

# REPORTS

*Ecology*, 85(1), 2004, pp. 86–92  
© 2004 by the Ecological Society of America

## RANDOMIZATION OF PRESENCE–ABSENCE MATRICES: COMMENTS AND NEW ALGORITHMS

I. MIKLÓS<sup>1</sup> AND J. PODANI<sup>2</sup>

*Department of Plant Taxonomy and Ecology, Eötvös University, Pázmány P. s. 1/c, Budapest, H-1117, Hungary*

**Abstract.** Randomization of presence–absence data matrices with fixed row and column totals is an important tool in ecological research wherever the significance of data-based statistics (e.g., species association measures) is to be evaluated. In the current literature of numerical ecology, however, there has been no algorithm that randomizes moderately large matrices in short time such that equidistribution of results is guaranteed. We show that a simple modification of the swap algorithm, called here the “trial-swap method,” satisfies the requirement for equidistribution. Since this is relatively slow, we suggest two fast algorithms that, combined with the trial-swap method, produce all possible results with equal chance. The three procedures are illustrated using actual examples taken from bird biogeography and vegetation ecology.

**Key words:** data matrices; null models; presence–absence data; randomization methods; species association; swap methods; Vanuatu data.

### INTRODUCTION

The use of null models has been a widespread practice in contemporary numerical ecology to evaluate the deviation of observed phenomena from random expectations (Gotelli and Graves 1996, Gotelli 2000, Manly and Sanderson 2002). Many null models, for example those referring to community assembly rules, involve the direct randomization of raw data matrices that contain only presence–absence information on species. The success of null-model-based studies greatly depends on two factors: (1) the definition of the constraints that affect randomization and (2) the algorithmic implementation of the randomization method. The first problem is rather practical: the ecologist decides on some basic features of the matrix to be retained in the computations. For example, the data may be reshuffled without any constraints, with preserving the column or row totals, or both. Each of these possibilities may be useful on its own, but it has been suggested that the last strategy is to be preferred in most situations, especially in the analysis of species co-occurrence patterns (Connor and Simberloff 1979). This choice is immediately related to the second factor, the algorithmic implementation, because complete randomization is the most difficult to achieve if both row and column totals are to be preserved. Gotelli and Ent-

sminger (2001) have recently compared and evaluated the strategies currently available for this purpose. They found that algorithms could be divided into two major groups, the *swap* and *random-fill* procedures. The swap methods take a start from the original matrix and involve random selections of  $2 \times 2$  submatrices containing 1's in the main diagonal and 0's outside, or vice versa (so-called “checkerboard units,” Stone and Roberts 1990). Swapping the 0's and 1's produces a new matrix such that the row and column totals remain unchanged, and this new matrix can be subjected to another random swap, and so on. Procedures in the other group start with a matrix containing only zeros and replace in each step one 0 by 1 randomly, provided that the corresponding row and column totals do not exceed the respective totals in the observed data matrix. The difficulty with the random-fill algorithms is that the process may be trapped if addition of a new 1 is impossible without exceeding some of the totals. Such cases are resolved by retreating one or more steps in order to find a different route that does not run into such dead ends (see Sanderson et al. [1998] and Gotelli and Entsminger [2001] for variants of the backtracking technique).

For both families of algorithms, a further and fundamental requirement is equal probability of the resulting matrices. This ensures that the null model is truly neutral, i.e., there is no preference for any particular matrix pattern. If equal probability is in doubt, then the distribution of any statistic estimated through this randomization will be biased and in fact no sig-

Manuscript received 11 February 2003; revised 9 May 2003; accepted 20 May 2003. Corresponding Editor: A. M. Ellison.

<sup>1</sup> Present address: Department of Statistics, University of Oxford, 1 South Parks Road, Oxford OX1 3TG, UK.

<sup>2</sup> Corresponding author. E-mail: podani@ludens.elte.hu

TABLE 1. All possible  $3 \times 3$  presence-absence matrices with  $[0,1,0]$  as both column and row totals.

