

Perfect simulation from unbiased simulation

George M. Leigh^{1*}, Wen-Hsi Yang^{2,3},
Montana E. Wickens¹ and Amanda R. Northrop⁴

¹Fisheries Queensland, Department of Agriculture and Fisheries
41 George St., Brisbane Qld 4000, Australia

²School of Mathematics and Physics, University of Queensland
St Lucia, 4072, Queensland, Australia

³School of Agriculture and Food Sciences, University of Queensland
St Lucia, 4072, Queensland, Australia

⁴Fisheries Queensland, Department of Agriculture and Fisheries
Mayers Road, Nambour Qld 4560, Australia

*Corresponding author email: george.leigh@daf.qld.gov.au, george.m.leigh@gmail.com
Contributing authors: w.yang@uq.edu.au;
montana.wickens@daf.qld.gov.au; amanda.northrop@daf.qld.gov.au

ORCiDs: GL 0000-0003-0513-2363, WY 0000-0001-8236-0082,
MW 0000-0002-7316-014X, AN 0000-0002-8661-2459

August 15, 2023

Abstract

We show that any application of the technique of unbiased simulation becomes perfect simulation when coalescence of the two coupled Markov chains can be practically assured in advance. This happens when a fixed number of iterations is high enough that the probability of needing any more to achieve coalescence is negligible; we suggest a value of 10^{-20} . This finding enormously increases the range of problems for which perfect simulation, which exactly follows the target distribution, can be implemented. We design a new algorithm to make practical use of the high number of iterations by producing extra perfect sample points with little extra computational effort, at a cost of a small, controllable amount of serial correlation within sample sets of about 20 points. Different sample sets remain completely independent. The algorithm includes maximal coupling for continuous processes, to bring together chains that are already close. We illustrate the methodology on a simple, two-state Markov chain and on standard normal distributions up to 20 dimensions. Our technical formulation involves a nonzero probability, which can be made arbitrarily small, that a single perfect sample point may have its place taken by a “string” of many points which are assigned weights, each equal to ± 1 , that sum to 1. A point with a weight of -1 is a “hole”, which is an object that can be cancelled by an equivalent point that has the same value but opposite weight $+1$.

Keywords: Markov chain Monte Carlo, maximal coupling, MCMC convergence diagnostics

1 Introduction

Monte Carlo methodology has become an indispensable numerical tool in science (Liu, 2004), most notably in the highly versatile form of Markov chain Monte Carlo (MCMC) (Brooks et al., 2011). MCMC uses random numbers to iterate a Markov process and produce chains of simulated random variables whose distributions gradually converge to a desired target distribution. A critical part of MCMC practice, therefore, comprises the evaluation of MCMC convergence.

Numerous numerical and visual tools for diagnosing MCMC convergence are commonly applied (see Roy, 2020, and references therein). None of these tools can detect convergence exactly in practice and show itself consistently superior to others; hence, practitioners often use several of them together. In addition, MCMC generally produces serially correlated samples, leading to slow convergence and ending in poor approximation to the target distribution. As a result, it is common to run independent chains for as many iterations as computationally feasible, and use thinning (Gelman and Shirley, 2011) to reduce serial correlation, although this results in discarding most of the generated MCMC sample points.

The development of unbiased simulation (Glynn and Rhee, 2014; Jacob et al., 2020) has shown that coupling two Markov chains can produce an unbiased simulation of target distributions without excessive computation. In the simulation, the two coupled chains X and Y have the same distribution of starting points, which are generated independently, after which their random numbers (which are used, for example, to generate a proposed sample point for the next MCMC step and to decide whether to accept or reject it) are offset by one iteration. The transition from Y_0 to Y_1 uses the same random numbers as that from X_1 to X_2 , etc. Unbiased simulation is computationally effective in requiring only two chains to be simulated and not requiring the amount of iteration to be known in advance. Simulation can be halted immediately upon coalescence when the use of the same random numbers makes $X_i = Y_{i-1}$ for some index i .

The flexibility of unbiased simulation has allowed it to be implemented in sequential Monte Carlo (van den Boom et al., 2022) and in Hamiltonian Monte Carlo (HMC) (Heng and Jacob, 2019; Leigh and Northrop, 2022). Its use in HMC is especially attractive and offers the capability, for quite general continuous processes, to completely overcome the convergence difficulties that plague current usage of MCMC. We will return to HMC in the Discussion section of this paper.

The disadvantage of unbiased simulation is that its output consists not of coalesced sample values but of potentially lengthy sums that can vary wildly and even produce values outside the allowed sample space, such as estimated probability values that are negative or greater than 1.

Perfect simulation overcomes the above drawback of unbiased simulation by generating samples that exactly follow the target distribution and are independent (see reviews by Craiu and Meng, 2011; Huber, 2016). Currently, few problems are amenable to perfect simulation, whereas a much greater range of problems is suited to unbiased simulation.

The most versatile existing algorithm that can achieve perfect simulation is coupling from the past (CFTP) (Propp and Wilson, 1996), but even that algorithm’s formal accomplishment of perfect simulation is limited to a small subset of the total range of problems suitable for MCMC. CFTP uses coupling to produce coalescence from different starting points, and constitutes perfect simulation when it can be proven that the same outcome occurs from every possible starting point. CFTP samples forward in time but achieves coalescence by prepending extra random numbers as needed to the beginning of the sequence, not the end, thereby avoiding bias towards “easier” sets of random numbers that yield faster coalescence. The variant read-once coupling from the past (ROCFTP) (Wilson, 2000) proceeds only in forward sequence, in blocks of fixed length, some of which exhibit coalescence while others may not.

Proof of coalescence from every possible starting point in CFTP can be achieved through a *monotonicity* property which is possessed by some processes (Propp and Wilson, 1996). In general, however, monotonicity is not available. Even a discrete-valued process whose transition probabilities are *a priori* not known numerically but are calculated on the run would have to visit every possible state to establish perfect simulation with CFTP or ROCFTP (Lovász and Winkler, 1995; Fill, 1998, p. 135). Continuous processes with non-countable states magnify this difficulty.

Diverse perfect simulation algorithms either build on CFTP or are constructed independently of CFTP. Notable examples include multigamma coupling (Murdoch and Green, 1998), perfect slice sampling (Mira et al., 2001) and strong stationary stopping times (Aldous and Diaconis, 1987). These algorithms are generally difficult to use and not widely applicable. Multigamma coupling has an impossibly small probability of satisfying the conditions for perfect simulation for most practical problems. Perfect slice sampling has the difficulty of establishing, at least approximately, the boundary of the feasible space, which leads to a curse of dimensionality (Bellman, 1955) in high-dimensional spaces.

In this paper, we will show in section 2 that any application of unbiased simulation becomes perfect simulation when coalescence can be practically assured in advance, i.e., when a high enough, fixed number of iterations are performed that the probability of needing any more to achieve coalescence is negligible. This finding enormously increases the range of problems for which perfect simulation can be implemented. In section 3, we design a new algorithm to make practical use of the extra computation, to produce extra perfect sample points. In section 4, we illustrate the methodology on two example processes, one discrete and one continuous.

2 From unbiased to perfect simulation

2.1 Recap of unbiased simulation

Unbiased simulation is formulated by two coupled Markov chains that define the following random variable G (see Jacob et al., 2020, p. 545):

$$G = g(X_k) + \sum_{i=k+1}^{\infty} \{g(X_i) - g(Y_{i-1})\}, \quad (1)$$

where X and Y are chains from the same homogeneous Markov process, whose starting points X_0 and Y_0 are independent and identically distributed; g is a pre-defined, fairly arbitrary function whose expectation $\mathbb{E}(g(X_i))$ exists for $i \geq 0$ and as $i \rightarrow \infty$; and $k \geq 0$ is a pre-chosen number of iterations akin to burn-in. Because X_0 and Y_0 are identically distributed, and X and Y follow the same Markov process, X_i and Y_i are also identically distributed for $i > 0$.

