3

Introduction to Markov Chain Monte Carlo Methods in Molecular Evolution

Bret Larget

Departments of Statistics and Botany,
University of Wisconsin, Madison, WI, USA, brlarget@wisc.edu

3.1 Introduction

Markov chain Monte Carlo (MCMC) is a general computational technique for evaluating sums and integrals, especially those that arise as probabilities or expectations under complex probability distributions. *Monte Carlo* implies that the method is based on using chance (in the form of a pseudo-random number generator). *Markov chain* indicates a *dependent* sampling scheme with the probability distribution of each sampled point depending on the value of the previous one. Due to this dependence, MCMC samplers typically require sample sizes that are substantially larger than the sizes of independent samples produced by Monte Carlo integration methods to be able to achieve similar accuracy. However, independent sampling methods often require detailed knowledge of characteristics of the function being integrated, as their computational efficiency relies upon having a close approximation of this function. MCMC has proved to be highly useful because of its great flexibility and its success at solving many high-dimensional integration problems where other methods are computationally prohibitive.

3.1.1 A Brief History of MCMC Methods

The primary ideas behind MCMC were created by physicist Nicholas Metropolis and colleagues over fifty years ago at Los Alamos National Laboratory in the years after the Manhattan Project as part of a solution to a problem in statistical physics [22]. Hastings provided an important generalization to this pioneering work [12]. Hastings' foundational paper was ahead of its time in the statistics literature, and it took more than a decade (and the start of a personal computing revolution) before MCMC methods attracted additional attention in the statistics community. Their first use was in the form of the Gibbs sampler applied to image analysis [8]. Interest in MCMC exploded in the 1990s as it proved to be a powerful and flexible technique for solving a variety of previously unsolvable computational problems, especially those arising in Bayesian analyses. Refinement and extension of MCMC methods and their application to new problems continues to be an area of active research. MCMC methodology has completely transformed the Bayesian approach to statistics and its application to large-scale complicated modeling problems. For a more extensive description of the historical development of MCMC methods, please see the article by Hitchcock [13].

MCMC approaches in molecular evolution

MCMC approaches to problems in molecular evolution first appeared in the mid-1990s as several authors developed various methods to calculate posterior probabilities of phylogenies on the basis of aligned DNA sequence data [27, 20, 21, 17, 18]. Bayesian approaches using MCMC have since been applied to a growing number of problems in molecular evolution [16, 6]. This volume includes several applications of MCMC, including relaxation of the molecular clock assumption, the detection of positive selection, Bayesian analysis of aligned molecular sequences, models of protein evolution, evolution by genome rearrangement, and the calculation of predictive distributions and posterior mappings [25, 1, 14, 4, 5, 2]. The remainder of this chapter describes the theory behind MCMC methodology and illustrates the methods using examples in molecular evolution.

3.2 Bayesian Inference

The Bayesian approach to statistical inference in molecular evolution most often fits into the following general framework. (In what follows, I use p to stand for a generic probability density and let the arguments distinguish them.) A likelihood function $p(D \mid \theta)$ describes the probability (or probability density) of data D given the values of parameters θ. The prior distribution $p(\theta)$ expresses the uncertainty in the parameters prior to observation of the data. Bayes' theorem provides the form of the posterior distribution $p(\theta \mid D)$, the probability distribution that describes uncertainty in the parameters after observing the data and the object of all Bayesian inference

$$p(\theta \mid D) = \frac{p(D \mid \theta)p(\theta)}{p(D)}. \tag{3.1}$$

The denominator $p(D)$ is the marginal probability of the data, averaged over all possible parameter values weighted by their prior distribution. Formally, we can write

$$p(D) = \int_{\Theta} p(D \mid \theta)p(\theta)\,\mathrm{d}\theta, \tag{3.2}$$

where Θ is the parameter space for θ. In almost all problems of practical interest, it is not tractable to compute $p(D)$ directly, the normalizing constant of the posterior distribution. MCMC offers a means to make Bayesian inferences without the need to compute this normalizing constant.

In a typical application in Bayesian inference in molecular evolution, the parameter θ contains both discrete and continuous components. For example, θ might include the discrete tree topology and the continuous branch lengths and nucleotide substitution model parameters. The single integral in (3.2) represents a multiple sum over discrete parameters and a multiple integral over continuous parameters.

Calculating expected values

Usually, the posterior distribution $p(\theta \mid D)$ is a complicated function over a large parameter space that cannot be described adequately in full. We typically are interested in various summaries of the posterior distribution, all of which are posterior expectations of some function of the parameters. For example, the posterior probability of a tree topology is the expected value of the indicator variable for that tree topology, and the posterior density of a branch length can be summarized in part by its mean.

To simplify notation by eliminating the explicit dependence on the observed data, define the unnormalized posterior distribution to be $h(\theta) \equiv p(D \mid \theta)p(\theta)$. With this notation, the posterior expected value of a function of the parameter space is defined to be

$$\mathbb{E}[g(\theta)] = \frac{\int_\Theta g(\theta)h(\theta)\,\mathrm{d}\theta}{\int_\Theta h(\theta)\,\mathrm{d}\theta}. \tag{3.3}$$

The idea behind MCMC is to take a (dependent) random sample of points $\{\theta_i\}$ from the unnormalized target function $h(\theta)$ by simulating a Markov chain whose stationary distribution is proportional to $h(\theta)$. We can then approximate expectations with simple arithmetic averages,

$$\mathbb{E}[g(\theta)] \approx \frac{1}{n}\sum_{i=1}^{n} g(\theta_i)\ . \tag{3.4}$$

We note that while most applications of MCMC to problems in molecular evolution have been part of Bayesian analyses, computations in the form of (3.4) can arise in non-Bayesian approaches as well. MCMC is a general-purpose tool.

