EXACT SAMPLING AND COUNTING FOR FIXED-MARGIN MATRICES

By Jeffrey W. Miller and Matthew T. Harrison[*]

Brown University

The uniform distribution on matrices with specified row and column sums is often a natural choice of null model when testing for structure in two-way tables (binary or nonnegative integer). Due to the difficulty of sampling from this distribution, many approximate methods have been developed. We will show that by exploiting certain symmetries, exact sampling and counting is in fact possible in many nontrivial real-world cases. We illustrate with real datasets including ecological co-occurrence matrices, social networks, and contingency tables, and we apply our method to finding the Ehrhart polynomials of the Birkhoff polytope. Further, we demonstrate how the method can be used to assess the accuracy of certain approximate samplers, by finding the total variation distance to the uniform distribution.

1. A motivating example. In ecology, co-occurrence tables are used to summarize biogeographical data. For instance, Table 1 indicates the presence/absence of 26 mammalian species in 28 mountain ranges in the American Southwest. When presented with such data, one might wonder: What factors control which species live in which habitats? In 1975, ecologist (and now, renowned author) Jared Diamond stunned the ecology community with the proposal of specific "assembly rules" governing the allocation of species to habitats. Diamond (1975) observed that certain pairs of species tended to occur together, and other pairs tended to be disjoint, suggesting that cooperation and competition play a key role. But did these patterns really reflect species interactions, or were they merely due to random chance?

To address this question, Connor and Simberloff (1979) suggested statistical hypothesis testing. Since some species are simply more prolific than others, and some habitats are larger than others, a sensible choice of null model is the uniform distribution on co-occurrence matrices with the observed numbers of habitats per species (row sums) and species per habitat (column sums). Connor and Simberloff (1979) presented a formidable chal-

[*]Both authors were supported in part by NSF grant DMS-1007593 and DARPA contract FA8650-11-1-715. The first author was also supported by a NDSEG fellowship.

AMS 2000 subject classifications: Primary 62H15; secondary 62H17, 05A15.

Keywords and phrases: exact sampling, exact counting, binary matrix, contingency table, integer points in polyhedra.

TABLE 1
26 mammalian species in 28 mountain ranges (Patterson and Atmar, 1986)

	Habitat																											
Species	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
A	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
B	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
C	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1
D	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1	0	0	0	0
E	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	0	0	0
F	1	1	1	1	1	0	0	0	1	1	1	1	1	0	1	1	0	1	1	1	1	0	1	0	1	0	0	0
G	1	1	1	1	1	1	1	0	1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
H	1	1	1	1	1	1	0	1	1	0	1	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
I	1	1	1	1	1	1	1	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
J	1	1	1	1	1	1	1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
K	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
L	1	1	1	1	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
M	1	1	1	1	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
N	1	1	1	1	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
O	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
P	1	1	0	1	1	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Q	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	1	1	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
T	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Y	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Z	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

lenge to Diamond's theory by showing that under this simple null model, the statistics observed by Diamond could easily have arisen by chance. A contentious debate erupted, yielding an extensive body of research on test statistics and null models for detecting various types of "structure" in ecological matrices. Decades later, the basic null model of Connor and Simberloff has withstood the test of time, and is now a mainstay in the analysis of ecological matrices (see e.g. Ulrich and Gotelli (2007); Gotelli and McCabe (2002), and references therein).

As a concrete example, consider the montane mammals in Table 1. Patterson and Atmar (1986) proposed a model in which, during the most recent glacial period, cold-adapted species inhabited a region spanning several mountain ranges and the low-lying areas between, but in the current (warmer) interglacial period these populations have receded into the mountains and become extinct in some areas. They suggest that this would cause the set of species found in one mountain range to tend to be a subset of those found in another. This led them to consider the following *nested subset statistic*, equal to the number of species-habitat pairs such that the species does not occur in that habitat but does occur in a less-populated habitat:

$$S_{nest} = \sum_{i,j} I(a_{ij} = 0,\, q_j > m_i),$$

where $\mathbf{A} = (a_{ij})$ is a binary matrix with species as rows and habitats as columns (such as Table 1), $q_j = \sum_i a_{ij}$, and $m_i = \min\{q_j : a_{ij} = 1, j = 1, \ldots, n\}$. Here, $I(E)$ is 1 if E is true, and 0 otherwise. (Note: Smaller S_{nest} means more "nestedness".) To perform a hypothesis test using the standard null model described above, one would estimate the p-value for S_{nest} by sampling from the uniform distribution over binary matrices with the observed row and column sums — in this case, (26, 26, 25, 22, 22, 18, 12, 12, 12, 11, 10, 10, 8, 8, 8, 7, 6, 6, 5, 5, 4, 4, 3, 3, 1, 1) and (26, 24, 23, 21, 19, 13, 13, 12, 11, 10, 10, 9, 9, 7, 7, 7, 7, 7, 7, 6, 6, 5, 5, 4, 3, 2, 1, 1), respectively. However, it is difficult to sample exactly from this distribution. Instead, Patterson and Atmar used an approximation in which the entries of each column are drawn proportionally to the row sums, conditioned on the column sum. (The row sums are not constrained in their approximation.) They drew 1000 samples from their approximation, estimated the p-value of S_{nest} to be 9×10^{-20} for Table 1, and concluded that the data does exhibit significantly more nestedness than one would expect under the null. Patterson and Atmar's (1986) article was highly influential, inspiring many subsequent studies into nestedness (see e.g. Ulrich and Gotelli (2007) and references therein).

The preceding scenario is commonplace — the combinatorial problem that arises from constraining the row and column sums makes it difficult to sample exactly from the desired uniform distribution. As a result, on all but the most trivial matrices, researchers have resorted to approximate methods, such as Markov chain Monte Carlo (MCMC), Sequential Importance Sampling, and heuristic approaches such as the one described above. With all these approximate methods, the nagging question remains: Did the use of an approximate distribution significantly affect the result?

In this work, we describe an efficient algorithm for sampling exactly from the uniform distribution over binary or nonnegative integer matrices with given row and column sums (provided that most of the sums are not too large). As a result, Monte Carlo estimates of quantities of interest, such as p-values, can be accompanied by exact confidence intervals (that is, true confidence intervals, rather than intervals based on asymptotic approximations). Further, our algorithm computes the exact number of such matrices.

Table 2
Sample statistics of S_{nest} for montane mammal data (Table 1)

Method	# samples	estimated p-value	mean	std. dev.	min	max
Exact	1,000,000	$0.0322 \pm .00018$	80.7	9.7	44	132
Heuristic	1,000	9×10^{-20}	227.9	18.1	180	287

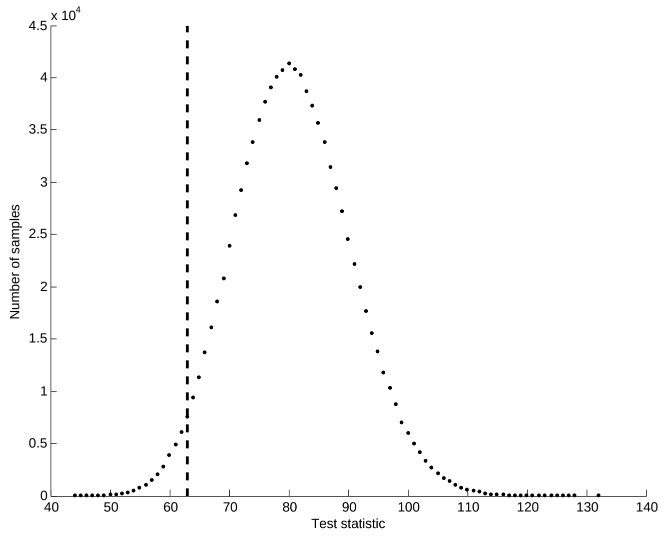

Fig 1: Histogram of S_{nest} for 10^6 exact samples from the uniform distribution over matrices with margins as in Table 1. The dashed line is the value of S_{nest} for Table 1.

In Table 2, we compare the results of Patterson and Atmar with results based on 10^6 exact samples using our algorithm. There are large discrepancies. We estimate the p-value to be $0.0322 \pm .00018$ ($\hat{p} \pm \hat{se}$), with an exact 95% confidence interval of $[0.0318, 0.0326]$ (based on the binomial c.d.f., not on $\hat{se}$). Their estimate of 9×10^{-20} is far smaller — in fact, the value of the test statistic on the observed matrix, $S_{nest} = 63$, is considerably less than the smallest value among all of their 1000 samples, $S_{nest} = 180$. (Note: It appears that they used a normal approximation to the distribution of S_{nest} to estimate the p-value.) Meanwhile, in view of the histogram of exact samples in Figure 1, the observed matrix appears relatively typical! Recall that in their approximation, the entries of each column are drawn proportionally to the row sums, conditioned on the column sum, and that the row sums are not constrained. Apparently, omitting the constraint on the row sums has a drastic effect. As a result, the analysis dramatically underestimated the p-value. This vividly illustrates the utility of exact sampling in these problems.

Our algorithm required 46 seconds to find that there are $2,663,296,694,330,271,332,856,672,902,543,209,853,700$ ($\approx 2.7 \times 10^{39}$) binary matrices with row and column sums as in Table 1, and subsequently, required 4.2 milliseconds per exact sample. (All computations reported in this paper were performed using a 2.8 GHz processor with 6 GB of RAM.) We know of no other algorithm capable of exact counting and sampling for matrices of this size.

2. Overview.

Let $N(\mathbf{p}, \mathbf{q})$ be the number of $m \times n$ binary matrices with margins (row and column sums) $\mathbf{p} = (p_1, \ldots, p_m) \in \mathbb{N}^m$ and $\mathbf{q} = (q_1, \ldots, q_n) \in \mathbb{N}^n$, respectively, and let $M(\mathbf{p}, \mathbf{q})$ be the corresponding number of $\mathbb{N}$-valued matrices. (We use $\mathbb{N} = \{0, 1, \ldots\}$ throughout.) In this paper we develop a technique for finding $N(\mathbf{p}, \mathbf{q})$ and $M(\mathbf{p}, \mathbf{q})$, and for exact uniform sampling from these sets of matrices — an important and challenging problem (Diaconis and Gangolli, 1995; Chen et al., 2005). Our method is feasible for modestly-sized matrices (roughly, $m + n \leq 100$ with current desktop computing power) or very sparse large matrices.

As described above, in the binary case, an important application is testing for structure in ecological co-occurrence matrices. It turns out that most real-world ecology matrices are small enough that our method is feasible. In the $\mathbb{N}$-valued case, an important application is the conditional volume test of Diaconis and Efron (1985) for two-way contingency tables (for which our method is feasible as long as the margins are relatively small). In addition to these direct applications, a major auxiliary benefit of having an exact method is that it enables one to measure the accuracy of certain approximate methods (which can scale to matrices far larger than our exact method can accommodate). We will show how to obtain precise estimates of the total variation distance between the uniform distribution and an approximate distribution, provided that one can sample from and compute probabilities under the approximation.

Since a bipartite graph with degree sequences $\mathbf{p} = (p_1, \ldots, p_m) \in \mathbb{N}^m$, $\mathbf{q} = (q_1, \ldots, q_n) \in \mathbb{N}^n$ (and m, n vertices in each part, respectively) can be viewed as a $m \times n$ matrix with row and column sums $(\mathbf{p}, \mathbf{q})$, our technique applies equally well to counting and uniformly sampling such bipartite graphs. Under this correspondence, simple graphs correspond to binary matrices, and multigraphs correspond to $\mathbb{N}$-valued matrices.

The distinguishing characteristic of our method is its tractability on matrices of nontrivial size. In general, computing $M(\mathbf{p}, \mathbf{q})$ is #P-complete (Dyer, Kannan and Mount, 1997), and perhaps $N(\mathbf{p}, \mathbf{q})$ is as well. However, if one assumes a bound on the column sums then our algorithm computes both numbers in polynomial time. After counting, uniform samples may be drawn in polynomial expected time for bounded column sums. To our knowledge, all previous algorithms *for the non-regular case* require super-polynomial time (in the worst case) to compute these numbers, even for bounded column sums. (We assume a description length of at least $m + n$ and no more than $m \log a + n \log b$, where $a = \max p_i$, $b = \max q_i$.) In general (without assuming a bound on the column sums), our algorithm computes $N(\mathbf{p}, \mathbf{q})$ or $M(\mathbf{p}, \mathbf{q})$ in $\mathrm{O}(m(ab+c)(a+b)^{b-1}(b+c)^{b-1}(\log c)^3)$ time for $m \times n$ matrices, where $a = \max p_i$, $b = \max q_i$, and $c = \sum p_i = \sum q_i$. After counting, uniform samples may be drawn in $\mathrm{O}(mc \log c)$ expected time.

