Exact Sampling from a Continuous State Space

D. J. Murdoch* P. J. Green†

September 18, 1997

Abstract

Propp and Wilson (1996) described a protocol, called *coupling from the past*, for exact sampling from a target distribution using a coupled Markov chain Monte Carlo algorithm. In this paper we extend coupling from the past to various MCMC samplers on a continuous state space; rather than following the monotone sampling device of Propp and Wilson, our approach uses methods related to gamma-coupling and rejection sampling to simulate the chain, and direct accounting of sample paths.

Keywords: coupling from the past, gamma coupling, Gibbs sampling, Markov chain Monte Carlo, Metropolis-Hastings, perfect simulation, rejection sampling.

1 Introduction

Propp and Wilson (1996; 1997) described a protocol called "coupling from the past" (CFTP) for exact sampling from a target distribution using a coupled Markov chain Monte Carlo (MCMC) algorithm. The essence of their idea

*Department of Mathematics and Statistics, Queen's University, Kingston, Ontario, Canada, K7L 3N6. This work was done while visiting the University of Bristol.
Email: dmurdoch@mast.queensu.ca.

†Department of Mathematics, University of Bristol, Bristol, BS8 1TW, UK.
Email: P.J.Green@bristol.ac.uk.

was to consider running an ergodic simulation of a Markov chain using a sequence of random numbers ("coins") with a known distribution. If the simulation had been running indefinitely, then the observation at time 0 would be distributed exactly according to the stationary distribution π of the chain. "If we can figure out the state at time 0 by looking at a finite number of these coins used in the recent past, then the result is an unbiased sample." (Propp and Wilson, 1997).

The structure of algorithms based on this idea is as follows. We consider runs of length M ending at time 0, for some arbitrary value of $M > 0$, starting at every possible state of the chain at time $t = -M$. Using a sequence of random numbers, we follow the paths from each of these states forward in time; occasionally two paths will merge ("coalesce"), and eventually all of the paths will have coalesced into one. If this has happened when we reach time 0, then we are done. If there are still multiple possible states, we choose a larger value of M and start again, using the *same realization* of the sequence of random numbers.

For their detailed implementations, Propp and Wilson (1996) made this process particularly efficient by working with simulations with a monotonicity property, allowing them to simulate only paths from the minimal and maximal states; once they had coalesced, those from all intermediate states must have coalesced as well.

The present paper addresses the question of extending coupling from the past to deal with (typically multivariate) *continuous* target distributions, motivated by potential applications to Bayesian parametric inference. The main difficulty in applying the Propp and Wilson versions of monotone coupling from the past to general continuous state spaces is that, for most implementations, paths from different starting values will not coalesce in finite time, though they may become arbitrarily close. Thus the best that can be achieved with such methods is an approximation to the steady-state distribution.

Reutter and Johnson (1995) used the idea of coupled MCMC paths to assess the rate of convergence of an MCMC simulation, rather than to create an exactly-sampled realisation. Their Markov-maximal coupling and mixture coupling are similar to our multigamma coupling (described below), but are applied to a finite collection of chains evolving into the future, while our methods are applied to uncountable collections evolving from the past to the present. Cai (1997) used CFTP together with monotonicity to derive a coupling algorithm for an independence sampler, and derived the rate of convergence to the target distribution when the state space was finite. In a paper on some theoretical underpinnings of CFTP, Foss and Tweedie (1997) show the importance of uniform ergodicity in verifying successful backward coupling for CFTP.

One special context in which coupling from the past has been developed for problems involving continuously-distributed random variables is that of spatial point processes. Several authors have developed such algorithms by careful construction of monotonic sampling functions which can be viewed as introducing discreteness into the continuous state space (Häggström, Lieshout and Møller, 1997; Kendall, 1996; Kendall, 1997). In this area, and more generally, there is currently intense research activity, and this brief literature review is necessarily immediately out-of-date. An up-to-date annotated bibliography is maintained by David Wilson at
`http://dimacs.rutgers.edu/~dbwilson/exact.html`

In this paper we show in section 2 that many common MCMC algorithms can be recast in a form in which large collections of states are all updated to a single new state. This provides the necessary discretisation of the state space, and the CFTP algorithm may then be used to obtain exact samples from the limiting distribution. In section 3 we apply some of these algorithms to an example of a hierarchical Bayes model, and we discuss the methods in section 4.

2 Algorithms

We will describe several approaches for simulating realisations of ergodic Markov chains (X_t) that couple from the past; in principle, these work in rather general state spaces. For simplicity and definiteness here, however, we assume that the state space is $\chi \subset \mathcal{R}^d$ and that the transition kernel of interest and its stationary distribution have densities with respect to d-dimensional Lebesgue measure. We write the density of X_{t+1} at y, given $X_t = x$ as $f(y|x)$ and that of the stationary distribution as $\pi(\cdot)$, so that for example

$$\int \pi(x) f(y|x) dx = \pi(y).$$

In Subsection 2.4 the assumption of the existence of f is relaxed slightly to allow us to consider Metropolis-Hastings samplers, which may have positive probability of not moving at each transition.

Any program for simulating a Markov chain takes the current state X_t, generates some random numbers U_{t+1}, independent of all other U_s and of X_t, and forms the new state X_{t+1} as a deterministic *update function*

$$X_{t+1} = \phi(X_t, U_{t+1}), \tag{1}$$

designed to achieve the correct transition densities $f(\cdot|\cdot)$. For example, the standard "random walk Metropolis" algorithm can be expressed via

$$\phi(X, U) = \begin{cases} X + U^{(2)} & \text{if } U^{(1)} < \pi(X + U^{(2)})/\pi(X) \\ X & \text{otherwise} \end{cases}$$

Here $U^{(1)}$ is a $U(0,1)$ random variable and $U^{(2)}$ is a random increment of the same dimension as X with a distribution symmetric about 0, both independent of X.

For normal purposes, the existence of this function ϕ is implicit, but in studying coupling from the past, it plays an explicit and crucial role. The algorithms we will consider generate parallel paths from different initial states, and may "re-play" segments of the paths from different initial times. It is conceptually convenient to imagine the U_t as generated once and for all, for all t, at the start of the simulation, although in practice we will maintain data structures that allow us to generate each U_t by the time it is first needed. This must be done in such a way that (1) remains literally true, and this typically requires careful housekeeping. We must ensure, for example, that if $X_t = x$ leads to $X_{t+1} = y$ when following one path then it does so in all paths regardless of the order in which the various possible transitions are actually simulated.

Coupling from the past is based on the observation that if there exists a (random) initial time $-T < 0$, such that for all initial states X_{-T}, X_0 is the same, then X_0 has density π; this is Propp and Wilson's Theorem 2. Equation (1) provides a representation that allows us to check the values of T and X_0, and the secret of any practical implementation of coupling from the past is a method of tracking paths to do all the checking required efficiently (and in particular, in finite time!) There are many possible representations (1) for a particular transition kernel; we are searching for one of these that generates highly dependent paths, with high probabilities of coalescing.

All of the algorithms presented in this paper have the same general two-stage implementation. In the interests of clarity and compactness, we present them in a pseudo-code, whose syntax should be self-explanatory. We start by re-expressing the general CFTP algorithm as follows, using the representation (1):

```
CFTP(M):
    t ← −M
    B_t ← χ
    while B_t infinite and t < 0
        t ← t + 1
        B_t ← set containing φ(B_{t−1}, U_t)
    while t < 0
        t ← t + 1
        B_t ← φ(B_{t−1}, U_t)
    if #B_0 = 1 then
        return(B_0)
    else
        CFTP(2M)
```

Once a suitable function ϕ satisfying (1) and following the recipe described above has been specified, the proof that if and when this algorithm exits, it produces an exact sample from the target distribution is identical to the proof of Theorem 3 in Propp and Wilson (1996), rewritten for a continuous state space. As for *whether* it exits (i.e. the coupling is successful), this relies on choice of an update function ϕ, and the design of the code for "$B_t \leftarrow$ `set containing` $\phi(B_{t-1}, U_t)$". This question is addressed in the detailed implementations that follow.