Matrix A	Matrix B	Matrix C	Matrix D	Matrix E
0 1 0	0 1 0	1 0 0	0 1 0	0 0 1
1 0 1	1 1 0	0 1 1	0 1 1	1 1 0
0 1 0	0 0 1	0 1 0	1 0 0	0 1 0

nificance testing can be justified. Current swap algorithms, if started from the original matrix, produce biased distributions due to dependence on the initial configuration for small numbers of steps. Even if the number of iterations is set to be so large that this dependence is diminished, the distribution reached is usually not of equal probabilities (Gotelli and Entsminger 2001, Zaman and Simberloff 2002). Moreover, there is no method yet to define the number of steps necessary to reach a start-independent random version of the observed matrix (even though there is an algorithmic route from any matrix to any other, so theoretically the full set of matrices can be produced by repeated swapping, see Brualdi 1980 and Appendix C). Gotelli and Entsminger (2001) concluded that the swapping methods appeared more efficient and reliable statistically than random-fill procedures. Recently, Zaman and Simberloff (2002) also recognized that current swap methods converge to a biased distribution, and they introduced a weighting method resembling the popular “important sampling” technique (Liu 2001) that corrects for this bias.

We show in this paper that equidistribution of all possible matrices in swap algorithms depends on how we understand an algorithmic step—whether the number of trials or the number of successful attempts is specified at the outset. We recommend that the number of trials—rather than the number of successes—should be specified in advance, and proofs are given to support this choice. Furthermore, two new randomization algorithms are introduced to produce start-independent random versions in reasonably short time. Since these fast procedures do not guarantee equidistribution of resulting matrices, we suggest the use of the trial swap algorithm to modify the results further.

SWAPPING WITH A CONSTANT NUMBER OF TRIALS

Gotelli and Entsminger (2001) distinguished two basic variants of swapping algorithms and found that neither the sequential-swap nor the independent-swap sample the possible presence-absence matrices uniformly. Here we introduce the *trial-swap* algorithm, a slight modification of the above algorithms, which converges to the uniform distribution.

In the independent- and sequential-swap algorithms, a step involves finding randomly a  $2 \times 2$  submatrix that can be swapped, no matter how many submatrices were found in the meantime that could not be swapped. In the new algorithm, a step is understood as the selection of any  $2 \times 2$  submatrix, i.e., the total number

of steps will be the number of submatrices examined and the number of swaps actually performed is therefore immaterial. Since only the

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

submatrices can be swapped, examination of other types of submatrices leaves the initial matrix unchanged.

We introduce the trial-swap process on  $3 \times 3$  matrices whose row and column totals are both fixed to (1,2,1), the same example used by Gotelli and Entsminger (2001). There are five possible matrices with such totals (Table 1). The transition (or rate) matrix of the independent- and sequential-swap algorithms is presented in Table 2a. Eigenanalysis of this matrix reveals that both swapping processes converge to a biased distribution, with matrix A oversampled and all others undersampled (Gotelli and Entsminger 2001, Zaman and Simberloff 2002). Note that diagonal elements are all zero, since each step by definition changes the matrix configuration. However, if we do not insist that a swap must be made in *all steps* of the process, that is, if we count the number of selected submatrices (trial swap) rather than the number of successful choices, the transition matrix will be the one given in Table 2b. The main difference between the two transition matrices is that the second is symmetric, with non-zero diagonal elements. The eigenvector solution gives the expected rate of different matrices produced in many attempts, which is the uniform distribution of matrices A-E. Both empirical and theoretical results show that only 10 trials are sufficient to get a nearly uniform distribution, described by the 10th power of the transition matrix (Table 3).

