Identification of Regeneration Times in MCMC Simulation, With Application to Adaptive Schemes

Anthony E. BROCKWELL and Joseph B. KADANE

Regeneration is a useful tool in Markov chain Monte Carlo simulation because it can be used to side-step the burn-in problem and to construct better estimates of the variance of parameter estimates themselves. It also provides a simple way to introduce adaptive behavior into a Markov chain, and to use parallel processors to build a single chain. Regeneration is often difficult to take advantage of because, for most chains, no recurrent proper atom exists, and it is not always easy to use Nummelin's splitting method to identify regeneration times. This article describes a constructive method for generating a Markov chain with a specified target distribution and identifying regeneration times. As a special case of the method, an algorithm which can be "wrapped" around an existing Markov transition kernel is given. In addition, a specific rule for adapting the transition kernel at regeneration times is introduced, which gradually replaces the original transition kernel with an independence-sampling Metropolis-Hastings kernel using a mixture normal approximation to the target density as its proposal density. Computational gains for the regenerative adaptive algorithm are demonstrated in examples.

Key Words: Adaptive Markov chain Monte Carlo; Atoms; Burn-in; Mixture approximations; Parallel processing; Regenerative simulation; Splitting.

1. INTRODUCTION

Markov chain Monte Carlo (MCMC) methods have become popular in the last decade as a tool for exploring properties of distributions which are known only up to a constant of proportionality. The basic idea is to construct an ergodic Markov chain whose limiting distribution is the same as the distribution of interest π (called the *target* distribution). Further details, as well as information about various aspects of implementation, are given in a variety of sources, including Tierney (1994), Gilks, Richardson, and Spiegelhalter (1996), and Robert and Casella (1999). Development of the underlying probabilistic theory

Anthony E. Brockwell is Assistant Professor, and Joseph B. Kadane is Professor, Department of Statistics, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213 (E-mail addresses: abrock@stat.cmu.edu *and* kadane@stat.cmu.edu).

Journal of Computational and Graphical Statistics, Volume 14, Number 2, Pages 436–458
DOI: 10.1198/106186005X47453

of Markov chains taking values in a general state-space can be found in Revuz (1975), Nummelin (1984), and Meyn and Tweedie (1993).

There are several important problems associated with standard methods of constructing Markov chains for Monte Carlo simulation. The first is a convergence problem, sometimes known as the "burn-in problem." One does not generally know how long it takes before the chain is (by some measure) sufficiently close to its limiting distribution. A standard approach to dealing with this is simply to discard some initial portion of the chain, labeling it as a "burn-in" component of the chain. However, without the ability to start the chain with a "perfect sample" from the target distribution, the problem, although reduced, still remains. A second problem is the inherent correlation between successive elements of the chain, which makes it difficult to estimate the variance of the Monte Carlo estimates. One widely adopted method for estimating this variance is the *batch-means* method, which relies on division of the Markov chain into equal-length segments. As pointed out by Glynn and Iglehart (1990), batch-means estimates using fixed batch sizes are not consistent for the true normalized variance, although Chien, Goldsman, and Melamed (1997) pointed out that by allowing the batch size to increase along with the length of the chain, one can obtain a consistent estimator. A related problem is that the chain might "mix" poorly. Even if the chain starts in exactly the limiting distribution, it may take a long time for the chain to explore all interesting parts of the state-space, particularly when the correlation between successive elements of the chain is high. A fourth problem is the simple computational burden required to carry out MCMC simulation. Errors in integrals approximated by MCMC simulation are approximately proportional to the inverse of the square root of the length of the chain, hence Markov chains often need to be relatively long to give accurate estimates of integrals.

Regenerative simulation provides a means of dealing with all of these problems. As discussed by Crane and Iglehart (1975a, 1975b), Crane and Lemoine (1977), Ripley (1987), and more recently in Mykland, Tierney, and Yu (1995), Jones and Hobert (2001), and Robert (1995), regenerative simulation can be used to avoid the burn-in problem and to get consistent estimates of the variance of estimators themselves. Gilks, Roberts, and Sahu (1998) showed how regeneration can be used to introduce so-called "adaptive" schemes into MCMC simulation. These schemes allow parameters of the transition kernel to be modified as the chain is being generated, to improve mixing properties. Mykland, Tierney, and Yu (1995) also hinted at the possibility of using regeneration to "parallelize" generation of a Markov chain. Constructing a Markov chain on parallel processors can give dramatic improvements in speed. [See, e.g., Geyer (1992) for discussion of the relative merits of constructing a single long chain rather than multiple short chains.]

Loosely speaking, a regenerative process "starts again" probabilistically at each of a set of random stopping times, called *regeneration times*. If regeneration times can be identified in an ergodic Markov chain, then tours of the chain between these times are independent and identically distributed (iid) entities. By generating a fixed number of tours (this is typically done by starting the chain in its regenerative state and stopping it at a regeneration time), one is able to estimate the variance of estimators themselves. Furthermore, there is no need for the tours to be generated on a single processor—they can be generated separately on

different processors and then "patched together" in any order that does not depend on the contents of the tours themselves. Because tours are iid, the problem of burn-in is avoided. In addition, results of Gilks, Roberts, and Sahu (1998) establish that parameters (which could be, for instance, variances of proposal distributions used in a Metropolis-Hastings chain) can be modified at each regeneration time, without disrupting consistency of MCMC estimators. This procedure, carried out carefully, can significantly improve mixing of the chain.

Although every ergodic Markov chain is regenerative (see Meyn and Tweedie 1993, theorem 5.2.3), identification of regeneration times in a Markov chain is generally a difficult problem (unless the chain takes values in a finite state-space). Mykland, Tierney, and Yu (1995) gave an ingenious method, based on the splitting technique of Nummelin (1978) for identifying regeneration times in a large class of Markov chains (including chains generated by the standard algorithms for MCMC simulation). Their method relies on establishing a so-called "minorization condition," which is essentially a decomposition of the transition kernel of the chain into two components, one of which does not depend on the current state of the chain. Geyer and Thompson (1995) also described a method of identifying regeneration times in a chain generated using their simulated tempering approach, assuming that it is possible to generate independent samples at a particular temperature level. However, it can be problematic to determine how to obtain the independent samples.

In this article, we present an alternative way of identifying regeneration times, which relies on constructing a Markov chain on an enlarged state-space. The chain is then (by construction) guaranteed to satisfy the minorization condition, and regeneration times are trivial to identify. The idea is to modify the initial target distribution π by mixing it with a point mass concentrated on an "artificial atom" α which is outside the state-space E. It is then a straightforward matter (for instance, using the Metropolis-Hastings algorithm) to construct a Markov chain with the new target distribution. For this chain, the state α is Harris-recurrent (i.e., with probability one, it occurs infinitely many times). By the Markov property, the times at which the new chain hits α are regeneration times. On the face of it, this might not seem useful, since the new chain does not have the desired target distribution. However, to recover an ergodic chain with limiting distribution π, one can simply delete every occurrence of the state α from the new chain. The points immediately after the (deleted) occurrences of the state α are then regeneration times in a Markov chain with limiting distribution π. This makes identification of regeneration times trivial. Møller and Nicholls (2005) used a generalized form of this mixed target distribution for purposes of perfect-sampling. They did not, however, address applications in regenerative simulation or parallel processing. (Note also that this approach cannot be regarded as a special case of simulated tempering with the distribution for one temperature level being a point mass concentrated on α, since the simulated tempering method requires distributions on adjacent levels to have the same support.)

In addition to presenting this "constructive" method for identification of regeneration times, we describe how an existing transition kernel can be embedded into a new kernel for which identification of regeneration times is trivial. This is described in terms of our state-

space augmentation approach, but the embedding can also be thought of as effectively adding an independence-sampler in between each application of the original kernel [Mykland et al. (1995) also advocated this idea at the end of their Section 4], and provides a convenient way to make the transition from standard to regenerative MCMC simulation.

We also describe an adaptive algorithm that progressively replaces an original kernel, with an independence sampler for which the proposal distribution is constructed as a crude mixture approximation to the target distribution. This adaptation is easily incorporated into the framework we use to identify regeneration times, and the idea is very similar to that used by Chauveau and Vandekerkhove (2002), but avoids problems associated with construction of a histogram to represent a current approximation to the target distribution. In terms of mixing, the adaptive algorithm typically outperforms nonadaptive algorithms. Other adaptive schemes have been proposed, including the adaptive direction sampling method of Gilks, Roberts, and George (1994) and the normal kernel coupling method of Warnes (2000). These methods are not adaptive in the sense that the transition kernel changes over time, however. Instead, they create parallel chains, and allow updates for each chain to depend not only on its own current value, but also on the current values of the other chains. Furthermore, both adaptive direction sampling and normal kernel coupling represent generalizations of random walk proposal based samplers. As such, they are not necessarily useful in cases where random walk proposals perform poorly. Tierney and Mira (1999) also described a scheme in which rejected proposals may be retried, effectively allowing the data to determine which of two forms of proposal distribution is preferred. In contrast with all of these approaches, the scheme described in this article allows the transition kernel itself to be modified (at regeneration times) based on the entire past history of the chain. In general, this allows for greater flexibility in the class of adaptive schemes. As Tierney and Mira (1999) pointed out, there is a significant variety of adaptive schemes, and usually, choice of the best scheme will be highly problem-specific. Thus, we do not carry out numerical comparisons of performance between our proposed scheme and other existing adaptive schemes, as one could almost always find examples which favor one particular scheme. Rather, our goal is to introduce a flexible and relatively generic adaptive scheme, and to demonstrate that it is easy to incorporate into the method we propose for identifying regeneration times.

