MPS-RR 1999-43

ISSN 1398-599X

Perfect Slice Samplers

A. Mira[1], J. Møller[2] and G.O. Roberts[3]
[1]Università dell'Insubria, Varese, Italy
[2]Aalborg University, Denmark
[3]Lancaster Univerity, U.K.

Abstract

Perfect sampling allows exact simulation of random variables from the stationary measure of a Markov chain. By exploiting monotonicity properties of the slice sampler we show that a perfect version of the algorithm can be easily implemented, at least when the target distribution is bounded. Various extensions, including perfect product slice samplers, and examples of applications are discussed.
Keywords: Auto-models; Auxiliary variables; Coupling from the past (CFTP); Ising model; Perfect simulation; Random fields; Slice sampler; Spatial point processes; Swendsen-Wang algorithm.

1 Introduction

In this paper we combine two quite powerful ideas that have recently appeared in the Markov chain Monte Carlo (MCMC) literature: the slice sampler and perfect simulation.

The slice sampler is a method of constructing a reversible Markov transition kernel with a specified invariant distribution. Auxiliary variables are introduced within the independence Metropolis-Hastings algorithm to ensure that the acceptance probability is always equal to one. The idea of introducing auxiliary variables in MCMC sampling arose in statistical physics [37], was generalised by [9], and brought into the mainstream statistical literature by Besag and Green in 1993 [2]. The *simple slice sampler* is a special case of the class of auxiliary variable methods, where a single auxiliary variable

is introduced, and can be though of as a Gibbs sampling algorithm beneath the graph of the target density.

Perfect simulation is a way of running a Markov chain which ensures that the terminal value of the implementation is an exact draw from the stationary distribution of the chain. The idea was introduced by Propp and Wilson in 1996 [31]. Since then, the research area has become extremely active, see for example [6, 10, 13, 15, 16, 18, 17, 19, 20, 25, 27, 28, 29, 38, 39]. The present paper is an extension and revision of [24]. Perfect slice samplers for mixtures of distributions have recently been studied in [5].

The paper is organised as follows. Section 2 contains a brief description of the simple slice sampler, together with a discussion of its monotonicity properties expressed in terms of a certain density ordering. Section 3 surveys perfect simulation based on stochastic recursive sequences. Section 4 is concerned with an explicit stochastic recursive sequence for the simple slice sampler, which can easily be used for perfect simple slice sampling when there exist a maximal and minimal state with respect to the density ordering. Extensions to cases where there is no maximal or minimal state are discussed in Section 5. Various (mainly spatial) examples and further extensions using more than one auxiliary variables are studied in Section 6. Section 7 contains some concluding remarks.

2 Slice Sampler

Suppose $\pi(x)$, $x \in \mathcal{X}$ is an unnormalised integrable density with respect to the measure μ and let ν_π be the corresponding probability measure:

$$\nu_\pi(A) = \frac{\int_A \pi(x)\mu(dx)}{\int_{\mathcal{X}} \pi(x)\mu(dx)}$$

for all measurable A. The simple slice sampler is an auxiliary variables simulation method that provides information about $\pi(x)$ by running a Markov chain that has ν_π as its unique stationary distribution.

The key idea behind the simple slice sampler is to introduce a latent or auxiliary variable, $u \in \mathcal{U}$, and to construct the joint distribution of u and x by taking the marginal distribution for x unchanged and defining the conditional distribution of u given x in a convenient way. In particular we take $\mathcal{U} = (0, \infty)$ and specify the conditional distribution of $u|x$ to be uniform over the interval $(0, \pi(x))$.

An irreducible and aperiodic Markov chain is then set up over the enlarged state space $\mathcal{X} \times \mathcal{U}$ having the probability measure corresponding to $\pi(x,u) \propto I_{\{u<\pi(x)\}}(x,u)$, as its unique stationary distribution, where $I_A(x)$ is the indicator function of the set A. In particular it is customary to implement the Markov chain by iteratively performing Gibbs updates on x and u:

- vertical update: $u|x$ is sampled uniformly over the interval $(0, \pi(x))$;
- horizontal update: $x|u$ is sampled from the normalisation of the restriction of μ to the set

$$A_\pi(u) = \{x : \pi(x) > u\}. \tag{1}$$

The marginal chain $\{X_n\}_{n=0}^\infty$ is easily seen to have ν_π as its stationary distribution, therefore, after a "sufficiently" long burn-in time, it can be used to estimate the integral of a function g with respect to π, by taking the average of the values that g takes over sample paths. Since at the end we are only interested in the marginal x-chain we will focus our attention on it and will regard the u-chain $\{U_n\}_{n=0}^\infty$ as an auxiliary construction.

In most applications considered in this paper, $\mathcal{X}$ is a subset of $\mathbf{R}^d$ and μ is absolutely continuous with respect to the Lebesgue or counting measure restricted to $\mathcal{X}$. Let $Q_\pi(u) = \mu[A_\pi(u)]$; if not strictly necessary we will omit the subscript π in $A_\pi(u)$ and $Q_\pi(u)$. Then $Q(u) \leq 1/u$ for any $u > 0$, and we can assume that, with probability one, $Q(U_n) > 0$ at each iteration of the u-chain. Hence at the horizontal update considered above, $x|u$ is sampled from $P_u(A) = \mu(A \cap A(u))/\mu(A(u))$ where $0 < \mu(A(u)) < \infty$. In the applications to be considered in Section 6, P_u is often a uniform distribution over $A(u)$, so, for simplicity, we shall in general write the vertical and horizontal updates as

$$u|x \sim U[(0, \pi(x))], \quad x|u \sim U[A(u)]$$

(even if the latter distribution is specified with reference to a non-uniform measure μ).

In the case where $\mathcal{X} = \mathbf{R}^d$ and μ is the d-dimensional Lebesgue measure, the simple slice sampler is shown to be uniformly ergodic if π is bounded and the support of π has finite Lebesgue measure, [23]. If the latter condition is not met the simple slice sampler is still geometrically ergodic under extremely weak regularity conditions on Q [35]. Thus the simple slice sampler

has robust ergodicity properties, making it an appealing algorithm from a theoretical point of view. Furthermore, [23] shows that, given any independence Metropolis-Hastings sampler, it is possible to construct a simple slice sampler that performs better in the sense of reducing autocorrelation in the realised chain on a sweep by sweep basis (Peskun ordering [30]).

It is easy to see that the transition kernel of the simple slice sampler only depends on x through $\pi(x)$. Therefore the projected chain $\pi(X)$ is also Markov in its own right. Moreover after at least one iteration of the chain, it is clear that the distribution of $X|\pi(X) = c$ is "uniform" over $\{x \in \mathcal{X} : \pi(x) = c\}$. As a consequence of this, the question of convergence of X reduces to that of $\pi(X)$ (see [35] for further details).

As noted in [35] (see also Section 6), the Markov chain generated via the simple slice sampler is *stochastically monotone* with respect to the following partial ordering defined on the $\mathcal{X}$ portion of the state space:

$$x \prec x' \text{ if and only if } \pi(x) \leq \pi(x'). \tag{2}$$

Strictly speaking this is actually a total ordering on the projected space $\pi(\mathcal{X})$. However it is notationally simple to allow $\mathcal{X}$ to inherit the ordering.

Stochastic monotonicity means that, for all fixed $z \in \mathcal{X}$, we have:

$$P(X_1 \prec z|X_0 = x) \geq P(X_1 \prec z|X_0 = x') \text{ whenever } x \prec x'.$$

Notice that this ordering can be naturally extended to an ordering on the enlarged state space by simply disregarding the value of u, that is:

$$(x, u) \prec (x', u') \text{ if and only if } x \prec x'.$$

3 Perfect Simulation

Coupling from the past (CFTP) takes a positive recurrent Markov chain and simulates a realisation from the chain in a non-standard way. The simulation is started at a negative time from all possible starting points (at least in theory) until time 0. If all the chains are at the same state at time 0, (have coalesced) then the simulation ceases, and this common value is distributed according to the stationary distribution of the Markov chain. If there is more than one terminal value at time 0, the start of the simulation needs to be extended "backwards in time" until such time as the number of terminal values is just one.

