Using Genetic Operators to Speed up Markov Chain Monte Carlo Integration

Tuomas J. Lukka
email: lukka@iki.fi

Janne V. Kujala
email: jvk@iki.fi

Dept. of Mathematical Information Technology
P.O.Box 35 (Agora)
FIN-40351 Jyväskylä
Finland

Abstract

A way of applying genetic crossing-over operators to Markov Chain Monte Carlo (MCMC) integration without introducing bias is presented. The crux of the method is to either accept or reject together *both* of the offsprings from reversible crossing-over operations. This method can converge faster than traditional MCMC methods in several problems; numerical computations for two model distributions are shown. The relationship between the presented method, Swendsen-Wang -type algorithms, and Metropolis-Coupled MCMC is discussed.

1 Introduction

Metropolis-Hastings MCMC[1, 2] is a popular numerical method for integrating over a multidimensional probability distribution $p(x)$. It is used extensively in Bayesian computation and statistical mechanics (cf. [3, 4]). We propose a method of speeding up the convergence of the basic method by using genetic crossing-over, an idea from genetic algorithms[5, 6].

The Metropolis algorithm simulates a process where at each time step, a new, proposed point x' is drawn from some probability distribution $q(x, x')$ depending on the current point x. In effect the proposal distribution defines a "neighborhood structure", i.e. what is considered to be close to the current point. The new point is accepted with the probability

$$\alpha(x, x') = \min\left\{1, \frac{p(x')}{p(x)}\right\}. \tag{1}$$

If the new point is not accepted, the old point is kept for the next time step. It can be shown that under certain simple conditions, the distribution of x over

time will converge to $p(x)$ [4, 7]. The most important condition is stationarity of the distribution $p(x)$, so that once reached, the correct distribution is maintained for ever. This follows from the local detailed balance condition, which holds if the probability distribution for generating the new points is symmetric:

$$q(x, x') = q(x', x). \tag{2}$$

Another requirement for convergence is that the system is irreducible, so that it has a possibility to eventually get from any point x to any other point x', possibly through several steps. This can be achieved through having q be positive everywhere, such as a Gaussian, or by knowing beforehand that the neighborhood structure determined by q connects all the modes of p through paths for which p is never zero.

The problem with MCMC calculations is that they can be excruciatingly slow for multidimensional systems and that it is often difficult to estimate convergence. This problem is emphasized if parts of the state space are connected only through paths with low probability. For example, consider a distribution consisting of two Gaussian peaks. As the distance between the peaks grows the convergence slows down rapidly since the system has to random-walk several times between the two modes before the resulting distribution approaches stationarity. This slowdown becomes even more pronounced as the number of dimensions and modes grows. The modes, however, are typically not randomly distributed but there is some structure. Most multidimensional real-world problems are simpler than general functions of the same dimensionality and are often approximately decomposable to several few-dimensional functions (the building block hypothesis, cf. [6, p. 45]). Unfortunately, there is generally no direct way to take advantage of that approximate structure, especially since the structure is usually not known.

Auxiliary variable methods such as the Swendsen-Wang -type algorithms[8, 9] attempt to get hold of the structure of the target distribution by introducing auxiliary variables such that conditional on the auxiliary state, the target distribution is decomposable in some specific way. The Metropolis-coupled MCMC and simulated tempering algorithms (cf. [10]) take a different approach by simulating gradually tempered or damped versions of the target distribution creating new, known structure to the model.

In optimization the problem can be attacked by genetic algorithms[5, 6] (GA) which are often able to implicitly exploit the near-decomposability of the objective function and converge to the optimum faster that way. Genetic algorithms work by evolving a population of points, denoted by $\{x_i\}$. At each round (generation), a new set of points is generated by randomly applying genetic operators to the previous set and the fittest offsprings are selected. The most important genetic operation is the crossing-over operation, in which a new point is generated by taking one part of the description of one point and another part of another point and combining them: for example, crossing (`D,E,F`) and (`J,K,L`) could result in (`D,K,L`). While the crossing-over operator enables global search, there is another operator, mutation, which is responsible for local variation and

is in effect similar to the proposal distribution $q(x, x')$ of the MCMC method. There are an enormous number of variations of genetic algorithms but the basic principle remains the same: the genetic recombination gives a different kind of a neighborhood structure that has proven to be advantageous for optimization.

The point of this article is to try to bridge the performance gap between optimization and MCMC by enabling the use of genetic operators in MCMC integration. The reason that genetic operations are not used for MCMC is that they are generally not reversible: if the points (`D,E,F`) and (`J,K,L`) as above are used to produce the point (`D,K,L`), the process cannot be reversed after natural selection has been applied to the offspring. Using non-reversible operators in an MCMC calculation will generally introduce bias into the calculated distribution.

The following sections introduce a way of using genetic operators in MCMC without introducing bias, consider the justification for such a method, prove its convergence, discuss its relationship with the Swendsen-Wang[8] and Metropolis-coupled MCMC[10] algorithms, and show numerical examples where using genetic operators speeds up the convergence of an MCMC integration considerably.

2 Genetic Operator MCMC (GO-MCMC)

Figure 1 gives the algorithm for using genetic operators in MCMC without introducing bias in the resulting sample.

The algorithm is based on the usual Metropolis-Hastings formulation, but on the power distribution

$$\mathbf{p}(x_1, \ldots, x_N) = \prod_i p(x_i) \tag{3}$$

instead of the target distribution. The power distribution is defined in a power space whose points are populations of states $\mathbf{x} = \{x_i\}$. A proposal distribution for the power space is of the form $\mathbf{q}(\mathbf{x}, \mathbf{x}')$, i.e., the proposals are from a population $\mathbf{x}$ to another population $\mathbf{x}'$. In this formulation, genetic crossing-over operators can be expressed as reversible proposals to recombine two individuals, x_i and x_j (see Figure 2):

$$(x_i', x_j') = T(x_i, x_j), \tag{4}$$

where the crossing-over operator T is its own inverse:

$$(x_i, x_j) = T(x_i', x_j'). \tag{5}$$

The new values x_i' and x_j' are either accepted or rejected but they must be accepted or rejected *together* as parts of $\mathbf{x}'$, depending simply on the probabilities of $\mathbf{x}$ and $\mathbf{x}'$. The acceptance probability then simplifies to

$$\alpha(\mathbf{x}, \mathbf{x}') = \min\left\{1, \frac{\mathbf{p}(\mathbf{x}')}{\mathbf{p}(\mathbf{x})}\right\} = \min\left\{1, \frac{p(x_i')p(x_j')}{p(x_i)p(x_j)}\right\}. \tag{6}$$

Note that since the old points are discarded from the population, the use of genetic operators will not cause an increase of the better genes — rather, it

0. Select recombination rate p_c and choose an initial population $x_1, \ldots, x_N$.

1. Choose a random number $0 \leq r < 1$.

2. If $r < p_c$, divide the population into random pairs and for each pair (x_i, x_j),

   2a. Apply a randomly chosen crossing-over operator T_m to produce $(x'_i, x'_j) = T_m(x_i, x_j)$.

   2b. Replace (x_i, x_j) by (x'_i, x'_j) with the probability min $\left\{1, \frac{p(x'_i)p(x'_j)}{p(x_i)p(x_j)}\right\}$.

