Improved Perfect Slice Sampling *

Wolfgang Hörmann (hormannw@boun.edu.tr)
Department for Applied Statistics and Data Processing, University of Economics and Business Administration, Augasse 2–6, A-1090 Vienna, Austria, and Department of Industrial Engineering, Bogazici University Istanbul, 80810 Bebek-Istanbul, Turkey

Josef Leydold (leydold@statistik.wu-wien.ac.at)
Department for Applied Statistics and Data Processing, University of Economics and Business Administration, Augasse 2–6, A-1090 Vienna, Austria

December 10, 2003

Abstract. Perfect slice sampling is a method to turn Markov Chain Monte Carlo (MCMC) samplers into exact generators for independent random variates. The originally proposed method is rather slow and thus several improvements have been suggested. However, two of them are erroneous. In this article we give a short introduction to perfect slice sampling, point out incorrect methods, and give a new improved version of the original algorithm.

Keywords: Markov chain Monte Carlo method, perfect slice sampling, coupling from the past

1. Introduction

Markov Chain Monte Carlo (MCMC) samplers are very powerful methods for drawing random samples for quite arbitrary distributions. In particular they are used in the case of simulations that invoke high dimensional integrals. However, there are two serious drawbacks: they produce dependent random variables (vectors); and they require some burn-in phase for the convergence of the Markov chain to the stationary distribution. For its length only empirical rules-of-thumb exist. To overcome these disadvantages Propp and Wilson (1996) suggested so called *perfect sampling algorithms* that allow to turn Markov Chain Monte Carlo (MCMC) samplers into exact algorithms without using approximate convergence arguments. Although first developed for discrete state spaces perfect sampling also can be applied to Markov chains with state space $\mathbb{R}^d$ albeit this is not easy, see (Green and Murdoch, 2000; Wilson, 2000; Murdoch, 2000; Murdoch and Meng, 2001).

Coupling from the past (CFTP) suggested by Propp and Wilson (1996) is probably the most popular of these perfect sampling algo-

* This work was supported by the Austrian Science Foundation (FWF), project no. P16767-N12.

rithms. The main building block of all CFTP algorithms is the *randomizing operation*. It is a deterministic function ϕ taking as input the state X_t of the chain $\mathbf{X}$ at time t and some intrinsic randomness U_t. The randomizing operation returns the new output state

$$X_{t+1} = \phi(X_t, U_t)\,.$$

It can be seen as a kind of modified transition kernel. Roughly speaking CFTP starts Markov chains from all possible points of the state space at some time $-T$ in the past. Using the randomizing operation with the same intrinsic randomness these chains develop in parallel or coalesce. If at time 0 all have coalesced in a single state this state is returned. Otherwise the chains are restarted at some earlier time $-T' < -T$; see e.g. Wilson (2000), for a short tutorial.

In general it is not easy to give an explicit randomizing operation for a Markov chain sampler. However, Mira et al. (2001) pointed out that the slice sampler is well suited for a combination with the CFTP algorithm as it is stochastically monotone with respect to the natural partial ordering induced by the density f of the given distribution. Hence only two chains (for the minimal and maximal starting point, resp.) have to be run in parallel to keep track of coalescence that is necessary for perfect sampling. A second useful property of the slice sampler is the fact that it seems to be not difficult to find an implementation with a natural possibility of coalescence.

The idea of perfect slice samplers was applied to different practical problems by Casella et al. (2002) and Philippe and Robert (2003). As the original version of the perfect slice samplers by Mira et al. (2001) can require substantial computations in every iteration these authors described a simpler and faster variant of the original algorithm. However, their improvements are not correct.

In this article we propose a new variant that is simple and correct. This paper is outlined as follows: Section 2 introduces the concept of the perfect slice sampler. In Sect. 3 we shortly point out the error in the improvement to the original algorithm. Section 4 collects the main ideas of our proposed new algorithm. Details (Sect. 5) and computational experiments (Sect. 6) are given.

2. The perfect slice sampler

We consider a density $f(x)$ with $x \in \mathbb{R}^d$. To sample from f the slice sampler (see Mira et al., 2001, and the references given there) introduces an auxiliary variable ε, which is uniformly distributed over $(0, f(x))$.

