Appendix 3

Markov Chain Monte Carlo and Gibbs Sampling

Far better an approximate answer to the right question, which is often vague, than an exact answer to the wrong question, which can always be made precise – Tukey (1962)

Draft version 21 April 2014

A historical impediment to the more widespread use of Bayesian methods was computational — obtaining the posterior distribution usually requires the integration of high-dimensional functions. This can be very difficult, but several analytic approximations short of direct integration have been proposed (reviewed by Smith 1991; Evans and Swartz 1995; Tanner 1996). Now, however, the widespread development of **Markov Chain Monte Carlo** (**MCMC**) methods has made obtaining very complex posteriors relatively easy from a conceptual standpoint (although they may be computationally rather demanding). Indeed, MCMC allows Bayesian approaches to handle much higher dimensional problems than other methods. MCMC approaches are so-named because one uses the previous sample value to randomly generate the next sample value, generating a **Markov chain**.

The realization in the early 1990's (Gelfand and Smith 1990) that one particular MCMC method, the **Gibbs sampler**, is widely applicable to a broad class of Bayesian problems sparked much of the current expansion in the use of Bayesian analysis. MCMC methods have their roots in the **Metropolis algorithm** (Metropolis and Ulam 1949; Metropolis et al. 1953), an attempt by physicists to compute complex integrals by expressing them as expectations of some far simpler distribution and then estimate this expectation by drawing samples from that distribution. While the Gibbs sampler had its origins outside of statistics, it is still somewhat surprising that the powerful machinery of MCMC had essentially no impact on the field of statistics until rather recently. MCMC methods are reviewed by Tanner (1996), Gammerman (1997), Chen et al. (2000), and Robert and Casella (2004). The types of MCMC machinery used in quantitative genetics (often Gibbs samplers on mixed linear models, e.g., Chapter 19) tends to be a bit different from those used in molecular population genetics (ABC methods for treating complex likelihoods, e.g., Chapters 9–10). Sorensen and Gianola (2002) extensively review the former, while Marjoram and Tavaré (2006) provide a brief review of the latter. The very nice roundtable discussion on practical issues of MCMC by Kass et al. (1998) is required reading for anyone using MCMC analysis.

MONTE CARLO INTEGRATION

The original **Monte Carlo** approach used random number generation to approximate integrals. Suppose we wish to compute a complex integral

$$\int_a^b h(x)\,dx \tag{A3.1a}$$

If we can decompose $h(x)$ into the product of a function $f(x)$ and a probability density

function $p(x)$ defined over the interval (a, b), then

$$\int_a^b h(x)\,dx = \int_a^b f(x)\,p(x)\,dx = E_{p(x)}[\,f(x)\,], \tag{A3.1b}$$

so that the integral can be expressed as an expectation of $f(x)$ over the density $p(x)$. Sampling a large number $x_1, \cdots, x_n$ of random variables from the density $p(x)$,

$$\int_a^b h(x)\,dx = E_{p(x)}[\,f(x)\,] \simeq \frac{1}{n}\sum_{i=1}^{n} f(x_i) \tag{A3.1c}$$

This approach is referred to as **Monte Carlo integration**, and can be used to approximate posterior (or marginal posterior) distributions required for a Bayesian analysis. Consider the integral $I(y) = \int f(y \mid x)\,p(x)\,dx$, which we approximate by

$$\widehat{I}(y) = \frac{1}{n}\sum_{i=1}^{n} f(y \mid x_i) \tag{A3.1d}$$

where again x_i are draws from the density $p(x)$. Since we have framed this integral as an expectation, individual random draws have expected value equal to the integral. Hence, the spread of those values around their mean allows us to estimate a **Monte Carlo standard error** for our approximation,

$$\mathrm{SE}^2[\,\widehat{I}(y)\,] = \frac{1}{n}\left(\frac{1}{n-1}\sum_{i=1}^{n}\Big(f(y \mid x_i) - \widehat{I}(y)\Big)^2\right) \tag{A3.1e}$$

where the inner term estimates the sampling variance.

Example A3.1. An interesting quantitative-genetic application of Monte Carlo integration was suggested by Ovaskainen et al. (2008) for comparing whether two $\mathbf{G}$ matrices are similar. As we detail in Volume 3, there are a large number of proposed methods for comparing matrices, but Ovaskainen suggest a (conceptually) simple approach. The $\mathbf{G}$ matrix for a given population is really a description of the distribution of breeding values, and as such we can think of comparing $\mathbf{G}$ matrices as being akin to comparing two multivariate population distributions, whose densities are denoted f and g. If $\mathbf{x}$ denotes a draw from one of these distributions, the probability it originates from distribution g is just $g(\mathbf{x})/[g(\mathbf{x}) + f(\mathbf{x})]$. Hence, the probability q that a random draw from f is incorrectly assigned to g is just

$$q(f, g) = \int \frac{g(\mathbf{x})}{g(\mathbf{x}) + f(\mathbf{x})}\,f(\mathbf{x})\,d\mathbf{x} \tag{A3.2a}$$

If the two probability distributions are essentially indistinquishable, then $q(f, g) = 0.5$, while if they are completely distinquishable then $q(f, g) = 0$. Hence $1 - 2q(f, g)$, which ranges from zero (indistinquishable) to one (fully distinquishable), provides a simple metric of the difference in between then, and hence the difference between the two $\mathbf{G}$ matrices that comprise the distributions f versus g. Ovaskainen et al. modify this further, suggesting the metric

$$d(f, g) = \sqrt{1 - 2q(f, g)} \tag{A3.2b}$$

While Equation A3.2b is a complex integral, Equation A3.1d suggests

$$\widehat{q}\,(f,g) = \frac{1}{n}\sum_{i=1}^{n}\frac{g(\mathbf{x}_i)}{g(\mathbf{x}_i)+f(\mathbf{x}_i)} \to q(f,g) \tag{A3.2c}$$

where $\mathbf{x}_1, \cdots, \mathbf{x}_n$ are random draws from f. For example, if f is a multivariate normal with mean vector $\mathbf{0}$ and variance-covariance matrix $\mathbf{G}_1$, then

$$f(\mathbf{x}) = (2\pi)^{-n/2} \mid \mathbf{G}_1 \mid \exp\left(-\frac{\mathbf{x}^T\mathbf{G}_1^{-1}\mathbf{x}}{2}\right)$$

with g is similarly defined, but with $\mathbf{G}_2$ replacing $\mathbf{G}_1$, giving

$$\widehat{q}\,(f,g) = \frac{1}{n}\sum_{i=1}^{n}\frac{\mid \mathbf{G}_2 \mid \exp\left(-\mathbf{x}_i^T\mathbf{G}_2^{-1}\mathbf{x}_i/2\right)}{\mid \mathbf{G}_2 \mid \exp\left(-\mathbf{x}_i^T\mathbf{G}_2^{-1}\mathbf{x}_i/2\right) + \mid \mathbf{G}_1 \mid \exp\left(-\mathbf{x}_i^T\mathbf{G}_1^{-1}\mathbf{x}_i/2\right)} \tag{A3.2d}$$

The expression in the sum is not nearly as imposing as it appears. Both $\mid \mathbf{G}_1 \mid$ and $\mid \mathbf{G}_2 \mid$ are fixed constants, and only need to be computed once. Likewise, $\mathbf{G}_1^{-1}$ and $\mathbf{G}_2^{-1}$ need only be inverted once and then stored. Monte Carlo integration proceeds by generating random vectors $\mathbf{x}_1, \cdots \mathbf{x}_n$ from a MVN $\sim (\mathbf{0}, \mathbf{G}_1)$.

Importance Sampling

Consider any distribution whose probability density function $p(x)$ has the same support as some target density $q(x)$ (i.e., is nonzero over the same range of x values). Then

$$\int f(x)\,q(x)dx = \int f(x)\left(\frac{q(x)}{p(x)}\right)p(x)dx = E_{p(x)}\left[\,f(x)\left(\frac{q(x)}{p(x)}\right)\right] \tag{A3.3a}$$

This forms the basis for the method of **importance sampling**, with

$$\int f(x)\,q(x)dx \simeq \frac{1}{n}\sum_{i=1}^{n} f(x_i)\left(\frac{q(x_i)}{p(x_i)}\right) \tag{A3.3b}$$

where again the x_i are drawn from the distribution given by $p(x)$. For example, if we are interested in a marginal density as a function of y, $J(y) = \int f(y \mid x)\,q(x)dx$, we approximate this by

$$J(y) \simeq \frac{1}{n}\sum_{i=1}^{n} f(y \mid x_i)\left(\frac{q(x_i)}{p(x_i)}\right) \tag{A3.4}$$

where x_i are drawn from the approximating density p.

This method is so named in that a judicious choice of $p(x)$ ensures that values for which $f(x)$ is large (important) are appropriately sampled. Ideally, we would like $p(x)$ to be large when $f(x)q(x)$ is large (i.e., $p(x) \propto f(x)q(x)$), although assessing when $q(x)$ is large may be problematic. A second motivation for importance sampling is that an appropriate choice of p can result in a smaller Monte Carlo variance, and hence a more accurate estimate of the integral.

