Proof of uniform sampling of binary matrices with fixed row sums and column sums for the fast Curveball algorithm

C. J. Carstens

School of Mathematical and Geospatial Sciences, RMIT University, Melbourne, Victoria 3000, Australia

(Received 15 January 2015; published 29 April 2015)

Randomization of binary matrices has become one of the most important quantitative tools in modern computational biology. The equivalent problem of generating random directed networks with fixed degree sequences has also attracted a lot of attention. However, it is very challenging to generate truly unbiased random matrices with fixed row and column sums. Strona *et al.* [Nat. Commun. **5**, 4114 (2014)] introduce the innovative Curveball algorithm and give numerical support for the proposition that it generates truly random matrices. In this paper, we present a rigorous proof of convergence to the uniform distribution. Furthermore, we show the Curveball algorithm must include certain failed trades to ensure uniform sampling.

DOI: 10.1103/PhysRevE.91.042812 PACS number(s): 89.75.Hc, 89.75.Fb, 02.50.Ga, 02.70.−c

I. INTRODUCTION

The randomization of binary matrices with fixed row sums and column sums is used extensively throughout computational biology. For instance, in numerical ecology, it is known as the recombining of presence-absence matrices where matrices represent either species per site or species interactions [1–3]. In biochemistry, it is used in the analysis of protein interaction networks [4,5]. Furthermore, square binary matrices with fixed row sums and column sums are equivalent to directed networks with fixed degree sequences, which have attracted a lot of interest in network science [6–10]. We discuss these equivalent problems in terms of matrix randomization. However, the reader should keep in mind that all results hold equally well for directed networks.

It is hard to generate truly unbiased samples of binary matrices with fixed row sums and column sums. Existing randomization methods fall into two categories: the "fill methods" and the "swap methods." Fill methods *construct* a matrix, starting with a matrix of zeros, then adding ones until reaching the desired row sums and column sums [6–8]. These methods are generally fast. However they either produce a biased sample or only rarely produce a binary matrix [10,11].

Swap methods [12], on the other hand, *randomize* a given matrix by repeatedly making small changes. Care must be taken when implementing swap methods since removing repeated matrices causes these algorithms to sample with bias [3,9,11,13–15]. However, when correctly implemented, the switching method [16] has been shown to sample uniformly [3,9].

Recently the Curveball algorithm was introduced as a *much faster* procedure to sample random binary matrices with fixed row sums and column sums [17]. Similar to swap methods, the Curveball algorithm randomizes a binary matrix by repeatedly making small changes. The supposition that this algorithm samples uniformly, although backed by numerical experiments, is speculative. In this paper we rigorously prove that the Curveball algorithm generates truly random matrices.

The Curveball algorithm has two distinct properties that cause repeated states. We show that one type of repeated states can be excluded while maintaining uniform sampling, whereas excluding the other type leads to biased sampling. Repeated states generally slow down convergence, leading us to investigate the convergence and run-time behavior of the Curveball algorithm with and without the former type of repeated states. We found that the modified Curveball algorithm outperforms the original Curveball algorithms for small matrices. However, for larger matrices, the original Curveball algorithm is slightly faster.

The remainder of this paper is organized as follows. Section II describes the Curveball algorithm in Markov chain terminology, pointing out its similarity to the switching method. Furthermore, it presents our proof of uniform sampling for the Curveball algorithm. Section III discusses the convergence properties of the Curveball algorithm when repeated states are excluded. Section IV presents numerical results on the convergence and run time of the original and modified Curveball algorithm. Finally, Sec. V concludes with a recommendation to use the original Curveball algorithm.

II. THE CURVEBALL ALGORITHM

The Curveball algorithm was introduced in Ref. [17]. This section provides a proof that the sampling of the Curveball algorithm is unbiased. Our proof uses the following well-known theorem about Markov chains (see, for instance, Ref. [18, Theorem 7.10]).

Theorem II.1. A finite, irreducible, and aperiodic Markov chain converges to a unique stationary distribution on its state space S. Let P_{AB} denote the transition probability from state A to B. If there exists a probability distribution $\boldsymbol{\pi}$ on S such that the detailed balance equations,

$$\pi_A P_{AB} = \pi_B P_{BA}$$

hold for all states $A, B \in S$, then $\boldsymbol{\pi}$ is this unique stationary distribution.

This theorem implies that a finite, irreducible, and aperiodic Markov chain converges to the uniform distribution if $P_{AB} = P_{BA}$ for all states A, B.