Section 2 introduces the proposed method for constructing a regenerative chain. Section 3 gives more explicit details on how it can be used for purposes of regenerative and adaptive simulation. A general adaptive algorithm is given for estimating the expectation of a function of an unknown parameter as well as the variance of the estimator itself, and a specific adaptation mechanism is described. Section 4 presents examples of application of the method. Proofs, additional theoretical results, and additional analysis of examples are given in an online supplemental document available at http://www.amstat.org/publications/jcgs/ftp.html.

2. THE METHOD

We propose the following method for constructing a Markov chain and identifying regeneration times.

The main idea is to enlarge the state-space from E to

$$E^* = E \cup \alpha,$$

where α is a new state called the *artificial atom*. (Note that the artificial atom is an abstract creation which should not be confused with, for instance, any member of the set E, or any particular number on the real line.) Then it is possible (as described in Section 3.1) to construct a Markov chain $\{Y_t,\ t = 0, 1, 2, \ldots\}$ with $Y_0 = \alpha$ and with limiting distribution

$$\pi_p^*(A) = (1-p)\pi(A \setminus \alpha) + pI_A(\alpha), \tag{2.1}$$

defined on appropriate subsets A of E^*, where p is some constant in the interval $(0, 1)$. The new limiting distribution π_p^* is a mixture distribution which assigns mass p to the new state α and mass $(1-p)$ to the original distribution π. Next let $\tau_Y(j),\ j = 1, 2, \ldots,$ denote the jth time after time zero that the chain $\{Y_t\}$ hits the state α, with $\tau_Y(0) = 0$, so that

$$\tau_Y(j) = \min\{k > \tau_Y(j-1) : Y_k = \alpha\}, \quad j = 1, 2, 3, \ldots, \tag{2.2}$$

and define the tours $Y^1, Y^2, \ldots$ to be the segments of the chain between the hitting times, that is

$$Y^j = \{Y_t,\ \tau_Y(j-1) < t \leq \tau_Y(j)\}, \quad j = 1, 2, \ldots. \tag{2.3}$$

Once the Markov chain $\{Y_t\}$ has been generated, the next step is to recover a regenerative chain with limiting distribution π. This can be done quite simply. Define the chain $\{Z_t,\ t = 0, 1, 2, \ldots\}$ to be exactly the chain $\{Y_t\}$, with every occurrence of the state α removed. Because the state α occurs exactly once at the end of each tour Y^j, $\{Z_t\}$ can be constructed by stringing together the tours Y^j whose length is larger than one, after removing the last element (which is equal to α) from each one. Tours Z^j of $\{Z_t\}$ are then defined by their correspondence to the truncated tours of $\{Y_t\}$. We denote by T_j the time at which the $(j+1)$th tour of $\{Z_t\}$ begins, for $j = 0, 1, 2, \ldots$, with the convention that T_0 is equal to zero, so we can then write the tour lengths as $N_j = T_j - T_{j-1},\ j = 1, 2, \ldots$.

The following result, proved in the online supplemental document, establishes the validity of this procedure.

Theorem 1. *Suppose that $\{Y_t,\ t = 0,1,2,\ldots\}$ is an ergodic Markov chain taking values in E^* with $Y_0 = \alpha$ and limiting distribution π_p^* given by (2.1). Let the process $\{Z_t\}$ be constructed from $\{Y_t\}$ in the manner described above. Then $\{Z_t\}$ is an ergodic Markov chain taking values in E with limiting distribution π. Furthermore, the times $T_j,\ j = 0,1,2,\ldots$ are regeneration times for the chain. In other words, the Markov chains $\{Z_{t-T_i},\ t = T_i, T_i+1, \ldots\},\ i \geq 0$ are identically distributed, and given any T_i and nonnegative integers s and t, Z_s and Z_t are independent if $s < T_i < t$.*

Furthermore, the following result, also proved in the online supplemental document, establishes a connection between this method and the minorization condition approach of Mykland, Tierney, and Yu (1995).

Theorem 2. *Let $\{Z_t\}$ be the Markov chain constructed as described above. Also define the transition kernels $Q(x, A) = Pr(Z_{t+1} \in A | Z_t = x)$ and $P(x, A) = Pr(Y_{t+1} \in A | Y_t = x)$. Then $Q(\cdot, \cdot)$ satisfies the minorization condition*

$$Q(x, A) \geq s(x)\nu(A),$$

with $s(x) = P(x, \alpha)$ and $\nu(A) = \frac{P(\alpha, A)}{1 - P(\alpha, \alpha)}$.

Theorem 2 states that the prescribed construction of $\{Z_t\}$ automatically ensures that the minorization condition is satisfied. In fact, with the given values of $s(x)$ and $\nu(A)$, the procedure given by Mykland, Tierney, and Yu (1995) would identify exactly the same regeneration times as those specified by Theorem 1.

3. SIMULATION ALGORITHMS

This section briefly reviews the technique of estimation using regenerative simulation, as discussed by Crane and Lemoine (1977) and, more recently, by others. We also explicitly state regenerative and adaptive MCMC simulation algorithms based on the construction we have introduced.

3.1 ESTIMATING FUNCTIONALS OF THE TARGET DISTRIBUTION

One is often interested primarily in estimating

$$\pi_h = \int_E h(x) d\pi(x)$$

for some integrable function $h : E \to \mathbb{R}$. Let tours Z^j, tour lengths N_j, and cumulative tour lengths T_j of the Markov chain $\{Z_t\}$ with limiting distribution π be as defined in the previous section. Also let

$$H_j = \sum_{t=T_{j-1}}^{T_j - 1} h(Z_t), \tag{3.1}$$

and assume that H_j and N_j have finite variances. [This assumption is generally difficult to check, but Hobert et al. (2002) gave sufficient conditions for it to hold.] Then by the strong law of large numbers, the ratio estimator

$$\hat{H}_n = \frac{\sum_{j=1}^n H_j}{\sum_{j=1}^n N_j} = \frac{\sum_{j=1}^n H_j}{T_j} \tag{3.2}$$

converges almost surely to π_h, and it follows from the central limit theorem that $\sqrt{n}(\hat{H}_n - \pi_h)$ converges to a $\mathrm{N}(0, \sigma^2)$ distribution.

One way of estimating σ^2 is as follows. Let $V_j = H_j - \pi_h N_j$, so that $\{V_j\}$ is an iid sequence of random variables, and define $\bar{V}_n = n^{-1} \sum_{j=1}^n V_j$, $\bar{N}_n = n^{-1} \sum_{j=1}^n N_j$, and

$\mu_N = \mathbf{E}N_1$. Then

$$\begin{aligned}\mathbf{E}(\sqrt{n}(\hat{H}_n - \pi_h)^2) &= n\mathbf{E}(\bar{V}_n/\bar{N}_n)^2 \\ &\simeq n\mathbf{E}(\bar{V}_n/\mu_N)^2 \\ &= \frac{1}{\mu_N^2}\text{var}(V_1) \simeq \hat{\sigma}_n^2, \end{aligned} \tag{3.3}$$

where

$$\hat{\sigma}_n^2 = \frac{n^{-1}\sum_{j=1}^n (H_j - \hat{H}_n N_j)^2}{\bar{N}^2}. \tag{3.4}$$

In the preceding argument, in order to avoid dealing with the ratio of random variables in (3.3), $\bar{N}$ is effectively treated as the constant μ_N. Hence, although $\hat{\sigma}_n^2$ is a consistent estimator of σ^2, to limit the error in the approximation for finite n, it is desirable for the coefficient of variation c_n of $\bar{N}$ to be small. Mykland, Tierney, and Yu (1995) suggested that the estimator should not be used if the coefficient is larger than .01. The coefficient of variation c_n depends on the distribution of the tour lengths N_j, which is usually not known. However, it may be estimated by

$$\hat{c}_n = \sum_{j=1}^n (N_j/T_n - 1/n)^2. \tag{3.5}$$

Furthermore, noting that c_n (apart from some random variation) is proportional to n^{-1}, if we have $\hat{c}_{n_1} > \epsilon$, then approximately $n_2 = n_1(\hat{c}_{n_1}\epsilon^{-1} - 1)$ additional tours are required to ensure that $c_{n_1+n_2}$ is less than ϵ.

3.2 CONSTRUCTING A CHAIN WITH LIMITING DISTRIBUTION π_p^*

This section gives two methods that can be used to construct a chain taking values in E^* with limiting distribution π_p^* (recall the definition (2.1)), (Note that the constant p is not specified in advance, but the methods given below yield a Markov chain with limiting distribution π_p^* for some unknown p in the interval $(0, 1)$. The value of p depends on the unknown normalizing constant β.)