In complement to most approaches to exactly computing $M(\mathbf{p}, \mathbf{q})$, which are efficient for small tables with large margins, our algorithm is efficient for large tables with small margins. For instance, computing $M(\mathbf{p}, \mathbf{q})$ for the 100×100 matrices with $\mathbf{p} = \mathbf{q} = (5^{(20)}, 4^{(20)}, 3^{(20)}, 2^{(20)}, 1^{(20)})$, where $x^{(k)}$ denotes x repeated k times, takes 701 seconds (see Appendix for the exact number, which is approximately 2.96×10^{434}). Likewise, computing $N(\mathbf{p}, \mathbf{q})$ for the same $\mathbf{p}$ and $\mathbf{q}$ takes 688 seconds (see Appendix for the exact number, approximately 2.35×10^{431}).

The remainder of the paper is organized as follows. In the rest of this section, we describe related work from the literature. In Section 3, we first describe the intuition behind our technique, then formally state the recursions and the resulting algorithm, and give bounds on computation time. In Section 4, we illustrate a variety of applications. In Section 5 we prove the recursions, and in Section 6, we prove the bounds on computation time.

2.1. *Previous work.*

We briefly survey the previous work on this problem. This review is not exhaustive, focusing instead on those results which are particularly significant or closely related to the present work. Let $H_n(r)$ and $H_n^*(r)$ denote $M(\mathbf{p}, \mathbf{q})$ and $N(\mathbf{p}, \mathbf{q})$, respectively, when $\mathbf{p} = \mathbf{q} = (r, \ldots, r) \in \mathbb{N}^n$. The predominant focus has been on the regular cases $H_n(r)$ and $H_n^*(r)$.

Work on counting these matrices goes back at least as far as MacMahon (1915, see Vol II, p.161), who applied his expansive theory to find the polynomial for $H_3(r)$. Redfield's theorem (Redfield, 1927), inspired by MacMahon, can be used to derive summations for some special cases, such as $H_n(r), H_n^*(r)$ for $r = 2, 3$, and in similar work, Read (1959; 1960) used Pólya theory to derive these summations for $r = 3$. Two beautiful theoretical results must also be mentioned: Stanley (1973) proved that for fixed n, $H_n(r)$ is a polynomial in r, and Gessel (1987; 1990) showed that for fixed r, both $H_n(r)$ and $H_n^*(r)$ are P-recursive in n, vastly generalizing the recursions for $H_n(2)$, $H_n^*(2)$ found by Anand, Dumir and Gupta (1966).

We turn next to algorithmic results more closely related to the present work. McKay (1983) and Canfield and McKay (2005) have demonstrated a coefficient extraction technique for computing $N(\mathbf{p}, \mathbf{q})$ in the semi-regular case (in which $\mathbf{p} = (a, \ldots, a) \in \mathbb{N}^m$ and $\mathbf{q} = (b, \ldots, b) \in \mathbb{N}^n$). To our knowledge, McKay's is the most efficient method known previously for $N(\mathbf{p},\mathbf{q})$. By our analysis it requires at least $\Omega(mn^b)$ time for bounded a, b, while the method presented here is $\mathrm{O}(mn^b(\log n)^3)$ in this case. Since this latter bound is quite crude, we expect that our method should have comparable or better performance, and indeed empirically we find that typically it is more efficient. If only b is bounded, McKay's algorithm is still $\Omega(mn^b)$, but the bound on our performance increases to $\mathrm{O}(mn^{2b-1}(\log n)^3)$, so it is possible that McKay's algorithm will outperform ours in these cases. Nonetheless, it is important to bear in mind that McKay's algorithm is efficient only in the semi-regular case (while our method permits non-regular margins). If neither a nor b is bounded, McKay's method is exponential in b (as is ours).

McKay and Wormald (1990) presented an intriguing randomized algorithm for exact uniform sampling from the set of binary matrices with margins $\mathbf{p},\mathbf{q}$, provided that all of the margins are sufficiently small. It takes $\mathrm{O}((m+n)^2k^4)$ expected time per sample, where k is an upper bound on all of the margins. Unfortunately, the conditions under which it applies are rather restrictive — in fact, for most of the real-world problems we have encountered, it reduces to a simple rejection sampling scheme that is rather inefficient.

Regarding $M(\mathbf{p},\mathbf{q})$, one of the most efficient algorithms known to date is LattE (Lattice point Enumeration) (De Loera et al., 2004), which uses the algorithm of Barvinok (1994) to count lattice points contained in convex polyhedra. It runs in polynomial time for any fixed dimension, and as a result it can compute $M(\mathbf{p},\mathbf{q})$ for astoundingly large margins, provided that m and n are small. However, since the computation time grows very quickly with the dimension, LattE is currently inapplicable when m and n are larger than 6. There are similar algorithms (Mount, 2000; De Loera and Sturmfels, 2003; Beck and Pixton, 2003) that are efficient for small matrices.

In addition, several other algorithms have been presented for finding $N(\mathbf{p},\mathbf{q})$ (such as Johnsen and Straume, 1987; Wang, 1988; Wang and Zhang, 1998; Pérez-Salvador et al., 2002) and $M(\mathbf{p},\mathbf{q})$ (see review by Diaconis and Gangolli, 1995) allowing non-regular margins, however, it appears that all are exponential in the size of the matrix, even for bounded margins. While in this work we are concerned solely with exact results, we note that many useful approximations for $N(\mathbf{p},\mathbf{q})$ and $M(\mathbf{p},\mathbf{q})$ (in the general case) have been found, as well as approximate sampling algorithms (Holmes and Jones, 1996; Chen et al., 2005; Greenhill, McKay and Wang, 2006; Canfield, Greenhill and McKay, 2008; Harrison and Miller, 2013).

3. Main results.

3.1. *Idea of the algorithm.* Before formally presenting the results, we introduce the simple observation underlying our approach. The basic idea is very straightforward — the surprising part is that it leads to an algorithm that is efficient for many nontrivial datasets.

For simplicity, consider the binary case. The first thing to notice is that $N(\mathbf{p}, \mathbf{q})$ is unchanged by permutations of the entries of $\mathbf{p}$ or $\mathbf{q}$ (as is easy to see). Now, suppose $\mathbf{p} = (6, 6, 4, 3, 1, 1, 1)$ and $\mathbf{q} = (0, 0, 1, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3)$, and consider the partially-filled matrix in Table 3 in which the first row is $\mathbf{u} = (0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0)$ and the rest of the matrix is undetermined.

TABLE 3

0	0	1	1	0	1	0	1	0	1	0	1	0	6
													6
													4
													3
													1
													1
													1
0	0	1	1	1	1	2	2	2	3	3	3	3	

There are $N(\mathbf{p}, \mathbf{q})$ binary matrices with margins $(\mathbf{p}, \mathbf{q})$, and there are clearly $N(L\mathbf{p}, \mathbf{q}-\mathbf{u})$ of these with $\mathbf{u}$ as the first row, where L denotes the left-shift map: $L\mathbf{p} = (p_2, \dots, p_m)$. Divide the first row into blocks $0, 1, 2, \dots, m$ where block k contains the columns i such that $q_i = k$. Since the number of matrices is unchanged under permutations of the margins, then $N(L\mathbf{p}, \mathbf{q} - \mathbf{u}) = N(L\mathbf{p}, \mathbf{q}-\mathbf{v})$ for any permutation $\mathbf{v}$ of $\mathbf{u}$ such that the number of ones in each block is unchanged. If r_k is the size of block k, and s_k is the number of ones in block k, then there are

$$\binom{\mathbf{r}}{\mathbf{s}} := \binom{r_1}{s_1} \cdots \binom{r_m}{s_m}$$

such permutations $\mathbf{v}$, where $\mathbf{r} = (r_1, \dots, r_m)$ and $\mathbf{s} = (s_1, \dots, s_m)$. For instance, in this example, $\binom{\mathbf{r}}{\mathbf{s}} = \binom{4}{3}\binom{3}{1}\binom{4}{2}$. Note that s_0 will be 0, and thus $\binom{r_0}{s_0}$ will be 1, for any $\mathbf{u}$ such that $N(L\mathbf{p}, \mathbf{q} - \mathbf{u}) \neq 0$. Therefore,

$$N(\mathbf{p}, \mathbf{q}) = \sum_{\mathbf{u}\in\{0,1\}^n} N(L\mathbf{p}, \mathbf{q} - \mathbf{u}) = \sum_{\mathbf{s}} \binom{\mathbf{r}}{\mathbf{s}} N(L\mathbf{p}, \mathbf{q} - \mathbf{u}_{\mathbf{s}})$$

where the second sum is over nonnegative integer vectors $\mathbf{s} = (s_1, \dots, s_m)$ summing to p_1, and $\mathbf{u}_{\mathbf{s}} \in \{0, 1\}^n$ is any binary vector with s_k ones in block

k for each $k = 1, \dots, m$, and zero ones in block 0. This defines a recursion for $N(\mathbf{p}, \mathbf{q})$ which, when carefully implemented using dynamic programming and the Gale–Ryser criterion (described below), is the basis for our algorithm. This computation yields a data structure that enables efficient sampling in a row-by-row fashion. There is a similar (although more subtle) recursion for the case of $M(\mathbf{p}, \mathbf{q})$.

3.2. *Recursions, algorithms, and bounds.* Introducing the following notation will be useful. We consider $\mathbb{N}^n$ to be the subset of $\mathbb{N}^\infty := \{(r_1, r_2, \dots) : r_i \in \mathbb{N} \text{ for } i = 1, 2, \dots\}$ such that all but the first n components are zero. Let $L : \mathbb{N}^\infty \to \mathbb{N}^\infty$ denote the left-shift map: $L\mathbf{r} = (r_2, r_3, \dots)$. Given $\mathbf{r}, \mathbf{s} \in \mathbb{N}^\infty$, let $\mathbf{r} \backslash \mathbf{s} := \mathbf{r} - \mathbf{s} + L\mathbf{s}$, (which may be read as "$\mathbf{r}$ *reduce* $\mathbf{s}$"), and let $\bar{\mathbf{r}}$ denote the vector of counts, $\bar{\mathbf{r}} := (\bar{r}_1, \bar{r}_2, \dots)$ where $\bar{r}_i := \#\{j : r_j = i\}$. We write $\mathbf{r} \le \mathbf{s}$ if $r_i \le s_i$ for all i. Given $n \in \mathbb{N}$, let $C_n(k) := \{\mathbf{r} \in \mathbb{N}^n : \sum_i r_i = k\}$ be the n-part compositions (including zeros) of k, and given $\mathbf{s} \in \mathbb{N}^n$, let $C^{\mathbf{s}}(k) := \{\mathbf{r} \in C_n(k) : \mathbf{r} \le \mathbf{s}\}$. Since $N(\mathbf{p}, \mathbf{q})$ and $M(\mathbf{p}, \mathbf{q})$ are fixed under permutations of the row sums $\mathbf{p}$ or column sums $\mathbf{q}$, and since zeros in the margins do not affect the number of matrices, then we may define $\bar{N}(\mathbf{p}, \bar{\mathbf{q}}) := N(\mathbf{p}, \mathbf{q})$ and $\bar{M}(\mathbf{p}, \bar{\mathbf{q}}) := M(\mathbf{p}, \mathbf{q})$ without ambiguity.

Theorem 3.1 (Recursions). *The number of matrices with margins* $(\mathbf{p}, \mathbf{q}) \in \mathbb{N}^m \times \mathbb{N}^n$ *is given by*

$$(1)\quad \bar{N}(\mathbf{p}, \mathbf{r}) = \sum_{\mathbf{s} \in C^{\mathbf{r}}(p_1)} \binom{\mathbf{r}}{\mathbf{s}} \bar{N}(L\mathbf{p}, \mathbf{r} \backslash \mathbf{s}) \qquad \textit{for binary matrices, and}$$

$$(2)\quad \bar{M}(\mathbf{p}, \mathbf{r}) = \sum_{\mathbf{s} \in C^{\mathbf{r}+L\mathbf{s}}(p_1)} \binom{\mathbf{r} + L\mathbf{s}}{\mathbf{s}} \bar{M}(L\mathbf{p}, \mathbf{r} \backslash \mathbf{s}) \qquad \textit{for } \mathbb{N}\textit{-valued matrices,}$$

where $\mathbf{r} = \bar{\mathbf{q}}$, *and in* (2), *we sum over all* $\mathbf{s}$ *such that* $\mathbf{s} \in C^{\mathbf{r}+L\mathbf{s}}(p_1)$.

