Statistics and Computing (2023) 33:102
https://doi.org/10.1007/s11222-023-10276-6

ORIGINAL PAPER

Latent uniform samplers on multivariate binary spaces

Yanxin Li[1] · **Antonio Linero**[1] · **Stephen G. Walker**[2]

Received: 15 October 2022 / Accepted: 1 July 2023 / Published online: 14 July 2023

Abstract
We consider sampling from a probability distribution on $\{0, 1\}^M$, or an equivalent high-dimensional binary space. A number of important applications rely on sampling from such distributions, including Bayesian variable selection problems and fitting Bayesian regression trees. Direct sampling is prohibitive when the dimension is large due to the fact that there are 2^M possible states. One approach to sampling such distributions is to use a Metropolis–Hastings algorithm, which can require choosing a decent proposal mechanism, with a default choice being the single-component switch proposal move. This is problematic when multiple modes exist. In this paper, we propose a latent variable uniform sampling algorithm, such as a latent slice sampler, which allows for large moves and proposal paths which give non-negligible probabilities for moving between modes, even when the probabilities of states between these modes is low. A number of illustrations are presented, focusing primarily on demonstrating the advantages over current generic samplers.

Keywords Ising model · Latent slice sampler · Markov chain Monte Carlo · Variable selection · Regression trees

1 Introduction

Slice sampling is a powerful technique for generating random variables from complicated density functions; see Besag and Green (1993), Damien et al. (1999), and Neal (2003). The motivation is simple enough; for a target density $\pi(x)$, $x \in \mathbb{R}^M$, the idea is to introduce the latent variable w and consider the joint density $f(x, w) = \mathbf{1}(w < \pi(x))$. A Gibbs sampler can be implemented in which the sampling of w is straightforward and the sampling of x, conditional on w, involves sampling uniformly from the interval $A_w = \{x : \pi(x) > w\}$. It is the sampling of this latter uniform distribution which poses the problem for slice samplers. Indeed, uniform sampling from high dimensional spaces is a problem in its own right; see, for example, Chen et al. (2018). However, the aim would be to achieve this without recourse to complicated MCMC algorithms since this would defeat the object of the slice sampler, and one could well be better off performing a direct MCMC on $\pi(x)$.

✉ Stephen G. Walker
s.g.walker@math.utexas.edu

Antonio Linero
Antonio.Linero@austin.utexas.edu

[1] SDS, University of Texas at Austin, Austin, USA

[2] Department of Mathematics, University of Texas at Austin, Austin, USA

In Neal (2003) a clever procedure for sampling the uniform distribution on A_w using a transition density $f(x' \mid x)$ satisfying $f(x' \mid x, w) = f(x \mid x', w)$, and hence is stationary with respect to the target uniform density. A strategy for proposing a x' given x satisfying the reversible condition is given in Neal (2003). This involves a stepping out or doubling procedure combined with a shrinkage procedure. The former procedures require choosing a width parameter which becomes fixed over the run of the chain and is, particularly in high dimensions when one is required for each dimension, potentially a tricky tuning parameter to set. Further, the stepping out and doubling procedures need to be performed sequentially with a computation of $\pi(x)$ after each step. This makes it difficult to implement in high dimensions.

On the other hand, Li and Walker (2022) introduce further latent variables which facilitates the uniform sampling via a Gibbs framework. The algorithm avoids the stepping out or doubling procedures and hence avoids the need for the tuning width parameters and a potentially slow sequential search for a valid substitute for A_w when the dimensions are large. They start with, writing in the one dimensional case, the joint density

$$f(x, w, s, l) = \mathbf{1}(w < \pi(x))\, s^{-1} p(s)\, \mathbf{1}(|x - l| < s/2) \quad (1)$$

for some density function $p(s)$ on $(0, \infty)$. As before, all variables are easy to sample and now the required den-

sity for x we need to sample is $f(x \mid w, s, l) \propto \mathbf{1}(w < \pi(x))\,\mathbf{1}(l - s/2 < x < l + s/2)$. This structure allows for an easy search for a valid substitute for A_w, just from the sampling of s. It is then also easy to set up a sequence of proposals x' satisfying $f(x' \mid x, w, s, l) = f(x \mid x', w, s, l)$. The algorithm is detailed in Li and Walker (2022); briefly here, take an initial proposal x^* uniformly from the interval $(L_-, L_+) = (l - s/2, l + s/2)$ and keep repeating this until $w < \pi(x^*)$. After each rejection, the uniform interval, currently (L_-, L_+), can be narrowed to (x^*, L_+) if $x^* < x$ or (L_-, x^*) if $x^* > x$, where x is the current value. As the rejections mount, the interval will start to concentrate on x. This is effectively equivalent to the shrinkage procedure described in Neal (2003). Hence, at the very least, the algorithm can become a local sampler, but with the possibility of having large jumps.

The accepted sample x' and the current value are reversible according to this procedure; i.e. $f(x' \mid x, \ldots) = f(x \mid x', \ldots)$. This is one of the ideas behind (Neal 2003), though we avoid the stepping out and doubling parts of his algorithm. Extending the dimensions is straightforward and numerous illustrations are presented in Li and Walker (2022).

Walker (2014) and Ekin et al. (2022) cover the case when $x \in \mathbb{N}^M$. However, a class of problematic density functions is given by

$$\pi(z_1, \ldots, z_M) \tag{2}$$

where each $z_j \in \{0, 1\}$, or an equivalent binary set. The two aforementioned papers do not cover this case; they would both collapse to a Gibbs sampler, which is not in general suitable for the distributions we are going to look at.

When M is large these binary joint distributions can be difficult to sample, particularly if they are multimodal. Typically, Metropolis algorithms (Metropolis et al. 1953) or Gibbs samplers (Tierney 1994) are applied where one of the variables is updated at a time. Such algorithms may not produce sufficiently well-mixing chains, or even a chain that moves at all. We demonstrate through a number of illustrations the mixing abilities of the latent slice sampler for multivariate binary distirbutions. We compare with other generic algorithms, such as the single flip proposal Metropolis algorithm, which is named the Metropolized Gibbs sampler in Schafer (2012). Our algorithm is closely related, but an extension of the random walk kernel sampler, described, for example, in Schafer and Chopin (2013). The algorithm of Schafer and Chopin (2013) is sequential Monte Carlo (SMC). However, there is a necessity for resampling which requires a Metropolis step for each π_t, where t indexes the temperature, which moves to temperature T for which $\pi_T = \pi$. The Metropolis step uses an independence proposal which depends on an estimated parameter, and is not always trivial to estimate, requiring in some instances a Newton-Raphson sequence to find the best proposal density. The paper of Schafer and Chopin (2013) also describes a number of MCMC algorithms and discusses the merits of the two types of algorithm; i.e. SMC and MCMC.

Two recent papers for sampling *continuous* multimodal distributions include (Tak et al. 2018) and Pompe et al. (2020), but we do not believe these are suited to binary variables.

There are models, such as the Ising model, where due to the nature of the distributions, specialized algorithms work, such as the Swendsen–Wang (Swendsen and Wang 1987) and Wolff (Wolff 1989) algorithms. We will develop an alternative to the uniform sampler based on slices for this model. Just as the latent slice sampler provides a uniform conditional density for the variable of interest, this aspect is retained though the uniform conditional density is obtained in a different way.

In this paper, we apply a latent uniform sampling algorithm to the joint density (2). For the latent slice sampler the idea is to introduce a latent variable, say y_j for each z_j, and set $z_j = \mathbf{1}(y_j > 0)$. This gives us a joint density in (y_j) which can be sampled as in Li and Walker (2022). If it is possible to allow the sampler on the (y_j) to move sufficiently around some bounded space in M-dimensions, we should be able to construct a sampler which also jumps around in the $\{0, 1\}^M$ space. For the Ising model we use a different construct to obtain a latent uniform sampler directly on the (z_j).

The layout of the paper is as follows: in Sect. 2 we describe the details of the algorithm. We also provide some theory about the algorithm and prove the reversibility of the sampling of $f(x' \mid w, s, l)$. In Sect. 3 we first present a couple of introductory examples with some further substantial illustrations presented subsequently; in Sect. 3.3 we consider a conditional logistic distribution, Sect. 3.4 a Bayesian variable selection model, and in Sect. 3.5 we consider a Bayesian decision tree model. In Sect. 3.6 we present a comparison of the latent slice sampler and Metropolis algorithms by looking at eigenvalues of transition probability matrices. Section 4 considers the special case of the Ising model and the paper concludes with a brief discussion in Sect. 5.

2 Slice sampling algorithm

Sampling from $\pi(z)$ is equivalent to sampling from the joint density

$$f(y, w, s, l) \propto \mathbf{1}\Big(w < \pi(z_1, \ldots, z_M)\Big) \times \prod_{j=1}^{M} s_j^{-1} p(s_j)\, \mathbf{1}\Big(y_j - s_j/2 < l_j < y_j + s_j/2,\ |y_j| < a\Big),$$

where $z_j = \mathbf{1}(y_j > 0)$, for some $a > 0$. The introduction of the finite a here is to ensure the joint density is proper. As with the continuous case, the variables are all easy to sample, and the $y = (y_1, \ldots, y_M)$ can be sampled jointly, as in the continuous case, and with the shrinking procedure, until the proposal y^* satisfies $w < \pi(z^* = \mathbf{1}(y^* > 0))$. Write $y^* = y_0^*$ as the initial proposal and, if all are rejected, let (y_r^*) be the sequence of proposals.