3.2.1 Metropolis-Hastings

One of the most obvious ways to construct $\{Y_t\}$ is simply to use the standard Metropolis-Hastings algorithm, making appropriate modifications to densities so that they are defined on E^* instead of only on E.

Assume that the measure π has a density (not necessarily with respect to Lebesgue measure), which, for the sake of simplicity, we will also denote by π. In order to allow for the usual case where the density is known only to within a constant of proportionality, we will assume that the density integrates to an unknown constant β.

Define the density

$$\pi^*(y) = \begin{cases} \pi(y), & y \in E, \\ k, & y = \alpha \end{cases}$$

on E^*, where k is some positive constant, and let $g_\theta(Y_t, \cdot)$ denote a family of proposal distributions on E^*, satisfying the property that for each θ, $g_\theta(y, \alpha) > 0$ for all $y \in E$ and $g_\theta(\alpha, A) > 0$ for all sets A such that $\pi(A) > 0$. Also define the acceptance probabilities

$$a_1(Y_t, Z) = \min\left(1, \frac{\pi^*(Z) g_\theta(Z, Y_t)}{\pi^*(Y_t) g_\theta(Y_t, Z)}\right), \quad Y, Z \in E^*.$$

We introduce the parameter θ so that we can use adaptive methods for modifying the proposal distribution as the chain runs; this is described in the next section. Then the standard Metropolis-Hastings algorithm using proposal density $g_\theta(\cdot, \cdot)$, acceptance probabilities $a_1(\cdot, \cdot)$, and target density π^*, generates a Markov chain with limiting distribution π_p^*, $p = k(\beta + k)^{-1}$, on the augmented space E. (Note that p is always strictly between 0 and 1, as required, in spite of the fact that β is not known.)

Example 1. *Suppose the original un-normalized target density on $\mathbb{R}$ is $\pi(y) = exp(-y^2/2)$. For $y \in \{\mathbb{R} \cup \alpha\}$, define $\pi^*(y) = \pi(y)$ if $y \in \mathbb{R}$, and $\pi^*(y) = 1$ if $y = \alpha$. In order for the Metropolis-Hastings chain to be irreducible, the proposals must allow the chain to be able to reach the state α with positive probability, and to be able to reach sets in $\mathbb{R}$ of positive Lebesgue measure with positive probability. The proposal density could be chosen for instance, as*

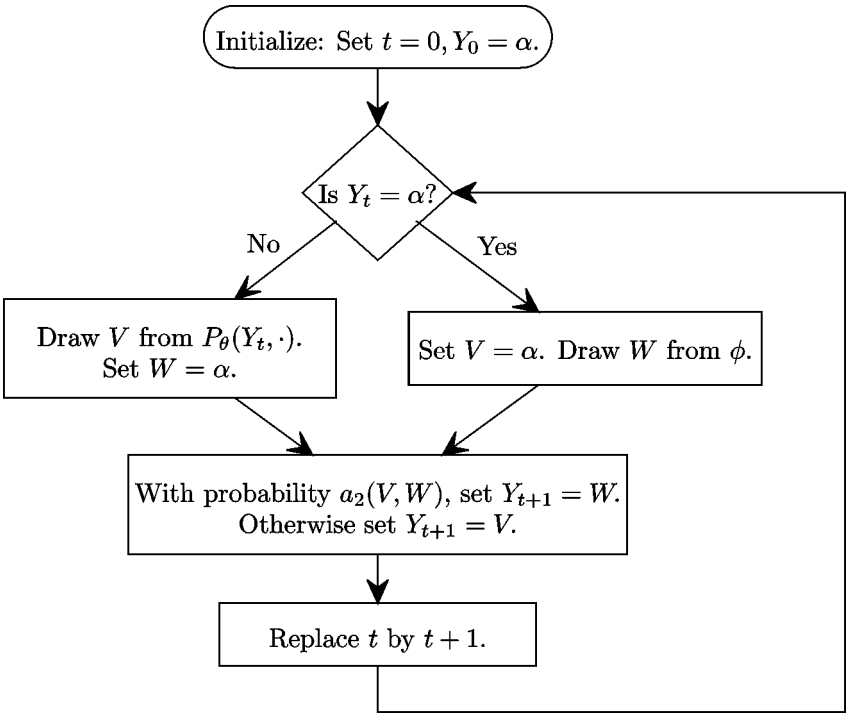
$$g(y, z) = \begin{cases} .1, & y \in \mathbb{R}, z = \alpha, \\ .9(2\pi)^{-1/2} exp[-(z-y)^2/2], & y \in \mathbb{R}, z \in \mathbb{R} \\ (20\pi)^{-1/2} exp(-z^2/20) & y = \alpha, z \in \mathbb{R}, \\ 0 & y = \alpha, z = \alpha. \end{cases}$$

Thus, for $y \in \mathbb{R}$, the proposal z is a mixture of a random walk proposal on the real line, and a point mass concentrated on α. For $y = \alpha$, the proposal is on the real line, and has a N(0,10) distribution.

3.2.2 A Hybrid Kernel

An alternative method for constructing $\{Y_t\}$ is to build a hybrid kernel around an existing transition kernel. [Gilks et al. (1998) also briefly mentioned the possibility of using a hybrid kernel.]

Suppose that a kernel $P_\theta(\cdot, \cdot)$ is given for an ergodic Markov chain with limiting distribution π, for instance, a Metropolis-Hastings or Gibbs sampling chain. The hybrid kernel consists of two components. The first component leaves the state at α if it is already α, otherwise it updates the state using the π-invariant kernel $P_\theta(\cdot, \cdot)$. (Again, θ is a parameter which will enable us to use adaptive methods as described in the next section.) The second component is a Metropolis-Hastings step, in which the proposal is α if the current state is



Algorithm 1. Hybrid Metropolis-Hastings.

in E, and is drawn from a "re-entry proposal distribution" ϕ on E if the current state is α. It is not difficult to verify that each of these components has invariant distribution π_p^* for some $p \in (0, 1)$, and that the hybrid kernel inherits properties of irreducibility, aperiodicity, and so on from $P_\theta(\cdot, \cdot)$.

Let

$$a_2(V, W) = \begin{cases} \min(1, \frac{k\phi(V)}{\pi(V)}), & W = \alpha \\ \min(1, \frac{\pi(W)}{k\phi(W)}), & W \in E. \end{cases},$$

where k is some positive constant. Algorithm 1 above uses this hybrid kernel approach, building around an existing π-invariant kernel $P_\theta(\cdot, \cdot)$ to obtain a Markov chain with limiting distribution π_p^*, where (as in the previous subsection) $p = k(\beta + k)^{-1}$.

In the online supplemental document, it is proved that, as long as the re-entry proposal distribution ϕ has the same support as π (i.e., $\phi(A) > 0 \iff \pi(A) > 0$), the chain $\{Y_t,\ t = 0, 1, 2, \ldots\}$ generated by Algorithm 1 is ergodic with limiting distribution π_p^*.

Inspecting the form of the acceptance probability $a_2(V, W)$, it is clear that in order for the chain to have high probabilities of acceptance in going from E to α and *also* in going from α to E, it is necessary to have $k\phi(x)$ approximately equal to $\pi(x)$ for values x in high-density regions of π. Thus, as a general rule, it is desirable to choose a re-entry proposal distribution ϕ which is reasonably close to the normalized version of the target distribution π, and to choose k to be roughly of the order of magnitude of $\phi(x)/\pi(x)$, where

x is some point in a high-density region for π. [Recall that we do not assume that the density π is normalized. Also note the parallel discussion by Mykland et al. (1995) after Theorem 3, of the desirability of choosing a constant c appropriately.] Empirical observations indicate that as ϕ becomes less similar to the normalized version of π, the coefficient of variation of the distribution of tour lengths N_j tends to increase. In particular, more very large and very small tour lengths are observed.

One can control the expected tour length (but not its variance) by adjusting the parameter k. Increasing k reduces the expected tour length, while decreasing it increases the expected length. (This follows directly from Theorem A.1, stated and proved in the online supplemental document.) In the extreme case, if $k\phi(x) \geq \pi(x)$ for all $x \in E$, then the algorithm becomes exactly a rejection-sampling algorithm, yielding a chain which alternates between α and E, in which the non-α values are iid draws from the target distribution.

In practice, a small segment of a chain generated using the original kernel $P_\theta(\cdot, \cdot)$ can be used to determine ϕ and k. In many cases, ϕ can be taken to be a mixture normal approximation to the distribution of the elements of the initial segment, obtained, for instance, using the method described later in this section. The constant k can then be selected to be approximately the average value of π, evaluated over the elements of the initial segment (recall that π is not normalized), divided by the average value of the density ϕ, evaluated over a number of draws from ϕ itself.

Algorithm 1 is particularly useful because it provides a simple means of wrapping up an existing MCMC algorithm for sampling from π to yield a Markov chain with limiting distribution π_p^*. However, it is important to note that Algorithm 1 does not necessarily improve mixing of the underlying kernel $P_\theta(\cdot, \cdot)$. If $P_\theta(\cdot, \cdot)$ mixes badly and the re-entry distribution is a poor approximation to π, then the resulting chain $\{Y_t\}$ is also likely to mix badly. On the other hand, if $P_\theta(\cdot, \cdot)$ mixes badly and the re-entry distribution is a good approximation of π, the resulting chain will typically exhibit improved mixing, since every tour's first element will be close in distribution to π.