It may appear that dividing the algorithm into the two stages is artificial; we could replace the test for finite B_t in the first loop with a test that it contains a single value. However, the implementations of the algorithm that we give below all have this two-stage character, because representing an infinite set is quite different from representing a finite set.

When a pass through the algorithm fails to reduce to a single state and we choose a new value of M, our algorithm always doubles the previous choice.

As Propp and Wilson (1996) show, this is not necessary; any increase in M would work. However, doubling is close to the optimal choice if our aim is to minimize the total running time of the algorithm.

In the remainder of this section we describe implementations of the CFTP algorithm that may be used in different Markov chain simulations.

2.1 The Multigamma Coupler

This algorithm exploits ideas related to, but different from, Doeblin's small sets (Meyn and Tweedie, 1993), splitting (Nummelin, 1984), gamma coupling (Lindvall, 1992) and Markov-maximal coupling (Reutter and Johnson, 1995). The first three of these are devices introduced to facilitate theoretical arguments; the last provides an algorithm used in convergence diagnosis. Though typically not fast enough to be used alone, multigamma coupling serves as a component of or a motivation for several other couplers.

We suppose here that we know explicitly the update kernel density $f(\cdot|x)$ of the Markov chain, the density from which X_{t+1} is sampled given $X_t = x$. Furthermore, we suppose that

$$f(y|x) \geq r(y) \qquad \forall x, y \in \chi \tag{2}$$

for some nonnegative function $r(\cdot)$, for which $\rho = \int r(y)dy > 0$. For notational convenience we restrict the discussion to one dimensional χ.

To define what we term the multigamma coupler, write

$$\begin{aligned} R(y) &= \rho^{-1} \int_{-\infty}^{y} r(v)dv, \\ Q(y|x) &= (1-\rho)^{-1} \int_{-\infty}^{y} [f(v|x) - r(v)]dv. \end{aligned}$$

The update function used by the algorithm is

$$\phi(x, U) = \begin{cases} R^{-1}(U^{(2)}) & \text{if } U^{(1)} < \rho \\ Q^{-1}(U^{(2)}|x) & \text{otherwise ,} \end{cases}$$

using a pair $U = (U^{(1)}, U^{(2)})$ of Uniform random numbers.

It is easy to see that $P(X_{t+1} \leq y | X_t = x) = \rho R(y) + (1-\rho)Q(y|x)$, so that $\phi(x, U)$ is indeed a draw from $f(\cdot|x)$.

The multigamma coupler is defined by the following algorithm:

```
MultiGamma(M):
    t ← −M
    B_t ← χ
    while B_t infinite and t < 0
        t ← t + 1
        if U_t^(1) < ρ then
            B_t ← {R^{-1}(U_t^(2))}
        else
            B_t ← χ
    while t < 0
        t ← t + 1
        if U_t^(1) < ρ then
            B_t ← {R^{-1}(U_t^(2))}
        else
            B_t ← {Q^{-1}(U_t^(2)|x)} where B_{t−1} = {x}
    if #B_0 = 1 then
        return(B_0)
    else
        MultiGamma(2M)
```

Having explicitly demonstrated the form of the update function ϕ, the proof that this is a valid implementation of the CFTP algorithm depends only on showing that B_t almost surely becomes finite in finite time. But this is immediate: in this coupler, at each time, B_t collapses to a single element with probability ρ.

If the inverse distribution functions R^{-1} and Q^{-1} are not available, the algorithm must be modified to replace the references to them by calls to subprograms that produce draws from R or Q respectively. For example, if $r(\cdot)$ is bounded above by $Kh(\cdot)$, where h is a density from which we can simulate directly, we can use rejection:

```
InverseR(U_t^(2)):
    set random number seed using U_t^(2)
```

```
draw auxiliary random numbers V_1, V_2, ...
j ← 1
repeat
    draw Y from h using V_j
    if V_{j+1} < r(Y)/[Kh(Y)] then
        exit
    j ← j + 2
return(Y)
```

An alternative, purely constructive, implementation of the multigamma coupler is as follows. Since there is a fixed probability ρ that R^{-1} is called at any given time t, we can choose the most recent such occurrence before $t = 0$ by sampling a random variable T from a Geometric distribution with success probability ρ: $P(T = t) = \rho(1 - \rho)^t, t = 0, 1, 2, \ldots$, and sampling the value X_{-T} at this time from the density $r(\cdot)/\rho$. This allows us to avoid the search over M in the outer loop of the general CFTP algorithm. We carry X_t forward in time using $X_t \leftarrow Q^{-1}(U_t^{(2)}|X_{t-1})$, and obtain $X_0 \sim \pi(\cdot)$ exactly. The validity of this version of the algorithm may be proved directly, either analytically or probabilistically, without coupling arguments.

The multigamma coupler as described here will have a very limited range of application, for two reasons. First, the lower bound function $r(\cdot)$ may be too small for practical use: note that the coupler takes an average of $1/\rho$ transitions to couple. In many practical applications the only lower bound is $r(\cdot) \equiv 0$, and multigamma coupling cannot be used at all. Second, it requires the user to know the normalized update densities. Often (for example in hierarchical Bayes modelling) the update densities are only known up to a constant of proportionality.

2.2 The Partitioned Multigamma Coupler

While it is often difficult or impossible to find a uniform non-zero lower bound on the update densities as in (2), quite commonly the state space can be partitioned into a finite collection of disjoint cells $\mathcal{A} = \{A_i, i = 1, \ldots, m\}$, and separate lower bounds can be placed on the densities within each cell. That is,

$$f(y|x) \geq r_i(y) \qquad \forall x \in A_i, \quad \forall y \in \chi.$$

We further assume that $\int r_i(y)dy = \rho > 0$ is constant over all cells; this assumption will be relaxed in a later section.

The partitioned multigamma coupler is then a modification of the multigamma coupler:

PartitionedMultiGamma(M):
    $t \leftarrow -M$
    $B_t \leftarrow \chi$
    while B_t infinite and $t < 0$
        $t \leftarrow t+1$
        if $U_t^{(1)} < \rho$ then
            $B_t \leftarrow \{R_i^{-1}(U_t^{(2)}) : i = 1, 2, \ldots, m\}$
        else
            $B_t \leftarrow \chi$
    while $t < 0$
        $t \leftarrow t+1$
        if $U_t^{(1)} < \rho$ then
            $B_t \leftarrow \{R_i^{-1}(U_t^{(2)}) : i = 1, 2, \ldots, m, A_i \cap B_{t-1} \neq \emptyset\}$
        else
            $B_t \leftarrow \cup_{i=1}^m \{Q_i^{-1}(U_t^{(2)}|x) : x \in A_i \cap B_{t-1}\}$
    if $\#B_0 = 1$ then
        return(B_0)
    else
        PartitionedMultiGamma($2M$)

In this algorithm, R_i and Q_i are defined by

$$\begin{aligned} R_i(y) &= \rho^{-1} \int_{-\infty}^{y} r_i(v)dv, \\ Q_i(y|x) &= (1-\rho)^{-1} \int_{-\infty}^{y} [f(v|x) - r_i(v)]dv, \end{aligned}$$

and the update function used is

$$\phi(x, U) = \begin{cases} R_i^{-1}(U^{(2)}) & \text{if } U^{(1)} < \rho \\ Q_i^{-1}(U^{(2)}|x) & \text{otherwise} \end{cases},$$

for $x \in A_i$. We see that $P(X_{t+1} \le y | X_t = x) = \rho R_i(y) + (1-\rho)Q_i(y|x)$ is a proper mixture of distributions when $x \in A_i$, so that $\phi(x, U)$ is again a draw from $f(\cdot|x)$. As in the previous algorithm, if an inverse distribution function is not available, an appropriate alternative needs to be provided, for example a rejection sampler.

