The Metropolis-Hastings Algorithm

Econ 690

Purdue University

Outline

Motivation

- The Gibbs sampler is an incredibly powerful tool, and can be utilized in a wide variety of situations.
- In some cases, however, the conditional (or joint) distributions of interest will not take a recognizable form.
- We have discussed some alternatives for these types of cases - rejection sampling and the weighted bootstrap, for example.
- In this lecture, we describe another useful procedure for generating draws from conditional or joint distributions whose kernels are not "recognizable."
- This algorithm is termed the Metropolis-Hastings algorithm.

To motivate the potential need for such an algorithm, consider the following example:
Suppose

$$\begin{bmatrix} y_{1i} \\ y_{2i} \end{bmatrix} \overset{iid}{\sim} N\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right], \quad i = 1, 2, \ldots, n.$$

In addition, suppose that we employ a flat prior on ρ:

$$p(\rho) = \frac{1}{2} I(-1 \leq \rho \leq 1).$$

What is the posterior distribution of ρ?

First, let

$$y_i = [y_{1i}\ y_{2i}]'.$$

Then,

-

Since

$$\begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}^{-1} = (1-\rho^2)^{-1} \begin{bmatrix} 1 & -\rho \\ -\rho & 1 \end{bmatrix},$$

it follows that

$$L(\rho) \propto (1-\rho^2)^{-n/2} \exp\left(-\frac{1}{2(1-\rho^2)} \sum_{i=1}^{n} [y_{1i}^2 - 2\rho y_{1i} y_{2i} + y_{2i}^2] \right).$$

Let

$$S_{kj} \equiv \sum_i y_{ki} y_{ji}, \;\; k, j = 1, 2.$$

Then, we can write:

-

- Since $p(\rho|y) \propto L(\rho) I(-1 \leq \rho \leq 1)$ here, this could be plotted over $\rho \in (-1, 1)$ to get an idea of the shape of the density.
- However, the expression above does not correspond to any "recognizable" distribution.
- Rejection sampling or other procedures could potentially be applied. Here we will describe another possibility using the Metropolis-Hastings algorithm.

The Metropolis-Hastings Algorithm

- Pick a starting value of θ.
- Find a candidate density, proposal density or jumping distribution q. We will use the notation $q(\theta|\theta^{t-1})$ to denote that this density can (potentially) depend on the last value in the chain, θ^{t-1}. Sample a candidate value θ^* from this proposal density.
- At iteration t, accept θ^* as a draw from $p(\theta|y)$ *with probability*

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

- If θ^* is not accepted, then set

$$\theta^t = \theta^{t-1}.$$

In other words, at iteration t:

-
-
- If
-
-
- Note that the (unknown) normalizing constant of the target cancels in the ratio.

The Target is a Stationary Distribution

Before discussing more about the M-H algorithm, we will first show, for the case of a discrete state space, that the target distribution is an invariant distribution of the chain.

Suppose that θ is a discrete-valued random variable with probability mass function $p(\theta)$, and let θ^{t-1} be the current value of the Metropolis-Hastings chain, where it is assumed that $\theta^{t-1} \sim p$.

We seek to show that $\theta^t \sim p$, when θ^t is obtained according to the M-H algorithm.

Without loss of generality, let us choose two distinct points in the support of θ, and call them θ^a and θ^b.

As introduced previously, we will define the Metropolis-Hastings "acceptance ratio" as

$$r_t(\theta^a|\theta^b) = \frac{p(\theta^a)}{q(\theta^a|\theta^b)} \frac{q(\theta^b|\theta^a)}{p(\theta^b)}.$$

This ratio denotes the probability of accepting the candidate draw θ^a given that the chain is currently at θ^b, noting, of course, that the draw is automatically accepted if this ratio exceeds one.

Without loss of generality, let us label these points so that $r_t(\theta^a|\theta^b) > 1$, implying that $r_t(\theta^b|\theta^a) < 1$, since these terms are reciprocals.

Now consider

-

The first term on the right-hand side of the equation represents the probability that the next state of the chain is θ^a given that the chain is currently at θ^b.

To arrive at θ^a at the next iteration, two things must happen:

1.

2.

Our labeling of the ratio r_t implies that θ^a will always be accepted once it is drawn. As a result,

-

where the last line follows from our assumption that $\theta^{t-1} \sim p$.