The same random numbers are used in X and Y , but they are offset by one iteration so that those for X_i are used for Y_{i-1} . Chains X and Y are designed to coalesce, with probability 1, to the same values, again offset by one iteration: after some random number of iterations τ , $X_\tau = Y_{\tau-1}$. Because X and Y are homogeneous Markov chains whose transitions depend only on the current state, they remain coalesced, so that $X_i = Y_{i-1}$ for all $i \geq \tau$. We emphasize that, despite this coalescence, it is X_i and Y_i (with no offset) that are identically distributed. The offset variables X_i and Y_{i-1} are, in general, not identically distributed for any fixed value of i , because the time to coalesce, τ , has no fixed upper bound.

Due to the coalescence, the infinite sum in (1) converges and any realization of it contains, with probability 1, only a finite, although random, number of terms. Being finite, the sum can be calculated exactly, subject only to available numerical precision, with no need for asymptotic approximations.

Because X and Y , this time with no offset in the subscript, are identically distributed, $\mathbb{E}(g(Y_{i-1}))$ can be replaced by $\mathbb{E}(g(X_{i-1}))$ in the expectation of the random variable G in (1), which becomes

$$\begin{aligned} \mathbb{E}(G) &= \mathbb{E}(g(X_k)) + \sum_{i=k+1}^{\infty} \{\mathbb{E}(g(X_i)) - \mathbb{E}(g(X_{i-1}))\} \\ &= \lim_{i \rightarrow \infty} \mathbb{E}(g(X_i)) = \mathbb{E}(g(Z)), \end{aligned} \quad (2)$$

where the random variable Z is drawn from the stationary distribution of X and Y . Therefore G is an unbiased simulation of the distribution of $g(Z)$. We do not refer to it as a ‘‘sample’’ because it may lie outside the range of $g(Z)$.

It is essential that the coalescence after τ iterations occur for the original chains X and Y : it is not sufficient that $g(X_\tau) = g(Y_{\tau-1})$, because then it may be possible for $g(X)$ and $g(Y)$ to later diverge.

2.2 High probability of a perfect sample

It is important that this section be taken in combination with the next section 2.3. We state below that the probability that the theory in section 2.3 will be required in practice can be made arbitrarily small, but this does not mean that the event it covers can be neglected. To establish perfect sampling, that theory remains critical.

We postulate a random variable Q that does not depend on the function g and whose conditional expectation satisfies, for any choice of g ,

$$\mathbb{E}(g(Q) \mid X, Y) = G = g(X_k) + \sum_{i=k+1}^{\infty} \{g(X_i) - g(Y_{i-1})\}. \quad (3)$$

In this section we show that such a random variable, if it exists, is a perfect sample from the target distribution, and that, when a large value is chosen for the burn-in length k , there is a high probability that $Q = X_k$. The form of Q when $Q \neq X_k$ is discussed in the next section 2.3.

Consider the case of taking $g = I_A$, an indicator function defined on an arbitrary measurable subset A of the range of X and Y . Then, from (2) and (3),

$$\mathbb{E}(I_A(Q)) = \mathbb{E}(G) = \mathbb{E}(I_A(Z)) = \mathbb{P}(Z \in A). \quad (4)$$

This equation holds for any choice of A ; and Q , as postulated, does not depend on g or A . The probability distributions of Q and Z are completely determined by the probabilities that these variables belong to an arbitrary subset of their range. Therefore, (4) implies that these distributions must be the same; i.e., if a random variable Q can be found that satisfies (3) for all choices of g , it must be a perfect sample from the stationary distribution of X and Y .

If the number of iterations τ , at which X and Y coalesce, is less than or equal to $k + 1$, Q is simply equal to X_k , as the sum in (3) is then empty.

Therefore, setting k to a high quantile of τ results in a high probability that X_k is a perfect sample, provided that Q can still be defined when $\tau > k$. We explore the latter case in the next section. The probability $\mathbb{P}(\tau \leq k)$ can be made arbitrarily close to 1 by choosing a high enough value for k .

2.3 Relaxed perfect sampling

However large k is chosen, the probability that $\tau > k + 1$, i.e., that X and Y do not coalesce within $k + 1$ iterations, will still be nonzero. Here we describe what to do when this occurs.

We set Q to an improper conditional distribution which we call a “string” of values q_i and associated weights w_i :

$$\begin{aligned} Q &= ((q_1, w_1), (q_2, w_2), \dots) \\ &= ((X_k, 1), (Y_k, -1), (X_{k+1}, 1), (Y_{k+1}, -1), \dots) \end{aligned} \quad (5)$$

where a weight of 1 signifies a conventional sample point, while a weight of -1 signifies an improper sample point which we call a “hole”. A hole is defined as an object that would be cancelled by a conventional sample point with the same value q but opposite weight $w = 1$. It is similar to a positively charged hole in chemistry which does not exist as an actual particle but is cancelled by a negatively charged electron that moves into the vicinity.

As in (1), due to the coupling of X and Y , the sequence in a realization of (5) is finite, terminating after $\nu = 2(\tau - k) - 1$ terms at the final element $(X_{\tau-1}, 1)$. Therefore a realization of Q can be calculated exactly, in the same way as a realization of G in (1). The weights within a string satisfy $w_i = (-1)^{i-1}$ and $\sum_{i=1}^{\nu} w_i = 1$.

We define the conditional expectation of a function of Q in the same way as for a discrete distribution, with the weights w_i in place of a probability function:

$$\mathbb{E}(g(Q) \mid X, Y) = w_1 g(q_1) + \sum_{i=1}^{\infty} \{w_{2i} g(q_{2i}) + w_{2i+1} g(q_{2i+1})\}. \quad (6)$$

Therefore, Q satisfies (3), as the w 's are equal to ± 1 and the q 's are equal to X and Y . By the theory in section 2.2, Q is, in a relaxed sense due to the presence of negative weights, a perfect sample.

The presence of holes in a sample does not affect the validity of (4), but does affect the dispersion of the sample mean of $I_A(Q)$ (see examples in the final column of Table 1 in section 4.1 below). When Q is a string consisting of both conventional sample points and holes, it contributes ν points instead of only one. Therefore, its presence or absence in the sample will make a bigger difference to the sample mean than a single point would make. If strings with $\nu > 1$ are rare due to a high choice for k , their effect on the overall precision of estimation will be small.

2.4 Comments

We stress that the target distribution should be explored both theoretically and numerically, using a widely dispersed distribution of starting points, before the value of k is set. Consequences of the presence of holes can be severe: a single string can be very long and contain many holes, e.g., when the target distribution is multi-modal, with one region of the sample space almost cut off from the rest, and this is not appreciated before setting k .

In the following section, we present methodology to reduce the computational effort per perfect sample point, at the expense of a small amount of serial correlation within relatively small sample sets of typically 20 sample points. Different sample sets remain independent.

3 Practical algorithm for perfect simulation

3.1 Algorithm structure

The theory in section 2 achieves perfect simulation by setting the number of iterations, k in equation (1), very high. In this section, we present a practical algorithm to reduce the amount of computation.

