An extension of Fill's exact sampling algorithm to non-monotone chains*

by

Duncan J. Murdoch** and Jeffrey S. Rosenthal***

University of Western Ontario and University of Toronto

(June 25, 1998; last revised August 4, 1998.)

Abstract. We provide an extension of Fill's (1998) exact sampler algorithm. Our algorithm is similar to Fill's, however it makes no assumptions regarding stochastic monotonicity, discreteness of the state space, the existence of densities, etc. We illustrate our algorithm on a simple example.

1. Introduction.

Markov chain Monte Carlo (MCMC) methods have become extremely popular in Bayesian inference problems as a way of approximately sampling from a complicated unknown probability distribution $\pi(\cdot)$ (cf. Gelfand and Smith, 1990; Smith and Roberts, 1993; Tierney, 1994; Gilks et al., 1996). MCMC algorithms produce a Markov chain $P(x, \cdot)$ which has $\pi(\cdot)$ as its stationary distribution; if the chain is run long enough, then under reasonably weak conditions (cf. Tierney, 1994) it will converge in distribution to $\pi(\cdot)$, facilitating approximate sampling.

One difficulty with these methods is that it is difficult to assess their convergence to stationarity. This necessitates the use of difficult theoretical analysis (Meyn and Tweedie, 1994; Rosenthal, 1995) or problematic convergence diagnostics (Cowles and Carlin, 1995; Brooks and Roberts, 1997) to draw reliable samples and do proper inference.

* Supported in part by NSERC of Canada.

** Department of Statistical and Actuarial Sciences, University of Western Ontario, London, Ontario, Canada N6G 2E9. Internet: `dmurdoch@pair.com`.

*** Department of Statistics, University of Toronto, Toronto, Ontario, Canada M5S 3G3. Internet: `jeff@utstat.toronto.edu`.

An interesting alternative algorithm, called *coupling from the past* (CFTP), was introduced by Propp and Wilson (1996, 1998) and has been studied and used by a number of authors (Kendall, 1997; Møller, 1997; Murdoch and Green, 1997; Foss and Tweedie, 1997; Kendall and Thönnes, 1998; Corcoran and Tweedie, 1998; Green and Murdoch, 1998; Murdoch and Rosenthal, 1998). By searching backwards in time until paths from all starting points have coalesced, this algorithm uses the Markov chain $P(x, \cdot)$ to sample *exactly* from $\pi(\cdot)$.

Another method of perfect simulation using monotonically coupled and uniformly ergodic chains was proposed by Fill (1998). Fill's algorithm is a form of rejection sampling. This algorithm was later extended by Møller and Schladitz (1998) and Thönnes (1997) to non-finite chains, motivated by applications to spatial point processes. Fill's algorithm has the advantage over CFTP of removing the correlation between the length of the run and the returned value, which is sometimes very desirable. However, it has only been used for stochastically monotone chains, making heavy use of the ordering of state space elements.

In this short paper, we extend Fill's algorithm to general, non-monotone chains. Like Fill, we justify our algorithm's validity through a simple conditional probability calculation.

Since completing this paper, it was brought to our attention that in unpublished work Fill (1995) had also extended his algorithm to non-monotone chains, for the case of finite state spaces. He alludes to this work in Section 11.2 of Fill (1998), under the heading 'Generalizability.' We regret our oversight, and look forward to Fill publishing his work on the subject.

2. The algorithm.

We assume that we are given a probability distribution $\pi(\cdot)$ on a state space $\mathcal{X}$. We also assume that we are given a Markov chain $P(x, \cdot)$ on $\mathcal{X}$, which we can simulate, and which has $\pi(\cdot)$ as a stationary distribution.

For notation, we let $N_{x,t} \sim P(x, \cdot)$ be a random variable which represents the "next" value of the chain (at time $t + 1$) in our particular method of simulation, given that the chain was at the point x at time t. We require that $N_{x,t}$ and $N_{y,s}$ be independent for $s \neq t$; however, the joint distribution of $N_{x,t}$ and $N_{y,t}$, for two different points x and y at the same time t, may be specified as convenient. Then, for $s \geq t$, we let $N_{x,t}^{s}$ be the value the chain will obtain at time s, given that it starts at x at time t and then moves consistently according to the $N_{x,i}$ values. Thus, $N_{x,t}^{t} = x$, $N_{x,t}^{t+1} = N_{x,t}$, and in general

$N_{x,t}^s = N_{N_{x,t}^{s-1},s-1}$ for $s \geq t+1$.

