2013

Polytopes Arising from Binary Multi-way Contingency Tables and Characteristic Imsets for Bayesian Networks

Jing Xi
University of Kentucky, tykiallen@gmail.com

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained and attached hereto needed written permission statements(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine).

I hereby grant to The University of Kentucky and its agents the non-exclusive license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless a preapproved embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's dissertation including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Jing Xi, Student

Dr. Ruriko Yoshida, Major Professor

Dr. Constance Wood, Director of Graduate Studies

Polytopes Arising from Binary Multi-way Contingency Tables and Characteristic Imsets for Bayesian Networks

DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy in the
College of Arts and Sciences
at the University of Kentucky

By
Jing Xi
Lexington, Kentucky

Director: Dr. Ruriko Yoshida, Professor of Statistics
Lexington, Kentucky

2013

ABSTRACT OF DISSERTATION

Polytopes Arising from Binary Multi-way Contingency Tables and Characteristic Imsets for Bayesian Networks

The main theme of this dissertation is the study of polytopes arising from binary multi-way contingency tables and characteristic imsets for Bayesian networks.

Firstly, we study on three-way tables whose entries are independent Bernoulli random variables with canonical parameters under no three-way interaction generalized linear models. Here, we use the sequential importance sampling (SIS) method with the conditional Poisson (CP) distribution to sample binary three-way tables with the sufficient statistics, i.e., all two-way marginal sums, fixed. Compared with Monte Carlo Markov Chain (MCMC) approach with a Markov basis (MB), SIS procedure has the advantage that it does not require expensive or prohibitive pre-computations. Note that this problem can also be considered as estimating the number of lattice points inside the polytope defined by the zero-one and two-way marginal constraints. The theorems in Chapter 2 give the parameters for the CP distribution on each column when it is sampled. In this chapter, we also present the algorithms, the simulation results, and the results for Samson's monks data.

Bayesian networks, a part of the family of probabilistic graphical models, are widely applied in many areas and much work has been done in model selections for Bayesian networks. The second part of this dissertation investigates the problem of finding the optimal graph by using characteristic imsets, where characteristic imsets are defined as 0-1 vector representations of Bayesian networks which are unique up to Markov equivalence. Characteristic imset polytopes are defined as the convex hull of all characteristic imsets we consider. It was proven that the problem of finding optimal Bayesian network for a specific dataset can be converted to a linear programming problem over the characteristic imset polytope [51]. In Chapter 3, we first consider characteristic imset polytopes for all diagnosis models and show that these polytopes are direct product of simplices. Then we give the combinatorial description of all edges and all facets of these polytopes. At the end of this chapter, we generalize these results to the characteristic imset polytopes for all Bayesian networks with a fixed underlying ordering of nodes.

Chapter 4 includes discussion and future work on these two topics.

KEYWORDS: Sequential importance sampling, Conditional Poisson, Counting problem, Learning Bayesian networks, Characteristic imset polytope

Author's Signature: Jing Xi

Date: June 11, 2013

Polytopes Arising from Binary Multi-way Contingency Tables and Characteristic Imsets for Bayesian Networks

By
Jing Xi

Director of Dissertation: Dr. Ruriko Yoshida

Director of Graduate Studies: Dr. Constance Wood

Date: June 11, 2013

ACKNOWLEDGMENTS

So many people helped me during my graduate study and the period of writing this thesis. The first thing that comes to my mind is always my gratitude to my parents for their consistent concern about me. Although they don't know much about statistics and math, they always try their best to help me and teach me how to get along with other people, and I know that of all people in the world, no matter what happens, they will always be by my side.

I would like to express my appreciation to my advisor, Dr. Ruriko Yoshida, for her constant help and guidance. She likes her research so much that I am always affected by her passion and consider her as a personal example. I am also impressed by her ability of collaborating with different people and performing as a bridge among math, statistics, and other research areas. She is my good friend and took care of me like a big sister. I really thank her for introducing me into her personal network of colleagues. She tried so hard pushing me to be outgoing and integrating me into the community of her research area. Frankly speaking, my academic career wouldn't start without her encouragement and motivation.