The algorithm groups sample points into sample sets of size K , within which sample points are allowed to have small correlations. A thinning ratio or block length, B , is chosen which determines the magnitude of the internal correlations. The burn-in number of iterations is set to $k = KB$. We offset the coupled chains X and Y in (1) by B iterations instead of just one iteration, and run each chain for K blocks each of B iterations, noting that the process whose "iteration" is a block of K iterations of X or Y is still a Markov process. Equation (1) is modified to

$$G = g(X_{KB}) + \sum_{i=K+1}^{\infty} \{g(X_{iB}) - g(Y_{(i-1)B})\}. \quad (7)$$

Parameter B is chosen to regulate the probability that the chains coalesce within one block:

$$\mathbb{P}(X_{2B} = Y_B) \approx 1 - P, \quad (8)$$

where the starting points X_0 and Y_0 are chosen independently from a widely dispersed distribution that is thought to cover the range of X and Y , and P is an acceptable probability of lack of coalescence after B iterations. A nonzero probability of non-coalescence allows some correlation between chains in the same sample set. This can be reduced by using a smaller value of P , at the expense of fewer points in the sample set for the same amount of computation. Equation (8) is an approximation because the value of B as a function of P generally cannot be found analytically and has to be determined by a preliminary simulation experiment.

The sample-set size K is determined by the desired probability of non-coalescence after $K + 1$ blocks of B iterations. With widely dispersed starting points, continuing from the end of the previous block should be no less successful than running from the original starting points. Hence the probability of non-coalescence can be expected to decay geometrically, reaching approximately P^K after $K + 1$ blocks. Lack of coalescence after $K + 1$ blocks produces holes in the results, as described in section 2.3. A larger choice of K reduces the probability of occurrence of holes.

We suggest that P can be set to 0.1 and K around 20, to produce a probability of a hole of about 10^{-20} , provided that the sample space has been thoroughly analysed and the distribution of starting points is widely dispersed. Higher values of K can be used if desired. The value of K should also be increased according to the total number of sample points to be generated, n , so that the probability of getting any holes within the entire set of results, approximately nP^K , is kept very low.

Our algorithm is shown in Fig. 1, in the form of a $K \times K$ matrix, and in Algorithm 1. Each element of the matrix corresponds to a block of B MCMC iterations. Initialization to a starting point takes place at the beginning of each block on the diagonal of the matrix. Blocks are then applied from left to right, the input to each off-diagonal block being the output from the block to the left of it. Blocks wrap around back to column 1 after reaching column K , and, with high probability, produce a perfect sample at the end of each block immediately to the left of the diagonal. For purposes of illustration, Algorithm 1 shows the result variable Q , which represents a column of Fig. 1, as a vector. For a d -dimensional process with $d > 1$, it is actually coded as a $K \times d$ matrix. The same is true for the variable Q_1 which stores the first row of Fig. 1.

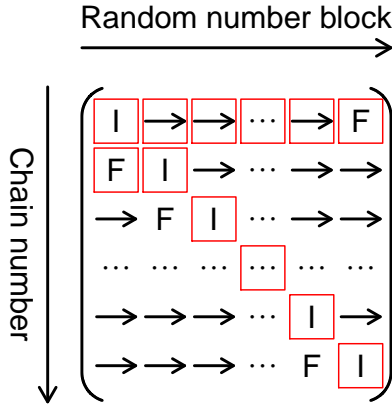


Figure 1: Chain \times block matrix for one sample set. Random-number blocks proceed from left to right and are common to all chains. Each chain is initialized on its diagonal element “I” to a starting point from some preset distribution. Blocks and chains wrap around when necessary. The final cell “F” in each chain is a perfect sample if it coalesces with the chain below it. Cells marked by red squares are always executed; others only if chains haven’t yet coalesced.

Each column of the matrix in Fig. 1 represents a block of random numbers: the same random numbers are used in that column in every row, to couple each row with the one below it. Then each row provides a perfect sample point after K blocks if it has coalesced with the row below at that time, or if it coalesces after one extra block with new random numbers, making $K + 1$ blocks for the upper row and K blocks for the lower row; the two rows are used as the processes X and Y in (7). Chain K is required to coalesce with row 1 in column $K - 1$, or after one extra block with new random numbers.

Algorithm 1, line 2, with continuous processes in mind, requires a metric to measure the relative closeness of one chain to another, and hence (lines 14 and 29) to choose the previous chain most likely to coalesce with the current chain. For continuous processes as discussed in section 3.2, this is intended to be the Euclidean metric, i.e., the usual absolute value function.

Algorithm 1 Practical algorithm for one perfect sample set of size K

Require: $K \geq 1$, $B \geq 1$, functions $\text{START}(K)$, $\text{RAND}(B)$, $\text{MCMC}(X_0, R, B)$.

Ensure: Q stores K perfect sample points from MCMC, if $\text{Error} = \text{false}$.

```
1: function MININD( $x, i_1, i_2$ ) ▷ Index of closest of  $x[i_1 : (i_2 - 1)]$  to  $x[i_2]$ 
2:   return WHICHMIN( $|x[i_2] - x[i_1 : \{i_2 - 1\}]|$ ) +  $i_1 - 1$  ▷ Metric required
3: end function ▷ WHICHMIN: index of minimum, first one if there are ties
4:  $Q \leftarrow \text{START}(K)$  ▷ Vector of random starting points on diagonal of Fig. 1
5:  $Q_1 \leftarrow \text{zeros}(K)$  ▷ Row vector to store first row of Fig. 1 for later use.
6:  $s \leftarrow \text{SEED}$ ;  $\text{Error} \leftarrow \text{false}$  ▷ Store random seed; set coalescence error.
7:  $a \leftarrow \text{zeros}(K)$  ▷ Mark all chains as active (not coalesced yet).
8: for  $j \leftarrow 1 : K$  do ▷ Loop over blocks (columns) in Fig. 1 upper triangle.
9:    $R \leftarrow \text{RAND}(B)$  ▷ Generate random numbers for  $B$  MCMC iterations.
10:  for  $i \leftarrow 1 : j$  do ▷ Loop over chains (rows) in Fig. 1 upper triangle.
11:    if  $a[i] > 0$  then  $Q[i] \leftarrow Q[a[i]]$  else ▷ Copy if already coalesced.
12:       $Q[i] \leftarrow \text{MCMC}(Q[i], B, R)$  ▷ MCMC from current value of  $Q$ 
13:    if  $i = 1$  then  $Q_1[j] \leftarrow Q[1]$  else ▷ Store first row of Fig. 1.
14:       $m \leftarrow \text{MININD}(Q, 1, i)$  ▷ Closest previous value in column  $j$ 
15:      if  $Q[i] = Q[m]$  then  $a[i] \leftarrow m$  end if ▷ Record coalescence.
16:    end if ▷ Nonzero  $a[i]$  records row with which row  $i$  coalesced.
17:  end if
18: end for ▷  $i$  loop (rows of Fig. 1)
19: end for ▷  $j$  loop (columns of Fig. 1)
20:  $\text{SEED} \leftarrow s$  ▷ Restore seed to regenerate previous random number blocks.
21: for  $j \leftarrow 1 : (K - 1)$  do ▷ Loop over blocks in Fig. 1 lower triangle.
22:    $\text{Error} \leftarrow \text{Error}$  or  $a[j + 1] \neq j$  ▷ Check final coalescence of chain  $j$ .
23:    $a[j + 1] \leftarrow a[j]$ ;  $a[a = j] \leftarrow j + 1$  ▷ Move coalescence from row  $j$  to  $j + 1$ .
24:    $Q[1] \leftarrow Q_1[j]$  ▷ Reload row 1 saved copy; no need to recompute.
25:    $R \leftarrow \text{RAND}(B)$  ▷ Regenerate random numbers for block  $j$ .
26:   for  $i \leftarrow (j + 1) : K$  do ▷ Loop over chains in Fig. 1 lower triangle.
27:     if  $a[i] > 0$  then  $Q[i] \leftarrow Q[a[i]]$  else ▷ Copy, else run MCMC.
28:        $Q[i] \leftarrow \text{MCMC}(Q[i], B, R)$  ▷ Run MCMC & check coalescence.
29:     if  $i = j + 1$  then  $m \leftarrow 1$  else  $m \leftarrow \text{MININD}(Q, j + 1, i)$  end if
30:     if  $Q[i] = Q[m]$  then  $a[i] \leftarrow m$  end if ▷ Coalesced with row  $m$ 
31:   end if ▷ Lower triangle of Fig. 1 concerns rows 1 and  $(j + 1) : i$ 
32: end for ▷  $i$  loop (rows of Fig. 1)
33: end for ▷  $j$  loop (columns of Fig. 1)
34:  $\text{Error} \leftarrow \text{Error}$  or  $a[K] \neq 1$ ;  $Q[1] \leftarrow Q_1[K]$  ▷ Check  $Q[K]$ , restore  $Q[1]$ .
```

For discrete processes with unordered states, a simple zero-one function suffices, taking the value zero when the states are equal and 1 when they are different.