We further assume that we may simulate the time reversal of the chain, using Markov transition probabilities $\widetilde{P}(x,\cdot)$, such that

$$\pi(dx)\widetilde{P}(x,dy) = \pi(dy)P(y,dx)\,. \tag{1}$$

(In the common situation of a reversible chain, we'll have $\widetilde{P}(x,\cdot) = P(x,\cdot)$, so no further information is required.) We shall write $P^t(x,dy)$ and $\widetilde{P}^t(x,dy)$, respectively, for the forward and backward t-step transition probabilities.

Finally, we assume that we may sample from the conditional distribution of $N_{x,t}$ given a particular path, i.e. given that $N_{X_0,0} = X_1, N_{X_1,1} = X_2, \ldots, N_{X_{t-1},t-1} = X_t$, we know how to sample $N_{x,s}$ for any $x \in \mathcal{X}$ and $s = 0,\ldots,t-1$. This may sometimes be a difficult task in practice. However, typically auxiliary variables U_t are introduced so that $N_{x,t} = \phi(x,U_t)$; and it is often the case that once we know $N_{x,t}$ for some x, then we know U_t, so that we can then easily compute $N_{y,t}$ for other points y. Alternatively, if we have decided to make $N_{x,t}$ independent of $N_{y,t}$ for all $x \neq y$, then this step is completely straightforward.

Our algorithm proceeds as follows. We choose an arbitrary positive integer T, and an arbitrary state $Z \in \mathcal{X}$. We set $X_T = Z$. Using $\widetilde{P}(x,\cdot)$, we simulate fresh values $X_{T-1},\ldots,X_0$ from the time reversed chain; and we correspondingly set $N_{X_s,s} = X_{s+1}$ for $s = 0,1,\ldots,T-1$. We then reverse direction, and simulate $N_{x,s}$, $s = 0,\ldots,T-1$ from the conditional distribution described above, for all the different states $x \in \mathcal{X}$. Finally, we check whether $N_{x,0}^T$ is independent of x, i.e. is the same for any $x \in \mathcal{X}$. If it is, then the algorithm succeeds and outputs X_0 as its sample. If not, then the algorithm fails, and we start again with an independent simulation (perhaps with a fresh choice of T and Z), and keep repeating until we succeed.

(The careful reader will note that besides our other changes in notation, we have reversed the direction of time compared to Fill. Furthermore, Fill's original algorithm also incorporated a search for a good value of T, e.g. by doubling the previous value of T after each failure; we don't consider such issues here, instead leaving the choice of T entirely up to the user.)

The main result of this paper is the following:

Theorem. *For any $T \in \mathbf{N}$, and for π-almost every $Z \in \mathcal{X}$ such that the algorithm has positive probability of succeeding, if we run the algorithm corresponding to T and Z,* *then conditional on success of the algorithm, the output value X_0 is distributed exactly according to $\pi(\cdot)$.*

That is, if the algorithm succeeds, then it produces an exact sample from $\pi(\cdot)$, just as we wanted. Note in particular that no assumption is made regarding monotonicity, discreteness of the state space, the existence of densities, etc.

Furthermore, the algorithm works from essentially any starting point Z, not just from certain special points such as minimal or maximal elements. The starting point must be such that the algorithm has positive probability of success, but this is not a difficulty in practice: if the algorithm does *not* have positive probability of success then it will never succeed, so it will never produce an incorrect value.

The algorithm also works only for π-almost every Z rather than every Z for which it has positive probability of success. This is unavoidable since the transition probabilities can be changed arbitrarily on a set of measure 0 without affecting stationarity. Of course, on a *discrete* space in which $\pi(x) > 0$ for all $x \in \mathcal{X}$, this puts absolutely no restriction on Z. Similarly, if no point of the state space is unusual, e.g. if the T-step transitions have densities which are continuous functions, then again the "π-almost every" condition can be ignored. In addition, we note that it is also acceptable to choose Z *randomly* (provided it is independent of the subsequent randomness of the algorithm). In this case, the "π-almost every" condition can be ignored whenever the distribution of Z is absolutely continuous with respect to $\pi(\cdot)$.

3. Proof of the theorem.

In this section, we prove the theorem, i.e. we prove the validity of our algorithm.