The Markov chain of (x, ε), $x \in \mathbb{R}^d$ and $\varepsilon \in \mathbb{R}$ is then generated by the transition kernel

$$(S1) \quad \varepsilon \sim U(0, f(X)) \qquad \text{and} \qquad (S2) \quad X \sim U(A_f(\varepsilon)),$$

where $U(C)$ denotes the uniform distribution over the set C and $A_g(y) = \{x|g(x) \geq y\}$. Thus the slice sampler can be described as a Gibbs sampler for the uniform distribution over the region below the density. Mira et al. (2001) pointed out that the slice sampler is well suited for a combination with the CFTP algorithm as it is stochastically monotone with respect to the natural partial ordering induced by f.

To be precise we define the ordering $\preceq$ such that

$$f(x) \leq f(y) \quad \text{implies} \quad x \preceq y.$$

Notice that such an ordering exists as for x, y with $f(x) \neq f(y)$ we find $x \preceq y$ if and only if $f(x) < f(y)$. For the perfect slice sampler an appropriate randomizing operation (called *stochastic recursive sequence* by Mira et al. (2001)) is required that guarantees that the resulting Markov chain is stochastically monotone. However, it is not easy to give an explicit randomizing operation for a general slice sampler as the slice sample includes uniform sampling over the set $A_f(\varepsilon)$ which depends on X_t and changes at every recursion step. Recall that a Markov chain $\mathbf{X}$ on a partially ordered space is *stochastically monotone*, if for all $z \in \mathbf{X}$, we have $\mathrm{P}(X_1 \preceq z|X_0 = x_1) \geq \mathrm{P}(X_1 \preceq z|X_0 = x_2)$ whenever $x_1 \preceq x_2$ (Daley, 1967). Thus a proper randomizing operation $\phi(x, U)$ must satisfy $\phi(x, U) \preceq \phi(y, U)$ for all U whenever $x \preceq y$.

To facilitate the notation and to concentrate at the main point we restrict our attention for the moment to the simple case of a bounded monotone decreasing density $f(x)$ on $[0, 1]$. Then we can easily formulate the probably simplest possible randomizing operation

$$\phi_0(x, V, U) = U\, f^{-1}(V\, f(x)),$$

where both U and V are $U(0, 1)$ uniform variates. ϕ_0 is simple and it is not difficult to see that it is stochastically monotone with respect to our ordering as for the same values of U and V, $x \preceq y$ clearly implies $\phi_0(x, V, U) \preceq \phi_0(y, V, U)$, since for decreasing f, $x \preceq y$ implies $x \geq y$. But it is also clear that for this randomizing operation $f(x) \neq f(y)$ implies $\phi_0(x, V, U) \neq \phi_0(y, V, U)$. Thus two distinct states can never coalesce and ϕ_0 is not a useful randomizing operation for perfect sampling.

Mira et al. (2001) have solved this problem by replacing U by a sequence $\mathbf{W} = (W_i)$ of uniform random variates. For the case of a monotone decreasing density f on $[0,1]$ it can be defined by the recursion

$$W_1 \sim U(0,1) \qquad \text{and} \qquad W_i \sim U(0, W_{i-1})\,.$$

Then they defined $I(y) = \inf\{j : f(W_j) > y\}$ and the randomizing operation as

$$\phi_1(x, V, \mathbf{W}) = W_{I(V f(x))}\,.$$

It is not difficult to show that $I(Vf(x))$ is almost surely finite; thus we only have to generate a finite subsequence of $\mathbf{W}$. It is also obvious that for a fixed V and a fixed sequence $\mathbf{W}$ the randomizing operation ϕ_1 is stochastically monotone. It is also quite clear that ϕ_1 maps different values x and y into the same value whenever $I(Vf(x)) = I(Vf(y))$. So there is some chance for coalescence. Another practical advantage of ϕ_1 is certainly that we do not need the inverse of the density. To demonstrate how randomizing operations are used in CFTP algorithms we give all necessary details as Algorithm 1. Notice that it is necessary to reuse the random input V_t and $\mathbf{W}_t$ when the first trial with a given T does not coalesce. Also note that for computing $\phi_1(X_t^{(0)}, V_t, \mathbf{W}_t)$ we never need more elements of $\mathbf{W}_t$ than in previous trials as we start from farer in the past and the slice sampler is monotone.