Extra iteration to that detailed in Algorithm 1, using new random numbers as needed, is required to calculate the right-hand side of (7) if a row does not coalesce with the row below it within K blocks. Holes occur if the two rows still do not coalesce after $K + 1$ blocks.

We emphasize that different sample sets remain completely independent. Correlations occur only within the same sample set of K points.

Our algorithm produces a sample set of K points with similar computational effort to that for two independent perfect sample points. Most of the chains coalesce after a single block, as determined by the probability of non-coalescence P in (8). After a row has coalesced with a row above it in Fig. 1, computation for the lower row can be curtailed and its results set to those for the higher row. Computation always has to be conducted for the cells marked by red squares: along row 1, the diagonal, and block 1 in row 2.

The algorithm can be parallelized easily by assigning different sample sets, which are uncon-

nected and contain roughly 20 points, to different processors.

3.2 Maximal coupling for continuous processes

Algorithm 1 is designed to facilitate maximal coupling in simulation of continuous processes. Maximal coupling originated as a method of analysis of convergence of a Markov chain to its stationary distribution, in which one of the coupled chains began from some prescribed initial distribution and the other from the stationary distribution (Griffeath, 1975). It evolved into a scheme to facilitate coalescence of any two chains whose probability densities overlap (see, e.g., Johnson, 1998).

In our setting, maximal coupling inserts into each chain a proposed jump to a new destination point, which is accepted with some probability according to a Metropolis–Hastings (M–H) update. The destination points from two different chains can, with a certain probability given by the overlap of their probability densities, be made equal while still following their preset jump distribution. An M–H update (Metropolis et al., 1953; Hastings, 1970) accepts a move from an existing point X to a proposed new point X^* with probability $\min(\exp\{U(X) - U(X^*)\}, 1)$, where U is the negative log-likelihood function.

It is sometimes feasible to maximally couple chains directly in their internal MCMC, without the need for a separate M–H update. The method in this section is general, making no assumptions about internal MCMC structure.

We advocate a solid spherical uniform distribution with a fixed radius r for maximal-coupling jumps. In d dimensions, the destination point X^* is chosen uniformly over the interior of a hypersphere of radius r , centred at the existing point X . We presume that, when $d > 1$, the d coordinates of the process have been scaled to be non-dimensional and comparable. The probability density function (p.d.f.) of the jump is constant throughout the hypersphere, not restricted to the surface. Therefore the p.d.f.s of the destination points of chains X and Y are identical wherever their hyperspheres overlap; see Fig. 2. Overlap occurs when their centres satisfy $|Y - X| < 2r$.

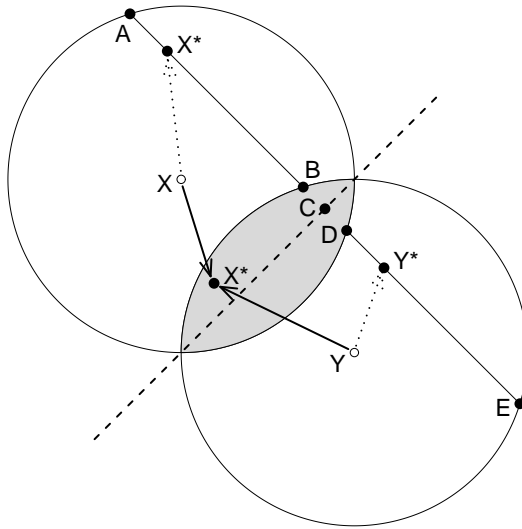


Figure 2: Maximal coupling of destination points, using a solid spherical uniform distribution for the Metropolis–Hastings jumps. Open and solid circles show origin points X and Y , and proposed destination points X^* and Y^* , of the jumps of the two processes. If the independently generated X^* happens to lie in the overlap (shaded) zone with the jump-hypersphere of Y , then Y^* is set to X^* (solid arrows). Otherwise, line AE passes through X^* parallel to XY , and segment AB is translated to segment DE . Then $Y^* - D = X^* - A$, which ensures that Y^* is also outside the overlap zone (dotted arrows)

Algorithm 2 Maximally coupled jumps for Metropolis–Hastings updates

Require: U : negative log-likelihood function; d : number of dimensions.

Ensure: X^*, Y^* are maximally coupled jump destinations of X and Y .

```

1: function JUMP( $X, r, R_{\text{dir}}, R_{\text{mag}}$ )                                ▷  $X$ : jump origin;  $r$ : radius of sphere
2:                                                                 ▷  $R_{\text{dir}}, R_{\text{mag}}$ : random numbers:  $d \times N(0, 1), 1 \times \text{uniform}(0, 1)$ 
3:    $\hat{u} \leftarrow R_{\text{dir}} / |R_{\text{dir}}|$                                 ▷ Jump direction, unit vector from Euclidean norm
4:    $X^* \leftarrow X + r R_{\text{mag}}^{1/d} \hat{u}$                             ▷ Spherical uniform distribution for jump
5:   return  $X^*$                                                     ▷ Destination point for  $X$ , to pass to M–H acceptance test
6: end function
7: function MAXCOUPLE( $X, X^*, Y, r$ )                                ▷  $Y$ : second jump origin
8:    $u_X \leftarrow X^* - X$                                         ▷ Jump vector used for  $X$ 
9:    $L \leftarrow \frac{1}{2}|Y - X|$                                     ▷ Distance to midpoint of  $XY$ 
10:   $v \leftarrow \frac{1}{2}(Y - X)/L$                                 ▷ Unit vector in direction of  $XY$ 
11:   $c \leftarrow Lv + u_X - (u_X \cdot v)v$                         ▷  $X$  to midpoint, plus projection of  $u_X$ 
12:  if  $|Y - X^*| \leq r$  then                                    ▷  $X^*$  is in overlap zone of  $X$  and  $Y$  spheres.
13:     $Y^* \leftarrow X^*$                                         ▷ Couple the two destination points.
14:  else if  $|c| < r$  then                                       ▷ Point C is in overlap zone.
15:     $w \leftarrow -L + \sqrt{L^2 + r^2 - c \cdot c}$                 ▷ Distance from point C to D
16:     $u_Y \leftarrow u_X + 2wv$                                   ▷ Translate to ensure  $Y^* \notin$  overlap zone.
17:     $Y^* \leftarrow Y + u_Y$ 
18:  else                                                         ▷  $C \notin$  overlap zone: no need for translation.
19:     $Y^* \leftarrow Y + u_X$                                        ▷ Use same jump vector for  $Y$  as for  $X$ .
20:  end if
21:  return  $Y^*$                                                 ▷ Destination point for  $Y$ , to pass to M–H acceptance test
22: end function
23: function MHTEST( $X, X^*, R_{\text{MH}}$ )                                ▷ Metropolis–Hastings acceptance test;
    $R_{\text{MH}}$ : uniform(0, 1) random number
24:  if  $R_{\text{MH}} \leq \exp(U(X) - U(X^*))$  then
25:    return  $X^*$                                                 ▷ Accept the jump.
26:  else
27:    return  $X$                                                   ▷ Reject the jump and remain at its origin.
28:  end if
29: end function