For ease of computation, we begin by assuming that $X_T \sim \pi(\cdot)$. We shall later condition on the event $X_T = Z$.

We define C to be the event that the $N_{x,t}$ values coalesce, i.e. that $N_{x,0}^T$ does not depend on x so the algorithm is successful. We then define the measure $\nu(\cdot)$ by

$$\nu(dy) \;=\; \mathbf{P}(C,\; X_T \in dy)\,,$$

so that $\nu(\mathcal{X}) = \mathbf{P}(C)$. Intuitively, $\nu(\cdot)/\mathbf{P}(C)$ is the conditional distribution of X_T, conditional on the event C.

Next, we note that for any $x \in \mathcal{X}$, we must have $\nu(dy) \leq P^T(x, dy)$ (i.e., the chance of coalescing *and* hitting dy is no more than the chance that the single path from x hits

dy). Hence, in particular $\nu(\cdot) \ll P^T(x,\cdot)$, so we can define the Radon-Nikodym derivative

$$h_x(\cdot) \;=\; \frac{d\nu(\cdot)}{dP^T(x,\cdot)}\,.$$

Our key computation is the following.

Lemma. *Suppose $X_T \sim \pi(\cdot)$, and that we then compute $X_{T-1},\ldots,X_0$ and $N_{x,t}$ as in our algorithm. Let C be the event of coalescence as above. Then*

$$\mathbf{P}(X_T \in dx_T,\; X_0 \in dx_0,\; C) \;=\; h_{x_0}(x_T)\,\widetilde{P}^T(x_T,dx_0)\,\pi(dx_T)\,.$$

Proof. We have that

$$\begin{aligned}
&\mathbf{P}(X_T \in dx_T,\, X_0 \in dx_0,\; C)\\
&\qquad = \mathbf{P}(C \,|\, X_T \in dx_T,\, X_0 \in dx_0)\;\mathbf{P}(X_0 \in dx_0 \,|\, X_T \in dx_T)\;\mathbf{P}(X_T \in dx_T)\\
&\qquad = \left(\frac{\mathbf{P}(C,\, X_T \in dx_T \,|\, X_0 \in dx_0)}{\mathbf{P}(X_T \in dx_T \,|\, X_0 \in dx_0)}\right)\;\widetilde{P}^T(x_T,dx_0)\;\pi(dx_T)\\
&\qquad = \left(\frac{\nu(dx_T)}{\mathbf{P}^T(x_0,\,dx_T)}\right)\;\widetilde{P}^T(x_T,dx_0)\;\pi(dx_T)\\
&\qquad = (h_{x_0}(x_T))\;\widetilde{P}^T(x_T,dx_0)\;\pi(dx_T)\,,
\end{aligned}$$

as claimed. ∎

To finish the proof, we now condition on the event $X_T = Z$ (in addition to conditioning on the success event C). Using (1), we have that with probability one,

$$\begin{aligned}
\mathbf{P}(X_0 \in dx_0 \,|\, C,\; X_T = Z) \;&=\; \frac{\mathbf{P}(C,\; X_T \in dZ,\; X_0 \in dx_0)}{\mathbf{P}(C,\; X_T \in dZ)}\\
&=\; \frac{h_{x_0}(Z)\;\widetilde{P}^T(Z,dx_0)\;\pi(dZ)}{\nu(dZ)}\\
&=\; \frac{h_{x_0}(Z)\;\mathbf{P}^T(x_0,dZ)\;\pi(dx_0)}{\nu(dZ)}\\
&=\; \frac{h_{x_0}(Z)\;\pi(dx_0)}{h_{x_0}(Z)}\\
&=\; \pi(dx_0)\,.
\end{aligned}$$

(Note that the Radon-Nikodym derivative $\nu(dZ)/\pi(dZ)$ is precisely the probability that the algorithm succeeds when started at $X_T = Z$; this probability is positive by assumption, so it is acceptable to use its reciprocal in the above computation.)