For the proof, see Section 5. In the binary case, computation of this sum is greatly simplified by summing only over those $\mathbf{s} \in C^{\mathbf{r}}(p_1)$ for which $\bar{N}(L\mathbf{p}, \mathbf{r} \backslash \mathbf{s})$ is nonzero. This can be efficiently achieved using the Gale–Ryser criterion (Gale, 1957; Ryser, 1957), which provides the following necessary and sufficient condition for the existence of a binary matrix with margins $(\mathbf{p}, \mathbf{q})$: when $q'_i := \#\{j : q_j \ge i\}$ and $p_1 \ge \cdots \ge p_m$, we have $N(\mathbf{p}, \mathbf{q}) \ne 0$ if and only if $\sum_{i=1}^{j} p_i \le \sum_{i=1}^{j} q'_i$ for all $j < m$ and $\sum_{i=1}^{m} p_i = \sum_{i=1}^{m} q'_i$. This is easily translated into a condition in terms of $(\mathbf{p}, \bar{\mathbf{q}})$ and $\bar{N}(\mathbf{p}, \bar{\mathbf{q}})$. In the $\mathbb{N}$-valued case, there is no analogue to the Gale–Ryser criterion (since $M(\mathbf{p}, \mathbf{q}) > 0$ for any nonnegative $\mathbf{p}, \mathbf{q}$ such that $\sum_i p_i = \sum_i q_i$). The following recursive procedure can be used to compute either $N(\mathbf{p}, \mathbf{q})$ or $M(\mathbf{p}, \mathbf{q})$.

ALGORITHM 3.2 (Counting).
Input: $(\mathbf{p}, \bar{\mathbf{q}})$, *where* $(\mathbf{p}, \mathbf{q}) \in \mathbb{N}^m \times \mathbb{N}^n$ *are margins such that* $\sum_i p_i = \sum_i q_i$.
Output: $N(\mathbf{p}, \mathbf{q})$ *(or* $M(\mathbf{p}, \mathbf{q})$*), the number of binary (or* $\mathbb{N}$*-valued) matrices.*
Storage: Lookup table initialized with $\bar{N}(\mathbf{0}, \mathbf{0}) = 1$ *(or* $\bar{M}(\mathbf{0}, \mathbf{0}) = 1$*).*

(1) *If* $\bar{N}(\mathbf{p}, \bar{\mathbf{q}})$ *(or* $\bar{M}(\mathbf{p}, \bar{\mathbf{q}})$*) is in the lookup table, return the result.*
(2) *In the binary case, if Gale–Ryser gives* $\bar{N}(\mathbf{p}, \bar{\mathbf{q}}) = 0$*, store the result and return* 0.
(3) *Evaluate the sum in Theorem 3.1, recursing to step* (1) *for each term.*
(4) *Store the result and return it.*

Let $T(\mathbf{p}, \mathbf{q})$ be the time (number of machine operations) required by Algorithm 3.2 to compute $N(\mathbf{p}, \mathbf{q})$ or $M(\mathbf{p}, \mathbf{q})$, after performing an $\mathrm{O}(n^3)$ preprocessing step to compute all needed binomial coefficients. (It turns out that computing $M(\mathbf{p}, \mathbf{q})$ always takes longer, but the bounds we prove apply to both $N(\mathbf{p}, \mathbf{q})$ and $M(\mathbf{p}, \mathbf{q})$.) We give a series of bounds on $T(\mathbf{p}, \mathbf{q})$ ranging from tighter but more complicated, to more crude but simpler. The bounds will absorb the $\mathrm{O}(n^3)$ pre-computation except in the trivial case when the maximum column sum is 1.

THEOREM 3.3 (Bounds). *Suppose* $(\mathbf{p}, \mathbf{q}) \in \mathbb{N}^m \times \mathbb{N}^n$, $a = \max p_i$, $b = \max q_i$, *and* $c = \sum p_i = \sum q_i$. *Then*

(1) $T(\mathbf{p}, \mathbf{q}) = \mathrm{O}((ab+c)(\log c)^3 \sum_{i=1}^{m} \binom{p_i + b - 1}{b-1}\binom{p_i + \cdots + p_m + b - 1}{b-1})$,
(2) $T(\mathbf{p}, \mathbf{q}) = \mathrm{O}(m(ab+c)(a+b)^{b-1}(b+c)^{b-1}(\log c)^3)$,
(3) $T(\mathbf{p}, \mathbf{q}) = \mathrm{O}(mn^{2b-1}(\log n)^3)$ *for bounded* b,
(4) $T(\mathbf{p}, \mathbf{q}) = \mathrm{O}(mn^{b}(\log n)^3)$ *for bounded* a, b.

For the proof, see Section 6. Since we may swap $\mathbf{p}$ and $\mathbf{q}$ without changing the number of matrices, we could use Algorithm 3.2 on $(\mathbf{q}, \bar{\mathbf{p}})$ to compute $N(\mathbf{p}, \mathbf{q})$ or $M(\mathbf{p}, \mathbf{q})$ using $T(\mathbf{q}, \mathbf{p})$ operations, which, for example, is $\mathrm{O}(nm^a(\log m)^3)$ for bounded a, b. $T(\mathbf{p}, \mathbf{q})$ also depends on the ordering of the row sums $p_1, \ldots, p_m$ as suggested by Theorem 3.3(1), and we find that putting them in decreasing order $p_1 \geq \cdots \geq p_m$ tends to work well. In the binary case, Algorithm 3.2 is typically made significantly more efficient by using the Gale–Ryser conditions, and this is not accounted for in these bounds.

It is worth mentioning that a significant further reduction in computation time can be achieved by factoring the sums in Theorem 3.1. For example, in the binary case, we use

$$\bar{N}(\mathbf{p},\mathbf{r}) = \sum_{s_1} \binom{r_1}{s_1} \sum_{s_2} \binom{r_2}{s_2} \cdots \sum_{s_m} \binom{r_m}{s_m} \bar{N}(L\mathbf{p}, \mathbf{r}\backslash\mathbf{s}),$$

where for each $k = 1, \ldots, m$, s_k is summed over a range of values chosen so that $\mathbf{s}$ will always satisfy both $\mathbf{s} \in C^{\mathbf{r}}(p_1)$ and the Gale–Ryser criterion. This improvement is also not accounted for in the bounds above.

Algorithm 3.2 traverses a directed acyclic graph in which each node represents a distinct set of input arguments $(\mathbf{p}, \bar{\mathbf{q}})$ to the algorithm. Node $(\mathbf{u}, \bar{\mathbf{v}})$ is the child of node $(\mathbf{p}, \bar{\mathbf{q}})$ if the algorithm is called (recursively) with arguments $(\mathbf{u}, \bar{\mathbf{v}})$ while executing a call with arguments $(\mathbf{p}, \bar{\mathbf{q}})$. We associate with each node its *count*: the number of matrices with the corresponding margins. If the initial input arguments are $(\mathbf{p}, \bar{\mathbf{q}})$, then all nodes are descendents of node $(\mathbf{p}, \bar{\mathbf{q}})$. Meanwhile, all nodes with positive count are ancestors of node $(\mathbf{0}, \mathbf{0})$. Note the correspondence between the children of a node $(\mathbf{u}, \bar{\mathbf{v}})$ and the compositions $\mathbf{s} \in C^{\bar{\mathbf{v}}}(u_1)$ in the binary case, and $\mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(u_1)$ in the $\mathbb{N}$-valued case, under which $\mathbf{s}$ corresponds with the child $(L\mathbf{u}, \bar{\mathbf{v}}\backslash\mathbf{s})$.

Once the counting is complete, these counts yield an efficient algorithm for uniform sampling from the set of $(\mathbf{p}, \mathbf{q})$ matrices (binary or $\mathbb{N}$-valued). It is straightforward to see that since the counts are exact, the following algorithm yields a sample from the uniform distribution.

Algorithm 3.4 (Sampling).
Input:
· *Row and column sums* $(\mathbf{p}, \mathbf{q}) \in \mathbb{N}^m \times \mathbb{N}^n$ *such that* $\sum_i p_i = \sum_i q_i$.
· *Lookup table of counts generated by Algorithm 3.2 on input* $(\mathbf{p}, \bar{\mathbf{q}})$.
Output: A uniformly-drawn binary (or $\mathbb{N}$*-valued) matrix with margins* $(\mathbf{p}, \mathbf{q})$.

```
(1) Initialize $(\mathbf{u}, \mathbf{v}) \leftarrow (\mathbf{p}, \mathbf{q})$.
(2) If $(\mathbf{u}, \mathbf{v}) = (\mathbf{0}, \mathbf{0})$, exit.
(3) Choose a child $(L\mathbf{u}, \bar{\mathbf{v}}\backslash\mathbf{s})$ of $(\mathbf{u}, \bar{\mathbf{v}})$ with probability proportional to its
    count times the number of corresponding rows (that is, the number of
    rows $\mathbf{r} \in C^{\mathbf{v}}(u_1)$ such that $\overline{\mathbf{v}-\mathbf{r}} = \bar{\mathbf{v}}\backslash\mathbf{s}$.)
(4) Choose a row $\mathbf{r}$ uniformly among the corresponding rows.
(5) $(\mathbf{u}, \mathbf{v}) \leftarrow (L\mathbf{u}, \mathbf{v} - \mathbf{r})$.
(6) Go to (2).
```

In step (3), there are $\binom{\bar{\mathbf{v}}}{\mathbf{s}}$ corresponding rows $\mathbf{r}$ in the binary case (in which $\mathbf{r} \in \{0,1\}^n$), and $\binom{\bar{\mathbf{v}}+L\mathbf{s}}{\mathbf{s}}$ in the $\mathbb{N}$-valued case. In Section 6, we prove that Algorithm 3.4 takes $\mathrm{O}(mc \log c)$ expected time per sample, where $c = \sum_i p_i$.

A software implementation of the algorithms above has been made available, and contains demonstrations of the applications in Section 1 above and Section 4 below.

4. Applications.

4.1. *Co-occurrence matrices in ecology.* As described in Section 1, a primary application is hypothesis testing for ecological matrices. It turns out that many real-world datasets can be accommodated by our method: out of 291 ecology matrices in a collection compiled by Atmar and Patterson (1995), it could handle 225.

In Section 1, we compared our approach to a heuristic approximation. However, it is now more common for researchers to use MCMC (e.g. Gotelli and McCabe, 2002; Ulrich and Gotelli, 2007), and recently, Sequential Importance Sampling (SIS) approaches have been developed that appear to improve upon MCMC (Chen et al., 2005; Harrison and Miller, 2013).

TABLE 4
13 species of finch in 17 of the Galápagos Islands (Chen et al., 2005)

	Habitat																
Species	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17
A	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
B	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0
C	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0
D	0	0	1	1	1	0	0	1	0	1	0	1	1	0	1	1	1
E	1	1	1	0	1	1	1	1	1	1	0	1	0	1	1	0	0
F	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0
G	0	0	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0
H	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
I	0	0	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0
J	0	0	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0
K	0	0	1	1	1	0	1	1	0	1	0	0	0	0	0	0	0
L	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
M	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

To compare with these alternatives, we consider a benchmark dataset for this application. Table 4 indicates the presence/absence of 13 species of finch on 17 of the Galápagos Islands. It comes equipped with the colorful name "Darwin's finches" because Charles Darwin's development of the theory of evolution was inspired in part by his observations of these birds. The row and column sums of the matrix are (14, 13, 14, 10, 12, 2, 10, 1, 10, 11, 6, 2, 17) and (4, 4, 11, 10, 10, 8, 9, 10, 8, 9, 3, 10, 4, 7, 9, 3, 3), respectively. This table is the subject of an analysis by Chen, Diaconis, Holmes, and Liu (2005), in which they use MCMC and their SIS algorithm to estimate the p-value for the $\bar{S}^2$ statistic of Roberts and Stone (1990):

$$\bar{S}^2 = \frac{1}{\binom{m}{2}} \sum_{i<j} c_{ij}^2,$$

where $\mathbf{C} = (c_{ij}) = \mathbf{A}\mathbf{A}^{\mathrm{T}}$ and $\mathbf{A}$ is an $m \times n$ co-occurrence matrix. For given margins, larger values of $\bar{S}^2$ are interpreted as indicating greater competition and cooperation among species (as measured by the variance of the number of habitats c_{ij} shared between a randomly selected pair (i, j) of species).