```

If the destination point X^* is in the overlap zone, maximal coupling takes the same point to be Y^* , the destination point for Y . Then X and Y coalesce if both of their M–H updates are accepted. If X^* is outside the overlap zone, we must ensure that Y^* is also forced outside, but still within its own hypersphere, in order to maintain the correct jump distribution for Y . A straightforward way to achieve this outcome with the solid spherical uniform jump distribution is shown by the dotted arrows in Fig. 2:

$$u_Y = u_X + 2wv, \quad (9)$$

where $u_X = X^* - X$ and $u_Y = Y^* - Y$ are the jump vectors, $w = |D - C|$ in Fig. 2, and v is the unit vector in the direction XY : $v = \frac{1}{2}(Y - X)/L$ where $L = \frac{1}{2}|Y - X|$. Point C is the midpoint of XY plus the projection of u_X perpendicular to XY in the dashed line in Fig. 2: $C = X + c$ where

$$c = Lv + u_X - (u_X \cdot v)v. \quad (10)$$

Point D, on the surface of the hypersphere of X , then has position vector $D = C + wv$, so w must satisfy $w > 0$ and $|c + wv| = r$. Using the equality $c \cdot v = L$, the equation for w is

$$c^2 + 2Lw + w^2 = r^2. \quad (11)$$

For $|c| < r$ and $w > 0$, this equation is solved by the quadratic formula:

$$w = -L + \sqrt{L^2 + r^2 - c^2}. \quad (12)$$

If $|c| \geq r$, the line AE does not intersect the overlap zone, and we set $u_Y = u_X$ with no adjustment. Our algorithm for maximal coupling is shown in Algorithm 2.

Updates to Algorithm 1 to enable maximal coupling are shown in Algorithms 3 and 4. Jumps for chain 1 are simulated freely without coupling. In the upper triangle of Fig. 1 ($1 < i \leq j$), during block j , chain i is maximally coupled with the one of chains $1, \dots, i-1$ whose pre-jump value is closest to that of chain i . In the lower triangle ($i > j$), if $i > j+1$, chain i is maximally coupled with the one of chains $j+1, \dots, i-1$ whose pre-jump value is closest to that of chain i . Chain $j+1$ is maximally coupled with chain 1 which was computed and stored when processing the upper triangle.

Algorithm 3 Alg. 1 upper triangle (lines 1–19) with maximal coupling

Require: $K \geq 1$, $B \geq 1$, functions $\text{START}(K)$, $\text{RAND}(B)$, $\text{MCMC}(X_0, R, B)$.

Algorithm 2; r set to maximum jump distance.

$M = \text{no. of MCMC iterations per maximal coupling step (divisor of } B)$.

Q^-, Q_1^- to hold pre-jump versions of Q (current column) and Q_1 (row 1).

Q^*, Q_1^* to hold jumped versions of Q and Q_1 , prior to M–H tests.

Q_1^-, Q_1^*, Q_1 have dimensions $[K, B/M]$ to hold all max. coupling results.

RAND now returns four components: R_{MCMC} , R_{dir} , R_{mag} and R_{MH} .

Ensure: Q returns K perfect sample points from MCMC, if $\text{Error} = \text{false}$.

```

1: function MININD( $x, i_1, i_2$ )                                ▷ Index of closest of  $x[i_1 : (i_2 - 1)]$  to  $x[i_2]$ 
2:   return WHICHMIN( $|x[i_2] - x[i_1 : \{i_2 - 1\}]|$ ) +  $i_1 - 1$   ▷ Metric required
3: end function                                             ▷ WHICHMIN: index of minimum, first one if there are ties
4:  $J \leftarrow B/M$                                          ▷ Number of maximal coupling steps per random-number block
5:  $Q \leftarrow \text{START}(K)$ ;  $s \leftarrow \text{SEED}$ ;  $\text{Error} \leftarrow \text{false}$ ;  $a \leftarrow \text{zeros}(K)$ 
6: for  $j \leftarrow 1 : K$  do                                ▷ Loop over blocks (columns) in Fig. 1 upper triangle.
7:   for  $j_1 \leftarrow 1 : J$  do                              ▷ Loop over sub-blocks, each maximally coupled.
8:      $(R_{\text{MCMC}}, R_{\text{dir}}, R_{\text{mag}}, R_{\text{MH}}) \leftarrow \text{RAND}(M)$   ▷ Random numbers  $\times M$ 
9:     for  $i \leftarrow 1 : j$  do                              ▷ Loop over chains (rows of Fig. 1).
10:      if  $a[i] > 0$  then  $Q[i] \leftarrow Q[a[i]]$  else        ▷ Copy if already coalesced.
11:         $Q^-[i] \leftarrow \text{MCMC}(Q[i], M, R_{\text{MCMC}})$         ▷ Run MCMC, pre-jump.
12:        if  $i = 1$  then                                     ▷ Row 1 of Fig. 1: jump freely.
13:           $Q^*[1] \leftarrow \text{JUMP}(Q^-[1], r, R_{\text{dir}}, R_{\text{mag}})$ 
14:           $Q_1 \leftarrow \text{MHTEST}(Q^-[1], Q^*[1], R_{\text{MH}})$ 
15:           $Q_1^-[j, j_1] \leftarrow Q^-[1]$ ;  $Q_1^*[j, j_1] \leftarrow Q^*[1]$ ;  $Q_1[j, j_1] \leftarrow Q[1]$ 
16:        else                                             ▷  $i > 1$ : maximally couple with closest previous row.
17:           $m \leftarrow \text{MININD}(Q^-, 1, i)$ 
18:           $Q^*[i] \leftarrow \text{MAXCOUPLE}(Q^-[m], Q^*[m], Q^-[i], r)$ 
19:           $Q[i] \leftarrow \text{MHTEST}(Q^-[i], Q^*[i], R_{\text{MH}})$ 
20:          if  $Q[i] = Q[m]$  then                             ▷ If coalesced, reset  $m$  to earliest.
21:            if  $a[m] > 0$  then  $m \leftarrow a[m]$  end if
22:             $a[i] \leftarrow m$ ;  $a[a = i] \leftarrow m$         ▷ Mark row  $i$ , remark others.
23:          end if
24:        end if                                           ▷  $a[i] > 0$  records row with which row  $i$  coalesced.
25:      end if
26:    end for                                             ▷  $i$  loop (rows of Fig. 1)
27:  end for                                             ▷  $j_1$  loop over sub-blocks
28: end for                                             ▷  $j$  loop (columns of Fig. 1)

```

We insert one Metropolis–Hastings update, with maximal coupling and a test for coalescence, after every M MCMC iterations in each cell of Fig. 1, where M is a divisor of the block length B