We start this section by describing the Curveball algorithm as a Markov chain. We then show that this Markov chain is finite, irreducible, and aperiodic. Finally, we derive its

*corriejacobien.carstens@rmit.edu.au

 ©2015 American Physical Society

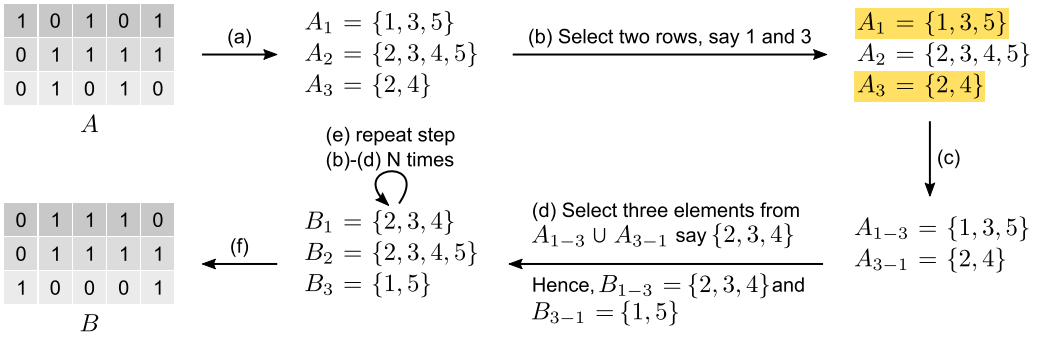


FIG. 1. (Color online) A *trade* in the Curveball algorithm consists of (a) converting a binary matrix into lists of indices A_i for each row i. In step (b) two rows are selected, in this case rows 1 and 3. (c) The set differences A_{1-3} and A_{3-1} are extracted. (d) Set B_1 is formed by removing A_{1-3} from A_1 and adding $|A_{1-3}|$ elements randomly chosen from $A_{1-3} \cup A_{3-1}$, in this case {2,3,4}. B_3 is formed by removing A_{3-1} from A_3 and adding the remaining elements of $A_{1-3} \cup A_{3-1}$. Step (f) converts the resulting lists of indices B_i into matrix B. In the original description of the Curveball algorithm step (b)-(d) are repeated N times in step (e). This is equivalent to repeating step (a)-(d) and step (f) N times.

transition probabilities and show that the detailed balance equations hold with respect to the uniform distribution.

A. Description of the Curveball algorithm

The Curveball algorithm randomizes a binary $n \times m$ matrix A using the following steps (see also Fig. 1). (a) Transform A into lists of indices, A_i for each row i, corresponding to ones in that row [19], (b) select two of these lists A_i and A_j at random, (c) compare the lists and let A_{i-j} be all indices that are in A_i but not in A_j. Similarly define A_{j-i}. (d) Create a new list B_i by removing A_{i-j} from A_i and adding the same number of elements randomly chosen from $A_{i-j} \cup A_{j-i}$. Combine $A_j \backslash A_{j-i}$ with the remaining elements of $A_{i-j} \cup A_{j-i}$ to form B_j [20]. (e) Reiterate steps (b)–(d) N times for a certain fixed number N, and (f) form a new matrix from the resulting lists.

Notice that step (a) is the inverse of step (f) and vice versa, hence a run of the Curveball algorithm (a)–(f) is equivalent to reiterating steps (a)–(d) and (f) N times. We use the latter description to show that the Curveball algorithm corresponds to a Markov chain with state space all binary matrices with row sums and column sums equal to those of A. We will refer to one iteration of steps (a)–(d) and (f) as a *trade* and the number of exchanged indices $|B_i \backslash A_i| = |B_j \backslash A_j|$ as the *size* of the trade.

The Curveball algorithm corresponds to a discrete stochastic process: a sequence of random variables corresponding to binary matrices with fixed row sums and column sums. The first matrix in the sequence is A, and each trade results in a subsequent matrix in the sequence. Notice that there is always a nonzero probability for consecutive matrices to be equal since trades can have size zero. That is, either A_{i-j} or A_{j-i} could be empty, and even if they are not, the elements randomly selected from $A_{i-j} \cup A_{j-i}$ to create B_i could be exactly the elements in A_{i-j}. This stochastic process is clearly Markovian: The probability of a matrix appearing in the sequence depends only on its immediate predecessor. Indeed, the Curveball algorithm corresponds to a Markov chain. This Markov chain is *finite* since its state space, the set of all binary matrices with row sums and column sums equal to those of the initial matrix A, is finite.

B. Comparison to the switching method

As mentioned in Ref. [17], the Curveball algorithm and switching method are "in a certain way closely related." Here, we use this similarity to show that the Markov chain corresponding to the Curveball algorithm is irreducible. Using Theorem II.1, the switching method has been shown to sample uniformly [3,9,10,13]. In particular, its Markov chain has been shown to be irreducible [13,21].