I am very grateful to Dr. Bernd Sturmfels, Dr. Milan Studený and Dr. Raymond Hemmecke who gave me so many valuable ideas and comments about my research. Even though they are very busy, they still took their personal time listening to me so carefully and answering my questions with so much patience. I really owe them a lot. Many other people, like Dr. Seth Sullivant, Dr. Matthew Schofield, Dr. Uwe Nagel, Dr. Constance Wood and my other committee members, also had nice conversations with me which improved my results and gave me many wonderful new ideas. I won't be able to finish this thesis without their inspiration and encouragement.

I would also like to thank Dr. Richard Kryscio who supported me for one and a half years as a member of the consulting lab in Biostatistic Department. He taught me a lot about how to communicate and work with biologists and therapists, and most important, the way to translate their requests into statistical problems. I believe these experiences and skills will be very helpful to me in the future.

My appreciations also go to all of my friends in U.S. and China. I thank them for being with me to get through the difficult time I had when I first came to U.S., and all their help and the fun they had brought to me in these years.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

1.1 Sequential importance sampling with conditional Poisson distribution

Zero-one tables are widely used in many areas. For example, they are used to represent relational data in social networks [35], data in educational / psychological tests (say, Rasch model in [43]), occurrence matrices in ecological studies [9], etc. Zero-one tables are part of sparse contingency tables. It commonly occurs that the number of variables grows faster than the sample size, and those goodness-of-fit tests which are usually performed based on large sample approximation to the null distribution of test statistics (such as Pearson's χ^2 statistic and likelihood ratio G^2 statistic) may be poor because many expected cell counts are small or even zero [29]. To deal with this issue, we propose to estimate the p-values of goodness-of-fit tests by sampling tables. One can find applications of sampling zero-one constrained contingency tables in combinatorics [31], statistics of social networks [8, 50], and regulatory networks [20]. We are going use an example to illustrate how to estimate the p-values of goodness-of-fit tests via sampling tables.

Table 1.1 gives an example of occurrence matrix for Darwin's Finch Data [9] where the rows correspond to species and the columns correspond to geological locations. If one species presents at one location, then the corresponding cell has entry "1", otherwise the entry is "0". Some other occurrence matrices can be found in [12]. A question that ecologists may ask is "Is the pattern of occurrence of finches on the islands a result of chance, or is there an affection of competitive pressures?". Translated into statistical language, the question becomes "Is there an interaction between the adaptability of species and environment of islands?", i.e. "Are these

Table 1.1: Occurrence Matrix for Darwin's Finch Data

	Island																
Finch	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
Large ground finch	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
Medium ground finch	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0
Small ground finch	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0
Sharp-beaked ground finch	0	0	1	1	1	0	0	1	0	1	0	1	1	0	1	1	1
Cactus ground finch	1	1	1	0	1	1	1	1	1	1	0	1	0	1	1	0	0
Large cactus ground finch	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0
Large tree finch	0	0	1	1	1	1	1	1	1	0	0	1	0	1	1	0	0
Medium tree finch	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Small tree finch	0	0	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0
Vegetarian finch	0	0	1	1	1	1	1	1	1	1	0	1	0	1	1	0	0
Woodpecker finch	0	0	1	1	1	0	1	1	0	1	0	0	0	0	0	0	0
Mangrove finch	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Warbler finch	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Island name code: A = Seymour, B = Baltra, C = Isabella, D = Fernandina, E = Santiago, F = Rábida, G = Pinzón, H = Santa Cruz, I = Santa Fe, J = San Cristóbal, K = Española, L = Floreana, M = Genovesa, N = Marchena, O = Pinta, P = Darwin, Q = Wolf.