Algorithm 4 Alg. 1 lower triangle (lines 20–34) with maximal coupling

```

1: SEED  $\leftarrow s$  ▷ Restore seed to regenerate previous random number blocks.
2: for  $j \leftarrow 1 : (K - 1)$  do ▷ Loop over blocks in Fig. 1 lower triangle.
3:   Error  $\leftarrow$  Error or  $a[j + 1] \neq j$  ▷ Check final coalescence of chain  $j$ .
4:    $a[j + 1] \leftarrow a[j]$ ;  $a[a = j] \leftarrow j + 1$  ▷ Move coalescence from row  $j$  to  $j + 1$ .
5:   for  $j_1 \leftarrow 1 : J$  do ▷ Loop over sub-blocks.
6:      $Q^-[1] \leftarrow Q_1^-[j, j_1]$ ;  $Q^*[1] \leftarrow Q_1^*[j, j_1]$ ;  $Q[1] \leftarrow Q_1[j, j_1]$ 
7:      $(R_{\text{MCMC}}, R_{\text{dir}}, R_{\text{mag}}, R_{\text{MH}}) \leftarrow \text{RAND}(M)$ 
8:     for  $i \leftarrow (j + 1) : K$  do ▷ Loop over chains.
9:       if  $a[i] > 0$  then  $Q[i] \leftarrow Q[a[i]]$  else ▷ Copy, else run MCMC.
10:         $Q^-[i] \leftarrow \text{MCMC}(Q[i], M, R_{\text{MCMC}})$ 
11:        if  $i = j + 1$  then
12:           $m \leftarrow 1$  ▷ Maximally couple row  $j + 1$  with row 1.
13:        else ▷  $i > j + 1$ : Couple with some row  $\in (j + 1) : (i - 1)$ .
14:           $m \leftarrow \text{MININD}(Q^-, j + 1, i)$  ▷ Choose closest row.
15:        end if
16:         $Q^*[i] \leftarrow \text{MAXCOUPLE}(Q^-[m], Q^*[m], Q^-[i], r)$ 
17:         $Q[i] \leftarrow \text{MHTEST}(Q^-[i], Q^*[i], R_{\text{MH}})$ 
18:        if  $Q[i] = Q[m]$  then ▷  $m \leftarrow$  earliest coalescence row for  $i$ .
19:          if  $a[m] > 1$  then  $m \leftarrow a[m]$  end if
20:           $a[i] \leftarrow m$  ▷ Record coalescence of row  $i$  with earliest.
21:          if  $m > 1$  then  $a[a = i] \leftarrow m$  end if
22:        end if
23:      end if
24:    end for ▷  $i$  loop (rows of Fig. 1)
25:  end for ▷  $j_1$  loop over sub-blocks
26: end for ▷  $j$  loop (columns of Fig. 1)
27: Error  $\leftarrow$  Error or  $a[K] \neq 1$ ;  $Q[1] \leftarrow Q_1[K, J]$  ▷ Check  $K$ ; restore 1.

```

(see lines 12–24 in Algorithm 3 and 11–22 in Algorithm 4). In row 1 of Fig. 1, jump vectors are simulated freely from a solid spherical uniform distribution (Algorithm 3, line 13). In each subsequent row Y , each jump vector is calculated from that of the row X with which it is maximally coupled, according to function `MAXCOUPLE` of Algorithm 2, and there is no extra random input (Algorithm 3, line 18, and Algorithm 4, line 16).

Three versions of the result column-vector Q are recorded through the algorithm: the pre-jump Q^- , the post-jump Q^* , and the final version Q which is equal to either Q^* or Q^- depending on whether the jump is accepted or rejected in the M–H test. These are overwritten in each random-number sub-block of M MCMC iterations (sub-column of Fig. 1). After each sub-block of row 1 of Fig. 1, all three versions are stored for later use in row-vectors Q_1^- , Q_1^* and Q_1 , to save having to recompute them in the lower triangle of Fig. 1.

4 Examples

4.1 Two-state discrete process

We illustrate the methodology of section 2 on a two-state process with the transition matrix shown in Fig. 3 and parameters θ and p satisfying $0 \leq \theta \leq 1$ and $0 < p \leq \frac{1}{2}$.

Starting points for different chains are generated independently, with starts from states 1 and 2 having probabilities (0.5, 0.5). We set $\theta = 1/9$ and $p = 0.1$. Chains converge to the stationary distribution $(1/(1 + \theta), \theta/(1 + \theta)) = (0.9, 0.1)$. The probability p regulates the speed of convergence. The determinant of the transition matrix, which we use below, is $\Delta = 1 - p - \theta p = 8/9$.

$$\begin{pmatrix} 1 - \theta p & \theta p \\ p & 1 - p \end{pmatrix}$$

Figure 3: Markov transition matrix for the two-state process. Parameter values are $\theta = 1/9$ and $p = 0.1$

Chains X and Y are coupled by generating a single sequence S_1, S_2, \dots of independent uniform $(0, 1)$ random numbers, and using S_i to transition from X_{i-1} to X_i and from Y_{i-2} to Y_{i-1} . At time step i , if chain X is in state 1, it moves to state 2 if $S_i > 1 - \theta p$. If it is in state 2, it moves to state 1 if $S_i < p$. Chain Y uses S_{i+1} instead of S_i , and does not use S_1 .

In this example, the probability that $X_i \neq Y_{i-1}$ for $i \geq 1$ can be found analytically. For $i = 1$ it is equal to $\frac{1}{2}$ because Y_0 is sampled independently of X_1 and has a probability of $\frac{1}{2}$ of being equal to X_1 . For $i > 1$, if $X_{i-1} \neq Y_{i-2}$, X_i and Y_{i-1} will coalesce to state 1 if $S_i < p$, and to state 2 if $S_i > 1 - \theta p$. The probability of not coalescing at step i is the determinant Δ calculated above, and the overall probability of failing to coalesce by step i is $\frac{1}{2}\Delta^{i-1}$. This probability takes the values 0.31 at $i = 5$, 0.053 at $i = 20$, 4.3×10^{-6} at $i = 100$, and 1.5×10^{-26} at $i = 500$.

We ran one million independent simulations of X and Y , and applied the methodology of section 2, varying the number of burn-in iterations k from 5 to 120. For $k \geq 106$, X and Y coalesced by iteration $k + 1$ in all simulations, so that no holes remained. Table 1 shows the results for selected values of k . The proportion of holes in the results, as a function of k , is plotted in Fig. 4. At $k = 50$, the number of holes becomes low, although the sample standard deviation of the sum of the weights (which take values ± 1) of the sample points is still substantially greater than the converged value of 0.3. The contrast of columns 2 and 3 in Table 1 shows the effectiveness of the unbiased sampling methodology, which produces an estimated stationary probability of being in state 1 that does not differ significantly from the true stationary value of 0.9.

Table 1: Results from samples of 1 million independent simulations of the two-state process, for various lengths of the burn-in time k . Other columns are the unadjusted proportion of values of X_k in state 1; the adjusted proportion using values X_k, \dots, X_τ and $Y_k, \dots, Y_{\tau-1}$ and weights of ± 1 from the theory in section 2; the proportion of strings of length $\nu > 1$ in the sample; the total number of holes as a proportion of the sample size; and the sample standard deviation of the sum of weights $w_i = \pm 1$ of string elements (from all strings, whether $\nu = 1$ or $\nu > 1$) that are in state 1

k	Unadj.	Adj.	$\nu > 1$	Holes	S.d.
5	0.677422	0.899944	0.2776	2.4944	6.6841
10	0.776260	0.902078	0.1541	1.3817	4.9703
20	0.861767	0.900215	0.0475	0.4204	2.7460
50	0.898480	0.899104	0.0013	0.0118	0.5454
100	0.899329	0.899335	4×10^{-6}	1.2×10^{-5}	0.3010
110	0.899829	0.899829	0	0	0.3002

When perfect simulation is desired, k should be set before the results are generated, to a value high enough to make the probability of any lack of coalescence after k iterations very small. It would break the rules to allow the final simulations to influence the choice of k .