The state space of the switching method is the same as that of the Curveball algorithm, namely the set of all binary matrices that have the same row sums and column sums as the initial matrix. Therefore, the state graph of the switching method and the state graph of the Curveball algorithm have the same set of vertices. However, in general, they have different edges.

The switching method generates a sequence of matrices where each subsequent matrix is the result of a *switch* instead of a *trade*. A switch from a matrix A replaces a checkerboard, $A_{ik} = A_{jl} = 1$ and $A_{il} = A_{jk} = 0$, by the opposite checkerboard, $A_{ik} = A_{jl} = 0$ and $A_{il} = A_{jk} = 1$ (see Fig. 2). This switch corresponds to a trade of size one: Row A_i trades index k for index l with row A_j.

Hence, each switch in the switching method corresponds to a trade in the Curveball algorithm. This implies that the state graph of the switching method is a *subgraph* of the state graph of the Curveball algorithm with the same set of vertices (see Fig. 3). In particular, since irreducibility of a Markov chain equals strong connectivity of its state graph, this means that the Markov chain of the Curveball algorithm is *irreducible*.

C. Proof of uniform sampling

In the previous sections we have seen that the Markov chain corresponding to the Curveball algorithm is finite and

$$\begin{array}{c} \\ i \\ j \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} & k & & l & \end{array} \\ \begin{pmatrix} \vdots & & \vdots & \\ \cdots & 1 & \cdots & 0 & \cdots \\ \vdots & & \vdots & \\ \cdots & 0 & \cdots & 1 & \cdots \\ \vdots & & \vdots & \end{pmatrix} \end{array} \leftrightarrow \begin{array}{c} \\ i \\ j \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} & k & & l & \end{array} \\ \begin{pmatrix} \vdots & & \vdots & \\ \cdots & 0 & \cdots & 1 & \cdots \\ \vdots & & \vdots & \\ \cdots & 1 & \cdots & 0 & \cdots \\ \vdots & & \vdots & \end{pmatrix} \end{array}$$

FIG. 2. A switch in the switching algorithm.

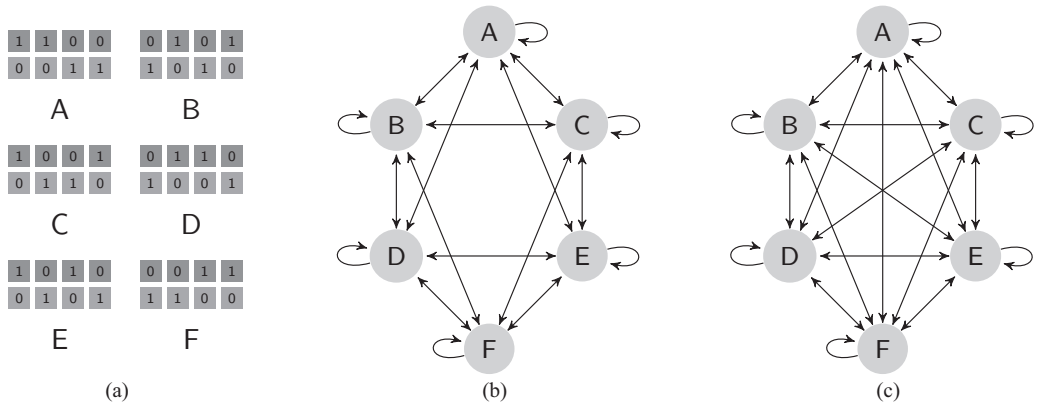


FIG. 3. (a) There are six different matrices with row sums (2,2) and column sums (1,1,1,1). (b) The structure of the state graph for the switching method. Notice that there are no edges between states A and F, between states B and E, and between states C and D since these pairs of matrices differ by two switches. (c) The structure of the state graph of the Curveball algorithm. There are edges between states A and F, between states B and E, and between states C and D since these pairs of matrices differ by one trade (of size two).

irreducible. It is also aperiodic since step (d) ensures that there is a nonzero probability to repeat each state and hence each vertex in the state graph has a self-loop. Thus, to prove that the Curveball algorithm converges to the uniform distribution, it remains to show that its transition probabilities satisfy $P_{AB} = P_{BA}$ for all A and B.

Lemma II.2. The transition probabilities P_{AB} of the Curveball algorithm are given by

$$P_{AB} = \begin{cases} \frac{2}{n(n-1)} \frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}|+|A_{j-i}|)!}, & \text{if} \quad A \text{ and } B \text{ differ in a trade between rows } i \text{ and } j, \\ 1 - \sum_{C \neq A} P_{AC}, & \text{if} \quad A = B, \\ 0, & \text{otherwise,} \end{cases}$$

where n is the number of rows of A (and B) [22].