two variables independent?". Considering a null hypothesis to be the independence between the two variables, the sufficient statistics will be the row sums and column sums [2]. Hence, under this null hypothesis, the observed table can be considered as an observation sampled from a uniform distribution over the set of all possible zero-one tables with the same row and column sums. Many test statistics were suggested for this hypothesis over the past few decades [44, 47]. We will take the one suggested by Roberts and Stone in [44] to illustrate how to carry out these tests via sampling contingency tables, and other tests can be carried out similarly based on different test statistics. The procedure is as following: first, we sample $\mathbf{X_1}, \ldots, \mathbf{X}_{\mathfrak{N}}$ i.i.d. and uniformly from Σ, where Σ is the set of all zero-one tables which share the same row sums and column sums with $\mathbf{x_0}$, the observed table; second, since the test statistic proposed in [44] is defined as

$$\bar{S}^2(\mathbf{X}) = \frac{1}{m(m-1)} \sum_{i \neq j} s_{ij}^2$$

where m is the number of species, s_{ij} is the ith row and jth column element in matrix $\mathbf{X}\mathbf{X}^T$ where $\mathbf{X}$ is the occurence matrix, the **conditional inference p-value** (see more details in Section 1.1.1) will be defined as the expected value of an indicator function based on this test statistic, i.e. $\mathbb{E}_p[\mathbf{1}_{\bar{S}^2(\mathbf{X}) \geq \bar{S}^2(\mathbf{x_0})}(\mathbf{X}) | \text{fixed row sums and column sums}]$ where $p(\cdot)$ is the hypergeometric distribution, which degenerate to the uniform distribution in this case, on Σ; last, we approximate the conditional inference p-value using $\frac{1}{\mathfrak{N}} \sum_{i=1}^{\mathfrak{N}} \mathbf{1}_{\bar{S}^2(\mathbf{X_i}) \geq \bar{S}^2(\mathbf{x_0})}(\mathbf{X_i})$ which is an unbiased estimator of the conditional inference p-value [9].

In Section 1.1, we will first review the general idea of how to use sequential importance sampling (SIS) procedure to sample contingency tables with linear constraints. Then we recall the concept of conditional Poisson (CP) distribution and explain how to apply it in SIS procedures. Next we will introduce the main results in [9] and [8] on how to use SIS procedures with CP distribution to sample zero-one two-way tables

with fixed row sums and column sums. We will end this section with an algebraic geometric view of SIS procedure.

1.1.1 Sequential importance sampling (SIS)

In this section, we are going to illustrate how SIS procedures are used to sample contingency tables with linear constraints which come from the sufficient statistics of a specific model, and what kind of advantages these SIS procedures have.

Consider a contingency table $\mathbf{X}$ which can be vectorized as $\mathbf{X} = (x_1, \cdots, x_t)$, where t is the number of cells in $\mathbf{X}$. Suppose the cell counts $x_1, \ldots, x_t$ are independent Poisson random variables, and the expected frequencies are $\mu_1, \ldots, \mu_t$ for the t cells, respectively. Then a **log-linear model** for contingency tables is that $\exists$ a sequence of constants $\mathbf{h} = (h_1, \ldots, h_t) \in \mathbb{R}^t$, a matrix of integers $\mathcal{A} = (a_{ij})_{n_\lambda \times t} \in \mathbb{Z}^{n_\lambda \times t}$ such that $\mathbf{1}_t^T$ is in the row span of $\mathcal{A}$, and a vector of parameters $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_{n_\lambda}) \in \mathbb{R}^{n_\lambda}$ satisfying

$$log\mu_j = h_j + \sum_{i=1}^{n_\lambda} a_{ij}\lambda_i \ , \ j = 1, \ldots, t. \tag{1.1.1}$$

Note that Equation (1.1.1) gives a generalization of the well known form of the saturated loglinear model for two-way $m \times n$ contingency tables [2, Section 8.1.3]:

$$log\mu_{ij} = \lambda + \lambda_i^M + \lambda_j^N + \lambda_{ij}^{MN}, \tag{1.1.2}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, n$, where M and N denote the two nominal-scale factors. If we let $\mathbf{h}$ be a vector with all zeros, let $\boldsymbol{\lambda} = (\lambda, \lambda_1^M, \ldots \lambda_m^M, \lambda_1^N, \ldots, \lambda_{mn}^{MN})$, and let $\mathcal{A}$ be the design matrix for this model, then Equation (1.1.1) and Equation (1.1.2) coincide.