Since it will rarely be possible in practice to simulate chains from all possible starting values, it is important to exploit structure, in the state space and in the transition kernel, that allows a smaller number of simulated chains to be constructed. Stochastic monotonicity is crucial in this respect. Furthermore, the description of perfect simulation above does not prescribe the joint distribution of the different chains. The freedom to choose this dependence structure is important for the implementation of CFTP.

An important device for defining the dependence structure between chains is the Stochastic Recursive Sequence (SRS) construction (see [4] and the references on perfect simulation listed in Section 1). With an SRS we can rewrite a Markov chain in terms of a measurable function, $f : \mathcal{X} \times (0,1) \to \mathcal{X}$, and a doubly infinite sequence of independent and identically distributed (iid) random variables, $\{\gamma_n\}_{n=-\infty}^{\infty}$, with $\gamma_n \sim U[(0,1)]$. The function f has to be chosen so that the following recursive formula holds:

$$X_0 = x_0, \qquad X_{n+1} = f(X_n, \gamma_n), \qquad n > 0; \tag{3}$$

and the sequence $\{X_n\}_{n=0}^{\infty}$ has the transition probabilities specified by the Markov chain. To simplify the notation, in the previous formula only a single doubly infinite sequence of uniform random variates is explicitly used but nothing prevents from defining the SRS in terms of more independent sequences. Furthermore, in practice, every software that claims to generates iid random variables with some specified cumulative distribution function (cdf) relies on the fact that there is a sequence of pseudo-random numbers available (in theory an infinite sequence, in practice the period of any random number generator is finite) and uses some algorithm, such as the inverse cdf method, to produce the output. Thus, all generators of random numbers (not uniform) can be rewritten in terms of an SRS constructions. Therefore, without loss of generality, we can take the sequence $\{\gamma_n\}_{n=-\infty}^{\infty}$ to be any sequence of (in theory) iid uniform random variables and then, if necessary, think of the function f as obtained by composition of functions.

Perfect simulation is considerably easier to implement for Markov chains that are stochastically monotone, since it turns out that we can often find an SRS that is monotone in its first argument (realizable monotonicity [11]). This allows the construction of a collection of Markov chains started at different states, which preserve their initial ordering [21]. Since the simple slice sampler is stochastically monotone according to the ordering given in (2), it turns out that we can in fact produce such an explicit SRS. Therefore, in

the sequel we will restrict our attention to stochastically monotone chains for which such an explicit SRS is available.

Having defined a partial order, it might be possible to find a maximal, x_{max}, and minimal, x_{min}, state. This is always the case for finite state spaces, and for the time being we shall assume the existence of these states (though in Section 5 we will show how this hypothesis can be eliminated). Notice that in our context it is possible for more than one maximal state to exist. However, since the set of maximal states is an atom in the sense of [22], it is straightforward to produce an SRS which merges all chains started at maximal states in a single iteration.

Let $X_n^{(x,T)}$ be the value at time n of a chain started in x at time $-T$. The perfect simulation algorithm works as follows. Choose a time $T_1 > 0$. Generate a stream of uniform random variables $\{\gamma_n\}_{n=-T_1}^{-1}$ and let

$$X_{-T_1}^{(x_{max},T_1)} = x_{max}, \quad X_{n+1}^{(x_{max},T_1)} = f(X_n^{(x_{max},T_1)}, \gamma_n), \quad -T_1 \leq n < 0, \tag{4}$$

$$X_{-T_1}^{(x_{min},T_1)} = x_{min}, \quad X_{n+1}^{(x_{min},T_1)} = f(X_n^{(x_{min},T_1)}, \gamma_n), \quad -T_1 \leq n < 0, \tag{5}$$

where f is the SRS function chosen for the transition kernel of interest. Notice that it is crucial to use the same stream of random innovations to update both chains: this ensures that, if the two sample paths meet at some time t', that is $X_{t'}^{(x_{max},T_1)} = X_{t'}^{(x_{min},T_1)}$, then $X_t^{(x_{max},T_1)} = X_t^{(x_{min},T_1)}$ for all $t > t'$, i.e. they coalesce.

If $X_0^{(x_{max},T_1)} = X_0^{(x_{min},T_1)} = X$, that is if the maximal and minimal chain have coalesced by time zero, then their random position at time zero, X, is distributed according to the target distribution. A *vertical backward coupling time* is defined to be $T = \sup\{t : X_0^{(x_{max},t)} = X_0^{(x_{min},t)}\}$. Following [13], if T is almost surely finite we call it a *successful* vertical backward coupling time. Strictly speaking a vertical backward coupling time is a coalescing time for the Markov chain starting from any $x \in \mathcal{X}$, [13]. But clearly if the maximal and minimal chain started at time $-T$ have the same value by time zero, then any chain started at time $-T$ will also have that same value by time zero because of the monotonicity property of the SRS function in its first component.

If the paths started at time T_1 have not coalesced by time zero, we choose a new value $T_2 > T_1 > 0$ and restart our backward simulation from time $-T_2$. In this second stage, when running the simulation over the time range

$[-T_1, 0]$, we need to reuse the same random numbers, $\{\gamma_n\}_{n=-T_1}^{-1}$, used in the first stage of the simulation. Although any value of T_2 would work, reasons for taking $T_k = 2^k$ are given in [31]. Another strategy is Wilson's read-once CFTP algorithm [38], which runs forwards in time, and never restarts at previous times in the past.

4 An Explicit SRS for the Slice Sampler

In this section we give an explicit SRS for the simple slice sampler. Armed with this construction, we can then carry out perfect simulation for the simple slice sampler. Assume for simplicity that a maximal and minimal state with respect to the order given in (2) exist and let them be x_{max} and x_{min} respectively; in fact, as explained at the end of this section, all we need to assume is that $\mu(\mathcal{X})$ and $\sup_{\mathcal{X}} \pi$ are both finite. We will discuss cases where these assumptions are further relaxed in Section 5.

We shall carry out the vertical slice first, followed by the horizontal slice. This will be done in a way that preserves monotonicity. Then from the construction, we will be able to extract the explicit SRS function.

We shall assume the existence of an infinite sequence of independent random variables at each stage of the recursion. Moreover, in order to ensure that we can start the simple slice sampler in x_{min} (and hence in any state $x \in \mathcal{X}$), and since $A(u) = \mathcal{X}$ for $u \sim U[(0, \pi(x_{min})]$, we assume that $\mu(\mathcal{X}) < \infty$.

For all $t < 0$ define a vertical slice random variable, $\epsilon_t \sim U[(0, 1)]$. Then, for the Markov chain that, at time t, is in state x, set $U_t(x) = \epsilon_t \pi(x)$.

The horizontal slice is more complicated. At each time $t < 0$ construct an infinite sequence of random variables, $\mathbf{W}_t = \{W_{t,j} : \ j = 1, 2, \dots\}$ by

$$W_{t,1} \sim U[A(U_t(x_{min}))] \tag{6}$$

and

$$W_{t,j} \sim U[A(\pi(W_{t,j-1}))] \ .$$

Define $\sigma_t(x)$ by

$$\sigma_t(x) = \inf\{j \geq 1 : \ \pi(W_{t,j}) \geq U_t(x)\} \ ,$$

and set

$$f(x, (\epsilon_t, \mathbf{W}_t)) = W_{t,\sigma_t(x)} \ . \tag{7}$$

Now $(\epsilon_t, \mathbf{W}_t)$ is a random innovation independent of x so that the above is indeed an SRS representation for some Markov chain.

It is easy to see that $\sigma_t(x)$ is almost surely finite for all $x \in \mathcal{X}$: let $\mathcal{F}_n$ denote the σ-algebra generated by $U_t(x), W_{t,1}, \ldots, W_{t,n}$, then $P(\sigma_t(x) = n | \sigma_t > (n-1) | \mathcal{F}_{n-1}) \geq Q(U_t(x))/Q(U_t(x_{min}))$ since the sequence $W_{t,\cdot}$ is non-decreasing with respect to the defined ordering (the region from which the trial values are generated becomes a successively smaller superset of $A(U_t(x))$).