Proof. Let A and B be distinct binary $n \times m$ matrices that differ by a trade between two rows i and j. The probability of selecting these rows in A is $\frac{2}{n(n-1)}$. In step (d) of the Curveball algorithm, set A_{i-j} of indices in row i but not in row j is taken from A_i and put together with set A_{j-i} taken from A_j. The resulting set is shuffled, the first $|A_{i-j}|$ elements are returned to A_i, and the remaining elements are returned to A_j. The probability that shuffling results in state B equals the inverse of the number of ways you can select $|A_{i-j}|$ unordered elements from a set of $|A_{i-j}| + |A_{j-i}|$ elements. This probability thus equals $\frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}|+|A_{j-i}|)!}$. ■

Theorem II.3. The Markov chain corresponding to the Curveball algorithm converges to the uniform distribution on its state space S.

Proof. We have seen that the Markov chain corresponding to the Curveball algorithm is finite, irreducible, and aperiodic. It remains to show that $P_{AB} = P_{BA}$ for all states $A, B \in S$. If $A = B$ or if A and B differ by more than one trade, this equality clearly holds.

Let A and B differ by one trade between rows i and j. Now B_{i-j} is exactly the set of $|A_{i-j}|$ elements randomly chosen from $A_{i-j} \cup A_{j-i}$ in step (d) to form B_i, and hence $|B_{i-j}| = |A_{i-j}|$. Similarly B_{j-i} is the set of remaining elements in $A_{i-j} \cup A_{j-i}$, and hence $|B_{j-i}| = |A_{i-j} \cup A_{j-i}| - |A_{i-j}|$. This equals $|A_{j-i}|$ since $A_{i-j} \cap A_{j-i}$ is by definition empty.

Thus we find

$$P_{AB} = \frac{2}{n(n-1)} \frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}| + |A_{j-i}|)!} = \frac{2}{n(n-1)} \frac{|B_{i-j}|!|B_{j-i}|!}{(|B_{i-j}| + |B_{j-i}|)!} = P_{BA}.$$

■

III. MODIFYING THE CURVEBALL ALGORITHM

In this section we discuss two modifications of the Curveball algorithm and their stationary distributions. When the probabilities of repeating states are high, the mixing time of a Markov chain increases [9]. It is thus desirable to refrain from unnecessary repetitions. The following two situations cause repeated states in the Curveball algorithm: First, when two rows are selected that do not allow any trades and second when the shuffling of $A_{i-j} \cup A_{j-i}$ results in sets $B_i = A_i$ and $B_j = A_j$, in other words, when this shuffling leaves rows i and j unchanged. We will refer to the former as *no-trade row pairs* and to the latter as *no-trade shuffles*.

A. Excluding no-trade shuffles

We first show that the Curveball algorithm may be adjusted by excluding no-trade shuffles and that this modification only changes the stationary distribution for a pathological class P of matrices. The no-trade shuffles can be excluded by modifying

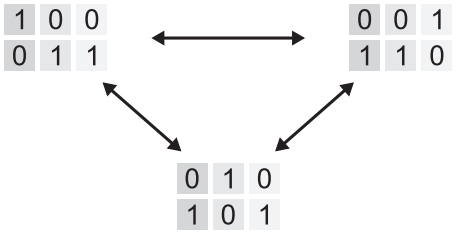


FIG. 4. When a matrix A contains two rows i and j that can make a trade and differ at more than two indices, it contains a submatrix of one of the forms above. This implies that there is a path of three trades as well as a path of two trades starting and ending at A. Thus state A is aperiodic, and hence the finite Markov chain is aperiodic.

step (d). Instead of always accepting the newly created lists B_i and B_j, repeat this step until $B_i \neq A_i$ and $B_j \neq A_j$, in other words until a trade of size at least one has been made. The transition probability for distinct neighboring states A and B then becomes

$$P_{AB} = \frac{2}{n(n-1)} \frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}| + |A_{j-i}|)! - (|A_{i-j}|!|A_{j-i}|!)}.$$

This probability only depends on n, $|A_{i-j}|$, and $|A_{j-i}|$, and thus $P_{AB} = P_{BA}$. Furthermore, this Markov chain is irreducible by the same argument as before. Finally, we need to find out under which conditions this chain is aperiodic. The argument for the aperiodicity of the Curveball algorithm cannot be used since it relied on the inclusion of no-trade shuffles.

Lemma III.1. Let P be the subset of binary matrices that are, up to a reordering of columns, equal to the identity matrix together with an arbitrary number of columns of just ones and an arbitrary number of columns of just zeros.

The Markov chain corresponding to the Curveball algorithm without no-trade shuffles is aperiodic for a matrix A if and only if $A \notin P$.

Proof. We prove this lemma by contrapositive, that is we prove that the Markov chain corresponding to the Curveball algorithm without no-trade shuffles is periodic if and only if $A \in P$.