Recall that a fundamental statistical result [2] says that given the sum of all cells in the table $n_x = \sum_{j=1}^{t} x_j$, the conditional distribution of $(x_1, \cdots, x_t)$ is the multinomial

distribution $Mult(n_x, \mathbf{p})$ where $\mathbf{p} = (\frac{\mu_1}{n_x}, \ldots, \frac{\mu_t}{n_x})$. Thus the likelihood function is:

$$\begin{aligned}
\mathcal{L}_{\mathcal{A},\mathbf{h}}(\boldsymbol{\lambda} \mid \mathbf{X}) &= \frac{n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (\frac{\mu_j}{n_x})^{x_j} \\
&= \frac{n_x^{-n_x} n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (exp\{h_j + \sum_{i=1}^{n_\lambda} a_{ij}\lambda_i\})^{x_j} \\
&= \frac{n_x^{-n_x} n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (e^{h_j})^{x_j} \cdot exp\{\sum_{j=1}^{t} \sum_{i=1}^{n_\lambda} a_{ij}\lambda_i x_j\} \\
&= \frac{n_x^{-n_x} n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (e^{h_j})^{x_j} \cdot exp\{\sum_{i=1}^{n_\lambda} \lambda_i \sum_{j=1}^{t} a_{ij} x_j\} \\
&= \frac{n_x^{-n_x} n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (e^{h_j})^{x_j} \cdot exp\{\boldsymbol{\lambda}^T (\mathcal{A}\mathbf{X})\}. \qquad (1.1.3)
\end{aligned}$$

Equation (1.1.3) implies that $\mathcal{A}\mathbf{X}$ are sufficient statistics of the log-linear model defined in Equation (1.1.1). In fact, we have the conditional likelihood function:

$$\begin{aligned}
\mathcal{L}_{\mathcal{A},\mathbf{h}}(\boldsymbol{\lambda} \mid \mathbf{X}, \mathcal{A}\mathbf{X} = \mathcal{b}) &= \frac{\frac{n_x^{-n_x} n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (e^{h_j})^{x_j} \cdot exp\{\boldsymbol{\lambda}^T (\mathcal{A}\mathbf{X})\}}{\sum_{\mathbf{Y}=(y_1,\ldots,y_t) \in \mathbb{Z}^t,\ \mathcal{A}\mathbf{Y}=\mathcal{b}} \frac{n_y^{-n_y} n_y!}{y_1! \cdots y_t!} \prod_{j=1}^{t} (e^{h_j})^{y_j} \cdot exp\{\boldsymbol{\lambda}^T (\mathcal{A}\mathbf{Y})\}} \\
&= \frac{\frac{n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (e^{h_j})^{x_j}}{\sum_{\mathbf{Y}=(y_1,\ldots,y_t) \in \mathbb{Z}^t,\ \mathcal{A}\mathbf{Y}=\mathcal{b}} \frac{n_y!}{y_1! \cdots y_t!} \prod_{j=1}^{t} (e^{h_j})^{y_j}} \\
&\propto \frac{n_x!}{x_1! \cdots x_t!} \prod_{j=1}^{t} (e^{h_j})^{x_j}, \qquad (1.1.4)
\end{aligned}$$

which implies that conditional likelihood inference for a log-linear model given $\mathcal{A}\mathbf{X} = \mathcal{b}$ does not rely on the value of $\boldsymbol{\lambda}$, and hence in some articles they are called nuisance parameters [1].