Proof of the correctness of this algorithm includes checking that the sets of candidate states in the second loop do indeed shrink to a single state in a finite time. This follows from the irreducibility of the original chain, which implies that the cells of the partition inter-communicate.

As a simple illustration of the partitioned multigamma coupler, consider the following toy example on the one-dimensional state space $\chi = [0, 1)$. For the update transition kernel, we take

$$f(y|x) = (1+\omega)[\min\{y/x, (1-y)/(1-x)\}]^{\omega}$$

for a fixed parameter $\omega \ge 0$. Increasing ω reduces the overlap between $f(y|x)$ for different x, and so makes coalescence harder; in our simulation we took $\omega = 6$. We partitioned χ into 8 equal intervals $A_i = [(i-1)/8, i/8)$, and took $r_i(y) = \min_{x \in A_i} f(y|x)$; we find $\int r_i(y)dy = \rho = (8/9)^{\omega}$, independent of i as required. The function $R_i^{-1}(\cdot)$ can be computed explicitly, but we use a numerical binary search to calculate $Q_i^{-1}(U^{(2)}|x)$. In Figure 1 we display results from a single run of the coupler; coupling from the past was successful with $M = 16$, and B_t became finite at $t = -14$ and reduced to a single state at $t = 0$. The non-monotonic nature of the update function ϕ is clearly seen.

2.3 The Rejection Coupler

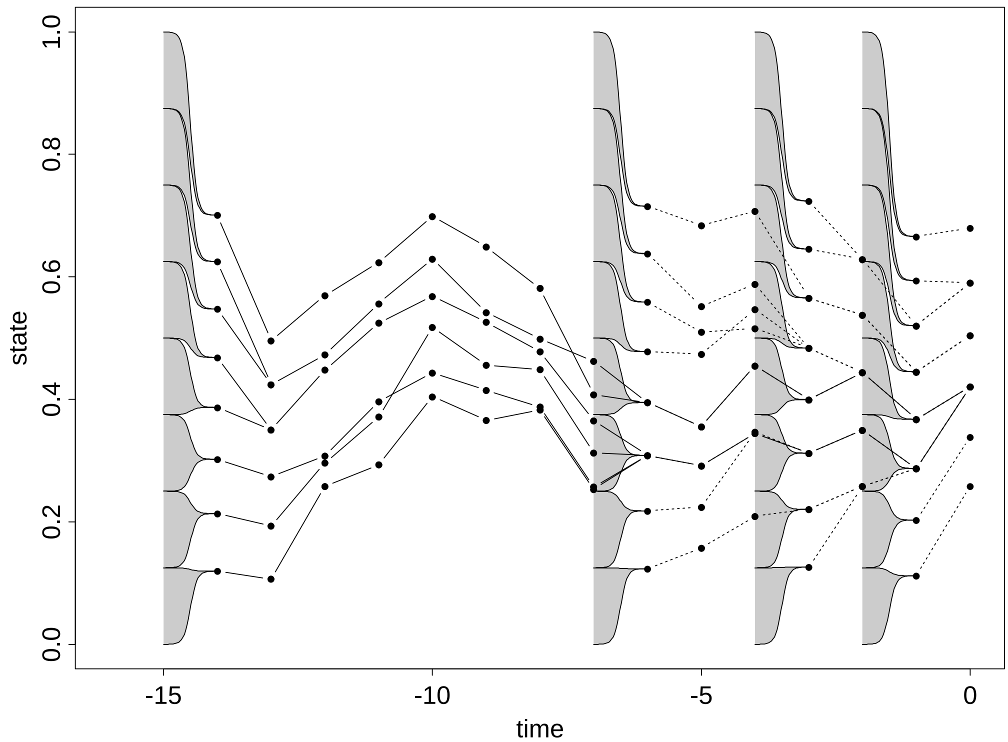
In this section we provide an alternative basic recipe for coupling from the past, that has the attraction of applicability to certain problems where the updating densities are known only up to a constant of proportionality, a situation rather common in Gibbs sampling.

We suppose that the conditional update kernel is $f(y|x) = k(x)g(y|x)$ where the un-normalized density $g(y|x)$ is known, but $k(x)$ is unknown. We also assume that there exists an envelope function $h(y)$ such that

$$g(y|x) \le h(y) \qquad \forall x, y \in \chi, \tag{3}$$

i.e. the un-normalized densities are bounded above uniformly in x. We assume $\nu = \int h(y)dy < \infty$, quite a demanding condition in an unbounded

Figure 1: A run of the partitioned multigamma coupler on the toy example. The solid line texture corresponds to paths from $-\infty$ coalescing by time 0.



state space. Finally, we assume that we have a way of sampling from the density $h(y)/\nu$.

The algorithm is similar to the partitioned multigamma coupler, but the partition is determined randomly during each update. Specifically, we imagine using a rejection sampler to sample from $f(y|X_{t-1})$ using $h(y)/\nu$ as a proposal density, and partitioning the space of X_{t-1} values into those that would be accepted on each attempt. The detailed layout of the algorithm is:

```
RawRejectionCoupler(M):
    t ← −M
    set random number seed using U_t
    j ← 0
    repeat
        j ← j + 1
        draw Y_j from h(·)/ν
        draw V_j from U(0,1)
        A_j ← {x : g(Y_j|x) ≥ V_j h(Y_j)}
        if ∪_{i=1}^{j} A_i = χ then
            J ← j
            exit
    B_t ← {Y_1, Y_2, ..., Y_J}
    while t < 0
        t ← t + 1
        set random number seed using U_t
        j ← 0
        repeat
            j ← j + 1
            draw Y_j from h(·)/ν
            draw V_j from U(0,1)
            A_j ← {x ∈ B_{t−1} : g(Y_j|x) ≥ V_j h(Y_j)}
            if ∪_{i=1}^{j} A_i = B_{t−1} then
                J ← j
                exit
        B_t ← {Y_1, Y_2, ..., Y_J}
    if #B_0 = 1 then
        return(B_0)
```

```
else
    RawRejectionCoupler(2M)
```

Success at attaining a finite partition almost surely, and hence coupling from the past, is a subtle matter depending on the form of $k(x)g(y|x)$: it is necessary but not sufficient that $k(x)$ be bounded.

We apply this algorithm to the same toy example used in the previous section; for this transition density, the sets A_j are easily computed explicitly; they are all intervals, so it is straightforward to accumulate their union, and to check whether they cover specified values. The results of a single run are displayed in Figure 2. As necessarily occurs with rejection coupling, B_{-M+1} is finite for every M, and in this case has only two elements on the successful pass, with $M = 16$, on which coalescence occurred at $t = -5$. The unsuccessful passes with $M = 2, 4$ and 8 produced $\#B_0 = 7, 6$ and 2 respectively.