3.3 The Metropolis-Hastings Algorithm

The most common form of MCMC is the Metropolis-Hastings algorithm. The idea is to create a *proposal distribution* q on the parameter space Θ. Instead of using q to generate a sequence of points sampled from Θ, we use q to generate a *candidate* for the next sampled point that is either accepted or rejected with some probability. If the candidate is rejected, the current point is sampled again. The random acceptance of proposals effectively changes the transition probabilities. A clever choice of acceptance probabilities results in a "metropolized" Markov transition matrix q' whose stationary distribution is proportional to the unnormalized target distribution h. Remarkably, the choice of q is nearly arbitrary. It suffices for q to be *irreducible*–from any points $x, y \in \Theta$, it should be possible to get from x to y through a finite number of transitions under q.

The initial sample point θ_0 may be arbitrary. If the current state is $\theta_i = x$, the Metropolis-Hastings algorithm generates candidate y from the distribution $q(\cdot \,|\, x)$. The acceptance probability is

$$r(y \,|\, x) = \min \left\{ 1, \frac{h(y)q(x \,|\, y)}{h(x)q(y \,|\, x)} \right\} . \tag{3.5}$$

With probability $r(y \,|\, x)$, we set $\theta_{i+1} = y$; otherwise $\theta_{i+1} = x$. In the special case where $q(x \,|\, y) = q(y \,|\, x)$ for each x and y, the proposal density drops out of (3.5). The original method in Metropolis et al. [22] assumed this symmetry, and Hastings [12] made the generalization that allowed nonsymmetric proposal distributions. Notice as well that the target distribution only needs to be known up to a normalizing constant, as it is only necessary to be able to compute the ratio of the target distribution evaluated at any two points. The ratio $q(x \,|\, y)/q(y \,|\, x)$ is known as the *Hastings ratio* or the *proposal ratio*. The *target ratio* $h(y)/h(x)$ is the *posterior ratio* in a Bayesian setting where it is the product of a *likelihood ratio* and a *prior ratio*.

3.3.1 Why Does the Metropolis-Hastings Algorithm Work?

The stationary distribution π of a Markov chain with transition function $q'(y \,|\, x)$ satisfies

$$\int_{x \in \Theta} \pi(x) q'(y \,|\, x) \, \mathrm{d}x = \pi(y) \qquad \text{for each } y \in \Theta. \tag{3.6}$$

A stronger condition is for the chain to satisfy *detailed balance*,

$$\pi(x) q'(y \,|\, x) = \pi(y) q'(x \,|\, y) \qquad \text{for all } x, y \in \Theta. \tag{3.7}$$

Markov chains that satisfy detailed balance are *time-reversible*. The rate of transition from x to y is the same as that from y to x for each x and y, so the probability of any sequence of transitions would be the same in forward and backward time. Detailed balance of the target distribution h is easy to check for the Metropolis-Hastings algorithm. First, notice that the actual transition probability density is $q'(y \,|\, x) = q(y \,|\, x) r(y \,|\, x)$ for $x \neq y$. Therefore, we have

$$\begin{aligned} h(x)q'(y \mid x) &= h(x)q(y \mid x) \min\left\{1, \frac{h(y)q(x \mid y)}{h(x)q(y \mid x)}\right\} \\ &= \min\left\{h(x)q(y \mid x), h(y)q(x \mid y)\right\}. \end{aligned}$$

The last expression is symmetric in x and y, which implies that the first expression must have the same value if x and y are switched, so detailed balance is satisfied.

The Gibbs sampler

The Gibbs sampler is a special case of the Metropolis-Hastings algorithm in which the proposal distributions are the full conditional distributions of some part of the parameter space conditional on the rest. Suppose that the parameter vector $\theta = (\theta_{[1]}, \theta_{[2]}, \ldots, \theta_{[d]})$ is partitioned into d blocks. The idea behind the Gibbs sampler is to propose new values of a block of parameters $\theta_{[k]}$ from their full conditional distribution given the current values of all other parameters, denoted $p(\cdot \mid \theta_{[-k]})$, where $\theta_{[-k]}$ includes all of θ except for the kth block. The proposed values are always accepted. The *systematic-scan* Gibbs sampler updates blocks in a fixed order, cycling through them all. The *random-scan* Gibbs sampler randomly picks a block of parameters to estimate repeatedly.

We can understand why the Gibbs sampler works by checking the Metropolis-Hastings acceptance probability for one step of the Gibbs sampler. In updating the kth block given the current state θ, the candidate is

$$\theta^* = (\theta_{[1]}, \ldots, \theta_{[k-1]}, \theta^*_{[k]}, \theta_{[k+1]}, \ldots, \theta_{[d]}).$$

The posterior ratio is $h(\theta^*)/h(\theta) = p(\theta^*)/p(\theta)$ and the proposal ratio is $p(\theta_{[k]} \mid \theta_{[-k]})/p(\theta^*_{[k]} \mid \theta_{[-k]})$. Conditioning on parameters outside the kth block leads to $p(\theta^*) = p(\theta_{[-k]} \cap \theta^*_{[k]}) = p(\theta_{[-k]})p(\theta^*_{[k]} \mid \theta_{[-k]})$ with a similar expression for $p(\theta)$. The acceptance probability is then

$$\begin{aligned} r &= \min\left\{1, \frac{p(\theta^*)}{p(\theta)} \times \frac{p(\theta_{[k]} \mid \theta_{[-k]})}{p(\theta^*_{[k]} \mid \theta_{[-k]})}\right\} \\ &= \min\left\{1, \frac{p(\theta_{[-k]})p(\theta^*_{[k]} \mid \theta_{[-k]})}{p(\theta_{[-k]})p(\theta_{[k]} \mid \theta_{[-k]}))} \times \frac{p(\theta_{[k]} \mid \theta_{[-k]})}{p(\theta^*_{[k]} \mid \theta_{[-k]})}\right\} \\ &= 1. \end{aligned}$$

The advantage of the Gibbs sampler is that proposals are always accepted. While one might think that this feature would invariably lead to a sampler that moves through the parameter space rapidly, this is not always the case. It is well-known that the Gibbs sampler can mix slowly if highly correlated parameters are in different blocks. The other practical difficulty is that the flexibility of the Metropolis-Hastings approach in choosing a proposal distribution is lost. Candidates from the full conditional distributions are often not easy to simulate, which can make the problem difficult. In the case where full conditional distributions are available and easy to simulate, Gibbs sampling will be a good choice. Experience indicates that the more general Metropolis-Hastings approaches are often a more practical solution for many statistical problems in molecular evolution.