One issue that can arise when q is a posterior is that the integration constant is unknown, so that $q(x)$ as presented may not be a formal probability distribution, while $C \cdot \int q(x)$ is. Fortunately, a simple trick allows importance sampling to obtain C. Since $C\int q(x)dx = 1$,

$$C^{-1} = \int q(x)dx \simeq \frac{1}{n}\sum_{i=1}^{n} w_i, \quad \text{where} \quad w_i = \frac{g(x_i)}{p(x_i)} \tag{A3.5a}$$

with the x_i drawn from $p(x)$. Hence,

$$C \cdot \int f(x)\, q(x) dx \simeq \widehat{I} = \sum_{i=1}^{n} w_i f(x_i) \Big/ \sum_{i=1}^{n} w_i \tag{A3.5b}$$

which has an associated Monte Carlo variance of

$$\text{Var}\left(\widehat{I}\right) = \sum_{i=1}^{n} w_i \left(f(x_i) - \widehat{I}\right)^2 \Big/ \sum_{i=1}^{n} w_i \tag{A3.5c}$$

INTRODUCTION TO MARKOV CHAINS

Before introducing two of the most common MCMC methods, the Metropolis-Hastings algorithm and the Gibbs sampler, a few introductory comments on Markov chains are in order. Let X_t denote the value of a random variable at time t, and let the **state space** refer to the range of possible X values. The random variable is said to follow a **Markov process** if the transition probabilities between different values in the state space depend only on the random variable's current state, i.e.,

$$\Pr(X_{t+1} = s_j \,|\, X_0 = s_k, \cdots, X_t = s_i) = \Pr(X_{t+1} = s_j \,|\, X_t = s_i) \tag{A3.6}$$

For a Markov random variable, the only information about the past needed to predict the future is the current state of the random variable, as knowledge of the values of earlier states does not change the transition probability. A **Markov chain** refers to a sequence of random variables $(X_0, \cdots, X_n)$ generated by a Markov process. A particular chain is defined by its **transition probabilities** (or the **transition kernel**), $P(i,j) = P(i \to j)$, which is the probability that a process at state space s_i moves to state s_j in a single step,

$$P(i,j) = P(i \to j) = \Pr(X_{t+1} = s_j \,|\, X_t = s_i) \tag{A3.7a}$$

We use the notation $P(i \to j)$ to imply a move from i to j, as some texts define $P(i,j) = P(j \to i)$, so we will use the arrow notation to avoid any potential confusion. Let

$$\pi_j(t) = \Pr(X_t = s_j) \tag{A3.7b}$$

denote the probability that the chain is in state j at time t, and let $\boldsymbol{\pi}(t)$ denote the row vector of the state space probabilities at step/time t. We start the chain by specifying a starting vector $\boldsymbol{\pi}(0)$. Often all the elements of $\boldsymbol{\pi}(0)$ are zero except for a single element of 1, corresponding to the process starting in that particular state. As the chain progresses, the probability values become more dispersed over the state space. The probability that the chain has state value s_i at time/step $t+1$ is given by the **Chapman-Kolomogrov (CK) equation**, which sums over the probability of being in a particular state at the current step t and the transition probability from that state into state s_i,

$$\begin{aligned} \pi_i(t+1) &= \Pr(X_{t+1} = s_i) \\ &= \sum_k \Pr(X_{t+1} = s_i \,|\, X_t = s_k) \cdot \Pr(X_t = s_k) \\ &= \sum_k P(k \to i)\, \pi_k(t) = \sum_k P(k,i)\, \pi_k(t) \end{aligned} \tag{A3.7c}$$

Successive iteration of the CK equations describe the evolution of the chain.

We can more compactly write the CK equations in matrix form as follows. Define the **probability transition matrix P** as the matrix whose ij-th element is $P(i,j)$, the probability of moving from state i to state j, $P(i \to j)$. This implies that the rows of $\mathbf{P}$ sum to one, as considering the ith row, $\sum_j P(i,j) = \sum_j P(i \to j) = 1$. In matrix form, the Chapman-Kolomogrov equation becomes

$$\boldsymbol{\pi}(t+1) = \boldsymbol{\pi}(t)\mathbf{P} \tag{A3.8a}$$

Hence,

$$\boldsymbol{\pi}(t) = \boldsymbol{\pi}(t-1)\mathbf{P} = (\boldsymbol{\pi}(t-2)\mathbf{P})\mathbf{P} = \boldsymbol{\pi}(t-2)\mathbf{P}^2 \tag{A3.8b}$$

Continuing in this fashion yields the probability distribution in generation t as

$$\boldsymbol{\pi}(t) = \boldsymbol{\pi}(0)\mathbf{P}^t \tag{A3.8c}$$

Defining the n-step transition probability $p_{ij}^{(n)}$ as the probability that the process is in state j given that it started in state i n steps ago, i.e.,

$$p_{ij}^{(n)} = \Pr(X_{t+n} = s_j \,|\, X_t = s_i) \tag{A3.8d}$$

it immediately follows that $p_{ij}^{(n)}$ is just the ij-th element of $\mathbf{P}^n$, the nth power of the single-step transition matrix.

Finally, a Markov chain is said to be **irreducibile** if there exists a positive integer n_{ij} such that $p_{ij}^{(n_{ij})} > 0$ for all ij. That is, all states **communicate** with each other, in that the process can always move from any state to any other state (although this may require multiple steps). Likewise, a chain is said to be **aperiodic** when the number of steps required to move between two states (say x and y) is not required to be multiple of some integer. Put another way, the chain is not forced into cycles of fixed length between certain states.

Example A3.1. Suppose the state space consists of three possible weather conditions (Rain, Sunny, Cloudy) and that weather patterns follow a Markov process (of course, they do not!). Under this assumption, the probability of tomorrow's weather simply depends on today's weather, and not on any other previous days. If this is the case, the observation that it has rained for three straight days does not alter the probability of tomorrow's weather compared to the situation where (say) it rained today but was sunny for the last week. Suppose the probability transitions given today is rainy are

P(Rain tomorrow | Rain today) = 0.5,
P(Sunny tomorrow | Rain today) = 0.25,
P(Cloudy tomorrow | Rain today) = 0.25,

giving the first row of the transition probability matrix as $(0.5, 0.25, 0.25)$. Suppose the rest of the transition matrix is given by

$$\mathbf{P} = \begin{pmatrix} 0.5 & 0.25 & 0.25 \\ 0.5 & 0 & 0.5 \\ 0.25 & 0.25 & 0.5 \end{pmatrix}$$

Note that this Markov chain is irreducible, as all states communicate with each other. Suppose today is sunny. What is the expected weather two days from now? Seven days? Here $\boldsymbol{\pi}(0) = (\,0 \quad 1 \quad 0\,)$, giving

$$\boldsymbol{\pi}(2) = \boldsymbol{\pi}(0)\mathbf{P}^2 = (\,0.375 \quad 0.25 \quad 0.375\,)$$

and

$$\boldsymbol{\pi}(7) = \boldsymbol{\pi}(0)\mathbf{P}^7 = (\,0.4 \quad 0.2 \quad 0.4\,)$$

Conversely, suppose today is rainy, so that $\boldsymbol{\pi}(0) = (\,1 \quad 0 \quad 0\,)$. The expected weather becomes

$$\boldsymbol{\pi}(2) = (\,0.4375 \quad 0.1875 \quad 0.375\,) \quad \text{and} \quad \boldsymbol{\pi}(7) = (\,0.4 \quad 0.2 \quad 0.4\,)$$

Note that after a sufficient amount of time, the expected weather is *independent of the starting value*. In other words, the chain has reached a stationary distribution, where the state space probability values are independent of the actual starting value.

As the above example illustrates, a Markov chain may reach a **stationary distribution** $\boldsymbol{\pi}^*$, where the vector of probabilities of being in a particular state is independent of the initial starting distribution. The stationary distribution satisfies

$$\boldsymbol{\pi}^* = \boldsymbol{\pi}^*\mathbf{P} \tag{A3.9}$$

In other words, $\boldsymbol{\pi}^*$ is the left eigenvector associated with the eigenvalue $\lambda = 1$ of $\mathbf{P}$ (Appendix 5). Note that this also implies that the impact of the initial conditions after t steps decays as λ_2^t, where $\lambda_2 < 1$ is the second largest eigenvalue of $\mathbf{P}$. If this eigenvalue is very close to one, the impact of the initial conditions can persist for a substantial amount of time. Specifically, if $\lambda_2 = 1 - \delta$, then $\lambda_2^t = (1-\delta)^t \simeq \exp(-\delta t)$. One condition for the existance of a stationary distribution is that the chain is irreducible and aperiodic. When a chain is periodic, it can cycle in a deterministic fashion between states and never settle down to a stationary distribution (in effect, this cycling *is* the stationary distribution for this chain). A little thought will show that if $\mathbf{P}$ has no eigenvalues equal to -1, it is aperiodic.

A sufficient condition for a unique stationary distribution is that the **detailed balance equation** holds (for all i and j),

$$P(j \to i)\,\pi_j^* = P(i \to j)\,\pi_i^* \tag{A3.10}$$

That is, at equilibrium, the amount of probability flux from state j to stage i is exactly balanced by the probability flux on the opposite direction (for i to j), so that a balance is reached, with no *net* flow of probability over the states. If Equation A3.10 holds for all ik, the Markov chain is said to be **reversible**, and hence Equation A3.10 is also called the **reversibility condition**. Note that this condition implies $\boldsymbol{\pi}^* = \boldsymbol{\pi}^*\mathbf{P}$, as the jth element of $\boldsymbol{\pi}^*\mathbf{P}$ is

$$(\boldsymbol{\pi}^*\mathbf{P})_j = \sum_i \pi_i^*\,P(i \to j) = \sum_i \pi_j^*\,P(j \to i) = \pi_j^* \sum_i P(j \to i) = \pi_j^*$$

where the key middle step following from Equation A3.10, while the last step follows since rows of $\mathbf{P}$ sum to one.