It remains to check that the construction preserves the monotonicity property and that it does indeed simulate the simple slice sampler Markov chain.

Theorem 1. *The function in* (7) *is monotone in its first argument, so that the Markov chain obtained using* (7) *as its SRS function is stochastically monotone. Moreover, the Markov chain actually simulated is in fact a simple slice sampler.*

Proof. The monotonicity property follows from the fact that, for all t, $W_{t,\cdot}$ and $\sigma_t(\cdot)$ are non-decreasing sequences (the W's with respect to $\prec$) by construction.

To prove that the SRS does indeed simulate a simple slice sampler, it is enough to prove that $W_{t,\sigma_t(X_{t-1})}$ given $U_t(X_{t-1}) = u$ is distributed as P_u (see Section 2 for notation). However this is just an adaptive rejection sampling scheme where the rejection region becomes more and more refined as the simulation proceeds so that the result follows. □

We can now define a backward coupling for the simple slice sampler in the following way. Suppose $T_1, T_2, \ldots$ is an increasing sequence of integers which determine lengths of runs for simulations started back in time. As in Section 3 we shall often assume something similar to $T_k = 2^k$. However we will just consider the general case here.

The algorithm can be written as follows:

1. Set $i = 1$.
2. Compute $X_0^{(x_{max}, T_i)}$ and $X_0^{(x_{min}, T_i)}$ from the SRS formulae (4) and (5), using the explicit construction given in (7). If $X_0^{(x_{max}, T_i)} = X_0^{(x_{min}, T_i)}$ then the algorithm is complete and we set $I = i$.
3. Otherwise increment i by 1 and go to 2.

From the construction it follows that $X_0^{(x_{max},T_I)}$ is distributed according to π. Notice that since $\sigma_t(\cdot)$ is non-decreasing, it will never be necessary to increase the number of simulated $W_{t,j}$s since sample paths started at time point further back will be sandwiched in between sample paths started at later times (funneling property).

Modifications on the algorithm are possible, some of them reducing the computation needed. For instance for $t \in (-T_1, 0)$ it is not necessary to compute the sample path beginning at x_{min} at time t. A more efficient algorithm can be constructed by replacing x_{min} by $X_t^{(x_{min},T_1)}$ in (5).

Finally we notice that existence of x_{min} and x_{max} is really not required as long as $\mu(\mathcal{X}) < \infty$ and $\sup_{\mathcal{X}} \pi < \infty$: in (6) we actually generate $W_{t,1}$ from $\mu(\cdot)/\mu(\mathcal{X})$ for every t, and initially in step 2. above we have that, for all T_i,

$$X_{T_i}^{(x_{min},T_i)} = W_{T_i,1}$$

and

$$X_{T_i}^{(x_{max},T_i)} = W_{T_i,\max \sigma_{T_i}}$$

where

$$\max \sigma_t = \inf\{j \geq 1 : \pi(W_{t,j}) \geq \epsilon_t \sup_{\mathcal{X}} \pi\} \tag{8}$$

is almost surely finite.

5 Upper and Lower Process

Even if an ordering has been defined on the state space there are cases where it is hard or impossible to find a maximal and/or a minimal state — or just the value of $\sup_{\mathcal{X}} \pi$ as used in (8) above. A number of truncation and approximation schemes have been devised to circumvent this problem (see for example [14, 16, 18, 19, 20, 25]). Note that in contrast to the situation in Section 5 we do not require that the dominating measure μ is finite in the sequel.

Following ideas used in [16, 18, 19, 25], it is possible to construct upper and lower bounding stationary processes which allow us to define upper and lower starting values for our sandwiching algorithm.

Assume, for the time being, that we are in the general setting of a stochastically monotone Markov chain as in (3) with an SRS function $f(x, \gamma)$ which is non-decreasing in x.

Suppose that we have upper and lower bounding processes with transition probabilities P^{ub} and P^{lb} respectively, sandwiching the process of interest's probabilities P, that is, for all fixed $y \in \mathcal{X}$

$$P_x^{lb}(X_1 \prec y) \geq P_x(X_1 \prec y) \geq P_x^{ub}(X_1 \prec y) \ .$$

In our simple slice sampler example, bounding processes can be constructed out of simple slice samplers having different stationary distributions, and we shall implicitly construct SRS functions for the bounding processes, f^{ub} and f^{lb}, such that

$$f^{lb} \prec f \prec f^{ub}$$

where the inequalities are assumed to hold pointwise for all values of the arguments of the functions. These SRS constructions serve to guide the simulation to preserve all the relevant monotonicities.

The algorithm simulates stationary versions of the upper and lower processes, S and L respectively, backwards in time till time $-T_1$, noting the forward seeds, $\epsilon_{-T_1}, \ldots, \epsilon_{-1}$. The maximal process $X^{(x_{max},T_1)}$ now starts at $X_{-T_1}^{(x_{max},T_1)} = S_{-T_1}$. The minimal process $X^{(x_{min},T_1)}$ now begins at $X_{-T_1}^{(x_{min},T_1)} = L_{-T_1}$.

If $X_0^{(x_{max},-T_1)} = X_0^{(x_{min},-T_1)}$, the algorithm ceases outputting the common value. Otherwise, the upper and lower processes are extended backwards in time to time $-T_2$ and the procedure is repeated making sure to retain the seeds from the original time interval. This whole construction is extended back in time until we get a k such that $X_0^{(x_{max},-T_k)} = X_0^{(x_{min},-T_k)}$ and the common value is output. This value can be shown to be distributed as π, see for example the arguments in [18, 19, 25].

Here we shall restrict attention to the case where π is bounded. This means that according to the ordering $\prec$, x_{max} exists, but a lower bounding process construction is needed. The ideas here readily extend to the case where π is not bounded.

Here we will describe two ways of potentially exhibiting a bounding process for the simple slice sampler. The first is probabilistically the most natural, since it is constructed entirely from the Q function itself (which completely characterises the algorithm, cf. Section 2). However, it turns out that

the second method is easier to implement, and the example that we will give exploits this second procedure.

5.1 Bounding processes for the simple slice sampler: the Q method

In this section we assume that $\mathcal{X} = \mathbf{R}^d$ and μ is the Lebesgue measure. In order to define a minimal process for the simple slice sampler Markov chain X induced by π we find an unnormalised density π_2 such that the following condition is verified:

$$\left[\frac{Q'_{\pi}(u)}{Q_{\pi}(u)} - \frac{Q'_{\pi_2}(u)}{Q_{\pi_2}(u)}\right] \leq 0. \tag{9}$$

where $Q'(u) = \frac{\partial Q(u)}{\partial u}$.

Theorem 2. *Suppose that π and π_2 are densities with corresponding Q functions satisfying (9). Let $X(2)$ be the simple slice sampler Markov chain induced by π_2. Then the Markov chains $Y = \pi(X)$ and $Y(2) = \pi_2(X(2))$, are stochastically ordered in the sense that*

$$P(Y_1 \leq \lambda | Y_0 = \beta) \geq P(Y(2)_1 \leq \lambda | Y(2)_0 = \beta)$$

for all $\lambda, \beta \in \mathbf{R}$.

Proof. Writing π_1 for π, $Y(1)$ for Y, from [35], for i=1,2

$$P(Y(i)_1 \leq \lambda | Y(i)_0 = \beta) = \frac{1}{\beta}\int_0^{\beta} \max\left\{0, 1 - \frac{Q_{\pi_i}(\lambda)}{Q_{\pi_i}(z)}\right\} dz. \tag{10}$$

However, (9) implies that for all $z < \lambda$

$$\int_z^{\lambda} \frac{Q'_{\pi_1}(w)}{Q_{\pi_1}(w)} dw \leq \int_z^{\lambda} \frac{Q'_{\pi_2}(w)}{Q_{\pi_2}(w)} dw$$

so that for all $z < \lambda$

$$1 - \frac{Q_{\pi_1}(\lambda)}{Q_{\pi_1}(z)} \geq 1 - \frac{Q_{\pi_2}(\lambda)}{Q_{\pi_2}(z)}.$$

Hence, the integrand in (10) for $Y(1)$ dominates that for $Y(2)$ and the result follows. □

Although this is a natural way to bound the process in principle, the practical implementation of it is hampered by the fact that explicit information about Q'/Q is needed, and this is unlikely to be available in real problems.