TABLE 2. Transition matrices for the five sample matrices presented in Table 1: (a) transition probabilities for the independent-swap and sequential-swap algorithms (Gotelli and Entsminger 2001); (b) transition probabilities for the trial-swap algorithm presented here.

a		Initial matrix				
Swapped matrix	A	B	C	D	E	
A	0	0.33	0.33	0.33	0.33	
B	0.25	0	0	0.33	0.33	
C	0.25	0	0	0.33	0.33	
D	0.25	0.33	0.33	0	0	
E	0.25	0.33	0.33	0	0	

b		Initial matrix				
Swapped matrix	A	B	C	D	E	
A	5/9	1/9	1/9	1/9	1/9	
B	1/9	6/9	0	1/9	1/9	
C	1/9	0	6/9	1/9	1/9	
D	1/9	1/9	1/9	6/9	0	
E	1/9	1/9	1/9	0	6/9	

TABLE 3. The 10th power of the transition matrix of Table 2b, illustrating that fair equidistribution of matrices is reached after as few as 10 iterations by the trial-swap method.

Output matrix	Initial matrix				
	A	B	C	D	E
A	0.200241	0.199940	0.199940	0.199940	0.199940
B	0.199940	0.208761	0.191419	0.199940	0.199940
C	0.199940	0.191419	0.208761	0.199940	0.199940
D	0.199940	0.199940	0.199940	0.208761	0.191419
E	0.199940	0.199940	0.199940	0.191419	0.208761

It might be surprising that such a slight modification is so influential, but we can give a heuristic explanation why trial swapping yields uniform distribution. There are four submatrices that can be swapped in matrix A, while the other four matrices contain only three such submatrices each. As a consequence, if we consider the trial-swap approach as a discrete-time Markov model, then there is a greater chance for leaving matrix A than leaving the others, hence, the trial-swap process spends less time in matrix A than the independent- and sequential-swap processes. That is, the trial-swap algorithm samples matrix A less frequently than the other two swap processes do. Indeed, matrix A is oversampled by the independent- and sequential-swap algorithms (Gotelli and Entsminger 2001).

The introduced case is not a “lucky accident”; the trial-swap algorithm yields uniform distribution, as shown by Kannan et al. (1999). We prove in Appendix A that the trial-swap algorithm is a Metropolis-type Markov-chain Monte Carlo process (MCMC, Metropolis et al. 1953; see Liu [2001] for a recent account on the subject), and thus it indeed leads to uniform distribution, irrespective of the row and column totals, except the trivial case with (1,1) and (1,1).

#### FAST RANDOMIZATION ALGORITHMS

The trial swap is proven to converge to the uniform distribution, but our experiments suggest that convergence is slow for large matrices. Although we do not have analytical results for expressing the efficiency of the trial-swap algorithm, Goldberg and Jerrum (2002) showed that a shuffling process could be exponentially slow even if every state can be reached from every other with a single transition. Kannan et al. (1999) investigated the mixing time of the swapping algorithm and proved that in some cases mixing time is short but, when the variance of the column and row totals is large, mixing may take longer. Slow convergence is certainly an obstacle to a widespread application of this algorithm to actual ecological problems.

Now we introduce two new algorithms that produce a randomized matrix much faster than the swap methods. We note in advance, however, that—unlike the trial-swap procedure—neither of them is proven to lead to the uniform distribution. Thus, we shall recommend a combined approach. The fast algorithms could be used first to create a random matrix that is independent

of the starting matrix. Then, the newly obtained matrix is subjected to perturbations by the trial-swap algorithm, which ensures equidistribution of randomized variants of the original matrix.

#### Algorithm 1: sum-of-squares reduction

For the sum-of-squares reduction algorithm, we shall use the following *definition*: The so-called *quasi-swap* replaces the sub-matrix

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \text{ by } \begin{pmatrix} a-1 & b+1 \\ c+1 & d-1 \end{pmatrix}$$

if  $a + d - b - c - 2 \geq 0$ , or by

$$\begin{pmatrix} a+1 & b-1 \\ c-1 & d+1 \end{pmatrix}$$

if  $b + c - a - d - 2 \geq 0$ .

In other words, the condition of replacement is that the sum of squares of the entire matrix does not increase. Note that  $a$ ,  $b$ ,  $c$ , and  $d$  might be any nonnegative integer number. For example, the matrix

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

can be transformed into

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

using a quasi-swap.