The distribution showed in Equation (1.1.4) is called the **hypergeometric distribution**. Similarly with the Darwin's Finch Data example in the beginning of Section 1.1, the resulting statistical tests can be carried out by computing the expected values of certain test statistics over the set $\Sigma = \{\mathbf{X} \in \mathbb{Z}_+^t : \mathcal{A}\mathbf{X} = \mathcal{b}\}$ with respect to this distribution. More specifically, a **conditional inference p-value** [10] is an

expected value of the form $\mathbb{E}_p[f(\mathbf{X})|\mathcal{A}\mathbf{X} = \mathbf{b}]$, where p is the underlying distribution over Σ, i.e. the hypergeometric distribution, and $f(\mathbf{X})$ is a function of $\mathbf{X}$ defined based on a certain test statistic. This p-value can be estimated by $\frac{1}{\mathfrak{N}}\sum_{i=1}^{\mathfrak{N}} f(\mathbf{X_i})$, where $\mathbf{X_1}, \ldots, \mathbf{X}_{\mathfrak{N}}$ are sampled from the hypergeometric distribution over Σ. For example, given the observed table $\mathbf{x_0}$, the corresponding function $f(\cdot)$ for the conditional inference p-value of the Exact Test can be defined as $f(\mathbf{X}) = \mathbf{1}_{p(\mathbf{X})\leq p(\mathbf{x_0})}(\mathbf{X})$, where $p(\cdot)$ is the hypergeometric distribution over Σ.

In this dissertation, we will focus on sampling with uniform distribution instead of hypergeometric distribution for two reasons: first, assuming identical h_i's, $i = 1, \ldots, t$, the hypergeometric distribution for zero-one tables will degenerate to the uniform distribution; second, sampling with hypergeometric distribution is hard for sparse tables, in contrast, we can use the uniform distribution as the underlying distribution p in $\mathbb{E}_p[f(\mathbf{X})|\mathcal{A}\mathbf{X} = \mathbf{b}]$ for contingency tables without zero-one constraints, and carry out volume tests [18] for a variety of test statistics via sampling over Σ uniformly. In [18], Diaconis and Efron illustrated this topic with an example of the volume test based on the Pearsons χ^2 statistic $\chi^2(\mathbf{X}) = \sum_{j=1}^{t} \frac{e_j - x_j}{e_j}$, where $\mathbf{e} = (e_1, \ldots, e_t)$ is the maximum likelihood estimate of the cell counts under the log-linear model: given the observed table $\mathbf{x_0}$, the p-value of this volume test is $\mathbb{E}_p[\mathbf{1}_{\chi^2(\mathbf{X})\geq\chi^2(\mathbf{x_0})}(\mathbf{X})|\mathcal{A}\mathbf{X} = \mathbf{b}]$, where p is the uniform distribution over Σ, and this p-value can be interpreted as the ratio of number of tables in $\{\mathbf{X} \in \Sigma : \chi^2(\mathbf{X}) \geq \chi^2(\mathbf{x_0})\}$ to the total number of tables in Σ. They claimed that this volume test is adjusted for the disadvantage in Pearsons χ^2 test that: for large t, Pearsons χ^2 test tends to almost always reject the null hypothesis, and in general, little information can be obtained from the value of $\chi^2(\mathbf{X})$ once the null hypothesis of independence is rejected. Volume tests based on other test statistics can be defined similarly with this example.

In practice when the rows of $\mathcal{A}$ are not linearly independent, we can choose a matrix A which collects a subset of rows of $\mathcal{A}$ but still remains the same row space,

where it is obvious that $A\mathbf{X}$ are still sufficient statistics of log-linear model (1.1.1) and the set Σ is the same with the set $\{\mathbf{X} \in \mathbb{Z}_+^t : A\mathbf{X} = b\}$. Note here the rows of A may not necessarily be linearly independent, which implies that $A\mathbf{X}$ may not be minimal sufficient statistics, and we can choose a proper A based on models (see the case of no three-way interaction model in Section 2.1 for example). In the following context of Chapter 1 and Chapter 2, we will focus on sampling over the set of all contingency tables which satisfies the linear constraints $A\mathbf{X} = b$, i.e. the set Σ, uniformly, where in practice a specific b is decided by the observed table.