5.2 Bounding processes for the simple slice sampler: the subset method

Let us consider a target distribution π which is bounded with x_{max} achieving this supremum. Repeating the arguments at the end of Section 5 we see that existence of x_{max} (or x_{min}) is really not needed in the following. Suppose however, that there exists an unnormalised density π^{lb} (again with reference to μ) such that for all $x \in \mathcal{X}, \lambda \in [0,1]$,

$$A_{\pi^{lb}}(\lambda \pi^{lb}(x)) \supseteq A_{\pi}(\lambda \pi(x)) \ . \tag{11}$$

Notice that by taking $\lambda = 1$, (11) implies that the partial orderings defined on $\mathcal{X}$ by π and π^{lb} are identical. Let X and X^{lb} denote simple slice samplers on the densities π and π^{lb} respectively.

Lemma 1. *X^{lb} is stochastically dominated by X according to $\prec$.*

We omit the proof of Lemma 1 since the SRS constructed below will serve as a direct constructive proof by inspection. This lemma allows X^{lb} to be used as a lower bounding process. The assumption is (in practice) that simulation from π^{lb} is feasible directly. The following is a coupling construction which jointly constructs a stationary version of X^{lb} together with versions of X started at any starting point that dominates (in the sense defined in (2)) the starting point of the lower bounding process. The construction preserves the almost sure order of the processes, and also respects (in an almost sure sense) the stochastic monotonicity of X by preserving the order of processes started at different values.

Forward coupling. For the purpose of this construction we shall assume that the simulation begins at time 0. Suppose that $X_n^{lb} = x_-$ and the position at time n of the version of X started at x is denoted by $X_n^{(x,0)}$. We shall only consider $X_n^{(x,0)}$ for $x \succ X_0^{lb}$ and assume that $X_n^{(x,0)}$ is a non-decreasing function of x. (This is ultimately verified by induction.)

At each iteration, the algorithm proceeds as follows.

1. Choose $\epsilon_{n+1} \sim U(0,1)$. Set $U_{n+1}^{lb} = \epsilon_{n+1}\pi^{lb}(x_-)$, and for $X_0^{lb} \prec x \prec x_{max}$, set $U_{n+1}^{(x,0)} = \epsilon_{n+1}\pi(X_n^{(x,0)})$.

2. Define a sequence of iid random variables $\tilde{V}_{n+1,1}; \tilde{V}_{n+1,2}; \ldots$ from $U[A_{\pi^{lb}}(U_{n+1}^{lb})]$. Set $X_{n+1}^{lb} = \tilde{V}_{n+1,1}$.

3. Let $\tau_{n+1} = \inf\{k : \tilde{V}_{n+1,k} \in A_\pi(U_{n+1}^{(x_-,0)})\}$.

4. Set $X_{n+1}^{(x_-,0)} = \tilde{V}_{n+1,\tau_{n+1}} = W_{n+1,1}$. For $i \geq 2$, recursively define $W_{n+1,i}$ to be drawn from $U[A_\pi(\pi(W_{n+1,i-1}))]$.

5. Set $X_{n+1}^{(x,0)} = W_{n+1,\sigma(n+1,x)}$ where $\sigma(n+1,x) = \inf\{i : W_{n+1,i} \geq Y_{n+1}^{(x,0)}\}$.

Essentially, 1. and 2. construct the minimal process while 3. 4. and 5. are adapted from the construction in Section 4. The following result is a consequence of the fact that the horizontal sampling in the above algorithm is just an elaborate adaptive rejection sampling scheme similar to that given in Section 4.

Lemma 2. *$X^{(x,0)}$ is a simple slice sampler for the density π.*

In order to perform backwards simulation with the bounding process, we shall need to construct the time reversed simple slice sampler for π^{lb}. Fortunately, by reversibility of the simple slice sampler, this is identical to the forwards simple slice sampler if we reverse the order in which the vertical and horizontal updates are done (see Section 2).

However we need to be careful in translating the backwards iterations to the seeds needed to construct forward iterations for other processes — for this we consider the conditional distribution of the forwards seeds $\epsilon_n, \{\tilde{V}_{n,i}\}, \{W_{n,i}\}$ needed at time n given (X_{n-1}^{lb}, X_n^{lb}) (these forwards seeds given (X_{n-1}^{lb}, X_n^{lb}) are conditionally independent of anything else in the "simulation past" of the forwards simple slice sampler; see Section 3 in [19]).

Backwards simple slice for π^{lb}. Given $X_n^{lb} \sim \pi^{lb}$, generate ϵ_{n-1}^* from $U(0,1)$, and then generate X_{n-1}^{lb} from $U[A_{\pi^{lb}}(\epsilon_{n-1}^* \pi^{lb}(X_n^{lb}))]$. Reconstruct the forward seeds as follows:

$$\epsilon_n = \frac{\epsilon_{n-1}^* \pi^{lb}(X_n^{lb})}{\pi^{lb}(X_{n-1}^{lb})}, \tag{12}$$

and

$$\tilde{V}_{n,1} = X_n^{lb} \ . \tag{13}$$

All other seeds (needed to perform steps 2. and 4.) can be chosen according to the forward coupling construction.

It remains to give the full CFTP algorithm.

CFTP algorithm. Choose a sequence of increasing positive integers $T_1, T_2, \ldots$ Generate X_0^{lb} from π^{lb}. Set $i = 1$, $T_0 = 0$.

1. Construct the simple slice sampler on π^{lb} backwards from $-T_{i-1}$ to $-T_i$, noting the forward seeds from the backwards construction above. Call this process X^{lb}.

2. Using the forward seeds from 1. construct two simple slice samplers on π started at x_{max} and $X^{lb}_{-T_i}$, from time $-T_i$ to $-T_{i-1}$.

3. If $X_0^{(X^{lb}_{-T_i},T_i)} = X_0^{(x_{max},T_i)}$, stop, reporting this value.

4. Otherwise, set $i = i + 1$ and go to 1.

6 Examples and extensions

Example 1 below shows that for one-dimensional distributions other and simpler perfect simulation methods such as rejection sampling may prove to be at least as efficient as perfect simple slice sampling. Another problem which typically occurs for multivariate distributions is how to determine the set $A(u)$ used in the horizontal update; rejection sampling may here not be feasible. However, [5] demonstrate that perfect slice sampling can be achieved for realistic statistical models and not only for toy problems.

In Sections 6.1–6.2 we study various extensions of the perfect simple slice sampler considered so far.

Example 1. Suppose we wish to generate iid observations from

$$\pi(x) \propto e^{-x}(1+x)^{-\alpha} \qquad x \geq 0 \tag{14}$$

for some $\alpha > 0$ and with reference to μ, the Lebesgue measure restricted to $\mathcal{X} = [0, \infty)$. We are in the situation where the target density is bounded but its support is not. In this case there is a maximal state, namely $x_{max} = 0$, but there is no minimal state. We thus construct a lower bounding process following the theory given in Section 5.2. Let

$$\pi^{lb}(x) \propto e^{-qx} \qquad x \geq 0. \tag{15}$$

For $0 < q \leq 1$ condition (11) is verified. We shall consider the case where $\alpha = 1$ and compare values for q varying between 0 and 1.

Implementing the perfect simple slice sampler using lower bounding simple slice sampler with target π^{lb} with $q = 1$ required a backward simulation

time T which took the values $(1, 2, 4, 8, 16)$ with frequencies $(407, 281, 225, 83, 4)$ respectively in a simulations study of 1000 replicates of our perfect simulation algorithm.