Example 2. *Suppose again that the un-normalized target density is* $\pi(y) = exp(-y^2/2)$. *Assume that* $P_\theta(\cdot, \cdot)$ *is a random walk Metropolis-Hastings step. Given* $Y_t \in \mathbb{R}$, *it generates a random walk proposal* $Z \sim N(Y_t, \theta)$, *computes the corresponding acceptance probability, and either accepts or rejects the proposal. Assume that* $\theta = 1$. *The re-entry proposal density could be chosen as* $\phi(y) = (20\pi)^{-1/2} exp(-y^2/20)$ *and* k *could be chosen to be equal to one. The chain would have initial state* $Y_0 = \alpha$, *and Algorithm 1 would then generate* Y_{t+1} *from* Y_t *by the following procedure.*

1. *If* $Y_t = \alpha$, *then set* $V = \alpha$. *Otherwise generate* V *using the standard Metropolis-Hastings update as follows.*
   (a) *Draw a proposal* $Z \sim N(Y_t, \theta)$.
   (b) *With probability* $\min(\pi(Z)/\pi(Y_t), 1)$, *set* $V = Z$. *Otherwise set* $V = Y_t$.
2. *If* $V \in \mathbb{R}$, *then set*

$$Y_{t+1} = \begin{cases} \alpha & \text{with probability } min(1, \phi(V)/\pi(V)), \\ V & \text{otherwise.} \end{cases}$$

On the other hand, if $V = \alpha$, draw a proposal W from ϕ, and set

$$Y_{t+1} = \begin{cases} W & \textit{with probability } \min(1, \pi(W)/\phi(W)), \\ \alpha, & \textit{otherwise.} \end{cases}$$

3.3 ADAPTIVE MODIFICATION OF THE KERNEL

Implementation of MCMC sampling schemes typically requires significant effort to be devoted to design of a transition kernel in order to ensure that the chain is well-mixing. This design procedure can include the selection of an appropriate parameterization of the quantities of interest, as well as potential parameters of proposal distributions in a Metropolis-Hastings chain. It is tempting to construct algorithms that modify these design parameters automatically as the chain runs. However, even if for each fixed set of parameter values the kernel has the correct limiting distribution, introduction of self-tuning can alter the limiting distribution of the process (which will not necessarily be Markovian any more). Gelfand and Sahu (1994) gave an interesting example where this problem occurs. One way around this problem is to adopt the approach of Gilks, Roberts, and Sahu (1998), who showed that if the Markov chain is regenerative, then one can modify design parameters at each regeneration time, and estimators are still well-behaved. In particular, Theorems 1 and 2 of Gilks, Roberts, and Sahu (1998) establish that under this self-tuning scheme, if some technical conditions are satisfied, then (recall the definitions (3.2) and (3.4)) $\hat{H}_n$ is MSE-consistent for π_h, and the quantity

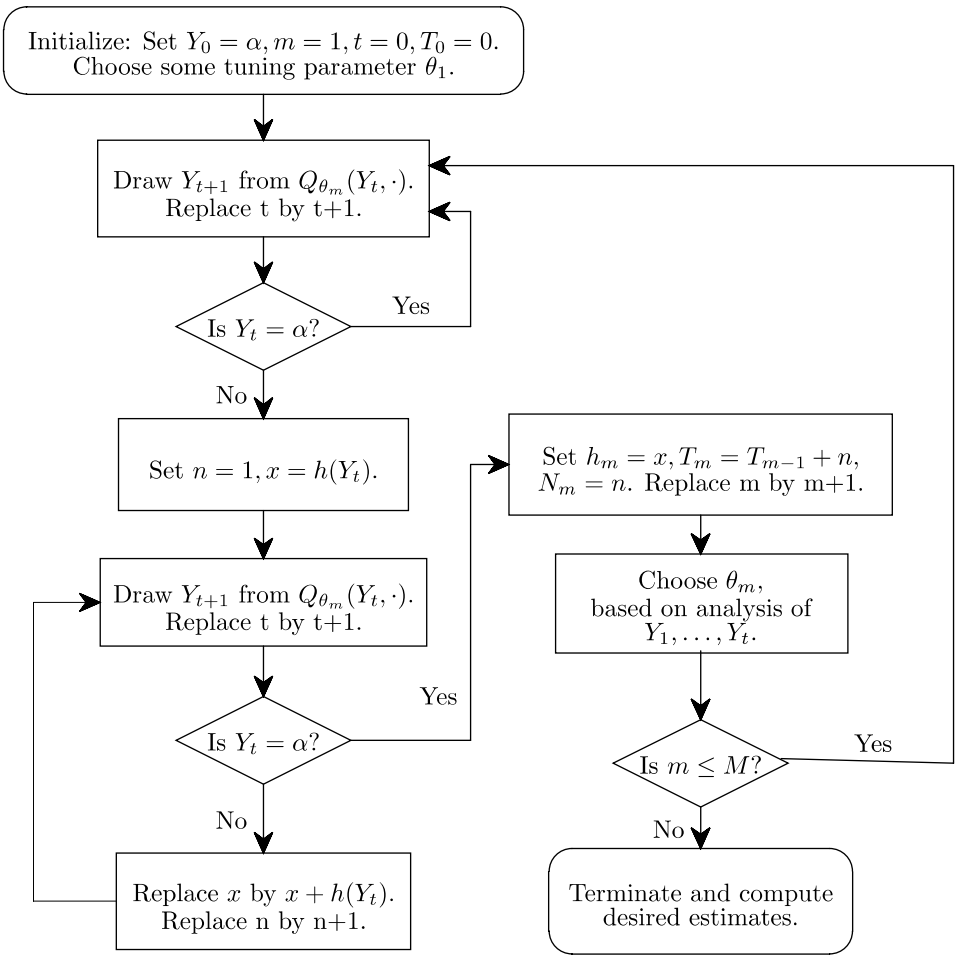
$$\sigma_n^{2\,*} = \frac{n \sum_{j=1}^{n} (H_j - \pi_h N_j)^2}{T_n^2} \tag{3.6}$$

converges in probability to σ^2. The difference $(\sigma_n^{2\,*} - \hat{\sigma}_n^2)$ (recall the definition (3.4)) is

$$\begin{aligned} \sigma_n^{2\,*} - \hat{\sigma}_n^2 &= \frac{1}{n\bar{N}^2} \sum_{j=1}^{n} \left[N_j^2 (\hat{H}_n^2 - \pi_h^2) - 2 H_j N_j (\hat{H}_n - \pi_h) \right] \\ &= \bar{N}^{-2} (\hat{H}_n + \pi_h)(\hat{H}_n - \pi_h) n^{-1} \sum_{j=1}^{n} N_j^2 \; - \; 2\bar{N}^{-2} (\hat{H}_n - \pi_h) n^{-1} \sum_{j=1}^{n} H_j N_j. \end{aligned}$$

Assuming that $\mathbf{E}H_jN_j$, $\mathbf{E}N_j$, $\mathbf{E}N_j^2$, $\mathbf{E}H_j^2N_j^2$, and $\mathbf{E}N_j^4$ exist and are finite (as mentioned before, this is generally difficult to check), then each of the terms $(n\bar{N})^{-2}$, $\hat{H}_n + \pi_h$, $\hat{H}_n - \pi_h$, $n^{-1}\sum N_j^2$, and $n^{-1}\sum H_jN_j$, converges in probability to some finite constant, with $(\hat{H}_n - \pi_h)$ converging in probability to zero. It follows that $(\sigma_n^{2\,*} - \hat{\sigma}_n^2)$ converges in probability to zero, and hence that $\hat{\sigma}_n^2$ is a consistent estimator of σ^2.

Thus, in Algorithm 1, the kernel parameter θ can be changed every time the chain hits the state α, based on an analysis of the past behavior of the chain. Even though the process itself is no longer (necessarily) Markovian, it consists of a sequence of tours which, in isolation, are each Markovian, and the estimates $\hat{H}_n$ and $\hat{\sigma}_n^2$ still have the desired consistency properties.



Algorithm 2. Regenerative Adaptive Scheme.

We now state an algorithm, based on the ideas presented in the previous sections, which estimates the expection (with respect to the probability measure π) of a "function of interest" $h(\cdot)$, and allows for adaptive modification of a tuning parameter θ. Let $Q_\theta(\cdot, \cdot)$ denote the transition kernel for a chain whose limiting distribution is π_p^*. Given a desired number of tours $M > 0$, the following algorithm constructs a regenerative chain, an estimate $\hat{H}_M$ of $\int h(x)d\pi(x)$, and an estimate of the variance of $\hat{H}_M$ itself.

Intuitively Algorithm 2, given above, works as follows. It repeatedly generates tours of a Markov chain $\{Y_t\}$ with limiting distribution π_p^*, using the kernel $Q_\theta(\cdot, \cdot)$. Tours of length one (that is, tours which consist only of the state α) are thrown out. Remaining tours are truncated just before they hit α so as to become tours of a chain $\{Z_t\}$ with limiting distribution π. The sum over each tour of the function of interest, evaluated at the chain's state, is recorded, as is the length of the tour. At the end of the mth tour, the tuning parameter θ_{m+1} can be chosen based on analysis of past history of the chain. Once a total of M tours

have been generated, the algorithm terminates, and Equations (3.2) and (3.4) can be used to obtain the desired estimates.

