```

Where the acceptance probability is $r(X, Y)$:

$$r(X, Y) = min\left(1, \frac{\pi(Y)q(X|Y)}{\pi(X)q(Y|X)}\right) \tag{2}$$

The explanation of Metropolis-Hastings is as follows:

- Metropolis Hasting Algorithm use a Proposal Distribution ($q(x, y)$) in order to generate candidates.
- If $q(x, y)$ satisfies the reversibility condition (Definition 1) by itself, then we can be sure that $\pi(.)$ is the stationary distribution (Theorem 1).
- Otherwise, we might have the case that for some x, y

$$\pi(x)q(x, y) > \pi(y)q(y, x) \tag{3}$$

  In words, it means that the movement from x to y is too often and from y to x is too rarely.

- Due on this case, Metropolis-Hastings tries to correct this inequality reducing the number of moves from x to y. And it introduces a probability $r(x, y) < 1$ when move is made. If the move is not made, the process returns x as value from the target distribution. It worthwhile to recall that $r(x, y)$ is the acceptance probability or the probability of move.
- Thus, transition from x to y are:

$$P_{MH}(x, y) \equiv q(x, y)r(x, y) \qquad \text{where } x \neq y \tag{4}$$

- The value of proposal distribution $P_{MH}(x, y)$ is essentially the value for the transition matrix entry $P_{x,y}$ of the transition matrix P for the Markov Chain. Here x and y are the states in the sate space S.

How is $r(x, y)$ determined ?

- From (3), we know that the movement from y to x is not often. So $r(y, x)$ should be as large as possible, where its upper limit is 1 (because $r(x, y)$ is probability).
- Because it is important that P_{MH} satisfies the reversibility condition then we have that :

$$\begin{aligned} \pi(x)q(x, y)r(x, y) &= \pi(y)q(y, x)r(y, x) \\ \pi(x)q(x, y)r(x, y) &= \pi(y)q(y, x) \\ r(x, y) &= \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)} \end{aligned} \tag{5}$$

- If the inequality (3) is reversed then $r(x, y) = 1$ and

$$r(y, x) = \frac{\pi(x)q(x, y)}{\pi(y)q(y, x)} \tag{6}$$

- Then (5) and (6) ensure that the two sides of (3) are in balanced. In other words, $P_{MH}(x, y)$ satisfies the reversibility . Thus its stationary distribution is $\pi(x)$.

How do we choose the proposal Distribution $q(x, y)$:

For practical simulation studies, we can choose the $q(x, y)$ from different options:

1. We can consider $q(x, y)$ as only symmetric proposals, having the form $q(y|x) = q(x|y)$ for all x and y. This option was given by Metropolis Algorithm (1953).

2. Candidates might be drawn independently of the current location x. Also these candidates are drawn uniformly from the set of states.

3. Other option is to exploit the known form of π to specify a candidate-generating distribution.

2.2 Metropolis-Hastings and Ising Model

The Ising model can be described as in the framework of a d-dimensional lattice. Let $G = L^d$ is d dimensional lattice. We can attach the value of a random variable X_i to the states of the Ising Model. A state in Ising model space is combination of $\{-1, +1\}$ at each site. So in d dimension one state consists of a vector of the form $-1, +1^d$. Now collecting samples from the state space when an stationary distribution is reached is quite hard. We start with a sample and generate the next sample from the current sample. The transitions between samples are constructed so that in steady state the samples we obtain from a stationary distribution.

For Ising model collecting samples from exact distribution when the transition matrix is unknown is an appropriate example of metropolis-hasting algorithm.

The metropolis-hasting algorithm requires two things as follows,

1. A target probabbility distribution π_X.

2. A proposal distribution $q(x'|x)$ that tells how to generate x' given a current sample x.

The method has been outlined in Algorithm 1.

As discussed in the previous section choosing correct proposal distribution is an important task and many subsequent calculations and the efficiency of the algorithm will depend on q. For Ising model the situation is same except few more intuitive notions are involved to form a better q. We can denote the the spins as $\{t_1, \ldots, t_i, \ldots, t_n\}$ where $t_i \in \{-1, +1\}$. We can further define a function $W_i : \{t_1, \ldots, t_i, \ldots, t_n\} \to \{t_1, \ldots, -t_i, \ldots, t_n\}$ which in other words can be termed as one-spin-flip. This configuration can be denoted as s. Let the probability of state s is $P(s)$ and probability that ith spin takes the value t_i is $P_{eq}(t_i)$.

For Ising model

$$r_i(t_i) = min\{1, exp(-2\beta(t_i t_{i+1} + t_{i-1} t_i))\}$$

Algorithm 2

```
1: procedure IsingMH(r_i,MAXCOUNT)
2:     Specify an initial configuration
3:     Choose a lattice site i
4:     Compute r_i
5:     loop
6:         Generate a random number U ∈ [0, 1]
7:         if r_i(t_i) > U then
8:             t_i → −t_i
9:         else
10:            Continue until MAXCOUNT is reached
11:        end if
12:    end loop
13: end procedure
```

If t_i is the ith lattice point then two neighbouring points are t_{i-1} and t_{i+1}.

Clearly proposal distribution is uniform here and the transition is in form of the inequality in the loop. r_i is known as metropolis function.

3 Propp-Wilson Algorithm

Let's recall the goal before to explain Propp-Wilson Algorithm.