3. If $r \geq p_c$, for each individual x_i, do an ordinary MCMC update, e.g.:

   3a. Draw x'_i from the proposal distribution $q(x_i, x'_i)$.

   3b. Replace x_i by x'_i with the probability min $\{1, p(x'_i)/p(x_i)\}$.

4. Output the population $x_1, \ldots, x_N$.

5. Goto 1.

Figure 1: One way to apply genetic operators to MCMC simulation.

$$
\begin{array}{lcccc}
 & & x_i & & x_j \\
\mathbf{x} & \{(\texttt{A},\texttt{B},\texttt{C}), & (\texttt{D},\texttt{E},\texttt{F}), & (\texttt{G},\texttt{H},\texttt{I}), & (\texttt{J},\texttt{K},\texttt{L})\} \\
 & & & \searrow \quad \swarrow & \\
 & & & T & \\
 & & & \swarrow \quad \searrow & \\
 & & x'_i & & x'_j \\
\mathbf{x}' & \{(\texttt{A},\texttt{B},\texttt{C}), & (\texttt{D},\texttt{K},\texttt{L}), & (\texttt{G},\texttt{H},\texttt{I}), & (\texttt{J},\texttt{E},\texttt{F})\}
\end{array}
$$

Figure 2: An example of a reversible crossing-over proposal in a population of four individuals.

simply allows the system to faster explore parts of the state space. The crossing-over operation can exchange approximately independent components between the individuals with a high probability, thus reducing spurious correlations between variables.

The crossing-over operation alone is not sufficient to guarantee convergence. To assure convergence, we additionally use standard MCMC steps. As the target distributions for the individual states x_i are independent, we can efficiently update them separately using the ordinary proposal distribution $q(x_i, x_i')$, accepting or rejecting the new x_i' separately as well with the ordinary acceptance probability (1). In terms of the power space, these proposals are generated by the distributions $\mathbf{q}_i(\mathbf{x}, \mathbf{x}') = q(x_i, x_i') \prod_{j \neq i} \delta_{x_j}(x_j')$, where δ_x is the Kronecker or Dirac delta function. All x_i or just a randomly chosen subset can be updated in this way at each time step, corresponding to mutations in GA. These mutation steps are then combined with the above crossing-over steps to form an ergodic hybrid sampler.

The next section gives a proof of convergence for the algorithm. In short, the reason this algorithm works correctly within the MCMC framework without biasing the estimates is that both of the above operations are proper and reversible MCMC steps in terms of the whole population and so, the probability distribution will eventually converge to $\mathbf{p}$, which is a product of the independent distributions $p(x_i)$. This in turns means that the distribution for each x_i will converge to $p(x_i)$, and so we can approximate the target distribution $p(x)$ by combining the samples.

Note that real implementations use a set $\{T_m\}_m$ of crossing over-operators from which one is randomly chosen for each operation. The different operators correspond to, for example, different points of where to cut the chromosome.

3 Mathematics and asymptotic convergence

In this section, we attempt to illustrate why the present algorithm can significantly speed up the convergence of MCMC simulations. We then show that the present algorithm converges and at least does not slow down the asymptotic convergence of the underlying kernel.

It is hard to come up with a rigid proof of why genetic algorithms (and thereby algorithms like the present one) are useful since they excel in treating "real-world" functions, for which there is not yet an adequate theoretical description. Some attempts have been made with schema theory (cf. [6, 11]), where the key assumption is that the descriptions of good individuals can be made up from many short "building blocks". Mathematically, this can be viewed as assuming that the distribution is approximately the product of lower-dimensional, multimodal distributions. The basic idea of using genetic recombination instead of simple mutations is that good building blocks "discovered" in different individuals can be brought together to form new, good individuals.

As a toy example of a situation where using genetic operators will help,

consider the exactly decomposable distribution

$$p(x) \propto \prod_l \Big(k(x^{(l)} - 1) + k(x^{(l)} + 1) \Big) \tag{7}$$

where $k(t)$ is a one-dimensional function with a sharp peak at zero, e.g. $\exp(-15t^2)$, and $x^{(l)}$ is a scalar, one component of the vector x. This function has peaks at the corners of an n-dimensional hypercube. Due to the sharpness of the peaks it will take a number of iterations for MCMC to get from one mode to the other. When the number of dimensions grows, the number of local maxima grows exponentially. With traditional MCMC, the random walk will take more and more time to reach enough different maxima, even when started at several points at once, like the GO-MCMC. With genetic crossing-over, however, the mixing between the local maxima will be faster and the simulation will converge more rapidly.

Simple examples show that GO-MCMC can improve convergence over the traditonal algorithm in specific cases. In the following, we consider what can be said about the algorithm in general. In particular, we show that geometric and uniform limits for asymptotic convergence are generally preserved.

3.1 Definitions

Formally, we consider Markov chains given by a transition kernel $P(x, dx')$ on an arbitrary state space E with a countably generated σ-algebra $\mathcal{E}$. A thorough development of general state space Markov chain theory can be found in Meyn & Tweedie[12].

A transition kernel P is called *reversible* with respect to a measure π, if the detailed balance relation

$$\pi(dx)P(x, dx') = \pi(dx')P(x', dx) \tag{8}$$

is satisfied (cf. Tierney[13]). This condition implies that π is an invariant distribution for P, that is

$$\pi(A) = (\pi P)(A) := \int \pi(dx)P(x, A) \tag{9}$$

for all $A \in \mathcal{E}$. A transition kernel P is called *irreducible* if there is a non-zero measure φ such that for each $x \in E$ and $A \in \mathcal{E}$ with $\varphi(A) > 0$ there is an integer n such that $P^n(x, A) > 0$. An invariant measure π of an irreducible kernel is a maximal irreducibility measure in the sense that any φ is absolutely continuous with respect to π.

It is assumed that an ordinary MCMC transition kernel $P(x, dx')$ is given with the invariant distribution $\pi(dx)$. We then expand the state space to E^N and use

$$\pi^{\otimes N}(d\mathbf{x}) = \pi(dx_1) \cdots \pi(dx_N) \tag{10}$$

as a target distribution on the power σ-algebra $\mathcal{E}^{\otimes N}$. The ordinary MCMC transition kernel $P(x_i, dx_i')$ is used for each individual x_i in the population

$\mathbf{x} = (x_1, \ldots, x_N) \in E^N$. In terms of the power space, the ordinary kernel corresponds to

$$P_i(\mathbf{x}, d\mathbf{x}') := P(x_i, dx_i') \prod_{j \neq i} \delta_{x_j}(dx_j') \tag{11}$$

when applied to x_i. Here δ_x is the Dirac measure. Formally we use the product $P_1 \cdots P_N$ of the component kernels for mutation steps. Note that if the original kernel is reversible, so is this product.

We view the set $\{T_m\}_m$ of genetic operators as one-to-one mappings on the power space E^N, operating on whole populations instead of on a pair of individuals. Generally, the set $\{T_m\}_m$ then contains separate copies of the crossing-over operations for each combination of two individuals. Denote by $P_{Tm}(\mathbf{x}, d\mathbf{x}')$ the transition kernel corresponding to proposing a crossing-over $\mathbf{x} \mapsto T_m(\mathbf{x})$ and either accepting or rejecting it. Suppose that the population is divided into pairs and crossing-over operations are applied to each pair as in the example algorithm given in Figure 1. This kind of a transition can be represented as a product

$$P_{TM} := \prod_{m \in M} P_{Tm} \tag{12}$$

of the transition kernels corresponding to the single crossings. It is essential that each T_m indexed by $m \in M$ operates only on individuals that are fixed by all other T_m so that the kernels P_{Tm} commute and the product is well-defined. It also follows that P_{TM} inherits the reversibility of the component kernels. Now, let c_M be the probability of applying a particular set of crossings indexed by M. The kernel used for recombination steps is then some convex combination

$$P_T := \sum c_M P_{TM} \tag{13}$$

of the kernels P_{TM}, each of which is selected with a possibly zero probability c_M.