Algorithm 1 Perfect slice sampler Mira-Møller-Roberts

Set $T \leftarrow 1$.
loop
  Set $X_{-T}^{(1)} \leftarrow 0$ and $X_{-T}^{(0)} \leftarrow 1$. /* Start at first and last point in order */
  for $t = -T$ up to -1 **do**
    if $t < -T/2$ **then**
      Generate and store $V_t \sim U(0,1)$ and sufficiently long part of $\mathbf{W}_t$.
    else
      Use stored values for V_t and $\mathbf{W}_t$.
    Set $X_{t+1}^{(0)} \leftarrow \phi_1(X_t^{(0)}, V_t, \mathbf{W}_t)$. /* Step in chain of minimal element */
    Set $X_{t+1}^{(1)} \leftarrow \phi_1(X_t^{(1)}, V_t, \mathbf{W}_t)$. /* Step in chain of maximal element */
  if $X_0^{(1)} = X_0^{(0)}$ **then** /* Coalescence */
    return $X_0^{(0)}$.
  else
    Set $T \leftarrow 2T$.

Algorithm 1 has an obvious disadvantage: We have to generate and store the (possibly long) sequence of $\mathbf{W}_t$ for every time t necessary in our simulation. It is possible to compute the expected length of that sequence.

THEOREM 1. *For a monotone decreasing density f on $[0,1]$ and $X_t^{(1)} = 0$ the expected length of the generated W-(sub-)sequence* $\mathrm{E}\,(\#W_i)$ *is*

$$\mathrm{E}\,(I(V\,f(0))) = \int_0^1 (1-\log(x))\frac{-f'(x)}{f(0)}\,dx\,.$$

Proof. If we denote $p_0 = f^{-1}(f(0)\,V)$ it is clear that $\mathrm{P}\,(\#W_i = 1) = p_0$. To calculate the probability that $\#W_i = 2$ we use that $W_1 \sim U(0,1)$ and get

$$\begin{aligned}\mathrm{P}\,(\#W_i = 2) &\int_{p_0}^1 \mathrm{P}\,(\#W_i = 2|W_1 = w_1)\;dw_1\\ &= \int_{p_0}^1 \frac{p_0}{w_1}\,dw_1 = -p_0\log(p_0)\,.\end{aligned}$$

For the case that $\#W_i = 3$ we use that $W_2 \sim U(0,W_1)$ and get:

$$\mathrm{P}\,(\#W_i = 3) = \int_{p_0}^1\int_{p_0}^{w_1} \frac{p_0}{w_2}\frac{1}{w_1}\,dw_2\,dw_1 = \frac{1}{2}\,p_0\,(\log(p_0))^2\;,$$

in a similar way we can find

$$\mathrm{P}\,(\#W_i = 4) = -\frac{1}{6}\,p_0\,(\log(p_0))^3\;,$$

and by induction

$$\mathrm{P}\,(\#W_i = i-1) = \frac{1}{i!}\,p_0\,(-\log(p_0))^i\;.$$

Thus we have proven that for fixed p_0, $\#W_i - 1$ follows a Poisson distribution with mean $-\log p_0$ and $\#W_i$ has expectation $1-\log p_0$. To finish the proof it is only necessary to find the distribution of $P_0 = f^{-1}(f(0)\,V)$. We have

$$\mathrm{P}\,(P_0 \le x) = 1 - \frac{f(x)}{f(0)}\,,$$

and thus P_0 has the density $-f'(x)/f(0)$. With

$$\mathrm{E}\,(\#W_i) = \int_0^1 \mathrm{E}\,(\#W_i|P_0 = x)\,\frac{-f'(x)}{f(0)}\,dx = \int_0^1 (1-\log(x))\frac{-f'(x)}{f(0)}\,dx$$

our proof is finished.

Theorem 1 clearly indicates that for densities with a spike at 0 the W-sequence is on average longer than for distributions with linear or concave densities. For example, for the density $f(x) = 2-2x$ on $[0,1]$ we obtain $\mathrm{E}\,(\#W_i) = 2$ which is the same as the expected number of iterations for the naive rejection algorithm from a constant hat. However, $\mathrm{E}(\#W_i)$ is just the average length of the W-sequence we have to generate in the first step of our recursion of the CFTP algorithm. The situation gets even worse when we consider the family of densities $f(x) = \max(2k(1-k\,x), 0)$ on $[0,1]$ for $k > 0$. We then find $\mathrm{E}(\#W_i) = 2 + \log(k)$. This is less than the expected number of iterations for naive rejection which is $2\,k$. But the very simple rejection Algorithm of Devroye (1986), Sect. VII.3, for monotone densities on bounded domain (see also Chap. 6.1 in Hörmann et al., 2004) has an expected number of iterations equal to $1+\log(2)+\log(k)$ which is a bit smaller than $\mathrm{E}(\#W_i)$. If we remember that we have to repeat the steps of our recursion of the CFTP algorithm several times and that even the expected length of the W-sequence of the first step is larger than the expected number of recursions in simple rejection algorithms this is a clear indication that this realization of the perfect slice sampler is not at all competitive with standard rejection algorithms.