In real problems, it is sometimes awkward to keep track of the $\{A_j\}$ explicitly. When there is additionally a *lower* envelope $r(y)$ for the functions $g(y|x)$, satisfying

$$0 \leq r(y) \leq g(y|x)$$

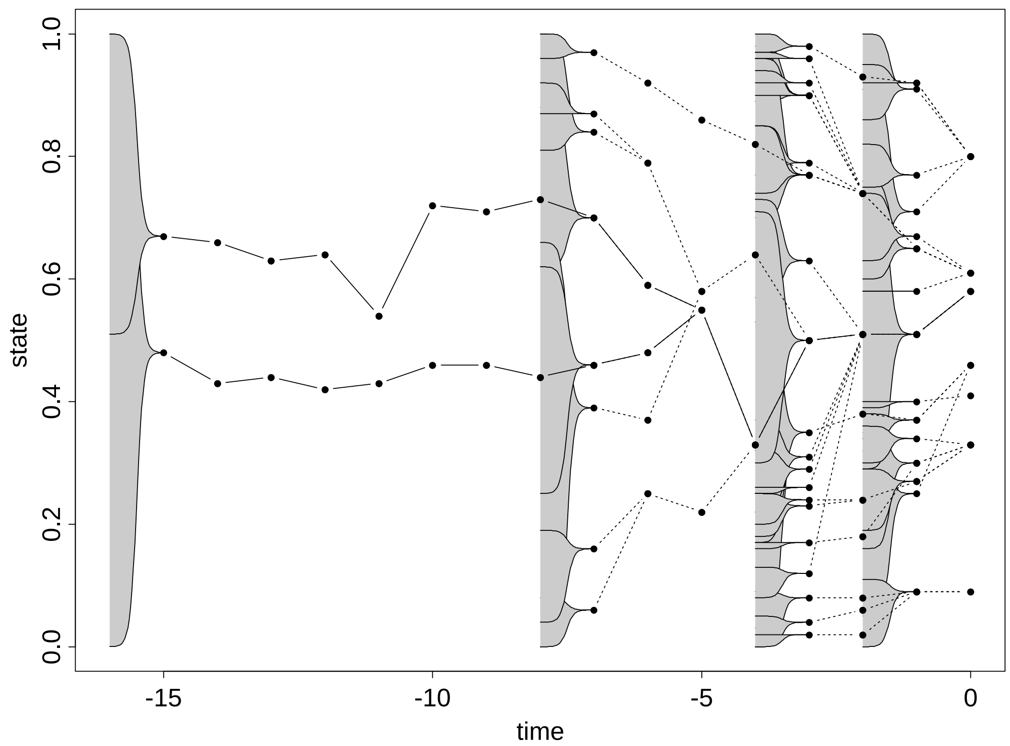
with $\rho = \int r(y)dy / \int h(y)dy > 0$, we can get round this difficulty by continuing to repeat the inner loop until $V_j < r(Y_j)/h(Y_j)$. Exit from the loop occurs almost surely, since $\rho > 0$, and then there is certain to be at least one Y_j satisfying $V_j < g(Y_j|x)/h(Y_j)$, so we have updated all states, and implicitly created a partition of χ into J cells $A_j = C_j \backslash A_{j-1}$, $j = 1, \ldots, J$, where

$$\begin{aligned} A_0 &= \emptyset \\ C_j &= \{x | V_j < g(Y_j|x)/h(Y_j)\}. \end{aligned}$$

This is our standard implementation of the rejection coupler.

We can further modify the rejection coupler by partitioning the state space in advance as in the partitioned multigamma coupler. In this variation, separate upper and lower bounds are used on each partition. We assume that $0 \leq r_i(y) \leq g(y|x) \leq h_i(y)$ for all $x \in A_i$ and $y \in \chi$, with $\nu_i = \int h_i(y)dy < \infty$ and $\rho_i = \int r_i(y)dy/\nu_i > 0$ for $i = 1, \ldots, m$. In contrast to the partitioned multigamma case, there is no need for ρ_i to be the same across partition cells.

Figure 2: A run of the raw rejection coupler on the toy example. The solid line texture corresponds to paths from $-\infty$ coalescing by time 0.



```
PartitionedRejectionCoupler(M):
    t ← −M
    for i = 1, ..., m
        set random number seed using U_t
        j ← 0
        repeat
            j ← j + 1
            draw Y_ij from h_i(·)/ν_i
            draw V_ij from U(0,1)
            if V_ij < r_i(Y_ij)/h_i(Y_ij) then
                J_i ← j
                exit repeat
    B_t ← {Y_ij : i = 1, ..., m, j = 1, ..., J_i}
    while t < 0
        t ← t + 1
        for i = 1, ..., m
            set random number seed using U_t
            j ← 0
            repeat
                j ← j + 1
                draw Y_ij from h_i(·)/ν_i
                draw V_ij from U(0,1)
                if V_ij < r_i(Y_ij)/h_i(Y_ij) then
                    J_i ← j
                    exit repeat
        B_t ← ∅
        for x ∈ B_{t−1}
            i ← cell index such that x ∈ A_i
            B_t ← B_t ∪ {first Y_ij in (Y_i1, ..., Y_iJ_i) : V_ij < g(Y_ij|x)/h(Y_ij)}
    if #B_0 = 1 then
        return(B_0)
    else
        PartitionedRejectionCoupler(2M)
```

To decide whether or not to partition, we might aim to minimize the size of the candidate list at the end of the first stage. The expected number of items in the list in the partitioned algorithm is $\sum 1/\rho_i$. As will be seen in the example below, a careful choice of partition can make this many orders of magnitude smaller than the $1/\rho$ items expected with the simple algorithm.

2.4 The Metropolis-Hastings Coupler

In this section we show how to adapt a class of Metropolis-Hastings samplers to allow coupling from the past.

We suppose that the target density $\pi(\cdot)$ is known up to a constant of proportionality, and that we use a proposal distribution with density $q(y|x)$.

The familiar Metropolis-Hastings algorithm that updates all variables at once is implemented as follows. We assume that at time $t-1$ we are in state X_{t-1}, and wish to generate X_t. We sample values $U_t^{(1)} \sim U(0,1)$ and $U_t^{(2)}$ from some fixed continuous distribution; a function $\psi(\cdot,\cdot)$ can then be constructed such that $\psi(x, U_t^{(2)})$ has density $q(y|x)$. It is important to note that while the distribution $Y_t = \psi(X_{t-1}, U_t^{(2)})$ may depend on X_{t-1}, the distribution of $U_t^{(2)}$ does not.

Then

$$X_t = \begin{cases} Y_t & \text{if } U_t^{(1)} < \alpha(Y_t|X_{t-1}) \\ X_{t-1} & \text{otherwise} \end{cases} \tag{4}$$

where

$$\alpha(Y|X) = \frac{\pi(Y)q(X|Y)}{\pi(X)q(Y|X)}. \tag{5}$$

For our implementation of CFTP in this context, we require that the proposal distribution depends on the current state only in a discrete way, so that the function $\psi(\cdot, U_t^{(2)})$ takes on only a finite list of m possible values when $U_t^{(2)}$ is held fixed. For example, for the "independence sampler", $q(y|x)$ does not depend on x, so that ψ takes on only one value; the general case could be seen as a separate independence sampler on each cell of a finite partition of the state space.

We further assume that for given values of $U_t^{(1)}$ and $U_t^{(2)}$ we can determine a set C_t containing all states which would not be updated, i.e.

$$\{x : U_t^{(1)} \geq \alpha[\psi(x, U_t^{(2)})|x]\} \subset C_t$$

and that, with probability 1 there exists T such that $\cap_{t=1}^{T} C_t$ is empty. In the description below we split B_t into C_t and a finite set F_t:

`MetropolisHastingsCoupler(`M`):`
    $t \leftarrow -M$
    $F_t \leftarrow \emptyset$
    $C_t \leftarrow \chi$
    `while` $t < 0$ `and` $C_t \neq \emptyset$
        $t \leftarrow t + 1$
        $C_t \leftarrow$ `set containing` $\{x \in C_{t-1} : U_t^{(1)} > \alpha[\psi(x, U_t^{(2)})|x]\}$
        $F_t \leftarrow \psi(C_{t-1}, U_t^{(2)})$
        $F_t \leftarrow F_t \cup \{\psi(x, U_t^{(2)}) : x \in F_{t-1}$ `and` $U_t^{(1)} \leq \alpha[\psi(x, U_t^{(2)})|x]\}$
        $F_t \leftarrow F_t \cup \{x : x \in F_{t-1}$ `and` $U_t^{(1)} > \alpha[\psi(x, U_t^{(2)})|x]\}$
    `while` $t < 0$
        $t \leftarrow t + 1$
        $F_t \leftarrow \{\psi(x, U_t^{(2)}) : x \in F_{t-1}$ `and` $U_t^{(1)} \leq \alpha[\psi(x, U_t^{(2)})|x]\}$
        $F_t \leftarrow F_t \cup \{x : x \in F_{t-1}$ `and` $U_t^{(1)} > \alpha[\psi(x, U_t^{(2)})|x]\}$
    `if` $\#F_0 = 1$ `then`
        `return(`F_0`)`
    `else`
        `MetropolisHastingsCoupler(`$2M$`)`

The main difficulty in applying this algorithm is in calculating the sets C_t. Unless there is a special relationship between the proposal distribution and the target distribution, characterization of the sets of states which would be updated at each step can be an intractable problem. We now present an important special case where this calculation is straightforward.