Most of Algorithm 2 is easily implemented. The only complicated part, obtaining draws from a kernel $Q_{\theta_m}(\cdot,\cdot)$ with invariant distribution π_p^*, can be carried out using the basic Metropolis-Hastings algorithm appropriately modified as described in Section 3.2.1, or Algorithm 1. Some consideration should be given also to the method by which the kernel parameter θ_m is updated, although in the simplest case, one can construct a nonadaptive chain by fixing $\theta_m = \theta^*$ for all m.

Example 3. *Suppose again that the un-normalized target density is* $\pi(y) = exp(-y^2/2)$. *The approach described in Example 2 can be used, augmented by the following additional step.*

3. *If* $Y_t = \alpha$ *and* $Y_{t+1} \in \mathbb{R}$, *then a new tour has just begun. At this point, alter the value of* θ *as follows. Let* q *be the proportion of acceptances in applications of* $P_\theta(\cdot,\cdot)$ *from time* 0 *to time* t. *If* $q <$ *1/2, then replace* θ *by .9* θ. *Otherwise replace* θ *by 1.1* θ.

This scheme would adaptively adjust the random walk step size θ, *aiming to reach an approximate average acceptance rate of 1/2. It relies on the basic property that if* θ *is too small, then the acceptance probability is roughly equal to one, while if* θ *is too large, the acceptance probability becomes close to zero.*

3.4 AN ADAPTATION RULE

Algorithm 2 allows for adaptation through selection (at the end of the mth tour) of the parameter θ_{m+1}, which determines the form of the kernel $Q_{\theta_{m+1}}(\cdot,\cdot)$ used in the $(m+1)$th tour. This section describes one method of doing this when the state space E is $\mathbb{R}^c$, based on the concept that an independence sampler with a proposal distribution closely matching the target distribution is near-optimal in terms of mixing. [There are, of course, many other possible ways of adapting parameters of a Markov chain; some of these were discussed briefly in Gilks et al. (1998).]

Our approach is to start with a kernel $Q_0(\cdot,\cdot)$, and as time goes by, to transform it progressively into an independence-sampling kernel which uses a normal mixture approximation to the target distribution as its proposal distribution. The mixture approximation itself is updated at the end of each tour, based on the contents of the tour itself. We use a simple recursive update procedure, described by Titterington (1984), to compute our mixture approximations. The update rule is as follows. Given a multivariate normal mixture

$$\xi_m \sim \sum_{i=1}^{d} \alpha_i \mathrm{N}(\mu_i, \Sigma_i),$$

for which the parameters $\alpha_i, \mu_i, \Sigma_i,\ i = 1, 2, \ldots, d$ are unknown, and an infinite sequence of draws $Y_1, Y_2, \ldots$ from ξ_m, parameter estimates $\hat{\alpha}_i^{(j)}, \hat{\mu}_i^{(j)}, \hat{\Sigma}_i^{(j)}$ can be updated sequentially, once after each draw, by the formulas

$$\begin{aligned}
\hat{\mu}_i^{(j+1)} &\leftarrow \hat{\mu}_i^{(j)} + \frac{w_i^{(j)}}{j\hat{\alpha}_i^{(j)}}(y_{j+1} - \hat{\mu}_i^{(j)}) \\
\hat{\Sigma}_i^{(j+1)} &\leftarrow \hat{\Sigma}_i^{(j)} + \frac{w_i^{(j)}}{j\hat{\alpha}_i^{(j)}}\left[(y_{j+1} - \hat{\mu}_i^{j})(y_{j+1} - \hat{\mu}_i^{(j)})^T - \hat{\Sigma}_i^{(j)}\right] \\
\hat{\alpha}_i^{(j+1)} &\leftarrow \hat{\alpha}_i^{(j)} + j^{-1}(w_i^{(j)} - \hat{\alpha}_i^{(j)}),
\end{aligned}$$

where $w_i^{(j)} \propto \hat{\alpha}_i^j \phi(y_{j+1}; \hat{\mu}_i^{(j)}, \hat{\Sigma}_i^{(j)})$ with $\sum_{i=1}^d w_i^{(j)} = 1$, $\phi(\cdot; \cdot, \cdot)$ denoting the multivariate normal density. As Titterington (1984) pointed out, there is no guarantee of consistency of these sequentially updated estimators, but for our purposes, because the mixture approximation is to be used to generate proposals, we only require a crude approximation to the target distribution, and consistency is not necessary for our adaptive method. Better methods for constructing mixture approximations have been widely studied (see, e.g., McLachlan and Peel 2000), but the majority of these methods cannot be implemented sequentially, which prevents them from being useful in the procedure we propose.

We now state the full adaptive procedure to be used in Algorithm 2. Let the adaptive parameters be defined by $\theta_m = (\eta_m, \xi_m)$, where, for each m, η_m is a constant in the interval $(0, 1)$ and ξ_m is a set of weights, mean vectors, and covariance matrices, defining a normal mixture distribution. We denote the density of the mixture distribution by $\xi_m(\cdot)$.

Let

$$Q_\theta(x, A) = (1 - \eta)Q_0(x, A) + \eta R_\xi(x, A),$$

where $R_\xi(x, A)$ is an independence-sampling Metropolis-Hastings kernel, in which the proposals have density $\xi(\cdot)$, that is,

$$R_\xi(x, A) = I_A(x) \int_E \xi(y)[1 - a_3(x, y)]dy + \int_A \xi(y)a_3(x, y)dy,$$

with $a_3(x, y) = \min(1, \pi(y)\xi(x)(\pi(x)\xi(y))^{-1})$. Thus, the kernel $Q_\theta(\cdot, \cdot)$ is a mixture of the original kernel $Q_0(\cdot, \cdot)$ and the independence-sampling Metropolis-Hastings kernel.

To initialize the adaptive parameters, let $\eta_1 = 0$, and let ξ_1 be some initial normal mixture approximation to π. Then, at the end of the mth tour ($m = 1, 2, \ldots$), carry out the following updates

1. Set $\eta_{m+1} = \min(1.0 - (1.0 - \eta_m) * \kappa, \zeta)$, for some constants κ and ζ in the interval $[0, 1]$.
2. Calculate the parameters of ξ_{m+1} by starting with the mixture distribution ξ_m and updating it sequentially using the rules given above, once for each element of the mth tour.

The first of these two updates increases the relative contribution of the independence sampler in $Q_{\theta_{m+1}}(\cdot, \cdot)$, to a "maximal" proportional contribution of ζ. Choosing $\zeta = 1$ allows the kernel to be (as time increases to infinity) completely replaced by the independence sampler, and is potentially dangerous if ζ becomes very close to one before ξ provides a reasonable approximation to the target density. Choosing $\zeta < 1$ ensures that some of the

original kernel $Q_0(\cdot,\cdot)$ is always retained, and guards against the possibility of building an independence sampler based on the potentially false belief that all important parts of the space have already been explored. The second update simply refines the normal mixture approximation to the target distribution.

Under ideal circumstances, this procedure for adapting θ will gradually replace the original kernel with an independence sampling Metropolis-Hastings kernel, whose proposal density ξ_m becomes close to the true target density as m increases.

In some cases (particularly when the state-space E is high-dimensional), it can be impractical to construct approximations to the target distribution on the entire state-space. However, if the original kernel $Q_0(\cdot,\cdot)$ consists of Metropolis-Hastings block-updates for subspaces of E, it often makes sense to use the same basic idea, but to restrict approximations to the appropriate conditional target densities on the subspaces, and progressively replace the original block-update rules with Metropolis-Hastings independence sampling steps whose proposals are the respective conditional approximations. If conditional target densities are difficult to approximate, a slightly less optimal approach is to use the respective marginal target densities.

4. EXAMPLES

To examine the performance of Algorithm 2 we apply to it two problems.

As a measure of the mixing quality of Markov chains obtained, we compute the *sample precision per iteration* (SPPI) of our estimates, which we define to be $(\hat{\sigma}_n^2 T_M/M)^{-1}$, where M is the number of tours generated, T_M is the total length of the chain, and $\hat{\sigma}_n^2$ is the estimator given by (3.4). For the sake of comparison with results obtained nonregeneratively, we will also compute the SPPI using the batch-means estimator for σ^2. Because the SPPI is itself a random variable, we generate multiple realizations of Markov chains, and show boxplots of the resulting SPPI values over the different realizations.

Assuming that computational cost to obtain an iteration is roughly invariant, precision per iteration is a direct measure of the computational efficiency of an MCMC simulation. It can be easily transformed into other mixing measures, such as the efficiency measure used by Roberts (1996, sec. 3.4). We adopt this particular measure for several reasons. First, it is a direct measure of the amount of information gained about a parameter per iteration of the chain. Given the SPPI and the chain length, we can immediately compute a confidence interval for the parameter. Second, it is additive—for instance, one would obtain roughly double the precision by doubling the total number of tours generated (and hence approximately doubling the length of the chain). Finally, the SPPI inherits consistency properties of $\hat{\sigma}_n^2$. It is consistent for the precision of the parameter estimate divided by the chain length.