The basic idea of discrete-state Markov chain can be generalized to a continuous state Markov process by having a probability kernel $P(x, y)$ that satisfies

$$\int P(x, y)\,dy = 1$$

and the continuous extension of the Chapman-Kolomogrov equation becomes

$$\pi_t(y) = \int \pi_{t-1}(x)P(x, y)\,dx \tag{A3.11a}$$

At equilibrium, the stationary distribution satisfies

$$\pi^*(y) = \int \pi^*(x)P(x,y)\,dx \tag{A3.11b}$$

THE METROPOLIS-HASTING ALGORITHM

The roots of MCMC methods trace back to a problem faced by mathematical physicists in applying Monte Carlo integration, namely generating random samples from some complex distribution in order to apply Equation A3.1. Their initial solution was the Metropolis-Hastings algorithm (Metropolis and Ulam 1949; Metropolis et al. 1953; Hastings 1970), reviewed by Chib and Greenberg (1995).

Suppose our goal is to draw samples from some distribution $p(\theta) = f(\theta)/K$, where the normalizing constant K may not be known, and very difficult to compute. The **Metropolis algorithm** (Metropolis and Ulam 1949; Metropolis et al. 1953), which generates a sequence of draws from this distribution, is as follows:

1. Start with any initial value θ_0 satisfying $f(\theta_0) > 0$.

2. Using current θ value, sample a **candidate point** θ^* from some **jumping distribution** $q(\theta_1, \theta_2) = \Pr(\theta_1 \rightarrow \theta_2)$, which is the probability of returning a value of θ_2 given a previous value of θ_1. This distribution is also referred to as the **proposal** or **candidate-generating distribution**. The only restriction on the jump density in the Metropolis algorithm is that it is symmetric, i.e., $q(\theta_1, \theta_2) = q(\theta_2, \theta_1)$.

3. Given the candidate point θ^*, calculate the ratio of the density at the candidate (θ^*) and current (θ_{t-1}) points,

$$\alpha = \frac{p(\theta^*)}{p(\theta_{t-1})} = \frac{f(\theta^*)}{f(\theta_{t-1})}$$

Notice that because we are considering the ratio of $p(x)$ under two different values, the normalizing constant K cancels out.

4. If the jump increases the density ($\alpha > 1$), accept the candidate point (set $\theta_t = \theta^*$) and return to step 2. If the jump decreases the density ($\alpha < 1$), then with probability α accept the candidate point, else reject it and return to step 2. Note that α is a ratio of two likelihoods, so the probability of accepting a move is the ratio of the candidate to current likelihoods.

We can summarize Metropolis sampling as first computing

$$\alpha = \min\left(\frac{f(\theta^*)}{f(\theta_{t-1})}, 1\right) \tag{A3.12}$$

and then accepting a candidate value θ^* with probability α (the **probability of a move**). This generates a Markov chain $(\theta_0, \theta_1, \cdots, \theta_k, \cdots)$, as the transition probabilities from θ_t to θ_{t+1} depends only on θ_t and not $(\theta_0, \cdots, \theta_{t-1})$. Following a sufficient **burn-in period** (of, say, k steps), the chain approaches its stationary distribution and (as we will demonstrate shortly), samples from the vector $(\theta_{k+1}, \cdots, \theta_{k+n})$ are samples from $p(x)$.

Hastings (1970) generalized the Metropolis algorithm by using an arbitrary (as opposed to strictly symmetric) transition probability function $q(\theta_1, \theta_2) = \Pr(\theta_1 \rightarrow \theta_2)$, and setting the acceptance probability for a candidate point as

$$\alpha = \min\left(\frac{f(\theta^*)\,q(\theta^*, \theta_{t-1})}{f(\theta_{t-1})\,q(\theta_{t-1}, \theta^*)}, 1\right) \tag{A3.13}$$

This is the **Metropolis-Hastings algorithm**, and Equation A3.13 is the **Hastings ratio**. Assuming the proposal distribution is symmetric, i.e., $q(x, y) = q(y, x)$, recovers the original Metropolis algorithm.

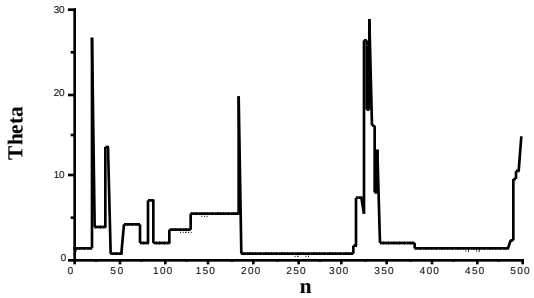
Example A3.2. Consider the scaled inverse-χ^2 distribution (Equation A2.28c),

$$p(x) = C \cdot x^{-(n/2+1)} \cdot \exp\left(\frac{-a}{2x}\right)$$

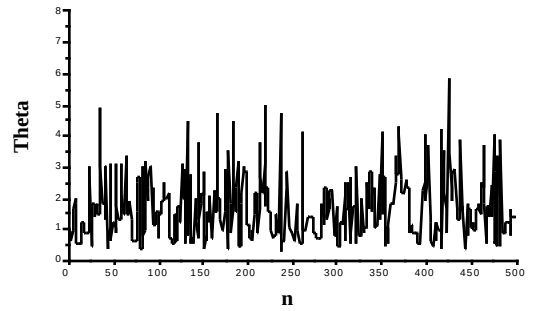
We wish to simulate draws from this distribution with $n = 3$ degrees of freedom and scaling factor $a = 4$, using the Metropolis algorithm. Here $f(x) = x^{-5/2} \exp[-4/(2x)]$. Suppose we take as our candidate-generating distribution a uniform over $(0, 100)$. Clearly, there is probability mass above 100 for the scaled inverse-χ^2, but we assume this is sufficiently small that we can ignore it. Now let's run the algorithm. Take $\theta_0 = 1$ as our starting value, and suppose the uniform returns a candidate value of $\theta^* = 39.82$. Computing α,

$$\alpha = \min\left(\frac{f(\theta^*)}{f(\theta_{t-1})}, 1\right) = \min\left(\frac{(39.82)^{-2.5} \cdot \exp(-2/39.82)}{(1)^{-2.5} \cdot \exp(-2/2 \cdot 1)}, 1\right) = 0.0007$$

Since $\alpha < 1$, θ^* is accepted with probability 0.007 — we randomly draw U from a uniform $(0, 1)$ and accept θ^* if $U \leq \alpha$. If $U > \alpha$, the candidate value is rejected, and we draw another candidate value from the proposal distribution and continue as above. A sample run resulting from first 500 values of θ is plotted below.



Notice that there are long flat periods (corresponding to all candidate values θ^* being rejected). Such a chain is called **poorly mixing**, and is numerically inefficient for sampling the entire probability spaced defined by the target distribution. Conversely, balancing the poor mixing is that when jumps occur, they tend to be large, exploring the extreme values of this distribution. In contrast, suppose our proposal distribution is a χ^2_1. As this distribution is not symmetric, we must employ Metropolis-Hastings (see Example A3.4 for the details). A resulting Metropolis-Hastings sampling run is shown below. Note that the time series looks like **white noise**, and the chain is said to be **well mixing**. However, note that most jumps are small (< 5), with fewer extreme values sampled than under the uniform.



Example A3.3. This somewhat technical example demonstrates that Metropolis-Hasting sampling generates a Markov chain whose equilibrium density is that candidate density $p(x)$. To prove this, it is sufficient to show that the Metropolis-Hasting transition kernel satisfies the detailed balance equation (A3.10), which our target distribution $p(x)$ can be stated as $\Pr(x \to y)p(x) = \Pr(y \to x)p(x)$. Under Metropolis-Hasting, we sample from $q(x, y) = \Pr(x \to y \,|\, q)$ and then accept the move with probability $\alpha(x, y)$, so that the transition probability kernel is given by

$$\Pr(x \to y) = q(x, y)\, \alpha(x, y) = q(x, y) \cdot \min \left[\frac{p(y)\, q(y, x)}{p(x)\, q(x, y)}, 1 \right] \tag{A3.14}$$

If the Metropolis-Hasting kernel satisfies $P(x \to y)\, p(x) = P(y \to x)\, p(y)$, or

$$q(x, y)\, \alpha(x, y)\, p(x) = q(y, x)\, \alpha(y, x)\, p(y) \quad \text{for all } x, y$$

then that stationary distribution from this kernel corresponds to draws from the target distribution $p(x)$. We show that the balance equation is indeed satisfied with this kernel by considering the three possible cases for any particular x, y pair.

1. $q(x, y)\, p(x) = q(y, x)\, p(y)$. Here $\alpha(x, y) = \alpha(y, x) = 1$ implying

$$P(x \to y)\, p(x) = q(x, y)\, p(x) \quad \text{and} \quad P(y \to x)p(y) = q(y, x)\, p(y)$$

and hence $P(x \to y)\, p(x) = P(y \to x)\, p(y)$, showing that (for this case), the detailed balance equation holds.