In Table 1 we report the running time in seconds (column labeled Time) of two algorithms each returning 5 000 iid samples from π. The first column refers to a rejection sampler, the second to a perfect simple slice sampler, and the star indicates the winning strategy. As $q \to 0$ the lower bounding process and the enveloping function become less tight. When sampling with rejection this results in higher rejection probability. In the perfect simple slice sampler we need to go further back in time to achieve coalescence of the maximal and minimal chains. As a result the time needed to obtain a fixed number of iid samples increases as q decreases in both cases but it seems that the negative effect of using a non-optimal enveloping function (bounding process) is worse for the rejection sampler.

	Rejection	Perfection
q	Time	Time
1	8.03*	40.01
0.9	10.07*	43.84
0.8	14.54*	45.15
0.7	17.45*	48.80
0.6	25.02*	53.37
0.5	37.10*	60.37
0.4	59.05*	65.96
0.3	105.51	77.41*
0.2	248.22	98.26*
0.1	1020.38	137.34*

Table 1: Comparing iid samplers (running times in seconds).

All the programs (for this and the other examples) are written in the Xlispstat language. The simulations have been implemented on a Pentium 300 MHz under the Linux operating system.

6.1 Perfect product slice sampling

The product slice sampler [2, 8, 9, 23, 35, 37] is an extension of the simple slice sampler where more than one auxiliary variable is introduced. Suppose

that the target density can be factorised as

$$\pi(x) = \prod_{i \in I} \pi_i(x)\,, \quad x \in \mathcal{X}, \tag{16}$$

where I is a finite index set and each π_i is a measurable non-negative function. Then we can extend the slice sampler idea as follows:

- vertical update: $\mathbf{u}|x$ is sampled from $U[\otimes_{i \in I}(0, \pi_i(x))]$;
- horizontal update: $x|\mathbf{u}$ is sampled from the normalisation of the restriction of μ to the set

$$A_\pi(\mathbf{u}) = \{x : \pi_i(x) > u_i \text{ for all } i \in I\} \tag{17}$$

  with $\mathbf{u} = (u_i : i \in I)$.

The product slice sampler is stochastically monotone with respect to the partial ordering given by

$$x \prec x' \text{ if and only if } \pi_i(x) \leq \pi_i(x') \text{ for all } i \in I. \tag{18}$$

However the problem is to realise how to construct a monotone SRS for the product slice sampler.

For any SRS construction of the product slice sampler, it seems natural to use vertical slice variables $\epsilon_t = (\epsilon_{t,i} : i \in I)$ at time t, where the variables $\epsilon_{t,i} \sim U[(0,1)]$, $t \in \mathbf{Z}$, $i \in I$ are independent. Then $X_t|(\epsilon_t, X_{t-1})$ follows the normalisation of μ restricted to $\{x : \pi_i(x) > \epsilon_{t,i}\pi(X_{t-1}) \text{ for all } i \in I\}$. This is like in the SRS's for the simple slice sampler where I is a singleton. But, as exemplified below, it is not straightforward to extend the coupling used for horizontal updates in any of the perfect slice samplers in Sections 4 and 5 to a monotone SRS in the present setting (I not being a singleton), using the ordering in (18).

Example 2: Ising and other random field models. Probably the most famous example of a product slice sampler is the Swendsen-Wang algorithm [37] which was introduced for tackling the critical slowing down problems with ordinary Metropolis-Hastings algorithms for Ising and Potts models. For the simplest case of the symmetric Ising model (no external field and constant coupling parameters), $I = E$ is the edge set of a finite

non-oriented graph $G = (V, E)$, μ is counting measure on $\mathcal{X} = \{0, 1\}^V$, and for $i = \{j, k\} \in E$,

$$\pi_i(\mathbf{x}) = \exp(-\beta \mathbf{1}[x_j \neq x_k]) .$$

Here $\beta > 0$ is some "coupling parameter" and $\mathbf{1}[\cdot]$ is the indicator function. In the horizontal update, if we define u-clusters C as the maximal connected components in the subgraph $G(\mathbf{u}) = (V, E(\mathbf{u}))$ with the same vertex set as G but edge set $E(\mathbf{u}) = \{i \in E : u_i > \exp(-\beta)\}$, the $\mathbf{x}_C = (x_j : j \in C)$ on different $\mathbf{u}$-clusters C are conditionally independent given $\mathbf{u} = (u_i : i \in I)$.

Further, conditional on $\mathbf{u}$, x_j is constant on each u-cluster with the common value drawn from a uniform distribution on $\{0, 1\}$. Note that Swendsen and Wang's algorithm depends only on u_i through the "bond" $z_i = \mathbf{1}[u_i > \exp(-\beta)]$, where $z_i|\mathbf{x}$ is 0 with probability $\exp(-\beta \mathbf{1}[x_j \neq x_k])$. The marginal equilibrium distribution for these bonds is a so-called random cluster model.

Propp and Wilson in [31] were unable to find monotonicity in the Swendsen-Wang algorithm. It is illuminating to see why monotonicity cannot possibly be present according to the partial ordering $\leq$. To see this simply note that $E(X_{t,j}|\mathbf{X}_{t-1}) = 1/2$ for the Swendsen-Wang algorithm. Therefore the Markov chain could only be stochastically monotone with respect to $\leq$ if its transition probabilities were independent of its current state, which is of course impossible here.

The perfect CFTP algorithm used in [31, 32] is based on first generating a perfect simulation of the random cluster model using Gibbs sampling, which is monotone with respect to $\leq$, and secondly conditionally on this simulation generating a perfect realisation of the symmetric Ising model.

Example 3: Modified Swendsen-Wang. Here we introduce a modified Swendsen-Wang algorithm for which we can actually obtain stochastic monotonicity and in fact realisable monotonicity. The new method is also somewhat more flexible than the standard Swendson-Wang method, being readily applicable in the case where there exists an external field, and we shall introduce it in a slightly more general context.

Consider the Markov random field with density $\prod_{\{j,k\} \in E} \exp(\beta_{j,k} x_j x_k)$ with respect to the dominating measure $\mu = \otimes_{l \in V} \mu_l = \otimes_{l \in V} \mathcal{B}(p_l)$ where $\mathcal{B}(p_l)$ is Bernoulli with mean p_l. The only constraint we need to impose is the positivity restriction $0 < p_l < 1$ and $\beta_{j,k} \geq 0$. Note that the symmetric

Ising model just sets $\beta_{j,k} = 2\beta$ for all $\{j,k\} \in E$ and $\log(p_l/(1-p_l)) = -\beta\delta_l$ for all j, where $\delta_l = \#\{k;\ \{k,l\} \in E\}$. An Ising model with external field is obtained if

$$\beta_{j,k} \equiv 2\beta, \quad \log(p_j/(1-p_j)) = (2y_l - 1)\log(c/(1-c)) - \beta\delta_l, \tag{19}$$

where $0 < c < 1$ and $y_l \in \{0,1\}$. This is the posterior distribution determined by a symmetric Ising prior and a likelihood for the y_l given x specified by independent Bernoulli variates so that $P(y_l = x_l|x) = c$. However our general formulation allows not only an external field, but also the possibility that the strength of the interaction parameter $\beta_{j,k}$ can vary spatially.

Now $\mathbf{X}_t|(\epsilon_t, \mathbf{X}_{t-1}) = (\epsilon, \mathbf{x})$ simply follows the distribution μ restricted to the set $A(\epsilon, \mathbf{x}) = \prod_{l \in V} A_l(\epsilon, \mathbf{x})$ where $A_j(\epsilon, \mathbf{x}) = \{1\}$ if both $\epsilon_{j,k} > \exp(-\beta_{j,k})$ and $x_j = x_k = 1$ for some k with $\{j,k\} \in E$, and $A_j(\epsilon, \mathbf{x}) = \{0,1\}$ otherwise. Hence it is possible to construct an explicit monotone SRS: for the coupling at horizontal updates, generate independent Bernoulli variates $B_{t,l} \sim \mu_l,\ t \in \mathbf{Z},\ l \in V$, and set

$$X_{t,l} = 1 \text{ if } A_l(\epsilon_t, \mathbf{X}_{t-1}) = \{1\}, \quad X_{t,l} = B_{t,l} \text{ otherwise.}$$

This is even monotone with respect to the usual partial ordering on $\{0,1\}^V$ (i.e. the ordering given by $\mathbf{x} \leq \mathbf{y}$ if and only if $x_l \leq y_l$ for all $l \in V$), which in turn implies than the partial ordering defined in (18).