Now, consider another joint probability

$$\Pr(\theta^{t-1} = \theta^a, \theta^t = \theta^b) = \Pr(\theta^t = \theta^b | \theta^{t-1} = \theta^a) \Pr(\theta^{t-1} = \theta^a).$$

In order for θ^b to be the next value of the chain, it must be both drawn from the proposal density and accepted once it has been drawn. Thus,

Note that the equality of the joint probabilities implies

$$\Pr(\theta^t = \theta^a | \theta^{t-1} = \theta^b)\Pr(\theta^{t-1} = \theta^b) = \Pr(\theta^{t-1} = \theta^a | \theta^t = \theta^b)\Pr(\theta^t = \theta^b).$$

Summing this equation over θ^a implies that

-

so that the marginals are indeed the same. Since $\theta^{t-1} \sim p$, it follows that $\theta^t \sim p$ as well.

Thus, the M-H algorithm is constructed so that the target density p is a stationary distribution of the chain.

Continuous State Spaces

We call a Markov transition density reversible for p if:

-

where $K(x, y)$ has the interpretation of the conditional density of the future state of the chain (y) given that we are currently at x. (The condition above is often called a detailed balance condition in the literature).

Note that reversibility implies $p-$invariance since:

-

or

$$\int_{\Theta} K(\theta, \theta^*|y)p(\theta|y)d\theta = p(\theta^*|y),$$

given our interpretation of the transition kernel K.

Continuous State Spaces

The transition density of the M-H chain can be thought of as involving two components: one for moves away from θ, given as

-

where

$$\alpha(\theta, \theta^*|y) \equiv \min\left\{\frac{p(\theta^*|y)}{p(\theta|y)}\frac{q(\theta|\theta^*, y)}{q(\theta^*|\theta, y)}, 1\right\}.$$

Similarly, the transition assigns a mass point to remaining at the current value of the chain, with probability given by

-

-

The M-H transition kernel is a proper density function since:

-

We can also establish that the M-H transition kernel is reversible (and thus $p-$ invariant). To see this, we will establish the invariance condition separately for both the "move away" and "remain at θ" parts. As for the latter, we need to verify

$$p(\theta|y)\delta_{\theta}(\theta^*)r(\theta|y) = p(\theta^*|y)\delta_{\theta^*}(\theta)r(\theta^*|y).$$

This is necessarily true. To see this, first consider all instances where $\theta \neq \theta^*$. In these cases, both the left and right hand sides of the equation are zero.

When $\theta = \theta^*$ both sides are obviously the same.

As for the "move away from θ" part, first suppose, without loss of generality, that $\alpha(\theta, \theta^*) < 1$ and thus, by consequence, $\alpha(\theta^*, \theta) = 1$. Thus, we consider

-

and thus the detailed balance condition is satisfied for the M-H transition density.

Gibbs Sampling as a Special Case

- Although the Gibbs sampler and M-H algorithms can be (and often are) thought of as separate instruments for posterior simulation, this distinction is rather artificial.
- In fact, one can regard the Gibbs sampler as a special case of the M-H algorithm, where the proposal density consists of the set of conditional distributions, and jumps along the conditionals are accepted with probability one.
- The following derivation illustrates this interpretation.

- Suppose we are at iteration t, and imagine breaking up the sampling of θ into k substeps (for each element of θ). Suppose we are at the j^{th} substep.
- Let all elements of θ other than θ_j be denoted as θ_{-j}, and consider the following choice of proposal density:

$$q(\theta^*|\theta^{t-1}) = p(\theta_j^*|\theta_{-j}^{t-1}, y)I(\theta_{-j}^* = \theta_{-j}^{t-1}).$$

  That is, we allow for sampling of θ_j^* from its conditional posterior distribution, and restrict all other elements of the candidate draw to equal the current values of the chain.
- In this case, the only "jumps" made are to those matching θ^{t-1} in all dimensions but the j^{th}, and for this dimension, we sample from the posterior conditional.

In this case, the M-H acceptance ratio is

-

Independence and Random Walk Chains

- Two popular M-H chains are the independence chain and the random walk chain.
- We will now discuss each of these, and later will provide examples involving their use.