At each iteration, the initial proposal z^* is being sampled approximately uniformly on $\{0, 1\}^M$. This is equivalent to restarting the chain. However, rather than the chain move aimlessly along points with low probability looking for a point with high probability, it drifts back to the current value, with each interim point being tested for a possible move. If nothing is accepted along the way, the chain stays at its current value and the next iteration proceeds with another uniform sample being generated. Viewed in this way, the algorithm provides a jump mechanism with multiple proposals and if these are all rejected it behaves as a local sampler.

Here we write the algorithm (detailed as a single loop) for the sampler for a given π and with $a = 2$ (the choice of a is without loss of generality), and $p(s) \propto s \exp(-s\lambda)$:

1. Initializing step: Given $s = s_{1:M}$ and $z = z_{1:M}$, and for each $i = 1, \ldots, n$, set $y_i > -a$ negative if $z_i = 0$ and $y_i < a$ positive if $z_i = 1$.
2. Set $a_i = b_i = 0$ for $i = 1, \ldots, n$. Sample $w = u_1 \pi(z)$ and for $i = 1, \ldots, n$, take $l_i = y_i - s_i/2 + u_{2i}\, s_i$, where u_1 is a uniform r.v. and the u_{2i} are i.i.d. uniform r.v.s
3. Sample s_i as an exponential r.v. with mean $1/\lambda$ and constrained to be greater than $2|l_i - y_i|$. Set $a_i = \max\{-a, l_i - s_i/2\}$ and $b_i = \min\{a, l_i + s_i/2\}$.
4. Propose the new y_i' as $a_i + v_i(b_i - a_i)$, where the v_i are i.i.d. uniform r.v.s, and $z_i' = 1(y_i' > 0)$.

   4a. Compute $w' = \pi(z')$.

   4b. If $w' > w$ then $z = z'$ and goto 2.

   4c. Else, set $q_i = 1(y_i' < y_i)$ then reset $a_i = q \max\{a_i, y_i'\} + (1-q)a_i$ and $b_i = (1-q)\min\{b_i, y_i'\} + q b_i$. Goto 4 and repeat until $w' > w$.

An illlustration of the change from y_0 to y_1 in a one dimensional case is presented in Figs. 1 and 2. The step occurs at the origin 0 and the end points are $\pm a$. In the first, both $\pi(0)$ and $\pi(1)$ are greater than w, whereas in the second it is only $\pi(1)$ which is greater than w.

To demonstrate the properties of the sampler, we now show that it can propose any point from any current location with high probability. Due to the independence nature of the proposals, we only need to consider one dimension. Let y_0 and s_0 be the current values and assume without loss of generality that $y_0 > 0$. We take $p(s)$ to be proportional to $se^{-s/\lambda}$ so the conditional for s constrained to be larger than η can be sampled as $\psi + \eta$, where ψ is an exponential random variable with mean λ.

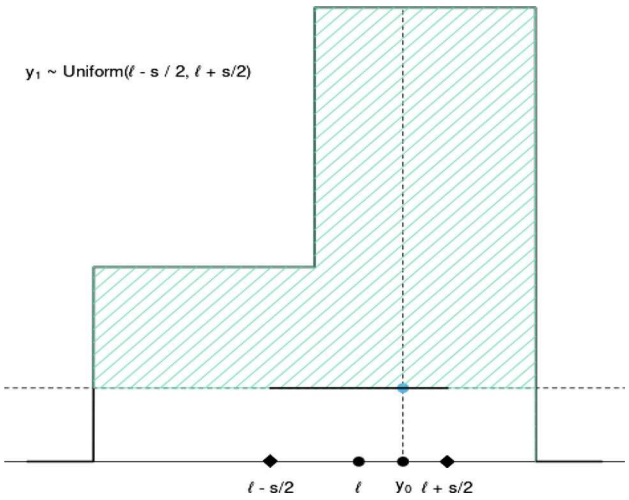

Fig. 1 Latent sampler for $\pi(1),\ \pi(0) > w$

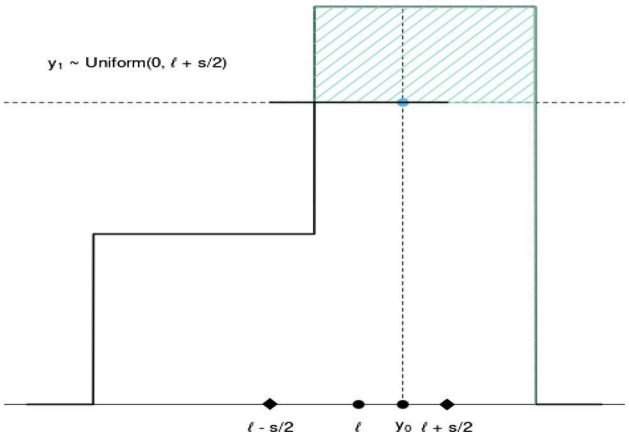

Fig. 2 Latent sampler for only $\pi(1) > w$

Lemma 1 *The probability that the proposal for y, i.e. y^*, is of opposite sign to $y_0 > 0$ is given by*

$$0.5 \max\left\{0, 1 - \frac{y_0 + s_0(2v-1)}{\psi + s_0|2v-1|}\right\},$$

where v is an independent standard uniform random variable

The proof is straightforward. Noting that $0 < y_0 < a$ we see that provided λ is sufficiently large, the probability of y^* being negative when y_0 is positive (and vica versa) can be close to $\frac{1}{2}$.

The role of a is to ensure the density for y from

$$p(y, l, s) \propto s^{-1}\, p(s)\, 1(y - s/2 < l < y + s/2)$$

is proper. Its value can be taken without loss of generality, but the choice of λ would then depend on it. The basic idea is to ensure the proposals for each y can be of either sign. If s_0 and y_0 are the current values, then the initial proposal for

y is from

$$(\max\{a, l - s/2\}, \min\{a, l + s/2\})$$

and in terms of y_0 and s_0 we have $l - s/2 = y_0 - (|c| - c)\, s_0/2 - \epsilon/2$ and $l + s/2 = y_0 + (|c| + c)\, s_0/2 + \epsilon/2$, where $c = 2v - 1$ and v is a standard uniform r.v., and ϵ is the exponential r.v. with mean $1/\lambda$. If we want these to cover both positive and negative values, but not excessively, we would as a rule of thumb take $\lambda = 1/(2a)$. In short, we ensure that the sign of the proposals for the y can switch with high probability.

Before proceeding we focus on the sampling of the $f(y \mid w, s, l)$ via a reversible Markov sequence. To do this we set up a more generic setting for the problem. So let $f(y) \propto \mathbf{1}(y \in C)\, \mathbf{1}(y \in B)$ where C is an unknown interval, but a specific value of y can be tested to see whether it lies in C or not; i.e. $y \in C \iff w < g(y)$ for some continuous function g, and B is a known single connected interval. Define the initial $B = B_0 = (a_0, b_0)$ and let y_0 be the current point which lies in $C \cap B$. The sequence of proposals $(y_r)_{r \geq 1}$ is given by

$$y_r = a_{r-1} + u_r (b_{r-1} - a_{r-1}), \tag{3}$$

where the (u_r) are an independent sequence of standard uniform random variables, and if $y_r \notin C$, the B_{r-1} is updated to B_r via

$$\begin{aligned} a_r &= a_{r-1}\, \mathbf{1}(y_r > y_0) + y_r\, \mathbf{1}(y_r < y_0) \\ b_r &= b_{r-1}\, \mathbf{1}(y_r < y_0) + y_r\, \mathbf{1}(y_r > y_0). \end{aligned} \tag{4}$$

This sequence continues until $y_r \in C$ for some r.

Lemma 2 *If I_r is the current length of the interval from which y_r is taken uniformly, then the size of the next interval is random and $I_{r+1} = u\, I_r$, where u is a uniform random variable from $(0, 1)$ and independent of I_r.*

Proof If the interval $B_r = (a_r, b_r)$, then $y_{r+1} = a_r + u(b_r - a_r)$, and

$$a_{r+1} = a_r\, \mathbf{1}(y_{r+1} > y_0) + (a_r + u(b_r - a_r))\, \mathbf{1}(y_{r+1} < y_0)$$

and

$$b_{r+1} = b_r\, \mathbf{1}(y_{r+1} < y_0) + (a_r + u(b_r - a_r))\, \mathbf{1}(y_{r+1} > y_0).$$

Hence,

$$\begin{aligned} I_{r+1} &= b_{r+1} - a_{r+1} = (b_r - a_r) \\ &\times \Big((1 - u)\mathbf{1}(y_{r+1} > y_0) + u\mathbf{1}(y_{r+1} < y_0) \Big). \end{aligned}$$

This completes the proof. □

Corollary 1 *In one dimension, if $m = \min_r\{I_r / I_0 < \epsilon\}$, then m is a $1 + Pois(-\log \epsilon)$ random variable. In n dimensions, and with the size of interval is represented as I_{ir} for $i = 1, \ldots, n$, then if $m = \min_r\{\max_i I_{ir}/I_{i0} < \epsilon\}$, then m is the largest order statistic of n independent $1 + Pois(-\log \epsilon)$ random variables.*

These two lemmas indicate how the sampler acts as both a jump, almost uniform, and local sampler. And recall that at each iteration as the sampler moves from its initial proposal y^* back to y_0, a new proposal is being made. In short, a sequence of proposals is being generated ranging from a jump proposal to a local proposal, the latter applying if all the jump proposals are rejected.