4.1 DUGONGS

First we consider a dataset used by Ratkowsky (1983), which was considered by Carlin and Gelfand (1991). Length (Y) and age (X) measurements were made of 27 specimens

of a particular species of sea cows (dugongs), captured near Townsville, Queensland. As discussed by Ratkowsky (1983), a frequently used model for such a dataset is

$$\begin{aligned} Y_i &\sim \mathrm{N}(\mu_i, \tau^{-1}) \\ \mu_i &= \alpha - \beta\gamma^{X_i} \end{aligned} \tag{4.1}$$

for the data, where X_i and Y_i are the age and length of the ith dugong, respectively, and $\alpha > 0$, $\beta > 0$, $\gamma \in (0,1)$, and $\tau > 0$ are unknown model parameters. We assign the (relatively uninformative) priors $\alpha \sim \mathrm{N}(0, 10{,}000)$, $\beta \sim \mathrm{N}(0, 10{,}000)$, $\gamma \sim \mathrm{U}(0,1)$, and $\tau \sim \mathrm{Gamma}(.001, .001)$. Our objective is to determine the posterior mean of α, β, γ, and τ^{-1}.

A Markov chain with the posterior as its limiting distribution can be constructed by computing full-conditional distributions and updating each of the four parameters in turn. The parameters α, β, and τ have conjugate priors for the likelihood (4.1). Hence they can be updated by sampling directly from their respective full-conditional distributions. The parameter γ does not have a conjugate prior. However, it can be updated by using a single Metropolis-Hastings step whose target density is proportional to the full-conditional density of γ. As our proposal distribution for the γ-update, we choose a uniform distribution on the interval $[0,1)$. Denote this kernel by $P(\cdot,\cdot)$.

We simulate using

(A) the kernel P (no regeneration);
(B) the kernel P embedded into Algorithm 1, with a "bad" re-entry distribution;
(C) the kernel P embedded into Algorithm 1, with a "good" re-entry distribution; and
(D) the kernel P embedded into Algorithm 1, along with the adaptive scheme of Section 3.4.

For case (C), the (four-dimensional) re-entry proposal distribution ϕ is constructed by running the original chain for 1,000 iterations, and using these iterations to construct a normal approximation to the target density, using the update rules described in Section 3.4. For case (B), we use the re-entry proposal distribution of case (C), after subtracting three marginal standard deviations from each component of the mean. In each case, the constant k is chosen so that $\log(k)$ is equal to the average log-target density over the 1,000 iterations, minus the average log-density of $\phi(\cdot)$ over 1,000 draws, minus a constant which is chosen (experimentally) to adjust the distribution of the tour-lengths. For case (D), we use Algorithm 2, with the adaptation rule described in Section 3.4, with parameters $\kappa = .01$ and $\zeta = .95$. Sampling from the full-conditional is retained for α, β, and τ, but the uniform proposals for the γ-update are gradually replaced with independent proposals coming from the conditional distribution of γ in a two-component mixture normal approximation to the marginal target density of the full parameter vector.

In cases (B), (C), and (D), we simulate 200 chains, each of 2,000 tours, with total length approximately 1,300,000, and compute estimates and SPPIs using the regenerative approach. For the nonregenerative case (A), we simulate 200 chains, each of length 1,300,000, and use the batch-means method with batch size 4,000 to compute SPPIs.

Table 1 gives posterior means along with their standard errors, estimated using both

Table 1. Posterior Means, Along With Standard Error Estimates Using the Regenerative Method, from Case (A): A Nonadaptive, Nonregenerative Chain (total length 1,300,000, batch length 4,000), and case (D): the corresponding regenerative adaptive chain (2,000 tours, total length 1295719).

	Parameter Estimates			
	Case A		*Case D*	
Function	*Post. mean*	*Std. err.*	*Post. mean*	*Std. err.*
α	2.6611	.0013	2.6636	.0004
β	.9800	.0003	.9807	.0002
γ	.8659	.0006	.8669	.0002
τ^{-1}	.00903	.00001	.00902	.00001

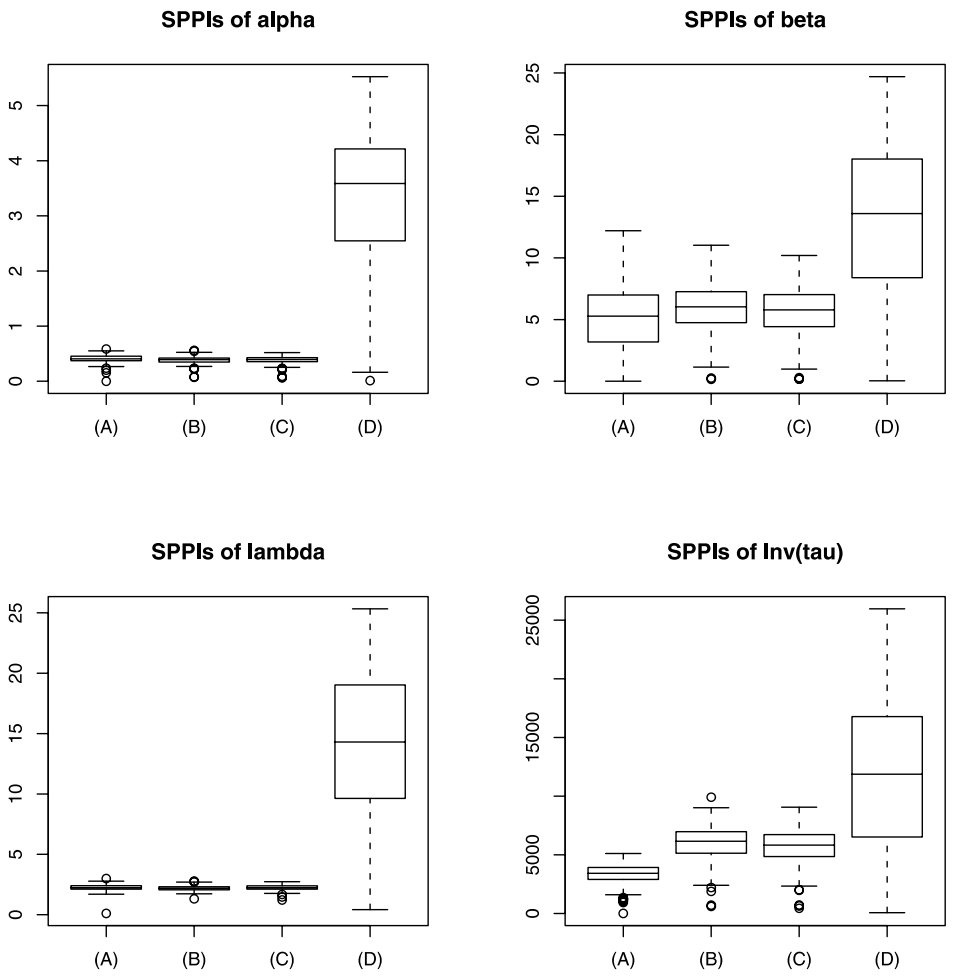


Figure 1. Boxplots of sample precision per iteration (SPPI) for each of the four parameters in the Dugong problem, for (A) nonregenerative case, as well as for regenerative cases with (B) poor re-entry distributions and (C) good re-entry distribution, and (D) regenerative and adaptive case.

Table 2. Median SPPI for Each of the Four Parameters in the Dugong Problem, for Each of the Four Cases

	Median SPPIs			
Parameter	*Case A*	*Case B*	*Case C*	*Case D*
α	.41	.39	.40	3.59
β	5.28	6.03	5.79	13.59
λ	2.24	2.20	2.24	14.30
τ^{-1}	3420.20	6156.24	5824.19	11867.50

the original kernel with no-regeneration, and using the regenerative method to construct an adaptive chain. Figure 1 shows boxplots of the SPPI over 200 separate regenerative chains, each one consisting of 2,000 tours, in all four cases, and the median SPPI values are given in Table 2. The improvement in SPPI due to use of the adaptive algorithm is clearly substantial. Roughly speaking, it appears that one iteration of the adaptive chain is worth two to ten iterations of the nonadaptive chain, with the gain also depending on which parameter is being considered. Furthermore, in this case, mixing performance is relatively robust to poor choice of the re-entry distribution. This robustness may be due in part to the fact that average tour length is around 650, so the distribution of the first element of a tour makes a relatively small contribution to the overall quality of mixing. It should be noted, however, that in cases where the re-entry distribution is substantially worse than in case (B), Algorithm 1 can become virtually impossible to implement, because the probability of obtaining an extremely long tour can increase to the point where the waiting time for the longest of 2,000 tours to be completed becomes unacceptable.

It is also informative to consider the distribution of tour lengths in both adaptive and nonadaptive cases. Histograms, along with discussion, are given in the online supplemental document.