3.3.2 An Example in Bayesian Phylogenetic Inference

The Bayesian approach to estimating phylogenies from aligned DNA sequence data as implemented in the programs BAMBE [24] and MrBayes [15] uses MCMC to sample from the joint posterior probability distribution of phylogenies and nucleotide substitution model parameters. The state space for the Markov chain takes the form $\theta = (\tau, t, Q)$, where τ is the tree topology, t is a vector branch length, and Q is the generator of the continuous-time nucleotide substitution process. The MCMC samplers used in both BAMBE and MrBayes are actually *hybrid samplers* that combine several Metropolis-Hastings samplers, each of which samples from only part of the parameter space. BAMBE has a proposal distribution q_1 that updates the tree (both τ and t) while leaving Q fixed and a second proposal q_2 that updates Q leaving the tree fixed. BAMBE cycles back and forth between q_1 and q_2 proposals. Effectively, the hybrid sampler in BAMBE is a systematic-scan Gibbs sampler with a Metropolis-Hastings proposal at each step. In contrast, MrBayes has a collection of proposals to update parts of Q and another collection of proposals to update the tree. At each stage, one of these proposals is selected at random. Running only one chain, MrBayes uses a hybrid sampler that is a random-scan Gibbs sampler with a Metropolis-Hastings update at each step. Tierney [26] provides further examples and theoretical justifications of the use of hybrid MCMC samplers.

Description of the Local algorithm

BAMBE and MrBayes both use a local update method first described by Larget and Simon [17] to update unrooted trees. A description of this algorithm and the associated acceptance probability serves to illustrate the ideas of this section on an application of MCMC in molecular evolution. The acceptance probability originally reported in [17] was, in fact, incorrect. I am extremely grateful to Mark Holder, Paul Lewis, and David Swofford, who reported this to me quite recently.

The LOCAL algorithm begins by selecting an internal branch of the tree at random. (Please see Figure 3.1 for a graphical description of this algorithm.) The nodes at the ends of this branch are each connected to two other branches. One of each pair is chosen at random. Imagine taking these three selected adjacent edges and stringing them like a clothesline from left to right, where the direction is also selected at random. The two endpoints of the first branch selected will each have a subtree hanging like a piece of clothing strung to the line. The algorithm proceeds by multiplying the three selected branches by a common random amount, akin to stretching or shrinking the clothesline. Finally, the leftmost of the two hanging subtrees is disconnected and reattached to the clothesline at a location selected uniformly at random. This is the candidate tree.

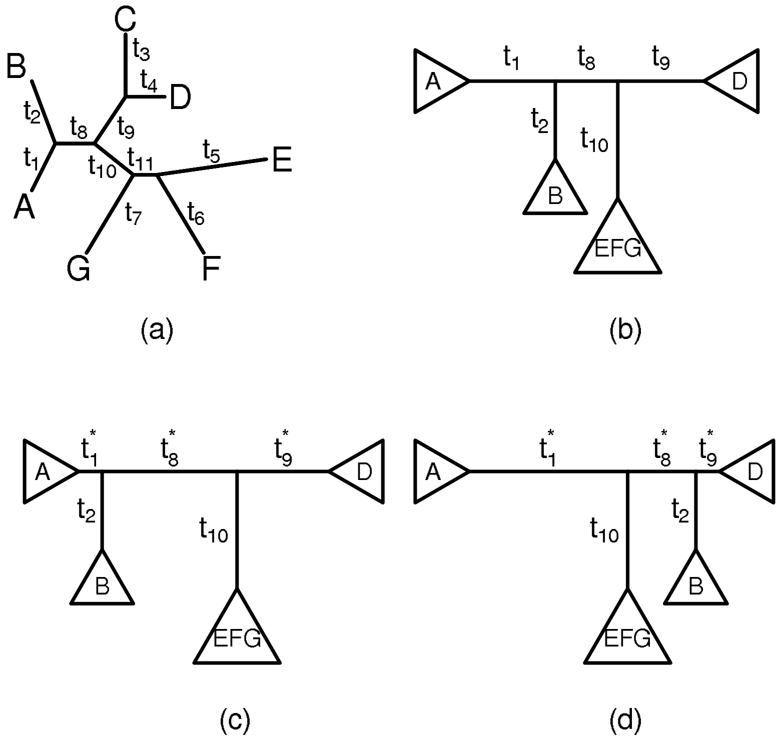

Fig. 3.1. Local update algorithm. (a) A seven-taxon unrooted tree. (b) A randomly chosen local neighborhood of the tree. Triangles represent subtrees. (c) A candidate tree with the same tree topology. (d) A candidate tree with a different tree topology.

Next, we then ask with what probability the candidate should be accepted. See Figure 3.1 (a), which displays a sample seven-taxon unrooted tree for the definition of the parameters in the following description. Suppose that we began by selecting the internal branch with length t_8 that separates taxa A and B from the rest, that we selected branches with lengths t_1 and t_9 from each side, and that we oriented these branches as shown in Figure 3.1 (b). The probability of this part of the proposal is $(1/b) \times (1/2)^3$ because there are $b = 4$ internal branches and we made three binary choices.