2. $q(x, y)\, p(x) > q(y, x)\, p(y)$, in which case

$$\alpha(x, y) = \frac{p(y)\, q(y, x)}{p(x)\, q(x, y)} \quad \text{and} \quad \alpha(y, x) = 1$$

Hence

$$\begin{aligned} P(x \to y)\, p(x) &= q(x, y)\, \alpha(x, y)\, p(x) \\ &= q(x, y)\, \frac{p(y)\, q(y, x)}{p(x)\, q(x, y)}\, p(x) \\ &= q(y, x)\, p(y) = q(y, x)\, \alpha(y, x)\, p(y) \\ &= P(y \to x)\, p(y) \end{aligned}$$

3. $q(x, y)\, p(x) < q(y, x)\, p(y)$. Here

$$\alpha(x, y) = 1 \quad \text{and} \quad \alpha(y, x) = \frac{q(x, y)\, p(x)}{q(y, x)\, p(y)}$$

Hence

$$\begin{aligned} P(y \to x)\, p(y) &= q(y, x)\, \alpha(y, x)\, p(y) \\ &= q(y, x) \left(\frac{q(x, y)\, p(x)}{q(y, x)\, p(y)} \right) p(y) \\ &= q(x, y)\, p(x) = q(x, y)\, \alpha(x, y)\, p(x) \\ &= P(x \to y)\, p(x) \end{aligned}$$

which concludes our proof.

Burning-in the Sampler

A key issue in the successful implementation of Metropolis-Hastings, or any other MCMC sampler, is the number of steps (iterations) until the chain approaches stationarity (the length of the **burn-in period**). Typically the first 1000 to 50,000 (or more) values of the chain are discarded, and then various convergence tests (see below) are used to assess whether stationarity has indeed been reached.

A poor choice of a proposal distribution and/or starting values can greatly increase the required burn-in time, and an area of current research is whether an optimal starting point and proposal distribution can be found. One suggestion is to initiate the chain as close to the center of the distribution as possible, for example taking a value close to the distribution's mode (such as using an approximate MLE as the starting value).

A chain is said to be **poorly mixing** if it says in small regions of the parameter space for long periods of time, as opposed to a **well mixing** chain that seems to happily explore the entire space (albeit perhaps by very small steps, requiring very many to survey the entire space). A poorly mixing chain can arise because the target distribution is multimodal and our choice of starting values traps us near one of the modes (such multimodal posteriors can arise if we have a strong prior in conflict with the observed data). Two approaches have been suggested for situations where the target distribution may have multiple peaks. The most straightforward is to use widely dispersed initial values to start several different chains (Gelman and Rubin 1992). A less obvious approach is to use **simulated annealing** on a single-chain.

Simulated Annealing

Simulated annealing was developed for finding the maximum of complex functions with multiple peaks where standard hill-climbing approaches may trap the algorithm on a less that optimal peak. The idea is that when we initially start sampling the space, we will accept a reasonable probability of a down-hill move in order to explore the entire space. As the process proceeds, we decrease the probability of such down-hill moves. The analogy (and hence the term) is the annealing of a crystal as temperate decreases — initially there is a lot of movement, which gets smaller and smaller as the temperature cools. Simulated annealing is very closely related to Metropolis sampling, differing only in that the probability α of a move is given by

$$\alpha_{SA} = \min\left[1, \left(\frac{p(\theta^*)}{p(\theta_{t-1})}\right)^{1/T(t)}\right] = \alpha^{1/T(t)} \tag{A3.15a}$$

where the function $T(t)$ is called the **cooling schedule** (setting $T = 1$ recovers Metropolis sampling), and the particular value of T at any point in the chain is called the **temperature**. For example, suppose that $p(\theta^*)/p(\theta_{t-1}) = 0.5$. With $T = 100$, $\alpha = 0.93$, while for $T = 1$, $\alpha = 0.5$, and for $T = 1/10$, $\alpha = 0.0098$. Hence, we start off with a high probability of a jump and then cool down to a very low jump probability. Note that replacing α with the Hastings ratio (Equation A3.13) shows that simulated annealing also applies to Metropolis-Hastings.

Typically, a function with geometric decline is used for $T(t)$. For example, to start at T_0 and "cool" down to a final "temperature" of T_f over n steps, we can set

$$T(t) = T_0\left(\frac{T_f}{T_0}\right)^{t/n} \tag{A3.15b}$$

More generally if we wish to cool off to T_f by time n, and then keep the temperature constant at T_f for the rest of the run, we can take

$$T(t) = \max\left(T_0\left(\frac{T_f}{T_0}\right)^{t/n}, T_f\right) \tag{A3.15c}$$

To cool down to Metropolis sampling (by step n), set $T_f = 1$ and the cooling schedule becomes

$$T(t) = \max\left(T_0^{1-t/n}, 1\right) \tag{A3.15c}$$

Choosing a Jumping (Proposal) Distribution

The Metropolis sampler works with any symmetric proposal (jumping) distribution, while Metropolis-Hastings allows more general distributions. So how do we determine our best option for a proposal distribution? There are two general approaches — random walks and independent chain sampling. Under a sampler using a proposal distribution based on a **random walk chain**, the new value y equals the current value x plus a random variable z,

$$y = x + z$$

In this case, $q(x, y) = g(y - x) = g(z)$, the density associated with the random variable z. If $g(z) = g(-z)$, i.e., the density for the random variable z is symmetric (as occurs with a normal or multivariate normal with mean zero, or a uniform centered around zero), then we can use Metropolis sampling as $q(x, y)/q(y, x) = g(z)/g(-z) = 1$. The variance of the proposal distribution can be thought of as a **tuning parameter** that is adjusted to get better mixing.

Under a proposal distribution using an **independent chain**, the probability of jumping to point y is independent of the current position (x) of the chain, i.e., $q(x, y) = g(y)$. The candidate value is simply drawn from a distribution of interest, independent of the current value. Again, any number of standard distributions can be used for $g(y)$. Note that in this case, the proposal distribution is generally not symmetric, as $g(x)$ is generally not equal to $g(y)$ (for $x \neq y$), and Metropolis-Hasting sampling must be used.

As mentioned, we can tune the proposal distribution to adjust the mixing, and in particular the acceptance probability, of the chain. This is generally done by adjusting the standard deviation (SD) of the proposal distribution. For example, by adjusting the variance (or the eigenvalues of the covariance matrix) for a normal (or multivariate normal), increasing or decreasing the range $(-a, a)$ if a uniform is used, or changing the degrees of freedom if a χ^2 is used (variance increasing with the df). To increase the acceptance probability, one *decreases* the proposal distribution SD. There is a tradeoff in that if the SD is too large, jumps are potentially large (which is good), but are usually not accepted (bad). This leads to high autocorrelation (see below) and very poor mixing, requiring much longer chains. If the proposal SD is too small, moves are generally accepted (high acceptance probability), but they are also small, again generating high autocorrelations and poor mixing.

Example A3.4. Suppose we wish to use a χ^2 distribution as our candidate density, by simply drawing from a χ^2 distribution independent of the current position. Recall from Equation A2.27b, that for $x \sim \chi^2_n$,

$$g(x) \propto x^{n/2-1}e^{-x/2}$$

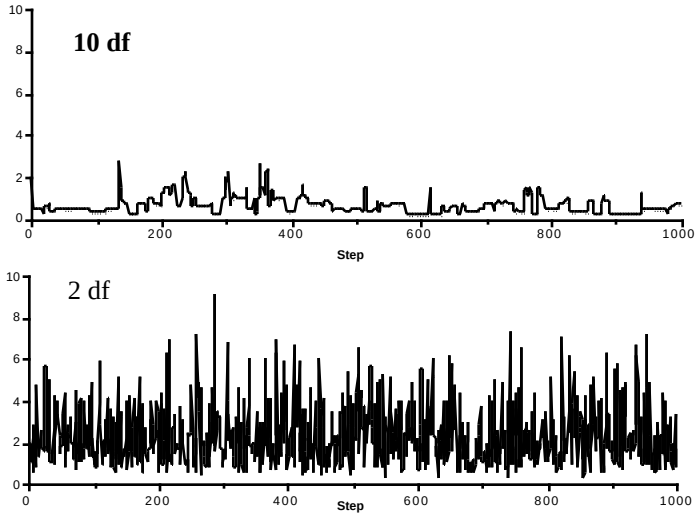
Thus, $q(x,y) = g(y) = C \cdot y^{n/2-1}e^{-y/2}$. Note that $q(x,y)$ is not symmetric, as $q(y,x) = g(x) \neq g(y) = q(x,y)$. Hence, we must use Metropolis-Hastings sampling, with acceptance probability

$$\alpha(x,y) = \min\left[\frac{p(y)\,q(y,x)}{p(x)\,q(x,y)}, 1\right] = \min\left[\frac{p(y)\,g(x)}{p(x)\,g(y)}, 1\right] = \min\left[\frac{p(y)\,x^{n/2-1}e^{-x/2}}{p(x)\,y^{n/2-1}e^{-y/2}}, 1\right]$$

Using the same target distribution as in Example A3.2 (a scaled inverse χ^2), $p(x) = C \cdot x^{-2.5}\,e^{-2/x}$, the rejection probability becomes

$$\alpha(x,y) = \min\left[\frac{(\,y^{-2.5}\,e^{-2/y}\,)\,(\,x^{n/2-1}e^{-x/2}\,)}{(\,x^{-2.5}\,e^{-2/x}\,)\,(\,y^{n/2-1}e^{-y/2}\,)}, 1\right]$$

Results for a run of the sampler under two different proposal distributions (a χ^2_2 and a χ^2_{10}) are plotted below. The χ^2_2 has the smaller variance, and thus a higher acceptance probability. Further, it seems to explore more of the distribution space relative to a χ^2_{10} (compare the number of values above 4 in both traces).