- As you might guess, the random walk chain centers the proposal density over the current value of the chain, so that
  -

  where ϵ is the *increment random variable*.
- Note that, in the case where
  -
- the M-H acceptance probability reduces to (why?):
  -
- whence, jumps to regions of higher posterior probability are always accepted, while jumps to regions of lower probability are sometimes accepted.

- An alternative to the random walk chain is the independence chain.
- Here, "independence" refers to the fact that the proposal density q need not depend on θ^{t-1}.
- For example, we could choose
  -

  and specify values for θ_0 and Σ.
- Note that, for this method to work well, we would typically need to *tailor* our proposal to the problem at hand.
- For example, we could choose θ_0 to be the posterior mean (or a good approximation to it) and choose, at the same time, Σ to be a good approximation to the covariance structure. Like importance sampling, it is desirable to let $q(\theta)$ have fatter tails than the target.

- On the other hand, the random walk chain requires less problem-specific effort to approximate $p(\theta|y)$.
- For example, a random-walk M-H algorithm could proceed like this:
  1. Pick a starting θ_0 and Σ. Let's assume that we are using a $\phi(\theta; \theta^{t-1}, \Sigma)$ proposal.
  2. Cycle through the algorithm a bunch of times. Discard the first set as the burn-in, and keep the last set.
  3. Update Σ from this initial set of simulations by choosing

$$\Sigma^1 \equiv \frac{1}{M}\sum_{i=1}^{M}(\theta^{(i)} - \overline{\theta})(\theta^{(i)} - \overline{\theta})'$$

     where $\theta^{(i)}$ denotes the i^{th} post-convergence draw.
  4. Run the algorithm again, using the last value as the new starting value, and this time using $\phi(\theta; \theta^{t-1}, \Sigma^1)$ as the jumping distribution.
  5. Repeat, if necessary, until a satisfactory set of post-convergence simulations is reached.

- In practice, people often look to the acceptance rate as a guide to determine whether or not the random walk M-H algorithm is performing adequately, or can be improved.
- To fix ideas, consider the random walk chain example where

$$\theta^* \sim N(\theta^{t-1}, c^2\Sigma),$$

and Σ has already been chosen to match the posterior covariance structure.

If c is chosen to be *too small* then nearly all candidates will be accepted since the acceptance probability will be near one. However, this is not desirable since the chain will only make very small local movements at each iteration.

On the other hand, suppose that c^2 is chosen to be very large. Then, the chain will tend to get stuck at a particular iteration for large periods of time, leading to slow mixing of the chain.

Gelman, Roberts and Gilks (1995) consider the specific case of a random walk chain where the proposal and target density are normal. They show that an optimal acceptance rate for this case (in terms of minimizing the autocorrelation in the simulations) is around .45 for a scalar parameter and around .25 for six parameters.

This rule of thumb has become more widely adopted, as researchers have regarded acceptance rates near .5 as "optimal" for single parameter problems, and rates near .25 as "optimal" for multi-parameter problems.

Sketch of program structure for a random walk M-H algorithm:

1. Get initial condition of chain. Calculate the (unnormalized) posterior density (OR ITS LOG!!!) at this starting value.
2. Start loop from 2 to n. At iteration t:
   - Draw a candidate value from the proposal.
   - Calculate the posterior (or its log) at the candidate value.
   - Calculate r, the ratio of the posterior density at the candidate value to the posterior ordinate at the current value of the chain. Note that, when the sample size is reasonably large, it may be necessary to take the difference of the log ordinates and then exponentiate the result.
   - Draw $u \sim U(0, 1)$.
   - If $u < r$, then set the new value of the chain equal to the candidate, and resent the "current" posterior ordinate to be the ordinate evaluated at the candidate.
   - If $u \geq r$, set the new value of the chain equal to its old value.
3. End loop.
4. Discard pre-convergence period and use post-convergence sample for computation.