Based on this SRS construction and using $\mathbf{x}_{min} = (0, \ldots, 0)$ and $\mathbf{x}_{max} = (1, \ldots, 1)$ as the minimal and maximal states, we obtain a perfect product slice sampler similar to that in Section 4. This perfect simulation algorithm is clearly simpler to implement than the perfect CFTP algorithm used in [31, 32] for the symmetric Ising model. For the model (19), it would be interesting to compare our perfect product slice sampler with perfect CFTP based on Gibbs sampling, which is still monotone in the case of an external field, and relate such a comparison to applications in image noise reduction algorithms (see e.g. [12]).

In the experiments reported on below $V \subset \mathbf{Z}^2$ is rectangular, E is specified via nearest-neighbours in $\mathbf{Z}^2$, and $\beta_{j,k} = 2\beta$ for all $\{j,k\} \in E$ (i.e. no external field, constant coupling parameter). All the running times are expressed in seconds. We also report the index I of T_I, the vertical backward coupling time, following the notation in Section 4.

Table 2 shows the performance of the perfect slice sampler for the Ising model with no external fields as the value of $\beta_{j,k}$ varies from 0.2 to 1.1. All the simulations results in this table are obtained for a square grid of size 20.

Table 3 shows the performance of the perfect slice sampler for the Ising model with no external fields as the grid size increases. All the simulations results in this table are obtained at the critical value of $\beta_{j,k} = 0.88$.

$\beta_{j,k}$	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.88	0.9	1.0	1.1
running time	26	53	53	110	110	222	222	225	451	452	459
value of I	3	4	4	5	5	6	6	6	7	7	7

Table 2: Comparing performance for different values of $\beta_{j,k}$ on a 20×20 grid.

grid size	5	7	9	11	13	15	17	19	21	30
running time	0.84	2.54	11.82	24.49	43.85	74.72	120	375	544	931
value of I	5	5	6	6	6	6	6	7	7	8

Table 3: Comparing performance for different grid sizes, $\beta_{j,k} = 0.88$.

The Swendsen-Wang algorithm can easily be extended to other types of random fields with pairwise interaction than just the Ising model, but the practical applications have often been somewhat disappointing, see e.g. [9]. We have yet not investigated to what extent we can construct useful perfect product slice samplers for such models, but the discussion above for the Ising model may indicate that it is not a simple task.

6.2 Perfect single-component slice sampling for multivariate distributions

In many applications we have that $\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j$ is a product space furnished with a product measure $\mu = \prod_{j=1}^{d} \mu_j$, and we write $\mathbf{x} = (x_1, \ldots, x_d)$ with $x_j \in \mathcal{X}_j$ being the j^{th} variable or component. All models considered in

the examples above can be presented in this way. In order to simplify the horizontal update in the simple slice sampler, one obvious idea is to use a single variable Gibbs sampling update instead: first chose $j \in \{1, \ldots, d\}$ uniform at random or in some systematic way like scanning through $1, \ldots, d$; then use the conditional distribution $x_j|(\mathbf{x}_{-j}, u)$ to update the j^{th} variable given all the others $\mathbf{x}_{-j} = (x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d)$ and the restriction to the set

$$A_j(u, \mathbf{x}_{-j}) = \{x_j \in \mathcal{X}_j : \pi(\mathbf{x}) > u\}\,.$$

In other words, the vertical update is unchanged, but the horizontal is replaced by a horizontal single-component update:

$$u|\mathbf{x} \sim U[(0, \pi(\mathbf{x}))]\,, \quad x_j|(\mathbf{x}_{-j}, u) \sim U_j[A_j(u, \mathbf{x}_{-j})]\,,$$

where $U_j[A]$ denotes the normalisation of the restriction of μ_j to the set $A \subseteq \mathcal{X}_j$. The chain obtained by alternating between vertical and horizontal single-component updates, using a random or systematic sequence $\{j_t\}_0^\infty$ so that at the t^{th} iteration the j_t^{th} variable is to be updated, is called a single-component slice sampler. Ergodicity requires that we insist that $\#\{t : j_t = j\} = \infty$ for all $j \in \{1, \ldots, d\}$. In order to ensure irreducibility, weak regularity condition need to be imposed on the target density. As long as π is everywhere finite, it is sufficient to impose the conditions that ensure the irreducibility of the corresponding Gibbs sampler on π with the given parameterisation. Fairly sharp conditions for this can be found in [36].

Using a notation as in Section 2 and integrating out the distribution of U_{t+1}, we obtain the transition probability P_j of $\mathbf{X}_{t+1}$ given that $\mathbf{X}_t = \mathbf{x}$ and $j_{t+1} = j$: for any $z > 0$ and $\mathbf{x} \in \mathcal{X}$ with $\pi(\mathbf{x}) > 0$, setting $Q_j(z|\mathbf{x}_{-j}) = \mu_j(A_j(z, \mathbf{x}_{-j}))$, we have that

$$P_j(\pi(\mathbf{X}_{n+1}) < z|\mathbf{X}_n = \mathbf{x}) = \frac{1}{\pi(\mathbf{x})} \int_0^{\pi(\mathbf{x})} \max\{1 - \frac{Q_j(z|\mathbf{x}_{-j})}{Q_j(w|\mathbf{x}_{-j})}, 0\} dw\,.$$

This is an average over the interval $(0, \pi(\mathbf{x}))$ of the function $q(w) = \max\{1 - \frac{Q_j(z|\mathbf{x}_{-j})}{Q_j(w|\mathbf{x}_{-j})}, 0\}$, which is non-increasing, so $P_j(\pi(\mathbf{X}_{n+1}) < z|\mathbf{X}_n = \mathbf{x})$ is in fact a non-increasing function of $\pi(\mathbf{x})$. Hence the $\mathbf{x}$-chain is seen to be stochastically monotone with respect to the density ordering (2).

However, for $d \geq 2$ (if $d = 1$ we are just considering a simple slice sampler as in Section 2) it still remains to realise how to construct a monotone SRS

for the single-component slice sampler. The vertical update is naturally done as in the simple slice sampler, but the horizontal single-component update is more complicated because of the conditioning on variables x_{-j}. This means that the constructions in Sections 4–5 can not immediately be extended to the present case.

As an illuminating example of the distinction between stochastic and realisable monotonicity, consider the case where $d = 2$, μ_j is Lebesgue measure restricted to $\mathcal{X}_j = [0, 1], j = 1, 2$, and $\pi(x_1, x_2) = \mathbf{1}[\min\{|x_1 - x_2|, 1 - |x_1 - x_2|\} < 1/4]$ (i.e. a hard core process with 2 points defined on the unit interval wrapped on a circle). Obviously we have stochastic monotonicity as $P_j(\pi(\mathbf{X}_{n+1}) \leq z|\mathbf{X}_n = x) = \mathbf{1}[z \geq 1]$, but this turns out not to be helpful in trying to construct a monotone single-component slice sampler. $x_1|(x_2, u)$ follows a uniform distribution on $A_1(x_2, u) = \{x_1 \in [0, 1] : |x_1 - x_2| \leq 1/4$ or $\geq 3/4\}$ (and similarly for $x_2|(x_1, u)$ if we exchange the roles of the indices 1 and 2). Now, it seems impossible to achieve realisable monotonicity. We leave the details of this as an easy exercise for the reader.

However, as exemplified below, imposing further "structure" on the target density π it may be possible to construct perfect single-component slice samplers even if the SRS is non-monotone.