Autocorrelation and Sample Size Inflation

We expect adjacent members from an MCMC sequence to be positively correlated, and we can quantify the nature of this correlation by using an **autocorrelation function**. Consider a sequence $(\theta_1, \cdots, \theta_n)$ of length n. Correlations can occur between adjacent members ($\rho(\theta_i, \theta_{i+1}) \neq 0$), and (more generally) between more distant members ($\rho(\theta_i, \theta_{i+k}) \neq 0$). The kth order autocorrelation ρ_k can be estimated by

$$\widehat{\rho}_k = \frac{\text{Cov}(\theta_t, \theta_{t+k})}{\text{Var}(\theta_t)} = \frac{\displaystyle\sum_{t=1}^{n-k}\left(\theta_t - \overline{\theta}\right)\left(\theta_{t+k} - \overline{\theta}\right)}{\displaystyle\sum_{t=1}^{n-k}\left(\theta_t - \overline{\theta}\right)^2}, \quad \text{with} \quad \overline{\theta} = \frac{1}{n}\sum_{t=1}^{n}\theta_t \tag{A3.16}$$

An important result from the theory of time series analysis is that if the θ_t are from a stationary (and correlated) process, correlated draws still provide an unbiased picture of the distribution ***provided the sample size is sufficiently large***.

Some indication of the required sample size comes from the theory of a **first-order autoregressive process** (or AR_1), where

$$\theta_t = \mu + \alpha(\,\theta_{t-1} - \mu\,) + \epsilon \tag{A3.17a}$$

where $|\,\alpha\,| < 1$ and ϵ is **white noise**, $\epsilon \sim N(0, \sigma^2)$. Here $\rho_1 = \alpha$ and the kth order autocorrelation is given by $\rho_k = \rho_1^k = \alpha^k$. Under this process, $E(\,\overline{\theta}\,) = \mu$ with standard error

$$\text{SE}\,(\,\overline{\theta}\,) = \frac{\sigma}{\sqrt{n}}\sqrt{\frac{1+\rho}{1-\rho}} \tag{A3.17b}$$

The first ratio is the standard error for white noise, while the second ratio, $\sqrt{(1+\rho)/(1-\rho)}$, is the **sample size inflation factor**, or **SSIF**, which shows how the autocorrelation inflates the sampling variance relative to independent samples. For example, for $\rho = 0.5$, 0.75, 0.9, 0.95, and 0.99, the associated SSIF are 3, 7, 19, 39, and 199 (respectively). With an autocorrelation of 0.95 (which is not uncommon in a Metropolis-Hastings sequence), roughly forty times as many iterations are required for the same precision as with an uncorrelated sequence.

One historical strategy for reducing autocorrelation is **thinning** (or **subsampling**) the output, storing only every m-th point after the burn-in period. Suppose a Metropolis-Hastings sequence follows an AR_1 model with $\rho_1 = 0.99$. In this case, sampling every 50, 100, and 500 points gives the correlation between the thinned samples as $0.605\,(= 0.99^{50})$, 0.366, and 0.007 (respectively), for SSIF factors of 2, 1.5, and 1.007. In addition to reducing autocorrelation, thinning the sequence also saves computer memory. However, this throws out a huge amount of the data, and Geyer (1992) and MacEachern and Berliner (1994) found that it is more efficient (yielding smaller Monte Carlo variances) to keep the entire (burned-in) chain than to use thinning.

The Monte Carlo Variance of a MCMC Based Estimate

Suppose we are interested in using a burned-in MCMC sequence $\theta_1, \cdots, \theta_n$ to estimate some function $h(\theta)$ of the target distribution, such as a mean, variance, or specific quantile (cumulative probability value). Since we are drawing random variables, there is sampling variance associated with the Monte Carlo estimate

$$\widehat{h} = \frac{1}{n}\sum_{i=1}^{n} h(\theta_i) \tag{A3.18}$$

A similar issue arose with the error in Monte Carlo integration, where we used the sample variance about $\widehat{h}$ to compute the Monte Carlo variance, dividing this by the number of iterations to yield a standard error (Equations A3.1e, A3.5c). This strategy exploited the fact that draws for Monte Carlo integration are *independent* and therefore uncorrelated. By stark contrast, we expect draws in an MCMC sequence to be highly *dependent*. A standard sample variance estimator ignores these correlations, yielding a highly biased estimate of the Monte Carlo variance (cf., Equation A3.17b).

One direct approach is to run several chains (or extract subsamples from a very long chain) and use the between-chain variance in $\widehat{h}$. Specifically, if $\widehat{h}_j$ denotes the estimate for chain j ($1 \le j \le c$) where each of the c chains has the same length, then the estimated variance of the Monte Carlo estimate is

$$\text{Var}\left(\widehat{h}\right) = \frac{1}{c-1}\sum_{j=1}^{c}\left(\widehat{h}_j - \widehat{h^*}\right)^2 \quad \text{where} \quad \widehat{h^*} = \frac{1}{c}\sum_{j=1}^{c}\widehat{h}_j \tag{A3.19}$$

Using only a single chain, an alternative approach is to use results from the theory of time series to account for the correlations among members in the chain. Estimate the lag-k autocovariance associated with h by

$$\widehat{\gamma}(k) = \frac{1}{n}\sum_{i=1}^{n-k}\left[\left(h(\theta_i) - \widehat{h}\right)\left(h(\theta_{i+k}) - \widehat{h}\right)\right] \tag{A3.20}$$

This is natural generalization of the k-th order autocorrelation (Equation A3.16) to the random variable generated by $h(\theta_i)$. Geyer (1992) gives the resulting estimate of the Monte Carlo variance as

$$\text{Var}\left(\widehat{h}\right) = \frac{1}{n}\left(\widehat{\gamma}(0) + 2\sum_{i=1}^{2\delta+1}\widehat{\gamma}(i)\right) \tag{A3.21}$$

Here δ is the largest positive integer satisfying $\widehat{\gamma}(2\delta) + \widehat{\gamma}(2\delta+1) > 0$, (i.e., the higher order (lag) autocovariances above δ are zero).

One measure of the effects of autocorrelation between elements in the sampler is the **effective chain size**,

$$\widehat{n} = \frac{\widehat{\gamma}(0)}{\text{Var}\left(\widehat{h}\right)} \tag{A3.22}$$

where $\widehat{\gamma}(0)$ is the standard sample variance. In the absence of autocorrelation between members, $\widehat{n} = n$.

CONVERGENCE DIAGONISTICS

The careful reader will note that we have still not answered the vexing question of how to determine whether the sampler has reached its stationary distribution. Just what formal tests are available to test for stationarity of the sampler? While many have been proposed, it is important to stress that while these can indeed show that stationarity has not yet been reached, negative results are ***not conclusive***. A few comments from recent reviews stress this point: "all of the methods can fail to detect the sorts of convergence failure that they were designed to identify" (Cowles and Carlin 1996), "It is clear that the usefulness of convergence assessment methods are limited by their potential unreliability" (Brooks and Roberts 1998), and "no method works in every case "(El Adlouni et al. 2006). Despite this very serious concern, a variety of methods can certainly show failure of convergence given the burn-in period used for a sampler. The greater issue of trust is what do we say when convergence methods *do not* indicate a problem. We consider a few basic approaches here, while additional diagnostics are reviewed by Geyer (1992), Gelman and Rubin (1992). Raftery and Lewis (1992b), Cowles and Carlin (1996), Brooks and Roberts (1998), Kass et al. (1998), Mengerson et al. (1999), Robert and Casella (2004), El Adlouni et al. (2006), and Peltonen et al. (2009).

Visual Analysis

As shown in Examples A3.2 and A3.4, one should always examine the **time series trace**, the plot of the random variable(s) being generated versus the number of iterations. In addition to showing evidence for poor mixing, such traces can also suggest a ***minimum*** burn-in period for some starting value. For example, suppose the trace moves very slowly away from the initial value to a rather different value (say after 5000 iterations) after which it appears to settle down. Clearly, the burn-in period is ***at least*** 5000 in this case. It must be cautioned that the actual burn-in time ***may be far longer*** than suggested by the trace. Nevertheless, the trace often indicates that more brun-on time is needed. A second approach is to run either several chains, or subsample from a much longer chain, and compare traces. The expectation

under stationarity is that these samples are drawn from the same distribution. Even if visual analysis suggests that two (or more) chains appear to be very similar and both well-behaved, the concern is that both might be stuck in the same local maximum region of a more complex posterior. In such cases, the impression of stationarity and consistency is presented, but significantly longer runs for each chain would show a different outcome.