The M-H Algorithm in our Motivating Example

Recall our original example:

$$\begin{bmatrix} y_{1i} \\ y_{2i} \end{bmatrix} \overset{iid}{\sim} N\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right], \quad i = 1, 2, \ldots, n.$$

which, combined with a uniform prior for ρ produced:

$$\begin{aligned} p(\rho|y) &\propto (1-\rho^2)^{-n/2} \exp\left(-\frac{1}{2(1-\rho^2)}[S_{11} - 2\rho S_{12} + S_{22}]\right) \\ &\quad I(-1 \leq \rho \leq 1) \end{aligned}$$

- Since this posterior is not of a recognizable form, the Gibbs sampler (which, here, would be direct Monte Carlo integration!) can not be implemented.
- Instead, we consider two different M-H algorithms, both with $n = 100$ and $\rho = -.6$.
- For the first case, we consider an independence chain M-H algorithm, where the proposal density is simply the prior, which is $U(-1, 1)$.
- Note that we have not gone to great lengths to produce a "tailored" proposal density for this application.
- For the second case, we consider a random walk M-H algorithm, where the proposal density is

$$q(\rho^*|\rho^{t-1}) = \phi(\rho^*|\rho^{t-1}, c\sigma^2)$$

  and σ^2 is approximated. (Is r always well-defined?)
- We consider $c = 1$ in what follows, but provide separate results when c is too small.

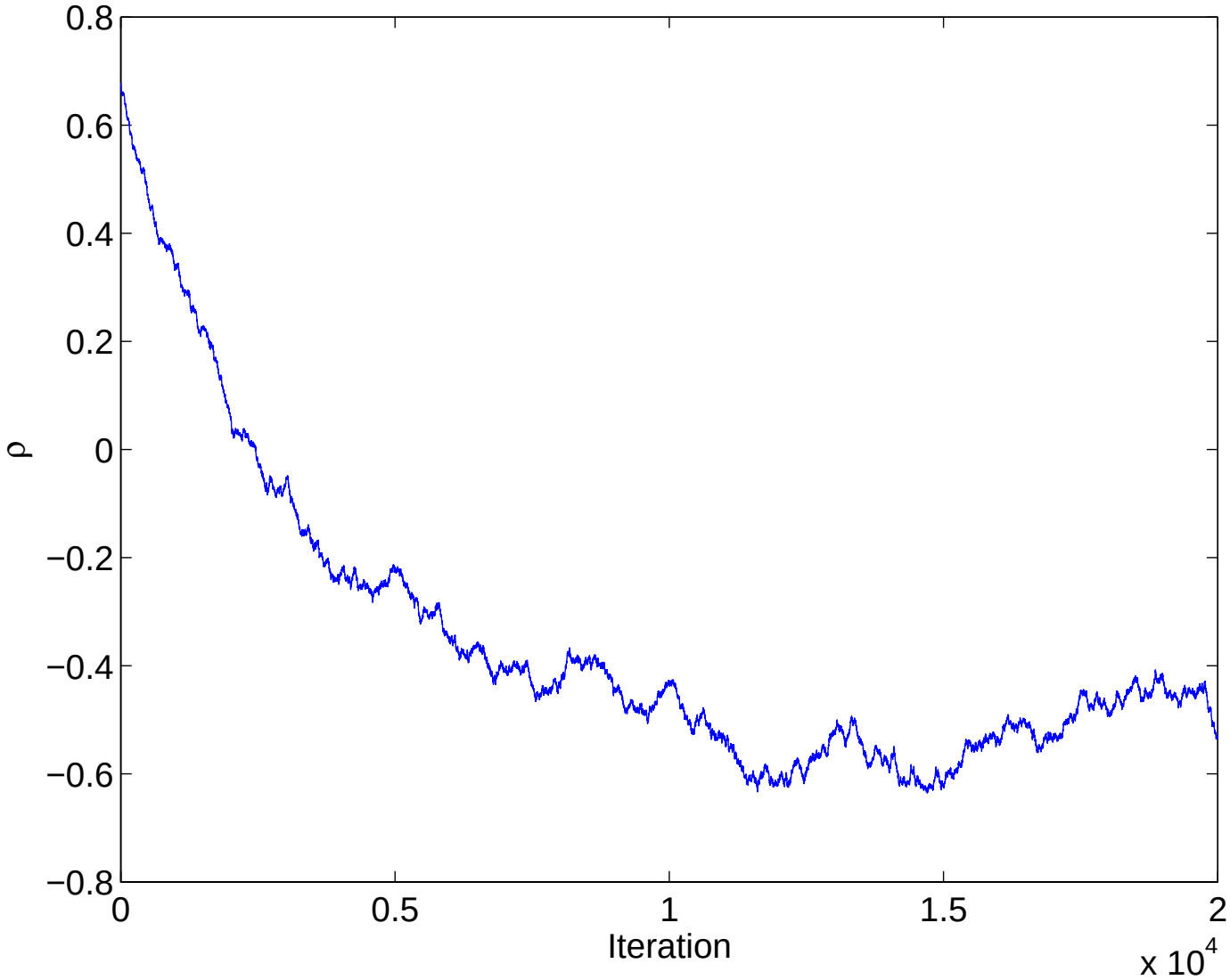
- For both chains, we choose our starting value by sampling from the prior, which in this case, is obtained by sampling from the $U(-1, 1)$ density.
- We run both algorithms for 20,000 iterations and discard the first 1,000 as the burn-in period.
- Posterior means, standard deviations, and acceptance probabilities for both algorithms are provided on the following page.