This process starts with a random matrix whose elements are nonnegative integers with the condition that its row and column totals agree with the respective totals of the original matrix. This matrix is achieved by starting with an all-0 matrix and then subsequently increasing randomly chosen entities. The randomization involves the use of quasi-swaps until a matrix containing only 0's and 1's is reached. The process runs in stochastic time, i.e., there is no fixed-time complexity of the algorithm. On a fast personal computer, for example, it can generate 1000 random matrices of size  $118 \times 80$  in an hour. In principle, sum-of-squares reduction can generate every possible matrix with given row and column totals, and we proved that the process stops with probability 1, namely, the algorithm

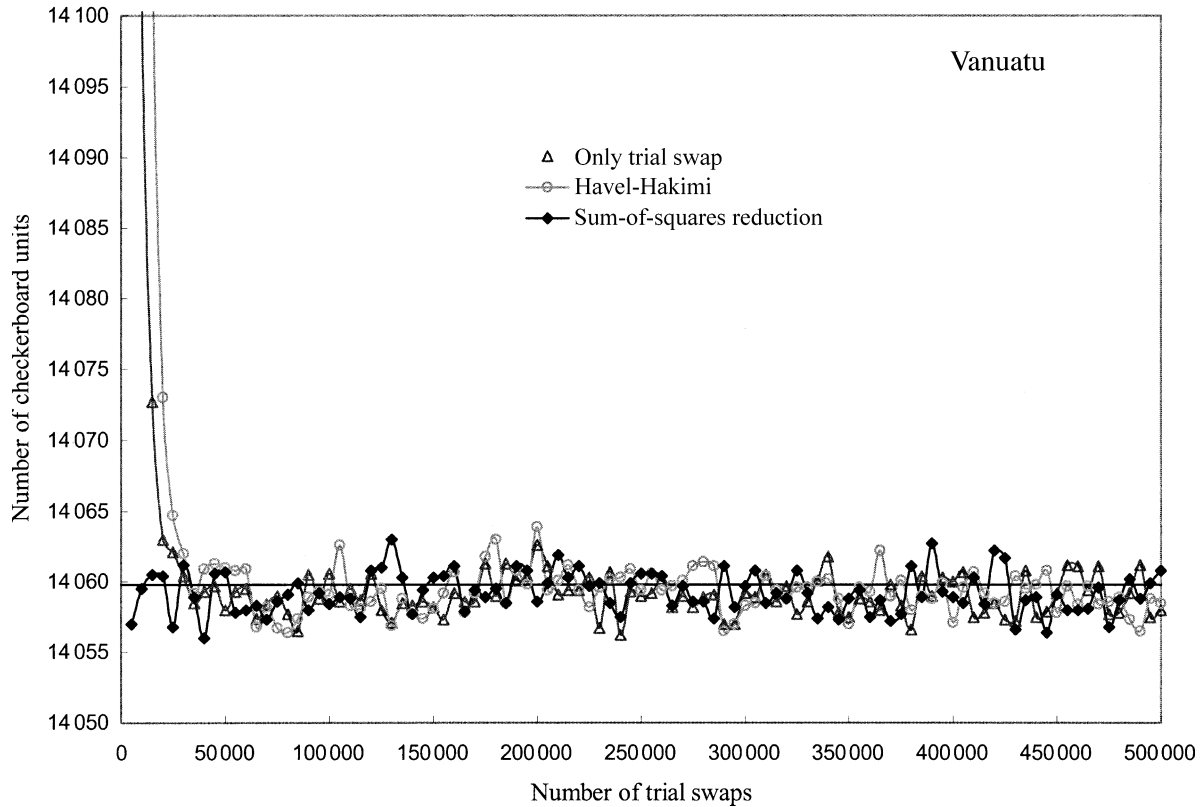


FIG. 1. The expected number of checkerboard units in the function of the number of trial swaps for matrices having the same row and column totals as the Vanuatu matrix. Values were estimated from  $10^4$  runs. The initial matrix was a random matrix obtained with the sum-of-squares reduction method (SR + trial swap) or a random matrix obtained with the Havel-Hakimi method (HH + trial swap) or the original Vanuatu matrix (only trial swap). The x-axis intersects the y-axis at 14 059.8, which is the expected number of checkerboard units estimated with  $10^6$  trial swaps in each of  $10^5$  runs.

does not fall into an infinite cycle (the proof is given in Appendix B).