Correlograms, which plot serial autocorrelations as a function of the time lag, are also very useful in accessing a run from an MCMC sampler. A plot of ρ_k vs. k (the kth order autocorrelation vs. the lag) should show geometric decay if the sampler series closely follows an AR_1 model. A plot of the **partial autocorrelations** as a function of lag is also useful. The kth partial autocorrelation is the excess correlation not accounted for by a $k-1$ order autogressive model (AR_{k-1}). Hence, if the first order model fits, the second order partial autocorrelation is zero, as the lagged autocorrelations are completed accounted for the AR_1 model (i.e., $\rho_k = \rho_1^k$). Both of these autocorrelation plots may indicate underlying correlation structure in the series not obvious from the time series trace.

More Formal Approaches

Gelman and Rubin's (1992) proposed to start m separate chains at initial locations widely dispersed in the parameter space, which each returning a burned-in and thinned (so that autocorrelations between estimates are near zero) sequence of length n. Their approach is essentially an ANOVA, contrasting the estimated between-chain variance with its within-chain value. For the univariate case, let

$$B = \frac{n}{m-1}\sum_{i=1}^{m}(\overline{\theta}_{i\cdot} - \overline{\theta}_{\cdot\cdot})^2 \tag{A3.23a}$$

be the between-chain variance, where $\overline{\theta}_{i\cdot}$ is the mean parameter value for chain i, and $\overline{\theta}_{\cdot\cdot}$ the grand mean over all chains. Likewise, for within-chain variance define

$$W = \frac{1}{m}\left(\frac{1}{n-1}\sum_{i=1}^{m}\sum_{j=1}^{n}(\overline{\theta}_{ij} - \overline{\theta}_{i\cdot})^2\right) \tag{A3.23b}$$

A combined estimate of the variance is

$$\widehat{\sigma^2} = \frac{n-1}{n}W + \frac{1}{n}B \tag{A3.23c}$$

Gelman and Rubin define the **potential scale reduction factor** (or **PSRF**) as

$$R = \frac{\widehat{\sigma^2}}{W} = 1 + \frac{1}{n}\left(\frac{B}{W} - 1\right) \tag{A3.23d}$$

with values of R near one being consistent with convergence. A plot of R over time should converge to a value near one if the chains are converging.

Since most MCMC involve many variables, one could either look at the univariate R trace for each or use the multivariate version given by Brooks and Gelman (1998),

$$R = \frac{n-1}{n} + \left(1 + \frac{1}{m}\right)\lambda_1 \tag{A3.24}$$

where λ_1 is the leading eigenvalue of $\mathbf{W}^{-1}\mathbf{B}/n$, and $\mathbf{W}$ and $\mathbf{B}$ the sample within- and between-covariance matrices for the parameters. Again a value near 1 is consistent with convergence.

The **Geweke test** (Geweke 1992) splits the sample (after removing a burn-in period) into two parts: say the first 10% and last 50%. If the chain is at stationarity, the means of the two samples should be equal. A modified z-test can be used to compare the two subsamples, and the resulting test statistic is often referred to as a **Geweke z-score**. A value larger than 2 indicates that the mean of the series is still drifting, and a longer burn-in is required before monitoring the chain (to extract a sampler) can begin.

A more informative approach is the **Raftery-Lewis test** (Raftery and Lewis 1992a). Here, one specifies a particular quantile q of the distribution of interest (typically 2.5% and 97.5%, to give a 95% confidence interval), an accuracy ϵ of the quantile, and a power $1-\beta$ for achieving this accuracy on the specified quantile. With these three parameters set, the Raftery-Lewis test breaks the chain into a (1,0) sequence — 1 if $\theta_t \leq q$, zero otherwise. This generates a two-state Markov chain, and the Raftery-Lewis test uses the sequence to estimate the transition probabilities between states. With these probabilities in hand, one can then estimate the number of addition burn-ins (if any) required to approach stationarity, the thinning ratio (how many points should be discarded for each sampled point), and the total chain length required to achieve the preset level of accuracy.

One Long Chain or Many Smaller Chains?

One can either use a single long chain (Geyer 1992; Raftery and Lewis 1992b) or multiple chains each starting from different initial values (Gelman and Rubin 1992). Note that with parallel processing, using multiple chains may be computationally more efficient than a single long chain. Geyer, however, argues that using a single longer chain is the best approach. If long burn-in periods are required, or if the chains have very high autocorrelations, using a number of smaller chains may result in each not being long enough to be of any value. One current practice (Kass et al. 1998) is to use five parallel chains with widely-separated starting values, starting the jth for parameter θ_i at $\mu_i + (j-3)\sigma_i$, where μ_i and σ_i^2 are the prior values for θ_i.

THE GIBBS SAMPLER

The **Gibbs sampler** (introduced in the context of image processing by Geman and Geman 1984), is a special case of Metropolis-Hastings sampling wherein the random value is always accepted (i.e., $\alpha = 1$). The key to the Gibbs sampler is that one only considers ***univariate*** conditional distributions — the distribution when all of the random variables but one are assigned fixed values. Such conditional distributions are far easier to simulate than complex joint distributions and usually have simple forms (often being normals, inverse χ^2, or other common prior distributions). Thus, one simulates n random variables sequentially from the n univariate conditionals rather than generating a single n-dimensional vector in a single pass using the full joint distribution.

To introduce the Gibbs sampler, consider a bivariate random variable (x, y), and suppose we wish to compute one or both marginals, $p(x)$ and $p(y)$. The idea behind the sampler is that it is far easier to consider a sequence of conditional distributions, $p(x \mid y)$ and $p(y \mid x)$, than it is to obtain the marginal by integration of the joint density $p(x, y)$, e.g., $p(x) = \int p(x, y)dy$. The sampler starts with some initial value y_0 for y and obtains x_0 by generating a random variable from the conditional distribution $p(x \mid y = y_0)$. The sampler then uses x_0 to generate a new value of y_1, drawing from the conditional distribution based on the value x_0, $p(y \mid x = x_0)$. The sampler proceeds as follows

$$x_i \sim p(x \mid y = y_{i-1}) \tag{A3.25a}$$

$$y_i \sim p(y \mid x = x_i) \tag{A3.25b}$$

Repeating this process k times, generates a **Gibbs sequence** of length k, where a subset of points (x_j, y_j) for $1 \leq j \leq m < k$ are taken as our simulated draws from the full joint distribution. One iteration of all the univariate distributions is often called a **scan** of the sampler. To obtain the desired total of m sample points (here each element in the sampler is a vector of realizations of the two random variables), one samples the chain (i) after a sufficient burn-in to removal the effects of the initial starting values and (ii) at set time points (say every τ samples) following the burn-in. The Gibbs sequence converges to a stationary (equilibrium) distribution that is independent of the starting values, and by construction this stationary distribution is the target distribution we are trying to simulate (Tierney 1994). Of course, a guarantee of convergence at some limit is no guarantee that our sample size is sufficiently large to justify the assumption of stationarity for a burned-in and trimmed sampler.

Example A3.5. Consider the following distribution from Casella and George (1992). Suppose the joint distribution of $x = 0, 1, \cdots n$ and $0 \leq y \leq 1$ is given by

$$p(x, y) = \frac{n!}{(n-x)!x!}\, y^{x+\alpha-1}\,(1-y)^{n-x+\beta-1}$$

Note that x is discrete and y continuous. While the joint density is complex, the conditional densities are simple distributions. To see this, first recall that a binomial random variable z has a density proportional to

$$p(z \,|\, q, n) \propto \frac{q^z(1-q)^{n-z}}{z!(n-z)!} \quad \text{for} \quad 0 \leq z \leq n$$

where $0 < q < 1$ is the success parameter and n the number of traits, and we denote $z \sim \text{B}(n, p)$. Likewise recall the density for $z \sim \text{Beta}(a, b)$, a beta distribution with shape parameters a and b is given by

$$p(z \,|\, a, b) \propto z^{a-1}(1-z)^{b-1} \quad \text{for} \quad 0 \leq z \leq 1$$

Observe that the conditional distribution of x (treating y as a fixed constant) is $x \,|\, y \sim \text{B}(n, y)$, while $y \,|\, x \sim \text{Beta}(x + \alpha, n - x + \beta)$.

The power of the Gibbs sampler is that by computing a sequence of these univariate conditional random variables (a binomial and then a beta) we can compute any feature of either marginal distribution. Suppose $n = 10$ and $\alpha = 1$, $\beta = 2$. Start the sampler with (say) $y_0 = 1/2$, and take the sampler through three full iterations.

(i) x_0 is obtained by generating a random $\text{B}(n, y_0) = \text{B}(10, 1/2)$ random variable, giving $x_0 = 5$ in our simulation.

(ii) y_1 is obtained from a $\text{Beta}(x_0+\alpha, n-x_0+\beta) = \text{Beta}(5+1, 10-5+2)$ random variable, giving $y_1 = 0.33$.

(iii) x_1 is a realization of a $\text{B}(n, y_1) = \text{B}(10, 0.33)$ random variable, giving $x_1 = 3$.

(iv) y_2 is obtained from a $\text{Beta}(x_1+\alpha, n-x_1+\beta) = \text{Beta}(3+1, 10-3+2)$ random variable, giving $y_2 = 0.56$.

(v) x_2 is obtained from a $\text{B}(n, y_2) = \text{B}(10, 0.56)$ random variable, giving $x_2 = 0.7$.