Let $m = t_1 + t_8 + t_9$ be the current length of the clothesline. We select the new length to be $m^* = m \exp(\lambda(U_1 - 0.5))$, where U_1 is a Uniform$(0, 1)$

random variable independent of everything else. It is straightforward to show that given m, m^* has density

$$q(m^* \mid m) = \frac{1}{\lambda m^*} \qquad \text{for } m\mathrm{e}^{-\lambda/2} < m^* < m\mathrm{e}^{\lambda/2}. \tag{3.8}$$

Suppose as well that $u = t_1$ is the current distance from the left endpoint of the clothesline to the B subtree. Given m^*, we pick a new distance $u^* = U_2 m^*$, where U_2 is another independent Uniform$(0,1)$ random variable. The distance from the left end point to the EFG subtree changes proportionally from $v = t_1 + t_8$ to $v^* = m^* v/m$. There are now two cases. If $u^* < v^*$ (see Figure 3.1 (c)), the tree topology does not change and the updated branch lengths are $t_1^* = u^*$, $t_8^* = v^* - u^*$, and $t_9^* = m^* - v^*$. Otherwise (see Figure 3.1 (d)), $v^* < u^*$ and the tree topology does change. The new branch lengths are $t_1^* = v^*$, $t_8^* = u^* - v^*$, and $t_9^* = m^* - u^*$. The probability density of this part of the proposal is the density of u^* given m^*, which is uniform, $q(u^* \mid m^*) = 1/m^*$ on $(0, m^*)$.

The joint proposal density given the local choice of the subtree to update is

$$\begin{aligned} q(u^*, v^*, m^* \mid m, u, v) &= q(m^* \mid m) q(u^* \mid m^*) \delta_{\{v^* = vm^*/m\}} \\ &= \frac{\delta_{\{v^* = vm^*/m\}}}{\lambda (m^*)^2}, \end{aligned} \tag{3.9}$$

where δ is Dirac's delta function. If x is the original tree and y is the candidate tree, the acceptance probability for the LOCAL proposal is

$$\min\left\{1, \frac{h(y)\left(\frac{1}{b}\right)\left(\frac{1}{2}\right)^3 \times \frac{\delta_{\{v = v^* m/m^*\}}}{(\lambda m^2)}}{h(x)\left(\frac{1}{b}\right)\left(\frac{1}{2}\right)^3 \times \frac{\delta_{\{v^* = vm^*/m\}}}{(\lambda (m^*)^2)}}\right\} = \min\left\{1, \frac{h(y)}{h(x)} \times \left(\frac{m^*}{m}\right)^3\right\}$$

since

$$\frac{\delta_{\{v^* m/m^*\}}}{\delta_{\{v^*\}}} = \frac{\delta_{\{v^*\}}/(m/m^*)}{\delta_{\{v^*\}}} = \frac{m^*}{m}$$

The incorrect acceptance probability published previously [17] had a power of 2 rather than the correct power of 3.

3.4 Reversible Jump MCMC

In all of the examples we have considered to this point, the state space has had a fixed number of parameters. One can imagine a number of problems arising in molecular evolution where this need not be the case. For example, consider a Bayesian approach to phylogeny estimation from aligned DNA sequence data in which there is a prior distribution on the class of likelihood model. Specifically, suppose we think, for example, that the HKY85 and TN93 models are equally likely. The HKY85 model has one fewer free parameter than the TN93 model. We could define different Metropolis-Hastings samplers to update the Q matrix separately within each model, but we would also need to be able to switch between models. In this example, the number of parameters is itself a parameter of the model. *Reversible Jump MCMC* describes how to extend the Metropolis-Hastings approach to allow jumps between subspaces of different dimensions [11].

A typical situation is that we want a set of proposal distributions between subspaces Θ_1 and Θ_2 where the kth subspace has m_k parameters and $m_1 \neq m_2$. The key idea to make this work is *dimension matching*. The basic idea is to supplement each set of parameters with different numbers of random variables so that the dimensions match and then to transform one set into the other with a bijection. Let θ_1 and θ_2 be two states in Θ_1 and Θ_2, respectively. Then the vectors $\phi^{(1)} = (\theta_1, u_1)$ and $\phi^{(2)} = (\theta_2, u_2)$ each have length $d = m_1 + n_1 = m_2 + n_2$, where u_k is an n_k-vector and n_k are chosen so that the dimensions match. (It is often the case that $n_k = 0$ for the larger subspace.) Suppose that T_1 is a bijection so that $\phi^{(2)} = T_1(\phi^{(1)})$ and $T_1^{-1} = T_2$.

The proposal from $\theta_x \in \Theta_x$ to $\theta_y \in \Theta_y$ follows this procedure.

1. Generate random vector u_x, which has length n_x.
2. Let $\phi^{(x)} = (\theta_x, u_x)$.
3. Evaluate $\phi^{(y)} = T_x(\phi^{(x)})$.
4. Project $\phi^{(y)} = (\theta_y, u_y)$ into first m_y coordinates to determine θ_y.