Goal: We want to sample from distribution π.

As we explain before MCMC is an option for reaching the goal. However MCMC raises the following questions:

- How long should the chain be run to attain equilibration?
- How can we ensure that the samples are not approximately but exactly distributed w.r.t to π?

Propp-Wilson Algorithm tries to answer the above questions where:

- Propp-Wilson is a Las Vegas variant of MCMC.
- s is distributed exactly w.r.t π (perfect simulation).
- Detection of equilibration should be automatic.

Explanation Let S be the state space.

Let k be the total number of states.

Let $(N_1, N_2, \ldots, N_m)$ be an increasing sequence of positive integers.The negative numbers $(-N_1, -N_2, \ldots)$will be used as starting times for the Markov chain.

Let $(U_0, U_1, \ldots)$ be a sequence of i.i.d. random numbers $\sim U[0, 1]$.

1. Set $m = 1$
2. For each $s \in S$, simulate the Markov chain starting at time $-N_m$ in state s and running up to time 0 using the trail move ϕ with the sequence $U_{-N_m+1}, U_{-N_m+2}, \ldots, U_{-1}, U_0$ (A subtle observation here is that we are simulating a markov chain using the same random walk we have in Metrapolis Hastings Algorithm.)
3. If all k chains end up in the same state s at time 0, output s and stop.
4. Set $m = m + 1$ and go to step 2

4 Coalescence

Following theorem is a crucial and necessary step while proving correctness of Propp-Wilson Algorithm and argue that it does what it claims in practice [3].

4.1 Theorem:

Let P be the transition matrix of an irreducible and aperiodic Markov chain with state space $S = \{s_1, s_2, \ldots, s_k\}$ and stationary distribution $\pi = (\pi_1, \pi_2, \ldots, \pi_k)$.

Let ϕ be a valid update function(trail move) for P, and consider the Propp-Wilson algorithm with $(N_1, N_2, \ldots) = (1, 2, 4, 8, \ldots)$.

Suppose that the algorithm terminates with probability 1, and write Y for its output. Then for any $i \in \{1, 2, 3, \ldots, k\}$, we have

$$P(Y = s_i) = \pi_i \tag{7}$$

Proof:

Fix an arbitrary state $s_i \in S$. In order to prove ,it is enough to show that for any $\epsilon > 0$, we have

$$|P(Y = s_i) - \pi_i| \leq \epsilon \tag{8}$$

By the assumption that the algorithm terminates with probability 1 we can make sure that

$$P(\text{the algorithm does not need to try starting times earlier than} - N_M) \geq 1 - \epsilon, \tag{9}$$

by picking M sufficiently large.

Important to note: Imagine running another Markov Chain from time $-N_M$ to 0, with the same update function ϕ and same random numbers $U_{-N_M+1}, \ldots, U_0$ as in the algorithm, but with the initial state at time $-N_M$ chosen according to the stationary distribution π.

Let $\tilde{Y}$ be the state at time 0 having the distribution π.
Further more, $\tilde{Y} \neq Y$ if the event in equation (9) does not happen, hence,

$$P(Y \neq \tilde{Y}) \leq \epsilon \tag{10}$$

Now we have,

$$\begin{aligned} P(Y = s_i) - \pi_i &= P(Y = s_i) - P(\tilde{Y} = s_i) \\ &= (P(Y = s_i, \tilde{Y} \neq s_i) + P(Y = s_i, \tilde{Y} = s_i)) \\ &- (P(\tilde{Y} = s_i, Y \neq s_i) + P(\tilde{Y} = s_i, Y = s_i)) \\ &\leq P(Y = s_i, \tilde{Y} \neq s_i) \\ &\leq P(Y \neq \tilde{Y}) \leq \epsilon \end{aligned}$$

And similarly,

$$\begin{aligned} \pi_i - P(Y = s_i) &= P(\tilde{Y} = s_i) - P(Y = s_i) \\ &= (P(\tilde{Y} = s_i, Y \neq s_i) + P(\tilde{Y} = s_i, Y = s_i)) \\ &- (P(Y = s_i, \tilde{Y} \neq s_i) + P(Y = s_i, \tilde{Y} = s_i)) \\ &\leq P(\tilde{Y} = s_i, Y \neq s_i) \\ &\leq P(Y \neq \tilde{Y}) \leq \epsilon \end{aligned}$$

Therefore we have,

$$\begin{aligned} \pi_i - P(Y = s_i) \leq \epsilon \\ P(Y = s_i) - \pi_i \leq \epsilon \end{aligned} \tag{11}$$

Hence proved.

4.2 Limitations

- Propp-Wilson is not feasible for large state problems, since the algorithm requires to start Markov chains from all possible states in the State space.
- We then resolve to **Sandwiching** as already discussed in the class.

References

[1] Siddhartha Chib and Edward Greenberg. 1995. Understanding the Metropolis-Hasting Algorithm. The American Statistician. 49 (November 1995), 327-335.

[2] James Gary Propp and David Bruce Wilson. 1996. Exact sampling with coupled Markov chains and applications to statistical mechanics. Random Struct. Algorithms 9, 1-2 (August 1996), 223-252.

[3] Olle Haggstrom, Finite Markov Chains and Algorithmic Applications. In London Mathematical Society Student Texts,Cambridge University Press, 2001.