Hence, it follows that, conditional on $X_T = Z$, and conditional on the success of the algorithm (the event C), the conditional distribution of the output value X_0 is precisely the target distribution $\pi(\cdot)$. The completes the proof of the validity of the algorithm. ∎

4. A simple example.

We illustrate our algorithm for a very simple example. We suppose that the state space is $\mathcal{X} = \{0, 1, 2\}$, i.e. consists of just three points. We suppose that the target distribution $\pi(\cdot)$ is uniform over these three points. We take as our Markov chain the Metropolis algorithm which proposes from a simple symmetric random walk and rejects any moves outside of $\mathcal{X}$; thus,

$$P(0,0) = P(0,1) = P(1,0) = P(1,2) = P(2,1) = P(2,2) = \frac{1}{2}\,.$$

Note that P is reversible with respect to $\pi(\cdot)$, so that $\widetilde{P} = P$.

To apply our algorithm, we need to specify the joint structure of the chain when moving simultaneously from two different points, i.e. the joint distribution of $N_{x,t}$ and $N_{y,t}$ for $x \neq y$. It is of course straightforward to do this in such a way that monotonicity is preserved, i.e. so that $N_{x,t} \leq N_{y,t}$ whenever $x \leq y$. However, let us instead specify that $N_{x,t}$ and $N_{y,t}$ are *independent* when $x \neq y$, i.e. that simultaneous moves from x and from y are independent.

This joint structure does *not* preserve monotonicity, e.g. there is probability $\frac{1}{4}$ that $N_{1,t} < N_{0,t}$ even though $1 > 0$. Thus, the original Fill (1998) algorithm can not be used in this setting. However, our modified algorithm can indeed be used.

Specifically, our algorithm proceeds as follows. We choose T and Z; to be definite, let us choose $T = Z = 2$. We then set $X_2 = 2$, choose $X_1 \sim P(X_2, \cdot)$, and choose $X_0 \sim P(X_1, \cdot)$. This automatically specifies $N_{X_0,0}$ and $N_{X_1,1}$. We then *independently* choose $N_{x,t}$ for the remaining four pairs (x,t), i.e. for $t \in \{0,1\}$ and $x \in \mathcal{X} \setminus \{X_t\}$.

To conclude the algorithm, we compute whether or not the three paths starting from 0, 1, and 2 at time 0 have all coalesced by time 2, i.e. whether or not $N^2_{0,0} = N^2_{1,0} = N^2_{2,0}$. If yes then we output X_0, otherwise we start the algorithm again.

This example is simple enough that we can analyze it exactly. We see that the algorithm involves a total of six random events, each of which choose between two equally likely outcomes. There are thus a total of $2^6 = 64$ possible overall outcomes, each having probability $1/64$. We compute that, of the 64 possible outcomes, exactly 12 of them cause coalescence. Of these 12 coalescing outcomes, exactly 4 have $X_0 = 0$, another 4 have $X_0 = 1$, and the final 4 have $X_0 = 2$.

We thus conclude that, for this algorithm, with independent joint updates and with $T = Z = 2$, the algorithm will succeed with probability $12/64$; and when it does succeed it will provide an exact sample from the uniform distribution $\pi(\cdot)$. (An identical result is true if instead $Z = 1$ or $Z = 0$.)

Of course, if the size of the state space were much larger, then the probability of success might be quite small. In this case, it might be preferable to choose a more sophisticated joint updating structure for the $N_{x,t}$ (e.g. having the chains all increase/decrease simultaneously, cf. Murdoch and Rosenthal, 1998), rather than having them all be independent. In general, the question of how to choose a good joint updating structure for a complicated Markov chain is quite difficult; for discussion see e.g. Propp and Wilson (1996), Murdoch and Green (1998), Møller and Schladitz (1998), and Foss and Tweedie (1997).

5. Discussion.

Fill (1998) assumed monotonicity of the Markov chain, i.e. that there is some partial order $\preceq$ on $\mathcal{X}$ such that $x \preceq y$ implies $N_{x,t} \preceq N_{y,t}$ almost surely. With monotonicity and unique minimal and maximal elements $\hat{0}$ and $\hat{1}$ respectively, testing whether or not $N_{x,0}^T$ is independent of x becomes very simple: one need only check that $N_{\hat{0},0}^T = N_{\hat{1},0}^T$, since all other values will be sandwiched between. In Fill's context of a discrete state space, the test could be even simpler: he chose $Z = \hat{0}$ and then only needed to test whether $N_{\hat{1},0}^T = \hat{0}$. Møller and Schladitz (1998) and Thönnes (1997) noted that $Z = \hat{0}$ may not always be a legal choice in a general state space, and they extended the algorithm to deal with this. However, they too assumed monotonicity of the chain.

