Ann Inst Stat Math (2018) 70:983–1011
https://doi.org/10.1007/s10463-017-0615-z

Hybrid schemes for exact conditional inference in discrete exponential families

David Kahle[1] · **Ruriko Yoshida**[2] · **Luis Garcia-Puente**[3]

Received: 11 November 2015 / Revised: 20 April 2017 / Published online: 4 September 2017

Abstract Exact conditional goodness-of-fit tests for discrete exponential family models can be conducted via Monte Carlo estimation of p values by sampling from the conditional distribution of multiway contingency tables. The two most popular methods for such sampling are Markov chain Monte Carlo (MCMC) and sequential importance sampling (SIS). In this work we consider various ways to hybridize the two schemes and propose one standout strategy as a good general purpose method for conducting inference. The proposed method runs many parallel chains initialized at SIS samples across the fiber. When a Markov basis is unavailable, the proposed scheme uses a lattice basis with intermittent SIS proposals to guarantee irreducibility and asymptotic unbiasedness. The scheme alleviates many of the challenges faced by the MCMC and SIS schemes individually while largely retaining their strengths. It also provides diagnostics that guide and lend credibility to the procedure. Simulations demonstrate the viability of the approach.

✉ David Kahle
david.kahle@gmail.com

Ruriko Yoshida
ryoshida@nps.edu

Luis Garcia-Puente
lgarcia@shsu.edu

[1] Department of Statistical Science, Baylor University, One Bear Place #97140, Waco, TX 76798, USA

[2] Department of Operations Research, Naval Postgraduate School, 1411 Cunningham Road, Monterey, CA 93943, USA

[3] Department of Mathematics and Statistics, Sam Houston State University, Huntsville, TX 77341, USA

Keywords Contingency tables · Exact inference · Markov chain Monte Carlo · Sequential importance sampling · Algebraic statistics

1 Introduction and background

The analysis of contingency tables is one of the oldest problems in statistics, and an enormous amount of literature has been dedicated to the topic. It has now been well over a century since Pearson introduced his ubiquitous asymptotic χ^2 tests (Pearson 1900). A few decades later, concerns about the applicability of that procedure to situations with small sample sizes motivated Fisher to discover the exact procedure that now bears his name, which assesses the simple problem of the independence of the two variables represented in a 2×2 contingency table (Fisher 1922a, 1934). In this article, we consider methods with which one might carry out exact conditional inference in discrete exponential family models on arbitrarily sized tables, a great generalization of Fisher's test. These kinds of procedures have applications all over science and are increasingly warranted in the light of big, small-celled datasets, where they are often the only tool in the statistician's toolkit.

Generalizing Fisherian exact conditional inference first to $R \times C$ tables, then to multiway tables, then to more complex models has proven to be a theoretically simple task that is incredibly challenging in practice. We refer the reader to the excellent though now somewhat dated review by Agresti (1992). Presented in more precise language in Sect. 2, the basic problem is the computation of a p value that is analytically intractable in the same way most Bayesian problems are intractable—the distribution of interest contains an integral (sum) that cannot be computed. As in that setting, Monte Carlo techniques have been employed to estimate probabilities of interest, here p values, to an arbitrary degree of accuracy. Unlike most Bayesian problems, however, in the setting of multiway tables the state space is discrete and implicitly defined, which introduces a number of complications that make the problem unique.

The two-way $R \times C$ independence model case was essentially solved in the late 1970's/mid-1980's via two quite different strategies: sophisticated methods of exhaustive enumeration (Mehta and Patel 1986; Clarkson et al. 1993) and Monte Carlo simulation (Boyett 1979; Patefield 1981). These strategies remain standard practice; for example R's `fisher.test()` implements both, with the former as the default (R Core Team 2014). For the more general multiway problem, the enumeration method is obviously untenable (in fact it is untenable for most $R \times C$ problems), so work in the area has turned to Monte Carlo strategies.

In this article we consider hybrid versions of the two prevailing strategies used to conduct exact conditional inference in discrete exponential families, Markov chain Monte Carlo (MCMC) and sequential importance sampling (SIS), propose one strategy that stands out among similar variants, and communicate simulation results indicative of how the method might perform in real-world applications. We also discuss the apparent limitations of the method.

The outline is as follows. After providing a notational introduction in Sect. 2, we present a concise review of the MCMC and SIS methods in Sects. 3 and 4. In Sect. 5 we consider various potential hybrid schemes that incorporate SIS samples into the

MCMC approach and propose one that stands out among the others. In Sect. 6 we consider the performance of the proposed methods via simulation experiments with independence and logistic models. We then conclude with a discussion of our findings and suggestions for the practical use of the method.

2 Basic notation

Let $X_1, \ldots, X_p$ be a collection of discrete random variables with X_k taking values in $\mathcal{X}_k = [r_k] := \{1, 2, \ldots, r_k\}$, and define $\boldsymbol{X} = [X_1 \ \cdots \ X_p]'$. The sample space of $\boldsymbol{X}$ is $\boldsymbol{\mathcal{X}} = \bigotimes_{k=1}^{p} \mathcal{X}_k$, which contains $r = \prod_{k=1}^{p} r_k$ elements. Let

$$\pi_{\boldsymbol{x}} = \pi_{x_1 x_2 \cdots x_p} = f(\boldsymbol{x}) = P[\boldsymbol{X} = \boldsymbol{x}]$$

all denote the joint probability of the element $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}$. The collection of all r probabilities $\boldsymbol{\pi}$ is then thought of as a p way array $\boldsymbol{\pi} \in \mathbb{R}^{r_1 \times \cdots \times r_p}$ or a vector $\boldsymbol{\pi} \in \mathbb{R}^r$, where some rule is agreed upon as to how to interconvert the two (e.g., lex order). As $\boldsymbol{\pi}$ can be viewed either way, we refer to an individual component either as $\pi_{\boldsymbol{x}}$ or π_k, where $k \in [r]$ is the index associated with $\boldsymbol{x}$, as convenient.

A discrete multivariate dataset is an independent and identically distributed collection of such random vectors $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N$ summarized in a contingency table with elements $T_{\boldsymbol{x}} = \sum_{k=1}^{N} 1[\boldsymbol{X}_k = \boldsymbol{x}]$, where $1[\cdot]$ is the indicator function. Like the $\pi_{\boldsymbol{x}}$'s, the collection of all such $T_{\boldsymbol{x}}$'s can be thought of as an array $\mathbf{T} \in \mathbb{N}_0^{r_1 \times \cdots \times r_p}$, which is the conventional p way contingency table, or a vector $\boldsymbol{T} \in \mathbb{N}_0^r$, which is an elongated form of it made by vectorization. The set $\mathbb{N}_0 = \mathbb{Z}_{\geq 0} = \{0, 1, 2, \ldots\}$ is the collection of nonnegative integers.

Standard approaches to modeling contingency tables place different models on $\boldsymbol{T}$ corresponding to different sampling schemes (Bishop et al. 1975; Agresti 2002; Lehmann and Romano 2005). If the sample size N is not known a priori, it is natural to assume that each cell of the table is an independent Poisson variate whose rate depends on the cell: $T_{\boldsymbol{x}} \sim \text{Pois}(\lambda_{\boldsymbol{x}})$. The sample size itself then follows a Poisson distribution: $N \sim \text{Pois}(\sum_{\boldsymbol{x}} \lambda_{\boldsymbol{x}})$; this is the Poisson sampling scheme. If the sample size $N = n$ is fixed by design, the table follows a multinomial distribution, $[\boldsymbol{T}|N = n] \sim \text{Multinom}_r(n, \boldsymbol{\pi})$, whose cell probabilities are determined by the relative magnitudes of the Poisson rates, $\pi_{\boldsymbol{x}} = \frac{\lambda_{\boldsymbol{x}}}{\sum_{\boldsymbol{x}} \lambda_{\boldsymbol{x}}}$; this is the multinomial sampling scheme. Let $\mathcal{T}_n$ denote the collection of all $\mathbf{T}$-sized p way tables with n observations, $\mathcal{T}_n = \left\{\boldsymbol{t} \in \mathbb{N}_0^r : \mathbf{1}_r' \boldsymbol{t} = n\right\}$. Recall that if $[\boldsymbol{T}|N = n] \sim \text{Multinom}_r(n, \boldsymbol{\pi})$, the table $[\boldsymbol{T}|N = n]$ has probability mass function

$$P_{\boldsymbol{\pi}}[\boldsymbol{T} = \boldsymbol{t}|N = n] = \frac{n!}{t_1! \cdots t_r!} \pi_1^{t_1} \cdots \pi_r^{t_r} \quad \text{for any} \quad \boldsymbol{t} \in \mathcal{T}_n.$$

In this article we always consider the sample size $N = n$ to be known; so to ease notation we simply write $\boldsymbol{T}$ for $[\boldsymbol{T}|N = n]$ and implicitly assume the multinomial sampling scheme.

In algebraic statistics, discrete exponential family models are defined via a configuration matrix $\mathbf{A} \in \mathbb{N}_0^{d \times r}$ that is the transpose of a design matrix for a cell-means ANOVA (Aoki et al. 2012). The matrix serves to compute the sufficient statistics of the exponential family via the linear map $\boldsymbol{S} : \mathbb{N}^r \to \mathbb{N}^d$, $\boldsymbol{S}(\boldsymbol{T}) = \mathbf{A}\boldsymbol{T}$, which we also sometimes refer to as simply $\boldsymbol{S}$ or the marginals, even though they may not be marginals in the traditional sense. It is assumed throughout this article that the vector of ones $\mathbf{1}_r$ is in the row space of $\mathbf{A}$; this is referred to in the algebraic statistics literature as the homogeneity assumption, as it implies the corresponding toric ideal is homogeneous (Sturmfels 1996).

As an illustrative toy example, a 2×2 contingency table would be written

$$\mathbf{T} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \quad \text{or} \quad \boldsymbol{T} = \begin{bmatrix} T_{11} \\ T_{12} \\ T_{21} \\ T_{22} \end{bmatrix},$$

and the independence model, which is an exponential family with sufficient statistics equal to the marginal sums T_{1+}, T_{2+}, T_{+1} and T_{+2} (a_+ index refers to summing over that index), is defined through the following configuration matrix $\mathbf{A}$ in the matrix equation

$$\mathbf{A}\boldsymbol{T} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} T_{11} \\ T_{12} \\ T_{21} \\ T_{22} \end{bmatrix} = \begin{bmatrix} T_{1+} \\ T_{2+} \\ T_{+1} \\ T_{+2} \end{bmatrix}. \tag{1}$$

Any discrete exponential family model can be written in this form for a suitably chosen $\mathbf{A}$, including log-linear and graphical models, logistic and Poisson regression models, and models for rank data; see Drton et al. (2009) or Aoki et al. (2012) for more details. The formulation therefore includes a broad class of very popular models.