We first show that if the Markov chain is periodic, then $A \in P$. It is clear that the Markov chain cannot be periodic if the initial matrix contains no-trade row pairs, and thus $|A_{i-j}| > 0$ for all i, j. Furthermore, if all rows in a matrix can make trades and there exists a pair of rows which differ at more than two indices, then the Markov chain is aperiodic as illustrated in Fig. 4. Hence $|A_{i-j}| \leqslant 1$ for all i, j.

Thus the only matrices A for which the Markov chain could potentially be periodic are those with $|A_{i-j}| = 1$ for all row pairs i and j. These are exactly the matrices in P.

We now show that the Markov chain is periodic for all matrices $A \in P$. Columns consisting of just ones or just zeros are left invariant by the Curveball algorithm. Hence without loss of generality we may assume that A is an $n \times n$ identity matrix. Each trade corresponds to a column swap, in other words a transposition of columns. The identity perturbation is even and can thus only be formed by an even number of transpositions of columns, which means that this Markov chain is two periodic. ■

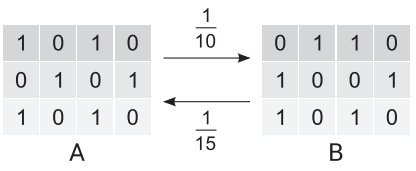
To summarize, no-trade shuffles can be removed from the Curveball algorithm without affecting the stationary distribution of the Markov chain for all matrices not in P. In practice all matrices of interest are randomized without bias. From now on, we will refer to this modified Curveball algorithm for matrices not in P as the good-shuffle Curveball algorithm [23].

B. Excluding no-trade row pairs

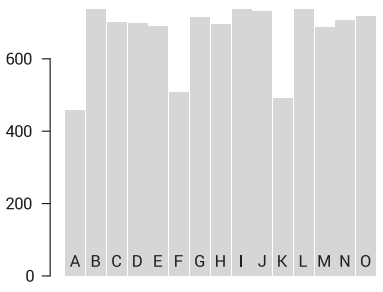
We now show that repeats caused by no-trade row pairs should not be excluded from the Curveball algorithm, or else sampling is no longer guaranteed to be uniform. This is in contradiction with a comment made by Strona *et al.* that removing all repeated states does not affect the Curveball algorithm. In the Supplementary Information of their paper [17], this argument is supported by presenting the transition matrix of a single example where all repeats are removed. However, it is a coincidence that sampling is uniform for this example. We give another example to show that in general sampling may be biased.

To remove repeats caused by no-trade row pairs, step (b) is modified: Instead of randomly selecting any row pair, randomly select a row pair that can make trades. The transition probability for distinct neighboring states A and B then

$$P_{AB} = \frac{1}{p_{tr}(A)} \frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}| + |A_{j-i}|)! - |A_{i-j}|!|A_{j-i}|!}$$