TABLE 5
Sample statistics of $\bar{S}^2$ for Darwin's finch data (Table 4)

Method	# samples	estimated p-value
Exact	1,000,000	$(4.67 \pm .22) \times 10^{-4}$
SIS	1,000,000	$(3.96 \pm .36) \times 10^{-4}$
MCMC	15,000,000	$(3.56 \pm .68) \times 10^{-4}$

The results of Chen et al. are reported in Table 5, alongside our results using exact sampling. Our results largely confirm the conclusion of Chen et al. — the p-value is small, leading one to reject the null hypothesis (see Figure 2). The computation time for our method is very competitive: for this dataset, counting the number of matrices takes 0.02 seconds, and exact sampling takes 0.16 milliseconds/sample, compared to 1.1 milliseconds/sample for SIS and 0.07 milliseconds/sample for MCMC. (However, on large matrices, MCMC and SIS should be significantly faster than our method, due to the overhead incurred by counting.) The SIS algorithm of Chen et al. also yields an estimate of 6.7150×10^{16} for the number of matrices. We compute the exact number of matrices to be 67,149,106,137,567,626, and note that this agrees with the exact number reported by Chen et al. (which they obtained by other means).

We should emphasize, however, that in comparison with MCMC and SIS, the appeal of our method is not its speed, but rather its exactness. There are no guarantees that the estimates coming from MCMC or SIS have adequately converged. Meanwhile, we are drawing exact i.i.d. samples from the null distribution, which provides many guarantees. For example, using the 10^6 exact samples above, we obtain an exact 95% confidence interval of $[4.26, 5.12] \times 10^{-4}$ for the p-value. (By "exact", we mean that it is a true confidence interval, with no approximations involved. Reported confidence intervals often involve two approximations: (1) approximate samples used for estimation, and/or (2) approximate interval construction based on asymptotics, such as the normal approximation to the binomial.) In a longer run

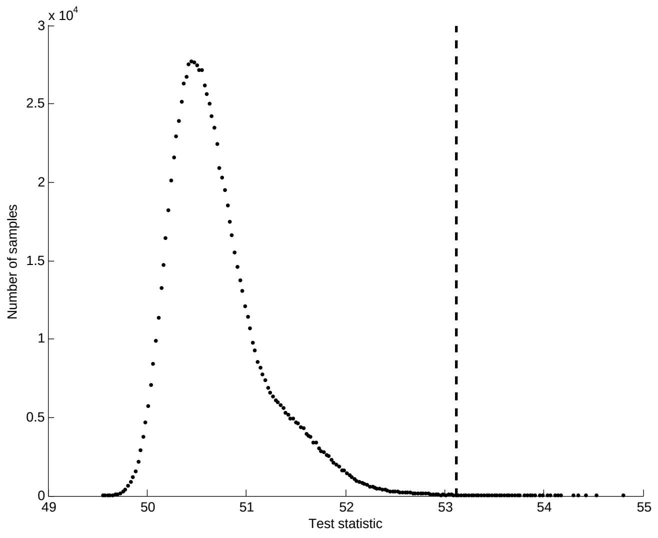

Fig 2: Histogram of $\bar{S}^2$ for 10^6 exact samples from the uniform distribution over matrices with margins as in Table 4. The dashed line is the value of $\bar{S}^2$ for Table 4.

of 10^8 exact samples, we estimate the p-value to be 4.69×10^{-4} and obtain an exact 99.9% confidence interval of $[4.62, 4.77] \times 10^{-4}$. Interestingly, this only barely overlaps with the SIS approximate 95% confidence interval, $[3.25, 4.67] \times 10^{-4}$. Clearly, it is advantageous to use the exact method.

4.2. *Social networks.* Another application is hypothesis testing on social networks. Although many social network datasets are far too large, some are within the range of feasibility for our method. We illustrate with two classic datasets. Over the period 1978–1981, Galaskiewicz (1985) recorded the membership status of 26 of the CEOs of major corporations in the Minneapolis – St. Paul area, with respect to 15 clubs and boards. This resulted in a 26×15 binary matrix with row sums (3, 3, 2, 3, 3, 3, 4, 3, 4, 2, 3, 2, 4, 7, 5, 5, 6, 5, 5, 5, 3, 3, 4, 5, 3, 3) and column sums (3, 11, 22, 12, 3, 4, 4, 4, 6, 3, 4, 5, 5, 3, 9). Since some individuals will tend to belong to more clubs and boards (being more sociable or more influential), and some clubs and boards will simply be larger than others, the uniform distribution on matrices with fixed margins seems to be a reasonable null model.

As before, our method enables efficient exact results: we count the number of matrices to be $25,533,540,876,226,059,861,329,182,058,955,286,218,365,274,646,344,655 \approx 2.55 \times 10^{52}$ in 0.92 seconds, and draw exact samples at a rate of 0.43 milliseconds/sample. As with the ecology matrices, the $\bar{S}^2$ statistic can be used to test for structure in affiliation matrices such as this — in particular, to detect whether individuals exhibit a tendency to segregate into cohesive groups. From 10^4 exact samples, we estimate the p-value to be 0.125 with an exact 95% confidence interval of $[0.118, 0.132]$. Apparently, the data does not strongly indicate such behavior.

Sometimes, social network data takes the form of a $\mathbb{N}$-valued matrix. For instance, Doreian, Batagelj and Ferligoj (2005) analyzed the network of marriages among noble families in the Republic of Ragusa during the 16th century. This data consists of a 24×24 (asymmetric) $\mathbb{N}$-valued matrix with entry (i, j) indicating the number of marriages with husband from family i and wife from family j. The row and column sums are (3, 0, 3, 0, 19, 0, 4, 7, 5, 4, 12, 7, 4, 8, 0, 4, 1, 1, 2, 5, 0, 14, 2, 8) and (0, 1, 4, 2, 20, 2, 4, 4, 5, 4, 15, 9, 3, 6, 1, 2, 0, 0, 2, 4, 1, 21, 0, 3), respectively. We count $949, 599, 133, 340, 064, 609, 956, 529, 916, 243, 690, 765, 923, 314, 405, 123, 631, 501, 076, 903, 661 \approx 9.5 \times 10^{62}$ matrices in 36,708 seconds, and sample at a rate of 1.9 seconds/sample. ($\mathbb{N}$-valued matrices can take significantly longer than binary matrices.)

4.3. *Contingency tables.* Two-way contingency tables are a large class of $\mathbb{N}$-valued matrices that arise frequently in statistics. Pearson's chi-square is the classical test of independence in such tables, however, when the independence hypothesis fails, it can be misleading to interpret the chi-square p-value as a measure of the deviation from independence. As a starting point for quantifying the departure from independence, Diaconis and Efron (1985) propose the conditional volume test (CVT): place the uniform distribution over tables with the observed margins, and compute the probability of observing a chi-square value less than that of the observed table (in other words, compute the p-value under this new null distribution). Larger values of the CVT p-value indicate greater deviation from independence.

Since our method enables exact sampling from the CVT null distribution, it can be used to estimate the p-value for the CVT. While many contingency tables that arise in practice will have margins that are too large for our method, some will fall within the feasible range.

To illustrate, Table 6(a) shows Francis Galton's (1889) data recording the heights (s = short, m = medium, t = tall) of 205 married couples (e.g. medium-short, tall-tall, etc.). Table 6(b) has the same margins but different entries, and in Table 6(c), every entry is exactly double that of Table 6(b). For (a), (b), and (c), Table 7 shows the results of Pearson's chi-square test (the approximate p-value) and the conditional volume test (the estimated p-

TABLE 6

(a)

t	12	20	18
m	25	51	28
s	9	28	14
	s	m	t

(b)

8	14	28
20	61	23
18	24	9

(c)

16	28	56
40	122	46
36	48	18

TABLE 7

Pearson's chi-square and the conditional volume test of Diaconis and Efron (1985)

	chi-square p-value	CVT p-value	CVT 95% CI
Table 6(a)	0.57	0.0011	$[0.0005, 0.0020]$
Table 6(b)	1.2×10^{-5}	0.13	$[0.121, 0.136]$
Table 6(c)	1.8×10^{-11}	0.13	$[0.123, 0.137]$

value and exact 95% confidence interval). The chi-square test and the CVT both indicate that (a) is close to independence, and that (b) is not close to independence. For (c), a naïve interpretation of the chi-square p-value would be that (c) is far further from independence than (b); meanwhile, the CVT indicates that it deviates from independence by roughly the same amount as (b), as one would expect.

These CVT results were each obtained using 10^4 exact samples, drawn at a rate of 0.14 seconds/sample for Tables 6(a) and (b) (after 0.3 seconds required to count the 1,268,792 tables), and 1.94 seconds/sample for Table 6(c) (after 7.7 seconds required to count the 19,151,218 tables).

We test our method further with two examples from Diaconis and Gangolli (1995). The first is an artificial 5×3 table, with margins $(10, 62, 13, 11, 39)$, $(65, 25, 45)$. Our algorithm takes 2.3 seconds to count the 239,382,173 corresponding tables, and 0.3 seconds/sample. Their second example is significantly more challenging: a 4×4 table recording the eye color and hair color of 592 subjects, with margins $(220, 215, 93, 64)$, $(108, 286, 71, 127)$. Our algorithm takes 16,145 seconds to count the 1,225,914,276,768,514 corresponding tables, and 90 seconds/sample. These counting results match the exact numbers reported by Diaconis and Gangolli (1995), obtained by other means.

Our method is not well-suited to small tables with large margins (like the last example), since it exploits the symmetries that arise when there are many columns. For small tables, there are significantly faster algorithms for counting, such as LattE (De Loera et al., 2004). In particular, these algorithms can handle extremely large margins (while ours cannot). However, such algorithms do not scale well to larger tables. In contrast, our method can handle somewhat larger tables, as long as the margins are sufficiently small — consider, for example, the 100×100 example at the end of Section 2.

4.4. *Ehrhart polynomials of the Birkhoff polytope.* Let $H_n(r) = M(\mathbf{p}, \mathbf{q})$ with $\mathbf{p} = \mathbf{q} = (r, r, \dots, r) \in \mathbb{N}^n$. Stanley (1973) proved that for any $n \in \{1, 2, \dots\}$, $H_n(r)$ is a polynomial in r. For example, $H_4(r)$ is

$$H_4(r) = 1 + (65/18)r + (379/63)r^2 + (35117/5670)r^3 + (43/10)r^4$$
$$+(1109/540)r^5 + (2/3)r^6 + (19/135)r^7 + (11/630)r^8 + (11/11340)r^9.$$

Given $H_n(1), H_n(2), \dots, H_n(\binom{n-1}{2})$, one can solve for the coefficients of $H_n(r)$ (as we describe below). These polynomials have been computed for $n \leq 9$ by Beck and Pixton (2003). As an application of our method, we computed them for $n = 4, 5, 6, 7, 8$, requiring $0.2, 0.3, 0.6, 12.9, 5043$ seconds, respectively. This is comparable to (or slightly faster than) the computation time for Beck and Pixton's algorithm (even though theirs is specialized for this purpose). The polynomials we computed matched those reported by Beck and Pixton.

The coefficients of $H_n(r)$ can be determined by the following method. By Stanley (1973), $H_n(r)$ is a polynomial in r such that (a) $\deg H_n(r) = (n-1)^2$, (b) $H_n(r) = 0$ for all $r \in \mathbb{Z}$ such that $-n+1 \leq r \leq -1$, and (c) $H_n(-n-r) = (-1)^{n-1} H_n(r)$ for all $r \in \mathbb{N}$. Let $k = \binom{n-1}{2}$ and $d = (n-1)^2$. Compute the numbers $H_n(r)$ for $r = 0, 1, \dots, k$ using Algorithm 3.2, and form the vector $\mathbf{v} := (H_n(-n-k+1), \dots, H_n(k))^{\mathrm{T}} \in \mathbb{Z}^{d+1}$ using (b) and (c). Form the matrix $\mathbf{A} = \big(a_{ij}\big)_{i,j=1}^{d+1} \in \mathbb{Z}^{(d+1)\times(d+1)}$ where $a_{ij} = (-n-k+i)^{j-1}$, and compute $\mathbf{u} = \mathbf{A}^{-1}\mathbf{v}$. Then by (a), $H_n(r) = \sum_{j=1}^{d+1} u_j r^{j-1}$.