One could argue that we have only discussed monotone distributions. But if we try arbitrary bounded densities on $[0,1]$ we have to reformulate the randomizing operation. If we have no special trick available to sample from a uniform distribution over the set $A_f(\varepsilon)$ we have to use naive rejection to generate the W-sequence. In the first step of the algorithm we have to sample $U(0,1)$ variates till $V f(0) < f(U)$. But this experiment is the same as rejection from a constant hat and therefore takes the same expected number of trials. Thus this version of the perfect slice sampler is not competitive for the general case as well.

3. An incorrect improvement

In the light of the above theorem it would be of course much easier and faster to use randomizing operation ϕ_0 instead of ϕ_1 as this would take from us the load of generating the W-sequence. But we need some trick to obtain a chance of coalescence. This is what Casella et al. (2002) and Philippe and Robert (2003) have tried. Changed to our simple setting of a monotone decreasing density on $[0,1]$ we formulate their methods as Algorithm 2.

We can see that ϕ_1 was replaced by ϕ_0. Then it is enough to generate a single variate U instead of the sequence $\mathbf{W}$. To add the possibility of coalescence they set $X^{(1)}_{t+1} \leftarrow X^{(0)}_{t+1}$ for the case that $f(X^{(0)}_{t+1})$ is large enough. Clearly Algorithm 2 needs no W-sequence. It is also obvious that coalescence is possible and even the sequences $\mathbf{X}^{(0)}$ and $\mathbf{X}^{(1)}$ are realizations of the slice sampler. However, when we implemented this

Algorithm 2 Incorrect perfect slice sampler

Set $T \leftarrow 1$.
loop
  Set $X^{(1)}_{-T} \leftarrow 0$ and $X^{(0)}_{-T} \leftarrow 1$.
  for $t = -T$ up to -1 **do**
    if $t < -T/2$ **then**
      Generate and store $V_t \sim U(0,1)$ and $U_t \sim U(0,1)$.
    else
      Use stored values for V_t and U_t.
    Set $X^{(0)}_{t+1} \leftarrow \phi_0(X^{(0)}_t, V_t, U_t) = U_t\, f^{-1}(V_t\, f(X^{(0)}_t))$.
    if $f(X^{(0)}_{t+1}) \geq V_t\, f(X^{(1)}_t)$ **then**
      Set $X^{(1)}_{t+1} \leftarrow X^{(0)}_{t+1}$.
    else
      Set $\tilde{U}_t \leftarrow U_t$. /* Philippe and Robert (2003) */
      /* Or */
      Generate and store $\tilde{U}_t \sim U(0,1)$, (resp., use stored value).
      /* Casella et al. (2002), Fig. 3 */
      Set $X^{(1)}_{t+1} \leftarrow \phi_0(X^{(1)}_t, V_t, \tilde{U}_t) = \tilde{U}_t\, f^{-1}(V_t\, f(X^{(1)}_t))$.
  if $X^{(1)}_0 = X^{(0)}_0$ **then** /* Coalescence */
    return $X^{(0)}_0$.
  else
    Set $T \leftarrow 2T$.

algorithm and tested it using a χ^2 goodness-of-fit test and large sample sizes the p-value was always smaller than 0.001. After this observation we checked the reason for this fact and found the following simple explanation:

The sequence $\mathbf{X}^{(1)}$ depends on the values of $\mathbf{X}^{(0)}$ and is therefore not following the same randomizing operation as $\mathbf{X}^{(0)}$ itself. To be precise it is not a randomizing operation at all as it takes as input also the state of $\mathbf{X}^{(0)}$. This observation alone is no proof that the algorithm is wrong. But we can see for very simple examples that depending whether we start with $T = -1$ or $T = -2$ Algorithm 2 produces different variates which is against the fundamental idea of CFTP that coalescence can only occur if the output is fully determined by the randomness produced so far. Starting farer in the past can never change this output if the CFTP algorithm is correct. The following simple numerical example shows that Algorithm 2 is not correct.