Our particular realization of the Gibbs sequence after three iterations is thus (5, 0.5), (3, 0.33), (7, 0.56). We can continue this process to generate a chain of the desired length. Obviously, the initial values in the chain are dependent upon the y_0 value chosen to start the chain. This dependence decays as the sequence length increases and so we typically start recording the sequence only after a sufficient number of burn-in iterations have occurred.

When more than two variables are involved, the sampler is extended in the obvious fashion. In particular, the value of the kth variable is drawn from the distribution $p(\theta^{(k)} \mid \boldsymbol{\Theta}^{(-k)})$ where $\boldsymbol{\Theta}^{(-k)}$ denotes a vector containing all of the variables but k. Thus, during the ith iteration of the sample, to obtain the value of $\theta_i^{(k)}$ we draw from the distribution

$$\theta_i^{(k)} \sim p(\theta^{(k)} \mid \theta^{(1)} = \theta_i^{(1)}, \cdots, \theta^{(k-1)} = \theta_i^{(k-1)}, \theta^{(k+1)} = \theta_{i-1}^{(k+1)}, \cdots, \theta^{(n)} = \theta_{i-1}^{(n)})$$

For example, if there are four variables, (w, x, y, z), the sampler becomes

$$\begin{aligned} w_i &\sim p(w \mid x = x_{i-1}, y = y_{i-1}, z = z_{i-1}) \\ x_i &\sim p(x \mid w = w_i, y = y_{i-1}, z = z_{i-1}) \\ y_i &\sim p(y \mid w = w_i, x = x_i, z = z_{i-1}) \\ z_i &\sim p(z \mid w = w_i, x = x_i, y = y_i) \end{aligned}$$

Gelfand and Smith (1990) illustrated the power of the Gibbs sampler to address a wide variety of statistical issues, while Smith and Roberts (1993) showed the natural marriage of the Gibbs sampler with Bayesian statistics for obtaining posterior distributions. A nice introduction to the sampler is given by Casella and George (1992), while further details can be found in Tanner (1996), Besag et al. (1995), Sorensen and Gianola (2002), Robert and Casella (2004), and Lee (2013). Note that the Gibbs sampler can be thought of as a stochastic analog to the EM (Expectation-Maximization) approaches (LW Appendix 4) used to obtain likelihood functions when missing data are present. In the sampler, random sampling replaces the expectation and maximization steps.

Using the Gibbs Sampler to Approximate Marginal Distributions

Any feature of interest for the marginals can be computed from the m realizations of the Gibbs sequence. If $\theta_1, \cdots \theta_m$ is an appropriately burned-in set of realizations from a Gibbs sampler, the expectation of any function f of the random variable θ is approximated by

$$E[f(\theta)]_m = \frac{1}{m} \sum_{i=1}^{m} f(\theta_i) \tag{A3.26a}$$

This is the **Monte-Carlo** (MC) **estimate** of $f(x)$, as $E[f(\theta)]_m \to E[f(\theta)]$ as $m \to \infty$. Likewise, the MC estimate for any function of n variables $(\theta^{(1)}, \cdots, \theta^{(n)})$ is given by

$$E[f(\theta^{(1)}, \cdots, \theta^{(n)})]_m = \frac{1}{m} \sum_{i=1}^{m} f(\theta_i^{(1)}, \cdots, \theta_i^{(n)}) \tag{A3.26b}$$

Example A3.6. Although the sequence of length 3 computed in Example A3.5 is too short (and too dependent on the starting value) to be a proper Gibbs sequence, for illustrative

purposes we can use it to compute Monte-Carlo estimates. The MC estimate of the means of x and y are

$$\overline{x}_3 = \frac{5+3+7}{3} = 5, \quad \overline{y}_3 = \frac{0.5+0.33+0.56}{3} = 0.46$$

Similarly, $\left(\overline{x^2}\right)_3 = 27.67$, $\left(\overline{y^2}\right)_3 = 0.22$, and $(\,\overline{xy}\,)_3 = 2.47$, giving the MC estimates of the variances of x and y as

$$\text{Var}(x)_3 = \left(\overline{x^2}\right)_3 - (\overline{x}_3)^2 = 2.67, \qquad \text{Var}(y)_3 = \left(\overline{y^2}\right)_3 - (\overline{y}_3)^2 = 0.25$$

and their covariance as

$$\text{Cov}(x,y)_3 = (\,\overline{xy}\,)_3 - \overline{x}_3 \cdot \overline{y}_3 = 2.47 - 5 \cdot 0.46 = 0.16$$

While computing the MC estimate of any moment using the sampler is straightforward, computing the actual *shape* of the marginal density is slightly more involved. While one might use the empirical histogram of the Gibbs sequence as a rough approximation of the marginal distribution of x, this turns out to be inefficient, especially for obtaining the tails of the distribution. A better approach is to use the average of the conditional densities $p(x \,|\, y = y_i)$, as the function form of the conditional density contains more information about the shape of the entire distribution than the sequence of individual realizations x_i (Gelfand and Smith 1990; Liu et al. 1991). Since

$$p(x) = \int p(x \,|\, y)\, p(y)\, dy = E_y\left[\, p(x \,|\, y)\,\right] \tag{A3.27a}$$

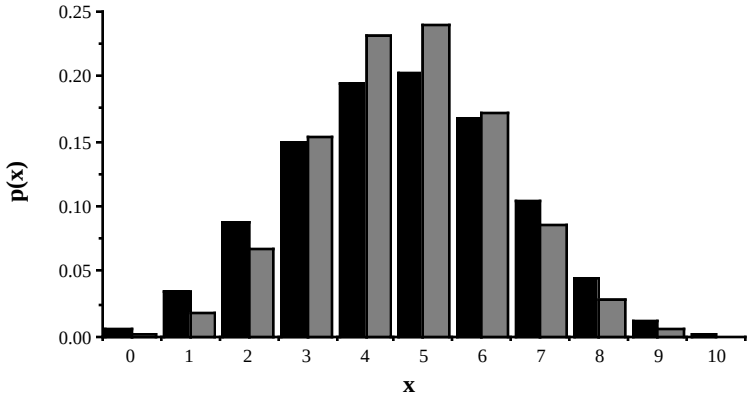
one can approximate the marginal density using

$$\widehat{p}_m(x) = \frac{1}{m} \sum_{i=1}^{m} p(x \,|\, y = y_i) \tag{A3.27b}$$

Example A3.7. Returning to the Gibbs sequence generated in Example A3.5, recall that the distribution of x given y is binomial, with $x \,|\, y \sim \text{B}(n, y)$. Applying Equation A3.27b the estimate (based on this sequence) of the marginal distribution of x is the weighted sum of three binomials with success parameters 0.5, 0.33, and 0.56, giving

$$p_3(x) = \frac{10!}{x!(10-x)!}\left[\frac{0.5^x(1-0.5)^{10-x} + 0.33^x(1-0.33)^{10-x} + 0.56^x(1-0.56)^{10-x}}{3}\right]$$

As the figure below shows, the resulting distribution (solid bars), although a weighted sum of binomials, departs substantially from the binomial based on the average (here 0.46) of the success parameter (stripped bars).



REJECTION SAMPLING AND APPROXIMATE BAYESIAN COMPUTATION (ABC)

In the words of Marjoram and Tavaré (2006), for many estimation problems in modern population genetics it is "far easier to simulate than to calculate". Consider the number of segregating sites in a sample of alleles from a population that went through a bottleneck of undetermined width at some unknown time in the past. A likelihood calculation running over all possible coalescent structures even in this simple case is extremely challenging. Conversely, if we specify the fractional reduction of the bottleneck and the time this occurred, it is very easy to simulate a coalescent from this population structure and them randomly apply mutations to generate a sample of alleles. This logic forms the basis an approach that Beaumont et al. (2002) called **approximate Bayesian computation** (**ABC**), although its roots trace back to Tavaré et al. (1997), Weiss and von Haeseler (1998), and Pritchard et al. (1999). Reviews of ABC are given by Marjoram et al. (2003), Beaumont (2010), and Csilléry et al. (2010). The use of ABC methods is a rapid growth area and, as such, our treatment is very basic and likely dated.

A very simple example motivates ABC. Suppose we wish to calculate the posterior distribution for the success probability p given that a sample of size 10 has three successes and a prior $\phi(p)$. We can obtain this exactly without ever computing the likelihood as follows. First, draw a random p value from the prior $\phi(p)$, and then generate a random draw from a binomial with $n = 10$ and p. If this value equals three, we record the p value (accept it into the posterior). If not, randomly draw a new p and then simulate using this p, repeating this procedure until we have a sufficiently large number of accepted values to approximate the posterior. This is an example of **rejection sampling**.

More generally, draw a candidate vector $\boldsymbol{\Theta}$ of model parameters from the prior, and then use these in a simulation to generate a set D of data. For our opening example, one would draw values of bottleneck width, time of the bottle neck, and scaled mutation rate from a joint prior, and then use this parameter set to generate D (the observed polymorphism information on our sample). If $D = D^*$ (the original data), then we accept this vector and draw again. If not, we reject it and draw again. This generates a value in the MCMC sample without ever having to compute a likelihood.