For an observed contingency table $\boldsymbol{t}_o \in \mathbb{N}_0^r$ and a configuration matrix $\mathbf{A}$, the sufficient statistics for the model defined by $\mathbf{A}$ constitute the vector of nonnegative integers $\mathbf{A}\boldsymbol{t}_o = \boldsymbol{s} = (s_1, \ldots, s_d)' \in \mathbb{N}_0^d$, and the fiber $\mathcal{F}_{\boldsymbol{s}}$ of $\boldsymbol{t}_o$, the conditional sample space Fisher called the "isostatistical region" (Fisher 1922b), is

$$\mathcal{F}_{\boldsymbol{s}} := \left\{ \boldsymbol{t} \in \mathbb{N}_0^r : \mathbf{A}\boldsymbol{t} = \boldsymbol{s} \right\} \subset \mathcal{T}_n.$$

The fiber $\mathcal{F}_{\boldsymbol{s}}$ is the collection of all contingency tables of n observations with sufficient statistics $\boldsymbol{s}$. From a geometric perspective, the elements of the fiber are nonnegative integer lattice points in a convex polytope. For most problems of practical interest, the number of such tables, the size of $\mathcal{F}_{\boldsymbol{s}}$, is enormous. For example, Diaconis and Sturmfels (1998) relate that Snee's hair and eye color dataset in Table 2 in Sect. 6 has an independence model fiber with 1,225,914,276,768,514 elements (Snee 1974).[1] Chen et al. (2005a) give an example of a 50×50 contingency table with $\{0, 1\}$ entries with an independence model fiber sized hundreds of orders of magnitude larger.

[1] This is the corrected value from that article, which is generally known to have been a typographical error.

If $\boldsymbol{s}$ is given, the conditional distribution of the table given its marginals is

$$P\left[\boldsymbol{T}=\boldsymbol{t}|\mathbf{A}\boldsymbol{T}=\boldsymbol{s}\right]=f(\boldsymbol{t}|\boldsymbol{s})=\frac{\frac{1}{t_1!\cdots t_r!}}{\sum_{\boldsymbol{x}\in\mathcal{F}_{\boldsymbol{s}}}\frac{1}{x_1!\cdots x_r!}}\quad\text{for}\quad\boldsymbol{t}\in\mathcal{F}_{\boldsymbol{s}},\tag{2}$$

which is the generalization of the hypergeometric distribution used in Fisher's exact test referred to as the hypergeometric distribution on the fiber (Drton et al. 2009, Prop. 1.1.11). Unfortunately, the size of the sum in the denominator is the size of the fiber, which is in general far too large to evaluate. Thus, (2) is almost always intractable. However, its un-normalized version

$$\widetilde{f}(\boldsymbol{t}|\boldsymbol{s})=\frac{1}{t_1!\cdots t_r!}\quad\text{for}\quad\boldsymbol{t}\in\mathcal{F}_{\boldsymbol{s}},\tag{3}$$

is quite tractable, especially on the log scale. We refer to $\log\widetilde{f}(\boldsymbol{t}|\boldsymbol{s})$ as the un-normalized log-likelihood (UNLL) of the table $\boldsymbol{t}\in\mathcal{F}_{\boldsymbol{s}}$. Note that tables with higher probabilities also have higher UNLLs, so while we cannot say in an absolute sense what the probability of a table $\boldsymbol{t}\in\mathcal{F}_{\boldsymbol{s}}$ is, we can determine relative likelihoods among several tables easily.

In this article we are interested in computing exact conditional goodness-of-fit p values for significance testing. Although the proper definition of p value has been disputed (Agresti 1992, Sections 2.1 and 3.1), the formulation we assume here is the most commonly agreed upon version:

$$p=P\left[f(\boldsymbol{T}|\boldsymbol{s})\leq f(\boldsymbol{t}_o|\boldsymbol{s})|\mathbf{A}\boldsymbol{T}=\boldsymbol{s}\right];$$

it corresponds to a two-tailed test. We note that it could also be considered from the vantage point of many other discrepancy measures (Read and Cressie 1988).

One way to view p is as a sum of probabilities in the form of (2) over all tables $\boldsymbol{t}$ in the fiber satisfying the condition $f(\boldsymbol{t}|\boldsymbol{s})\leq f(\boldsymbol{t}_o|\boldsymbol{s})$,

$$p=\sum_{\substack{\boldsymbol{t}\in\mathcal{F}_{\boldsymbol{s}}\\ f(\boldsymbol{t}|\boldsymbol{s})\leq f(\boldsymbol{t}_o|\boldsymbol{s})}}f(\boldsymbol{t}|\boldsymbol{s}).\tag{4}$$

However, since $f(\boldsymbol{t}|\boldsymbol{s})$ is intractable, the sum representation is not very helpful in practice. It is much more helpful to represent p as the expectation

$$p=\mathbb{E}\left[1[f(\boldsymbol{T}|\boldsymbol{s})\leq f(\boldsymbol{t}_o|\boldsymbol{s})]|\mathbf{A}\boldsymbol{T}=\boldsymbol{s}\right].\tag{5}$$

Estimating this expectation has been the main strategy of attack in every Monte Carlo approach to exact conditional inference. The most obvious way to do it is simply by using the law of large numbers: drawing samples from (2) and finding the proportion of tables that satisfy the condition. This is the MCMC approach. Alternatively, estimating expectations is the express goal of SIS, and a good deal of success has been achieved via that route. We discuss these in the next two sections.

3 Markov chain Monte Carlo (MCMC)

A complete theory of the MCMC approach grows out of the work of Diaconis and Sturmfels in the early 1990's, which initiated extensive and enduring investigations into the use of algebraic methods in statistics (Diaconis and Sturmfels 1998). While the details grow complicated, the basic goal is simple—sample from (2) and estimate p with the proportion of tables with probabilities $f(\boldsymbol{t}|\boldsymbol{s})$ less than or equal to that of the observed table, determined using their UNLLs. Using this strategy confidence bounds on p can be constructed using standard Monte Carlo techniques if the samples are drawn independently. MCMC is generally useful in these types of situations where normalizing constants are unknown.

When designing a MCMC to sample from (2), the basic problem is the construction of a proposal distribution. There are in fact two problems here. The first is constructing a proposal distribution at all. Since the state space is discrete (unlike most Bayesian problems), care must be taken to construct a proposal distribution that results in tables in the fiber all or nearly all the time. The second is constructing a proposal distribution whose steady-state distribution is the one of interest, namely the hypergeometric distribution on the fiber. We deal with these problems in turn.

The first problem is resolved with the concept of a Markov move. A Markov move, or more simply a move, is a vector $\boldsymbol{m}$ in the integer kernel of $\mathbf{A}$, $\ker_{\mathbb{Z}}(\mathbf{A}) = \ker \mathbf{A} \cap \mathbb{Z}^r$. Such vectors are called moves since one can add them to vectors in $\mathcal{F}_{\boldsymbol{s}}$ to obtain other vectors in $\mathcal{F}_{\boldsymbol{s}}$. For example, in the 2×2 independence model case

$$\boldsymbol{m}_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \quad \text{and} \quad -\boldsymbol{m}_1 = \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \tag{6}$$

are moves. Written in array format, these are

$$\mathbf{m}_1 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \quad \text{and} \quad -\mathbf{m}_1 = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}. \tag{7}$$

Obviously, adding either move to a 2×2 table leaves the marginals unchanged. As the notation suggests, moves are usually identified up to a factor of ± 1. Note that since the fiber $\mathcal{F}_{\mathbf{A}\boldsymbol{t}}$ is finite for a given $\boldsymbol{t}$, the total number of admissible moves is finite. We call a move $\boldsymbol{m}$ admissible, or more specifically admissible for $\boldsymbol{t}$, if $\boldsymbol{t} + \boldsymbol{m} \in \mathcal{F}_{\boldsymbol{s}}$ for some $\boldsymbol{t} \in \mathcal{F}_{\boldsymbol{s}}$, otherwise we call it inadmissible.

Any collection of moves $\mathcal{M}$ can be used as the proposal distribution for a Markov chain by randomly selecting a move from $\mathcal{M}$ and applying it to the current state to obtain a proposed state. Moreover, the observed table is in the fiber, which provides a natural starting point for the chain. This allows us to run the MCMC algorithm presented in Algorithm 1. It works just like any other MCMC algorithm: a step is proposed by the random selection of a move and a transition is made based on the relative likelihoods of the current and proposed states, which do not depend on the

large sum. It also exhibits the challenges of convergence and efficiency common to all MCMC samplers; we address these in Sect. 3.3.

Input:
$\mathcal{M}$ – a collection of moves in $\ker_{\mathbb{Z}}(\mathbf{A})$
$\boldsymbol{t}_0$ – an initial feasible table, e.g., $\boldsymbol{t}_o$
N – a desired number of samples from $f(\boldsymbol{t}|\boldsymbol{s}) = f(\boldsymbol{t}|\mathbf{A}\boldsymbol{t}_o)$

Output:
N samples $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N$ from $f(\boldsymbol{t}|\boldsymbol{s})$, assuming $\mathcal{M}$ connects $\mathcal{F}_{\mathbf{A}\boldsymbol{t}_o}$

for *i in 1:N* **do**
  Sample $\boldsymbol{m}$ from $\mathcal{M}$ and e from $\{1, -1\}$ (uniformly)
  if $any(\boldsymbol{t}_{i-1} + e\boldsymbol{m} < \mathbf{0}_r)$ **then**
    $\boldsymbol{t}_i \leftarrow \boldsymbol{t}_{i-1}$
  else
    Sample $u \leftarrow \text{Unif}(0, 1)$
    if $u < \frac{\widetilde{f}(\boldsymbol{t}_{i-1}+e\boldsymbol{m}|\boldsymbol{s})}{\widetilde{f}(\boldsymbol{t}_{i-1}|\boldsymbol{s})}$ **then**
      $\boldsymbol{t}_i \leftarrow \boldsymbol{t}_{i-1} + e\boldsymbol{m}$
    else
      $\boldsymbol{t}_i \leftarrow \boldsymbol{t}_{i-1}$
    end
  end
end

Algorithm 1: The basic MCMC algorithm given a collection of moves $\mathcal{M}$. Burn-in and thinning are omitted for clarity but are discussed in Sect. 6

Using Markov moves to solve the first problem, the second problem reduces to the determination of a collection of moves $\mathcal{M}$ that guarantees the irreducibility of the Markov chain. Clearly the set of fiber element differences suffices, but this collection is significantly larger than the fiber itself and so generally far too large to use. What is wanted is some kind of minimal collection of admissible moves that result in an irreducible Markov chain. One such collection is a Markov basis. A Markov basis $\mathcal{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_z\}$ for a model $\mathbf{A}$ is a subset of $\ker_{\mathbb{Z}}(\mathbf{A})$ such that for *any* table $\boldsymbol{t}_o \in \mathbb{N}_0^r$, for every pair $\boldsymbol{t}_1, \boldsymbol{t}_2 \in \mathcal{F}_{\boldsymbol{s}} = \mathcal{F}_{\mathbf{A}\boldsymbol{t}_o}$ there exists a sequence of moves $\boldsymbol{b}^{(1)}, \ldots, \boldsymbol{b}^{(K)}$, each in $\mathcal{B}$, such that

$$\boldsymbol{t}_2 = \boldsymbol{t}_1 + \sum_{k=1}^{K} \boldsymbol{b}^{(k)} \quad \text{and} \quad \boldsymbol{t}_1 + \sum_{k=1}^{l} \boldsymbol{b}^{(k)} \geq \mathbf{0}_r$$

for all $l \in [K]$, where $\geq$ is considered element-wise. The first condition requires that $\boldsymbol{t}_2$ is accessible from $\boldsymbol{t}_1$, and the second condition requires nonnegativity of the intermediate states, i.e., that each of the intermediate steps are realizable contingency tables in $\mathcal{F}_{\boldsymbol{s}} \subset \mathcal{T}_n$.