3.4.1 Acceptance probability

The acceptance probability for this proposal includes a Jacobian in addition to the usual ratios. The Jacobian for the transformation is $|\det J_x|$, where J_x is a $d \times d$ square matrix whose i, j entry is

$$\{J_x\}_{ij} = \frac{\partial \phi_i^{(y)}}{\partial \phi_j^{(x)}} .$$

The acceptance probability is

$$r(\theta_y \mid \theta_x) = \min \left\{ 1, \frac{h(\theta_y) q(\theta_x \mid \theta_y)}{h(\theta_x) q(\theta_y \mid \theta_x)} \times |\det J_x| \right\} .$$

Example

We illustrate these ideas with the example of modeling the nucleotide substitution process in which we have equal prior probabilities that Q is from either an HKY85 or a TN93 class of models. Each of these models has three free parameters for the base composition that do not need to change in moving between models. HKY85 has a single transition/transversion parameter κ, while TN93 allows two different transition rates, κ_1 for purines and κ_2 for pyrimidines. In a proposal from TN93 to HKY85, assume we set κ to be the mean of κ_1 and κ_2. To attain detailed balance, we need for the proposal density given κ to have support on all positive (κ_1, κ_2) such that $\kappa = (\kappa_1 + \kappa_2)/2$. We could do this by letting $u \,|\, \kappa$ be a Uniform$(-\kappa, \kappa)$ random variable. We have this bijection:

$$T_1(\kappa, u) = (\kappa - u, \kappa + u) \qquad \text{and} \qquad T_2(\kappa_1, \kappa_2) = \left(\frac{\kappa_1 + \kappa_2}{2}, \frac{\kappa_2 - \kappa_1}{2} \right) .$$

We have

$$J_1 = \begin{bmatrix} \dfrac{\partial(\kappa - u)}{\partial \kappa} & \dfrac{\partial(\kappa - u)}{\partial u} \\[2ex] \dfrac{\partial(\kappa + u)}{\partial \kappa} & \dfrac{\partial(\kappa + u)}{\partial u} \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

so that $|\det J_1| = 2$. By a similar calculation, $|\det J_2| = 1/2$.

Suppose that the unnormalized posterior distribution is h and we are interested in calculating the acceptance probabilities for a proposal from $\theta_1 = \kappa$ to $\theta_2 = (\kappa_1, \kappa_2)$. If a_1 is the probability that we propose that a TN93 Q matrix given the current model is HKY85 and a_2 is the probability of the reverse situation, the acceptance probability is determined as

$$r(\theta_2 \,|\, \theta_1) = \min \left\{ 1, \frac{h(\theta_2)}{h(\theta_1)} \times \frac{a_2}{a_1/(2\kappa)} \times 2 \right\}$$

provided that $(\kappa_1 + \kappa_2)/2 = \kappa$. The acceptance probability of a proposal in the other direction is

$$r(\theta_1 \,|\, \theta_2) = \min \left\{ 1, \frac{h(\theta_1)}{h(\theta_2)} \times \frac{a_1/(2\kappa)}{a_2} \times \frac{1}{2} \right\} .$$

3.5 Assessing Convergence

The theoretical justification of MCMC as a computational tool is that sample averages converge to their expected values. However, this result is asymptotic and, in practice, no chain can be run forever. We must therefore address the following practical questions: How long should a chain be run? Should we discard the initial portion of a sample? Should we subsample the Markov chain? How do we assess the accuracy of the MCMC estimates? How can we compare MCMC samplers? None of these questions has a definitive answer, and rarely can we have absolute trust in the MCMC calculations. Despite this, there are steps that will increase confidence in the results.

We will illustrate these ideas in the context of a very simple example. Suppose that we are interested in estimating a branch length from a two-taxon tree under the Jukes-Cantor model for a data set in which n_1 sites are unvaried and n_2 sites are variable. We will assume an exponential prior distribution with rate λ. The density is $p(t) = \lambda \mathrm{e}^{-\lambda t}$. The probabilities of the possible site patterns are $\frac{1}{4}\left(\frac{1}{4}+\frac{3}{4}\mathrm{e}^{-\frac{4}{3}t}\right)$ for unvaried sites and $\frac{1}{4}\left(\frac{1}{4}-\frac{1}{4}\mathrm{e}^{-\frac{4}{3}t}\right)$ for varied sites. Putting these two probabilities together, the unnormalized posterior distribution is as follows.

$$h(t)=\left(\frac{1}{4}\right)^{n_1+n_2}\left(\frac{1}{4}+\frac{3}{4}\mathrm{e}^{-\frac{4}{3}t}\right)^{n_1}\left(\frac{1}{4}-\frac{1}{4}\mathrm{e}^{-\frac{4}{3}t}\right)^{n_2}\left(\lambda\mathrm{e}^{-\lambda t}\right).$$

Consider updating the branch length by choosing a new value uniformly at random from a window of half-width w centered at the current value, reflecting off the origin when a negative branch length is proposed. Specifically, $t^* = |t+U|$, where U is uniformly distributed between $-w$ and w. It is straightforward to show that the proposal ratio is one. Acceptance probabilities are then $\min\{1, h(t^*)/h(t)\}$.

In a specific numerical example, suppose that $n_1 = 70$, $n_2 = 30$, and $\lambda = 5$. We will compare results for two choices of w, namely $w = 0.1$ and $w = 0.5$. In each case, we will begin with an initial edge length of 5.0 (a poor choice) and update the edge length 2000 times (much shorter than we might typically do). Figure 3.2 displays summaries of the MCMC samples.

3.5.1 Burn-in

The initial portion of an MCMC sample is often discarded as *burn-in*. The logic behind this practice is that the initial portion of a run will typically be highly dependent on the starting value of the Markov chain, and if this value is not likely under the stationary distribution, the sample would be biased toward the initial point. The estimate

$$\sum_{i=m+1}^{n} g(\theta_i)/(n-m),$$

which discards the first m sample points, is typically a more accurate estimate of the expectation of g under the target distribution when m is substantially larger than one in the usual case of an atypical initial state.

3.5.2 Trace Plots

This then begs the question of how one should determine the portion of a sample to discard. *Trace plots* of one-dimensional summaries of the state space are a crude but often effective way of determining burn-in. For Bayesian MCMC sampling from a posterior distribution of trees, both BAMBE and MrBayes produce trace plots of the log likelihood. When beginning runs at random initial trees, it is typical for the log-likelihood to increase dramatically as the chain rapidly approaches an area of the state space of relatively high posterior probability before changing behavior dramatically and reaching a plateau around which the likelihood fluctuates for the remainder of the run.