Consider the independence sampler, where $q(y|x) = q(y)$ and $\psi(x, U) = \psi(U)$. Suppose that

$$\sup \frac{\pi(x)}{q(x)} = K < \infty.$$

Then α in (5) satisfies

$$\alpha(Y|X) \geq \frac{\pi(Y)}{q(Y)K}.$$

It follows that if $U_t^{(1)} < \pi(Y_t)/[q(Y_t)K]$, *all* paths will couple simultaneously. In fact, if we set $C_t = \emptyset$ when this occurs and otherwise set it to the whole space, then we simply have a rejection sampler with proposal density $Kq(\cdot)$, and the value X_t when C_t becomes empty will itself be an exact sample from $\pi(\cdot)$.

In many cases such a rejection sampler will not exist, or will be too slow to be of practical use (though to our surprise, this was not the case in the example we present below). However, we will often be able to find a finite set of relatively efficient proposal distributions $q_i(\cdot)$, $i = 1, \ldots, m$, such that $\pi(x)/q_i(x) \leq K_{ij}$ for $x \in A_j$, where $\mathcal{A} = \{A_j\}$ is a partition of the sample space, and at least one K_{ij} is finite for each j. If we then use the proposal distribution $q(Y|X_{t-1}) = q_i(Y)$ when $X_{t-1} \in A_i$, we see that when $Y \in A_j$, $U_t^{(1)} < \pi(Y)/[q_i(Y)K_{ji}]$ implies all paths with $X_{t-1} \in A_i$ will couple simultaneously. We use this result as the basis of the following coupling algorithm, expressed in terms of I_t instead of C_t; the relation is $C_t = \cup_{i \in I_t} A_i$.

`BoundedMHCoupler(`M`):`

- $t \leftarrow -M$
- $F_t \leftarrow \emptyset$
- $I_t \leftarrow \{1, \ldots, m\}$
- `while` $t < 0$ `and` $I_t \neq \emptyset$
  - $t \leftarrow t + 1$
  - `for` $i = 1 \ldots m$
    - $Y_i \leftarrow \psi_i(U^{(2)})$
    - $j_i \leftarrow$ `cell index such that` $Y_i \in A_{j_i}$
  - $I_t \leftarrow \{i \in I_{t-1} : U_t^{(1)} > \pi(Y_i)/[q_i(Y_i)K_{j_i i}]\}$
  - $F_t \leftarrow \{Y_i : i \in I_{t-1}\}$
  - $F_t \leftarrow F_t \cup \{Y_i : \exists x \in F_{t-1} \cap A_i$ `and` $U_t^{(1)} \leq [\pi(Y_i)/q_i(Y_i)]/[\pi(x)/q_{j_i}(x)]\}$
  - $F_t \leftarrow F_t \cup \{x : x \in F_{t-1} \cap A_i$ `and` $U_t^{(1)} > [\pi(Y_i)/q_i(Y_i)]/[\pi(x)/q_{j_i}(x)]\}$
- `while` $t < 0$
  - $t \leftarrow t + 1$
  - `for` $i = 1 \ldots m$
    - $Y_i \leftarrow \psi_i(U^{(2)})$
    - $j_i \leftarrow$ `cell index such that` $Y_i \in A_{j_i}$
  - $F_t \leftarrow \{Y_i : \exists x \in F_{t-1} \cap A_i$ `and` $U_t^{(1)} \leq [\pi(Y_i)/q_i(Y_i)]/[\pi(x)/q_{j_i}(x)]\}$

$F_t \leftarrow F_t \cup \{x : x \in F_{t-1} \cap A_i$ `and` $U_t^{(1)} > [\pi(Y_i)/q_i(Y_i)]/[\pi(x)/q_{j_i}(x)]\}$

```
if #F_0 = 1 then
    return(F_0)
else
    BoundedMHCoupler(2M)
```

2.5 Cyclic Couplers

Many MCMC algorithms update components of a state vector sequentially. For example, if the state X_t is a vector $(X_{t1}, \ldots, X_{tp})$, the cyclic Gibbs sampler updates X_{ti} conditional on the current values of the other components.

It is possible to construct componentwise couplers, using the algorithms of the previous sections (or others) as building blocks. In general, to reduce B_t to a finite list will require that all p component couplers achieve finite candidate lists in sequence, and the overall list may comprise all combinations of values from the p lists. However, there are many special cases which may make coupling faster or more manageable. For example, if the update of component i depends only on the values of component j, then once component j has only a finite list of possibilities component i will automatically inherit a list of the same or shorter length. Use is made of this in a two component Gibbs sampler in the next section.

3 Example: hierarchical Poisson/gamma model

In this section we construct algorithms to sample exactly from the posterior distribution of the parameters in a model fit to a dataset on pump reliability. We take our description of the data from Reutter and Johnson (1995), and use their formulation of the model. The model and data were originally presented by Gelfand and Smith (1990). These data have also been analysed by others (Gaver and O'Muircheartaigh, 1987; Rosenthal, 1995).

The dataset records the counts $\mathbf{s} = (s_1, \ldots, s_{10})$ of failures in ten pump systems at a nuclear power plant. The times of operation $\mathbf{t} = (t_1, \ldots, t_{10})$ for each system are also known. The model is a hierarchical model taking advantage of conjugate priors. First,

$$s_k \sim \text{Poisson}(\lambda_k t_k)$$

is assumed, where λ_k is the failure rate for the k^{th} pump. The prior densities for λ_k are

$$\lambda_k \sim \Gamma(\alpha, \beta)$$

where the shape parameter $\alpha = 1.802$ was determined by a method of moments argument. The inverse scale β has the relatively diffuse prior

$$\beta \sim \Gamma(\gamma, \delta)$$

where $\gamma = 0.01$ and $\delta = 1$. The values for α, γ and δ are those used by Reutter and Johnson (1995).

The full conditional distributions for this model are

$$\lambda_k | \beta, \mathbf{s} \sim \Gamma(\alpha + s_k, \beta + t_k) \tag{6}$$

and

$$\beta | \boldsymbol{\lambda}, \mathbf{s} \sim \Gamma(\gamma + 10\alpha, \delta + \sum \lambda_k) \tag{7}$$

where $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_{10})$.

3.1 Bounds on gamma densities

The full conditional distributions in this model are Gamma distributions with known shape and unknown inverse scale parameters. In this section we derive lower bound functions $r(y)$ for families of distributions of this type.

Let $X \sim \Gamma(a, b)$ and suppose a is fixed, and $b \in [b_0, b_1]$. Then the density for X is

$$f(x; a, b) = x^{a-1} b^a \exp(-xb)/\Gamma(a)$$

and we have the uniform lower bound

$$\begin{aligned} r(x) &= x^{a-1} b_0^a \exp(-x b_1)/\Gamma(a) \\ &\leq f(x; a, b) \end{aligned} \tag{8}$$

Since $r(x)$ is a rescaled $\Gamma(a, b_1)$ density, we see that $\rho = \int r(x)dx = (b_0/b_1)^a$.