While the notion of Markov bases may appear to make the problem more difficult since a Markov basis must connect every fiber for all sample sizes n, it in fact can make it tractable. The important discovery of Diaconis and Sturmfels (1998) was that Markov bases can be identified with well understood structures in toric algebra. While

the details needlessly digress from our current focus, three facts are helpful; see Drton et al. (2009) and Aoki et al. (2012) for more detailed treatments. First, as a consequence of the Hilbert basis theorem, Markov bases always exist and are finite. Second, they are not unique, since adding more moves to $\mathcal{B}$ retains the required properties. Third, they can be computed using the elimination algorithm from computational algebraic geometry (Cox et al. 1997). Unfortunately, however, this algorithm is known to have very bad worst-case computational complexity, and so in general Markov bases cannot be computed. We revisit this problem in Sect. 3.3.

3.1 Other moves: lattice and other bases

For a particular model $\mathbf{A}$ and observed table $\boldsymbol{t}_o$, a Markov basis is sufficient but not necessary for constructing a proposal distribution for a MCMC whose asymptotic distribution is the hypergeometric distribution on the fiber. A Markov sub-basis is a collection of moves that satisfies the properties of a Markov basis in this restricted situation, i.e., it is a collection of moves that connects the fiber of a particular table at hand (Chen et al. 2006). Markov subbases can be significantly smaller than full Markov bases, but there is no general algorithm known to compute them that is computationally easier than computing the full Markov basis. Thus, there is no general strategy known to compute Markov subbases that is feasible in practice.

Nevertheless, one can run a MCMC with any collection of moves, even collections that do not form a Markov sub-basis or are not even checked. One simply runs the risk that (1) the stationary distribution is not the anticipated one and (2) does not approximate it well. Either of these can be false even when using a collection of moves that do not form a full Markov basis. Moreover, one is forced to use such a collection of moves if a Markov basis is unavailable. We call MCMC run with moves that are known to form a Markov basis MB-MCMC.

In addition to Markov bases, various other collections of moves can be distinguished by their properties. The first and most obvious choice for a collection of moves is motivated by the observation that $\ker_{\mathbb{Z}}(\mathbf{A})$ is the subset of an integer lattice, a collection of integer vectors of $\mathbb{Z}^r$ closed under integer linear combinations. It is well known that lattices admit bases: a *lattice basis* is a collection of linearly independent vectors $\mathcal{L} \subset \ker_{\mathbb{Z}}(\mathbf{A})$ such that every vector in $\ker_{\mathbb{Z}}(\mathbf{A})$ can be written as a linear combination of elements of $\mathcal{L}$. Such a basis contains $l = r - \operatorname{rank}(\mathbf{A}) \leq r - 1$ elements, typically far fewer than a Markov basis, but they are not unique. Lattice bases can be computed very quickly using the Smith decomposition of the integer matrix $\mathbf{A}$ (Schrijver 1986). However, there is no guarantee that a lattice basis will form a Markov basis or sub-basis, and so using only a lattice basis as the collection of moves $\mathcal{M}$ runs the risk of having an asymptotic distribution that is not the hypergeometric distribution on the fiber and does not approximate it well. We call MCMC run with lattice basis moves LB-MCMC; we do not demand the moves be linearly independent.

Other common collections of moves exist as well. Such collections typically come from other areas of mathematics such as graph theory and optimization. Chief among these are the Gröbner and Graver bases and their variants (Drton et al. 2009; Sturmfels 1996). There is a clear hierarchy of such bases: a Graver basis contains a Gröbner basis which contains a Markov basis which contains a lattice basis, and their sizes can be

Table 1 The politics dataset

	Democrat	Republican	
Introvert	3	7	**10**
Extrovert	6	4	**10**
	9	**11**	**20**

Marginal totals are in bold

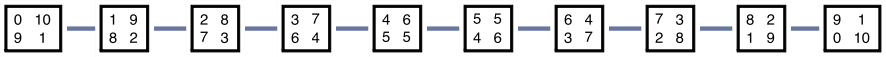

Fig. 1 The fiber graph of the independence model on the politics dataset in Table 1 with ± moves

dramatically different. Since mere Markov bases are sufficiently difficult to compute as to not be generally applicable in practice, in this article we only consider lattice and Markov bases.

3.2 The graphical perspective

It is often helpful to consider Markov chains from the perspective of graph theory. This perspective views the elements of the fiber as vertices of an undirected graph called a fiber graph with adjoining edges if the move set $\mathcal{M}$ contains a move from one to the other. Explicitly, for a given collection of moves $\mathcal{M}$, two vertices $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$ are joined by an edge if there is a move $\boldsymbol{m} \in \mathcal{M}$ such that $\boldsymbol{t}_1 = \boldsymbol{t}_2 + \boldsymbol{m}$, often seen as $\boldsymbol{t}_1 - \boldsymbol{t}_2 \in \mathcal{M}$. For example, Table 1 contains a version of Sheskin's politics dataset with variables Personality and Party (Sheskin 2007, p. 622). The independence model on the table has the configuration matrix in (1); (6)/(7) is a collection of moves; and the corresponding fiber graph is shown in Fig. 1.

In this case, the $\begin{smallmatrix}+&-\\-&+\end{smallmatrix}$ basic move that has a pair of ±1's in each ordinate connects the entire fiber; it is in fact a lattice, Markov, Gröbner and Graver basis. In the larger case of $R \times C$ tables, $\begin{smallmatrix}+&-\\-&+\end{smallmatrix}$ basic moves with the +'s and −'s in strategic locations still generate the lattice, Markov, Gröbner and Graver bases, but different numbers of them are needed. For example, the 3×3 independence model case demands 4, 9, 9, and 15 such moves, respectively.

The situation changes for fibers of different models and different sized tables. A simple example is the $3 \times 3 \times 3$ table with the no-three-way interaction model. The lattice basis contains 8 elements, all $\begin{smallmatrix}+&-\\-&+\end{smallmatrix}$ moves, and is shown in Fig. 2. A minimal Markov basis contains 81 elements that consist of more than simple single $\begin{smallmatrix}+&-\\-&+\end{smallmatrix}$ moves. The corresponding fiber graphs, when $\boldsymbol{s} = [2 \cdots 2]'$, are shown in Fig. 3. The Gröbner and Graver bases, not shown, have 110 and 795 elements, about one and six times the size of the fiber, respectively. As indications of their complexity, many Markov and Gröbner bases moves have more than one $\begin{smallmatrix}+&-\\-&+\end{smallmatrix}$ configuration in the same move, and some of the Graver basis moves have ±2's.

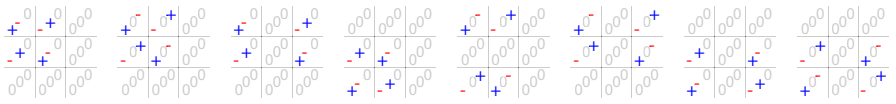

Fig. 2 Eight moves for the $3 \times 3 \times 3$ no-three-way interaction model constituting a lattice basis for the corresponding $\mathbf{A}$. *Blue* $+$'s are $+1$ and *Red* $-$'s are -1

3.3 Challenges of MCMC schemes

It has already been said that generating a Markov basis is a difficult process. In fact, it is an NP-hard process, growing with the size of the table and the complexity of the model. Typically log-linear models that pose problems in other situations (e.g., ones that do not have closed-form MLEs) also have problematic Markov bases. A standard example is the no-three-way interaction model, whose minimal Markov bases are known to grow to be arbitrarily large as the table grows (De Loera and Onn 2005).

As a concrete example of the difficulty of computing Markov bases, 4ti2's `markov` program (4ti2 team 2008), widely regarded as a state-of-the-art implementation for computing Markov bases, took nearly 24 hours to compute the Markov basis of the configuration matrix $\mathbf{A}$ of the no-three-way interaction model on a $4 \times 4 \times 4$ contingency table, a 48×64 0–1 matrix, on a 3.4GHz Intel Xeon processor with 32GB of memory. The resulting basis has a whopping 148,968 elements. The full independence model on the same sized table takes about a second and contains 1080 elements. By contrast, the lattice bases (computed with 4ti2's `zbasis` program) contain 27 and 54 elements, respectively, and were computed in a fraction of a second.

When the model exhibits a nice structure, e.g., it is graphical and highly decomposable, it is possible to generate Markov bases very quickly by aggregating the bases of subtables (Dobra and Sullivant 2004; Dobra 2003). Ample literature has been logged where Markov bases have been determined by leveraging symmetries in the underlying model (Aoki et al. 2012). Moreover, for a given model $\mathbf{A}$ the basis need only be computed once, which prompted Kahle and Rauh (2011) to construct an online database of Markov bases. Nevertheless, it is well-known that the problem of determining a Markov basis is not, in general, one that can be resolved by additional computational power.

Apart from irreducibility, the two major problems that MCMC strategies face are (1) convergence to the hypergeometric distribution on the fiber, called the mixing of the chain, and (2) the efficiency of the chain. By mixing we mean at what point t^* in the chain $\{\boldsymbol{T}_t\}_t$ are we able to assume that the marginal distribution of $\boldsymbol{T}_{t^*}$ is the hypergeometric distribution on the fiber. Answers to this question are provided by the general theory of finite state space Markov chains but may not be very helpful in this context since the distribution of interest is not uniform and properties of the chain, such as the diameter of the graph, are often unknown. By efficiency we mean once the chain has converged, how many steps need one take before the autocorrelation tamps down to an acceptable level? In other words, how many steps do we need to "thin", discard between iterations, in order for the remaining samples to be reliable for estimating standard errors and effective sample sizes? In general, there are few known answers to these problems. We revisit them in a concrete way in Sects. 5 and 6.