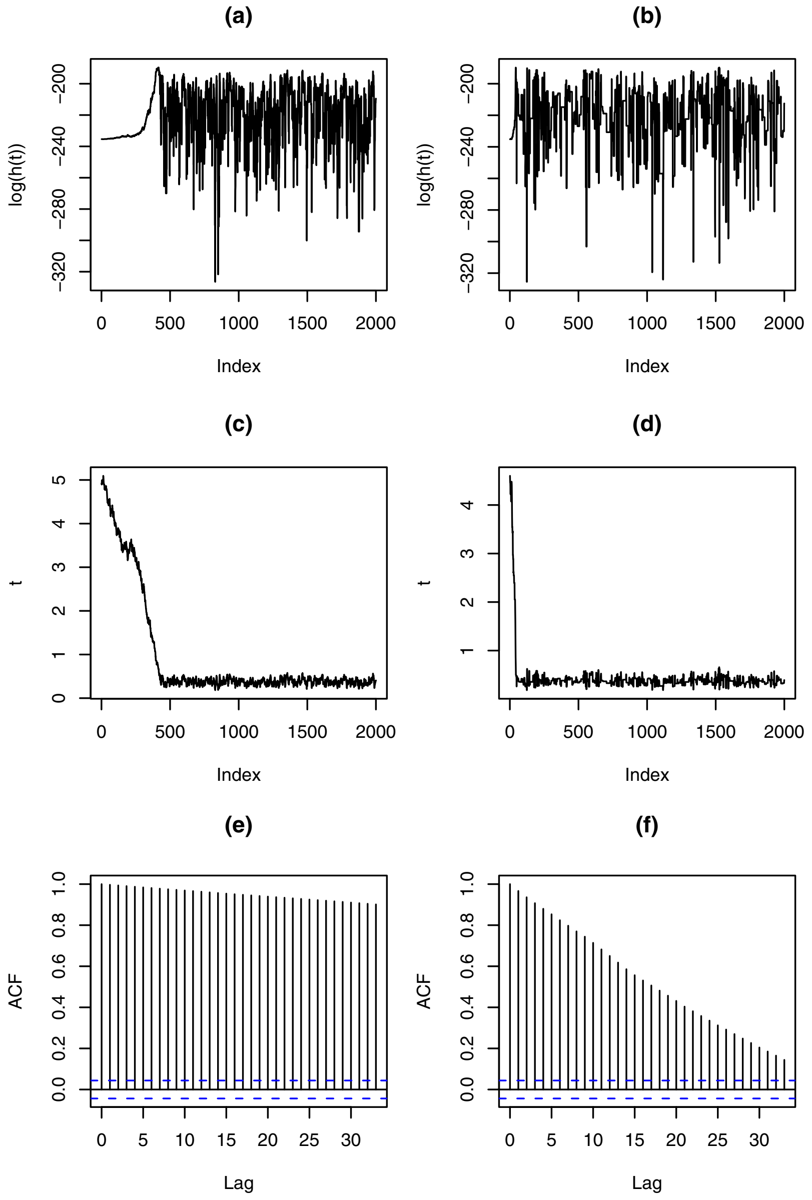

Fig. 3.2. Trace plots in the Jukes-Cantor example. (a) Unnormalized posterior versus index with window size $w = 0.1$. (b) Unnormalized posterior versus index with window size $w = 0.5$. (c) Branch length versus index with window size $w = 0.1$. (d) Branch length versus index with window size $w = 0.5$. (e) Autocorrelation plot of sampled edge lengths with $w = 0.1$. (f) Autocorrelation plot of sampled edge lengths with $w = 0.5$.

Trace plots of the log-likelihood are good indicators of minimum values for burn-in but are insufficient on their own to assess convergence. If the chain were stuck in a local minimum, the behavior exhibited in the trace plot would be indistinguishable from the trace plot behavior of a well-mixing chain. Trace plots of other one-dimensional summaries of the state space, such as parameter values in the substitution models or the sum of all branch lengths of the tree, should also be examined for visual evidence that after burn-in the initial portion of the sample looks similar to the end portion.

The trace plots of h and t displayed in Figure 3.2 for the Jukes-Cantor example provide a means to informally assess convergence. The trace plots of the edge length in each run more clearly indicate the necessary time to convergence. In the run with $w = 0.1$, we need to discard at least the first 500 sample points, and I would discard a few more to be safe, say at least the first 10% of the sample after apparent convergence. Discarding the first $m = 700$ points of each run suffices for this example.

For the run with $w = 0.1$, a 95% credibility region for the edge length is $(0.24, 0.52)$. The post-burn-in credible region for the run with $w = 0.5$ is quite similar, $(0.25, 0.52)$. Had we not discarded the initial part of the run, the 95% credible region would have been either $(0.25, 4.48)$ or $(0.26, 0.62)$, with right endpoints substantially too large in both cases. Of course, we could have lessened the bias due to burn-in by either running the chains for many more iterations or by using an initial edge length closer to the center of the posterior distribution.

Figure 3.2 also displays the autocorrelation function of the sampled branch lengths for both window sizes. Notice that in this example mixing is significantly faster using the larger window size. With $w = 0.5$, the Markov chain has reached approximate independence after about 40 steps. Dependence decreases much more slowly in the case with a smaller window. Acceptance probabilities can offer a clue about convergence speed. In this example, updates with $w = 0.1$ were accepted 73% of the time as opposed to only 23% of the time for $w = 0.5$. Acceptance probabilities between 0.15 and 0.40 often indicate chains that mix relatively well. This simple example suffices to show that adjustment of tuning parameters can have a large effect on mixing properties; running slowly mixing chains for a long time can compensate. Notice also in Figure 3.2 as well that the trace plots of the edge lengths are more informative about burn-in than are the trace plots of the posterior distribution. With larger trees, larger models, and longer sequences, it is highly advisable to examine trace plots of many posterior summaries and to complete several very long MCMC simulations.