4.5. *Total variation distance of approximate methods.* Another interesting application of our exact method is using it to measure the accuracy of certain approximate sampling methods. Here, we show how to obtain a precise estimate of the total variation distance between the uniform distribution and an approximation to it, provided that one can sample from the approximate distribution and compute the probability of any given matrix under it.

Let P, Q be two probability measures on a finite space $\mathcal{X}$. (We will be thinking of $\mathcal{X}$ as a set of matrices with given margins, P as the uniform distribution, and Q as an approximation.) If $R = \frac{1}{2}P + \frac{1}{2}Q$ and $R(x) > 0$

for all $x \in \mathcal{X}$, then

$$\begin{aligned} d_{\mathrm{TV}}(P, Q) &= \frac{1}{2}\sum_{x\in\mathcal{X}} |P(x) - Q(x)| = \sum_{x\in\mathcal{X}} \frac{|P(x) - Q(x)|}{P(x) + Q(x)} R(x) \\ &= \mathbb{E}\left[\frac{|P(X_1) - Q(X_1)|}{P(X_1) + Q(X_1)}\right] \approx \frac{1}{n}\sum_{i=1}^{n} \frac{|P(X_i) - Q(X_i)|}{P(X_i) + Q(X_i)} =: \hat{d}_n, \end{aligned}$$

where $X_1, \ldots, X_n \sim R$ i.i.d.. Since $0 \le |a - b|/(a + b) \le 1$ for any $a, b \ge 0$ such that $a + b > 0$, by Hoeffding's inequality we have

$$\mathbb{P}(|\hat{d}_n - d_{\mathrm{TV}}(P, Q)| \ge \varepsilon) \le 2\exp(-2n\varepsilon^2)$$

for any $\varepsilon > 0$. Therefore, for any $\alpha \in (0, 1)$, if $\varepsilon = \sqrt{(1/2n)\log(2/\alpha)}$ then $[\hat{d}_n - \varepsilon,\ \hat{d}_n + \varepsilon]$ is a $100(1 - \alpha)\%$ confidence interval for $d_{\mathrm{TV}}(P, Q)$ — that is, $\mathbb{P}(|\hat{d}_n - d_{\mathrm{TV}}(P, Q)| \le \varepsilon) \ge 1 - \alpha$.

This makes it possible to precisely measure the accuracy of an approximation to the uniform distribution. To illustrate, we assess the Sequential Importance Sampling (SIS) approximation of Harrison and Miller (2013). Choose P to be the uniform distribution on a set $\mathcal{X}$ of binary matrices with given margins $(\mathbf{p}, \mathbf{q})$, and choose Q to be this approximation. To sample X_i from $R = \frac{1}{2}P + \frac{1}{2}Q$, flip a fair coin — if it is heads then sample from P, if it is tails then sample from Q. To get the value of $\hat{d}_n$, we must compute $Q(X_1), \ldots, Q(X_n)$, and for this choice of Q, this is provided for by the algorithm of Harrison and Miller (2013). Since this Q is supported on $\mathcal{X}$, and P is uniform on $\mathcal{X}$, then $P(X_1) = \cdots = P(X_n) = 1/|\mathcal{X}| = 1/N(\mathbf{p}, \mathbf{q})$, which we precompute using our counting algorithm.

For example, for the 100×100 matrices with $\mathbf{p} = \mathbf{q} = (5^{(20)}, 4^{(20)}, 3^{(20)}, 2^{(20)}, 1^{(20)})$, taking $n = 10^4$, we estimate $d_{\mathrm{TV}}(P, Q)$ to be $\hat{d}_n = 0.002$ and obtain a 99.9% confidence interval of $\hat{d}_n \pm 0.0195$. The approximation appears to be quite good in this case. Note that this matrix is sparse, with fairly regular margins.

As a contrasting example, for the 13×17 finch matrix from Section 4.1, again taking $n = 10^4$, we estimate $d_{\mathrm{TV}}(P, Q)$ to be $\hat{d}_n = 0.253$ and again obtain a 99.9% confidence interval of $\hat{d}_n \pm 0.0195$. This matrix is not sparse, and both margins are highly irregular.

By assessing an approximation on a variety of examples, one can characterize when it does well and when it does poorly. This provides a guide for selecting an approximation that is well-suited to a given situation, and facilitates the design of better approximations.

5. Proof of recursions. In this section, we prove Theorem 3.1, making rigorous the intuitive argument given in Section 3.1. As an alternative to the "direct" proof below, in (Miller and Harrison, 2011) we also provide a generating function proof that employs some of the beautiful properties of symmetric functions, and yields results of a more general nature.

5.1. *Preliminary observations.* For $\mathbf{r} \in \mathbb{N}^n$, let $\mathbf{r}'$ denote the conjugate of $\mathbf{r}$, that is, $r'_i = \#\{j : r_j \geq i\}$ for $i = 1, 2, 3, \dots$. For $\mathbf{r}, \mathbf{s} \in \mathbb{N}^\infty$, let $\mathbf{r} \wedge \mathbf{s}$ denote the component-wise minimum, that is, $(r_1 \wedge s_1, r_2 \wedge s_2, \dots)$. In particular, $\mathbf{r} \wedge \mathbf{1} = (r_1 \wedge 1, r_2 \wedge 1, \dots)$. Recall our convention that $\mathbb{N}^n$ is considered to be the subset of $\mathbb{N}^\infty$ such that all but the first n components are zero. (Similarly, we consider $\mathbb{Z}^n \subset \mathbb{Z}^\infty$.)

Lemma 5.1. *Let $\mathbf{u}, \mathbf{v} \in \mathbb{N}^n$ such that $\mathbf{u} \leq \mathbf{v}$.*

(1) *Suppose $\mathbf{s} \in \mathbb{N}^d$ for some $d \in \mathbb{N}$. Then $\overline{\mathbf{v} - \mathbf{u}} = \bar{\mathbf{v}} \backslash \mathbf{s}$ if and only if $\mathbf{s} = \mathbf{v}' - (\mathbf{v} - \mathbf{u})'$.*
(2) *If $\mathbf{s} = \mathbf{v}' - (\mathbf{v} - \mathbf{u})'$ then $\sum s_i = \sum u_i$ and $\mathbf{s} \leq \bar{\mathbf{v}} + L\mathbf{s}$.*
(3) *If $\mathbf{s} = \mathbf{v}' - (\mathbf{v} - \mathbf{u})'$ and $\mathbf{u} \leq \mathbf{v} \wedge \mathbf{1}$ then $\mathbf{s} \leq \bar{\mathbf{v}}$.*

Proof. (1) Letting I be the identity operator, a straightforward calculation shows that for any $d \in \mathbb{N}$, $\mathbf{r} \in \mathbb{Z}^d$, we have $\left(\sum_{k=0}^\infty L^k\right)(I - L)\mathbf{r} = \mathbf{r}$ and $(I - L)\left(\sum_{k=0}^\infty L^k\right)\mathbf{r} = \mathbf{r}$, that is, $(I - L)^{-1} = \sum_{k=0}^\infty L^k$ on $\mathbb{Z}^d$. Further, $(I - L)^{-1}\bar{\mathbf{r}} = \mathbf{r}'$. Thus, $\overline{\mathbf{v} - \mathbf{u}} = \bar{\mathbf{v}} \backslash \mathbf{s} = \bar{\mathbf{v}} - \mathbf{s} + L\mathbf{s}$ if and only if $(I - L)\mathbf{s} = \bar{\mathbf{v}} - \overline{\mathbf{v} - \mathbf{u}}$ if and only if $\mathbf{s} = (I - L)^{-1}(\bar{\mathbf{v}} - \overline{\mathbf{v} - \mathbf{u}}) = \mathbf{v}' - (\mathbf{v} - \mathbf{u})'$.

(2) If $\mathbf{s} = \mathbf{v}' - (\mathbf{v} - \mathbf{u})'$ then $\sum s_i = \sum v'_i - \sum (\mathbf{v} - \mathbf{u})'_i = \sum v_i - \sum (v_i - u_i) = \sum u_i$ since $\sum r'_i = \sum r_i$ for all $\mathbf{r} \in \mathbb{N}^n$. By (1), $\bar{\mathbf{v}} - \mathbf{s} + L\mathbf{s} = \bar{\mathbf{v}} \backslash \mathbf{s} = \overline{\mathbf{v} - \mathbf{u}} \geq \mathbf{0}$ and so $\mathbf{s} \leq \bar{\mathbf{v}} + L\mathbf{s}$.

(3) If $\mathbf{s} = \mathbf{v}' - (\mathbf{v} - \mathbf{u})'$ and $\mathbf{u} \leq \mathbf{v} \wedge \mathbf{1}$ then by definition, $s_i = \#\{j : v_j \geq i\} - \#\{j : v_j - u_j \geq i\} = \#\{j : v_j = i \text{ and } u_j = 1\} \leq \#\{j : v_j = i\} = \bar{v}_i$. ☐

We follow the usual notational convention that $f(A) = \{f(a) : a \in A\}$ when A is a subset of the domain of a function f.

Lemma 5.2. *Let $\mathbf{v} \in \mathbb{N}^n$, $k \in \mathbb{N}$, and let $f(\mathbf{u}) = \mathbf{v}' - (\mathbf{v} - \mathbf{u})'$ for $\mathbf{u} \in \mathbb{N}^n$ such that $\mathbf{u} \leq \mathbf{v}$.*

(1) *$f(C^{\mathbf{v} \wedge \mathbf{1}}(k)) = C^{\bar{\mathbf{v}}}(k)$, and for any $\mathbf{s} \in C^{\bar{\mathbf{v}}}(k)$,*

$$\#\{\mathbf{u} \in C^{\mathbf{v} \wedge \mathbf{1}}(k) : f(\mathbf{u}) = \mathbf{s}\} = \binom{\bar{\mathbf{v}}}{\mathbf{s}}.$$

(2) *$f(C^{\mathbf{v}}(k)) = \{\mathbf{s} : \mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(k)\}$, and for any $\mathbf{s}$ such that $\mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(k)$,*

$$\#\{\mathbf{u} \in C^{\mathbf{v}}(k) : f(\mathbf{u}) = \mathbf{s}\} = \binom{\bar{\mathbf{v}} + L\mathbf{s}}{\mathbf{s}}.$$

Proof. (1) $f(C^{\mathbf{v}\wedge\mathbf{1}}(k)) \subset C^{\bar{\mathbf{v}}}(k)$ follows from Lemma 5.1(2 and 3). Let $\mathbf{s} \in C^{\bar{\mathbf{v}}}(k)$. Choose $\mathbf{u}$ as follows. For $i = 1, 2, 3, \dots$, choose s_i of the $\bar{v}_i$ positions j such that $v_j = i$, and set $u_j = 1$ for each chosen j. (Set $u_j = 0$ for all remaining j.) This determines some $\mathbf{u} \in C^{\mathbf{v}\wedge\mathbf{1}}(k)$ such that $s_i = \#\{j : v_j = i \text{ and } u_j = 1\}$ for all i. Furthermore, it is not hard to see that any such $\mathbf{u}$ is obtained by such a sequence of choices. Now, as in the proof of Lemma 5.1(3), $s_i = \#\{j : v_j = i \text{ and } u_j = 1\}$ if and only if $f(\mathbf{u}) = \mathbf{s}$ (when $\mathbf{u} \le \mathbf{v} \wedge \mathbf{1}$). Hence, $f(C^{\mathbf{v}\wedge\mathbf{1}}(k)) \supset C^{\bar{\mathbf{v}}}(k)$, and since there were $\binom{\bar{\mathbf{v}}}{\mathbf{s}}$ possible ways to choose $\mathbf{u}$, then this proves (1).