The actual transition kernel used by the GO-MCMC algorithm is a mixture $p_c P_T + (1 - p_c) P_1 \cdots P_N$ of the recombination and mutation kernels. In the following we first define the kernel P_{Tm} corresponding to a deterministic proposal mapping T_m so that the kernel is reversible with respect to the target distribution. Then we consider asymptotic convergence. For simplicity and generality, we will denote the power space again simply by E.

The formulation of the recombination kernel is similar to the treatment of the general case of the reversible jump kernel in Green[14]. The case of deterministic proposals is also sketched in Tierney[13]. Some results on combining different kernels can be found in Tierney[7].

3.2 Deterministic proposals in general state space

A deterministic proposal is merely one special case of the proposal distribution. The Metropolis-Hastings method, however, is usually formulated assuming that the proposal and target distributions are defined as densities with respect to a

common dominating measure and as such, does not apply for our case. We need to consider the deterministic proposals in a more general setting.

Let $T : E \to E$, $T^{-1} = T$, be a measurable, deterministic proposal mapping on an arbitrary state space $(E, \mathcal{E})$. Then the corresponding transition kernel is

$$P_T(x, dx') = \alpha(x)\delta_{T(x)}(dx') + (1 - \alpha(x))\delta_x(dx'), \tag{14}$$

where $\alpha : E \to [0, 1]$, $\alpha(x) := \alpha(x, T(x))$, is the acceptance probability of the proposed move from x to $T(x)$. The first term corresponds to accepted transitions and the second term accounts for rejected proposals. Next, we consider under which conditions the transition kernel P_T above will be reversible (cf. Tierney[13]).

Proposition 3.1 *For $P_T(x, dx')$ given by (14) the following are equivalent:*

(i) *$P_T(x, dx')$ is reversible with respect to $\pi(dx)$.*

(ii) *$\alpha(x)\pi(dx) = \alpha(T(x))\pi(dT(x))$.*

(iii) *For π-almost all x,*

$$\alpha(x) = \frac{\pi_a(dT(x))}{\pi(dx)}\alpha(T(x)),$$

where $\pi_a(dT(x))$ is the absolutely continuous part of the Lebesgue decomposition $\pi(dT(x)) = \pi_a(dT(x)) + \pi_c(dT(x))$ with respect to $\pi(dx)$.

(iv) *For a set R given by the Lebesgue decomposition such that $\pi(R) = 1$, $\pi_c(T(R)) = 0$, and $\pi(dT(x)) = \pi_a(dT(x))$ restricted in R,*

$$\alpha(x) = \begin{cases} \dfrac{\pi(dT(x))}{\pi(dx)}\alpha(T(x)), & \pi\text{-a.e. in } R \cap T(R) \\ 0, & \pi\text{-a.e. in } R \setminus T(R). \end{cases}$$

Furthermore, these conditions only determine α up to π-equivalence.

Proof. (i) $\Longleftrightarrow$ (ii): Substituting (14) in the reversibility condition (8) yields

$$\begin{aligned} &\pi(dx)[\alpha(x)\delta_{T(x)}(dx') + (1 - \alpha(x))\delta_x(dx')] = \\ &\pi(dx')[\alpha(x')\delta_{T(x')}(dx) + (1 - \alpha(x'))\delta_x(dx)]. \end{aligned} \tag{15}$$

Because the first terms are concentrated at $x' = T(x)$ on both sides and the second terms are equal, reversibility is equivalent to

$$\pi(dx)\alpha(x) = \pi(dT(x))\alpha(T(x)). \tag{16}$$

It is clear from this equation that α is only determined up to π-equivalence.

(ii) $\Longrightarrow$ (iv): Because $\pi(dT(x))$ is absolutely continuous with respect to $\pi(dx)$ in R, the Radon-Nikodým theorem states that for π-almost all $x \in R$,

$$\alpha(x) = \frac{\pi(dT(x))}{\pi(dx)}\alpha(T(x)). \tag{17}$$

Furthermore, $\pi(T(R \setminus T(R))) = 0$ and so for $x \in R \setminus T(R)$ the right side can equivalently be written as $(0/\pi(dx))\alpha(T(x)) = 0$.

(iv) $\Longrightarrow$ (iii): Equation (17) above is equivalent to (iv) and implies (iii) because $\pi(R) = 1$ and $\pi_a(dT(x)) = \pi(dT(x))$ in R.

(iii) $\Longrightarrow$ (ii): For $x \in R$, $\pi_a(dT(x)) = \pi(dT(x))$ and we can write $\alpha(x)\pi(dx) = \alpha(T(x))\pi(dT(x))$ for R. By symmetry, (16) also holds for $T(R)$. The rest of the state space is null for both sides and so, (16) holds everywhere. □

Proposition 3.2 *A deterministic proposal kernel given by* (14) *using the Metropolis acceptance probability*

$$\alpha(x) = \min\left\{1, \frac{\pi_a(dT(x))}{\pi(dx)}\right\} \tag{18}$$

satisfies detailed balance.

Proof. Detailed balance holds by Proposition 3.1(iv): Restricted to $R \cap T(R)$, we have $\pi_a(dx) = \pi(dx)$, $\pi_a(dT(x)) = \pi(dT(x))$, and so

$$\begin{aligned}\frac{\pi(dT(x))}{\pi(dx)}\alpha(T(x)) &= \min\left\{\frac{\pi(dT(x))}{\pi(dx)}, \frac{\pi(dT(x))}{\pi(dx)}\frac{\pi(dx)}{\pi(dT(x))}\right\} \\ &= \min\left\{\frac{\pi(dT(x))}{\pi(dx)}, 1\right\} = \alpha(x)\end{aligned} \tag{19}$$

for π-almost all $x \in R \cap T(R)$. Because $\pi(T(R \setminus T(R))) = 0$, we have $\alpha(x) = \min\{1, \pi(dT(x))/\pi(dx)\} = 0$ for π-almost all $x \in R \setminus T(R)$. □

To illustrate the acceptance probability expression, assume π has a density $p(x)$ with respect to a measure, say μ. If T is measure-preserving with respect to μ, the Radon-Nikodým derivative in (18) will be just $p(T(x))/p(x)$. In case of a discrete state space, μ can be taken as the counting measure and T will obviously preserve count. If T is differentiable and π is given with respect to the Lebesgue measure, we can take $p(T(x))/p(x) \cdot |\partial T(x)/\partial x|$, where $|\cdots|$ is the absolute value of the determinant of the Jacobian matrix. This last case is in fact the same as a deterministic *reversible jump* transition (cf. Green[14]) within a single subspace.

3.3 Rates of convergence

There are differing definitions of convergence limits. Our definitions coincide with Roberts & Rosenthal[15] and Tierney[7]. We denote the *total variation norm* of a signed measure μ by $\|\mu\| := |\mu|(E) = \sup_A \mu(A) - \inf_A \mu(A)$.