*Algorithm 2: the Havel-Hakimi process*

The Havel-Hakimi theorem (Havel 1955, Hakimi 1962) states that there exists a matrix  $\mathbf{X}_{n,m}$  of 0's and 1's with row totals  $\mathbf{a}_0 = (a_1, a_2, \dots, a_n)$  and column totals  $\mathbf{b}_0 = (b_1, b_2, \dots, b_m)$  such that  $b_i \geq b_{i+1}$  for every  $0 < i < m$  if and only if another matrix  $\mathbf{X}_{n-1,m}$  of 0's and 1's with row totals  $\mathbf{a}_1 = (a_2, a_3, \dots, a_n)$  and column totals  $\mathbf{b}_1 = (b_1 - 1, b_2 - 1, \dots, b_{a_1} - 1, b_{a_1+1}, \dots, b_m)$  also exists. Using the Havel-Hakimi theorem, we can reduce the problem of the existence of a matrix with given row and column totals to the problem of the existence of a smaller matrix with given row and column totals. Reducing the size of the matrix iteratively leads either to a matrix with some negative numbers among the column totals or to a matrix with only one row and with column totals with a maximum value of 1 in each. The former case indicates that there is no matrix with row totals  $\mathbf{a}_0$  and column totals  $\mathbf{b}_0$ . In the latter case, there is a unique way for constructing a matrix with row totals  $\mathbf{a}_0$  and column totals  $\mathbf{b}_0$ , building

up the larger matrices from the smaller matrices (Appendix C).

The Havel-Hakimi theorem holds for any arbitrary ordering of row totals. Therefore, we have complete freedom to permute the rows, so we can generate several random matrices with given row and column totals. Moreover, we can permute the columns having the same column total, which leads to even more possible random matrices. However, there are certain row and column totals for which this process cannot generate all possible matrices, i.e., no equidistribution is reached under all circumstances.

The Havel-Hakimi process does not generate uniformly distributed matrices; its main advantage lies in its efficiency. This process runs in  $O(nm^2)$  time and is even faster on the average than the sum of squares algorithm: it can generate 10 000 random matrices of size  $118 \times 80$  in an hour. Again, as we show below (see *Empirical results*), the main disadvantage of the Havel-Hakimi process is that the resulting matrix distribution is biased.

*Perturbation by trial swap*

Twice the number of 1's in a presence-absence matrix is an upper bound to the number of swaps needed