(2) $f(C^{\mathbf{v}}(k)) \subset \{\mathbf{s} : \mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(k)\}$ follows from Lemma 5.1(2). Suppose $\mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(k)$. Let $\mathbf{r} = \bar{\mathbf{v}}$ and $\mathbf{t} = \mathbf{r}\backslash\mathbf{s}$. Note that $\mathbf{t} \ge \mathbf{0}$ since $\mathbf{s} \le \mathbf{r} + L\mathbf{s}$. Also, $\mathbf{r}, \mathbf{s}, \mathbf{t} \in \mathbb{N}^d$ where $d = \max v_j$. Choose $\mathbf{u}$ as follows. First, consider the r_d positions j in $\mathbf{v}$ at which $v_j = d$. There are $\binom{r_d}{t_d}$ ways to choose t_d of these r_d positions. Having made such a choice, we set $u_j = v_j - d = 0$ for each such j that was chosen. Next, consider the r_{d-1} positions j at which $v_j = d-1$, in addition to the $r_d - t_d$ remaining positions at which $v_j = d$. There are $\binom{r_{d-1}+(r_d-t_d)}{t_{d-1}}$ ways to choose t_{d-1} of these. Having made such a choice, we set $u_j = v_j - (d-1)$ for each such j that was chosen. Continuing in this way, for $i = d-2, \dots, 1$: consider the r_i positions j in $\mathbf{v}$ at which $v_j = i$, in addition to the $r_{i+1} + \cdots + r_d - t_d - \cdots - t_{i+1}$ remaining positions at which $v_j > i$, choose t_i of these (in one of $\binom{r_i + r_{i+1} + \cdots + r_d - t_d - \cdots - t_{i+1}}{t_i}$ ways), and set $u_j = v_j - i$ for each such j that was chosen. After following these steps for each i, set $u_j = v_j$ for any remaining positions j. This determines some $\mathbf{u}$ such that $\mathbf{0} \le \mathbf{u} \le \mathbf{v}$.

Now, for $i = d, d-1, \dots, 1$, we have chosen t_i positions j and we have set $u_j = v_j - i$. That is, $t_i = \#\{j : v_j - u_j = i\}$, and so $\mathbf{t} = \overline{\mathbf{v} - \mathbf{u}}$. Hence, $\overline{\mathbf{v} - \mathbf{u}} = \bar{\mathbf{v}}\backslash\mathbf{s}$ (by the definition of $\mathbf{t}$), so $\mathbf{s} = f(\mathbf{u})$ by Lemma 5.1(1), and additionally, $\sum u_j = \sum s_j = k$ by Lemma 5.1(2). Thus, we have shown that $f(C^{\mathbf{v}}(k)) \supset \{\mathbf{s} : \mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(k)\}$.

Using $t_j = r_j - s_j + s_{j+1}$ (the definition of $\mathbf{t}$), we see that there were

$$\binom{r_d}{t_d}\binom{r_{d-1} + (r_d - t_d)}{t_{d-1}} \cdots \binom{r_1 + r_2 + \cdots + r_d - t_d - \cdots - t_2}{t_1}$$

$$= \binom{r_d}{s_d}\binom{r_{d-1} + s_d}{s_{d-1}} \cdots \binom{r_1 + s_2}{s_1} = \binom{\mathbf{r} + L\mathbf{s}}{\mathbf{s}} > 0$$

ways to make such a sequence of choices, where the inequality holds since $\mathbf{s} \leq \mathbf{r}+L\mathbf{s}$. Hence, there are at least $\binom{\mathbf{r}+L\mathbf{s}}{\mathbf{s}}$ distinct choices of $\mathbf{u} \in C^{\mathbf{v}}(k)$ such that $f(\mathbf{u}) = \mathbf{s}$. On the other hand, given any $\mathbf{u} \in C^{\mathbf{v}}(k)$ such that $f(\mathbf{u}) = \mathbf{s}$, we have $\mathbf{t} = \overline{\mathbf{v}-\mathbf{u}}$ (by Lemma 5.1(1)), thus $t_i = \#\{j : u_j = v_j - i\}$, and since $v_j \geq i$ for any j such that $u_j = v_j - i$, such a $\mathbf{u}$ is obtained by one of the sequences of choices above. Hence, $\#\{\mathbf{u} \in C^{\mathbf{v}}(k) : f(\mathbf{u}) = \mathbf{s}\} = \binom{\bar{\mathbf{v}}+L\mathbf{s}}{\mathbf{s}}$. □

5.2. *Proof of Theorem 3.1.* (1) First, we prove the binary case. Let $(\mathbf{p}, \mathbf{q}) \in \mathbb{N}^m \times \mathbb{N}^n$, $\mathbf{r} = \bar{\mathbf{q}}$. Using Lemma 5.2(1), define the surjection $f : C^{\mathbf{q}\wedge\mathbf{1}}(p_1) \to C^{\mathbf{r}}(p_1)$ by $f(\mathbf{u}) = \mathbf{q}' - (\mathbf{q}-\mathbf{u})'$. Then

$$\bar{N}(\mathbf{p},\mathbf{r}) = N(\mathbf{p},\mathbf{q}) \overset{(a)}{=} \sum_{\mathbf{u}\in C^{\mathbf{q}\wedge\mathbf{1}}(p_1)} N(L\mathbf{p},\mathbf{q}-\mathbf{u})$$

$$\overset{(b)}{=} \sum_{\mathbf{s}\in C^{\mathbf{r}}(p_1)} \sum_{\mathbf{u}\in f^{-1}(\mathbf{s})} N(L\mathbf{p},\mathbf{q}-\mathbf{u}) \overset{(c)}{=} \sum_{\mathbf{s}\in C^{\mathbf{r}}(p_1)} \binom{\mathbf{r}}{\mathbf{s}} \bar{N}(L\mathbf{p},\mathbf{r}\backslash\mathbf{s}).$$

Step (a) follows from partitioning the set of $(\mathbf{p},\mathbf{q})$ matrices according to the first row $\mathbf{u} \in C^{\mathbf{q}\wedge\mathbf{1}}(p_1)$ of the matrix. Step (b) partitions $C^{\mathbf{q}\wedge\mathbf{1}}(p_1)$ into the level sets of f, that is, the sets $f^{-1}(\mathbf{s}) = \{\mathbf{u} \in C^{\mathbf{q}\wedge\mathbf{1}}(p_1) : f(\mathbf{u}) = \mathbf{s}\}$ as $\mathbf{s}$ ranges over $f(C^{\mathbf{q}\wedge\mathbf{1}}(p_1)) = C^{\mathbf{r}}(p_1)$. Step (c) follows since if $f(\mathbf{u}) = \mathbf{s}$ then $\overline{\mathbf{q}-\mathbf{u}} = \mathbf{r}\backslash\mathbf{s}$ (by Lemma 5.1(1)) and thus $N(L\mathbf{p},\mathbf{q}-\mathbf{u}) = \bar{N}(L\mathbf{p},\mathbf{r}\backslash\mathbf{s})$, and since $\#f^{-1}(\mathbf{s}) = \binom{\mathbf{r}}{\mathbf{s}}$ (by Lemma 5.2(1)) . This proves Theorem 3.1(1).

(2) Now, we consider the $\mathbb{N}$-valued case. Let $S = \{\mathbf{s} : \mathbf{s} \in C^{\mathbf{r}+L\mathbf{s}}(p_1)\}$. Using Lemma 5.2(2), define the surjection $g : C^{\mathbf{q}}(p_1) \to S$ by $g(\mathbf{u}) = \mathbf{q}' - (\mathbf{q}-\mathbf{u})'$. Then, similarly,

$$\bar{M}(\mathbf{p},\mathbf{r}) = M(\mathbf{p},\mathbf{q}) \overset{(a)}{=} \sum_{\mathbf{u}\in C^{\mathbf{q}}(p_1)} M(L\mathbf{p},\mathbf{q}-\mathbf{u})$$

$$\overset{(b)}{=} \sum_{\mathbf{s}\in S} \sum_{\mathbf{u}\in g^{-1}(\mathbf{s})} M(L\mathbf{p},\mathbf{q}-\mathbf{u}) \overset{(c)}{=} \sum_{\mathbf{s}\in S} \binom{\mathbf{r}+L\mathbf{s}}{\mathbf{s}} \bar{M}(L\mathbf{p},\mathbf{r}\backslash\mathbf{s}).$$

As before, step (a) follows from partitioning the set of matrices according to the first row $\mathbf{u} \in C^{\mathbf{q}}(p_1)$, step (b) partitions $C^{\mathbf{q}}(p_1)$ into the level sets of g, and step (c) follows since $\#g^{-1}(\mathbf{s}) = \binom{\mathbf{r}+L\mathbf{s}}{\mathbf{s}}$ (by Lemma 5.2(2)). This proves Theorem 3.1(2), completing the proof of Theorem 3.1.

6. Computation time.

Let $W(\mathbf{r}) := \sum_{k=1}^{n} k r_k =$ the *weight* of $\mathbf{r} \in \mathbb{Z}^n$.

LEMMA 6.1 (Properties of the weight). *If* $\mathbf{r}, \mathbf{s} \in \mathbb{Z}^n$ *then*

(1) $W(\mathbf{r}+\mathbf{s}) = W(\mathbf{r}) + W(\mathbf{s})$
(2) $W(\mathbf{s} - L\mathbf{s}) = \sum s_i$
(3) $W(\mathbf{r}\backslash\mathbf{s}) = W(\mathbf{r}) - \sum s_i$
(4) $W(\bar{\mathbf{s}}) = \sum s_i$.

PROOF. All four are simple calculations. □

For the rest of this section, fix $(\mathbf{p}, \mathbf{q}) \in \mathbb{N}^m \times \mathbb{N}^n$ such that $\sum p_i = \sum q_i$, and consider $(\mathbf{p}, \mathbf{q})$ to be the margins of a set of $m \times n$ matrices. First, we address the time to compute $N(\mathbf{p}, \mathbf{q})$ using Algorithm 3.2, and $M(\mathbf{p}, \mathbf{q})$ will follow easily.

Let $\mathcal{D}(\mathbf{p}, \mathbf{q})$ denote the set of nontrivial nodes $(\mathbf{u}, \bar{\mathbf{v}})$ in the directed acyclic graph (as discussed in Section 3) descending from $(\mathbf{p}, \bar{\mathbf{q}})$ (including $(\mathbf{p}, \bar{\mathbf{q}})$), where nontrivial means $(\mathbf{u}, \bar{\mathbf{v}}) \neq (\mathbf{0}, \mathbf{0})$. Let $\Delta_k(j) := \{\mathbf{s} \in \mathbb{N}^k : W(\mathbf{s}) = j\}$ for $j, k \in \mathbb{N}$. The intuitive content of the following lemma is that the graph descending from $(\mathbf{p}, \bar{\mathbf{q}})$ is contained in a union of sets $\Delta_k(j)$ with weights decreasing by steps of $p_1, \dots, p_m$.

LEMMA 6.2 (Descendants). *If* $t_j = \sum_{i=j}^{m} p_i$ *and* $b = \max q_i$ *then*

$$\mathcal{D}(\mathbf{p}, \mathbf{q}) \subset \left\{(\mathbf{u}, \bar{\mathbf{v}}) : \mathbf{u} = L^{j-1}\mathbf{p},\ \ \bar{\mathbf{v}} \in \Delta_b(t_j),\ \ j = 1, \dots, m\right\}.$$

PROOF. By the form of the recursion, $(\mathbf{u}, \bar{\mathbf{v}}) \in \mathcal{D}(\mathbf{p}, \mathbf{q})$ if and only if for some $1 \leq j \leq m$ there exist $\mathbf{s}^1, \dots, \mathbf{s}^{j-1}$ in $C^{\mathbf{r}^1}(p_1), \dots, C^{\mathbf{r}^{j-1}}(p_{j-1})$ respectively, with $\mathbf{r}^1 = \bar{\mathbf{q}}$, $\mathbf{r}^{i+1} = \mathbf{r}^i \backslash \mathbf{s}^i$ for $i = 1, \dots, j-1$, such that $(\mathbf{u}, \bar{\mathbf{v}}) = (L^{j-1}\mathbf{p}, \mathbf{r}^j)$. For $j \geq 2$, by Lemma 6.1(3 and 4),

$$W(\mathbf{r}^j) = W(\mathbf{r}^{j-1}\backslash\mathbf{s}^{j-1}) = W(\mathbf{r}^{j-1}) - p_{j-1} = W(\mathbf{r}^{j-2}) - p_{j-2} - p_{j-1}$$

$$= \cdots = W(\mathbf{r}^1) - (p_1 + \cdots + p_{j-1}) = \sum_{i=1}^{n} q_i - \sum_{i=1}^{j-1} p_i = \sum_{i=1}^{m} p_i - \sum_{i=1}^{j-1} p_i = t_j,$$

and $\mathbf{r}^j \in \mathbb{N}^b$ by construction, so $\mathbf{r}^j \in \Delta_b(t_j)$. Hence, $(\mathbf{u}, \bar{\mathbf{v}}) = (L^{j-1}\mathbf{p}, \mathbf{r}^j)$ belongs to the set as claimed. □

Let $T(\mathbf{p}, \mathbf{q})$ be the time (number of machine operations) required by the algorithm (Algorithm 3.2) to compute $N(\mathbf{p}, \mathbf{q})$ after precomputing all needed binomial coefficients. Let $\tau(\mathbf{u}, \bar{\mathbf{v}})$ be the time to compute $\bar{N}(\mathbf{u}, \bar{\mathbf{v}})$ given

$\bar{N}(L\mathbf{u}, \bar{\mathbf{v}}\backslash\mathbf{s})$ for all $\mathbf{s} \in C^{\bar{\mathbf{v}}}(u_1)$. That is, $T(\mathbf{p}, \mathbf{q})$ is the time to perform the entire recursive computation, whereas $\tau(\mathbf{u}, \bar{\mathbf{v}})$ is the time to perform a given call to the algorithm not including time spent in subcalls.