Example. We consider the density $f(x) = 2 - 2x$ on (0,1). For $T = -1$ we generate and store $V_{-1} = 0.5$ and $U_{-1} = 0.1$. We obtain

$X_0^{(0)} \leftarrow \phi_0(X_{-1}^{(0)}, 0.5, 0.1) = 0.1 f^{-1}(0.5\, f(1)) = 0.1$. As $f(0.1) = 1.8 > V f(X_{-1}^{(1)}) = 1$, $X_0^{(1)}$ is set to 0.1 as well. Coalescence is reached at time $t = 0$ and we are finished.

For $T = -2$ we generate and store $V_{-2} = 0.3$ and $U_{-2} = 0.2$. Similar to above we obtain $X_{-1}^{(0)} = X_{-1}^{(1)} = 0.2$. For the last step with $t = -1$ we have to use $V_{-1} = 0.5$ and $U_{-1} = 0.1$ as above. We then get $X_0^{(0)} = X_0^{(1)} = 0.06$. So starting at $T = -2$ leads to a different result than starting with $T = -1$. This fact is not at all influenced by our choice of the variant in Step 13 of Algorithm 2.

4. Multiscale coupling and the perfect slice sampler

Of course there remains the question whether it is possible to find a correct version of perfect slice sampling without generating a W-sequence. Wilson (2000) introduces *layered multishift coupling* which is a randomizing operation that allows for coalescence when shifted versions $s + X$ for different values of s are generated for the Markov chain. However, we cannot apply this method directly. In particular the application to the second step (S2) of a general slice sampler would be difficult as it may be multi-dimensional and depends on the density. For the first step (S1) of the slice sampler, however, we have to generate a uniform random variate between 0 and $f(x)$; this means that for different chains we have to generate uniform distributions with different scale parameters. Thus we need *layered multiscale coupling*. This can be done when we generate shifted versions of standard exponential variates $s + E$ with different shifts $s = -\log(f(x))$. Then we have $\exp(-s - E) \sim U(0, f(x))$ as desired and we have gained the chance of coalescence.

Before we describe the details of the new algorithm we have to explain the layered multishift coupler and introduce the idea of multiscale coupling. We follow Wilson (2000) but restrict ourselves to the standard exponential distribution.

4.1. Layered multishift coupling

To generate a uniform random variate $U(s, r+s)$, $r > 0$, with arbitrary shift parameter s in a way that allows for coalescence in the case of fixed r but different values for s we can use the following randomizing operation

$$\phi_u(s, r, U) = r \cdot \left(\left\lfloor \frac{s}{r} + 1 - U \right\rfloor + U \right) ,$$

where U denotes the $U(0,1)$ distributed random input. To see that we really obtain the desired distribution we consider the case that $\frac{s}{r}+1-U$ is an integer (clearly this can be the case for any fixed choice of s and r as $U \sim U(0,1)$). In this case we obtain the largest possible value for $\phi_u(s,r,U)$ which is equal to $s+r$. If $\frac{s}{r}+1-U$ is not an integer its fractional part is subtracted by the floor operation. It is easy to see that the fractional part of $\frac{s}{r}+1-U$ is a $U(0,1)$ random variate. Thus a $U(0,r)$ random variate is subtracted from the maximal possible value $s+r$ which shows that $\phi_u(s,r,U)$ is really a $U(s,r+s)$ random variate.

Coalescence occurs for different shift parameters s_1 and s_2 if the floor operation leads to the same integer for both values of s. For the probability of coalescence we can easily find:

$$\mathrm{P}\left(\phi_u(s_1,r,U) = \phi_u(s_2,r,U)\right) = \mathrm{P}\left(\left\lfloor \frac{s_1}{r}+1-U \right\rfloor = \left\lfloor \frac{s_2}{r}+1-U \right\rfloor\right) =$$

$$= \mathrm{P}\left(\left\lfloor \frac{|s_1-s_2|}{r}+1-U \right\rfloor = 0\right) = \max\left(1-\frac{|s_1-s_2|}{r}, 0\right) .$$

Note that this probability is the highest possible probability of coalescence as this is exactly the relative area of the intersection of the intervals $[s_1, r+s_1]$ and $[s_2, r+s_2]$. Clearly these intervals do not overlap when $|s_1-s_2| > r$.