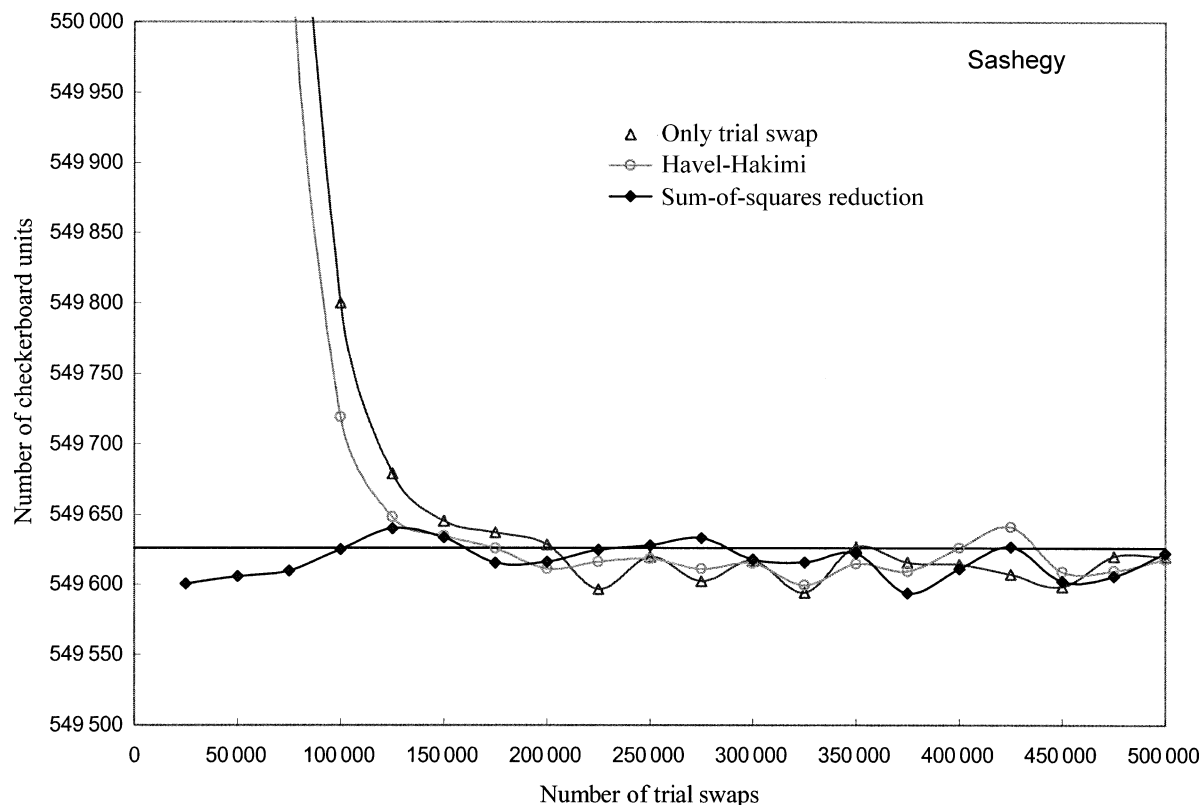


FIG. 2. The expected number of checkerboard units in the function of the number of trial swaps for matrices having the same row and column totals as the Sashegy matrix. Values were estimated from  $10^4$  runs. The initial matrix was a random matrix obtained with the sum-of-squares reduction method (SR + trial swap) or a random matrix obtained with the Havel-Hakimi method (HH + trial swap) or the original Sashegy matrix (only trial swap). The  $x$ -axis intersects the  $y$ -axis at 549 625.9, which is the expected number of checkerboard units estimated with  $10^6$  trial swaps in each of  $10^5$  runs.

for turning this, by trial swaps, into another arbitrary matrix with the same row and column totals (Appendix C). Therefore, we suggest that the number of trials should be set such that the expected number of swaps equals twice the number of 1's in the matrix. Given an initial matrix, both the number of checkerboard units and the number of possible  $2 \times 2$  submatrices can be calculated, and their ratio can be used as estimation for the proportion of the successful trials. The number of trials we suggest is about 35 000 and 100 000 for the Vanuatu and Sashegy data matrices, respectively (see next section).

#### EMPIRICAL RESULTS

All the three methods (trial swaps only, and the two fast random-fill algorithms followed by trial swaps) were investigated on two actual data matrices (see Supplement for program availability). The first one comprises bird presence-absence data for 56 species on 28 islands of the Vanuatu Archipelago, New Hebrides (Gotelli and Entsminger 2001). The Sashegy data matrix contains presence-absence information on 118 species in 80 vegetational quadrats of size  $3 \times 3$  m located in the grassland communities of the Sashegy Nature

Reserve, Budapest, Hungary (see Podani [1985] and Podani et al. [2000] for details). We applied 500 000 trials to the initial matrix (which was either the Vanuatu or the Sashegy data matrix in the first case, and a random-fill matrix obtained with the two methods in the other two cases). The number of checkerboard units (see *Introduction*) was calculated after every 5000 trials. The process was repeated 20 000 times for the Vanuatu data, and 50 000 times for the Sashegy data. The average number of checkerboard units obtained in the function of the number of trial swaps is plotted in Figs. 1 and 2.

Both the Vanuatu and Sashegy data matrices contain more checkerboard units than expected for a random matrix with the same row and column totals, as a manifestation of biogeographical and ecological processes (e.g., competition) affecting species coexistence. When these matrices were used to initialize the trial-swap process, the average number of checkerboard units decreased with the number of trial swaps.