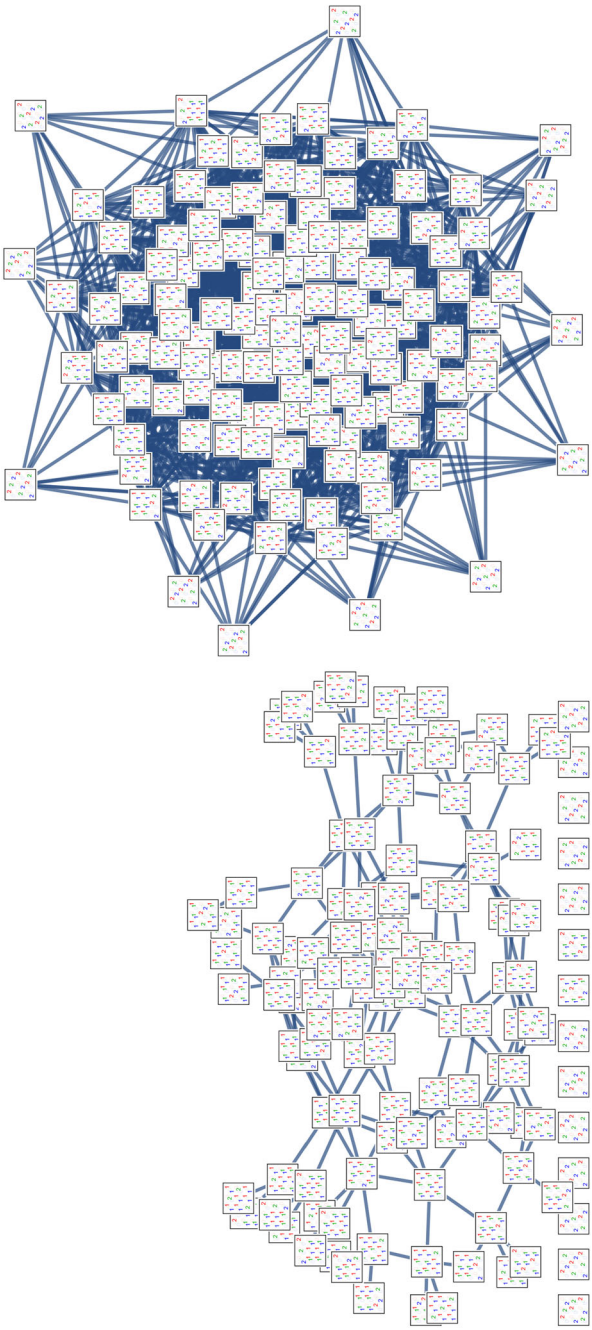

Fig. 3 The fiber graph of a $3 \times 3 \times 3 \times 3$ contingency table with all sufficient statistics equal to 2 in the no-three-way interaction model using a lattice basis (*left*, 200 edges) and a minimal Markov basis (*right*, 1701 edges). Note the isolated row of tables when using a lattice basis; these are boundary elements of the fiber. The fiber has 132 elements

3.4 MCMC enhancements

Various schemes exist in the literature that attempt to enhance the MCMC strategy described above. In this section we simply mention two such attempts before moving on. Since they work with every MCMC, they work with the methods considered in this article as well, but since the aim is not to better understand these kinds of enhancements we do not consider them further in this work.

Recognizing that moves in a basis are typically small, e.g., consisting of ± 1 entries in a fiber that may have points that are very distant, one idea is to enlarge them somehow. Diaconis and Sturmfels (1998) describe a strategy based on the hit and run algorithm of Bélisle et al. (1993). Given a set of moves $\mathcal{M}$, the idea is to effectively to make proposals in a two-step process. The first step is to pick a move uniformly at random from the collection of possible moves; this it the standard proposal step. The second step is determine the collection of multiples of the move that still result in a proposal in the fiber, and pick one of these multiples uniformly at random (say) and propose that multiple of the move. Clearly, this allows the chain to "get random much more rapidly" (Diaconis and Sturmfels 1998, p. 374).

A similar set of strategies are discussed in Chapter 16 of Aoki et al. (2012), synthesizing previous work by the same authors (Hara et al. 2012). In that discussion it is suggested that when a Markov basis is unavailable it is reasonable to use random integer combinations of a lattice basis moves, with the coefficients being drawn from either a Poisson or a geometric distribution. If the move-multiples do not provide proposals in the fiber, they are simply rejected. They find that both methods seem to work reasonably well in simulations and offer the suggestion that as the "size" of the table increases (in a loose sense relating to cell size), increasing the rates of the Poisson or geometric distributions to achieve bigger moves increases performance.

4 Sequential importance sampling (SIS)

Apart from the MCMC methods, alternative strategies to estimate p based on sequential importance sampling (SIS) have been proposed. A sequential importance sampler, a variant of sequential Monte Carlo (SMC, also called particle filtering), is an importance sampler for a multivariate distribution whose proposal distribution is constructed iteratively via conditional univariate distributions (Liu 2008; Lange 2010). As an importance sampler, it is designed not to sample from a particular distribution per se, but rather provide samples that can be used to approximate expectations respective of a particular distribution. Since p values can be represented as expectations, SIS is a prime alternative to the MCMC routine previously described.

Like the MCMC approach, the SIS approach relies on a proposal distribution, here denoted $q(\boldsymbol{t})$. Suppose $q(\boldsymbol{t})$ is a proposal distribution with support $\mathcal{Q} \subset \mathbb{N}_0^r$ that we can sample from efficiently and that $\mathcal{F}_s \subseteq \mathcal{Q}$. Then, for an arbitrary distribution $p(\boldsymbol{t})$ and function $g(\boldsymbol{t})$ on $\mathcal{F}_s$, one can approximate the expectation

$$\mathbb{E}_p\left[g(\boldsymbol{T})\right] \approx \frac{\sum_{i=1}^{N} g(\boldsymbol{t}_i)\frac{\widetilde{p}(\boldsymbol{t}_i)}{\widetilde{q}(\boldsymbol{t}_i)}}{\sum_{i=1}^{N}\frac{\widetilde{p}(\boldsymbol{t}_i)}{\widetilde{q}(\boldsymbol{t}_i)}} = \frac{\sum_{i=1}^{N} w_i g(\boldsymbol{t}_i)}{\sum_{i=1}^{N} w_i}$$

where $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N$ are N independent and identically distributed (iid) draws from the distribution $q(\boldsymbol{t})$, and $\widetilde{p}(\boldsymbol{t}_i)$ and $\widetilde{q}(\boldsymbol{t}_i)$ are un-normalized versions of p and q. The quantities $w_i := \frac{\widetilde{p}(\boldsymbol{t}_i)}{\widetilde{q}(\boldsymbol{t}_i)}$ are referred to as importance weights. The independence of the samples $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N$ makes the SIS procedure an example of independent Monte Carlo (IMC), which is in sharp contrast to MCMC.

As with the MCMC procedure, selection of a proposal distribution is central to SIS. In SIS, $q(\boldsymbol{t})$ is constructed as a product of conditional distributions. Specifically, for $k \in [r]$, define $\boldsymbol{t}_{1:k} := [t_1 \ \cdots \ t_k]'$. The SIS scheme then constructs $q(\boldsymbol{t})$ as

$$q(\boldsymbol{t}) = q_1(t_1)\prod_{k=2}^{r} q_k(t_k|\boldsymbol{t}_{1:(k-1)}), \tag{8}$$

which simply swaps a multivariate sampler for a series of conditional univariate samplers. The question then becomes: how are the q_k's selected? It is well known that the efficiency of a proposal distribution for an importance sampling procedures depends on how closely $q(\boldsymbol{t})$ resembles $|g(\boldsymbol{t})|p(\boldsymbol{t})$ (Lange 2010); however, in practice determining the marginal distributions q_k that result in this distribution is difficult, and it is more common to design the q_k's to result in a q that targets p.

Chen et al. (2005a), motivated by previous work (Halton 1969; Boyett 1979; Snijders 1991; Booth and Butler 1999; Caffo and Booth 2001, and references therein), provide a look into how to construct proposal distributions q_k for two-way tables. They considered sampling from the uniform and hypergeometric distributions on the fibers of two kinds of tables: tables with 0–1 entries and tables with more typical cell sizes. Their general findings were that 0–1 tables are significantly more difficult to sample than ordinary tables due to support issues, and that good q_k's are much easier to find for the uniform distribution than for the hypergeometric distribution. For sampling from the uniform, for non-0–1 celled tables one can simply use discrete uniform distributions over the support given by the Fréchet bounds, updating column/row totals as one iterates through the table. For 0–1 tables, sampling columns at a time from a conditional-Poisson (CP) proposal distribution works well. Sampling from the hypergeometric distribution requires much more attention to detail. Instead of sampling uniformly over the support given by the Fréchet bounds, Chen et al. (2006) proposed sampling values from the hypergeometric distribution. It appeared to work well if the table was not sparse but poorly if it was.

The multiway case, presented in the same work, is significantly more complex. In particular, the support of the conditional distributions comes into question, and the interval of integers support provided by the Fréchet bounds, called the sequential interval property (SIP), was found to only hold when certain algebraic conditions were met. Worse, the conditions are difficult to check. However, they note that in the special case of sampling from the uniform distribution on a fiber of a non-0–1 table the naive approach of sampling from conditional uniform distributions over an interval of integers given by the solution of a linear program still works remarkably well (i.e.,

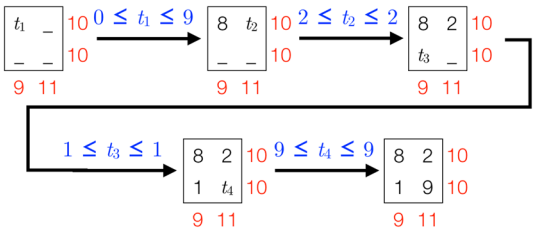

Fig. 4 The SIS approach to sampling uniformly from $\mathcal{F}_{\boldsymbol{s}}$ for the politics dataset in Table 1

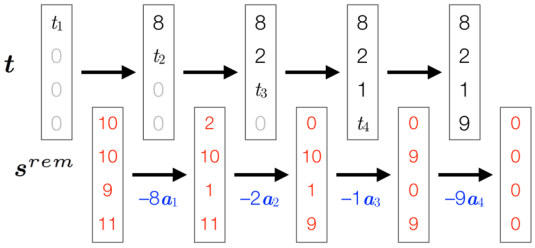

Fig. 5 Algorithm 2 applied to the politics dataset. This is Fig. 4 in the notation of Algorithm 2; the $\boldsymbol{a}_k$'s are the columns of $\mathbf{A}$ in (1)

is surprisingly efficient), even when the algebraic conditions are not satisfied or not even checked (Chen et al. 2006, p. 543). This very practical observation is leveraged by the hybrid schemes presented in Sect. 5 and illustrated in Sect. 6.

In this work we use SIS for sampling uniformly from the fiber exclusively, and not for estimating expectations directly. In particular, the type of Monte Carlo that we use and refer to loosely as SIS is a rejection sampler that uses a SIS-like proposal $q(\boldsymbol{t})$ that factors according to (8) and uses discrete uniform q_k's. The basic idea of the procedure is simple: iterate through the cells of the table populating them with uniformly sampled integers in the range of feasible integers as determined by a linear program and check the resulting table to see if it is in the fiber; if it is not discard it and run it again. This is illustrated with the 2×2 politics dataset in Figs. 4 and 5. We provide the process in algorithmic notation in Algorithm 2.