If the methods of section 3.1 were applied to this example, we might decide on a sample-set size of $K = 20$ and a block length of $B = 25$. This would yield an effective value of k of about $KB = 500$, and probabilities of lack of coalescence after $K + 1$ blocks approximately equal to 10^{-26} as calculated above for an individual simulation, and 10^{-20} in a set of 10^6 simulations. It is straightforward to show that raising the transition matrix to the power B is equivalent to

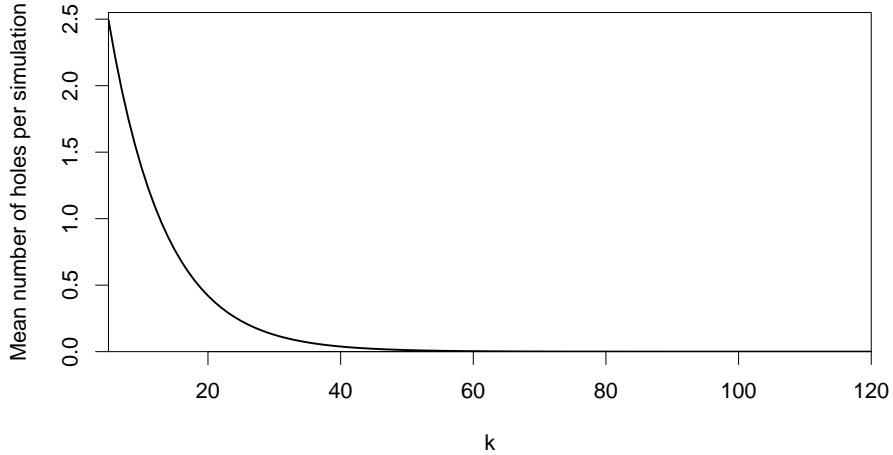


Figure 4: Sample mean number of holes per simulation of the two-state process, as a function of burn-in time k for $k \geq 5$. No holes were observed for $k \geq 106$

replacing p by $p(1 - \Delta^B)/(1 - \Delta)$, and then that if X_{KB} and $X_{(K+1)B}$ both follow the stationary distribution, their correlation coefficient is $\Delta^B = 0.0526$, thereby quantifying analytically the serial correlation of perfect sample points in the same sample set in this example.

4.2 Normal distribution

We applied the methodology of sections 3.1 and 3.2 to the simulation of a standard normal distribution in d dimensions, using random-walk MCMC with an internal Metropolis–Hastings algorithm. We used a $N(0, \sigma I)$ distribution for the jumps in the internal MCMC, with $\sigma = 2/\sqrt{d}$, where I denotes the d -dimensional identity matrix. For the maximal coupling steps, we set the radius of the solid spherical uniform jump distribution to $r = 3$.

The K chains in a sample set were coupled by using the same relative jumps from each chain’s current position. Thereby, two chains stayed the same distance apart unless a jump was accepted for one chain but not the other. Coalescence relied on different chains “piling up” around the effective boundary of the sample space, beyond which, even though the space was of infinite extent, the M–H probability of acceptance was low. M–H rejection could stop one chain from going through the boundary, allowing another chain that had been further from that boundary to catch up with it. The internal M–H acceptance test used the same uniform $(0, 1)$ random variable for all chains.

Due to the relative clumsiness with which coalescence is achieved in random-walk MCMC, maximal coupling, with its own, separate M–H update, was conducted after every MCMC step, i.e., $M = 1$ in section 3.2. Hamiltonian Monte Carlo, although beyond the scope of this paper, performs this and similar tasks more efficiently than random-walk MCMC (see section 5 below).

All chains began by sampling each of the d coordinates independently from a uniform $(-6, 6)$ distribution. As suggested in section 3.1, we set the block length B from an exploratory analysis, to aim for a probability of 0.1 that two chains would fail to coalesce within B iterations. We set the sample-set size K to 20, with the aim of achieving a probability that two chains would fail to coalesce after K blocks (or KB MCMC iterations) of roughly 10^{-20} .

Results are shown in Table 2. The maximum number of blocks for a chain to coalesce to the chain below it in Fig. 1 for any of the runs is 9, and does not approach the sample-set size $K = 20$. Therefore, holes do not occur in any of these results, and the algorithms have generated perfect samples without any need to process holes.

Geometric decay in the number of blocks to coalesce is followed tolerably closely, as indicated by the sample means and maxima. The means remain close to the target of about $1/0.9 \approx 1.11$;

Table 2: Results for simulation of a standard normal distribution by random-walk MCMC, using the methodology of sections 3.1 and 3.2, for various numbers of dimensions d . Other columns are the chosen block length B , the total number of chains or sample points N , the sample mean and maximum number of blocks for each chain to coalesce with the one below it in Fig. 1, and the sample coefficient ρ of the serial correlation which occurs by design between points in the same sample set. The sample-set size K was set to 20 in all simulations, so that the effective number of iterations run for each chain was $20B$ and the number of independent sample sets was $N/20$

d	B	N	Mean	Max.	ρ
1	5	10^7	1.111	9	0.00940
2	10	10^6	1.085	6	0.00263
5	25	10^6	1.107	7	0.00201
10	95	10^6	1.105	7	-0.00055
15	425	10^6	1.114	9	0.00055
20	3500	10^5	1.141	9	-0.00084

i.e., 90% of non-coalesced chains coalesce within one further block. Detailed inspection of the numbers for $d = 15$ and 20, however, indicated some departure in this desideratum. For $d = 15$, the numbers of chains requiring 3, 4, 5, 6, 7, 8 and 9 blocks to coalesce were 11341, 2088, 424, 98, 17, 8 and 1, which imply that, beyond 3 blocks, only 80% or fewer were coalescing in one further block. For $d = 20$, the numbers were 1532, 316, 56, 6, 3, 1 and 1, which imply much the same. The required block length B to achieve these rates of coalescence also increased with d at a rate greater than exponential. To us these problems indicate that $d = 20$ is close to the limit at which random-walk MCMC can function effectively. As a comparison, for $d = 5$ the numbers of chains requiring 2, 3, 4, 5, 6 and 7 blocks were 95896, 5082, 397, 36, 1 and 1, showing that, in all but the final category, more than 90% of non-coalesced chains coalesced in one further block.

Serial correlation of chains in the same sample set (final column of Table 2) is a design feature of the algorithm in section 3.1, introduced to enable more perfect samples to be generated for very little extra computation. The sample correlation was statistically significant (about 30 standard deviations) at $d = 1$, for which the block length B was only 5. It was marginally significant (about 4 s.d.) at $d = 2$ and $d = 5$. It was not statistically significant for $d \geq 10$.

5 Discussion

We have demonstrated how unbiased simulation can be converted into perfect simulation with an arbitrarily small probability that complications will arise, which we have set to roughly 10^{-20} . To handle cases where complications nevertheless do arise, we have developed the theory of holes in section 2.3, and shown the effect that holes have on the standard deviation of a parameter estimator in one simple example (Table 1). In our other example (section 4.2 and Table 2), the number of blocks for neighbouring chains to coalesce and enable perfect simulation was less than the limiting number of blocks $K = 20$ by a comfortable margin in all of our runs. Hence, holes did not arise.

The nonzero probability of occurrence of holes in a sample appeases the theoretical limitation that, unless a property such as monotonicity is available, a general, strictly perfect simulation scheme must visit every possible state of a process (see section 1). In that sense, the possibility of holes is to be welcomed rather than cursed. In practice, due to their low theoretical probability, the occurrence of holes in a sample would suggest that some feature of the likelihood function has been overlooked during theoretical examination. This in turn may highlight the need to redesign the internal MCMC algorithm that is called by our algorithms 1, 3 and 4.

Our algorithms in section 3 introduce a trade-off between computational effort and serial correlation within a sample set. Despite this trade-off, all of the generated points remain perfect

sample points, and different sample sets, each containing about 20 points, remain completely independent. We view serial correlation of perfect samples as a smaller price to pay than the disagreeable behaviour that can arise in unbiased sampling with low to moderate burn-in times k (see sections 1, 2.1 and 4.1). The computation per final sample point in our algorithm is about the same as that of unbiased simulation with k set to our block length B . Unbiased simulation runs the same block of random numbers on two chains, to generate one point, with relatively little extra computation when chains don't coalesce after one block. Our algorithm runs a minimum of $2K$ blocks of random numbers (marked red in Fig. 1) to generate K points, again with a little extra when coalescence takes more than one block.

