STA 365: Applied Bayesian Statistics

Boris Babic
Assistant Professor, University of Toronto

Week 7C: Markov Chain Monte Carlo

Stationary Distribution

- Let w be a probability vector, a row vector with S components so that $\sum_{i \in S} w_i = 1$ and $w_i \geq 0$ for all $i \in S$. If $w = wP$, then this probability vector w is called the stationary distribution.
- That is,
$$w_i = \sum_S w_j p_{ji}$$
for all i in S. In words, w_i is the dot product between w and the $i$the column of P.
- In our example, $w_1 = 0.1$ and the stationary distribution is $(0.1, 0.2, 0.2, 0.2, 0.2, 0.1)'$. And indeed
$$\begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ .1 \end{bmatrix} \cdot \begin{bmatrix} 0.5 \\ 0.25 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = 0.1.$$
- The typical MCMC algorithm constructs the Markov chain so that it converges to a stationary distribution regardless of our starting points.
- We can devise a Markov chain whose stationary distribution is our desired posterior distribution $\pi(\theta|y)$, then we can run this chain to get draws that are approximately from $\pi(\theta|y)$ once the chain has converged.

Stationary Distribution: Our example

- Suppose that the person begins at state 3 which is represented by the vector $p = (0, 0, 1, 0, 0, 0)$.
- If we multiply this vector by the matrix P, we obtain the probabilities of being in all six states after one move. If we multiply p by P n times we get the probabilities of being in the different states after n moves.
- In our example, if we multiply P 100 times, we obtain the constant vector w that is equal to $(0.1, 0.2, 0.2, 0.2, 0.2, 0.1)$. This is indeed the stationary distribution.

Ergodic Theorem

- Let $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(M)}$ be M values from a Markov chain that is *aperiodic*, *irreducible* and *positive recurrent* (then the chain is ergodic), and $E[g(\theta)] < \infty$.
- Then with probability 1,

$$\frac{1}{M}\sum_{i=1}^{M} g(\theta_i) \to \int_{\Theta} g(\boldsymbol{\theta})\pi(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}, \text{ as } M \to \infty$$

where π is the stationary distribution

- This is the Markov chain analog to the strong law of large numbers (SLLN), and it allows us to ignore the dependence between draws of the Markov chain when we calculate quantities of interest from the draws.

Ergodicity

- If our Markov chain is aperiodic, irreducible, and positive recurrent, then it is ergodic and it has a unique stationary distribution.
- The ergodic theorem allows us to do Monte Carlo integration by calculating quantities of interest from our draws, ignoring the dependence between draws.
- Notice that given the finite expectation requirement, we have to be careful to have a posterior distribution that is proper.

Simulation

- Another method for demonstrating the existence of the stationary distribution of our Markov chain is by running a simulation experiment.
- We could start our random walk at a particular state, say location 3, and then simulate many steps of the Markov chain using the transition matrix P.
- The relative frequencies of our traveler in the six locations after many steps will eventually approach their respective probabilities in the stationary distribution.
- If we look at a trace plot of the chain – showing the relative frequency of a state after each step – we would expect it to jump around at first and then stabilize around the probability of that state in the stationary distribution.

Burn-in

- Since convergence usually occurs regardless of our starting point, we can usually pick any feasible (for example, picking starting draws that are in the parameter space) starting point.
- However, the time it takes for the chain to converge varies depending on the starting point.
- As a matter of practice, most people throw out a certain number of the first draws, known as the burn-in. This is to make our draws closer to the stationary distribution and less dependent on the starting point.
- However, it is unclear how much we should discard as burn-in since our draws are all slightly dependent and we do not know exactly when convergence occurs.
- The goal is to set up our model so that we are sampling from the posterior distribution. Some of the diagnostic tools we will look at there to enable us to check if we have achieved stationarity.