Let $n_0 := \#\{i : q_i > 0\}$ denote the number of nonempty columns. By constructing Pascal's triangle, we precompute all of the binomial coefficients that may be needed, and store them in a lookup table. We only need binomial coefficients with entries less or equal to n_0, for the following reason. In the binary case, the recursion involves numbers of the form $\binom{\bar{\mathbf{v}}}{\mathbf{s}}$ with $\mathbf{s} \le \bar{\mathbf{v}}$, and for any descendent $(\mathbf{u}, \bar{\mathbf{v}})$ and any $i > 0$ we have $\bar{v}_i \le n_0$ since the number of columns with sum i is less or equal to the total number of nonempty columns. For the $\mathbb{N}$-valued case, the same set of binomial coefficients will be sufficient, since then we have numbers of the form $\binom{\bar{\mathbf{v}}+L\mathbf{s}}{\mathbf{s}}$ with $\mathbf{s} \le \bar{\mathbf{v}} + L\mathbf{s}$, and thus

$$\bar{v}_i + s_{i+1} \le \bar{v}_i + \bar{v}_{i+1} + s_{i+2} \le \cdots \le \bar{v}_i + \bar{v}_{i+1} + \bar{v}_{i+2} + \cdots \le n_0,$$

where the last inequality holds because the number of columns j with sum greater or equal to i is no more than the total number of nonempty columns. Since the addition of two d-digit numbers takes $\Theta(d)$ time, and there are $\binom{n_0+2}{2}$ binomial coefficients with entries less or equal to n_0, then the bound $\log_{10}\binom{j}{k} + 1 \le n_0 \log_{10} 2 + 1$ on the number of digits for such a binomial coefficient shows that this pre-computation can be done in $\mathrm{O}(n_0^3)$ time. Except in trivial cases (when the largest column sum is 1), the additional time needed does not affect the bounds on $T(\mathbf{p}, \mathbf{q})$ that we will prove below.

We now bound the time required for a given call to the algorithm.

LEMMA 6.3 (Time per call). *$\tau(\mathbf{u}, \bar{\mathbf{v}}) = \mathrm{O}((ab + c)(\log c)^3 |C_b(u_1)|)$ for $(\mathbf{u}, \bar{\mathbf{v}}) \in \mathcal{D}(\mathbf{p}, \mathbf{q})$, where $a = \max p_i, b = \max q_i$, and $c = \sum p_i$.*

PROOF. Note that we always have $\bar{v}_i \le n_0$, since the number of columns with sum i cannot exceed the number of nonempty columns. Thus, in the recursion formula, for each $\mathbf{s}$ in the sum corresponding to $(\mathbf{u}, \bar{\mathbf{v}})$, we have the bound

$$\binom{\bar{\mathbf{v}}}{\mathbf{s}} = \prod_{i=1}^{b} \binom{\bar{v}_i}{s_i} \le \prod_{i=1}^{b} \bar{v}_i^{s_i} \le n_0^{\sum_i s_i} \le n_0^a \le c^a.$$

Let $T_m(k)$ be the time required to multiply two numbers of magnitude k or less. By the algorithm of Schönhage and Strassen (1971), $T_m(k) = \mathrm{O}((\log k)(\log\log k)(\log\log\log k))$. Therefore, $T_m(\binom{\bar{\mathbf{v}}}{\mathbf{s}}) \le T_m(c^a) = \mathrm{O}(a(\log c)^3)$. Since we have precomputed the binomial coefficients, the time required to compute $\binom{\bar{\mathbf{v}}}{\mathbf{s}}$ is thus bounded by $\mathrm{O}(ab(\log c)^3)$. To finish computing the term corresponding to $\mathbf{s}$ in the recursion formula, we must multiply

$\binom{\bar{\mathbf{v}}}{\mathbf{s}}$ by $\bar{N}(L\mathbf{u}, \bar{\mathbf{v}}\backslash\mathbf{s})$. Since

$$\bar{N}(L\mathbf{u}, \bar{\mathbf{v}}\backslash\mathbf{s}) \leq N(\mathbf{p},\mathbf{q}) \leq \prod_{i=1}^{m} \binom{n_0}{p_i} \leq \prod_{i=1}^{m} n_0^{p_i} = n_0^c \leq c^c,$$

then this multiplication can be done in $T_m(N(\mathbf{p},\mathbf{q})) \leq T_m(c^c) = \mathrm{O}(c(\log c)^3)$ time. Since we are summing over $C^{\bar{\mathbf{v}}}(u_1)$, and $C^{\bar{\mathbf{v}}}(u_1) \subset C_b(u_1)$, then altogether we have $\tau(\mathbf{u},\bar{\mathbf{v}}) = \mathrm{O}((ab+c)(\log c)^3|C_b(u_1)|)$ for the time per call. □

Lemma 6.4. *$\#\Delta_k(j) \leq \binom{j+k-1}{k-1}$ for any $j,k \in \mathbb{N}$.*

Proof. The map $f(\mathbf{r}) = (1r_1, 2r_2, \ldots, kr_k)$ is an injection $f : \Delta_k(j) \to C_k(j)$. Thus, $\#\Delta_k(j) \leq \#C_k(j) = \binom{j+k-1}{k-1}$. □

We are now ready to prove Theorem 3.3.

Proof of Theorem 3.3 for $N(\mathbf{p},\mathbf{q})$. By storing intermediate results in a lookup table, once we have computed $\bar{N}(\mathbf{u},\bar{\mathbf{v}})$ upon our first visit to node $(\mathbf{u},\bar{\mathbf{v}})$, we can simply reuse the result for later visits. Hence, we need only expend $\tau(\mathbf{u},\bar{\mathbf{v}})$ time for each node $(\mathbf{u},\bar{\mathbf{v}})$ occuring in the graph. Let $t_j = \sum_{i=j}^{m} p_i$ and $d = (ab+c)(\log c)^3$. Then

$$\begin{aligned}
T(\mathbf{p},\mathbf{q}) &= \sum_{(\mathbf{u},\bar{\mathbf{v}})\in\mathcal{D}(\mathbf{p},\mathbf{q})} \tau(\mathbf{u},\bar{\mathbf{v}}) \overset{\text{(a)}}{\leq} \sum_{j=1}^{m} \sum_{\bar{\mathbf{v}}\in\Delta_b(t_j)} \tau(L^{j-1}\mathbf{p},\bar{\mathbf{v}}) \\
&\overset{\text{(b)}}{\leq} \sum_j \sum_{\bar{\mathbf{v}}} \mathrm{O}(d|C_b(p_j)|) = \sum_j \mathrm{O}(d|C_b(p_j)||\Delta_b(t_j)|) \\
&\overset{\text{(c)}}{\leq} \sum_j \mathrm{O}(d\binom{p_j+b-1}{b-1}\binom{t_j+b-1}{b-1}) \\
&\overset{\text{(d)}}{\leq} \sum_j \mathrm{O}(d\binom{a+b-1}{b-1}\binom{c+b-1}{b-1}) \\
&\leq \mathrm{O}(dm(a+b-1)^{b-1}(c+b-1)^{b-1}),
\end{aligned}$$

where (a) follows by Lemma 6.2, (b) by Lemma 6.3, (c) by Lemma 6.4, and (d) since $p_j \leq a$ and $t_j \leq c$. This proves (1) and (2). Now, (3) and (4) follow from (2) since $a \leq c \leq bn$. □

Proof of Theorem 3.3 for $M(\mathbf{p},\mathbf{q})$. Other than the coefficients, the only difference between the recursion for $\bar{M}(\mathbf{p},\bar{\mathbf{q}})$ and that for $\bar{N}(\mathbf{p},\bar{\mathbf{q}})$ is that we are summing over $\mathbf{s}$ such that $\mathbf{s} \in C^{\mathbf{r}+L\mathbf{s}}(p_1)$. Lemma 6.2 holds with the same proof, except with $C^{\mathbf{r}^1}(p_1), \ldots, C^{\mathbf{r}^{j-1}}(p_{j-1})$ replaced by $C^{\mathbf{r}^1+L\mathbf{s}^1}(p_1)$, $\ldots$, $C^{\mathbf{r}^{j-1}+L\mathbf{s}^{j-1}}(p_{j-1})$, respectively. Considering Lemma 6.3, let $(\mathbf{u}, \bar{\mathbf{v}})$ be a descendent of $(\mathbf{p}, \bar{\mathbf{q}})$ in the graph for $\bar{M}(\mathbf{p}, \bar{\mathbf{q}})$, and let $\mathbf{s}$ be such that $\mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(u_1)$. Similarly to before, recalling that $\bar{v}_i + s_{i+1} \leq n_0$ (as proven above in our discussion of precomputing the binomial coefficients), we have

$$\binom{\bar{\mathbf{v}} + L\mathbf{s}}{\mathbf{s}} = \prod_{i=1}^{b} \binom{\bar{v}_i + s_{i+1}}{s_i} \leq \prod_i (\bar{v}_i + s_{i+1})^{s_i} \leq n_0^{\sum s_i} \leq n_0^a \leq c^a.$$

This yields $T_m(\binom{\bar{\mathbf{v}}+L\mathbf{s}}{\mathbf{s}}) \leq T_m(c^a) = \mathrm{O}(a(\log c)^3)$, just as before. Since

$$\bar{M}(L\mathbf{u}, \bar{\mathbf{v}} \backslash \mathbf{s}) \leq M(\mathbf{p}, \mathbf{q}) \leq \prod_{i=1}^{m} \binom{p_i + n_0 - 1}{p_i} \leq \prod_i (2c)^{p_i} = (2c)^c,$$

then we also obtain $T_m(M(\mathbf{p}, \mathbf{q})) \leq T_m((2c)^c) = \mathrm{O}(c(\log c)^3)$ as before. Further, $\{\mathbf{s} : \mathbf{s} \in C^{\bar{\mathbf{v}}+L\mathbf{s}}(u_1)\} \subset C_b(u_1)$, so altogether the time per call is $\mathrm{O}((ab + c)(\log c)^3 |C_b(u_1)|)$, and thus the result of Lemma 6.3 continues to hold. With this result, the proof of the bounds goes through as well. $\square$

This completes the proof of Theorem 3.3. Now, we address the time required to uniformly sample a matrix with specified margins. Let $T_r(k)$ be the maximum over $1 \leq j \leq k$ of the expected time to generate a random integer uniformly between 1 and j. If we are given a random bitstream (independent and identically distributed Bernoulli(1/2) random variables) with constant cost per bit, then $T_r(k) = \mathrm{O}(\log k)$, since for any $j \leq k$, $\lceil \log_2 j \rceil \leq \lceil \log_2 k \rceil$ random bits can be used to generate an integer uniformly between 1 and $2^{\lceil \log_2 j \rceil}$ and then rejection sampling can be used to generate uniform samples over $\{1, \ldots, j\}$. Since the expected value of a Geometric(p) random variable is $1/p$, then the expected number of samples required to obtain one that falls in $\{1, \ldots, j\}$ is always less than 2. More generally, for any fixed $d \in \mathbb{N}$, if we can draw uniform samples from $\{1, \ldots, d\}$, then we have $T_r(k) = \mathrm{O}(\log k)$ by considering the base-d analogue of the preceding argument.