The Vanuatu matrix contains 14 676 checkerboard units. This seems to be an extremely high value and its significance certainly deserves attention. Gotelli and Entsminger (2001) used the sequential-swap method to

examine this problem and concluded that the Vanuatu matrix has a  $P < 0.039$  among the random variants. We generated 10 000 random matrices with 500 000 trial swaps in each and did not find any matrix having more checkerboard units than the Vanuatu matrix, i.e.,  $P < 0.0001$ , implying that there is a much more obvious nonrandom structure in the data than indicated by earlier studies. The evaluation of the Sashegy data set has led to similar conclusions, the number of checkerboard units was 594 373, never exceeded in the simulations.

For both actual data sets used here, the Havel-Hakimi process produces matrices with more checkerboard units than expected randomly. During the process, 1's are always written into columns having the greatest actual column totals. The column totals are recalculated after filling each row, and the columns are reordered according to the new column totals. This reordering does not provide sufficient shuffling to obtain uniform sampling.

The most reliable sampling was obtained when initial matrices were generated with the sum-of-squares reduction method. In this case, we cannot see any deviation from the average. However, we would like to warn the readers that notwithstanding zero deviation the sum-of-squares reduction does not provide uniform distribution theoretically. Therefore, we suggest the use of trial swaps after fast initialization with sum-of-squares reduction.

Using  $10^6$  trial swaps in each of 100 000 runs, we estimated the expected number of checkerboard units for the Vanuatu data to be  $14\,059.8 \pm 0.43$  (mean  $\pm$  1 SD) and for the Sashegy data to be  $549\,625.9 \pm 4.61$ . This estimation accords with the estimates obtained using only  $10^5$  trial swaps, therefore we conclude that 500 000 trial swaps are surely enough for reaching the equilibrium distribution.

#### CONCLUDING REMARKS

The main message of the present paper is that randomization of presence-absence data matrices, constrained by the row and column totals, is best achieved through a two-step procedure. The first step ensures independence from the starting matrix and involves the use of a new, sum-of-squares reduction algorithm. The Havel-Hakimi process seems also applicable to this purpose, but our simulation results suggest that this algorithm is less consistent than the sum-of-squares-based method. Independence is not associated with equal probability, however. Therefore, fast initialization must be followed by shuffling algorithms for which we propose a slight modification of the swap method, the trial swap. On theoretical grounds we recommend that the number of trials should set run time such that the expected number of swaps actually performed is twice the number of 1's in the matrix.

We proved that the trial-swap process converges to the uniform distribution, while the independent- and sequential-swap methods are biased since they tend to

produce matrices having more checkerboard units than the average. Therefore, analyses that are based on sequential or independent swap (Gotelli and McCabe 2002, Gotelli and Entsminger 2003) represent conservative tests for competitive structuring. We investigated two ecological data sets with the trial-swap algorithm. The Vanuatu matrix has been investigated in several papers, with different conclusions (Roberts and Stone 1990, Sanderson et al. 1998, Gotelli and Entsminger 2001, Zaman and Simberloff 2002). Here we found that the Vanuatu matrix is significantly nonrandom, having a probability of  $P < 0.0001$ . This is in accordance with the method of Zaman and Simberloff (2002), which is also proven to provide unbiased estimation for the  $P$  value. The independent- and sequential-swap algorithms (Roberts and Stone 1990, Gotelli and Entsminger 2001) give a similar result; however, they are slightly biased. The worst performance is provided by the knight's-tour algorithm (Sanderson et al. 1998), which is a variant of the random-fill algorithms.

#### ACKNOWLEDGMENTS

I. Miklós started this work at Collegium Budapest, which he wants to thank for its generous support. He was also supported by the Engineering and Physical Sciences Research Council and the Medical Research Council, under contracts HAMJW and HAMKA, respectively. J. Podani is grateful to the Hungarian National Research Grant (OTKA No. T43732) for financial support. The Vanuatu data were kindly placed at our disposal by Dr. N. Gotelli. We are grateful to Richard A. Brualdi for reading the manuscript and for useful comments and suggestions. Thanks are due to I. Scheuring for his help in computations, to N. Gotelli and two anonymous referees for comments, and to J. Garay and A. Gyárfás for discussions.