Algorithm 2 is actually slightly over-simplified to ease the exposition. It can be corrected and improved by making a few relatively minor modifications. For example, it is possible that a partial completion of the table cannot yield a total completion, and there are conditions that can be checked to make sure it can be completed. In practice, however, as with many other Monte Carlo algorithms it is a good deal easier to simply run the algorithm and monitor its progress.

5 Hybrid schemes

In this section we discuss hybrid MCMC/SIS schemes that seek to leverage the strengths of both methods while minimizing their weaknesses. All schemes incorporate SIS into the MCMC paradigm, because ultimately we want to sample from the hypergeometric distribution on the fiber and doing so appears easier with MCMC than SIS. The outline is as follows: In Sect. 5.1, we list the key advantages and disadvantages of both strategies in order to frame the discussion. In Sect. 5.2, we describe various points at which SIS can be inserted into the MCMC process to generate hybrid schemes.

Input:
$\mathbf{A}$ – the model configuration matrix
$\boldsymbol{s}$ – the sufficient statistics; $\boldsymbol{s} = \mathbf{A}\boldsymbol{t}_o$ for the exact inference problems

Output:
1 sample $\boldsymbol{t}$ approximately uniform over $\mathcal{F}_{\boldsymbol{s}}$

Initialize:
$\boldsymbol{t} \leftarrow \mathbf{0}_r$
$\boldsymbol{s}^{rem} \leftarrow \boldsymbol{s}$

for *k in 1:(r-1)* **do**
Compute bounds l and u for the support of q_k as rounded up and down solutions to the linear program

$$[l, u] = \min_{\mathfrak{X}_k} / \max_{\mathfrak{X}_k} x_1$$

where

$$\mathfrak{X}_k = \left\{ \boldsymbol{x} \in \mathbb{R}^{r-k+1}_{\geq 0} : \mathbf{A}_{k:r}\boldsymbol{x} = \boldsymbol{s}^{rem} \right\}$$

and $\mathbf{A}_{k:r}$ is the submatrix of $\mathbf{A}$ of columns k to r
Sample t_k uniformly from $\{l, l+1, \ldots, u\}$
$\boldsymbol{s}^{rem} \leftarrow \boldsymbol{s}^{rem} - t_k \boldsymbol{a}_k$
end
$t_r \leftarrow s^{rem}_{(1)} / a_{(1)r}$, where $_{(1)}$ denotes the first nonzero element
If $\mathbf{A}\boldsymbol{t} \neq \boldsymbol{s}$, repeat the entire process.

Algorithm 2: The basic SIS algorithm to sample one table roughly uniformly from $\mathcal{F}_{\boldsymbol{s}}$. Multiple samples are obtained by running the algorithm many times, which can be done in a distributed computing framework

In Sect. 5.3, we introduce a particular combination of those components well-suited for general use in the kinds of exact inference problems described in Sects. 1 and 2.

5.1 Advantages and disadvantages of MCMC and SIS

To see where each strategy excels and falters, it is helpful to summarize aspects of each:

1. MCMC
   (a) Advantages :
      – *Speed.* The algorithm uses simple calculations.
      – *Correctness.* Once converged, the algorithm generates samples from the hypergeometric distribution on the fiber (2), the distribution of interest for exact inference.
   (b) Disadvantages :
      – *Prohibitive pre-computations.* Constructing an irreducible chain by computing a Markov basis is computationally infeasible in many practical cases. LB-MCMC does not have this problem, but runs risks mentioned in Sect. 3.
      – *Mixing.* The chain may mix slowly, so that convergence to the steady-state distribution may require a large number of steps.

- *Inefficiency.* After convergence to the steady-state distribution, the autocorrelation in the series of samples can be very large, requiring expensive thinning.
- *Serial computations.* The MCMC algorithm is sequential in nature, and thus cannot leverage added computing capacity.

2. SIS
   (a) Advantages :
   - *Independent Monte Carlo (IMC).* The samples are independent by construction.
   - *Pre-computation free.*
   - *Parallelizability.* The simultaneous computation of many tables can be distributed across multiple computing units.

   (b) Disadvantages :
   - *Incorrectness.* Using uniform conditional distributions q_k, SIS samples from the uniform distribution on the fiber, not the hypergeometric distribution in (2). In principle it can be adapted to the hypergeometric distribution, but the conditional distributions become complex and as of it yet does not perform as well in that setting.
   - *Expensive run-time computations.* Several increasingly simple linear programs must be solved to sample a single table, so tons of linear programs must be solved to sample many tables.

Notice that the paradigms are effectively opposite in their strengths and weaknesses.

In summary, both procedures are conceptually simple but have key complex details. For MCMC, these are (1) the construction of the moves for an irreducible chain, (2) monitoring convergence, and (3) checking sampling efficiency. For SIS, they are (1) computing the support of the conditional distribution of a cell given previously sampled cells and (2) constructing a proposal distribution on that support. The choice to sample from conditional uniform distributions q_k, resulting in the uniform distribution on the fiber $\mathcal{F}_s$ via rejection, alleviates the challenge of the latter entirely; this is the strategy we take for all of the methods described below.

5.2 Hybrid schemes

We now explore different hybrid schemes and their advantages and disadvantages.

1. SIS Initialized MCMCs. The tremendous computational advantage provided by parallel computing with SIS suggests efforts to parallelize the MCMC. A natural way to do so is to run several chains in parallel. It is well-known in Bayesian computing that running multiple chains initialized at different places in the state space can be advantageous (Lunn et al. 2012). One advantage is computational: the chains can be run in parallel in a distributed computing framework at very little extra cost. Another is diagnostic: the many chains can be analyzed with trace plots or even hypothesis tests to assess mixing (Gelman and Rubin 1992); trace plots are described in Sect. 6. One can envision two extremes when using SIS to initialize MCMCs at various points in the fiber. In what follows, let C be the desired number of chains.

(a) *All SIS initializations.* Generate $I = C$ SIS tables and initialize one chain at each table.
Advantages : Chains initialized at different locations on the fiber
- can be used to assess mixing.
- can be used to assess irreducibility.
- via SIS are independent of one another.
- leverage distributed computing capacity.

Disadvantages :
- Since they are uniformly sampled from the fiber, many SIS tables will be regions of low hypergeometric probability. If the null hypothesis is true, $\boldsymbol{t}_o$ is likely to be in the high probability region of $f(\boldsymbol{t}|\boldsymbol{s})$, and so using SIS initializations is likely to increase mixing times.

(b) *Best SIS initialization.* Generate I SIS tables and initialize C chains at the table with the highest hypergeometric probability (SIS or observed), determined by UNLLs.
Advantages :
- The chains mix faster because they start closer to the high probability regions of the hypergeometric distribution on the fiber.
- Many chains leverage distributed computing capacity.

Disadvantages : Chains initialized at the same location
- are less useful in assessing mixing than those initialized at different locations.
- cannot be used to assess irreducibility.
- are more dependent than those initialized at different locations.

2. SIS Moves. SIS can be used to generate in the proposal process to generate moves or augment the move set $\mathcal{M}$. This can be done in two ways:

(a) *Static SIS moves.* SIS can be used to pre-compute an arbitrary number of moves by generating two SIS tables and differencing them. We call these static SIS moves. The collection of static SIS moves generated by M pairs of SIS tables we denote $\mathcal{S}_M$.
Advantages :
- As M increases, the fiber will be connected almost surely since the fiber is finite.
- No additional run-time computations over those of Algorithm 1.

Disadvantages :
- Static SIS moves are typically very large and only admissible for a small collection of points in the fiber, dramatically diminishing efficiency.

(b) *Dynamic SIS moves.* Instead of pre-computing a collection of moves for the MCMC, SIS generated tables can be proposed instead of move-based proposals from $\mathcal{M}$. We call such moves dynamic SIS moves. While technically the move is the implicitly defined difference between the current state and the SIS generated table, we often use the term for the proposed table as well.
Advantages :
- Dynamic SIS moves are always admissible and consequently have much higher acceptance rates than static SIS moves.

– Using dynamic SIS moves results in irreducible chains, regardless of $\mathcal{M}$. This is a consequence of the SIS sampling procedure placing nonzero probability on every outcome.
– Only one SIS table need be generated, as opposed to two for static SIS moves.

Disadvantages :

– Additional run-time computations are required.
– SIS proposals, being roughly uniformly distributed on the fiber, are often far from the mass of the hypergeometric distribution and therefore have high rejection rates, reducing efficiency.

5.3 The proposed scheme

Both of the insertion points of SIS into MCMC, at the initial stage and at the proposal, have advantages and disadvantages. We presented both cases in extremes: for initializing, start *all* chains at different SIS tables or *all* chains at the best one; for moving, propose a move from $\mathcal{M} \cup \mathcal{S}_M$ *every time* or a dynamic SIS move *every time*. However, the most sensible general purpose strategy makes compromises between these extremes.

Technically detailed in Algorithm 3, we propose the following scheme. To initialize the chain, generate I SIS tables and initialize C/K chains at the K tables with the highest UNLL values, where K is some small percentage of I, e.g., 10%. This allows us to run chains initialized at different, perhaps disconnected, components of the fiber, which enables convergence diagnostics while not paying for it too much in terms of mixing. To move the chain, if a Markov basis is available, use it with no SIS embelishments unless mixing appears problematic (e.g., the fiber is very large and the trace plots do not stabilize or blend). If a Markov basis is not known, use a lattice basis with intermittent dynamic SIS moves. That is, at each step, propose a dynamic SIS move with some probability α and a move from $\mathcal{M}$ with probability $1 - \alpha$.

Algorithm 3 strikes a good compromise between the strengths and weaknesses of each of the hybridization mechanisms to provide a good general purpose algorithm, especially when a Markov basis is not available. In particular, it has the following advantages: (1) The resulting estimator of p is asymptotically unbiased even in the event a Markov (sub-)basis is not available, because it uses dynamic SIS moves; (2) it leverages additional computing capacity; and (3) it enables the user to diagnose poor mixing. We consider these from a practical vantage point in Sect. 6.