It can be argued that perfect simulation is beyond price, given the very high uncertainty around MCMC convergence that exists with current tests (see section 1). We note that any failure of our methodology to produce perfect sample points is immediately clear from the associated lack of coalescence after K blocks. Such clarity is not a feature of current MCMC convergence tests.

We conclude with a discussion of Hamiltonian Monte Carlo (HMC), a technique whose combination with the developments in this paper offers exceptional performance and almost complete achievement of the objective of replacing uncertain MCMC convergence tests by perfect simulation, for continuous processes with differentiable likelihoods. HMC was developed by Duane et al. (1987) and is discussed in detail by Neal (1993, 1995, 2011). Originally known as Hybrid Monte Carlo, HMC alternates random and deterministic phases, thereby exploring the sample space much more effectively than older MCMC methods that are based on random walks (see Neal, 2011, sec. 5.3.3).

Originally, each instance of the deterministic phase, termed a “trajectory”, ran for a fixed length of notional time. Advanced HMC algorithms, of which the first was the No-U-Turn Sampler (NUTS) (Hoffman and Gelman, 2014), adjust the trajectory length according to the curvature of the likelihood function. Leigh and Northrop (2022) developed one completely new advanced HMC algorithm, and another algorithm that is an improved version of NUTS.

Unbiased sampling was applied to original HMC by Heng and Jacob (2019), using the observation that coupled trajectories with different starting points are drawn closer together when their trajectory time-span is equal to one quarter of the period of a complete orbit. Leigh and Northrop (2022) applied unbiased sampling to the advanced HMC algorithms by coupling the selection of a trajectory's destination point, as a random proportion of the traversal time from the back endpoint to the forward one. They achieved perfect sampling from a wide range of distributions, including long-tailed t -distributions and Gaussian mixtures, up to 100 dimensions. Bou-Rabee et al. (2020) applied coupling to HMC and included varying the initial velocities of coupled trajectories, whereas other couplings set the initial velocities equal.

Unlike random-walk MCMC, we know of no limit to the number of dimensions in problems to which HMC can be applied.

Declarations

No funding was received for conducting this study.

The authors have no relevant financial or non-financial interests to disclose.

The authors have no conflicts of interest to declare that are relevant to the content of this article.

All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

The authors have no financial or proprietary interests in any material discussed in this article.

Code for the examples in section 4 is available at <https://github.com/George-Leigh/PerfectSimulation>, written in R (R Core Team, 2023).

Authors' note

This paper is a restructured and expanded version of some of the material contained in Leigh and Northrop (2022). This paper contains new theory for maximal coupling, which we believe to be superior to the congruential coupling (called a “rounding step”) of the original paper. The example applications of the methodology are also new.

References

- Aldous, D. and P. Diaconis (1987). Strong uniform times and finite random walks. *Adv. Appl. Math.* 8(1), 69–97.
- Bellman, R. (1955). Review of Transactions of the Symposium on Computing, Mechanics, Statistics, and Partial Differential Equations, Vol. II. *J. Amer. Stat. Assoc.* 50(272), 1357–1359.
- Bou-Rabee, N., A. Eberle, and R. Zimmer (2020). Coupling and convergence for Hamiltonian Monte Carlo. *Ann. Appl. Probab.* 30(3), 1209–1250. Publisher: Institute of Mathematical Statistics.
- Brooks, S., A. Gelman, G. Jones, and X.-L. Meng (2011). *Handbook of Markov Chain Monte Carlo*. Chapman and Hall.
- Craiu, R. V. and X.-L. Meng (2011). Perfection within reach: Exact MCMC sampling. In S. Brooks, A. Gelman, G. Jones, and X.-L. Meng (Eds.), *Handbook of Markov Chain Monte Carlo*, pp. 199–226. Chapman and Hall.
- Duane, S., A. D. Kennedy, B. J. Pendleton, and D. Roweth (1987). Hybrid Monte Carlo. *Phys. Lett. B* 195(2), 216–222.
- Fill, J. A. (1998). An interruptible algorithm for perfect sampling via Markov chains. *Ann. Appl. Probab.* 8(1), 131–162.
- Gelman, A. and K. Shirley (2011). Inference and monitoring convergence. In S. Brooks, A. Gelman, G. Jones, and X.-L. Meng (Eds.), *Handbook of Markov Chain Monte Carlo*, pp. 163–174. Chapman and Hall.
- Glynn, P. W. and C.-H. Rhee (2014). Exact estimation for Markov chain equilibrium expectations. *J. Appl. Probab.* 51(A), 377–389.
- Griffeath, D. (1975). A maximal coupling for Markov chains. *Z. Wahrscheinlichkeitstheorie Verwandte Geb.* 31(2), 95–106.
- Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* 57(1), 97–109.
- Heng, J. and P. E. Jacob (2019). Unbiased Hamiltonian Monte Carlo with couplings. *Biometrika* 106(2), 287–302.
- Hoffman, M. D. and A. Gelman (2014). The No-U-Turn Sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.* 15, 1593–1623.
- Huber, M. L. (2016). *Perfect Simulation*. Boca Raton, FL: CRC Press.
- Jacob, P. E., J. O’Leary, and Y. F. Atchadé (2020). Unbiased Markov chain Monte Carlo methods with couplings. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* 82(3), 543–600.

- Johnson, V. E. (1998). A coupling-regeneration scheme for diagnosing convergence in Markov chain Monte Carlo algorithms. *J. Am. Stat. Assoc.* *93*(441), 238–248. Publisher: Taylor & Francis _eprint: <https://www.tandfonline.com/doi/pdf/10.1080/01621459.1998.10474105>.
- Leigh, G. M. and A. R. Northrop (2022). Design of Hamiltonian Monte Carlo for perfect simulation of general continuous distributions. Technical Report arXiv:2212.12140, arXiv.
- Liu, J. S. (2004). *Monte Carlo Strategies in Scientific Computing*. Springer Series in Statistics. New York, NY: Springer.
- Lovász, L. and P. Winkler (1995). Exact mixing in an unknown Markov chain. *Electron. J. Comb.* *2*, article R15.
- Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys.* *21*, 1087–1092.
- Mira, A., J. Møller, and G. O. Roberts (2001). Perfect slice samplers. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* *63*(3), 593–606.
- Murdoch, D. J. and P. J. Green (1998). Exact sampling from a continuous state space. *Scand. J. Stat.* *25*(3), 483–502.
- Neal, R. M. (1993). Probabilistic Inference Using Markov Chain Monte Carlo Methods. Research report, Department of Computer Science, University of Toronto.
- Neal, R. M. (1995). *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, Graduate Department of Computer Science.
- Neal, R. M. (2011). MCMC using Hamiltonian dynamics. In S. Brooks, A. Gelman, G. Jones, and X.-L. Meng (Eds.), *Handbook of Markov Chain Monte Carlo*, pp. 113–162. Chapman and Hall.
- Propp, J. G. and D. B. Wilson (1996). Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Struct. Algorithms* *9*(1–2), 223–252.
- R Core Team (2023). *R: A language and environment for statistical computing*. Vienna: R Foundation for Statistical Computing.
- Roy, V. (2020, March). Convergence diagnostics for Markov chain Monte Carlo. *Annu. Rev. Stat. Appl.* *7*, 387–412.
- van den Boom, W., A. Jasra, M. De Iorio, A. Beskos, and J. G. Eriksson (2022). Unbiased approximation of posteriors via coupled particle Markov chain Monte Carlo. *Stat. Comput.* *32*(3), 36.
- Wilson, D. B. (2000). How to couple from the past using a read-once source of randomness. *Random Struct. Algorithms* *16*(1), 85–113.