4.2. Multiscale coupling

As we have explained above we need a randomizing operation for shifted versions of the exponential distribution. In order to benefit from randomizing operations similar to ϕ_u above we cannot generate exponential random variates with the inversion method. Instead we generate a random point (X,Y) uniformly distributed on the area between density and x-axis. Then we consider the "layer" with height Y, i.e. all points between $(0,Y)$ and (R,Y) with $R = -\log(Y)$. It is not difficult to see that if X, Y, and R are generated as described above the random variate $U \sim U(0,R)$ is exponentially distributed. Clearly we do not need X and the height Y of the layer explicitly, we just need R and it is not difficult to show that R selected in the way described above follows a Gamma(2) distribution with density $x\,exp(-x)$. So we can easily generate R directly either taking the logarithm of the product of two uniform variates or, if we do not want to waste uniform variates, we could use one of the very fast automatic methods for log-concave densities (see Hörmann et al., 2004). We can now define the new randomizing operation for shifted versions of the standard exponential distribution:

$$\phi_e(s, R, U) = R\left(\left\lfloor \frac{s}{R} + 1 - U \right\rfloor + U\right),$$

where R denotes a Gamma(2) random variate and U an independent $U(0,1)$ variate.

For fixed $R = R_0$ we can use the result above to get the conditional probability

$$\begin{aligned}\mathrm{P}\left(\text{coalesence}|R = R_0\right) &= \mathrm{P}\left(\phi_e(s_1, R_0, U) = \phi_e(s_2, R_0, U)\right)\\ &= 1 - |s_1 - s_2|/R_0\,.\end{aligned}$$

The unconditional probability of coalescence for random R is thus

$$\mathrm{E}\left(\mathrm{P}\left(\text{coalesence}\right)\right) = \int_0^\infty \max\left(0, 1 - \frac{|s_1 - s_2|}{R}\right) R\, e^{-R}\, dR = e^{-|s_1 - s_2|}\,.$$

Note that this is again the maximal possible probability of coalescence.

5. The new perfect slice-sampler

To obtain the new version of the perfect slice sampler it is now enough to collect the ideas of the last sections. First we define the randomizing operation of the multiscale coupler for the uniform distributions on $(0, b)$:

$$\phi_{mu}(b, R, U) = \exp\left(-R\left(\left\lfloor \frac{-\log(b)}{R} + 1 - U \right\rfloor + U\right)\right),$$

where again R denotes a Gamma(2) random variate and U an independent $U(0,1)$ uniform variate. For two different value of b, b_1 and b_2, the probability of coalescence follows directly from the result for the multishift coupler and is $\exp(-|\log(b_1) - \log(b_2)|) = \min(b_1/b_2, b_2/b_1)$, again the maximal possible value. For the case of monotone decreasing densities on $\mathbb{R}$ we could use the multiscale coupler for both the first (vertical) and the second (horizontal) step of the slice sampler but our empirical experiments indicated that using it only for the vertical step results in practically the same performance. Therefore we are presenting the details of a perfect slice sampler using multiscale coupling in the first step as Algorithm 3.

The only thing left is that we have to consider the monotonicity of the new randomizing operation. It is obvious that ϕ_{mu} is monotone

Algorithm 3 New perfect slice sampler, monotone densities on $[0, b_r]$

Set $T \leftarrow 1$.
loop
  Set $X_{-T}^{(1)} \leftarrow 0$ and $X_{-T}^{(0)} \leftarrow b_r$. /* Start at first and last point in order */
  for $t = -T$ up to -1 **do**
    if $t < -T/2$ **then**
      Generate and store random variates $R_t \sim \mathrm{Gamma}(2)$ and $U_t, V_t \sim U(0,1)$.
    Set $Y_0 \leftarrow \phi_{mu}(f(X_t^{(0)}), R_t, U_t)$.
    Set $Y_1 \leftarrow \phi_{mu}(f(X_t^{(1)}), R_t, U_t)$.
    Set $X_{t+1}^{(0)} \leftarrow V_t f^{-1}(Y_0)$. /* X_0 uniformly distributed over $A_f(Y_0)$ */
    Set $X_{t+1}^{(1)} \leftarrow V_t f^{-1}(Y_1)$. /* X_1 uniformly distributed over $A_f(Y_1)$ */
  if $X_0^{(1)} = X_0^{(0)}$ **then** /* Coalescence */
    return $X_0^{(0)}$.
  else
    Set $T \leftarrow 2T$.

in its first variable. Thus monotonicity is guaranteed for monotone densities on $\mathbb{R}$ as the second step of the randomizing operation

$$\phi_{(2)}(V, U) = U\, f^{-1}(V)\,,$$

is clearly monotone.