6 Simulation examples

In this section we perform some concrete simulations to develop an intuition for the various hybrid strategies in Sect. 5. We begin with a simple 4×4 independence model example, where the Markov basis is well known, and introduce criteria with which to judge simulations. We then proceed to exact inference in logistic regression,

Input:
A – the model configuration matrix
The observed table $\boldsymbol{t}_o$
$N = 10,000$ – a desired number of samples from $f(\boldsymbol{t}|\boldsymbol{s}) = f(\boldsymbol{t}|\mathbf{A}\boldsymbol{t}_o)$
$C = N/P$ – the desired number of chains (P is the number of processing units available)
$\alpha = .01$ – the probability of proposing a dynamic SIS move
$I = 100$ – the number of SIS tables generated for initialization
$\beta = .05$ – the proportion of top UNLL SIS tables to use for initialization

Output:
N samples $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_N$ from $f(\boldsymbol{t}|\boldsymbol{s})$

$\mathcal{M} \leftarrow$ lattice basis for **A**, unless a Markov (sub-)basis is known
Generate C SIS tables $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_C$ according to Algorithm 2
Compute UNLLs $\log \widetilde{f}(\boldsymbol{c}_k)$ for $k \in [C]$ and $\log \widetilde{f}(\boldsymbol{t}_o)$
foreach *chain* **do**
  for *i in 1:N/C* **do**
    Sample $u \leftarrow \text{Unif}(0, 1)$
    if $u < \alpha$ **then**
      Sample $\boldsymbol{v}$ via SIS according to Algorithm 2
      Sample $u \leftarrow \text{Unif}(0, 1)$
      if $u < \widetilde{f}(\boldsymbol{v}|\boldsymbol{s})/\widetilde{f}(\boldsymbol{t}_{i-1}|\boldsymbol{s})$ **then**
        $\boldsymbol{t}_i \leftarrow \boldsymbol{v}$
      else
        $\boldsymbol{t}_i \leftarrow \boldsymbol{t}_{i-1}$
      end
    else
      Sample $\boldsymbol{m}$ from $\mathcal{M}$ and e from $\{1, -1\}$ (uniformly)
      Set $\boldsymbol{v} \leftarrow \boldsymbol{t}_{i-1} + e\boldsymbol{m}$
      if $any(\boldsymbol{v} < \mathbf{0}_r)$ **then**
        $\boldsymbol{t}_i \leftarrow \boldsymbol{t}_{i-1}$
      else
        Sample $u \leftarrow \text{Unif}(0, 1)$
        if $u < \widetilde{f}(\boldsymbol{v}|\boldsymbol{s})/\widetilde{f}(\boldsymbol{t}_{i-1}|\boldsymbol{s})$ **then**
          $\boldsymbol{t}_i \leftarrow \boldsymbol{v}$
        else
          $\boldsymbol{t}_i \leftarrow \boldsymbol{t}_{i-1}$
        end
      end
    end
  end
end

Algorithm 3: The proposed hybrid scheme given a collection of moves not known to. Burn-in and thinning are omitted for clarity but are discussed in Sect. 6

which is also studied in Aoki et al. (2012) and where Markov bases are known to be difficult to compute. All the simulations were done in R using the **algstat** package, which implements the MCMC and SIS sampling procedures in C++ using R's **Rcpp** package (Kahle et al. 2015; Eddelbuettel and François 2011; Eddelbuettel 2013); code is available upon request to the authors. The linear programming software used for the SIS samples was R's **lpSolve** package (Berkelaar et al. 2015). Markov and lattice bases are computed through **algstat** using 4ti2's `markov` and `zbasis` programs (4ti2 team 2008).

Table 2 Snee's hair and eye color dataset (Snee 1974)

	Black	Brown	Red	Blond
Brown	68	119	26	7
Blue	20	84	17	94
Hazel	15	54	14	10
Green	5	29	14	16

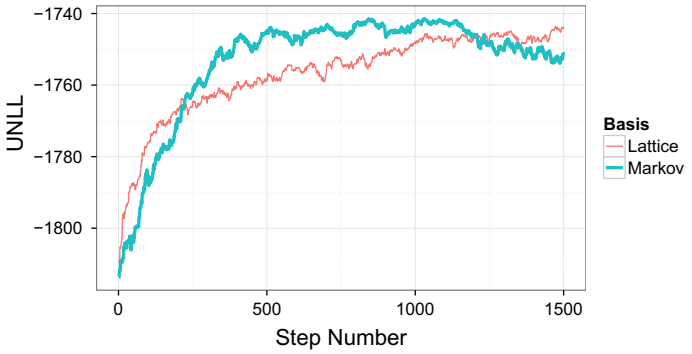

Fig. 6 A single simulated MCMC chain using lattice and Markov basis moves

6.1 Independence model

To get a better feel for how we will use simulations to compare strategies, we begin by studying a simple dataset, Snee's hair and eye color in dataset in Table 2, with the 4×4 independence model. The lattice and Markov bases of the configuration matrix of the model, $\mathbf{A} \in \{0, 1\}^{8\times 16}$, are well known but distinct. The lattice basis contains nine $\begin{smallmatrix} + & - \\ - & + \end{smallmatrix}$ moves, while the Markov basis contains 36 such moves. The fiber, referred to in Sect. 2, has over 10^{15} elements.

Assessing the quality of different MCMC strategies via simulation can be difficult because they are stochastic processes. Nevertheless looking at trace plots is a standard useful technique. A trace plot of a MCMC is simply a line graph illustrating some feature of a sampled instance of the chain against the step count. In Bayesian statistics, the feature plotted is typically the sampled value itself, representing a marginal sample from a joint posterior distribution. In Fig. 6 we plot the UNLLs of the tables sampled in the chain against the step count. These chains have no burn-in (discarded initial samples) and no thinning (discarded iterations); they are precisely as described in Algorithm 1. We run the chains for a small number of iterations for illustration purposes.

Comparing the trace plots of a single MCMC run using both methods, as in Fig. 6, is not enough to assess how the methods perform in general, even for a single dataset and model. To make a more valid comparison, we run several chains and superimpose the trace plots. In practice, convergence of a single chain is declared when the plots level off and look like a "fat hairy caterpillar" (Lunn et al. 2012, p. 73). When running

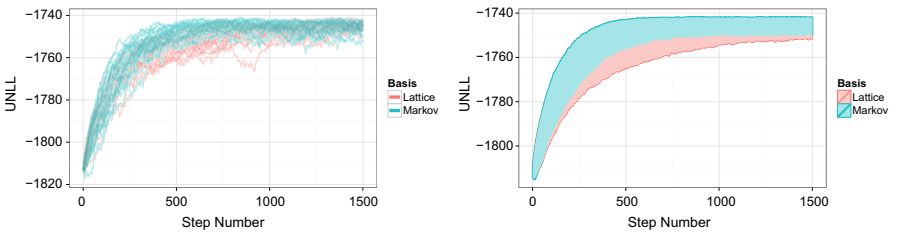

Fig. 7 (*Left*) 25 LB and MB-MCMCs initialized at the observed table, communicating a sense of the variability and mixing times of the chains. The MB-MCMCs appear to mix more quickly. (*Right*) Cross-sectional 95% intervals of the distribution of LB and MB-MCMCs based on 1000 simulations each, confirming the findings from the left graphic

multiple chains, convergence is declared when the chains localize to the same general region of space, thus the term "mixing". Figure 7 (left) illustrates this, with the Markov basis appearing to have converged after about 1,000 iterations and the lattice basis converging a little later. Thus, a burn-in of a about 1,000 would be reasonable for MB-MCMC and somewhat more for LB-MCMC. To ease the visual, we can make ribbon plots that show cross-sectional percentile-based intervals of the distribution for each of the steps; it is illustrated in Fig. 7 (right).

We can run the chains for longer to be more convinced of convergence, but Fig. 7 already suffices to communicate the message that while the lattice and Markov bases work very similarly, the Markov basis MCMC mixes slightly faster than the lattice basis. Of course, this is practically immaterial, since the entire simulation is so fast that Fig. 7, which included thousands of chains run in sequence, can be computed in less than 10 seconds on a standard laptop.

Another area of interest is the efficiency of the sampler. A MCMC sampler is said to be efficient if the autocorrelation between its samples is low and inefficient if it is high. High autocorrelation is bad for a MCMC because (1) it suggests that the sampler may not be fully exploring the support of the probability distribution, and consequently may be providing un-representative samples from the distribution that can lead to biased estimates of its functionals like p and (2) the samples, being correlated, cannot be used with the simple central limit theorem formula to obtain standard errors of estimated functionals. Poor efficiency can often be diagnosed from trace plots as the kind of "snaking" seen in Fig. 6 that makes it look nearly continuous. After diagnosed, it is alleviated by thinning, a post-processing technique used on MCMCs intended to reduce the autocorrelation of chains by retaining multiples k of the step index t, e.g., the 100th, 200th, 300th, samples and so on. In practice the value of k is determined by simply re-checking diagnostic plots for indications of autocorrelation.

The more robust diagnostic strategy for inefficiency in a MCMC sampler is autocorrelation (function) plots, also called ACF plots, borrowed from time series analysis (Lunn et al. 2012). These plots visually communicate the autocorrelation of a time series at several lag values. Figure 8 (left) shows the combined ACF plots of the LB- and MB-MCMC chains as determined by averaging the autocorrelations at various lag values across 1000 simulated chains with 1500 iterations each. It indicates that both chains exhibit significant autocorrelation and are thus quite inefficient, likely due to the

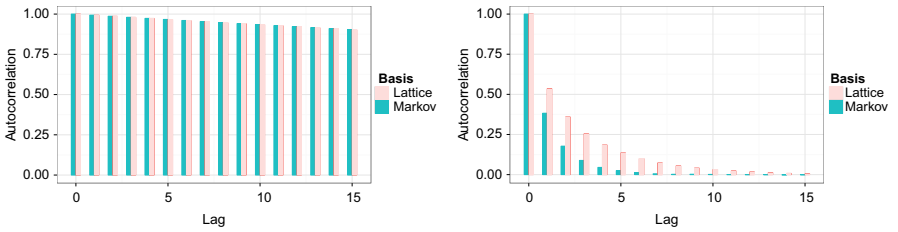

Fig. 8 (*Left*) Average autocorrelations across 1000 simulated chains with 1500 iterations indicating very strong autocorrelation and therefore inefficient chains. (*Right*) Average autocorrelations across 1000 simulated chains with 1500 iterations and 100 step thinning indicating that the MB-MCMC is more efficient than the LB-MCMC

fact that neither basis contains large moves. Thinning by 100 (keeping only index multiples of 100) yields Fig. 8 (right), which illustrates that, in addition to mixing faster, the MB-MCMC is also more efficient than the LB-MCMC. In further simulations (not shown), the MB-MCMC was found to require a thinning of roughly 250 samples to eliminate autocorrelation, whereas the LB-MCMC required a thinning of about 750.

What about SIS initializations? Such initializations are beneficial because samples from different chains are fully independent by construction, so if samples are generated by taking the last values of each of the chains, no thinning is required. Figure 9 shows 100 LB-MCMC chains initialized at 100 SIS tables. Notice the dramatic difference the initial table makes in terms of mixing: depending on the initial table, mixing can be achieved almost instantly, eliminating the need for a burn-in, or delayed substantially, requiring a much larger burn-in. The LB- and MB-MCMC chains from Fig. 6 are included for visual reference.

This illustrates the trade off between using all SIS initialized chains and mixing time described in Sect. 5. The higher the UNLL of the initial value of the chain, the less the mixing time. This suggests that if one were to initialize chains not at every SIS table, but only at those with the highest UNLLs as the proposed method does, one might do better. For example, if one wants 100 samples, one might sample 100 SIS tables, evaluate the UNLL of each, and initialize 20 chains at each of the top five tables; this is illustrated in Fig. 10. Since the burn-in is typically longer than the thinning, the last values of each of the 20 chains starting at the same initial values can likely be considered independent; the last values of different chains from different initial points are independent by construction. Moreover, a Gelman-Rubin test comparing within-chain variability to between-chain variability could be used to test for convergence (Gelman and Rubin 1992; Brooks and Gelman 1998).