3.5.3 How Many Chains?

While there is no consensus on how many chains should be run, I advocate running several long chains from widely disparate starting values. The advantage is that if the post-burn-in summaries of important characteristics of the target distribution are similar, there is evidence that the Markov chains are successfully mixing. In contrast, summaries from independent chains that are wildly different are a certain indicator that one or more chains has not reached stationarity or that the chains are mixing so slowly that substantially longer runs are needed to obtain more accurate calculations. If one has access to several processors, the real time to take several long samples is the same as the time to complete a single run on one machine. The other advantage to having several independent estimates of posterior characteristics is that simple and accurate estimates of Monte Carlo error are easily computed. Estimates of Monte Carlo error from single runs depend on estimates of the dependence in a single chain. Such estimates can vary considerably with the method used to estimate the dependence.

3.5.4 How Often Should the Markov Chain Be Subsampled?

From a purely statistical perspective, there is nothing to gain from subsampling–a loss of data represents a potential loss of information. However, from a practical sense, because chains tend to be highly dependent, regular subsamples of the Markov chain output will typically be just as accurate as if the entire post-burn-in sample were saved and summarized. Practical issues involving the ease of the storage and analysis of the output of a long MCMC run often outweigh the negligible potential loss of information from subsampling.

3.5.5 How Long Should a Chain Be Run?

There are formal methods to decide upon chain convergence that are based on running a number of chains in parallel and stopping when variability in the chains' estimates of a number of scalar posterior summaries between chains is small relative to the variability within each chain [7]. A cruder yet effective approach is to learn from preliminary runs how much time is required to run a chain a specified number of steps, extrapolate this to the time available (such as overnight), run several independent chains in parallel in that time, and calculate the Monte Carlo standard error of each important scalar posterior characteristic from the estimates in each independent chain. If this Monte Carlo error estimate is too big for the problem at hand, then it may be that longer runs are necessary (or that a better proposal distribution is required).

3.6 Metropolis-Coupled MCMC

There are many strategies for improving the sampling properties of MCMC approaches. One of the most useful is Metropolis-coupled MCMC, or MCMCMC [9]. The idea is to run several simultaneous chains on the state space Θ.

Only one of these chains, the *cold chain*, needs to have the correct stationary distribution. The other *heated* chains are typically selected to have stationary distributions that are flatter than the stationary distribution of the cold target chain. The heated chains are able to move more easily between regions where the target is relatively high.

After some number of steps of each chain, a move that swaps the states of two of the chains is proposed and accepted or rejected according to a Metropolis-Hastings rule. This type of proposal can effectively jump the cold chain to a different portion of the parameter space. Only the sampled points from the cold chain are saved as a sample from the target distribution. Suppose that the chains have unnormalized target distributions $\{h_i\}$ for $i = 1, \ldots, m$. If the current states in chains i and j are x_i and x_j, respectively, the probability of accepting a proposed swap of the two states is

$$\min\left\{1, \frac{h_i(x_j)h_j(x_i)}{h_i(x_i)h_j(x_j)}\right\} .$$

Generally speaking, running m chains requires m times the computational effort that running a single chain would require. This trade-off can be worthwhile if the cold chain is very slow-mixing.

Figure 3.3 illustrates these ideas in a small artificial example. The target function (solid line in Figure 3.3(a)) contains two separate modes of relatively high probability separated by a region of very low probability. We are using a proposal chain that proposes new values in a small uniform window extending one unit below and above the current position. Crossing the valley between the two peaks in the cold chain requires an unlikely proposal and acceptance of several consecutive steps through the low region between the modes. The dashed line is a single heated distribution. The same proposal distribution will more easily cross between the two modes. In a simulation, both chains began at the value $x = 20$, are updated by Metropolis-Hastings individually, and are then followed by a proposed swap after each set of updates. The chains ran for 100,000 cycles of updates. Figure 3.3(b) shows a histogram of the sampled values that matches the target quite well. Figure 3.3(c) shows the same sampled values in a trace plot versus the iteration number. It is clear that the sampled chain jumped between modes many times during the simulation. Figure 3.3(d) shows the sampled values from a regular Metropolis-Hastings MCMC simulation in a trace plot versus iteration number. This particular realization jumped between modes only once. Simulation-based sample estimates of target characteristics will likely be inaccurate and will be highly sensitive to the decision on when to stop the chain. A substantially longer simulation in which the sampled chain crossed the low region several times would be required for accurate estimation.