For other densities and the multivariate case it is necessary to define the second step of the randomizing operation $\phi_{(2)}(V, U)$ in a monotone way. Thus for fixed V, $\phi_{(2)}(V, U) \sim U(A_f(V))$ is not enough. We also need that for a fixed randomness (vector or sequence) U, $V < \tilde{V}$ implies $\phi_{(2)}(V, U) \preceq \phi_{(2)}(\tilde{V}, U)$. This can be reached by naive rejection from the uniform distribution over the total domain but then the algorithm will have a very poor performance for most distributions. So the success of any version of the perfect slice sampling algorithm mainly depends on a good way to sample from the uniform distribution over the set $A_f(V)$ for all possible values of V.

6. Computational experience

We have coded Algorithm 3 and tested it for the standard exponential and for the Cauchy distribution both restricted to the intervals $[0, 1]$, $[0, 10]$, $[0, 100]$, and $[0, 1000]$. Table I reports our numerical results for the expected necessary length of the chain and compares it with the

Table I. Performance of Algorithm 3 compared to rejection

	domain	new perfect slice sampler	naive rejection	rejection with known $\int f$
exponential	[0,1]	1.94	1.58	1.46
exponential	[0,10]	5.76	10.00	3.30
exponential	[0,100]	9.29	100	5.61
exponential	[0,1000]	12.81	1000	7.91
Cauchy	[0,1]	1.64	1.27	1.24
Cauchy	[0,10]	5.54	6.80	2.91
Cauchy	[0,100]	11.72	64.07	5.16
Cauchy	[0,1000]	18.34	637.03	7.56

expected number of repetitions for the naive rejection algorithm and for the rejection algorithm using the knowledge of the area below the density (see Hörmann et al., 2004, p. 128, Algorithm 6.2).

The results of Table I indicate that for the case that the area below the density is unknown and the naive (constant) hat has a very bad fit the new algorithm is superior to all other known naive algorithms.

7. Conclusions

We have developed a new perfect slice sampling algorithm that considerably improves an existing algorithm and corrects a wrong algorithm given in the literature. The new algorithm outperforms simple automatic rejection algorithms for the case of long-tailed distributions with unknown area below the (non-normalized) density.

References

Casella, G., K. L. Mengersen, C. P. Robert, and D. M. Titterington: 2002, 'Perfect slice samplers for mixture distributions'. *Journal of the Royal Statistical Society B* **64**, 777–790.

Daley, D. J.: 1967, 'Stochastically monotone Markov chains'. *Z. Wahrsch. Ver. Geb.* **10**, 305–317.

Devroye, L.: 1986, *Non-Uniform Random Variate Generation*. New-York: Springer-Verlag.

Green, P. J. and D. J. Murdoch: 2000, 'Exact sampling for Bayesian inference: towards general purpose algorithms (with discussion)'. In: J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith (eds.): *Monte Carlo Methods*, Vol. 6 of *Bayesian Statistics*. Oxford, pp. 301–321, Oxford University Press.

Hörmann, W., J. Leydold, and G. Derflinger: 2004, *Automatic Nonuniform Random Variate Generation*. Berlin Heidelberg: Springer-Verlag.

Mira, A., J. Møller, and G. O. Roberts: 2001, 'Perfect slice samplers'. *Journal of the Royal Statistical Society B* **63**, 593 – 606.

Murdoch, D. J.: 2000, 'Exact sampling for Bayesian inference: unbounded state space'. In: N. Madras (ed.): *Monte Carlo Methods*, Vol. 26 of *Fields Institute Communications*. pp. 111–121, Americal Mathematical Society.

Murdoch, D. J. and X.-L. Meng: 2001, 'Towards perfect sampling for Bayesian mixture priors'. In: *ISBA 2000, Proceedings*. p. ??, ISBA and Eurostat.

Philippe, A. and C. P. Robert: 2003, 'Perfect simulation of positive Gaussian distributions'. *Statistics and Computing* **13**, 179–186.

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

Wilson, D. B.: 2000, 'Layered multishift coupling for use in perfect sampling algorithms (with a primer on CFTP)'. In: N. Madras (ed.): *Monte Carlo Methods*, Vol. 26 of *Fields Institute Communications*. pp. 141–176, American Mathematical Society.