Posterior Estimates from 2 M-H Algorithms

Method	$E(\rho\|y)$	$Std(\rho\|y)$	Acceptance Rate
Numerical	-.557	.060	N/A
Ind. Chain	- .556	.061	.092
Rand. Walk, $c = 1$	- .556	.060	.693

Both chains clearly perform very well.

Consequences of choosing c = .001:



Sampling from a Double Exponential

Assume you wish to carry out Bayesian inference on a parameter θ with posterior given as:

-

This is a special case of the Laplace or double exponential distribution which has mean 0 and variance 8.

(a) Investigate the performance of an Independence chain Metropolis-Hastings algorithm using a $N\left(0, d^2\right)$ proposal density with $d = 1, 2, 3, 6, 20, 100$.

(b) Implement a random walk chain Metropolis-Hastings algorithm for this problem.

(a) Recall that the general M-H acceptance probability has the form

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

where $q\left(\right)$ is the candidate generating density.

If we substitute in the posterior and the suggested $N\left(0, d^2\right)$ candidate generating density, we obtain an acceptance probability equal to :

-

The parameter d should be chosen to optimize the performance of the M-H algorithm.

(b). The random walk chain Metropolis-Hastings algorithm we consider here generates draws according to

$$\theta^* \sim N\left(\theta^{(t-1)}, c^2\right)$$

where c varies, and ultimately is calibrated to ensure a reasonable acceptance rate.

In contrast to the independence chain M-H algorithm, a rule of thumb is available which suggests choosing c so that about 50% of the candidate draws are accepted.

For the random walk chain M-H algorithm, our acceptance probability is

$$\min\left[\frac{p\left(\theta=\theta^{*}|y\right)}{p\left(\theta=\theta^{(t-1)}|y\right)},1\right].$$

Plugging in the form given for the posterior, we obtain an acceptance probability equal to

-

The following table presents results of the posterior calculations for various values of c and d.

	Posterior Mean	Posterior Variance	Accept. Rate
True Value	0.00	8.00	—
Independence Chain M-H Algorithm			
$d = 1.00$	0.16	3.48	0.65
$d = 2.00$	0.04	7.15	0.84
$d = 3.00$	0.01	7.68	0.79
$d = 6.00$	−0.05	8.03	0.49
$d = 20.00$	−0.07	7.87	0.16
$d = 100.00$	0.12	7.41	0.03
Random Walk Chain M-H Algorithm			
$c = 0.10$	0.33	1.13	0.98
$c = 0.50$	0.20	6.87	0.91
$c = 1.00$	−0.08	7.01	0.83
$c = 4.00$	−0.03	7.93	0.52
$c = 10.00$	0.05	7.62	0.28
$c = 100.00$	0.23	6.25	0.03

- In all cases, we include $10,000$ replications (after discarding an initial 100 burn-in replications).
- For the independence chain M-H algorithm, we can see that candidate generating densities which are either too dispersed (e.g., $d = 100$) or not dispersed enough ($d = 1.0$) yield poor results.
- However, the problems associated with being not dispersed enough are clearly worse. This is consistent with our need to choose a proposal density that has heavier tails than the target (even though, strictly speaking, this is not the case).
- For the random walk chain M-H algorithm we get a similar pattern of results.
- Note that the commonly-used rule of thumb, which says you should choose c to yield approximately a 50% acceptance rate, does seem to be working well in this case.

Further Reading


Chib, S. and E. Greenberg.
Understanding the Metropolis-Hastings Algorithm
The American Statistician 49, 327-335, 1995.


Gelman, A., Roberts, G. and W. Gilks.
Efficient Metropolis Jumping Rules
in *Bayesian Statistics 5*, 1995