Definition 3.3 A transition kernel is said to converge from a starting point x if

$$\lim_{n\to\infty} \|P^n(x,\cdot) - \pi(\cdot)\| = 0.$$

The transition kernel P is *ergodic* if it converges for all $x \in E$.

Definition 3.4 A transition kernel P is called π-almost everywhere *geometrically ergodic* if there exist a π-almost everywhere finite function $M(x)$ and a constant $r < 1$ such that

$$\|P^n(x,\cdot) - \pi(\cdot)\| \leq M(x)r^n$$

for all $x \in E$. If $M(x)$ is constant, the transition kernel P is called *uniformly ergodic.*

Proposition 3.5 *If P is irreducible and aperiodic and $\pi P = \pi P_T = \pi$, the mixture $p_c P_T + (1-p_c)P$, $0 \leq p_c < 1$, is irreducible, aperiodic, and converges π-almost everywhere.*

Proof. Clearly the mixture $p_c P_T + (1-p_c)P$ inherits aperiodicity, irreducibility, and the invariant distribution π which, by Theorem 1 of Tierney[7], implies π-a.e. convergence. □

Proposition 3.6 *Suppose that P_T is given by* (14) *and P is ergodic with $\pi = \pi P = \pi P_T$. Then the mixture $p_c P_T + (1-p_c)P$, $0 \leq p_c < 1$, converges π-a.e. and can be made ergodic with a specific choice of the acceptance probability function $\alpha(x)$. Furthermore, the same statement holds about $P_T := P_{TM}$ given by* (12).

Proof. As an ergodic kernel is always aperiodic and irreducible, π-a.e. convergence follows from the previous proposition. Now, let

$$D := \{x \in E : p_c P_T + (1-p_c)P \text{ does not converge from } x\}.$$

By changing the definition of the acceptance probability in the π-null set D so that $\alpha(x) = 0$ for all $x \in D$, the chain can be made to converge started from D. This does not affect the convergence from points outside D, because such a chain reaches D with probability 0 — otherwise the chain would not have converged in the first place.

The same argument holds for a product $P_T = P_{T1} \cdots P_{Tk}$ if the acceptance probability is modified for each of the component kernels. Then, as above, only P will be effective for all $x \in D$. □

Proposition 3.7 *Suppose that $\pi = \pi P = \pi P_T$ and P is uniformly ergodic. Then the mixture $p_c P_T + (1-p_c)P$, $0 \leq p_c < 1$, is uniformly ergodic.*

Proof. Proposition 3 in Tierney[7]. □

Definition 3.8 The $L^2(\pi)$ Hilbert space of signed measures is defined as

$$L^2(\pi) := \{\, \mu : \mathcal{E} \to [-\infty, \infty] \mid \|\mu\|_{L^2(\pi)} < \infty \,\},$$

where the norm is given by

$$\|\mu\|^2_{L^2(\pi)} = \int \left| \frac{\mu(dx)}{\pi(dx)} \right|^2 \pi(dx)$$

when μ is absultely continuous with respect to π and $\|\mu\|_{L^2(\pi)} = \infty$ otherwise.

Proposition 3.9 *Any Markov transition kernel P with an invariant distribution $\pi = \pi P$ is a weak $L^2(\pi)$-contraction, that is, $\|\mu P\|_{L^2(\pi)} \leq \|\mu\|_{L^2(\pi)}$ for all $\mu \in L^2(\pi)$.*

Proof. This follows from the corresponding general result for all $L^p(\pi)$, $1 \leq p < \infty$ (Corollary to Lemma 1 in Baxter & Rosenthal[16]). □

Proposition 3.10 *Suppose that P_T is reversible, P is irreducible, reversible, and π-a.e. geometrically ergodic, and $\pi = \pi P = \pi P_T$. Then $p_c P_T + (1 - p_c)P$, $0 \leq p_c < 1$, is irreducible, reversible and π-a.e. geometrically ergodic.*

Proof. Because P and P_T are reversible and P is irreducible, $p_c P_T + (1-p_c)P$ is also irreducible and reversible. By Theorem 2 of Roberts & Tweedie[17], π-a.e. geometric ergodicity is equivalent to $L^2(\pi)$-geometric ergodicity for chains that are reversible and irreducible. Thus, it is sufficient to prove the statement for $L^2(\pi)$-geometric ergodicity.

By Theorem 2 of Roberts & Rosenthal[15], $L^2(\pi)$-geometric ergodicity for a reversible chain is equivalent to having $r < 1$ such that for all $\mu \in L^2(\pi)$ with $\mu(E) = 0$,

$$\|\mu P\|_{L^2(\pi)} \leq r \|\mu\|_{L^2(\pi)}. \tag{20}$$

This condition along with the previous proposition and triangle inequality yields

$$\begin{aligned} \|\mu[p_c T + (1 - p_c)P]\|_{L^2(\pi)} &\leq p_c \|\mu P_T\|_{L^2(\pi)} + (1 - p_c)\|\mu P\|_{L^2(\pi)} \\ &\leq p_c \|\mu\|_{L^2(\pi)} + (1 - p_c) r \|\mu\|_{L^2(\pi)} \\ &= [p_c + (1 - p_c)r] \|\mu\|_{L^2(\pi)}, \end{aligned} \tag{21}$$

where $p_c + (1-p_c)r < p_c + (1-p_c) = 1$. Because $p_c P_T + (1-p_c)P$ is reversible, this implies that $p_c P_T + (1 - p_c)P$ is $L^2(\pi)$-geometrically ergodic. □

Finally, we need to consider how to relate the convergence on the power space to the convergence of the original one-chain kernel. The total variaton distance of a probability distribution μ from π can be related to the distance of the power distribution $\mu^{\otimes N}$ from $\pi^{\otimes N}$ as follows:

$$\|\mu - \pi\| \leq \|\mu^{\otimes N} - \pi^{\otimes N}\| \leq N \|\mu - \pi\|. \tag{22}$$

This implies that asymptotic convergence for a power chain is equivalent to the same kind of convergence for all of the component chains. Thus, any assumptions on the convergence rate about the ordinary one-chain kernel $P(x, dx')$ are equivalent to same kind of assumptions about the corresponding power kernel that updates each of the components *independently* using $P(x, dx')$. Hence, (by mixing in the crossing-over product kernels P_{TM} one at a time), we can summarize the above results as follows.