4.2 MONKEYS AND FREE-KNOT SPLINES

We next consider a more complicated problem. Ventura et al. (2002) described experiments in which a macaque monkey watches images appear on a screen. The monkey is trained to move its eyes in response to certain visual cues, and an electrode measures numbers of "neuron-spikes" occurring in a neuron in the monkey's supplementary eye field. (The supplementary eye field is thought to be involved in generating eye movements in response to stimuli.)

Figure 2 shows the number of spikes y_k observed in time intervals $[.03k, .03(k+1)),\ k = 0, 1, 2, \ldots, 99$, time measured in seconds, for one such task.

We assume that the observations $\{y_0, \ldots, y_{99}\}$ are Poisson with a time-varying rate, and can be modeled by

$$Y_k | r(\cdot) \sim \text{Poisson}(\exp(r(k/99)), \sigma^2), \tag{4.2}$$

where $r(\cdot)$ is a cubic spline function with four knots, that is,

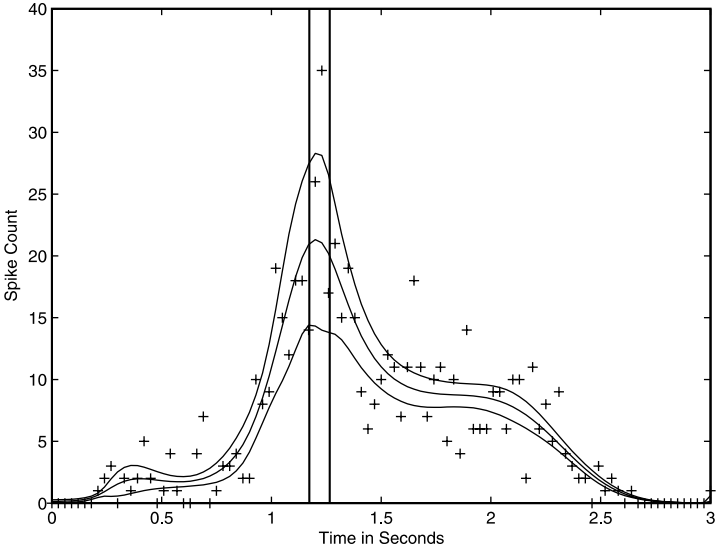


Figure 2. Monkey neuron spike counts, denoted by + symbols, in successive 30 millisecond intervals, along with (dashed lines) posterior mean of rates exp(r(t)), plus/minus 1.96 times the square root of posterior variance of the fitted values, for one realization of an adaptive chain, with 1,000 tours and a total length of 131,560. Vertical lines show the estimated 95% posterior interval for arg max r(x), rescaled to the [0, 3] time scale of the original data.

$$r(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 (x - \xi_1)_+^3 + \beta_4 (x - \xi_2)_+^3 + \beta_5 (x - \xi_3)_+^3 + \beta_6 (x - \xi_4)_+^3, \quad x \in [0, 1],$$

where $(x)_+ = \max(x, 0)$. The knot positions $\xi_1, \ldots, \xi_4 \in [0, 1]$ and the coefficients $\beta_0, \ldots, \beta_6$ are not known.

We adopt a Bayesian approach, assigning a Dirichlet$(3, 3, 3, 3, 3)$ prior distribution to the gaps between the knots, that is, to the vector $(\xi_1, \xi_2 - \xi_1, \xi_3 - \xi_2, \xi_4 - \xi_3, 1 - \xi_4)^T$. The coefficients β_j are assigned (the relatively uninformative) independent normal priors with mean zero and variance 10^{12}. The likelihood for the model is easily computed from (4.2). Our objective is to determine the posterior distribution of the "function of interest" $h(\cdot)$ consisting of the 100 fitted values $r(k/99),\ k = 0, \ldots, 99$, as well as the squares of these values, arg max $r(\cdot)$, and [arg max $r(\cdot)$]^2. Keeping track of the squares enables us to compute posterior standard deviations for $r(k/99),\ k = 0, \ldots, 99$ and arg max $r(\cdot)$.

In order to apply Algorithm 2, we first construct a transition kernel for a chain whose limiting distribution is the posterior distribution of the parameter vector $(\xi_1, \ldots, \xi_4, \beta_0, \ldots, \beta_6)$. We build a hybrid kernel, which consists of Metropolis-Hastings kernels P_1, P_2, and P_3, with proposals which can be summarized as follows. Let $\hat{\beta}(\xi_1, \ldots, \xi_4)$ denote the maximum likelihood estimate of the coefficient vector $(\beta_0, \ldots, \beta_6)$ given the knot positions $\xi_1, \ldots, \xi_4$, and let $\Sigma_\beta(\xi_1, \ldots, \xi_4)$ denote the estimated covariance matrix of $\hat{\beta}(\xi_1, \ldots, \xi_4)$, plus a small constant times the identity matrix. (These are straightforward to obtain, since conditioned on knot positions, the model (4.2) is simply a generalized linear model.) The

proposal for $P_1(\cdot,\cdot)$ is generated by adding a Dirichlet(30, 30, 30, 30, 30) random variable, minus the vector (.2, .2, .2, .2, .2), to the vector of knot gaps. Conditioned on the proposed knot position $\xi^* = (\xi_1^*, \ldots, \xi_4^*)$, the proposed coefficients are drawn from a multivariate normal distribution with mean $\hat{\beta}(\xi^*)$ and covariance matrix $\Sigma_\beta(\xi^*)$. For P_2, the proposal is generated by leaving the knots alone and choosing an entirely new set of coefficients, in the same manner as for $P_1(\cdot,\cdot)$. For P_3, the (random walk) proposal is generated by leaving the knot positions alone and moving each of the coefficients a random distance proportional to the square root of the corresponding element of the covariance matrix $\Sigma_\beta(\xi_1, \ldots, \xi_4)$.

The hybrid kernel consisting of successive application of kernels P_1, P_2, and P_3 generates a Markov chain on $\mathbb{R}^{11}$ whose limiting distribution is the posterior distribution of the parameter vector. As in the previous examples, in order to identify regeneration points for our chain, we embed this hybrid kernel into Algorithm 1, and then into Algorithm 2.

For this problem, the re-entry proposals are generated by first drawing the knots gaps from a Dirichlet(3, 3, 3, 3, 3) distribution, and then conditioning on those knot positions, drawing a coefficient vector from the multivariate normal as described for kernels P_1, P_2, and P_3 above. Using this re-entry distribution, the constant k is determined using the same procedure as in the previous example.

To construct an adaptive version of this algorithm, we use the approach described in Section 3.4, constructing a mixture normal approximation to the set of knot positions and gradually replacing the knot-shifting proposals in P_1 with independence proposals drawn from the mixture approximation. Adaptation parameters are chosen to be $\kappa = .1$ and $\zeta = .95$.

Figure 2 shows posterior fitted values, and posterior fitted values plus and minus 1.96 times their estimated standard deviations, obtained using an adaptive chain of 1,000 tours, with a total length of 131,560 iterations. Vertical lines are shown at the estimate of $\arg\max r(\cdot)$ plus and minus 1.96 times its estimated standard deviation.

To measure performance of the adaptive chain relative to the nonadaptive chain, we generate 100 independent chains using each method, each consisting of 2,000 tours, with average chain length approximately 180,000. The SPPI for $\arg\max r(\cdot)$ is computed for each chain, and boxplots of the results are given in Figure 3. For this example, the gains in SPPI are again noticeable. Again, it is informative to consider the distribution of tour-lengths in both the adaptive and nonadaptive cases. Histograms and discussion are given in the online supplemental document.

Both the nonadaptive and adaptive chains perform an order of magnitude better than the original hybrid kernel $P_3 \circ P_2 \circ P_1$. Generating a chain of length 80,000,000 using the original kernel and using the batch means approach with batch sizes 1,000,000, we obtain a SPPI of approximately .737, which is over 100 times worse than those obtained using the regenerative scheme. In this case, it appears to be largely due to the fact that the posterior distribution for knot positions is bimodal, and the original chain has great difficulty making the transition between modes, while introduction of the Dirichlet proposals for knot positions in the re-entry distribution gets around this problem. (Note that as discussed in Section 5 of this article, if the re-entry distribution is chosen to cover only one of the modes, the variance of the tour-length distribution can become unacceptably large.)

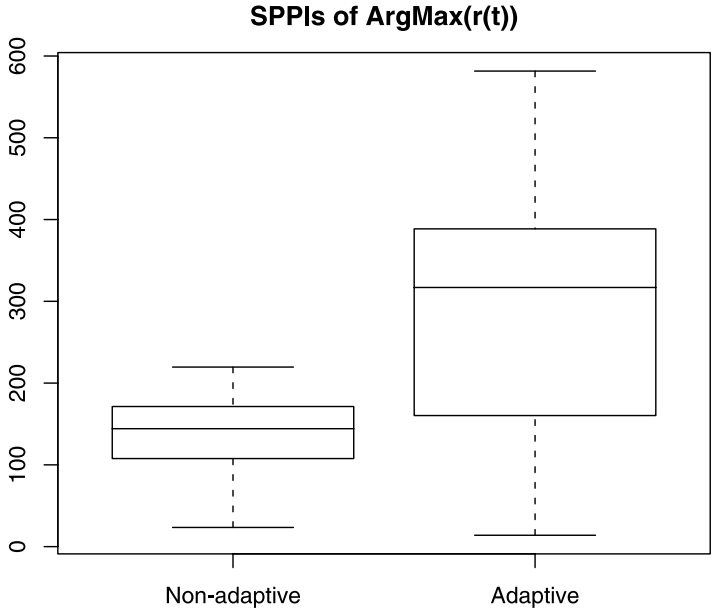


Figure 3. Boxplots of sample precision per iteration (SPPI) for the posterior mean of arg max $r(\cdot)$ in the free-knot spline problem, for both nonadaptive and adaptive chains generated using Algorithm 2. SPPIs are evaluated for 100 nonadaptive and 100 adaptive chains, each consisting of exactly 2,000 tours, with length approximately equal to 180,000.