The next two results establish that our shrinking procedure leaves the posterior distribution invariant. First, we note that Corollary 1 implies that the shrinking procedure will terminate almost-surely for almost-all starting values y. In particular, it will terminate when y is a continuity point of $g(y)$, and the set of discontinuity points of $g(y)$ has Lebesgue measure 0.

Lemma 3 *Let J denote the number of rejected points in the shrinking procedure and suppose that y is a continuity point of $g(y)$. Then J is finite almost-surely.*

Proof With probability 1 we will have $w < g(y)$, and by continuity we will have $w < g(y \pm \epsilon)$ for sufficiently small ϵ. Hence, if the shrinking procedure is eventually contained in an ϵ-neighborhood of y the procedure will terminate. Corollary 1 implies that the time for this to occur is Poisson distributed, and hence finite almost surely. □

Theorem 1 *The shrinking procedure defined by (3) and (4) defines a Markov transition function $Q(y \mid y', w, s, l)$ which is* reversible *in the sense that $f(y \mid w, s, l)\, Q(y' \mid y, w, s, l) = f(y' \mid w, s, l)\, Q(y \mid y', w, s, l)$.*

Proof For simplicity, we will consider only the case of univariate y; the proof for multivariate y is essentially the same. Also, we suppress dependence of Q on w, s, l to lighten notation. First, we note that $f(y \mid w, s, l)$ is uniform on the set $\{y : w \leq g(y), l \in [y - s/2, y + s/2], |y| \leq a\}$. If either y or y' are outside of this set, we will have $f(y \mid w, s, l)\, Q(y' \mid y) = f(y' \mid w, s, l)\, Q(y \mid y') = 0$ trivially, so assume without loss-of-generality that this is not the case.

Following Neal (2003), we let $r = (r_1, \ldots, r_J)$ denote the sequence of rejected points in the shrinking procedure; by Lemma 3, r is a random vector of finite length. Let $Q(y', r \mid y)$ denote the transition density of moving from y to y' via the intermediate rejected points r; formally, $Q(y', r \mid y)$ is a density with respect to $dy \times \sum_{j=0}^{\infty} \lambda_j(dr)\, I(J = j)$ where λ_j denotes Lebesgue measure on $\mathbb{R}^J$. To show reversibility, it suffices to establish the stronger result that $Q(y', r \mid y) = Q(y, r \mid y')$ for all r. To show $Q(y', r \mid y) = Q(y, r \mid y')$,

we first consider the case that some r_j lies in between y and y'. In this case, the shrinking procedure starting from y will eliminate y' as a potential value, and vice-versa. Hence $Q(y', r \mid y) = Q(y, r \mid y') = 0$ in this case. Otherwise, by the uniformity of the sampling, we have $Q(y, r \mid y') = Q(y', r \mid y) = \prod_{j=0}^{J}(b_j - a_j)^{-1}$ where (a_0, b_0) is the starting interval, (a_1, b_1) is the interval after rejecting the joint r_1, and so forth. Hence $Q(y, r \mid y') = Q(y', r \mid y)$.

The logic behind extending this proof to the multivariate setting is essentially the same: we again introduce the set of intermediate moves r, where it will only be possible to transition from y to y' if none of the rejected proposed points $y_j^\star$ for coordinate j lies in between y_j and y'_j, and in this case the probability of transitioning from y to y' via r is the same as transitioning from y to y' via r by uniformity. □

3 Illustrations

Here we present a number of illustrations, starting with two simple expository examples. We then move to more substantive cases involving high dimensional models, including a conditional logistic distribution, a variable selection model, and a Bayesian decision tree model. In many cases we compare with the Metropolis–Hastings algorithm. We refer the reader to the paper (Li and Walker 2022), where in Sect. 3, the authors provide a detailed comparison of their slice sampling algorithm with the slice sampler of Neal (2003). In particular, they make comparisons involving EES. Results should be transferable because the latent slice sampler and the algorithm of the present paper share key properties in terms of how elements of the proposals are made.

3.1 Example 1

To demonstrate the accuracy of the algorithm we present a simple example where M is small enough so we know exactly the 2^M probabilities. We take $M = 3$ and

$$\pi(z_1, z_2, z_3) = e^{z'Az} / \sum_{z \in C} e^{z'Az}$$

where C is the set of 8 possible values of z. The matrix A is randomly generated with independent standard normal random variables. The matrix A is

$$A = \begin{pmatrix} -0.322 & -0.314 & -1.541 \\ 0.332 & 1.109 & -0.909 \\ -0.391 & 0.213 & 0.118 \end{pmatrix}$$

and the correct probabilities are $\pi_{0,0,0} = 0.099$, $\pi_{0,0,1} = 0.111$, $\pi_{0,1,1} = 0.168$, $\pi_{1,1,1} = 0.018$, $\pi_{1,0,1} = 0.012$, $\pi_{0,1,0} = 0.300$, $\pi_{1,0,0} = 0.072$, $\pi_{1,1,0} = 0.221$. The algorithm was run for 100,000 iterations and the estimated probabilities are $\widehat{\pi}_{0,0,0} = 0.097, \widehat{\pi}_{0,0,1} = 0.113, \widehat{\pi}_{0,1,1} = 0.171, \widehat{\pi}_{1,1,1} = 0.018$, $\widehat{\pi}_{1,0,1} = 0.012, \widehat{\pi}_{0,1,0} = 0.295$, $\widehat{\pi}_{1,0,0} = 0.072, \widehat{\pi}_{1,1,0} = 0.221$. The mixing is excellent; as an illustration we present a plot of the first 100 samples of the (z_1) variable in Fig. 3.

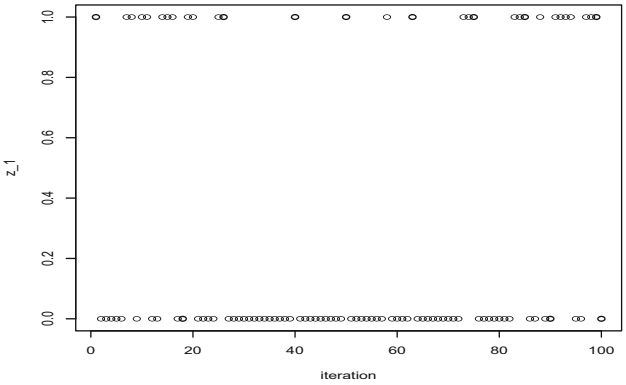

Fig. 3 Plot of first 100 samples of z_1

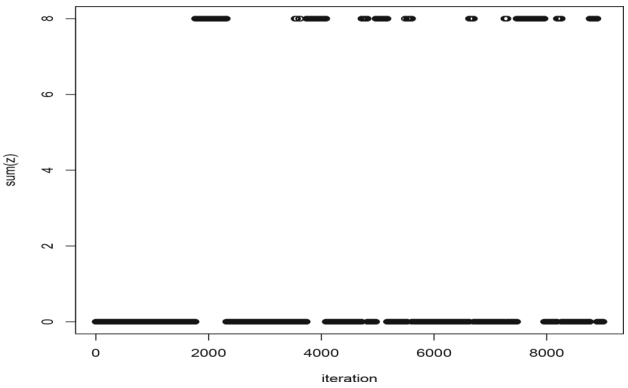

Fig. 4 Plot of sum of components of z vector

The choices for the algorithm include $p(s)$ and a. The idea is to enable the intervals $(l - s/2 < y < l + s/2 \cap |y| < a)$ to be large; therefore a is not such an important choice, and we fix it at 2, while to ensure the largest intervals we take $p(s) \propto se^{-\lambda s}$ with $\lambda = 0.05$. Note then that the sampling of the s within an iteration is an exponential random variable with parameter λ added to $2|y - l|$.

3.2 Example 2

Another example, but a demanding one, is taking $M = 8$ and $\log \pi(z_j \equiv 1) = \log \pi(z_j \equiv 0) \propto 100$ with all the other vectors for z have $\log \pi(z) \propto 1$, so the probabilities differ by 100 on log scale. This distribution is bimodal with no route via local sampling from one to the other. Indeed, any local sampler would fail to move from one of the modes once there.

This illustration is as difficult for local samplers as it possibly can be for distributions on $\{0, 1\}^M$. There are two separated modes with single points and with all other probabilities effectively 0. The only way to be able to jump between modes in this case is to have uniform proposals. Our algorithm has this as a key component. Needless to say, the Metropolis sampler or Gibbs sampler does not switch modes once one has been reached.

The algorithm mixes over the two modes well; see Fig. 4. The vertical axis represents the sum of the components of the z vector which has modes at 0 and m. A pure local sampler would of course not leave a mode once reached. To test this illustration to an extreme, we set $M = 20$. On a number of runs of size 10^6 we get at least one switch between the two modes. Note that 2^{20} is just over 1,000,000. Hence, the nature of the sampler is as if we restart the chain randomly at a location for each iteration. However, instead of the chain then moving locally about this location, it moves—with proposals at each step, which can be accepted—towards the previous location and hence can then at least mimic a local sampler.

3.3 Logistic conditional distribution

The joint distribution on $(z_1, \ldots, z_M)$ here is given by $\pi(z) = \prod_{i=1}^{M} p_i^{z_i}(1-p_i)^{1-z_i}$, where $p_i = p(z_{1:i-1}) = [1+\exp(-s_i)]^{-1}$ and $s_i = a_{ii} + \sum_{j=1}^{i-1} a_{ij}\, z_j$ with $A = (a_{ij})_{j\le i}$ a lower triangular matrix. With $i = 1$ we have $s_1 = a_{1,1}$. This joint binary distribution appears in the PhD Thesis of Schafer (Schafer, 2012).