Example 4: Ising and other auto-models. Consider the case where each $\mathcal{X}_j \subseteq [0, \infty)$ and we have pairwise interactions as in

$$\pi(x_1, \ldots, x_d) = \exp\Big(\sum_{1 \leq j < k \leq d} \beta_{j,k} x_j x_k\Big),$$

where the $\beta_{j,k}$ are real parameters chosen so that π is integrable with respect to a specified μ. Examples include the Ising model and the following auto-models ([3, 7]):

- auto-binomial: $\mathcal{X}_j = \{0, \ldots, n_j\}$, μ_j is Binomial(n_j, p_j) with $0 < p_j < 1$, $n_j \in \mathbf{N}$, and $\beta_{j,k} \in \mathbf{R}$;
- auto-Poisson: $\mathcal{X}_j = \{0, 1, \ldots\}$, μ_j is Poisson(λ_j) with $\lambda_j > 0$ and $\beta_{j,k} \leq 0$;
- auto-gamma: $\mathcal{X}_j = [0, \infty)$, μ_j is Gamma(α_j, β_j) with $\alpha_j > 0$, $\beta_j > 0$, and $\beta_{j,k} \leq 0$.

Note that the symmetric Ising model in Example 2 and the extension studied in Example 3 are particular cases of the auto-binomial model (taking all $n_j = 1$) except that we now allow for repulsive interaction (the case where all $\beta_{j,k} \leq 0$). Perfect simulation of such models based on Gibbs sampling and extensions of Fill algorithms [10] are studied in [25, 27, 39].

Below we consider perfect single-component slice sampling in the case when all $\mu_j(\mathcal{X}_j) < \infty$.

One SRS construction is the following modification of that in Section 4. For vertical updates we use, as before, $U_t(\mathbf{x}) = \epsilon_t \pi(\mathbf{x})$ with $\epsilon_t \sim U[(0,1)]$. If $u = \epsilon\pi(\mathbf{x})$ and we set $\beta_{k,j} = \beta_{j,k}$ and $\beta_{j,j} = 0$, then

$$A_j(u, \mathbf{x}_{-j}) = \mathcal{X}_j \cap J_j(\epsilon, \mathbf{x})$$

where

$$J_j(\epsilon, \mathbf{x}) = \{y_j \geq 0 : (y_j - x_j) \sum_k \beta_{j,k} x_k > \log \epsilon\}$$

is an interval. Now suppose $j_t = j$ and draw

$$V_{t,1} \sim U_j[\mathcal{X}_j]\,,\ V_{t,m} \sim U_j[\mathcal{X}_j \cap (V_{t,m-1}, \infty)],\ m = 2, 3, \ldots$$

(setting $V_{t,m} = \infty$ if $\mathcal{X}_j \cap (V_{t,m-1}, \infty) = \emptyset$). Then the SRS is given by

$$f(\mathbf{x}, (\epsilon_t, \{V_{t,m}\}_{m=1}^{\infty}, j)) = \text{first } V_{t,m} \text{ with } V_{t,m} \in J_j(\epsilon_t, \mathbf{x})\,. \tag{20}$$

This is only monotone in the case of attractive π, i.e. when all $\beta_{j,k} \geq 0$.

Furthermore, we define recursively upper $\mathbf{X}_t^{(\mathbf{x}_{max}, T_i)}$ and lower $\mathbf{X}_t^{(\mathbf{x}_{min}, T_i)}$ chains as follows, where we use again a notation as in Section 4 (though we do not exactly require the existence of a maximal and a minimal state). Assume first, for simplicity, that all $\beta_{j,k} \geq 0$ (the attractive case) and that each space $\mathcal{X}_j$ contains its upper bound $c_j = \sup \mathcal{X}_j < \infty$ (this is usually needed in order to ensure integrability). Then $J_j(\epsilon, \mathbf{x})$ decreases as $\mathbf{x}$ increases with respect to the natural ordering on $\mathbf{R}^d$ (which in the present attractive case is effectively identical to the density ordering). So we use the same perfect simulation algorithm as in Section 4 except that we now use the construction given in (20) for $\mathbf{X}^{(\mathbf{x}_{max}, T_i)}$ and respectively $\mathbf{X}^{(\mathbf{x}_{min}, T_i)}$, and set $\mathbf{c} = (c_1, \ldots, c_n)$,

$$\mathbf{X}_{T_i}^{(\mathbf{x}_{max}, T_i)} = \text{first } V_{T_i,m} \text{ with } V_{T_i,m} \in J_j(\epsilon_{T_i}, \mathbf{c}) \quad (j = j_{T_i})$$

and

$$\mathbf{X}_{T_i}^{(\mathbf{x}_{min},T_i)} = V_{T_i,1}\,.$$

In the opposite repulsive case where all $\beta_{j,k} \leq 0$, we have that $J_j(\epsilon, \mathbf{x})$ increases as x_j increases, while $J_j(\epsilon, \mathbf{x})$ still decreases as $\mathbf{x}_{-j}$ increases (with respect to the natural ordering on $\mathbf{R}^{d-1}$). Notice that, as in the auto-Poisson and auto-gamma models, we do not assume anymore that the c_j are finite. However we now assume that $0 \in \mathcal{X}_j,\ j = 1, \ldots, d$ and let $\mathbf{0} = (0, \ldots, 0)$ be the maximal state. Then we can set

$$\mathbf{X}_{T_i}^{(\mathbf{x}_{max},T_i)} = \text{first } V_{T_i,m} \text{ with } V_{T_i,m} \in J_j(\epsilon_{T_i}, \mathbf{0}) \quad (j = j_{T_i})\,,$$

$$\mathbf{X}_{T_i}^{\mathbf{x}_{min},T_i)} = V_{T_i,1}\,,$$

and use for $t > T_i$, $j = j_t$ and $\mathbf{X}_{t-1}^{(\mathbf{x}_{min},T_i)} = \mathbf{x} \prec \mathbf{X}_{t-1}^{(\mathbf{x}_{max},T_i)} = y$ a trick first introduced in [18]:

$$\mathbf{X}_t^{(\mathbf{x}_{max},T_i)} = \text{first } V_{t,m} \text{ with } V_{t,m} \in J_j(\epsilon_t, (x_1, \ldots, x_{j-1}, y_j, x_{j+1}, \ldots, x_d))\,,$$

$$\mathbf{X}_t^{(\mathbf{x}_{min},T_i)} = \text{first } V_{t,m} \text{ with } V_{t,m} \in J_j(\epsilon_t, (y_1, \ldots, y_{j-1}, x_j, y_{j+1}, \ldots, y_d))\,.$$

One can similarly handle the general case where the $\beta_{j,k}$ may have different signs.

7 Conclusion and discussion

Coupling techniques have been proven to be a very powerful tool in assessing convergence of Monte Carlo Markov chains. In [33] for example, a convergence diagnostic is obtained by running coupled chains from the present into the future (while in perfect simulations the coupled chains evolve from the past to the present). In [21] bounds on the distance to stationarity are provided by coupling two chains one of which is started in equilibrium. Shift coupling [1] has been used in [34] to examine convergence of ergodic averages of Markov chain distributions.

The present work provides procedures for the implementation of various versions of slice sampling by perfect simulation. The algorithm for the simple

slice sampler is quite general, however in the non-uniformly ergodic case, the need to produce an appropriate bounding process does limit the applicability of the result. Notice though that we only need to "bound" the tails of the target distribution. As done in [15] we can partition the support of the target distribution, use different samplers on different regions and then combine them in a single CFTP algorithm. In particular, on bounded regions, where we can find a minimal and maximal state, there is no need for a lower or upper bounding process, we can simply proceed as in Section 4. We only need to make sure that, when combining different samplers, the monotonicity property of the resulting SRS construction is preserved.