#### LITERATURE CITED

- Brualdi, R. A. 1980. Matrices of zeroes and ones with fixed row and column sum vectors. *Linear Algebra and its Applications* **33**:159–231.
- Connor, E. F., and D. Simberloff. 1979. The assembly of species communities: chance or competition? *Ecology* **60**:1132–1140.
- Goldberg, L. A., and M. Jerrum. 2002. The “Burnside process” converges slowly. *Combinatorics, Probability & Computing* **11**:21–34.
- Gotelli, N. 2000. Null model analysis of species co-occurrence patterns. *Ecology* **81**:2606–2621.
- Gotelli, N., and G. L. Entsminger. 2001. Swap and fill algorithms in null model analysis: rethinking the knight's tour. *Oecologia* **129**:281–291.
- Gotelli, N., and G. L. Entsminger. 2003. Swap algorithms in null model analysis. *Ecology* **84**:532–535.
- Gotelli, N., and G. R. Graves. 1996. *Null models in ecology*. Smithsonian Institution, Washington, D.C., USA.
- Gotelli, N., and D. J. McCabe. 2002. Species co-occurrence: a meta-analysis of J. M. Diamond's assembly rules model. *Ecology* **83**:2091–2096.
- Hakimi, S. 1962. On the realizability of a set of integers as degrees of the vertices of a graph. *SIAM Journal of Applied Mathematics* **10**:496–506.
- Havel, V. 1955. A remark on the existence of finite graphs. [In Czech.] *Casopis pro Pestování Matematiky* **80**:477–480.

- Kannan, R., P. Tetali, and S. Vempala. 1999. Simple Markov-chain algorithms for generating bipartite graphs and tournaments. *Random Structures & Algorithms* **14**:293–308.
- Liu, J. S. 2001. Monte Carlo strategies in scientific computing. Springer-Verlag, New York, New York, USA.
- Manly, B., and J. G. Sanderson. 2002. A note on null models: justifying the methodology. *Ecology* **83**:580–582.
- Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. 1953. Equations of state calculations by fast computing machines. *Journal of Chemical Physics* **21**:1087–1091.
- Podani, J. 1985. Syntaxonomic congruence in a small-scale vegetation survey. *Abstracta Botanica* **9**:99–128.
- Podani, J., P. Csontos, and J. Tamás. 2000. Additive trees in the analysis of community data. *Community Ecology* **1**:33–41.
- Roberts, A., and L. Stone. 1990. Island-sharing by archipelago species. *Oecologia* **83**:560–567.
- Sanderson, J. G., M. P. Moulton, and R. G. Selfridge. 1998. Null matrices and the analysis of species co-occurrences. *Oecologia* **116**:275–283.
- Stone, L., and A. Roberts. 1990. The checkerboard score and species distributions. *Oecologia* **85**:74–79.
- Zaman, A., and D. Simberloff. 2002. Random binary matrices in biogeographical ecology—instituting a good neighbor policy. *Environmental and Ecological Statistics* **9**:405–421.

#### APPENDIX A

Proofs showing the relationship of the trial swap to the Metropolis algorithm and the Hastings ratio are available in ESA's Electronic Data Archive: *Ecological Archives* E085-001-A1.

#### APPENDIX B

Proofs that the sum-of-squares reduction never falls into an infinite cycle are given in ESA's Electronic Data Archive: *Ecological Archives* E085-001-A2.

#### APPENDIX C

The Havel-Hakimi theorem is described in ESA's Electronic Data Archive: *Ecological Archives* E085-001-A3.

#### SUPPLEMENT

The C source code containing the algorithms described in this paper, and a short description of the program and its options are available online in ESA's Electronic Data Archive: *Ecological Archives* E085-001-S1.