Theorem 3.11 *Suppose $P(x, dx')$ is a transition kernel with the invariant distribution $\pi(dx)$, $\{T_m\}_m$ is a finite set of crossing-over operators $T_m : E^N \to E^N$, for which $T_m^{-1} = T_m$ holds, and c_M are weights for some sets M of indexes $m \in M$ to the operators T_m such that $\sum c_M = 1$ and for each $c_M > 0$, all T_m, $m \in M$, operate on disjoint sets of individuals x_i. Denote the corresponding GO-MCMC kernel by*

$$P_{GO} := p_c \sum_M c_M P_{TM} + (1 - p_c) \prod_i P_i,$$

where $P_{TM} = \prod_{m \in M} P_{Tm}$, $P_i(\mathbf{x}, d\mathbf{x}) = P(x_i, dx_i') \prod_{j \neq i} \delta_{x_j}(dx_j')$, $i = 1, \ldots, N$, P_{Tm} is given by (14) and (18), and $0 \leq p_c < 1$ is the recombination rate. Then the following statements hold:

1. *If P is irreducible and aperiodic, then P_{GO} is irreducible, aperiodic, and converges $\pi^{\otimes N}$-a.e.*
2. *If P is ergodic, then P_{GO} converges $\pi^{\otimes N}$-a.e and is ergodic with appropriate choices of acceptance functions, (which should be evident in applications).*
3. *If P is irreducible, reversible, and π-a.e. geometrically ergodic, then P_{GO} is irreducible, reversible, and $\pi^{\otimes n}$-a.e. geometrically ergodic.*
4. *If P is uniformly ergodic, then P_{GO} is uniformly ergodic.*

□

Furthermore, the total variation distance of a probability distribution μ from $\pi^{\otimes N}$ on the power space can be related to the total variation distances of its marginal distributions from π such that for all $i = 1, \ldots, N$,

$$\|\mu(E^{(i-1)} \times (\cdot) \times E^{(N-i)}) - \pi(\cdot)\| \leq \|\mu - \pi^{\otimes N}\|. \tag{23}$$

This means that the above convergence limits for the chain on the power space also hold for each of the component chains.

Thus, the genetic recombination operators can be incorporated into any MCMC kernel preserving the convergence properties in usual cases. We could also base the the algorithm on a kernel $P(\mathbf{x}, d\mathbf{x}')$ already on the power space

E^N with the invariant distribution $\pi^{\otimes N}$ with an analogous result on preserving convergence.

Note that in discrete case, a stronger notion of irreducibility than the φ-irreducibility of general state spaces is often used [12, Chapter 4]. The discrete irreducibility means that all states communicate, i.e., for all x, y there is n such that $P^n(x, \{y\}) > 0$. It then follows that $\pi(\{x\}) > 0$ for all x if π is invariant. Hence, if the discrete irreducibility is assumed for $P(x, dx')$, the above π-a.e. and $\pi^{\otimes n}$-a.e. are equivalent to everywhere.

4 Relationship with other MCMC speedup methods

The Swendsen-Wang algorithm[8] and other algorithms derived from it such as the Wolff algorithm[9] can speed up MCMC calculations of Ising models by several orders of magnitude. The algorithms work by flipping a block (or several blocks) of spins at a time instead of a single spin and thereby moving through the state space much faster, avoiding the potential energy barriers between the local minima (corresponding to maxima of the probability density function $p(x) \propto \exp(-E(x))$. There are several variations of the basic algorithm corresponding to different ways of choosing the block(s) of spins to be flipped. Generally the method uses auxiliary random variables indicating which adjacent spins are considered bonded and only flips clusters of bonded spins. These bond variables, however, are usually updated independent of their previous values and so the algorithm can be seen as a plain Markov process with a sophisticated transition probability.

It is illustrative to compare the Swendsen-Wang -type algorithms and the present algorithm to obtain a clearer picture of what the current algorithm actually does. Not surprisingly, the Swendsen-Wang -type algorithms and the present algorithm are related in a fundamental way. Both algorithms are useful for models where the "most obvious" choice for $q(x, x')$ suffers from a too weakly connected neighborhood structure. In the Ising case, flipping a single spin in the middle of a large frozen block can be next to impossible energetically, yet the whole block can be reversed without resistance. In our present case, the slow random walk between the multiple modes in one dimension has to be repeated again and again as the other dimensions go through different combinations of their modes, in order to obtain convergence.

Both algorithms exploit the fact that it is possible to add into $q(x, x')$ easier "channels" between the maxima because something about the structure of the space is assumed to be known. In the Ising model, it is known that flipping a block of spins has little effect on the probability and in our present genetic model, it is assumed that there is some decomposability in the function being modeled. The fact that we use the distribution in power space is interesting because we are in fact taking advantage of the structure, not in the distribution for the single individual x, but the composite distribution $\mathbf{p}(x_1, \ldots, x_N)$ shown

in Equation 3. The basic assumption is that if a relatively probable composite point is drawn from this distribution then it is likely that the composite point obtained by applying a crossing-over operation to this composite point is also relatively probable. In this way, we hope to be able to travel between the local maxima faster.

However, the similarities between the current and the Swendsen-Wang -type algorithms end at that abstract level: on a more detailed level, the Swendsen-Wang algorithms work by knowing that $(- - \ - -)$ is close to $(+ + ++)$ but our algorithm by assuming that if both $(- - \ - -)$ and $(+ + ++)$ have high probabilities in our distribution, then $(- - ++)$ and $(+ + --)$ should be tried as well, in case the function would be approximately decomposable in this way. Thus, the two algorithms solve completely different problems but simply use the same tool: adding more information about the nature of the problem to $q(x, x')$.

Redefining the locality of the transitions is not the only way of obtaining a better neighborhood structure: for instance, the Metropolis-Coupled MCMC (MCMCMC) method proposed by Geyer (see e.g. [10]) works by running parallel Markov chains for multiple gradually tempered distributions $p_1, p_2, \ldots, p_N$, where p_1 is the target distribution. The tempered distributions are usually defined as

$$p_i(x) \propto (p_1(x))^{1/T_i} \tag{24}$$

for some temperatures $1 = T_1 < T_2 < \ldots < T_N$. Mathematically, the algorithm simulates the composite probability distribution

$$\mathbf{p}(\mathbf{x}) = \mathbf{p}(x_1, \ldots, x_N) = p_1(x_1) \cdots p_N(x_N). \tag{25}$$

In addition to the standard updates of the individual chains, the algorithm proposes transitions that try to swap the states of two of the chains. For swapping the states x_i and x_j, the Metropolis acceptance probability is

$$\min\left\{1, \frac{\mathbf{p}(\mathbf{x}')}{\mathbf{p}(\mathbf{x})}\right\} = \min\left\{1, \frac{p_i(x_j)p_j(x_i)}{p_i(x_i)p_j(x_j)}\right\}. \tag{26}$$

The idea of MCMCMC is to transfer the faster mixing from the tempered chains to the target chain through the transitions between chains of different temperatures. Only the state of the first chain is used for actual integration.

The present algorithm is similar to the MCMCMC algorithm in that it also uses a chain on a power space and incorporates transitions affecting two of the parallel states. The most important difference is that MCMCMC connects the local maxima of the target distribution by more likely detours through the tempered distributions while GO-MCMC achieves faster mixing by (in the usual sense) *nonlocal* recombination transitions so that step-by-step travel between maxima is not necessary.

There are also some other algorithms that work in a product or power space such as adaptive direction sampling and adaptive Metropolis sampling (cf. [18])

and population-based Monte Carlo algorithms surveyed in Iba[19]. These algorithms, however, are related to the present algorithm only at an abstract level.

While preparing a revision of this article we came across a conference paper[20] describing a similar idea of applying genetic crossing-over in MCMC context.

5 Simulations

In this section, we discuss the results of numerical simulations carried out on two discrete systems. The first system is a qualitative model of an ideal target for GO-MCMC: a fully decomposable multidimensional system whose independent subsystems move only slowly between their modes. As a more realistic example, the second, non-ideal system is only near-decomposable to low-dimensional subsystems. In both cases, using genetic operators speeds up the integration substantially.

