Erratum: Proof of uniform sampling of binary matrices with fixed row sums and column sums for the fast Curveball algorithm [Phys. Rev. E **91**, 042812 (2015)]

C. J. Carstens

(Received 2 September 2016; published 27 September 2016)

DOI: [10.1103/PhysRevE.94.039902](https://doi.org/10.1103/PhysRevE.94.039902)

In our paper, we stated that the Curveball algorithm was first introduced in Ref. [1]. It has since been brought to our attention that several closely related algorithms were previously introduced in Ref. [2]. In particular, the key change from trades of size one (switches) to larger trades was independently discovered by Verhelst [2]. Furthermore, the stationary distributions of the algorithms introduced by Verhelst have also been analyzed. We were unaware of this paper prior to our publication and, hence, regrettably have not cited this work.

In Ref. [1] the Curveball algorithm is introduced using a card trading game analogy. The well-known switching chain then corresponds to a game where at each step a random pair of children (rows) exchanges a single card (columns). The Curveball algorithm corresponds to a game where a random pair of children exchanges a number of their differing cards. This intuitive explanation of the Curveball algorithm is unique to Ref. [1] and makes the Curveball algorithm more accessible than the algorithms discussed in Ref. [2].

The first three algorithms (Algorithms 1–3) introduced in Ref. [2] sample binary matrices with fixed row and column sums. Another algorithm for sampling binary matrices with fixed row and column sums is discussed but not named. We now briefly outline the workings of these three algorithms but using terminology from our own paper. Note that the algorithms in Ref. [2] apply trades to pairs of *columns* instead of rows. We discuss the algorithms in terms of trades between pairs of rows, which is equivalent to randomizing the transposed matrix.

Let t_{ij} be the number of possible trades of size at least one, between rows i and j . Furthermore let $k_2(A)$ be the number of row pairs (i, j) in A for which t_{ij} is at least one, in other words, the number of row pairs for which there exists a trade that alters the matrix A .

Algorithm 1: Uniform sampling from the binomial neighborhood of A .

- (1) Draw a pair of rows, (i, j) , with sampling probability proportional to t_{ij} .
- (2) Apply a random trade to the selected rows. If the trade is of size zero, then repeat step 2.

Algorithm 2: Importance sampling from the binomial neighborhood of A .

- (1) Randomly select a pair of rows, (i, j) , for which $t_{ij} > 0$.
- (2) Apply a random trade to the selected rows. If the trade is of size zero, then repeat step 2.

Algorithm 3: Importance sampling and Metropolis-Hastings

- (1) Randomly select a pair of rows, (i, j) , for which $t_{ij} > 0$.
- (2) Apply a random trade to the selected rows, yielding B .
 - (a) If $B = A$, then repeat step 2.
 - (b) If $k_2(B) \leq k_2(A)$, then the new state is B .
 - (c) If $k_2(B) > k_2(A)$, then the new state is A with probability $1 - k_2(A)/k_2(B)$, otherwise the new state is B .

Verhelst shows that the Markov chains corresponding to Algorithms 1–3 converge to a stationary distribution. Here he assumes that the chains are aperiodic; this is true for *almost* all matrices (see Lemma III.1 of our paper). For Algorithms 1 and 2 the stationary distribution is not uniform. For Algorithm 1 the probability of sampling a matrix is proportional to the size of its neighbourhood in the state graph. For Algorithm 2 the probability of sampling a matrix is proportional to the number of row pairs that allow trades.

Algorithm 3 samples uniformly; it forces the detailed balance equation to hold, i.e., $P(A, B) = P(B, A)$, using the Metropolis-Hastings approach to alter Algorithm 2. That is, let $P(A, B)$ be the transition probabilities of Algorithm 2, then Algorithm 3 is the same algorithm but a trade from matrix A to B is accepted with probability: 1 if $P(A, B) < P(B, A)$ and $P(B, A)/P(A, B)$ if $P(A, B) > P(B, A)$. Notice that this requires the computation of $k_2(B)$ at each step.

Finally, the fourth algorithm that is mentioned in Ref. [2], but not named, corresponds to the good-shuffle Curveball algorithm from our paper. Verhelst states that this algorithm converges to the uniform distribution but does not prove it. Instead, he introduces Algorithm 3 and claims it is faster. It is likely the case that Algorithm 3 has a faster mixing time than the good-shuffle Curveball algorithm, because the probability of repeating states is smaller. However, there are currently no theoretical results about the mixing time for this class of algorithms, and, hence, we cannot state this with certainty.

On the other hand, the additional complexity of a single step of Algorithm 3 as compared to the Curveball algorithm affects the overall run time. Specifically, in Algorithm 3 the value of $k_2(B)$ for the new state B has to be computed (or updated from the old state A) in every step. It is possible that, in practice, the run time of Algorithm 3 is slower than that of the Curveball

algorithm. An example of an algorithm with faster mixing time, yet slower run time, is the good-shuffle Curveball algorithm as compared to the original Curveball algorithm. This was shown in Sec. IV of our paper.

-
- [1] 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 \(2014\)](#).
- [2] N. D. Verhelst, An efficient MCMC algorithm to sample binary matrices with fixed marginals, [Psychometrika **73**, 705 \(2008\)](#).