The aim here is to compare the latent slice sampler with the Metropolis single flip algorithm. This is named the Metropolized Gibbs sampler in Schafer (2012) and in general can be shown to be an improvement on the Gibbs sampler; see Liu (1996). An alternative algorithm in which proposals z' are made uniformly on $|z' - z| < k$ for some k, Schafer and Chopin (2013), can be seen as a special case of our own algorithm, in that we have a framework in which k can be made random for each iteration.

The elements of the A matrix are generated using independent uniform random variables from the interval $(-2, 2)$. We take $M = 30$, so the size of space of possible outcomes is 2^{30} which is approximately 1000 million. To assess how different sampling algorithms work in this example, it is noted that the joint distribution can be sampled exactly using the sequence of conditional distributions $p(z_i \mid z_{1:i-1})$. We therefore ran three chains for 5, 000 iterations each; the correct sampling algorithm, the latent slice sampler, and the Metropolis algorithm.

We focus on the estimation of the correct mean values from the samples, taking as the benchmark the means from the samples from the correct conditional sampler. The results are presented in Fig. 5. As can be seen, from this perspective, the latent slice sampler significantly outperforms the Metropolis sampler. In fact, for some of the (z_i), specifically for those $i > 7$, the Metropolis sampler did not move from the starting value.

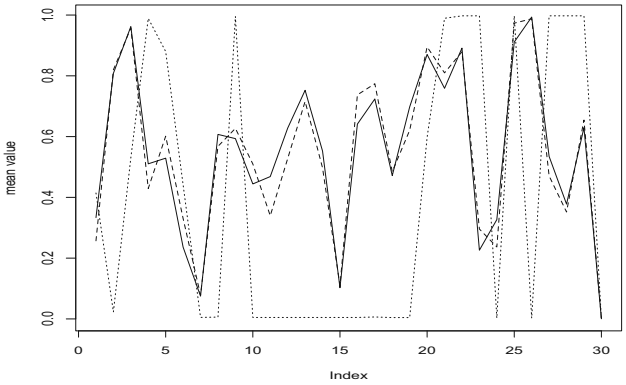

Fig. 5 Means from Metropolis algorithm (dotted line), slice sampler algorithm (dashed line) and the true means (bold line)

For this illustration we took $p(s) \propto s\exp(-s/10)$ and $a = 2$. At each iteration of the chain, the latent slice sampler makes a single jump proposal. As it shrinks back to the current value, with proposals being rejected, it makes new proposals as it goes. The average number of proposals per iteration done this way is 6. Hence, the total number of proposals is 6×5000. On the other hand, the Metropolis sampler makes M proposals, one for each ordinate, per iteration, resulting in the total number of 30×5000 proposals. Despite the more proposals coming from the Metropolis sampler, the slice sampler made more moves. Timewise, the algorithms take roughly the same amount of time—a matter of seconds for both.

3.4 Bayesian variable selection

In this subsection we obtain a joint distribution for the variable selection indicators of a linear model. The model is given by

$$y_i = \sum_{j=1}^{p} x_{ij} z_j \beta_j + \sigma\epsilon_i, \tag{5}$$

where the $z = (z_j)$ are the indicators, taking the values 0 or 1, and the (ϵ_i) are assumed to be independent standard normal. This model was first proposed in Kuo and Mallick (1998) as an alternative framework to the hierarchical model of George and McCulloch (1993). We adopt a slightly different prior set up compared to that of Kuo and Mallick (1998). We write the likelihood, using $\lambda = 1/\sigma^2$, as $\lambda^{n/2}\exp\left\{-\frac{1}{2}\lambda(y - X\beta)'(y - X\beta)\right\}$ where $X = X_0 Z$ with X_0 the design matrix based on the (x_{ij}) and $Z = \mathrm{diag}(z_j)$. We take a g-prior (Zellner 1983) for β; so for some $g > 0$,

$\beta \sim \mathrm{N}\left(0, g\sigma^2(X'X)^{-1}\right)$. If $Z \equiv 0$ then $\beta = 0$ which is compatible with the idea that no predictors are active. The prior for λ is taken to be gamma with parameters (a, a).

The aim now is to find the marginal posterior distribution for z given the data. This involves some straightforward integration. First

$$p(y \mid \lambda, x, z) \propto \lambda^{n/2+a-1} \exp\left\{-\lambda(a + \tfrac{1}{2}y'y)\right\} \left(\frac{g}{1+g}\right)^{|z|/2} \times \exp\left\{\tfrac{1}{2}\frac{\lambda}{1+g} y' H_X y\right\},$$

where H_X is the hat matrix corresponding to the X_0 and Z; effectively removing the columns for which the $z = 0$, and $|z|$ is the number of $\{z = 1\}$. Hence, assuming a uniform prior for z, we get

$$\pi(z \mid y, x) \propto \left(\frac{g}{1+g}\right)^{|z|/2} \left\{a + \tfrac{1}{2}y'y + \tfrac{1}{2}y' H_X y/(1+g)\right\}^{-a-n/2}.$$

On the other hand, Kuo and Mallick (1998) employed a MCMC algorithm which worked as a Gibbs sampler and sampled the conditional distributions of β, z and λ.

When the distribution $\pi(z \mid y, x)$ is unimodal both the latent slice sampler and Metropolis algorithms work well. The latter, using single move proposals, mixes slightly better, though all the marginal probabilities of the (z_j) are estimated exactly the same. To illustrate this we take a sample of size $n = 100$ and $p = 3$, $\sigma = 1$, and the design matrix elements are taken as independent standard normal. The true value of β is $\beta = (0.3, -0.3, 0)$. We ran the slice sampling algorithm for 10000 iterations and the means of the sampled indicator variables were $\bar{z}_1 = 0.359$, $\bar{z}_2 = 0.988$, $\bar{z}_3 = 0.090$. With the same dataset, we ran a Metropolis algorithm also over 10000 iterations. One iteration here involves proposing a flip of each indicator variable and the Metropolis accept/reject criterion is used to determine whether the flip occurs or not. The corresponding sampled means are $\bar{z}_1 = 0.359$, $\bar{z}_2 = 0.984$, $\bar{z}_3 = 0.097$, which are essentially the same as those from the slice sampler.

In Fig. 6 we illustrate the sampled indicator variable z_1 for both the slice sampling algorithm (top) and the Metropolis algorithm (bottom) over a period of 100 iterations. It can be seen the Metropolis algorithm mixes better than the slice sampling algorithm. However, in this simple case the local sampler is effective as the distribution of z is well behaved and nicely unimodal. The slice sampling algorithm acts as both a local and global sampler, explaining the differences.

However, when $\pi(z \mid x, y)$ is bi-modal, the mixing of the latent slice sampler is superior due to its ability to make large jumps in the z-space. A bimodal distribution can be arranged and can also occur naturally when there is high co-linearity

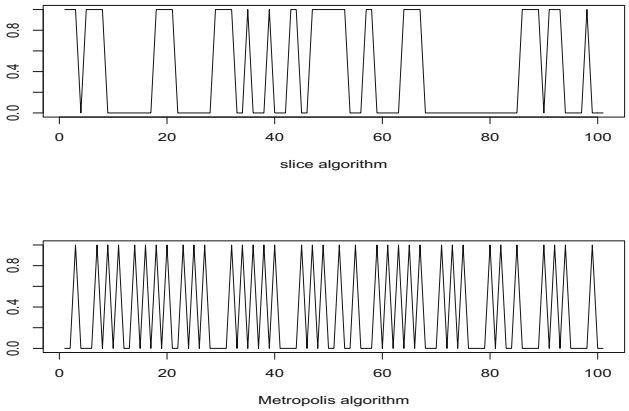

Fig. 6 Comparison of mixing of slice algorithm and Metropolis algorithm

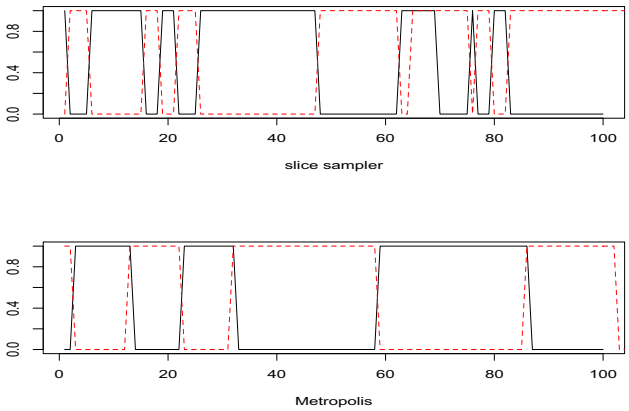

Fig. 7 Comparison of switching between modes for latent slice sampler (top) and Metropolis (bottom)

between predictor variables. To describe the experiment, we take $n = 100$ and $p = 10$, and only one predictor is active, say $x_1 = (x_{11}, \ldots, x_{1p})$, where the (x_{1j}) are taken as independent standard normal. The true $\beta_1 = 5$ and we generate the data with $\sigma = 1$. To create co-linearity we take $x_{2j} = 0.99\, x_{1j} + 0.01\, \xi_j$, with the (ξ_j) as independent standard normal. Hence, the $\pi(z \mid y, x)$ is bi-modal at $z = (1, 0, 0, \ldots)$ and $(0, 1, 0, \ldots)$ with approximately equal weight for each. Indeed, for both the slice sampling algorithm and the Metropolis algorithm, the mean values of the z_1 and z_2 are 0.59 and 0.47, respectively.