6.2 Logistic regression

Logistic regression is one of the most widely used tools in the applied statistician's toolbox. In order to specify a logistic regression in terms of a configuration matrix, it helps to first introduce Poisson regression. It is a straightforward exercise to show that the sufficient statistics for the coefficients of a univariate Poisson regression with a discrete covariate with J equally spaced levels is given by the configuration matrix

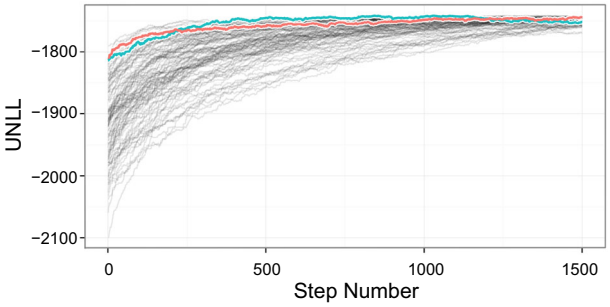

Fig. 9 100 LB-MCMC chains initialized at random SIS tables. The LB- and MB-MCMC chains near the top of the graphic indicate they mix quickly. Relative to the rest of the tables in the fiber, the observed table's probability is quite high, but its p value is still essentially 0 because while the fiber is large, the hypergeometric distribution on it is quite concentrated

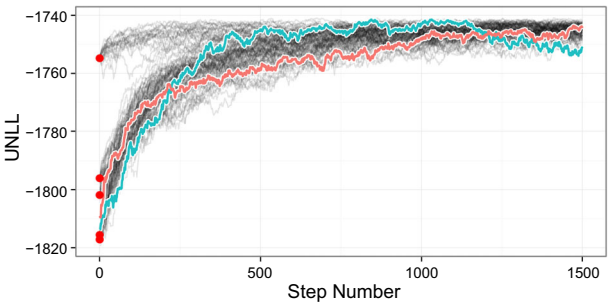

Fig. 10 Trace plots of chains initialized at the five top UNLL SIS tables out of 100 samples. The *red points* represent the SIS initial tables

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \cdots 1 \\ 1 & 2 \cdots J \end{bmatrix}.$$

The configuration matrix of the logistic regression of a binary variable Y onto a variable X with J equally spaced levels is provided by the Lawrence lifting of the Poisson regression configuration matrix $\mathbf{A}$. The Lawrence lifting of a configuration matrix $\mathbf{A}$ is defined

$$\Lambda(\mathbf{A}) := \begin{bmatrix} \mathbf{A} & \mathbf{0}_{d\times r} \\ \mathbf{I}_{r\times r} & \mathbf{I}_{r\times r} \end{bmatrix},$$

where the dimensions of the identity and zero matrices are given by the subscripts. The Lawrence lifting serves to constrain the total number of observations at each level (e.g., 20 *out of 100*). If $\mathbf{A}$ is the configuration matrix of a Poisson regression with J covariate levels, $\Lambda(\mathbf{A})$ is the configuration matrix of the logistic regression (Hara et al. 2012). The kth Lawrence lifting of $\mathbf{A}$ is

Table 3 The "big" logistic dataset with $n = 100$ observations at each design point

	1	2	3	4	5	6	7	8	9	10
0	96	91	92	84	89	85	88	83	85	79
1	4	9	8	16	11	15	12	17	15	21
n_x	100	100	100	100	100	100	100	100	100	100

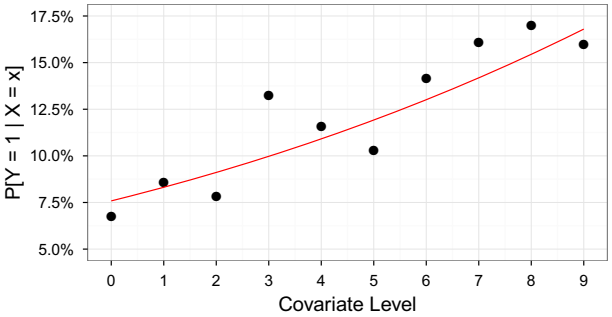

Fig. 11 The probabilities used to generate the logistic regression data in Tables 3 and 4 with the logistic model they were designed to mimic

$$\Lambda^k(\mathbf{A}) := \begin{bmatrix} \mathbf{A} & & & \\ & \mathbf{A} & & \\ & & \ddots & \\ & & & \mathbf{A} \\ \mathbf{I} & \mathbf{I} & \mathbf{I} & \mathbf{I}\ \mathbf{I} \end{bmatrix}$$

where there are k $\mathbf{A}$ matrices and the blanks are filled in by zero matrices. The kth Lawrence lifting of a Poisson regression configuration matrix corresponds to a logistic regression with $(k + 1)$ response levels.

It is well known that computing the Markov basis of Lawrence liftings is hard; however, their lattice bases are trivial to compute (Chen et al. 2005b; Hara et al. 2010, 2012). In this section we consider logistic regression with two datasets, both with $J = 10$ levels but with different designs yielding different sized cells and fibers. $J = 10$ is near the cusp of computability of the Markov basis of $\Lambda(\mathbf{A})$. When $J = 10$ the Markov basis has 1,830 elements, and it grows rapidly from there—at $J = 16$ it has about 125,000 elements (Hara et al. 2012). Thus, the model is representative of situations where a Markov basis would not be available. By contrast, the lattice basis of the model consists of only 8 moves.

The first dataset we will consider is the "big" table in Table 3; in it $n = 100$ observations are seen at each design point. Its fiber, computed with LattE through **algstat**, contains 75,565,950,866 tables (Baldoni et al. 2014). The "small" table samples fewer observations at larger design values as might be the case in a dose-response clinical trial; its fiber has 810 tables. Both datasets were sampled using a logistic model with $\beta_0 = -2.5$ and $\beta_1 = .1$ subject to some noise. These probabilities are shown in Fig. 11. Thus, they are both designed to mimic logistic behavior well but not exactly

Table 4 The "small" logistic dataset with shrinking numbers observations at higher design points as might be expected in dose-response clinical trials

	1	2	3	4	5	6	7	8	9	10
0	19	20	15	8	10	8	5	5	4	2
1	1	0	0	2	0	0	3	0	1	1
n_x	20	20	15	10	10	8	8	5	5	3

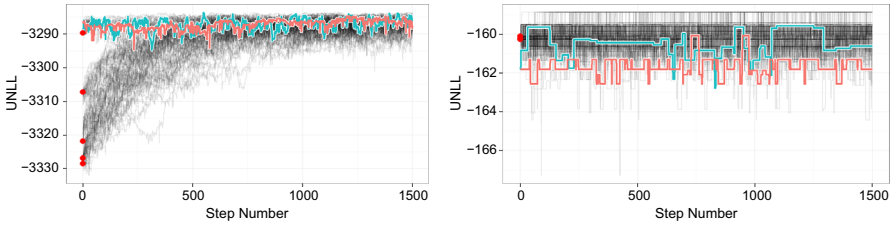

Fig. 12 20 LB-MCMC chains started at each of the top 5 UNLL SIS tables out of 100 sampled for Table 3 (*left*) and Table 4 (*right*). The colored chains are the LB- and MB-MCMC chains started at the observed table with the *same color* scheme as Fig. 7

(i.e., they exhibit more than simple sampling error). It is known that the lattice basis does not connect the fiber of the small table.

The proposed method in Algorithm 3 (without dynamic SIS moves as they were unnecessary) is used for exact inference for both assuming a Markov basis is not known. Specifically, we used LB-MCMC initialized at the top 5 UNLL SIS tables (out of 100) with 20 chains at each. Trace plots are shown in Fig. 12. The SIS tables are generated in a matter of seconds on a standard laptop parallelized across the cores, and the chains, also run in parallel, are run with no burn-in and no thinning. Convergence for the big dataset is clear in Fig. 12 (left); it is confirmed by running a MB-MCMC (which is possible here) for hundreds of millions of iterations. It is less clear for the small dataset in Fig. 12 (right), which has a very small fiber, but the fact that the chains are all in the same general locale is suggestive of convergence. Moreover, no intermittent jumping is observed from dynamic SIS moves. The autocorrelation is clearly strong in both graphics; however, values sampled from the ends of the chains are likely sufficiently independent as to be considered as such.

Taking the last values of the 100 total chains in both situations, we find $p \approx .63$ for the "big" table and $p \approx .06$ for the "small" table. The correct values, based on MB-MCMC with a million burn-in iterations, 250 thinning with clean ACF plots, and 10,000 samples was .5784 and .1307, respectively. Thus, the routines provide reasonable numbers up to Monte Carlo error in both cases.

Obviously, the 100 samples from the hypergeometric distribution on the fibers would be increased for real-world scenarios, the current simulation is more of a proof-of-concept for illustration purposes. The implementation used to sample the SIS tables was not optimized, but sampling 100 tables in both situations took only a few seconds, which seems to be a small price to pay relative to running the chains for longer. Thus, using LB-MCMC run in parallel at top-ranked SIS inits performed very well for both of the logistic regression scenarios; adding dynamic SIS proposals rarely at random

intervals would slightly decrease efficiency but add desirable theoretical guarantees in the limiting distribution. As previously noted, we recommend this strategy in real-world scenarios with a lot at stake.

7 Discussion

In this article we have introduced and illustrated novel hybrid schemes for exact conditional inference in discrete exponential families that combine many of the advantages of both existing Monte Carlo frameworks. The scheme leverages SIS for its parallelizability and independent samples and MCMC for distribution correctness and run-time speed. The resulting whole is greater than the sum of its parts: the resulting scheme can be used to diagnose convergence and reducibility problems of the MCMCs without having to worry excessively about the proposal distributions of either method, and it provides desirable asymptotic properties. On two examples, one relatively simple and one significantly more complicated, the method performed very well.

What about in general? For the most part, we expect the method to work well in situations where Markov bases are not available. Intuitively speaking, for many contingency tables and models if a LB-MCMC fiber graph is disconnected it is likely that it is composed of at most a few large disconnected components, perhaps only one, and many small isolated components on the periphery, as in Fig. 3. If this is the case, as long as the hypergeometric distribution is concentrated on the interior of a single connected component, the SIS initialized LB-MCMC procedure will likely work very well for estimating p and diagnostic plots can be used to assuage any convergence concerns, discarding samples of chains that are found to not be on the focal component. Intermittent dynamic SIS proposals can be added in suspect situations. If the distribution is concentrated on a small number of components of non-negligible probability, which could be identified using the diagnostics, one could augment the lattice basis with specially designed difference moves to connect the relevant components or perhaps even using the very rare SIS proposals described in Sect. 5. Thus, the method appears to be sufficiently flexible to address a broad array of situations.