We may choose to use the unnormalized algorithm rather than the normalized one. For that we could use $r(x)$ and the upper bound

$$\begin{aligned} h^\dagger(x) &= x^{a-1} b_1^a \exp(-x b_0)/\Gamma(a) \\ &\geq f(x; a, b) \end{aligned}$$

giving $\rho^\dagger = [\int r(x)dx]/[\int h^\dagger(x)dx] = (b_0/b_1)^{2a}$, but it is better to do away with the normalizing constants and bound the kernels $[\Gamma(a)/b^a]f(x;a,b) = x^{a-1}\exp(-xb)$:

$$\begin{aligned} r^\star(x) &= x^{a-1}\exp(-xb_1) \quad (9)\\ &\le x^{a-1}\exp(-xb)\\ &\le h^\star(x)\\ &= x^{a-1}\exp(-xb_0) \quad (10)\end{aligned}$$

With this choice $\rho^\star = [\int r^\star(x)dx]/[\int h^\star(x)dx] = (b_0/b_1)^a$ as in the normalized case.

3.2 Application to the pump data

There are 11 unknown parameters in the pump data model, but a two component Gibbs sampler may be used to sample them: β is chosen conditional on $\boldsymbol{\lambda}$, and then $\boldsymbol{\lambda}$ is chosen conditional on β. As described in section 2.5, the coupling algorithms of this paper may be applied to one or both parameters in this situation.

For the pump model, an immediate problem arises. Because β may be arbitrarily large, we cannot find a lower bound for the λ_k densities, and because $\sum \lambda_k$ may be arbitrarily large, we cannot find a lower bound for the β density. This problem is common to many models where the parameter space is unbounded, and a simple solution is to make a small modification to the prior distribution to introduce a bound on the parameters.

For example, we may change our prior by imposing the restriction $\sum \lambda_k < L$. If L is chosen so large that the prior and posterior probabilities of exceeding it are very small in the original formulation of the model, then this will be a negligible change. The effect of it on the full conditionals will be to place the same restriction on the distribution of $\boldsymbol{\lambda}|\beta, \mathbf{s}$, but it will make no change to the $\beta|\boldsymbol{\lambda}, \mathbf{s}$ conditional.

In this modified model, the range of update kernels for β will then be Gamma distributions with shape $\gamma + 10\alpha$ and inverse scale in the range $[\delta, \delta + L]$, and we can find a uniform lower bound $r(x)$ as described above with $\rho = [\delta/(\delta + L)]^{\gamma+10\alpha}$. Unfortunately, this ρ is too small to be useful. A 10,000 step run of the Gibbs sampler gave $\sum \lambda_k$ values up to 13.6, so we would want L to be much higher to avoid changing the model significantly.

But even at $L = 20$, $\rho \approx 10^{-24}$, and we would need on average about 10^{24} steps in our simulation to achieve coupling.

We could also investigate modifications to the prior where β is bounded, but these tend to be even worse, since the lower bound on the product of the λ_k densities has ρ equal to the product of the ρ values for the bounds on each component.

The solution to this problem is to use the partitioned algorithms as described in sections 2.2 and 2.3. If we partition the space of $(\beta, \boldsymbol{\lambda})$ values into regions according to the value of $\sum \lambda_k$, then we can put separate lower bounds on each cell of the partition. Specifically, set $0 = L_0 < L_1 < \ldots < L_m = L$ and use partition cells $A_i = \{(\beta, \boldsymbol{\lambda}) | L_{i-1} < \sum \lambda_k \leq L_i\}$, $i = 1, \ldots, m$.

As shown at the end of section 2.3, the unnormalized algorithm will have the smallest number of update candidates if we minimize $S = \sum 1/\rho_i$. Here

$$S = \sum_i [(\delta + L_i)/(\delta + L_{i-1})]^{(\gamma+10\alpha)}.$$

A straightforward calculation shows that this is minimized when m is the integer part of $(\gamma + 10\alpha)\ln[(\delta + L)/\delta]$ or the next greater integer. The ratios $(\delta + L_i)/(\delta + L_{i-1})$ are all equal to $[(\delta + L)/\delta]^{1/m}$. For large L, the ratios tend to $\exp[1/(\gamma + 10\alpha)] \approx 1.057$ and $\rho_i \to 1/e$. In the simulations described below, we used the limiting value for the ratio rather than the slightly better value that depends on L; this gives $S \approx e(\gamma + 10\alpha)\ln[(\delta + L)/\delta]$, so even very large values of L are quite feasible. In the multigamma simulation below we used $L = 10^6$, giving $m = 249$ and $S \approx 677$.

We are now in a position to describe sampling using the partitioned multigamma coupler for β as a component of a cyclic Gibbs coupler. We partition χ on the value of $\sum \lambda_k$ as described above. Within partition cell A_i we have the lower bound (8) with $b_0 = L_{i-1} = \delta \exp[(i-1)/(\gamma + 10\alpha)]$ and $b_1 = L_i$ and we may draw $R_i^{-1}(U_t^{(2)})$ by drawing β from $\Gamma(\gamma + 10\alpha, L_i)$. The drawing of $Q_i^{-1}(U_t^{(2)} | \beta, \boldsymbol{\lambda})$ is implemented using a rejection sampler with the full conditional (7) as the proposal distribution.

Implementation of the partitioned rejection coupler could use the same partition, bounds (9) and (10) within cell A_i, and draws from $h_i^\star(\cdot)/\nu_i$ as above. However, in the numerical simulations we took advantage of some special structure of this model as described in the next section.

3.3 Removing the restriction

To this point, the methods of construction of the exact sampling algorithms have followed an approach that could be expected to apply to a wide variety of models. In this section we make use of the special form of the present model to improve the partitioned rejection coupler and remove the restriction $\sum \lambda_k < L$.

Because the update distribution (7) depends on the previous state only through the scale of the distribution, when we reuse the same random seed U_t to update each separate cell, we can arrange that the pattern of β_{ij} values (where i denotes the cell and j denotes the draw number, as in the second algorithm of section 2.3) and their associated $\boldsymbol{\lambda}$ values are identical in each cell except for scale changes. We can use this to extend our finite partition to an infinite partition covering the whole positive range of possible $\sum \lambda_k$ values as follows.

We first note that as L_i increases, the sampled values β_{ij} will decrease to a limit of 0. Furthermore, smaller β_{ij} values will lead to larger simulated λ_k values. To avoid confusion we will write these as $\lambda_{k,ij}$ to emphasize the connection to β_{ij}.

Thus if we know the values of β_{1j}, then we can place an upper bound on the values of $\lambda_{k,ij}$ that holds for all i: it is simply $\lambda_{k,1j}(t_k + \beta_{1j})/t_k$. This allows us to conclude that

$$\sum_k \lambda_{k,ij} \leq \sum_k \lambda_{k,1j}(t_k + \beta_{1j})/t_k$$

uniformly over all i. The result is that we do not need a fixed bound L on the sum; each iteration will produce a separate random bound which we can use to the same purpose.

To incorporate this into our partitioned rejection coupler we proceed as follows. Insert these new steps before the first loop, just after $t \leftarrow -M$:

$j \leftarrow 0$
`repeat`
    $j \leftarrow j + 1$
    `draw` β_{1j} `from` $\Gamma(\gamma + 10\alpha, L_1)$
    `draw` V_{1j} `from` $U(0,1)$
    `if` $V_{1j} < r_1^\star(\beta_{1j})/h_1^\star(\beta_{1j})$ `then`

```
        J_1 ← j
        exit
for j = 1,...,J_1, k = 1,...,10
    draw λ_{k,1j} from Γ(α + s_k, β_{1j} + t_k)
L ← max_j Σ_k λ_{k,1j}(t_k + β_{1j})/t_k
m ← int (γ + 10α) ln[(δ + L)/δ]
t ← t + 1
```

A similar adjustment can also be made to the partitioned multigamma coupler.