However, for the single move Metropolis algorithm, the only way from one mode to the other is to go via $(1, 1, \ldots)$. The probability of this combination is 0.05 and it is this probability which determines the mixing ability of the Metropolis algorithm. For example, over 100 iterations, we would expect 5 switches. This is demonstrated in Fig. 7. The bold lines are the z_1 values and the lines in red are the z_2 values. As is seen the number of switches for the Metropolis algorithm is 6, while for the latent slice sampler it is 13, since for this

algorithm the number of switches does not depend on the probability of $\pi(1, 1, \ldots)$.

If the probability of $\pi(1, 1, \ldots)$ becomes too small then the ability of the Metropolis algorithm to move between the two modes becomes increasingly improbable. To make this point we take $p = 2$ and the value of g as 10^{-6} with all other settings remaining the same. This makes the $\pi(1, 1)$ probability very small. The latent slice sampler chain mixes well and the mean values for z_1 and z_2 are 0.504 and 0.406, respectively. On the other hand, the corresponding values for the Metropolis sampler are 1 and 0, respectively, indicating the chain is fixed at one of the modes. See also Example 2 in Sect. 3.2. The slice sampler can generate effectively uniform proposals in z space that, if rejected, set up a sequence of proposals contracting back to the current point; on the way back, we can then accept small local moves. So if the Metropolis chain of only local moves based on flips of a single z are switched to a uniform proposal to solve the bottleneck problem, the inferiority to the slice sampler becomes very apparent in that now the probability of a small move is becoming negligible.

In Sect. 3.6 we discuss the mixing of the two types of chain via transition matrices in z space. By only considering 4 states we can easily compute the second largest eigenvalues of each transition matrix. Generally speaking, the second largest eigenvalue quantifies the mixing of the chain, with smaller eigenvalues corresponding to faster mixing chains. The Metropolis chain has the bottleneck which creates a large second-largest-eigenvalue, whereas the jumping potential of our latent slice sampler allows the second largest eigenvalue to be small. Though the setting is zooming in on a few states, the problem is going to be the same whatever the overall dimension of the z space is.

3.5 Bayesian decision tree methods

Methods based on Bayesian decision trees, and ensembles thereof, have recently seen remarkable success across a broad range of applications, including causal inference (Hill 2011), survival analysis (Basak et al. 2020) and density regression (Li et al. 2020). For a review of Bayesian decision tree methods, see Linero (2017). Let $g(x; \mathcal{T}, \mathcal{M})$ denote a *regression tree* such that $g(x; \mathcal{T}, \mathcal{M}) = \mu_\ell$ if x is associated to leaf node ℓ of *decision tree* $\mathcal{T}$. Associated to the regression tree defined by the decision tree $\mathcal{T}$ and the leaf node parameters $\mathcal{M} = \{\mu_\ell : \ell = 1, \ldots, L\}$ is a partition of the predictor space; see Fig. 8 for a schematic depiction of this. Bayesian regression tree models, such as the Bayesian CART (Chipman et al. 1998; Denison et al. 1998), estimate $\mathcal{T}$ via Metropolis–Hastings. The proposals used in practice are typically local in nature: we can convert a leaf node to a branch with two new children (BIRTH), delete a leaf-pair and convert their parent to a leaf (DEATH), change the splitting rule

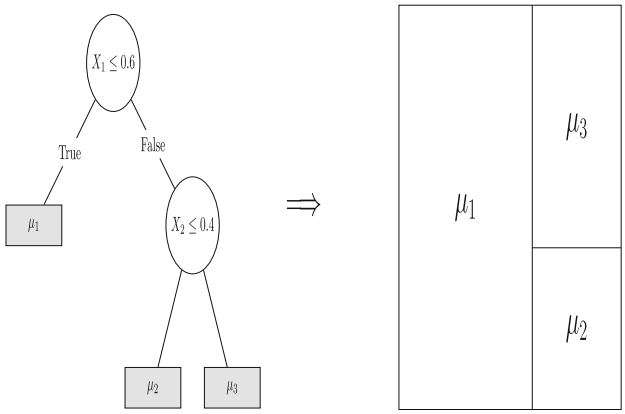

Fig. 8 Left: an example of a decision tree $\mathcal{T}$ with leaf node predictions $\{\mu_1, \mu_2, \mu_3\}$. Right: the induced partition of $[0, 1]^2$

of a branch (CHANGE), or swap the decision rules of two neighboring branches (SWAP).

A challenge for inference with Bayesian decision trees is that the posterior distribution frequently has several well-separated modes. To make this point, consider the "checkerboard" setting in Fig. 9. The response Y_i is sampled from a $\mathrm{N}(g_0(X_i), 0.1^2)$ distribution where $g_0(x) = (\lfloor 4x_1 \rfloor + \lfloor 4x_2 \rfloor) \bmod 2$. The function $g_0(x))$ can be represented exactly using a regression tree; an example of such a tree is given in Fig. 10. However, due to permutation invariance of $g_0(x)$, the decision tree in Fig. 10 is not the only decision tree capable of capturing $g_0(x)$, and in fact there are many equivalent ways of partitioning $[0, 1]^2$ in a way which is consistent with $g_0(x)$.

What makes the posterior of a decision tree difficult to sample for this particular choice of $g_0(x)$ is that (i) typical Metropolis moves for decision trees, such as the BIRTH/DEATH/SWAP/CHANGE moves described by Chipman et al. (1998) are only capable of making small changes to the tree structure but (ii) moving to a different mode of the posterior, which is associated to a vastly different tree topology, must essentially be done in a single step. In other words, all local modifications to the tree result in models of exceedingly low posterior probability, so that all standard Metropolis–Hastings moves will have acceptance probability near 0. While this example is admittedly extreme, the general phenomenon of regression tree posteriors having many highly-separated modes is typical of practice (Chipman et al. 1998).

We show that the discrete latent slice sampler is capable of switching modes of the posterior of our decision tree, despite the fact that the sampler will make little use of the underlying structure of the space of decision trees. For simplicity, we will assume that the decision tree makes cuts only at the midpoints of each coordinate of the hypercube associated with a given branch. Let D denote an a-priori specified maximal depth

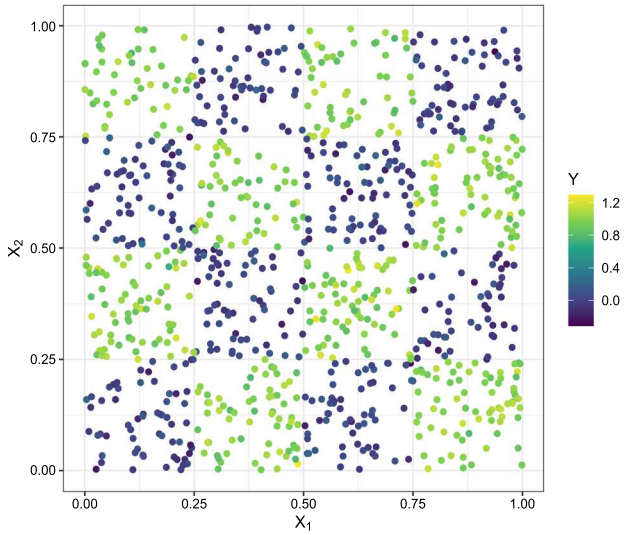

Fig. 9 Samples of a response $Y_i = g_0(X_i) + \epsilon_i$ with $\epsilon_i \sim \mathrm{N}(0, 0.1^2)$ plotted against the predictors X_{i1} and X_{i2}

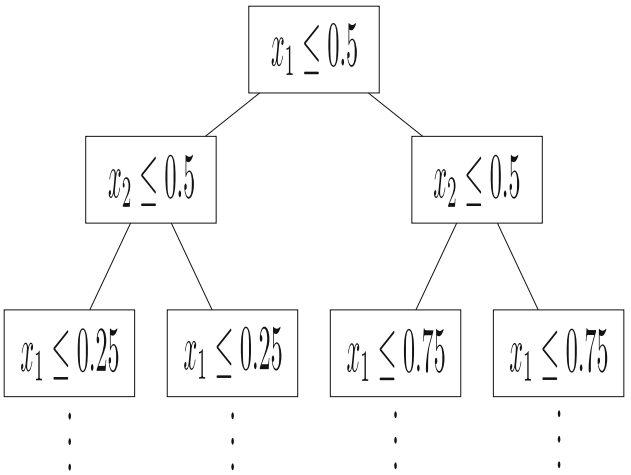

Fig. 10 Example of a regression tree which is capable of exactly calculating $g_0(x)$

of the decision tree. We represent a decision tree over two variables (x_1, x_2) with a pair of binary vectors v, u of length $2^{D+1} - 1$ (the maximal number of nodes of the tree), the first of which indicates whether the associated node of the tree is a leaf or a branch, and the second of which indicates which variable the node will split on. We choose our prior on $\pi(\mathcal{T}, \mathcal{M})$ so that $\mathcal{T}$ has a uniform distribution over the collection of all possible decision trees of maximal depth $D = 4$ with $\mu_\ell \sim \mathrm{N}(0, 300^2)$. We then applied the latent slice sampler to sample from $\pi(\mathcal{T} \mid Y_1, \ldots, Y_n)$ with μ_ℓ integrated out using via conjugacy of the normal distribution.