5. DISCUSSION

Regeneration in Markov chains is useful because it can be used to avoid the burn-in problem, to obtain good estimates of the variance of MCMC estimators, to introduce indefinite adaptation into a chain, and to use parallel processors to construct a single long chain with a specified target distribution.

We have described a new way to think about identification of regeneration times in a Markov chain and demonstrated potential computational gains, measured by the sample precision per iteration, obtained by using regeneration to introduce adaptive behavior into MCMC simulation. Of course, improved SPPI through the use of adaptive schemes is not the only reason to use the methods described here. As discussed in the introduction and pointed out previously by others, two benefits, which come about simply as results of the use of regenerative simulation, are that the issue of burn-in can be avoided and that consistent estimates of the variances of estimators are easily obtained.

Although we have not given explicit algorithms for parallel generation of a single Markov chain in this article, the concept is simple. Because tours of a regenerative chain are independent of each other, one can generate tours concurrently on separate processors and patch them together. There are, however, some important considerations when adaptive schemes are used. These are discussed briefly in the online supplemental document.

One issue which we have not considered in detail in this article is the manner in

which the chain makes transitions from the state α to the remainder of the state-space, or, in the context of Algorithm 2, selection of the re-entry distribution ϕ and the constant k. In theory, the methods work as long as the chain can, with positive probability, jump from α to any set of positive π-measure, but choice of the transition kernel from α affects the tour-length distribution. Although the dugong example exhibited a certain amount of robustness to poor choice of ϕ, at a certain point (for instance, in the dugong example, if the re-entry distribution is approximately the target distribution, shifted by 10 standard deviations) Algorithm 2 becomes virtually unusable, as it becomes highly likely that an extremely long tour will be generated.

A common problem in MCMC simulation is that of getting trapped in one mode of a multimodal distribution. If the original kernel has this problem, and Algorithm 2 is used, but ϕ only captures one mode of a multimodal distribution, then the probability of a jump to α becomes very small when the chain enters one of the other modes. This leads to a situation where one sees many small tours restricted to the main mode, and an occasional very long tour which enters another mode, then has to return to the original mode before it has a significant probability of returning to α. In such a case, the danger is that without generating enough tours, one might never observe one of the very long tours. A further danger in this situation is that many diagnostics fail. The SPPI measure used in Section 4, for instance, could give overly optimistic assessments of performance if only one mode was explored, since the variance would appear smaller. On the other hand, use of a very widely spread re-entry proposal distribution in the algorithms in this article can improve a chain's ability to make transition between modes, with the state α functioning as a "conduit" (as in the free-knot spline example of this article). One potentially useful way to addressing this multimodality problem more generally could be to embed the simulated tempering kernels discussed by Geyer and Thompson (1995) into the algorithms described in this article. Because the "hot" level of the simulated tempering kernel is designed to reduce the effect of multimodality, one could expect this to potentially reduce the associated tour-length distribution problems in the context of the algorithms discussed in this article.

ACKNOWLEDGMENTS

The authors are grateful to Brad Carlin, Chris Genovese, Steve Knox, Antonietta Mira, Peter Müller, Geoff Nicholls, Mark Schervish, Mike Steele, Luke Tierney, and Valerie Ventura, and two anonymous referees for their comments and suggestions. This research was supported in part by the National Science Foundation under grants DMS-9819950, DMS-9801401, and IIS-0083148.

[Received March 2003. Revised June 2004.]

REFERENCES

Carlin, B. P., and Gelfand, A. E. (1991), "An Iterative Monte Carlo Method for Nonconjugate Bayesian Analysis," *Statistics and Computing*, 1, 119–128.

Chauveau, D., and Vandekerkhove, P. (2002), "Improving Convergence of the Hastings-Metropolis Algorithm With an Adaptive Proposal," *Scandinavian Journal of Statistics*, 29, 13–29.

Chien, C., Goldsman, D., and Melamed, B. (1997), "Large-Sample Results for Batch Means," *Management Science*, 43, 1288–1295.

Crane, M. A., and Iglehart, D. L. (1975a), "Simulating Stable Stochasting Systems, I: General Multi-Server Queues," *Journal of the Association of Computing Machinery*, 21, 103–113.

——— (1975b), "Simulating Stable Stochasting Systems, II: Markov Chains," *Journal of the Association of Computing Machinery*, 21, 114–123.

Crane, M. A., and Lemoine, A. J. (1977), *An Introduction to the Regenerative Method for Simulation Analysis*, volume 4 of *Lecture Notes in Control and Information Sciences*, New York: Springer.

Gelfand, A. E., and Sahu, S. K. (1994), "On Markov Chain Monte Carlo Acceleration," *Journal of Computational and Graphical Statistics*, 3, 261–276.

Geyer, C. (1992), "Practical Markov Chain Monte Carlo," *Statistical Science*, 7, 473–483.

Geyer, C. J., and Thompson, E. A. (1995), "Annealing Markov Chain Monte Carlo with Applications to Ancestral Inference," *Journal of the American Statistical Association*, 90, 909–920.

Gilks, R., Roberts, G. O., and Sahu, S. K. (1998), "Adaptive Markov Chain Monte Carlo Through Regeneration," *Journal of the American Statistical Association*, 93, 1045–1054.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996), *Markov Chain Monte Carlo in Practice*, Boca Raton, FL: CRC Press.

Gilks, W. R., Roberts, G. O., and George, E. I. (1994), "Adaptive Direction Sampling," *The Statistician*, 43, 179–189.

Glynn, P. W., and Iglehart, D. L. (1990), "Simulation Output Analysis Using Standardized Time Series," *Mathematics of Operations Research*, 15, 1–16.

Hobert, J. P., Jones, G. L., Presnell, B., and Rosenthal, J. S. (2002), "On the Applicability of Regenerative Simulation in Markov Chain Monte Carlo," *Biometrika*, 89, 731–744.

Jones, G. L., and Hobert, J. P. (2001), "Honest Exploration of Intractable Probability Distributions via Markov Chain Monte Carlo," *Statistical Science*, 16, 312–334.

McLachlan, G., and Peel, D. (2000), *Finite Mixture Models*, New York: Wiley.

Meyn, S. P., and Tweedie, R. L. (1993), *Markov Chains and Stochastic Stability*, New York: Springer.

Møller, J., and Nicholls, G. K. (2005), "Perfect Simulation for Sample-Based Inference," *Statistics and Computing*, conditionally accepted.

Mykland, P., Tierney, L., and Yu, B. (1995), "Regeneration in Markov Chain Samplers," *Journal of the American Statistical Association*, 90, 233–241.

Nummelin, E. (1978), "A Splitting Technique for Harris Recurrent Markov Chains," *Zeitschrift für Wahrscheinlichkeitstheorie und Vervandte Gebiete*, 43, 309–318.

——— (1984), *General Irreducible Markov Chains and Non-Negative Operators*. Cambridge: Cambridge University Press.

Ratkowsky, D. A. (1983), *Nonlinear Regression Modeling*, New York: Dekker.

Revuz, D. (1975), *Markov Chains*, Amsterdam: North-Holland.

Ripley, B. D. (1987), *Stochastic Simulation*, New York: Wiley.

Robert, C. P. (1995), "Convergence Control Methods for Markov Chain Monte Carlo Algorithms," *Statistical Science*, 10, 230–253.

Robert, C. P., and Casella, G. (1999), *Monte Carlo Statistical Methods*, New York: Springer.

Roberts, G. O. (1996), "Markov Chain Concepts Related to Sampling Algorithms," in *Markov Chain Monte Carlo in Practice*, Boca Raton, FL: CRC Press, pp. 45–57.

Tierney, L. (1994), "Markov Chains for Exploring Posterior Distributions," *The Annals of Statistics*, 22, 1701–1728.

Tierney, L., and Mira, A. (1999), "Some Adaptive Monte Carlo Methods for Inference," *Statistics in Medicine*, 18, 2507–2515.

Titterington, D. M. (1984), "Recursive Parameter Estimation Using Incomplete Data," *Journal of the Royal Statistical Society*, Ser. B, 46, 257–267.

Ventura, V., Carta, R., Kass, R. E., Olson, C. R., and Gettner, S. N. (2002), "Statistical Analysis of Temporal Evolution in Single-Neuron Firing Rates," *Biostatistics*, 1, 1–20.

Warnes, G. (2000), *The Normal Kernel Coupler: An Adaptive Markov Chain Monte Carlo Method for Efficiently Sampling from Multi-modal Distributions*, PhD thesis, University of Washington.