The longest simulation runs for the sample models consumed in the order of 10^8 random numbers. All the simulations presented in this article were performed using the Mersenne Twister[21] (MT) random number generator from the GNU Scientific Library[22]. MT has an extremly long period of $2^{19937} - 1$ which is sufficient. Furthermore, different seeds give non-overlapping sequences with a high probability, so that it is possible to obtain independent test runs. No significant differences were found between the MT simulations and simulations carried out with the Linux[23] `/dev/random` generator.

In order to evaluate convergence over time, a measure of distance between the observed frequencies and the equilibrium distribution is needed. A natural choice is the Kullback-Leibler distance, which is simply the difference between cross-entropy and the entropy of the target distribution and is given by

$$d(p_{\text{obs}}, p_{\text{theor}}) = \sum_i p_{\text{theor}}(i) \cdot \log \left(\frac{p_{\text{obs}}(i)}{p_{\text{theor}}(i)} \right). \tag{27}$$

The observed frequency p_{obs} is given by the estimate

$$p_{\text{obs}}(i) = \frac{f_i + 1}{\sum_i (f_i + 1)}, \tag{28}$$

where f_i is the observed count of state i. Adding one to the actual counts is needed in order to obtain finite distance for a small run length T.

In the Figures, we use a special histogram of the frequencies f_i/T to illustrate the observed distribution given by an MCMC simulation. These histograms show the number of bit-vector counts f_i falling between evenly spaced ranges on simulated runs. The histogram does not graph the frequency of a state as a function of the state as one might first assume, but the number of *states* that have a particular frequency.

The implementations of the algorithm for the sample models follow the example algorithm given in Figure 1. Recombination rate $p_c = 0.4$ and population size $N = 4$ was used for all simulations.

5.1 Ideal case: a fully decomposable system

In its most basic form, a recombination operation swaps some components between individuals. If the exchanged components are independent of the other components, the transition does not affect the composite probability of the population and will always be accepted. Thus, the ideal target for GO-MCMC is a system consisting of entirely independent components that move only slowly between their modes. In such a case, the components making up the individuals can be freely recombined with no change in probability, thus creating shortcuts between the local maxima corresponding to different combinations of the independent components.

A product distribution consisting of two-modal independent components, such as Equation (7), is the simplest form of the above ideal case. We consider a discrete qualitative model of this distribution with n binary variables, corresponding to the independent components. The two values of a bit are considered to represent the two modes of the corresponding component. Suppose that a random-scan Metropolis algorithm is run on the modelled system and that the components move only through paths of low probability between their modes. We model the slow movement by flipping one randomly chosen bit with probability $1 - a$ on each time step, where $0 \leq a \leq 1$ is a constant autocorrelation parameter.

This simple model captures the essence of the ideal case in that it has slow-moving multimodal components and that the number of local maxima grows exponentially with the number of dimensions. For short simulations, the sampled states will differ only in some bit positions and thus will show correlation between variables even though the variables are in fact independent. Increasing the number of variables or choosing a closer to one will make the spurious correlation stronger requiring a longer run length.

In order to incorporate crossing-over operations to the system, consider a population $x_1, \ldots, x_N$ of N n-bit strings. The recombination operation will first divide the population into randomly chosen pairs and then for each pair, choose a random n-bit mask from the uniform distribution, and cross over the variables specified by the mask. The recombination proposition will always be accepted as the modelled variables are independent and so the acceptance probability will simplify to 1.

Each mutation or recombination round was counted as N time steps as they both would require N evaluations of $p(x)$ in a real system. After each round, the whole population was saved to the sample vector. If we take $p_c = 0$, the system will correspond to the standard Metropolis algorithm with N parallel chains. By increasing p_c we can model the effect of recombination on the perceived correlation between variables. Adjusting the parameter a allows us to model different levels of autocorrelation.

The starting population for the simulations was chosen randomly from the uniform distribution, which is also the target distribution of the model. Nevertheless, a burn-in period of 100000 time steps was inserted before collecting the sample vector in order to eliminate any systematic bias in the initial state of

the simulation. Furthemore, all graphs show the results of multiple consecutive test runs.

Figure 3 graphs the convergence of the observed distribution as a function of time for the cases without recombination, with recombination, and for the non-autocorrelated ideal case, where the samples are drawn from the equilibrium distribution. By comparing the frequency histograms in the graphs, one can see that GO-MCMC performs better than standard MCMC even when standard MCMC is run for four times as long as the GO-MCMC case. The figure shows also the Kullback-Leibler distance from the equilibrium distribution as a function of time. Figure 4 graphs the same kind of histograms with the Kullback-Leibler distance, but this time as a function of the autocorrelation parameter a. The figure shows that GO-MCMC maintains good convergence under slower step-by-step motion between local maxima than the plain traditional MCMC method.

5.2 Approximately decomposable system

In reality, functions are more often approximately than exactly decomposable. As a numerical example, we consider a discrete system which can be thought to model the general class of such functions, whether they are discrete or continuous. The function is not exactly decomposable to one-dimensional components but it is approximately decomposable to low-dimensional components so that the wandering between the local maxima can be speeded up through the genetic operators.

There are $n = 3k$ binary dimensions divided to groups of three adjacent bits. The probability distribution p has two terms: each group that is not 000 or 111 is penalized by multiplying the probability by a factor of $1/200$, and if all groups are 000 or 111, the function is multiplied by $1/2$ if the parity (number of 111 states) is odd. The starting state is random in all dimensions.

The first term can be thought of as the potential energy barrier separating the local maxima of the probability density function. This makes the random walk of the system through the states slow. The second term can be thought of as the actual interesting information that we are trying to extract by integrating this function.

The two MCMC algorithms used on this function, GO-MCMC and trad-MCMC differ only in that GO-MCMC includes the crossing-over operation. The two operators are mutation, which flips a random bit, and one-point crossing-over, which takes two individuals from the population and exchanges their tails. A recombination round consists of dividing the population in random pairs and proposing crossing-over on each pair. After every round, each individual is saved to the sample vector.

The simulation was carried out for $n = 3 \cdot 8$ and $3 \cdot 12$ variables. An additional 500000 time step burn-in was simulated before collecting the actual samples. The distribution of the frequencies of the 2^k "legal" states with only 000 or 111 groups along with the corresponding Kullback-Leibler distance from the

equilibrium is shown in Figure 5 and the ratio between odd and even states in Figure 6.

As is obvious from the Figures, GO-MCMC converged much faster for this problem, as expected: it can move through the state space far more rapidly by not having to mount the potential energy barrier of the "illegal" states in order to travel between the maxima.

6 Conclusions

A new method, GO-MCMC, for applying genetic operators in MCMC calculations has been presented. The crux of the method lies in accepting or rejecting the offspring of a reversible genetic operation all together as opposed to one by one. This ensures detailed balance so that no bias is introduced to the MCMC process by the genetic operators. The algorithm converged much faster than ordinary MCMC without genetic operators on a discrete example problem.

The algorithm is clearly related to earlier MCMC speedup algorithms like the Swendsen-Wang algorithm, as it makes it easier for the system to travel between local maxima, but the present algorithm is likely to be applicable to a greater variety of models because it is not tied to a particular problem. There are, of course, problems that this algorithm is unsuited for — the canonical example of a difficult problem is a distribution of many variables where the variables are very strongly correlated. In such problems, most algorithms will fail — the representation of the problem needs to be changed. However, the class of functions for which genetic algorithms are applicable is quite wide so there are likely to be several problems in which the current algorithm will be useful.