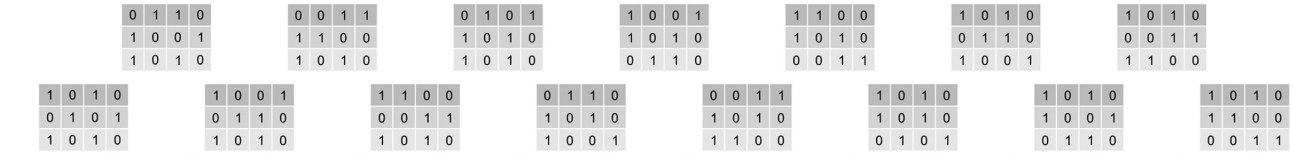


(a) Transition probabilities, no repeated states



(b) Sample histogram

FIG. 5. (a) When all repeated states are excluded from the Curveball algorithm, P_{AB} is not always equal to P_{BA}. In this example, both A_{1-2} and A_{2-1} contain two elements, that is $|A_{1-2}| = |A_{2-1}| = 2$. There are two row pairs in A that can make trades, resulting in $P_{AB} = \frac{1}{10}$. This does not equal P_{BA} since there are three row pairs in B that can make trades and hence $P_{BA} = \frac{1}{15}$. (b) The number of each of the 15 possible binary matrices with row sums (2,2,2) and column sums (2,1,2,1) in a biased sample, generated by the modified Curveball algorithm where all repeated states are excluded. The sample consists of 10 000 matrices sampled at every 1000th trade of the Markov chain.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
A	0	0.1	0.1	0.1	0.1	0.1	0	0	0	0	0.1	0.1	0.1	0.1	0.1
B	0.067	0	0.067	0.067	0.067	0.067	0.33	0	0	0	0	0	0.33	0	0
C	0.067	0.067	0	0.067	0.067	0.067	0	0.33	0	0	0	0.33	0	0	0
D	0.067	0.067	0.067	0	0.067	0.067	0	0	0.33	0	0	0	0	0	0.33
E	0.067	0.067	0.067	0.067	0	0.067	0	0	0	0.33	0	0	0	0.33	0
F	0.1	0.1	0.1	0.1	0.1	0	0.1	0.1	0.1	0.1	0.1	0	0	0	0
G	0	0.33	0	0	0	0.067	0	0.067	0.067	0.067	0.067	0.33	0	0	0
H	0	0	0.33	0	0	0.067	0.067	0	0.067	0.067	0.067	0	0.33	0	0
I	0	0	0	0.33	0	0.067	0.067	0.067	0	0.067	0.067	0	0	0.33	0
J	0	0	0	0	0.33	0.067	0.067	0.067	0.067	0	0.067	0	0	0	0.33
K	0.1	0	0	0	0	0.1	0.1	0.1	0.1	0.1	0	0.1	0.1	0.1	0.1
L	0.067	0	0.33	0	0	0	0.33	0	0	0	0.067	0	0.067	0.067	0.067
M	0.067	0.33	0	0	0	0	0	0.33	0	0	0.067	0.067	0	0.067	0.067
N	0.067	0	0	0	0.33	0	0	0	0.33	0	0.067	0.067	0.067	0	0.067
O	0.067	0	0	0.33	0	0	0	0	0	0.33	0.067	0.067	0.067	0.067	0

FIG. 6. The transition probabilities of the Curveball algorithm without repeated states for all 15 matrices with row sums (2,2,2) and column sums (2,1,2,1).

becomes

$$P_{AB} = \frac{1}{p_{tr}(A)} \frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}| + |A_{j-i}|)! - |A_{i-j}|!|A_{j-i}|!}$$

where $p_{\mathrm{tr}}(A)$ is the number of row pairs in A that can make trades. In this Markov chain, P_{AB} is no longer guaranteed to equal P_{BA}. See Fig. 5(b) for an example. Figure 6 shows all binary matrices with row sums (2,2,2) and column sums (2,1,2,1) and the transition probabilities for the Curveball algorithm without repeated states (i.e., without no-trade row pairs and no-trade shuffles). The Markov chain corresponding to this example is irreducible and aperiodic as can be checked from its transition matrix in Fig. 6. Furthermore, one can verify that the detailed balance equations hold for $\boldsymbol{\pi} = \frac{1}{42}(2,3,3,3,3,2,3,3,3,3,2,3,3,3,3)$. Thus, the Markov chain converges to $\boldsymbol{\pi}$ and not to the uniform distribution. It is less likely to generate matrices A, F, and K than the other matrices as shown experimentally in Fig. 5(b).

IV. NUMERICAL RESULTS

The mixing time $\tau(\epsilon)$ of a Markov chain quantifies the number of steps needed for the chain to get close to its stationary distribution [18]. Generally, it is intractable to compute the exact mixing time for the Curveball algorithm. However, it can be performed for small examples where it is possible to enumerate all binary matrices with given row sums and column sums. The mixing time by itself is not enough to determine the run time of a Markov chain. The time it takes to execute n steps in the chain needs to be taken into account too; we refer to this as the step run time $t_s(n)$.

In Table I the mixing time and step run time of the Curveball and good-shuffle Curveball algorithms are listed for several small matrices. The good-shuffle Curveball algorithm generally mixes faster, which can be explained by the exclusion of some of the repeated states. However, there is a trade-off in terms of step run time since step (d) is more complicated for the good-shuffle algorithm. Indeed, Table I clearly shows that the step run time is longer for the modified algorithm than for the original algorithm. Overall, the good-shuffle Curveball algorithm almost always outperforms the original Curveball algorithm.

Most matrices of interest are much larger than the above examples. Besides, if all matrices with given row sums and column sums can be enumerated, there is no need to use a Markov chain; sampling can be performed directly. For larger matrices, the mixing time needs to be estimated. This can be performed by measuring the perturbation of each matrix in the Markov chain with respect to the initial matrix [17]. The mixing time is approximated by the step at which the perturbation score stabilizes.

Figure 7 shows that for a 10×10 matrix, the perturbation score of the Curveball and good-shuffle Curveball algorithms stabilizes at roughly the same time. For a 100×100 matrix the perturbation scores are indistinguishable. This suggests that for larger matrices, it takes roughly the same number of steps to reach the uniform distribution for both algorithms. There is a good explanation for this: The sizes of sets A_{i-j}

TABLE I. Mixing times and run times of the Curveball and good-shuffle Curveball algorithms for some small matrices. Here $\epsilon = 10^{-6}$ and $n = 1000$. The total run times are approximate, computed from the mixing and step times. Specifically $t_s[\tau(\epsilon)] = \tau(\epsilon)\frac{t_s(n)}{n}$.