Figure 11 displays the mixing of the variables v_1, v_2, v_4 (the splitting variables for several branches, including the root), and the marginal log-likelihood of the tree for the model. The fact that the log-likelihood is constant is not a

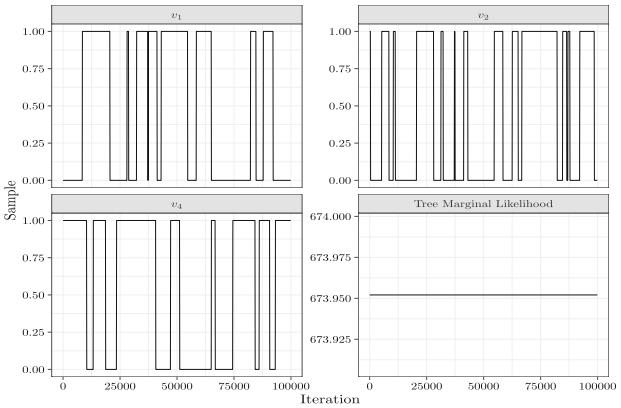

Fig. 11 Traceplots of the samples of the variables v_1, v_2, v_4, and the log-marginal likelihood of $\mathcal{T}$ produced from the latent slice sampler

concern, as the only trees with non-negligible posterior mass are the trees which exactly partition $g_0(x)$. We see, however, that the latent slice sampler is capable of navigating across modes without ever making a move which lowers the marginal likelihood. This behavior would be impossible if we made local Metropolis–Hastings modifications to the tree, because these modifications would inevitably result in the log-likelihood decreasing substantially.

We conclude from this experiment that the latent slice sampler provides an intriguing possibility for performing MCMC on binary decision trees. Further modifications are necessary to apply this approach to the decision tree models used in practice—in particular, we require a latent slice sampler which can accommodate both continuous and discrete variables. We leave such modifications to future work.

3.6 Eigenvalues

Consider the joint probability mass function $\pi(z_1, z_2)$ with $z_1, z_2 \in \{0, 1\}$ and $\pi(0, 0) = \pi(1, 1) = \frac{1}{2} - \epsilon$ and $\pi(0, 1) = \pi(1, 0) = \epsilon$, for some small ϵ.

The Metropolis transition matrix obtained from proposing a flip of a z_j, $j = 1, 2$, with probability $\frac{1}{2}$ each is given by

$$P_M = \begin{pmatrix} 1 - \frac{2\epsilon}{1-2\epsilon} & \frac{\epsilon}{1-2\epsilon} & \frac{\epsilon}{1-2\epsilon} & 0 \\ 1/2 & 0 & 0 & 1/2 \\ 1/2 & 0 & 0 & 1/2 \\ 0 & \frac{\epsilon}{1-2\epsilon} & \frac{\epsilon}{1-2\epsilon} & 1 - \frac{2\epsilon}{1-2\epsilon} \end{pmatrix}.$$

Here row 1 corresponds to $(0, 0)$, row 2 to $(0, 1)$, row 3 to $(1, 0)$ and row 4 to $(1, 1)$ with the same ordering for the columns. It is straightforward to confirm $\pi\, P_M = \pi$.

As can be seen, to arrive at $(1, 1)$ from $(0, 0)$, for example, the chain must pass through either $(0, 1)$ or $(1, 0)$, yet the probability of such a move is small. Hence there is a

bottleneck separating (1, 1) from (0, 0). This will hinder convergence of the chain, which can be measured in terms of the eigenvalues of P. Indeed, the largest eigenvalue is 1, and the second largest eigenvalue contributes to the convergence rate: the closer it is to 1, the slower the rate. The distance between π and π_k, where π_k is the probability mass function of z at iteration k, depends on λ_2^k, where λ_2 is the second largest eigenvalue of the transition matrix. See for example Diaconis and Strook (1991). The second largest eigenvalue is given by 0.89 when e.g. $\epsilon = 0.05$ and is 0.98 when $\epsilon = 0.01$.

When we consider the transition matrix for the latent slice sampler, we assume that the proposal is uniform, i.e. for any z the proposal for the new z is uniform over the 4 states. This is based on the idea that the y_j^*'s are positive or negative with probability roughly $\frac{1}{2}$ for $j = 1, 2$. For simplicity, we consider a simplified transition matrix with strictly inferior mixing compared to our latent slice sampler; specifically, we ignore the multiple proposals possible during the proposed states return to the current state. Such neglect of multiple proposals only occurs for moves from either (0, 0) or (1, 1) to either (0, 1) or (1, 0).

For a proposal from e.g. (0, 0) to (0, 1), acceptance occurs when $\epsilon > v\,(\frac{1}{2} - \epsilon)$ where v is a standard uniform random variable. Hence the probability of acceptance is $2\epsilon/(1-2\epsilon)$; conversely, the probability of accepting a transition from e.g. (0, 1) to (0, 0) is 1, assuming that $\epsilon < \frac{1}{4}$. Hence, the inferior mixing transition matrix for the latent slice sampler in this case is given by P_S equal to

$$\begin{pmatrix} \frac{1}{4} + \frac{1}{2}(1 - \frac{2\epsilon}{1-2\epsilon}) & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & 1/4 & 1/4 & 1/4 \\ 1/4 & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & \frac{1}{4}\frac{2\epsilon}{1-2\epsilon} & \frac{1}{4} + \frac{1}{2}(1 - \frac{2\epsilon}{1-2\epsilon}) \end{pmatrix}.$$

The second largest eigenvalues are given by 0.44 when $\epsilon = 0.05$ and by 0.49 when $\epsilon = 0.01$, which is approximately half the values from the Metropolis sampler.

To illustrate the theory, we simulate the latent slice sampler algorithm with $\epsilon = 0.05$. We take $a = 2$ and $p(s)/s$ to be an exponential density with mean 20. Over a run of 10,000 iterations, the estimated transition probability from (0, 0) to (0, 1) is given by 0.0283, whereas the value within P_S is evaluated at 0.0278, which is smaller than the estimated value, yet extremely close to it.

Further, to investigate the assumption of uniform proposals, we recorded the number of first proposals to be 0 for z_1. Out of the 10000 from the run of the chain, 5003 were 0. See also Lemma 1 for theoretical support for uniform proposals.

4 The Ising model

In this section we consider the special case of the Ising model and a new latent uniform sampler for sampling it. The probability model associated with the two dimensional Ising model (Ising 1925) has joint density on a two dimensional lattice given up to a constant of proportionality by

$$p(s) \propto \exp\left\{ J \sum_{<i,j>} s_i\, s_j \right\}.$$

The square lattice is of size d and the number of (s_i) is $D = d^2$; one located at every grid point within the lattice. The sum is over neighboring grid points and so, for example, if the indices run from left to right along the rows from first to last then $< i, j >$ includes $(1, 2)$, $(1, d+1)$, $(d, 2d)$, and so on. Further, each (s_i) is associated with a spin which is either $+1$ or -1. Here the $J > 0$ is the strength of interaction, usually scaled and represented by $J/(kT)$, where k is the Boltzmann constant and T is the temperature. However, for the purposes of the paper, we will simply use J.

The problem, which has attracted significant attention and which is the focus of the present paper, is how to sample from $p(s)$. The original sampling algorithm was the Metropolis; see Metropolis et al. (1953), in which a single point is proposed to be flipped in sign. The acceptance probability of the proposal is determined by the usual Metropolis acceptance probability based on the joint density function value of the current state and the proposed state. This algorithm can run into problems if the J is too large; see, for example, Diaconis and Saloff-Coste (1998).

The most popular algorithm is provided by Wolff (1989) which is a variant of the original cluster flipping algorithm presented in Swendsen and Wang (1987). The Wolff algorithm is popular due to its effectiveness and simplicity to code. An iteration proceeds by selecting an index at random and then taking all connected paths emanating from this point which have the same spin as the chosen point. Along each path, the path continues with probability $1 - \exp(-2J)$ or is terminated with probability $\exp(-2J)$. All points on the sampled paths are then flipped sign. A recent review is provided in Landau and Binder (2015).

The algorithm is a reversible Markov chain; in the sense that the proposed move $p(s' \mid s)$ which is always accepted satisfies $p(s)\, p(s' \mid s) = p(s')\, p(s \mid s')$. The reason why the Wolff algorithm works is quite straightforward to understand. The number of $1 - \exp(2J)$ in $p(s \mid s')$ and $p(s' \mid s)$ will be the same, the number of them being the number of different spins between s and s'. Then whereas $p(s)$ will have a $\exp(+J)$ to represent two neighboring points having the same spin, if one of the points gets rejected there will arise a $\exp(-2J)$ in $p(s' \mid s)$. And in $p(s')$ there will be a

$\exp(-J)$ due to the opposite spins for the two points, and the two points do not appear in $p(s \mid s')$ as they are of opposite spins.

4.1 Latent uniform sampler

We first transform to z variables on $\{0, 1\}$ by taking $z_i = (s_i + 1)/2$, the resulting model would then be classified as an occupancy model or a lattice gas model. Then

$$\pi(z) \propto \exp\left\{ J \sum_{<i,j>} (2z_i - 1)\,(2z_j - 1) \right\}$$

which can be written as

$$\pi(z) \propto \exp\left\{ J \sum_{<i,j>} 2z_i\, z_j \right\} \times \exp\left\{ J \sum_{<i,j>} 2(1 - z_i)\,(1 - z_j) \right\}.$$

Since both terms are non-negative we can introduce latent variables k_1 and k_2 and construct the joint density

$$p(z, k_1, k_2) \propto \frac{1}{k_1!}\left(J \sum_{<i,j>} 2z_i\, z_j \right)^{k_1} \times \frac{1}{k_2!}\left(J \sum_{<i,j>} 2(1 - z_i)\,(1 - z_j) \right)^{k_2}.$$

This clearly has the correct marginal for z by summing out the ks over the non-negative integers. Further, it is easy to see that k_1 conditional on z is Poisson with mean $J \sum_{<i,j>} 2z_i\, z_j$, with $k_1 = 0$ if the sum is zero. A similar Poisson conditional distribution holds for k_2.