7 Acknowledgements

We would like to thank prof. Pekka Orponen and prof. Antti Penttinen for discussions regarding this idea and prof. Pertti Mattila for measure theoretic advice.

References

[1] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092, 1953.

[2] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[3] J. Besag, P. J. Green, D. Higdon, and K. Mengersen. Bayesian computation and stochastic systems. *Statistical Science*, 10:3–66, 1995.

[4] Alan D. Sokal. Monte Carlo methods in statistical mechanics: Foundations and new algorithms. Lecture notes, Cargèse Summer School, September 1996.

[5] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[6] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, Reading, MA, 1989.

[7] Luke Tierney. Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22:1701–1762, 1994.

[8] J. S. Wang R. H. Swendsen. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58:86–88, 1987.

[9] Ulli Wolff. Comparison between cluster Monte Carlo algorithms in the Ising model. *Physical Letters B*, 228:379–382, 1989.

[10] C. J. Geyer and E. A. Thompson. Annealing Markov chain Monte Carlo with application to ancestral inference. *J. Am. Statist. Assoc.*, 90:909–920, 1995.

[11] D. B. Fogel. Some recent important foundational results in evolutionary computation. In Kaisa Miettinen, Marko M. Mäkelä, Pekka Neittaanmäki, and Jacques Périaux, editors, *Evolutionary Algorithms in Engineering and Computer Science*, chapter 4, pages 55–71. Wiley, West Sussex, England, 1999.

[12] S. P. Meyn and R. L. Tweedie. *Markov Chains and Stochastic Stability.* Springer, London, 1993.

[13] Luke Tierney. A note on Metropolis–Hastings kernels for general state spaces. *The Annals of Applied Probability*, 8:1–9, 1998.

[14] Peter J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[15] G. O. Roberts and J. S. Rosenthal. Geometric ergodicity and hybrid Markov chains. *Electronic Communications in Probability*, 2:13–25, 1997. Paper no. 2.

[16] J. R. Baxter and J. S. Rosenthal. Rates of convergence for everywhere positive Markov chains. *Stat. Prob. Lett.*, 22:333–338, 1995.

[17] G. O. Roberts and R. L. Tweedie. Geometric L^2 and L^1 convergence are equivalent for reversible Markov chains. Preprint, April 2000.

[18] Walter R. Gilks and Gareth O. Roberts. Strategies for improving MCMC. In W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors, *Markov Chain Monte Carlo in Practice*, chapter 6, pages 89–114. Chapman & Hall, London, 1996.

[19] Yukito Iba. Population-based monte carlo algorithms. *Proc. 2000 Workshop on Information-Based Induction Sciences*, pages 245–250, 2000.

[20] James Myers, Kathryn Laskey, and Tod Levitt. Learning Bayesian networks from incomplete data with stochastic search algorithms. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–99)*, pages 476–485, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[21] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Transactions on Modeling and Computer Simulation*, 8:3–30, 1998.

[22] Mark Galassi, Jim Davies, James Theiler, Brian Gough, Reid Priedhorsky, Gerard Jungman, Michael Booth, and Fabrice Rossi. GSL — The GNU Scientific Library, version 0.6. Available from `ftp://sourceware.cygnus.com:/pub/gsl/`.

[23] Linus Torvalds et al. Linux kernel, version 2.2.16. Available from `http://www.kernel.org`, 2000.

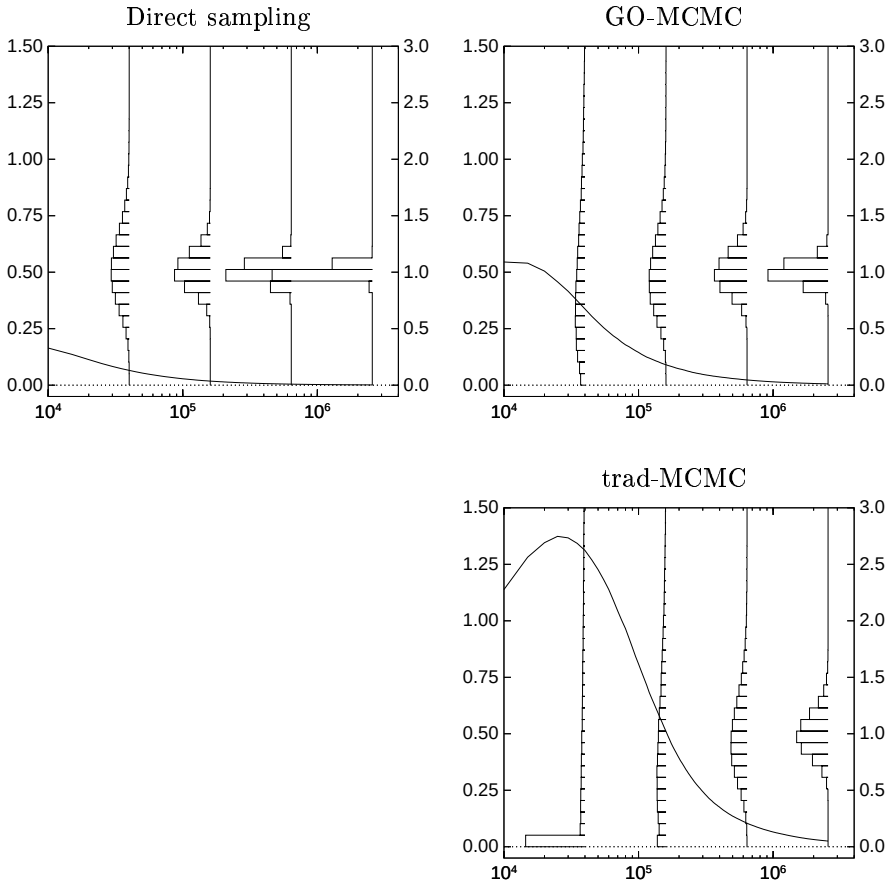

Figure 3: Special histograms (see Section 5) representing the distributions of the 2^n observed bit-vector frequencies (right scale) at times $T = 40000, 160000, 640000, 2560000$ in the qualitative model of an ideal fully decomposable system. The frequency scale is normalized so that 1 corresponds to the expected frequency of $1/2^n$. In addition, there are curves showing the Kullback-Leibler distance from the equilibrium distribution (left scale) as a function of time (note that the time scale is logarithmic). The three graphs represent direct sampling, GO-MCMC, and trad-MCMC algorithms. As the run length increases, the distributions of the three cases all get sharper and converge to the point mass of 2^n at 1. By comparing adjacent histograms one can see that GO-MCMC mixes better than trad-MCMC even when trad-MCMC is run for four times as long as GO-MCMC. Similarly, direct sampling converges about four times as fast as GO-MCMC. The graphed data is measured as an average of 10 test runs with the parameters $a = 0.9$, $n = 12$, and 100000 burn-in.