			Curveball			Good shuffle		
Row sums	Column sums	State count	Mixing $\tau(\epsilon)$	Step $t_s(n)$	Total $t_s[\tau(\epsilon)]$	Mixing $\tau(\epsilon)$	Step $t_s(n)$	Total $t_s[\tau(\epsilon)]$
121	121	5	33	0.668	0.022	19	0.888	0.017
313	2221	7	74	0.520	0.038	54	0.628	0.034
2211	1212	34	48	0.704	0.034	30	0.963	0.029
2222	1142	12	35	0.734	0.026	13	1.104	0.014
11422	23212	198	101	0.666	0.067	75	0.874	0.066
21021	31011	18	95	0.528	0.050	59	0.655	0.039
22032	22311	120	99	0.539	0.053	77	0.710	0.055
31243	24232	237	92	0.647	0.059	64	0.855	0.055
32111	11321	141	95	0.821	0.082	69	0.875	0.060

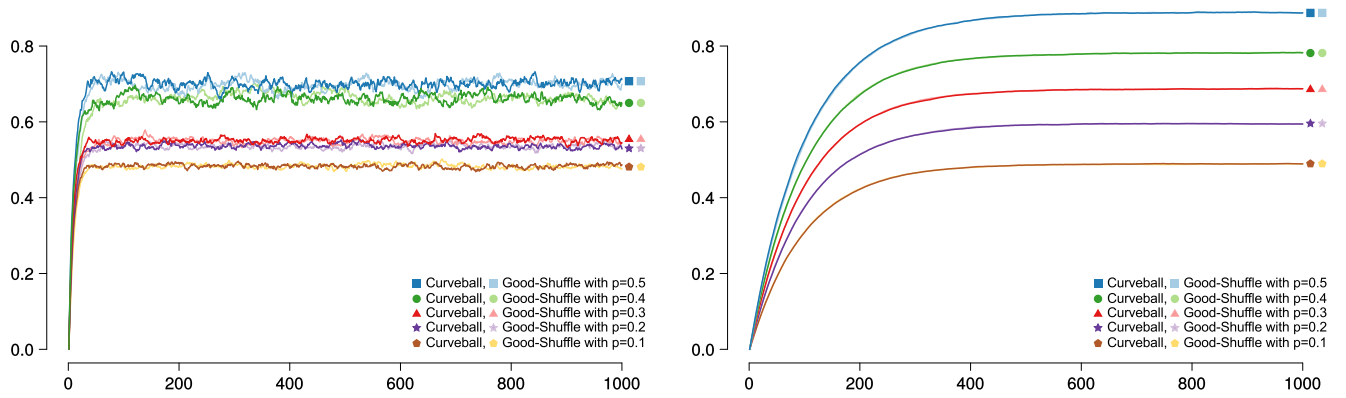


FIG. 7. (Color online) Due to the large variance in perturbation scores for different runs of each Markov chain, average perturbation scores over 100 runs are shown. (a) Five random 10×10 binary matrices were created by letting each matrix entry be one with probability 0.1, 0.2, 0.3, 0.4, or 0.5. For each matrix, the perturbation scores of the Curveball and good-shuffle Curveball algorithm stabilize at roughly the same point after about 50 steps. (b) The same experiment was repeated for 100×100 matrices, and here the perturbation scores of the Curveball and good-shuffle Curveball algorithm are indistinguishable.

and A_{j-i} are larger for larger matrices. For large $|A_{i-j}|$ and $|A_{j-i}|$, the difference between transition probabilities,

$$P_{AB}^{GS} - P_{AB}^{CB} = \frac{2}{n(n-1)}\left(\frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}|+|A_{j-i}|)! - (|A_{i-j}|!|A_{j-i}|!)} - \frac{|A_{i-j}|!|A_{j-i}|!}{(|A_{i-j}|+|A_{j-i}|)!}\right)$$

becomes negligible since $|A_{i-j}|!|A_{j-i}|! \ll (|A_{i-j}| + |A_{j-i}|)!$.

The step run time of the good-shuffle Curveball algorithm is still longer than that of the Curveball algorithm. Thus, the Curveball algorithm runs faster than the good-shuffle Curveball algorithm.

V. CONCLUSION

The Curveball algorithm is a fast algorithm for the randomization of binary matrices with fixed row sums and column sums. In this paper we proved that it generates truly unbiased samples. This is a crucial property since random matrices are usually used as a null hypothesis. Our proof gives a theoretical justification for using the Curveball algorithm instead of the more familiar switching method. Both sample uniformly, but the Curveball algorithm is much faster [17].