We started for simplicity with the case of the simple slice sampler (Sections 2–5). Example 1 and [5] demonstrate the applicability of perfect slice sampling. However, as pointed out in Section 6, perfect product and single-component slice samplers seem more relevant for many applications. The techniques apply in theory on a certain range of models, but whether our perfect samplers are useful in practice depends of course much on the properties of the particular models.

Like many other perfect samplers introduced so far in the literature, ours may turn out to be successful for rather limited cases of models of "real" interest. In our opinion perfect simulation has so far proved more successful for problems in spatial statistics, stochastic geometry and statistical physics (see e.g. [16, 18, 17, 19, 20, 25, 27, 31]) than in Bayesian statistics (see e.g. [5, 15, 26, 28, 29]); certainly, as perfect simulation is a rapidly developing area of research, this view may be changed in the future. In the examples considered we have also focused on spatial models. As indicated there exist alternative perfect simulation procedures described elsewhere in the literature, but at this exploratory stage we have only compared the various method in the case of the Ising model with no external field. Possibly the perfect product and single-component slice samplers for the Ising model with an external field included may prove to be useful for problems in Bayesian image analysis.

Acknowledgements

We wish to thank John Baxter for stimulating discussions about this work and Alberto Riva for precious assistance with the implementation of the algorithm.

This work has been supported by EU TMR network ERB-FMRX-CT96-0095 on "Computational and Statistical methods for the analysis of spatial

data". The work by Jesper Møller has also been supported by Centre for Mathematical Physics and Stochastics, funded by a grant from The Danish National Research Foundation.

References

[1] D.J. Aldous and H. Thorisson. Shift-coupling. *Stochastics Processes and Applications*, 44:1–14, 1993.

[2] J. Besag and P.J. Green. Spatial statistics and Bayesian computation. *Journal of the Royal Statistical Society*, B 55:25–37, 1993.

[3] J.E. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society*, B 36:192–236, 1974.

[4] A.A. Borovkov and S.G. Foss. Stochastically recursive sequences and their generalizations. *Siberian Advances in Mathematics*, 2:16–81, 1992. Translated from PRIM.

[5] G. Casella, K.L. Mengersen, C.P. Robert, and D.M. Titterington. Perfect slice samplers for mixtures of distributions. Technical report, Biometrics Unit, Cornell University, 1999. Preprint at http://www.maths.surrey.ac.uk/personal/st/S.Brooks/MCMC/.

[6] J.N. Corcoran and R.L. Tweedie. Perfect sampling from independent Metropolis-Hastings chains. Preprint at http://www.maths.surrey.ac.uk/personal/st/S.Brooks/MCMC, 1998.

[7] N. Cressie. *Statistics for Spatial Data*. Wiley, 1991.

[8] J. Damien, P. Wakefield and S. Walker. Gibbs sampling for Bayesian nonconjugate and hierarchical models using auxiliary variable. *Journal of the Royal Statistical Society*, B 61:331–344, 1999.

[9] R.G. Edwards and A.D. Sokal. Generalization of the Fortium-Kasteleyn-Swendsen-Wang representation and Monte Carlo algorithm. *Physical Review*, D 38:2009–2012, 1988.

[10] J.A. Fill. An interruptible algorithm for perfect sampling via Markov chains. *Annals of Applied Probability*, 8:131–162, 1998.

[11] J.A. Fill and M. Machida. Stochastic monotonicity and realizable monotonicity. Technical Report 573, Department of Mathematical Sciences, The John Hopkins University, 1998.

[12] M. Fismen. Exact sampling using Markov chains. Diploma thesis, Department of Mathematical Sciences, Norwegian University of Technology and Science, Trondheim, 1997.

[13] S. Foss and R.L. Tweedie. Perfect simulation and backward coupling. *Stochastic Models*, 14:187–203, 1998.

[14] S. Foss, R.L. Tweedie, and J. Corcoran. Simulating the invariant measures of Markov chains using backward coupling at regeneration times. *Probability in the Engineering and Informational Sciences*, 12:303–320, 1998.

[15] P.J. Green and D. Murdoch. Exact sampling for Bayesian inference: towards general purpose algorithms. In J. M. Bernardo, J.O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 6*, pages 301–321. Oxford University Press, 1999.

[16] O. Häggström, M.N.M. van Lieshout, and J. Møller. Characterization results and Markov chain Monte Carlo algorithms including exact simulation for some spatial point processes. *Bernoulli*, 5:641–659, 1999.

[17] W.S. Kendall. Perfect simulation for spatial point processes. In *Proceedings of the 51st Session of the ISI, Istambul*, 1997.

[18] W.S. Kendall. Perfect simulation for the area-interaction point process. In L. Accardi and C.C. Heyde, editors, *Probability Towards 2000*, pages 218–234, New York, 1998. Springer.

[19] W.S. Kendall and J. Møller. Perfect Metropolis-Hastings simulation of locally stable point processes. Technical Report R-99-2001, Department of Mathematics, Aalborg University, 1999. Preprint at http://www.maths.surrey.ac.uk/personal/st/S.Brooks/MCMC/.

[20] W.S. Kendall and E. Thönnes. Perfect simulation in stochastic geometry. *Journal of Pattern Recognition*, 1998. Special issue on random sets. To appear.

[21] T. Lindvall. *Lectures on the Coupling Method*. John Wiley and Sons, New York, 1992.

[22] S.P. Meyn and R.L. Tweedie. Stability of Markovian processes III: Foster-Lyapunov criteria for continuous time processes. *Advances in Applied Probability*, 25:518–548, 1993.

[23] A. Mira and Tierney L. On the use of auxiliary variables. Technical Report 63 (7–97), University of Pavia, Department of Quantitative Methods, 1997.

[24] A Mira and G.O. Roberts. Perfect slice samplers. Technical Report 102 (4–99), University di Pavia, Department of Quantitative Methods, 1999.

[25] J. Møller. Perfect simulation of conditionally specified models. *Journal of Royal Statistical Society*, B 61:251–264, 1999.

[26] J. Møller and G. Nicholls. Perfect simulation for sample-based inference. Technical Report R-99-2011, Department of Mathematical Sciences, Aalborg University, 1999. Preprint at http://www.maths.surrey.ac.uk/personal/st/S.Brooks/MCMC/.

[27] J. Møller and K. Schladitz. Extensions of Fill's algorithm for perfect simulation. *Journal of the Royal Statistical Society*, B 61:955–969, 1999.

[28] D Murdoch and P.J. Green. Exact sampling from a continuous state space. *Scandinavian Journal of Statistics*, 25:483–502, 1998.

[29] D. Murdoch and J.S. Rosenthal. Efficient use of exact samples. *Statistics and Computing*, 1999. To appear. Preprint at http://www.maths.surrey.ac.uk/personal/st/S.Brooks/MCMC/.

[30] P.H. Peskun. Optimum Monte Carlo sampling using Markov chains. *Biometrika*, 60:607–612, 1973.

[31] J. Propp and D. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9:223–252, 1996.

[32] J. Propp and D. Wilson. 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*, 27:170–217, 1998.

[33] A. Reuter and V. Johnson. General strategies for assessing convergence of MCMC algorithms using coupled sample paths. *Unpublished*, 1995.

[34] G.O. Roberts and J.S. Rosenthal. Shift-coupling and convergence rates of ergodic averages. *Communications in Statistics - Stochastic Models*, 13:147–165, 1994.

[35] G.O. Roberts and J.S. Rosenthal. Convergence of slice samper Markov chains. *Journal of Royal Statistical Society*, B 61:643–660, 1999.

[36] G.O. Roberts and A.F.M. Smith. Simple conditions for the convergence of the Gibbs sampler and Hastings-Metropolis algorithms. *Stochastic Processes and ther Applications*, 49:207–216, 1990.

[37] R.H. Swendsen and J.S. Wang. Non-universal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58:86–88, 1987.

[38] D.B. Wilson. How to couple from the past using a read-once source of randomness. *Random Structures and Algorithms*, 1999. To appear.

[39] D.B. Wilson. Layered multishift coupling for use in perfect sampling algorithms (with a primer on CFTP). To appear in the Fields Institue Proceedings, 1999.