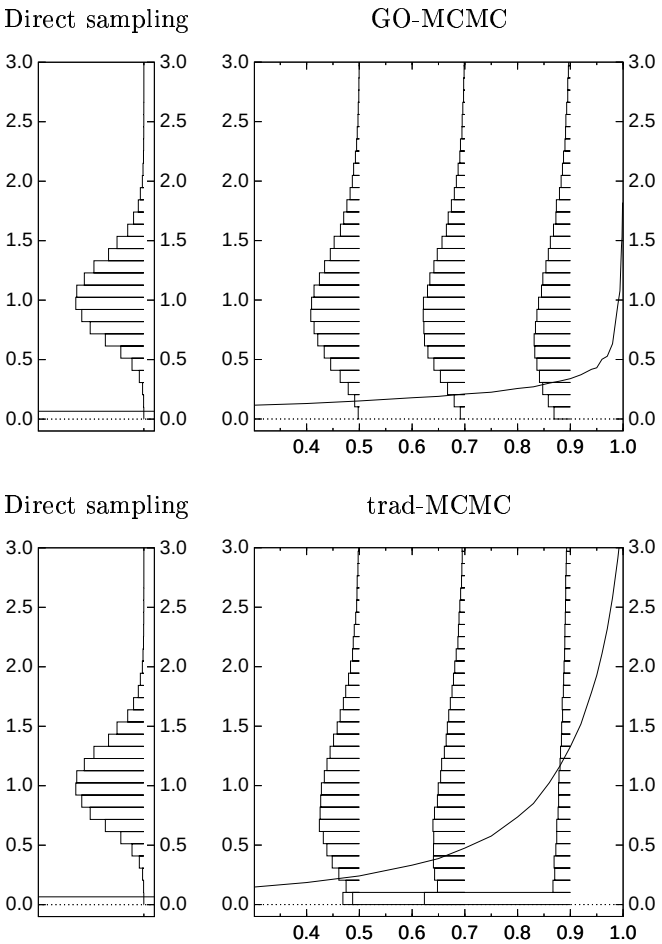

Figure 4: Frequency histograms and Kullback-Leibler distance curves as in Figure 3, representing convergence as a function of the autocorrelation parameter a. The three graphs correspond to GO-MCMC, trad-MCMC, and as a reference, direct sampling. As autocorrelation increases, the distribution obtained by the trad-MCMC sampler deteriorates. This indicates that fewer states are visited and a stronger correlation between variables is observed when the same states repeat over and over again. Using recombination keeps the distribution closer to the ideal case and greatly reduces the spurious correlations. The effect of recombination gets stronger as a gets closer to 1. The data is measured as an average of 10 test runs with $T = 40000$ time steps, 100000 burn-in, and $n = 12$ variables.

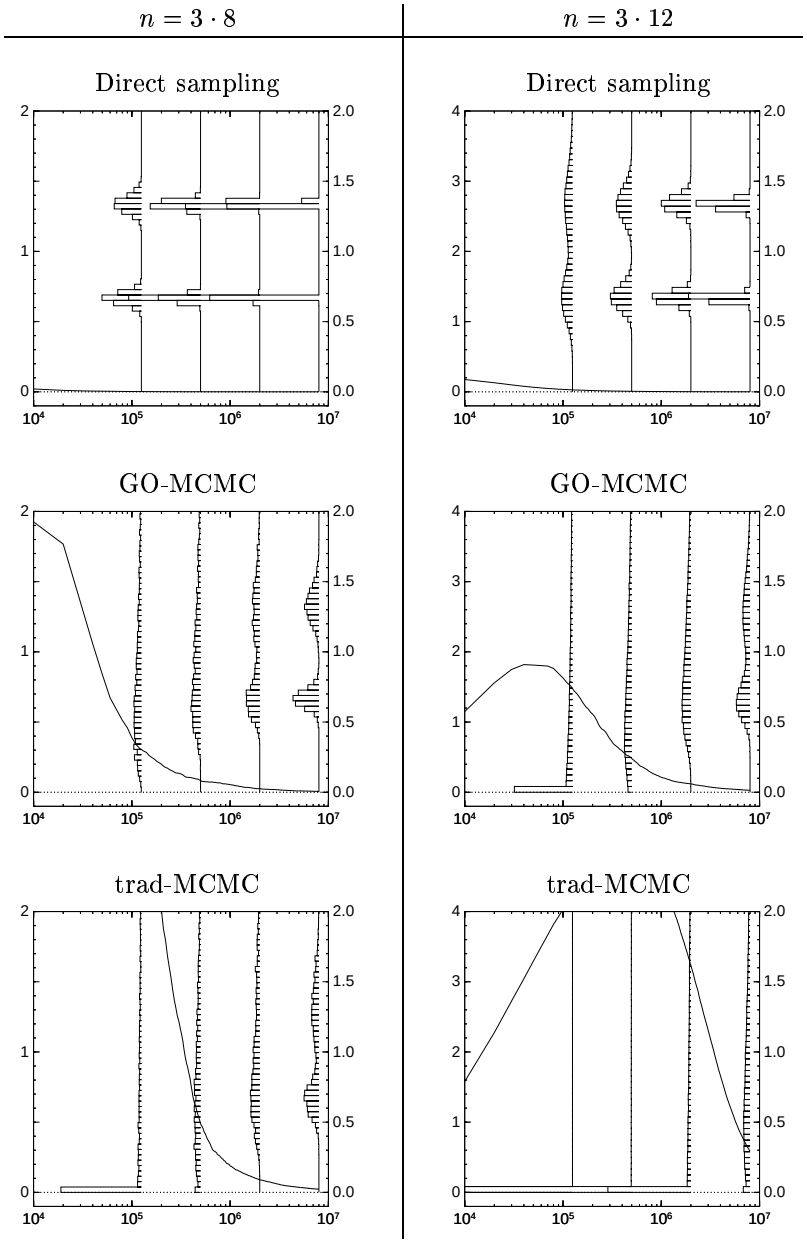

Figure 5: Convergence graphs for the approximately decomposable model with $n = 3 \cdot 8$ and $n = 3 \cdot 12$ variables as in Figure 3. The histograms show the distributions of the $2^{(n/3)}$ bit-vector frequencies at times $T = 125000, 500000, 2000000, 8000000$ and the curves show the Kullback-Leibler distance from the equilibrium distribution. The graphs show the average of 10 test runs.

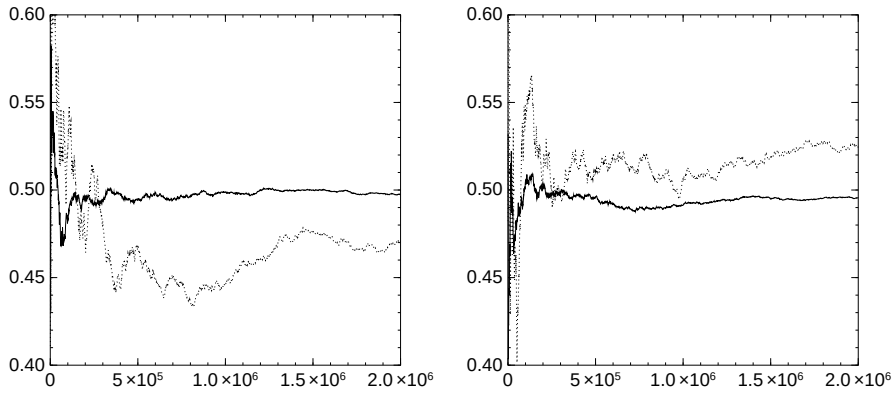

Figure 6: Ratio between states of odd and even parity as a function of time for $n = 3 \cdot 8$ (left) and $n = 3 \cdot 12$ (right) variables in the approximate decomposable model. The solid line represents one run of the GO-MCMC algorithm and the dotted line one run of the traditional MCMC.