To organize cluster flips for the z we introduce further variables. So consider the term

$$q_{k_1}(z) = \left(J \sum_{<i,j>} 2z_i\, z_j \right)^{k_1}$$

which we extend to

$$p(z, \alpha_l, \beta_l, l = 1, \ldots, k_1 \mid k_1) \propto \prod_{l=1}^{k_1} H_{\alpha_l,\beta_l}\, z_{\alpha_l}\, z_{\beta_l},$$

where $H_{\alpha,\beta} = 1$ if (α, β) belongs to the sum $< i, j >$, and is otherwise 0. Marginalizing over the (α_l, β_l) returns the $q_{k_1}(z)$. We do the same for the $q_{k_2}(z)$ term and then have

$$p(z, \alpha, \beta, \gamma, \delta \mid k_1, k_2) \propto \prod_{l=1}^{k_1} H_{\alpha_l,\beta_l}\, z_{\alpha_l}\, z_{\beta_l} \times \prod_{l=1}^{k_2} H_{\gamma_l,\delta_l}\,(1 - z_{\gamma_l})\,(1 - z_{\delta_l}).$$

Sampling the $(\alpha_l, \beta_l, \gamma_l, \delta_l)$ conditional on the z and k_1 and k_2 is straightforward; for example, (α_1, β_1) is sampled uniformly from the set $A_1 = \{(i, j) :\ H_{i,j} = 1,\ z_i = z_j = 1\}$. If this set is empty then k_1 is 0. All the other (α_l, β_l) follow likewise, while the (γ_l, δ_l) come independently and uniformly from the set $A_0 = \{(i, j) :\ H_{i,j} = 1,\ z_i = z_j = 0\}$.

Note in particular that

$$\prod_{l=1}^{k_1} H_{\alpha_l,\beta_l}\, z_{\alpha_l}\, z_{\beta_l} \prod_{l=1}^{k_2} H_{\gamma_l,\delta_l}\,(1 - z_{\gamma_l})\,(1 - z_{\delta_l})$$

must be equal to 1, and that both separate products must be 1. This forms the basis for the cluster flips. We can now take all the z_i for which $i \in \{\alpha_l\} \cup \{\beta_l\}$ and change $z_i \to 1 - z_i$, and also perform the same flip to all the z_i in $i \in \{\gamma_l\} \cup \{\delta_l\}$. The full proposal which gets accepted automatically would also involve switching the k_1 and k_2.

A single iteration of the algorithm proceeds as follows: starting with (z_i).

1. Sample independently k_1 and k_2 from the Poisson distributions.
2. Sample $((\alpha_l, \beta_l),\ l = 1 : k_1)$ and $((\gamma_l, \delta_l),\ l = 1 : k_2)$ uniformly and independently from the sets A_1 and A_0, respectively.
3. Flip the z_i to $1 - z_i$ if the index i appears as any of the $(\alpha_l, \beta_l, \gamma_l, \delta_l)$.
4. For all (z_i) for which i is not dealt with in 3., sample as an independent Bernoulli random variable with probability $1/2$.

The chain is a Gibbs sampler; each iteration first samples $[k_1, k_2 \mid z]$ followed by the $[\alpha_l, \beta_l, \gamma_l, \delta_l \mid k_1, k_2, z]$ and then finally $[z \mid \alpha, \beta, \gamma, \delta, k_1, k_2]$. When the chain is run the original variables (s_i) can be recovered by taking $s_i = 2z_i - 1$ for each $i = 1, \ldots, D$.

To focus on a specific point, say $s_w = 1$; if s_w is surrounded by spins of opposite sign then with the Wolff algorithm it can only get flipped in sign if it is selected as the starting point of a cluster, which arises with probability $1/D$. On the other hand, with the new algorithm, the new value for s_w will be an independent Bernoulli variable, and so will flip sign with probability $1/2$.

Now suppose s_w has $\kappa > 0$ neighbors all equal to 1. Then the probability of s_w being flipped is given by $P(\text{flip}\, s_w) =$

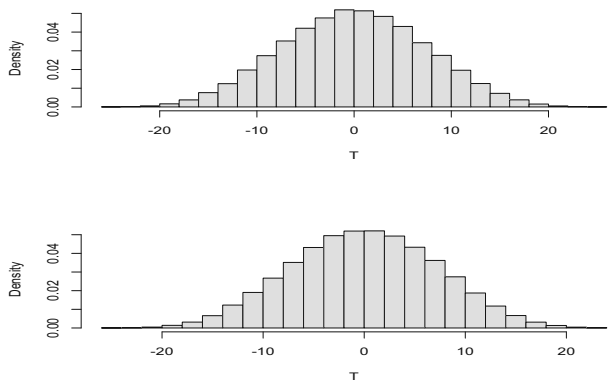

Fig. 12 Histogram samples of T from new algorithm (upper panel) and from Wolff algorithm (lower panel)

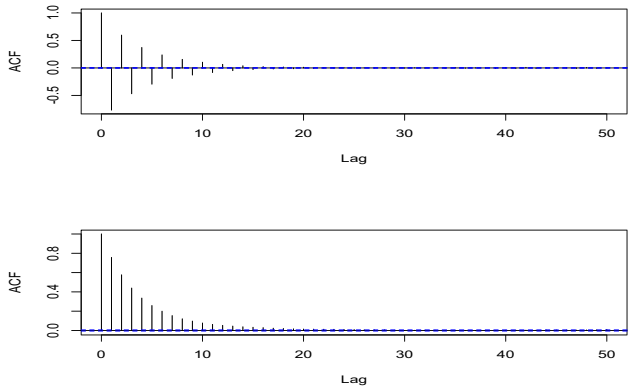

Fig. 13 ACF estimator for T samples from the new algorithm (upper panel) and from Wolff algorithm (lower panel)

$1-(1-\kappa/N)^{k_1}$. Now k_1 is Poisson with mean $2J\,N$ and so $P(\text{flip}\, s_w) = 1-\exp(-2J\kappa)$.

4.2 Illustrations

We first run the new algorithm in a small setting where the true density can be evaluated. So we take $d = 2$ and focus on the probabilities for the sum of the (z_i), writing $T = \sum_{i=1}^{D} z_i$. Then, taking $J = 0.2$, we get $P(T = 4) = P(T = -4) = 0.128$, $P(T = 2) = P(T = -2) = 0.231$ and $P(T = 0) = 0.282$. Over a run of 100,000 iterations we obtain the estimators $\widehat{P}(T = 0) = 0.283$, $\widehat{P}(T = 2) = 0.230$, $\widehat{P}(T = -2) = 0.230$, $\widehat{P}(T = 4) = 0.129$ and $\widehat{P}(T = -4) = 0.129$.

With a higher dimension, $d = 5$ so $D = 25$, while retaining $J = 0.2$, we compare with the Wolff algorithm. The outcome is that while the Wolff algorithm has positive autocorrelation on the T output, the new algorithm generates antithetic variables.

In Fig. 12 the output of the T samples represented by histograms from both the new algorithm and the Wolff algorithm are presented. As anticipated they are the same.

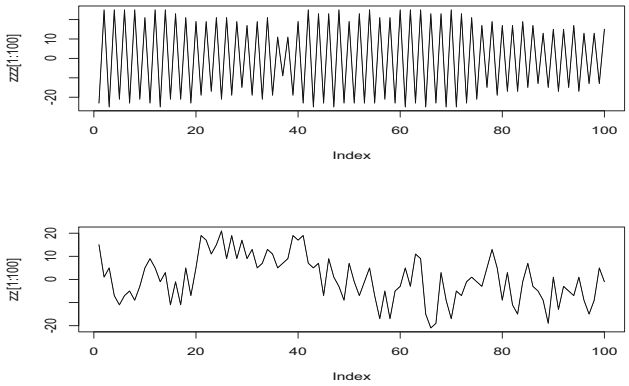

Fig. 14 Trace plot of first 100 T samples from the new algorithm (upper panel) and from Wolff algorithm (lower panel) with $d = 5$ and $J = 1$

In Fig. 13 we plot the corresponding ACF estimators. As can be see the new algorithm is generating antithetic variables, while the Wolff algorithm has the usual positive decay with higher lags. To explain this phenomenon, define C_0 to be the subset of A_0 to be flipped to 1, and C_1 the subset of A_1 to be flipped to 0, and C the remainder which have new values independently generated from the Bernoulli distribution. Then the current T and new T' are given by

$$T = |C_1| + \sum_{i \in C} z_i$$

and

$$T' = |C_0| + \sum_{i \in C} z_i' = D - |C_1| - |C| + \sum_{i \in C} z_i',$$

where the (z_i') are independent Bernoulli random variables.

The introduction of the latent variables k_1 and k_2 makes it very clear how the value of J influences the number of flips. If $J = 0$, an infinite temperature, then $k_1 = k_2 = 0$ with probability 1 and so all the (z_i) will be independent Bernoulli random variables. On the other hand, if J is very large (a low temperature), then k_1 and k_2 will be stochastically large. Hence, the sets A_0 and A_1 will be large and covering a large number, if not all, the clusters of common value (a set of indices $\{i\}$ connected as neighbors and with the (z_i) having the same value).