MCMC

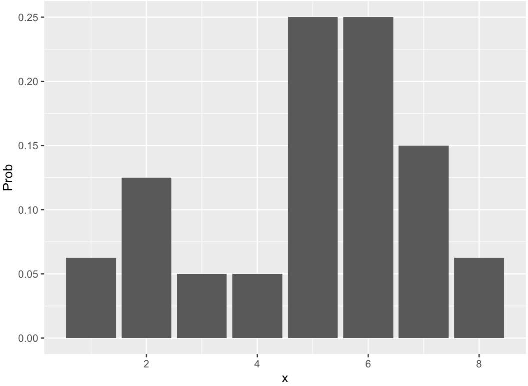
- MCMC is a class of methods in which we can simulate draws that are slightly dependent and are approximately from a (posterior) distribution
- We then take those draws and calculate quantities of interest for the (posterior) distribution
- In Bayesian statistics, there are generally two MCMC algorithms: the Gibbs sampler and the Metropolis-Hastings algorithm.
- There is also HMC (Hamiltonian Monte Carlo), implemented in Stan.

The Metropolis Algorithm

- An MCMC algorithm for obtaining a sequence of random samples from a probability distribution.
- The sequence can approximate the distribution or compute an integral.
- Named after Nicholas Metropolis, who authored the 1953 article “Equation of State Calculations by Fast Computing Machines” with Arianna W. Rosenbluth, Marshall Rosenbluth, Augusta H. Teller and Edward Teller, which to date has 50,000 citations.
- A matter of dispute which of the co-authors did the core work, with some historians suggesting it was the Rosenbluths.
- Initially developed in Los Alamos, New Mexico, for computing high dimensional integrals for work on the hydrogen bomb.

The Metropolis Algorithm

- Suppose we want to sample X which follows a discrete distribution on the integers 1-8.
- Suppose that the probabilities that $X = x$, for 1-8, are given by $p = (0.0625, 0.1250, 0.0500, 0.0500, 0.2500, 0.2500, 0.1500, 0.0625)$.



The Metropolis Algorithm

- To simulate from this probability distribution, we will take a simple random walk described as follows.

  1. We start at any possible location of our random variable X.
  2. To decide where to visit next, a fair coin is flipped. If the coin lands heads, we consider visiting the location one value to the left. If the coin lands tails, we consider visiting the location one value to the right. We call this location the candidate location.
  3. We compute
  $$R = \frac{p(candidate)}{p(current)}$$
  the ratio of the probabilities at the candidate and current locations. Notice that we only need to define the target distribution p up to proportionality.
  4. We spin a continuous spinner that lands anywhere from 0 to 1. Call the random spin Y. If $R > Y$, we indeed move to the candidate location. Otherwise, we remain at the current location.

- Steps 1 through 4 define an irreducible, aperiodic, positive recurrent Markov chain on the state values $(1, 2, ..., 8)$ where Step 1 gives the starting location and Steps 2-4 define the transition matrix P.

- We can “discover” the discrete probability distribution p by starting at any location and walking through the distribution many times repeating Steps 2, 3, and 4 (propose a candidate location, compute the ratio, and decide whether to visit the candidate location).

The Metropolis Algorithm

- The general algorithm is a generalization of the random walk example above.
- The MCMC sampling strategy sets up an irreducible, aperiodic, positive recurrent Markov chain for which the stationary distribution equals the posterior distribution of interest.

The Metropolis Algorithm

- Let $\pi(\theta|y) \propto \pi(\theta) f(y|\theta)$
- The general algorithm proceeds as follows:

  1. Start: Select a θ value for which there is positive posterior density. Call it $\theta^{(0)}$. This is the starting value.
  2. Propose: Given a current simulated value $\theta^{(j)}$, propose a new value $\theta^{(proposed)}$, which is selected at random in the interval $(\theta^{(j)} - C, \theta^{(j)} + C)$, where C is a pre-selected constant.
  3. Acceptance probability: Compute the ratio R of the posterior density at the proposed value and the current value:

     $$R = \frac{\pi(\theta^{(proposed)}|y)}{\pi(\theta^{(j)}|y)}$$

     The acceptance probability is the minimum of R and 1: $\mathrm{Pr} = \min(R, 1)$.
  4. Move or Stay: Simulate a uniform random variable U. If U is smaller than the acceptance probability Pr, move to the proposed value $\theta^{(proposed)}$, otherwise stay at the current value $\theta^{(j)}$. Hence

     $$\theta^{(j+1)} = \begin{cases} \theta^{(proposed)}, U < \mathrm{Pr} \\ \theta^{(j)} \text{ otherwise.} \end{cases}$$

- That is one step. One continues by returning to Step 2, proposing a new simulated value, computing an acceptance probability, deciding to move to the proposed value or stay, and so on.