Lemma 6.5 (Sampling time). *Algorithm 3.4 takes*

$$\mathrm{O}(mT_r(n^c) + maT_r(n) + mb\log(a + b))$$

expected time per sample in the binary case, and

$$\mathrm{O}(mT_r((2c)^c) + maT_r(n) + mb\log(a + b))$$

expected time per sample in the $\mathbb{N}$*-valued case. If* $T_r(k) = \mathrm{O}(\log k)$*, then this is* $\mathrm{O}(mc\log c)$ *expected time per sample in both cases.*

REMARK. If b is bounded then $\mathrm{O}(mc \log c) \leq \mathrm{O}(mn \log n)$ since $c \leq bn$, and so this is polynomial expected time for bounded column sums.

PROOF. By the form of the recursion, the depth of the graph descending from $(\mathbf{p}, \bar{\mathbf{q}})$ is equal to the number of rows m, since $\mathbf{p} \in \mathbb{N}^m$ and thus $L^m\mathbf{p} = \mathbf{0}$. For each of the m iterations of the sampling algorithm, we begin at some node $(\mathbf{u}, \bar{\mathbf{v}})$, and we must (A) randomly choose a child $(L\mathbf{u}, \bar{\mathbf{v}} \backslash \mathbf{s})$ with probability proportional to its count times the number of corresponding rows, and then (B) choose a row uniformly from among the $\binom{\bar{\mathbf{v}}}{\mathbf{s}}$ possible choices in the binary case (or $\binom{\bar{\mathbf{v}}+L\mathbf{s}}{\mathbf{s}}$ in the $\mathbb{N}$-valued case).

First consider the binary case. To randomly choose a child, consider a partition of the integers $1, \ldots, N(\mathbf{u}, \mathbf{v})$ with each part corresponding to a term in the recursion formula for $\bar{N}(\mathbf{u}, \bar{\mathbf{v}})$. Generate an integer uniformly at random between 1 and $N(\mathbf{u}, \mathbf{v})$, and choose the corresponding child. Generating such a random number takes $T_r(N(\mathbf{u}, \mathbf{v})) \leq T_r(N(\mathbf{p}, \mathbf{q})) \leq T_r(n_0^c)$ time. Since there are no more than $\binom{a+b-1}{b-1} \leq (a+b-1)^{b-1}$ children at any step, one can determine which child corresponds to the chosen number in $\mathrm{O}((b-1)\log(a+b-1))$ time by organizing the children in a binary tree. So (A) takes $\mathrm{O}(T_r(n_0^c) + b \log(a+b))$ time. Choosing a row consists of uniformly sampling a subset of size s_i from a set of $\bar{v}_i$ elements, for $i = 1, \ldots, b$. Sampling such a subset can be done by sampling without replacement s_i times, which takes $\sum_{j=0}^{s_i-1} T_r(\bar{v}_i - j) \leq s_i T_r(n_0)$ time. So (B) can be done in $\sum_{i=1}^{b} s_i T_r(n_0) \leq a T_r(n_0)$ time. Repeating this process m times, once for each row, we see that sampling a matrix takes $\mathrm{O}(mT_r(n_0^c) + mb\log(a+b) + maT_r(n_0))$ time. If $T_r(k) = \mathrm{O}(\log k)$, this is $\mathrm{O}(mc \log n_0 + mb \log(a+b) + ma \log n_0) \leq \mathrm{O}(mc \log c)$ since $a, b, n_0 \leq c$.

For the $\mathbb{N}$-valued case, the same argument applies, replacing $N(\mathbf{u}, \mathbf{v})$ with $M(\mathbf{u}, \mathbf{v})$, n_0^c with $(2c)^c$, and $\bar{v}_i$ with $\bar{v}_i + s_{i+1}$. ☐

APPENDIX A

Let $\mathbf{p} = \mathbf{q} = (5^{(20)}, 4^{(20)}, 3^{(20)}, 2^{(20)}, 1^{(20)})$, where $x^{(k)}$ denotes x repeated k times.

$N(\mathbf{p}, \mathbf{q})$ = 235, 147, 658, 512, 972, 346, 136, 720, 844, 732, 752, 867, 234, 190, 813, 428, 532, 503, 765, 177, 603, 865, 861, 155, 465, 020, 348, 800, 377, 149, 750, 640, 574, 119, 630, 917, 726, 455, 436, 243, 685, 355, 314, 182, 055, 905, 496, 772, 141, 379, 315, 815, 531, 771, 443, 489, 007, 534, 912, 882, 316, 530, 744, 246, 557, 700, 549, 985, 656, 553, 879, 075, 329, 865, 611, 504, 724, 448, 050, 744, 461, 432, 151, 755, 297, 042, 930, 358, 350, 343, 972, 327, 816, 377, 263, 569, 621, 228, 846, 426, 652, 684, 966, 621, 966, 838, 583, 286, 516, 872, 063, 741, 754, 047, 640, 401, 277, 490, 098, 051, 066, 579, 339, 199, 616, 742, 757, 648, 874, 362, 368, 765, 426, 913, 127, 210, 701, 207, 633, 920, 000, 000, 000, 000

$M(\mathbf{p},\mathbf{q})=$ 295, 805, 666, 559, 153, 333, 218, 627, 349, 404, 825, 478, 412, 279, 341, 291, 256, 310, 789, 770, 124, 587, 755, 158, 139, 247, 610, 594, 820, 909, 327, 345, 130, 630, 237, 874, 869, 835, 355, 865, 067, 328, 480, 221, 768, 746, 326, 533, 318, 518, 034, 068, 762, 147, 953, 091, 161, 124, 607, 645, 707, 089, 033, 147, 454, 208, 952, 389, 634, 816, 899, 177, 414, 716, 448, 735, 101, 144, 730, 357, 394, 276, 262, 070, 385, 948, 234, 353, 936, 879, 761, 227, 595, 488, 660, 906, 921, 645, 496, 874, 333, 081, 165, 039, 663, 928, 985, 440, 675, 621, 736, 614, 359, 726, 081, 137, 694, 371, 480, 321, 973, 707, 301, 121, 622, 703, 911, 176, 113, 569, 810, 500, 814, 339, 034, 022, 648, 523, 951, 964, 160, 000, 000, 000, 000

REFERENCES

Anand, H., Dumir, V. C. and Gupta, H. (1966). A combinatorial distribution problem. *Duke Mathematical Journal* **33** 757-769.

Atmar, W. and Patterson, B. D. (1995). The nestedness temperature calculator: a visual basic program, including 294 presence-absence matrices. *AICS Research Incorporate and The Field Museum*. http://www.aics-research.com/nestedness/tempcalc.html.

Barvinok, A. I. (1994). A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. *Mathematics of Operations Research* **19** 769-779.

Beck, M. and Pixton, D. (2003). The Ehrhart polynomial of the Birkhoff polytope. *Discrete and Computational Geometry* **30** 623-637.

Canfield, E. R., Greenhill, C. and McKay, B. (2008). Asymptotic enumeration of dense 0-1 matrices with specified line sums. *Journal of Combinatorial Theory, Series A* **115** 32-66.

Canfield, E. R. and McKay, B. D. (2005). Asymptotic enumeration of dense 0-1 matrices with equal row sums and equal column sums. *The Electronic Journal of Combinatorics* **12** R29.

Chen, Y., Diaconis, P., Holmes, S. P. and Liu, J. S. (2005). Sequential Monte Carlo methods for statistical analysis of tables. *Journal of the American Statistical Association* **100**.

Connor, E. F. and Simberloff, D. (1979). The assembly of species communities: chance or competition? *Ecology* **60** 1132-1140.

De Loera, J. A. and Sturmfels, B. (2003). Algebraic unimodular counting. *Mathematical Programming* **96** 183-203.

De Loera, J. A., Hemmecke, R., Tauzer, J. and Yoshida, R. (2004). Effective lattice point counting in rational convex polytopes. *Journal of Symbolic Computation* **38** 1273-1302.

Diaconis, P. and Efron, B. (1985). Testing for independence in a two-way table: new interpretations of the chi-square statistic. *Annals of Statistics* **13** 845-874.

Diaconis, P. and Gangolli, A. (1995). Rectangular arrays with fixed margins. In *Discrete Probability and Algorithms* 15-41. Springer-Verlag, New York.

Diamond, J. M. (1975). Assembly of species communities. In *Ecology and evolution of communities* 342-444. Harvard University press, Cambridge.

Doreian, P., Batagelj, V. and Ferligoj, A. (2005). *Generalized Blockmodeling*. Cambridge University Press.

Dyer, M., Kannan, R. and Mount, J. (1997). Sampling contingency tables. *Random Structures and Algorithms* **10** 487-506.

Galaskiewicz, J. (1985). *Social organization of an urban grants economy: A study of business philanthropy and nonprofit organizations*. Academic Press.

Gale, D. (1957). A theorem on flows in networks. *Pacific Journal of Mathematics* **7** 1073-1082.

Galton, F. (1889). *Natural Inheritance*. MacMillan.

Gessel, I. M. (1987). Enumerative applications of symmetric functions. *Séminaire Lotharingien de Combinatoire* **B17a** 5-21.

Gessel, I. M. (1990). Symmetric functions and P-recursiveness. *Journal of Combinatorial Theory, Series A* **53** 257-285.

Gotelli, N. J. and McCabe, D. J. (2002). Species co-occurrence: a meta-analysis of JM Diamond's assembly rules model. *Ecology* **83** 2091–2096.

Greenhill, C., McKay, B. and Wang, X. (2006). Asymptotic enumeration of sparse 0-1 matrices with irregular row and column sums. *Journal of Combinatorial Theory, Series A* **113** 291-324.

Harrison, M. T. and Miller, J. W. (2013). Importance sampling for weighted binary random matrices with specified margins. (Submitted.) arXiv:1301.3928 [stat.CO].

Holmes, R. B. and Jones, L. K. (1996). On uniform generation of two-way tables with fixed margins and the conditional volume test of Diaconis and Efron. *The Annals of Statistics* **24** 64-68.

Johnsen, B. and Straume, E. (1987). Counting binary matrices with given row and column sums. *Mathematics of Computation* **48** 737-750.

MacMahon, P. A. (1915). *Combinatory Analysis* **I,II**. Cambridge University Press, London.

McKay, B. D. (1983). Applications of a technique for labeled enumeration. *Congressus Numerantium* **40** 207-221.

McKay, B. D. and Wormald, N. C. (1990). Uniform generation of random regular graphs of moderate degree. *Journal of Algorithms* **11** 52–67.

Miller, J. W. and Harrison, M. T. (2011). Exact enumeration and sampling of matrices with specified margins. (Unpublished report.) arXiv:1104.0323 [stat.CO].

Mount, J. (2000). Fast unimodular counting. *Combinatorics, Probability, and Computing* **9**.

Patterson, B. and Atmar, W. (1986). Nested subsets and the structure of insular mammalian faunas and archipelagos. *Biological Journal of the Linnean Society* **28** 65-82.

Pérez-Salvador, B. R., de-los Cobos-Silva, S., Gutiérrez-Ándrade, M. A. and Torres-Chazaro, A. (2002). A reduced formula for the precise number of (0,1)-matrices in A(R,S). *Discrete Mathematics* **256** 361-372.

Read, R. C. (1959). The enumeration of locally restricted graphs (I). *Journal of the London Mathematical Society* **s1-34** 417-436.

Read, R. C. (1960). The enumeration of locally restricted graphs (II). *Journal of the London Mathematical Society* **s1-35** 344-351.

Redfield, J. H. (1927). The theory of group-reduced distributions. *American Journal of*

Mathematics **49** 433-455.

Roberts, A. and Stone, L. (1990). Island-sharing by archipelago species. *Oecologia* **83** 560-567.

Ryser, H. (1957). Combinatorial properties of matrices of zeros and ones. *Canadian Journal of Mathematics* **9** 371-377.

Schönhage, A. and Strassen, V. (1971). Schnelle multiplikation großer zahlen. *Computing* **7** 281-292.

Stanley, R. P. (1973). Linear homogeneous Diophantine equations and magic labelings of graphs. *Duke Mathematical Journal* **40** 607-632.

Ulrich, W. and Gotelli, N. J. (2007). Null model analysis of species nestedness patterns. *Ecology* **88** 1824–1831.

Wang, B. Y. (1988). Precise number of (0,1)-matrices in U(R,S). *Scientia Sinica, Series A* **XXXI** 1-6.

Wang, B. Y. and Zhang, F. (1998). On the precise number of (0,1)-matrices in U(R,S). *Discrete Mathematics* **187** 211-220.

Division of Applied Mathematics
Brown University
Providence, RI 02912
E-mail: Jeffrey_Miller@Brown.edu
Matthew_Harrison@Brown.edu