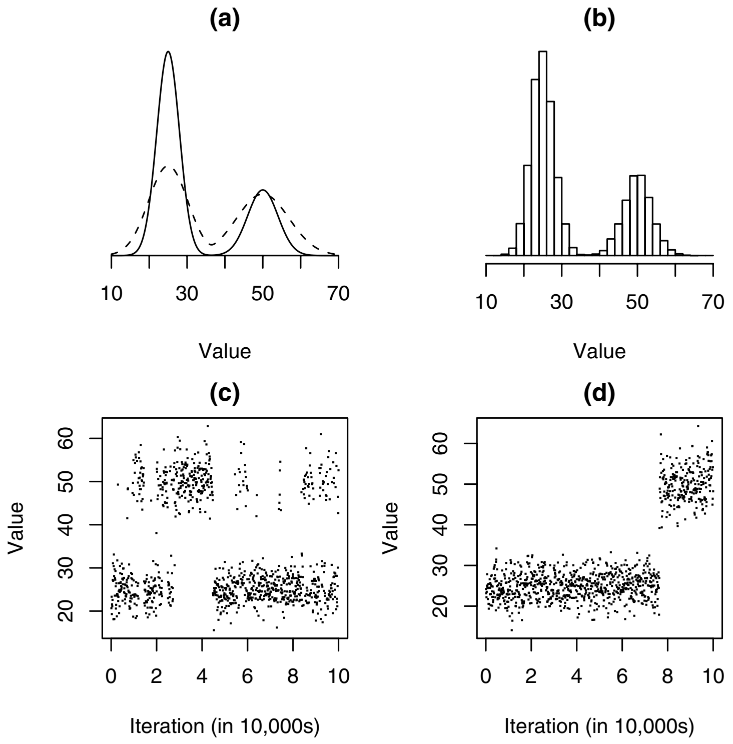

Fig. 3.3. Illustration of MCMCMC. (a) The solid line shows the target function, h. The dashed line is proportional to $h^{1/3}$ (rescaled to have a similar normalizing constant). A *heated* chain run under the dashed line will have the incorrect stationary distribution but will move more freely about the region. (b) Histogram of sampled points from the MCMCMC run. (c) Plot of the sampled points in the MCMCMC run versus iteration number. (d) Plot of the sampled points in a regular Metropolis-Hastings run versus iteration number.

3.7 Discussion

MCMC has become an indispensable tool for statistical computing, with special importance to the Bayesian approach. MCMC is especially useful for the high-dimensional calculation problems that arise in statistical models of molecular evolution. As evolutionary biologists address problems in molecular evolution of increasing complexity (larger trees, genome-scale data of varied type, more realistic and parameter-rich models of molecular evolution, accounting for additional forms of biological interaction), most of the tools that will be successful in providing answers to these questions are likely to be based on MCMC computation.

3.7.1 Other References about MCMC

MCMC is an important topic that is described in much greater detail in many other sources and is an area of much continuing active research. Gilks et al. have written an entire book on the topic of MCMC [10]. The books by Robert and Cassela and by Liu on Monte Carlo methods each include several chapters on MCMC [23, 19]. The books on Bayesian statistics by Gelman et al. and by Carlin and Louis include extensive descriptions of MCMC [7, 3]. Joe Felsenstein's recent book includes a chapter on Bayesian approaches to phylogenetic inference using MCMC as well as a chapter on using MCMC to make likelihood calculations on coalescent trees [6].

References

[1] J. P. Bielawski and Z. Yang. Maximum likelihood methods for detecting adaptive protein evolution. Chapter 5, this volume.

[2] J. P. Bollback. Posterior mappings and posterior predictive distributions. Chapter 16, this volume.

[3] B. P. Carlin and T. A. Louis. *Bayes and Empirical Bayes Methods for Data Analysis*. Chapman and Hall/CRC, Boca Raton, second edition, 2000.

[4] M. Dimmic. Markov models of protein sequence evolution. Chapter 9, this volume.

[5] R. Durrett. Genome rearrangement. Chapter 11, this volume.

[6] J. Felsenstein. *Inferring Phylogenies*. Sinauer Associates, Inc., Sunderland, MA, 2004.

[7] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, Boca Raton, 1995.

[8] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[9] C. J. Geyer. Markov chain Monte Carlo maximum likelihood. In E. M. Kerimidas, editor, *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, pages 156–163. Interface Foundation, Fairfax Station, VA, 1991.

[10] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors. *Markov Chain Monte Carlo in practice*. Chapman and Hall/CRC, Boca Raton, 1996.

[11] P. J. Green. Reversible jump MCMC computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[12] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[13] D. Hitchcock. A history of the Metropolis-Hastings algorithm. *The American Statistician*, 57:254–257, 2003.

[14] J. P. Huelsenbeck and F. Ronquist. Bayesian analysis of molecular evolution using MrBayes. Chapter 8, this volume.

[15] J. P. Huelsenbeck and F. Ronquist. MRBAYES: Bayesian inference of phylogenetic trees. *Bioinformatics*, 17:754–755, 2001.

[16] J. P. Huelsenbeck, F. Ronquist, R. Nielsen, and J. P. Bollback. Bayesian inference of phylogeny and its impact on evolutionary biology. *Science*, 294:2310–2314, 2001.

[17] B. Larget and D. L. Simon. Markov chain Monte Carlo algorithms for the Bayesian analysis of phylogenetic trees. *Molecular Biology and Evolution*, 16:750–759, 1999.

[18] S. Li, H. Doss, and D. Pearl. Phylogenetic tree reconstruction using Markov chain Monte Carlo. *Journal of the American Statistical Society*, 95:493–508, 2000.

[19] J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer, New York, 2001.

[20] B. Mau and M. A. Newton. Phylogenetic inference for binary data on dendograms using Markov chain Monte Carlo. *Journal of Computational and Graphical Statistics*, 6:122–131, 1997.

[21] B. Mau, M. A. Newton, and B. Larget. Bayesian phylogenetic inference via Markov chain Monte Carlo methods. *Biometrics*, 55:1–12, 1999.

[22] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[23] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, New York, 2002.

[24] D. Simon and B. Larget. Bayesian analysis in molecular biology and evolution (BAMBE). http://www.mathcs.duq.edu/larget/bambe.html, 2001.

[25] J. L. Thorne and H. Kishino. Estimation of divergence times from molecular sequence data. Chapter 9, this volume.

[26] L. Tierney. Markov chains for exploring posterior distributions (with discussion). *Annals of Statistics*, 22:1701–1762, 1994.

[27] Z. Yang and B. Rannala. Bayesian phylogenetic inference using DNA sequences: A Markov chain Monte Carlo method. *Molecular Biology and Evolution*, 14:717–724, 1997.