The complication with this basic idea obviously arises as data sets get even slightly complex, as the probability of $D = D^*$ becomes vanishingly small and the rejection probability is essentially one. In this case, one instead tries to match *summary statistics* from D and D^*. Ideally, these would be **sufficient statistics** for the parameters of interest. Recall that, condi-

tioned on the value of a sufficient statistic, the distribution of the data does not depend on the parameter of interest —- all of the information on that parameter in the data is provided by the sufficient statistic. Suppose $\mathbf{S}$ is a vector of sufficient statistics for the parameters of interest from the simulated data set D, with $\mathbf{S}^*$ the same from D^*. A strict rejection method would accept the current draw from the prior if $\mathbf{S}$ equals $\mathbf{S}^*$. Here, again, rejection probability is likely quite high. The approximate aspect of this approach arises when we don't require exact equality, rather that the two are "close", i.e., $|S - S^*| \leq \epsilon$ for a single sufficient statistic or $\rho(\mathbf{S} - \mathbf{S}^*) \leq \epsilon$ for some distance metric ρ (often Euclidean distance). In this case, the posterior generated is only an approximation. Beaumont et al. (2002) suggest regression approaches that attempt to adjust posterior values given the distance between the simulated and actual data. A far more serious issue is when we do not use sufficient statistics for the parameters of interest (and these are often unknown, if they even exist at all). This introduces a second source of error that can be rather critical, especially if a significant amount of the information about an unknown parameter (or parameters) is not captured by $\mathbf{S}$. A number of approaches for dealing with this issue have been suggested (e.g., Joyce and Marjoram 2008; Nunes and Balding 2010; Jung and Marjoram 2011; Aeschbacher et al. 2012), but this is a very active research area with no current clean answer.

References

Aeschbacher, S., M. A. Beaumont, and A. Futschik. 2012. A novel approach for choosing summary statistics in apprximate Bayesain computation. *Genet.* 192: 1027–1047. [A3]

Besag, J., P. J. Green, D. Higdon, and K. L. M. Mengersen. 1995. Bayesian computation and stochastic systems (with discussion). *Statistical Science* 10: 3–66. [A3]

Beaumont, M. A. 2010. Approximate Bayesian computation in evolution and ecology. *Ann. REv. Ecol. Evol. Stst.* 41: 379–406. [A3]

Beaumont, M. A., W. Zhang, and D. J. Balding. 2002. Approximate Bayesian computation in population genetics. *Genetics* 162: 2025–2035. [A3]

Brooks, S. P., and G. O. Roberts. 1998. Convergence assessment techniques for Markov chain Monte Carlo. *Stat. Comput.* 8: 319–335. [A3]

Casella, G., and E. I. George. 1992. Explaining the Gibbs sampler. *Am. Stat.* 46: 167–174. [A3]

Chen, M.-H., Q.-M. Shao, and J. G. Ibrahim. 2000. *Monte Carlo methods in Bayesian computation.* Springer-Verlag, New York. [A3]

Chib, S., and E. Greenberg. 1995. Understanding the Metropolis-Hastings algorithm. *American Statistician* 49: 327–335. [A3]

Cowles, M. K., and B. P. Carlin. 1996. Markov chain Monte Carlo diagnostics: A comparative review. *J. Am. Stat. Assoc.* 91: 883–904. [A3]

Csilléry, K., M. G. B. Blum, O. E. Gaggiotti, and O, François. 2010. Approximate Bayesian computation (ABC) in practice. *Trends Ecol. Evol.* 25: 410–418. [A3]

El Adlouni, S., A.-C. Favre, and B. Bobée. 2006. Comparison of methodologies to assess the convergnce of Markov chain Monte Carlo methods. *Comp. Stat. Data Anal.* 50: 2685–2701. [A3]

Evans, M., and T. Swartz. 1995. Methods for approximating integrals in statistics with special emphasis on Bayesian integration problems. *Stat. Sci.* 10: 254–272. [A3]

Gammerman, D. 1997. *Markov chain Monte Carlo.* Chapman and Hall, New York.[A3]

Gelfand, A. E., and A. F. M. Smith. 1990. Sampling-based approaches to calculating marginal densities. *J. Am. Stat. Assoc.* 85: 398–409. [A3]

Gelman, A., and D. B. Rubin. 1992. Inferences from iterative simulation using multiple sequences (with discussion). *Statistical Science* 7: 457 - 511. [A3]

Geman, S. and D. Geman. 1984. Stochastic relaxation, Gibbs distribution and Bayesian restoration of images. *IEE Transactions on Pattern Analysis and Machine Intelligence* 6: 721–741. [A3]

Geweke, J. 1992. Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. *In* J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (eds.), *Bayesian Statistics 4*, pp. 169-193. Oxford University Press. [A3]

Geyer, C. J. 1992. Practical Markov chain Monte Carlo (with discussion). *Stat. Sci.* 7: 473–511. [A3]

Hastings, W. K. 1970. Monte Carlo sampling methods using Markov Chains and their applications. *Biometrika* 57: 97–109. [A3]

Joyce, P., and P. Marjoram. 2008. Approximately sufficient statistics and Bayesian computation. *Stat. Appl. Genet. Mol. Biol.* 7: a26. [A3]

Jung, H., and P. Marjoram. 2011. Choice of summary statistic weights in approximate Bayesian computation. *Stat. Appl. Genet. Mol. Biol.* 10: a45. [A3]

Kass, R. E., B. P. Carlin, A. Gelman, and R. M. Neal. 1998. Markov chain Monte Carlo in practice: A roundtable discussion. *Amer. Stat.* 52: 93–100. [A3]

Lee, P. M. 2013. *Bayesian statistics: An introduction*, 4th ed. Wiley, New York. [A2, A3]

Liu, J., W. H. Wong, and A. Kong. 1991. Correlation structure and convergence rates of the Gibbs sampler (I): Application to the comparison of estimators and augmentation schemes. Technical Report 299, Dept. Statistics, University of Chicago. [A3]

MacEachern, S. N., and L. M. Berliner. 1994. Subsampling the Gibbs sampler. *Amer. Stat.* 48: 188–190. [A3]

Majoram, P., J. Molito, V. Plagnol, and S. Tavaré. 2003. Markov chain Monte Carlo without likelihoods. *Proc. Natl. Acad. Sci. USA* 100: 15324–15328. [A3]

Majoram, P., and S. Tavaré. 2006. Modern computational approaches for analysing molecular genetic variation data. *Nat. Rev. Gene.* 7: 759–770. [A3]

Mengerson, K. L., C. P. Robert, and C. Guihenneuc-Jouyaux. 1999. MCMC convergence diagnostics: A review. *In* J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (eds.), *Bayesian Statistics 4*, pp. 415–440. Oxford University Press. [A3]

Metropolis, N., and S. Ulam. 1949. The Monte Carlo method. *J. Amer. Statist. Assoc.* 44: 335–341. [A3]

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A.Teller, and H. Teller. 1953. Equations of state calculations by fast computing machines. *J. Chem. Phys.* 21: 1087–1091. [A3]

Nunes, M. A., and D. J. Balding. 2010. On optimal selection of summary statistics for approximate Bayesian computation. *Stat. Appl. Genet. Mol. Biol.* 9: a34. [A3]

Ovaskainen, O., J. M. Cano, and J. Merilä. 2008. A Bayesian framework for comparative quantitative genetics. *Proc. Roy. Soc. B* 275: 669–678. [A3]

Pritchard, J. K., M. T. Seielstad, A. Perez-Lezaun, and M. W. Feldman. 1999. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Mol. Bio. Evol.* 16: 1791–1798. [A3]

Peltonen, J., J. Venna, and S. Kaski. 2009. Visualizations for assessing convergence and mixing of Markov chain Monte Carlo simulations. *Compt. Stat. Data Anal.* 53: 4453–4470. [A3]

Raftery, A. E., and S. Lewis. 1992a. How many iterations in the Gibbs sampler? *In* J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (eds.), *Bayesian Statistics 4*, pp. 763–773. Oxford University Press. [A3]

Raftery, A. E., and S. Lewis. 1992b. Comment: One long run with diagnostics: Implementation strategies for Markov Chain Monte Carlo. *Stat. Sci.* 7: 493–497. [A3]

Robert, C. P., and G. Casella. 2004. *Monte Carlo statistical methods*, 2nd Ed Springer Verlag. [A3]

Smith, A. F. M. 1991. Bayesian computational methods. *Phil. Trans. R. Soc. Lond. A* 337: 369–386. [A3]

Smith, A. F. M., and G. O. Roberts. 1993. Bayesian computation via the Gibbs sampler and related Markov chain Monte-Carlo methods (with discussion). *J. Roy. Stat. Soc. Series B* 55: 3-23. [A3]

Sorensen, D. and D. Gianola. 2002. *Likelihood, Bayesian and MCMC methods in quantitative genetics.* Springer, New York. [A3]

Tanner, M. A. 1996. *Tools for statistical inference*, 3rd ed. Springer-Verlag, New York. [A3]

Tavaré, S., D. J. Balding, R. C. Griffiths, and P. Donnelly. 1997. Inferring coalescence times from DNA sequence data. *Genet.* 145: 505–518. [A3]

Tierney, L. 1994. Markov chains for exploring posterior distributions (with discussion). *Ann. Statist.* 22: 1701–1762. [A3]

Tukey, J. W. 1962. The future of data analysis. *Ann. Math. Stat.* 33: 1–67. [A3]

Weiss, G., and A. von Haeseler. 1998. Inference of population history using a likelihood approach. *Genet.* 149: 1539–1546. [A3]