We investigated the effect of excluding repeated states from the Curveball algorithm as this was claimed not to affect the unbiased sampling and could potentially improve the speed of the algorithm. We found that out of two types of repeated states, only one can be excluded without introducing bias in the sampling distribution. However, excluding these states only resulted in performance gains for very small matrices (fewer than ten rows and columns). In fact, the increased complexity of each step in the modified algorithm caused it to run slower than the original Curveball algorithm for larger matrices.

We recommend the use of the Curveball algorithm for the randomization of binary matrices. It produces unbiased samples as does the switching method but runs much faster. Furthermore, it is best to leave the Curveball algorithm as it is with inclusion of *all* repeated states.

ACKNOWLEDGMENTS

I would like to thank Professor K. J. Horadam for her support, help, and encouragement. I would like to thank Prof. L. Stone for introducing me to the Curveball algorithm and for fruitful discussions. Finally, I would like to thank Dr. G. Strona for encouraging discussions on the Curveball algorithm. This work was supported by the Commonwealth of Australia Department of Defence under Research Agreement No. 4500743680.

[1] A. Zaman and D. Simberloff, Random binary matrices in biogeographical ecology–Instituting a good neighbor policy, Environ, Ecol. Stat. **9**, 405 (2002).

[2] L. Stone and A. Roberts, The checkerboard score and species distributions, Oecologia **85**, 74 (1990).

[3] I. Miklós and J. Podani, Randomization of presence-absence matrices: Comments and new algorithms, Ecology **85**, 86 (2004).

[4] S. Maslov and K. Sneppen, Specificity and stability in topology of protein networks, Science **296**, 910 (2002).

[5] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, Network motifs: Simple building blocks of complex networks, Science **298**, 824 (2002).

[6] T. A. B. Snijders, Enumeration and simulation methods for 0–1 matrices with given marginals, Psychometrika **56**, 397 (1991).

[7] M. Molloy and B. Reed, A critical point for random graphs with a given degree sequence, Random Struct. Algorithms **6**, 161 (1995).

[8] M. E. J. Newman, S. H. Strogatz, and D. J. Watts, Random graphs with arbitrary degree distributions and their applications, Phys. Rev. E **64**, 026118 (2001).

[9] Y. Artzy-Randrup and L. Stone, Generating uniformly distributed random networks, Phys. Rev. E **72**, 056708 (2005).

[10] C. J. Carstens, A uniform random graph model for directed acyclic networks and its effect on motif-finding, J. Complex Networks **2**, 419 (2014).

[11] O. D. King, Comment on "Subgraphs in random networks", Phys. Rev. E **70**, 058101 (2004).

[12] Also known as rewiring methods, switching methods, and trade methods.

[13] A. R. Rao, R. Jana, and S. Bandyopadhyay, A Markov chain Monte Carlo method for generating random (0,1)-matrices with given marginals, Sankhyä: The Indian Journal of Statistics, Ser. A **58**, 225 (1996).

[14] R. Milo, N. Kashtan, S. Itzkovitz, M. E. J. Newman, and U. Alon, On the uniform generation of random graphs with prescribed degree sequences, arXiv:cond-mat/0312028.

[15] H. Klein-Hennig and A. K. Hartmann, Bias in generation of random graphs, Phys. Rev. E **85**, 026101 (2012).

[16] From now on we refer to the "switch and hold" method from Ref. [9] as "the switching method".

[17] G. Strona, D. Nappo, F. Boccacci, S. Fattorini, and J. San-Miguel-Ayanz, A fast and unbiased procedure to randomize ecological binary matrices with fixed row and column totals, Nat. Commun. **5**, 4114 (2014).

[18] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis* (Cambridge University Press, Cambridge, New York, 2005).

[19] Depending on the dimensions of the matrix it may be faster to make a list of the columns, and depending on the number of zeros and ones in the matrix it may be faster to make lists of the zeros.

[20] This explicit description of step (d) is based on the implementation of the Curveball algorithm as can be downloaded from http://www.nature.com/ncomms/2014/140611/ncomms5114/extref/ncomms5114-s6.txt.

[21] H. J. Ryser, *Combinatorial Mathematics*, Carus Mathematical Monographs Vol. 14 (The Mathematical Association of America, Buffalo, NY, 1963).

[22] These formulas for the transition probabilities are not in correspondence with those presented in Ref. [17] for the small example of matrices with row sums and column sums equal to (1,2,1). However, these formulas correspond to the algorithm as found in the Supplementary Material code of Ref. [17]. The probabilities presented in Ref. [17] correspond to the good-shuffle Curveball algorithm discussed in Sec. III A.

[23] An implementation of the good-shuffle Curveball algorithm can be found at http://github.com/queenBNE/Curveball.