However, for large J, the two dominant probabilities arise when the (z_i) are either all 0 or 1. The new algorithm in this case would be switching between these two states. From this perspective, it is to be noted that high values of J which are not as extreme as the aforementioned case, would perform well. In Fig. 14 the first 100 T samples are plotted for the new algorithm (upper panel) and the Wolff algorithm. as can be seen the mixing is substantially more pronounced for the new algorithm.

To investigate further large values of J when we know the correct answer we return to $d = 2$. Taking $J = 1$ it is possible to evaluate $P(T = 4)/P(T = 0)$ which is given by $\exp(4)/(4 + 2\exp(-4)) = 13.53$. The new algorithm, using a sample of size 100, 000 gives a value of 13.47, while the corresponding estimator from the Wolff algorithm gives a value of 11.25.

5 Discussion

In this paper we have used latent uniform sampling algorithms to sample a joint distribution on binary values. Such distributions arise in classic contexts and are known to be problematic to sample when the dimension is large and/or the distribution is multimodal.

For the slice sampler, when the distribution is in fact simple, in the sense it is unimodal, the single flip proposal Metropolis chain works well and mixes faster than the latent slice algorithm. However, this hides a couple of important issues. One is that in practice it would not necessarily be known that the distribution is unimodal and so other modes would be left undetected. A further point is that generic algorithms which have the ability to jump between modes are needed and currently such suitable algorithms are lacking, although specialized algorithms exists for some distributions, such as the Ising model. These new algorithms would also be required to exhibit certain flexibility, which is that local moves can occur if the large jumps get rejected, as a lot of them will be. This is precisely a feature of the latent slice sampler; as the shrinkage proceeds from the initial jump proposal, and if these get rejected, so the proposals become more local to the current point.

A succinct way to describe the performance of the latent slice sampler is that it is robust. It performs well if the distribution is simple, such as being unimodal, yet has the ability to find different modes if they exist.

In Sect. 3.1 we demonstrated the accuracy of the latent slice sampler. The example in Sect. 3.2 is extreme but makes a point very clearly about the ability of the latent slice sampler to jump between modes and maintain a correct stationary distribution. This is certainly a challenging problem and it is not clear there are even any alternative algorithm capable of achieving this outcome. Section 3.3 considered the conditional logistic distribution for which we can sample exactly and hence compare the performance of various algorithms. The latent slice sampler is shown to easily outperform the Metropolis sampler. In Sect. 3.4 we look at a variable selection problem. In this case, when the problem is regular, also referred to as “easy", the Metropolis sampler has an advantage over the latent slice sampler. Though as we have previously mentioned, this can be deceptive. For it might not be known that multi-modes exist. On the other hand, when high co-linearity exists the latent slice sampler outperforms the Metropolis, and with sufficiently high co-linearity the Metropolis could be forced to come to a stop. Section 3.5 looked at Bayesian decision trees and set up a problem in which the standard algorithms failed to mix adequately, whereas the latent slice sampler did well and was able to mix across modes in the space of decision trees.

Section 4 looks at the Ising model and we compare our latent uniform sampling algorithm with the Wolff algorithm. The new sampler compares favorably with the Wolff algorithm. Specifically, we have shown how a latent Poisson version of the Ising model has the ability to generate antithetic variables from the output of a Markov chain. Developing the algorithm, it is also possible to apply the same idea to the more general

$$p(s) \propto \exp\left\{\sum_{i,j} H_{i,j}\, s_i\, s_j\right\}$$

where now the $H_{i,j} \geq 0$ is the only constraint. This can be written as

$$p(s) \propto \exp\left\{\sum_{i,j} H_{i,j}(1 + s_i)(1 + s_j)/2\right\} \times \exp\left\{\sum_{i,j} H_{i,j}(1 - s_i)(1 - s_j)/2\right\}.$$

We can proceed as previously and the alteration to the algorithm is that now we would sample (α_l, β_l) from the set $A_1 = \{(i, j) : z_i = z_j = 1\}$ with probability proportional to $H_{i,j}$. All other aspects of the algorithm, including the flipping procedure, remain the same.

Future work can go in a number of directions. One is to look at problems involving multivariate distributions with mixed types of variable; e.g. the most simple being a joint distribution on $\{0, 1\} \times \mathbb{R}$. Obviously more complicated cases can be considered including mixture models where in a Bayesian setting there will be a joint distribution on the component indicator variables as well as the component parameters.

Another direction would be to consider an adaptive algorithm and this would be naturally arising via a general version of (1) letting the “free" density for s to depend on x; i.e.

$$f(x, w, s, l) = \mathbf{1}(w < \pi(x))\, s^{-1} p(s \mid x)\, \mathbf{1}(x - s/2 < l < x + s/2).$$

The marginal density for x remains as $\pi(x)$. The aim would be to adapt $p(s \mid x)$ as the chain proceeds so to better propose regions of higher probability, such as separated modes.

Acknowledgements The authors would like to thank an Editor, Associate Edtor and two referees for valuable comments on the first submssion of the paper.

References

Basak, P., Linero, A.R., Sinha, D., Lipsitz, S.: Semiparametric analysis of clustered interval-censored survival data using soft Bayesian additive regression trees (sbart). arXiv preprint arXiv:2005.02509 (2020)

Besag, J., Green, P.J.: Spatial statistics and Bayesian computation. J. R. Stat. Soc. B **55**, 25–37 (1993)

Chen, Y., Dwivedi, R., Wainright, M.J., Yu, B.: Fast MCMC sampling algorithms on polytopes. J. Mach. Learn. Res. **19**, 1–86 (2018)

Chipman, H.A., George, E.I., McCulloch, R.E.: Bayesian CART model search. J. Am. Stat. Assoc. 93, 935–948 (1998)

Damien, P., Wakefield, J.C., Walker, S.G.: Gibbs sampling for Bayesian nonconjugate and hierarchical models using auxiliary variables. J. R. Stat. Soc. B **61**, 331–344 (1999)

Denison, D., Mallick, B., Smith, A.F.M.: A Bayesian CART algorithm. Biometrika **85**, 363–377 (1998)

Diaconis, P., Saloff-Coste, L.: What do we know about the Metropolis algorithm? J. Comput. Syst. Sci. **57**, 20–36 (1998)

Diaconis, P., Strook, D.: Geometric bounds for eigenvalues of Markov chains. Ann. Appl. Probab. **1**, 36–61 (1991)

Ekin, T., Walker, S.G., Damien, P.: Augmented simulation methods for discrete stochastic optimization with recourse. Ann. Oper. Res. (2022). https://doi.org/10.1007/s10479-020-03836-w

George, E.I., McCulloch, R.E.: Variable selection via Gibbs sampling. J. Am. Stat. Assoc. **88**, 881–889 (1993)

Hill, J.L. (2011) Bayesian nonparametric modeling for causal inference. Journal of Computational and Graphical Statistics, vol 20, pp. 217–240

Ising, E.: Beitrag zur theorie des ferromagnetismus. Zeitsschrift fur Physik **31**, 253–258 (1925)

Kuo, L., Mallick, B.: Variable selection for regression models. Sankhya **60**, 65–81 (1998)

Landau, D.P., Binder, K.: A Guide to Monte Carlo Simulations in Statistical Physics, 4th edn. Cambridge University Press, Cambridge (2015)

Li, Y., Linero, A.R., Murray, J.S.: Adaptive conditional distribution estimation with Bayesian decision tree ensembles. arXiv preprint arXiv:2005.02490 (2020)

Li, Y., Walker, S.G.: A latent slice sampling algorithm. Revised for Computational Statistics and Data Analysis (2022)

Linero, A.R.: A review of tree-based Bayesian methods. Commun. Stat. Appl. Methods **24**, 543–559 (2017)

Liu, J.: Peskun's theorem and a modified discrete-state Gibbs sampler. Biometrika **83**, 681–682 (1996)

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. **21**, 1087–1092 (1953)

Neal, R.M.: Slice sampling. Ann. Stat. **31**, 705–767 (2003)

Pompe, E., Holmes, C., Atuszyski, K.: A framework for adaptive MCMC targeting multimodal distributions. Ann. Stat. **48**, 2930–2952 (2020)

Schafer, C.: Monte Carlo methods for sampling high-dimensional binary vectors. Doctoral Thesis, University Paris–Dauphine (2012)

Schafer, C., Chopin, N.: Sequential Monte Carlo on large binary sampling spaces. Stat. Comput. **23**, 163–184 (2013)

Swendsen, R.H., Wang, J.S.: Nonuniversal critical dynamics in Monte Carlo simulations. Phys. Rev. Lett. **58**, 86–88 (1987)

Tak, H., Meng, X.L., van Dyke, D.A.: A repelling-attracting metropolis algorithm for multimodality. J Comput Graph. Stat. **27**, 479–490 (2018)

Tierney, L.: Markov chains for exploring posterior distributions. Ann. Stat. **22**, 1701–1728 (1994)

Walker, S.G.: Sampling un-normalized probabilities: an alternative to the Metropolis–Hastings algorithm. SIAM J. Sci. Comput. **36**, A482–A494 (2014)

Wolff, U.: Collective Monte Carlo updating for spin systems. Phys. Rev. Lett. **62**, 361–364 (1989)

Zellner, A.: Applications of Bayesian analysis in econometrics. Statistician **32**, 23–34 (1983)