This adjustment is very reminiscent of Propp and Wilson's (1996) monotonicity property with states ordered by the value of $\sum \lambda_k$, though as far as we know only the simple multigamma coupler of Subsection 2.1 may be made monotone with respect to this ordering. (Cai (1997) showed that the independence coupler is always monotone with respect to a different ordering.) Møller (1997) used the same technique in a large class of exponential family models, and Kendall (1996) used similar ideas in simulating point processes.

3.4 The Metropolis-Hastings coupler for the pump data

In this section we return to more generally applicable methods and describe the construction of couplers based on independence samplers of the full posterior distribution $k\pi(\beta, \boldsymbol{\lambda})$ (where k is a constant of integration). We have

$$\log \pi(\beta, \boldsymbol{\lambda}) = (10\alpha+\gamma-1)\log\beta-\delta\beta+\sum_k\{(s_k+\alpha-1)\log\lambda_k-(\beta+t_k)\lambda_k\}. \quad (11)$$

Our first thought was to use a multivariate normal (MVN) proposal distribution. By matching moments to the results of a run of the Gelfand and Smith (1990) Gibbs sampler for the model, we were able to develop an independence sampler with a better than 60% acceptance rate. However, we could not use this in the Metropolis-Hastings coupler. The tails of the MVN distribution are much lighter than the tails of the posterior (11), so the ratio $\pi(\beta, \boldsymbol{\lambda})/q(\beta, \boldsymbol{\lambda})$ is unbounded.

Instead, we chose to use a gamma proposal for β, and used the exact conditional distributions (6) for $\boldsymbol{\lambda}$. Specifically, we used $\beta \sim \Gamma(a, b)$ giving

$$\log \frac{\pi(\beta, \boldsymbol{\lambda})}{q(\beta, \boldsymbol{\lambda})} \quad = \quad (10\alpha + \gamma - a)\log\beta - (\delta - b)\beta - a\log b + \log\Gamma(a)$$

$$-\sum_k \{(s_k+\alpha)\log(t_k+\beta) - \log\Gamma(\alpha+s_k)\} \tag{12}$$

It can be seen that (12) will be bounded provided that the following inequalities both hold:

$$\begin{aligned} a &\leq 10\alpha + \gamma && (13) \\ b &\leq \delta && (14) \end{aligned}$$

and a numerical search may be used to find the bound. We chose values of a and b such that the mean of the distribution of β matched the sample mean of our Gibbs simulation, and the variance was as close as possible to the sample variance, subject to the restrictions above.

As described below, this is an extraordinarily good rejection sampler; nevertheless, for the purposes of illustration, we sought to improve on it by partitioning the sample space. Of the two restrictions above, only (14) affected the choice of proposal distribution, and it had a substantial effect, reducing b from the moment-matching estimate of 4.8 to 1, so that the proposal distribution had 4.8 times larger variance than the target distribution. Examination of (12) shows that if β is bounded above, (14) becomes unnecessary, so we partitioned the sample space according to whether or not $\beta > \beta^\star$ for various values of $\beta^\star$. When $\beta_{t-1} \leq \beta^\star$ we used a gamma proposal subject only to (13), and otherwise used the same proposal as for the independence sampler.

3.5 Numerical examples

Figures 3 and 4 show the results of a single run of each of the partitioned multigamma and partitioned rejection coupling algorithms. The partitioned rejection coupler made use of the modification described in Subsection 3.3. In each case, the random number seeds were started at the same values at each time step.

The multigamma coupler (Figure 3) had coalescence events at times $t = -14, -12, -11, -9, -8, -7, -2$ and -1. Four of these are clearly visible, as they triggered the change of B_t to a finite set. Paths ending at the same value at time $t = 0$ are shown in the same texture. For example, it can be seen that the $M = 8$ paths, which first appear at $t = -7$, end at just two

Figure 3: A single run of the multigamma coupler on the Pump dataset. Different line textures correspond to different values for the path at time 0.

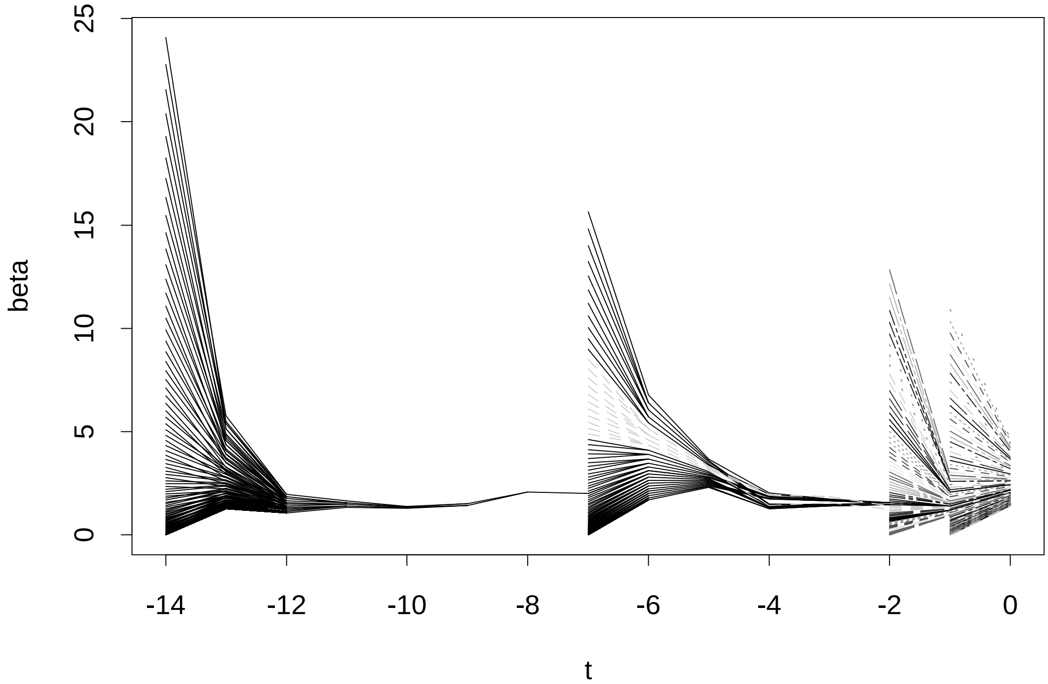
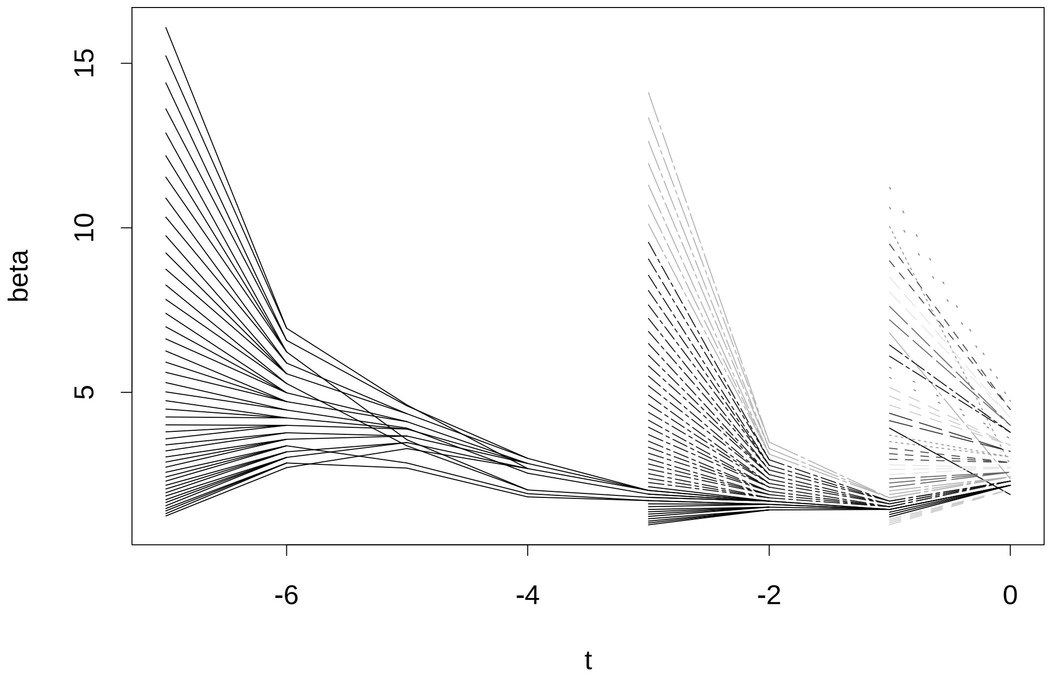