That being said, Algorithm 3 is not a panacea: running a MCMC without a connected state space always runs the risk of error. We would expect the method to fail if the cells are extremely small, the table is sparse, or the model/table combination is complicated. In fact, any one of these three may threaten the routine with spectacular failure. In these kinds of circumstances, the lattice basis simply does not have the diversity of moves to create even a remotely connected fiber. Dynamic SIS moves can alleviate the situation, but may be sufficiently inefficient as to be practically useless. Such situations appear to elude every known method, and heuristics would have to be used if an estimate is demanded. If a Markov basis is available SIS initialized MB-MCMC would seem always preferable to SIS initialized LB-MCMC. However, interestingly, this is not always the case: we have observed on more than one occasion MB-MCMC chains much less efficient than LB-MCMC chains. We believe this is because in those situations the lattice basis connected the fiber graph at hand while the Markov basis, which would apply broadly to any dataset, contained a bounty of rarely useful moves and not needed. Consequently, if MB-MCMC performance is poor, a LB-MCMC

provides a reasonable alternative, or even up-weighting the selection probabilities of the lattice basis moves in the Markov basis at the time of proposal.

Apart from the theory, a major factor determining the practical efficiency of the routine is its implementation. The current implementation uses R's **algstat** package, which has custom C++ MCMC and SIS implementations that are very fast. However, the connection within the latter to R's **lpSolve** package for the linear programs is not currently not optimized (it has to run back through R), and consequently any concrete time benchmarkings comparing the SIS initialized MCMC scheme to simply running a MCMC longer would not make for a valid comparison. It suffices here to make two points. First, the top-SIS hybrid routine is very fast, even when not optimized. All the simulations in this work put together can be done in less than a minute parallelized across the cores of a standard four-core laptop (not counting hyperthreading). Second, when MB-MCMC is not available, SIS initialized LB-MCMC is clearly a far superior strategy to standard LB-MCMC because of the added insight gained from running several chains initialized at disparate points in the fiber. In the future, providing **algstat** with access to a fast linear program solver at a very low level, preferably one tailored to the kinds of problems encountered in discrete multivariate analysis, would make the routine even faster. From there, one can imagine distributing the computations across the thousands of cores of a graphical processing unit (GPU). This step would really leverage the distributed computing framework to its fullest extent and would likely speed up the process by orders of magnitude.

Even after nearly two decades of intense research in the algebraic statistics community, questions concerning both the theory and practice of Monte Carlo schemes for exact conditional inference abound. On the theory side, questions exist for both SIS and MCMC procedures. For SIS, one might ask if there are better ways to try to obtain samples from the hypergeometric distribution on the fiber – explicit conditional distributions and the like. Caffo and Booth provided insight into this problem via a MCMC within SIS scheme and even provided an implementation (Caffo and Booth 2001; Caffo 2013). It would be interesting to see a comparison between the methods proposed here and those strategies. For MCMC, additional work into the computation of Markov subbases, collections of moves that connect specific fibers as opposed to any fiber for a particular model would also be helpful (Chen et al. 2006). On the application side, all these methods need practical implementations to become more mainstream. Preferably, such implementations would be general purpose, allowing for exact inference in a range of popular models such as log-linear models (perhaps even with graphical specification), logistic and Poisson regressions, and rank data models.

The hybrid schemes suggested in this article inherit these questions and poses others: can we characterize the mixing behavior of the schemes?; when using the top UNLL chains, what proportion should be used?; with what probability should dynamic SIS moves be proposed?; and what is the best way to implement adaptive strategies that abandon bad chains and focus on good ones? We eagerly anticipate further work in this area.

In closing, it should be noted that the methods described and the lessons learned in this article are not limited to conditional inference. Many routines of interest on fibers—for example optimization or enumeration—can also be approached with SIS

initialized MCMCs. It would be interesting to see strategies like these applied in those contexts.

Acknowledgements D. K. and R. Y. are supported by the National Science Foundation under Grant Nos. 1622449 and 1622369, respectively. The authors would like to thank an anonymous referee for suggesting that the validity of the scheme should be considered through the lens of unbiasedness.

References

Agresti, A. (1992). A survey of exact inference for contingency tables. *Statistical Science*, *7*(1), 131–153.
Agresti, A. (2002). *Categorical data analysis* (2nd ed.). Hoboken: Wiley.
Aoki, S., Hara, H., Takemura, A. (2012). *Markov bases in algebraic statistics* (Vol. 199). New York: Springer.
Baldoni, V., Berline, N., De Loera, J., Dutra, B., Koppe, M., Moreinis, S., Pinto, G., Vergne, M., Wu, J. (2014). A user's guide for LattE integral v1.7.2. URL: http://www.math.ucdavis.edu/~latte/.
Bélisle, C. J., Romeijn, H. E., Smith, R. L. (1993). Hit-and-run algorithms for generating multivariate distributions. *Mathematics of Operations Research*, *18*(2), 255–266.
Berkelaar, M., Eikland, K., Notebaert, P. (2015). lpSolve: Interface to `Lp_solve` v.5.5 to solve linear/integer programs. http://CRAN.R-project.org/package=lpSolve, R package version 5.6.11.
Bishop, Y. M. M., Fienberg, S. E., Holland, P. W. (1975). *Discrete multivariate analysis*: *Theory and practice*. Cambridge: The MIT Press.
Booth, J. G., Butler, R. W. (1999). An importance sampling algorithm for exact conditional tests in log-linear models. *Biometrika*, *86*(2), 321–332.
Boyett, J. M. (1979). Algorithm as 144: Random r× c tables with given row and column totals. *Journal of the Royal Statistical Society Series C-Applied Statistics*, *28*(3), 329–332.
Brooks, S. P., Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, *7*(4), 434–455.
Caffo, B. (2013). exactLoglinTest: Monte Carlo exact tests for log-linear models. http://CRAN.R-project.org/package=exactLoglinTest, R package version 1.4.2.
Caffo, B. S., Booth, J. G. (2001). A Markov chain Monte Carlo algorithm for approximating exact conditional probabilities. *Journal of Computational and Graphical Statistics*, *10*(4), 730–745.
Chen, Y., Diaconis, P., Holmes, S. P., Liu, J. S. (2005a). Sequential monte carlo methods for statistical analysis of tables. *Journal of the American Statistical Association*, *100*(469), 109–120.
Chen, Y., Dinwoodie, I., Dobra, A., Huber, M. (2005b). Lattice points, contingency tables, and sampling. *Contemporary Mathematics*, *374*, 65–78.
Chen, Y., Dinwoodie, I., Sullivant, S. (2006). Sequential importance sampling for multiway tables. *The Annals of Statistics*, *34*(1), 523–545.
Clarkson, D. B., Fan, Y., Joe, H. (1993). A remark on algorithm 643: Fexact: An algorithm for performing fisher's exact test in RXC contingency tables. *ACM Transactions on Mathematical Software*, *19*(4), 484–488.
Cox, D., Little, J., O'Shea, D. (1997). *Ideals, varieties, and algorithms* (2nd ed.). New York: Springer.
De Loera, J., Onn, S. (2005). Markov bases of three-way tables are arbitrarily complicated. *Journal of Symbolic Computation*, *41*(2), 173–181.
Diaconis, P., Sturmfels, B. (1998). Algebraic algorithms for sampling from conditional distributions. *The Annals of Statistics*, *26*(1), 363–397.
Dobra, A. (2003). Markov bases for decomposable graphical models. *Bernoulli*, *9*(6), 1093–1108.
Dobra, A., Sullivant, S. (2004). A divide-and-conquer algorithm for generating Markov bases of multi-way tables. *Computational Statistics*, *19*, 347–366.
Drton, M., Sturmfels, B., Sullivant, S. (2009). *Lectures on algebraic statistics*. Boston: Birkhauser Basel.
Eddelbuettel, D. (2013). *Seamless R and C++ integration with Rcpp*. New York: Springer.
Eddelbuettel, D., François, R. (2011). Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, *40*(8), 1–18.
Fisher, R. A. (1922a). On the interpretation of $\chi 2$ from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society, 85*(1), 87–94.

Fisher, R. A. (1922b). On the mathematical foundations of theoretical statistics. *Philosophical transactions of the royal society of London series A—Containing papers of a mathematical or physical character* (pp. 309–368).
Fisher, R. A. (1934). *Statistical methods for research workers* (5th ed.). Edinburgh: Oliver & Boyd.
Gelman, A., Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, *7*(4), 457–472.
Halton, J. H. (1969). A rigorous derivation of the exact contingency formula. *Mathematical Proceedings of the Cambridge Philosophical Society*, *65*(02), 527–530.
Hara, H., Takemura, A., Yoshida, R. (2010). On connectivity of fibers with positive marginals in multiple logistic regression. *Journal of Multivariate Analysis*, *101*(4), 909–925.
Hara, H., Aoki, S., Takemura, A. (2012). Running Markov chain without Markov basis. In *Proceedings of the second CREST-SBM international conference, Harmony of Gröbner bases and the modern industrial society*, Singapore (pp. 19–34).
Kahle, D., Garcia-Puente, L., Yoshida, R. (2015). algstat: Algebraic statistics in R. http://CRAN.R-project.org/package=algstat, R package version 0.1.0.
Kahle, T., Rauh, J. (2011). The Markov bases database. http://www.markov-bases.de.
Lange, K. (2010). *Numerical analysis for statisticians* (2nd ed.). New York: Springer.
Lehmann, E. L., Romano, J. P. (2005). *Testing statistical hypotheses* (3rd ed.). New York: Springer.
Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. New York: Springer.
Lunn, D., Jackson, C., Best, N., Thomas, A., Spiegelhalter, D. (2012). *The BUGS book*: *A practical introduction to Bayesian analysis*. Boca Raton: CRC Press.
Mehta, C. R., Patel, N. R. (1986). Algorithm 643: Fexact: A Fortran subroutine for fisher's exact test on unordered r× c contingency tables. *ACM Transactions on Mathematical Software*, *12*(2), 154–161.
Patefield, W. M. (1981). Algorithm as 159: An efficient method of generating random r× c tables with given row and column totals. *Journal of the Royal Statistical Society Series C-Applied Statistics*, *30*(1), 91–97.
Pearson, K. (1900). On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine Series 5*, *50*(302), 157–175.
R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/.
Read, T. R., Cressie, N. (1988). *Goodness-of-fit statistics for discrete multivariate data*. New York: Springer.
Schrijver, A. (1986). *Theory of linear and integer programming*. Chichester: Wiley.
Sheskin, D. J. (2007). *Handbook of parametric and nonparametric statistical procedures* (4th ed.). Boca Raton: Chapman and Hall/CRC Press.
Snee, R. D. (1974). Graphical display of two-way contingency tables. *The American Statistician*, *28*(1), 9–12.
Snijders, T. (1991). Enumeration and simulation methods for 0–1 matrices with given marginals. *Psychometrika*, *56*(3), 397–417.
Sturmfels, B. (1996). *Gröbner bases and convex polytopes* (Vol. 8). Providence: American Mathematical Society.
4ti2 team (2008). 4ti2—A software package for algebraic, geometric and combinatorial problems on linear spaces. http://www.4ti2.de.