In this paper, we have presented a further extension of Fill's algorithm, which no longer requires monotonicity.

Of course, our algorithm still requires that we check for coalescence. Without monotonicity, testing the values of $N_{x,0}^T$ is more difficult. Murdoch and Green (1998) and Green and Murdoch (1998) describe algorithms to achieve this in the context of coupling from the past; these algorithms appear to be limited to fairly low dimensional state spaces.

How does Fill's algorithm compare to CFTP? It has two main advantages. First, as Fill (1998) showed, the independence between path length and the sampled value removes the "user impatience bias" that would arise if a user gave up on long calculations and only reported short ones. Second, for fixed T, Fill's algorithm will often be implementable in a fixed amount of storage space. (This may depend on how difficult it is to carry out the test described in the previous paragraph.) Unfortunately, it is rare that a suitable value of T is known in advance, so larger and larger values of T (e.g. from Fill's doubling strategy) will usually be needed, and this advantage will be partially lost.

There are two main disadvantages to Fill's algorithm. First, the conditional sampling of $N_{x,t}$ given the initial path may be quite difficult to carry out. Second, it is sometimes possible with CFTP algorithms to extract more information from the CFTP runs and obtain correspondingly better estimators at little additional computational cost (Murdoch and Rosenthal, 1998). However, this does not seem possible with Fill's algorithm, because the intermediate values are generally not drawn from $\pi(\cdot)$. In our simple example, if Z were chosen at random from $\pi(\cdot)$, then X_1 would also be drawn from $\pi(\cdot)$, but if we modify the transition probabilities to favour states 0 and 2 over state 1, this is no longer true: conditioning on coalescence forces X_1 to visit state 1 too frequently.

Propp and Wilson's CFTP algorithm has already shown its worth in simulating point process models and large discrete models, and shows promise in Bayesian computations. We hope that this discussion of Fill's algorithm will stimulate further research into this less-used alternative for exact MCMC sampling.

REFERENCES

Brooks, S.P. and Roberts, G.O. (1997). Assessing Convergence of Markov Chain Monte Carlo Algorithms. Preprint, University of Bristol.

Corcoran, J. and Tweedie, R.L. (1998). Perfect simulation of Harris recurrent Markov chains. Preprint, Colorado State University.

Cowles, M.K. and Carlin, B.P. (May, 1995). Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review. Preprint, University of Iowa.

Fill, J.A. (1995). Unpublished.

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

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

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

Gilks, W.R., Richardson, S., and Spiegelhalter, D.J., editors (1996). *Markov Chain Monte Carlo in Practice*. Chapman and Hall.

Green, P.J. and Murdoch, D.J. (1998). Exact sampling for Bayesian inference: towards general purpose algorithms. Preprint.

Kendall, W. (1997). Perfect simulation for the area-interaction point process. In Heyde, C.C. and Accardi, L., editors, *Probability Perspective*. World Scientific Press. To appear.

Kendall, W. and Thönnes, E. (1998). Perfect simulation in stochastic geometry. Preprint, University of Warwick.

Meyn, S.P. and Tweedie, R.L. (1994). Computable bounds for convergence rates of markov chains. *Annals of Applied Probability*, 4:981–1011.

Møller, J. (1997). Perfect simulation of conditionally specified models. *Journal of the Royal Statistical Society, Series B*, to appear.

Møller, J. and Schladitz, K. (1998). Extensions of Fill's algorithm for perfect simulation. Preprint.

Murdoch, D.J. and Green, P.J. (1997). Exact sampling from a continuous state space. *Scandinavian Journal of Statistics*, to appear.

Murdoch, D.J. and Rosenthal, J.S. (1998). Efficient use of exact samples. Preprint.

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

Propp, J.G. and Wilson, D.B. (1998). How to get a perfectly random sample from a generic Markov chain and generate a random spanning tree of a directed graph. *Journal of Algorithms*, 27:170–217.

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

Smith, A.F.M. and Roberts, G.O. (1993). Bayesian computation via the gibbs sampler and related markov chain monte carlo methods (with discussion). *Journal of the Royal*

Statistical Society, Series B, 55:3–24.

Thönnes, E. (1997). Perfect Simulation of some Point Processes for the Impatient User. Preprint, University of Warwick.

Tierney, L. (1994). Markov chains for exploring posterior distributions (with discussion). *Annals of Statistics*, 22:1701–1762.