Figure 4: A single run of the rejection coupler on the Pump dataset. Different line textures correspond to different values for the path at time 0.



different values at $t = 0$, while the $M = 16$ runs all terminate at the same value.

The rejection coupler (Figure 4) required running from $t = -8$ in order that all states would coalesce; also shown in the figure are the paths from the other attempted starting positions. (The paths are not shown during the single step of the first stage.)

We ran the Metropolis-Hastings samplers for 10,000 steps to determine their acceptance rates. The Metropolis-Hastings independence sampler had an acceptance rate of approximately 52%, and paths coalesced on approximately 43% of the steps. Using a two-cell Metropolis-Hastings sampler allowed a marked improvement in the Metropolis-Hastings sampler: for example, with $\beta^\star = 4$, its acceptance rate rose to 97%. However, it did not result in a better coupler than the one based on the independence sampler, because values of β greater than 4 coupled only about 6% of the time.

Overall, these couplers ran very quickly. To measure the running times, we performed 1000 independent runs of each coupler. The *total* time taken for all runs of all four couplers was about 8 minutes on a 120 MHz Pentium PC. We compared the separate algorithms by counting the calls to the basic uniform random number generator; this provides a machine independent measure, and a rough measure of the relative machine dependent times. The Metropolis-Hastings independence coupler (i.e. the rejection sampler) was by far the simplest and fastest, requiring an average of 82 calls to the basic uniform random number generator for each vector generated from $\pi(\beta, \boldsymbol{\lambda})$. (Each generated gamma variate required several calls.) The two-cell Metropolis-Hastings, rejection and multigamma couplers required on average 3800, 31000, and 63000 calls respectively. These numbers could be reduced; our implementation reset the random number generator seed and made repeated calls rather than storing and re-using the random numbers themselves. However, the pump model is a particularly tractable and rapidly mixing example of MCMC, and we would not expect such fast execution times in most applications.

4 Conclusions

Propp and Wilson's (1996) CFTP algorithm is an exciting development, and we expect it to make a large impact on the way MCMC simulations are carried out. It allows the "burn-in" stage of a simulation to be done exactly, and repeated application of the algorithm can be used to generate i.i.d. samples from the target distribution.

Of course, CFTP is not yet a panacea. We have attempted here to extend the idea to various samplers for continuous state spaces, with a view particularly towards the needs of parametric Bayesian inference. Compared to a straightforward application of Gibbs sampling or a Metropolis-Hastings method, there is an overhead cost: modest in terms of programming and running time, but sometimes quite considerable in terms of the cumbersome algebraic manipulations that are needed by our implementations.

There is a great deal of flexibility in specifying a CFTP algorithm. Even with Foss and Tweedie's (1997) limitations on the usable Markov chains, there may be uncountably many with stationary distribution $\pi(\cdot)$, and for each chain uncountably many update functions $\phi(\cdot,\cdot)$ which satisfy the conditions of Propp and Wilson's (1996) Theorem 3. The practical task is to find one which couples reasonably rapidly and for which the CFTP "bookkeeping" is feasible.

Of the methods we introduce, the Metropolis-Hastings independence coupler had the best performance in our example, but it is equivalent to a well-known rejection sampler (Gilks, 1996). Nevertheless, we do not believe that this phenomenon will be typical. We have used CFTP to develop several new exact samplers, but we have by no means exhausted the possibilities. Combinations of our couplers (e.g. a rejection coupler on the proposal distribution for a random walk Metropolis-Hastings coupler), applications of them in other contexts (e.g. Reutter and Johnson's (1995) couplers run from the past instead of the present, to give approximate samplers without stopping time biases), and other unrelated coupling algorithms all seem likely future developments, and we anticipate that future work will find further statistical application for CFTP.

Acknowledgments

We are grateful to Steve Brooks for introducing us to gamma coupling and to Wilfrid Kendall, Richard Tweedie and David Wilson for many helpful comments. This research was supported in part by an NSERC Research Grant and a Queen's University sabbatical award to the first author, and by an EPSRC research grant to the second.

References

Cai, H. (1997). A note on an exact sampling algorithm and Metropolis-Hastings Markov chains. Unpublished.

Foss, S. G. and Tweedie, R. L. (1997). Perfect simulation and backward coupling. To appear in *Stochastic Models*.

Gaver, D. and O'Muircheartaigh, I. (1987). Robust empirical Bayes analysis of event rates. *Technometrics*, 29:1–15.

Gelfand, A. E. and Smith, A. F. M. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85:398–409.

Gilks, W. R. (1996). Full conditional distributions. In Gilks, W. R., Richardson, S., and Spiegelhalter, D. J., editors, *Markov Chain Monte Carlo in Practice*, pages 75–88. Chapman and Hall.

Häggström, O., Lieshout, M. N. M., and Møller, J. (1997). Characterisation results and Markov chain Monte Carlo algorithms including exact simulation for some spatial point processes. Technical Report 1997-03, Department of Mathematics, Aalborg University.

Kendall, W. (1996). Perfect simulation for the area-interaction point process. Technical Report 292, Department of Statistics, University of Warwick. To appear in *Probability Perspective*, C. C. Heyde and L. Accardi, eds.

Kendall, W. S. (1997). On some weighted Boolean models. In *Advances in Theory and Applications of Random Sets*, pages 105–120. World Scientific Publishing Company, Singapore.

Lindvall, T. (1992). *Lectures on the Coupling Method*. Wiley.

Meyn, S. P. and Tweedie, R. L. (1993). *Markov Chains and Stochastic Stability*. Springer-Verlag.

Møller, J. (1997). A note on perfect simulation of conditionally specified models. Technical Report R-97-2006, Department of Mathematics, Aalborg University.

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

Propp, J. G. and Wilson, D. B. (1996). Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9:223–252.

Propp, J. G. and Wilson, D. B. (1997). How to get a perfectly random sample from a generic Markov chain and generate a random spanning tree of a directed graph. *Journal of Algorithms*, To appear.

Reutter, A. and Johnson, V. (1995). General strategies for assessing convergence of MCMC algorithms using coupled sample paths. Technical Report 95-25, Institute of Statistics and Decision Sciences, Duke University.

Rosenthal, J. S. (1995). Minorization conditions and convergence rates for Markov chain Monte Carlo. *Journal of the American Statistical Association*, 90:558–566.

Cai (1997), Foss and Tweedie (1997), Häggström *et al.* (1997) and Møller (1997) are available from the MCMC preprint service at uniform resource locator `http://www.stats.bris.ac.uk/MCMC/`.