Let Σ be the set of contingency tables defined above and we assume $\Sigma \neq \emptyset$ in this dissertation. Our goal is sampling a table $\mathbf{X}$ uniformly from Σ. Notice that Σ can be written as

$$\Sigma = \{\mathbf{X} \in \mathbb{Z}^t \mid A\mathbf{X} = b, \mathbf{X} \geq 0\}, \tag{1.1.5}$$

where the design matrix $A \in \mathbb{Z}^{r \times t}$ and vector $b \in \mathbb{Z}^r$ define the r linear constrains. A simple example is sampling a 2×3 contingency table which has row sums $\mathbf{r} = (r_1, r_2)$ and column sums $\mathbf{c} = (c_1, c_2, c_3)$, then:

$$A = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}, \quad \begin{aligned} \mathbf{X} &= (x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23})^T \\ \& \quad b &= (r_1, r_2, c_1, c_2, c_3)^T, \end{aligned}$$

where x_{ij} is the entry in ith row and jth column. These constraints come from the sufficient statistics of the independence model. Under other models, say, diagonal models, quasi-independence models [30], uniform association models [28], we will have some linear constraints in addition to the row sums and column sums.

Let $p(\mathbf{X}) = 1/|\Sigma|$, $\forall \mathbf{X} \in \Sigma$, be the uniform distribution over Σ, where $|\Sigma|$ is the number of elements in Σ. Let $q(\cdot)$ be a trial distribution such that $q(\mathbf{X}) > 0$ for all

$\mathbf{X} \in \Sigma$, and it is designed to be a distribution close to $p(\mathbf{X})$. Then we have

$$\mathbb{E}_q\left[\frac{1}{q(\mathbf{X})}\right] = \sum_{\mathbf{X}\in\Sigma} \frac{1}{q(\mathbf{X})} q(\mathbf{X}) = |\Sigma|.$$

Thus we can estimate $|\Sigma|$, i.e. the total number of tables in Σ, by

$$\widehat{|\Sigma|} = \frac{1}{\mathfrak{N}} \sum_{i=1}^{\mathfrak{N}} \frac{1}{q(\mathbf{X_i})},$$

where $\mathbf{X_1}, \ldots, \mathbf{X}_{\mathfrak{N}}$ are tables drawn iid from $q(\mathbf{X})$. Here, this proposed distribution $q(\mathbf{X})$ is the distribution to sample tables via the SIS procedure.

Now by the multiplication rule we have

$$q(\mathbf{X} = (x_1, \cdots, x_t)) = q(x_1)q(x_2|x_1)q(x_3|x_2, x_1)\cdots q(x_t|x_{t-1}, \ldots, x_1).$$

After computing the lower bound and upper bound for every cell count x_i of $\mathbf{X}$ given previous cells $x_1, \ldots, x_{i-1}$ (see more details in Section 1.3.1), we are able to sample each cell from an interval of integers (a sequence of consecutive integers) and compute $q(x_i|x_{i-1}, \ldots, x_1)$, $i = 2, 3, \ldots, t$.

Note that we may have rejections because tables may be sampled from a bigger set Σ^* such that $\Sigma \subset \Sigma^*$. In this case, as long as conditional probabilities $q(x_i|x_{i-1}, \ldots, x_1)$, $i = 2, 3, \ldots$, and $q(x_1)$ are normalized, $q(\mathbf{X})$ is normalized over Σ^* since

$$\begin{aligned}
\textstyle\sum_{\mathbf{X}\in\Sigma^*} q(\mathbf{X}) &= \textstyle\sum_{x_1,\ldots,x_t} q(x_1)q(x_2|x_1)q(x_3|x_2, x_1)\cdots q(x_t|x_{t-1}, \ldots, x_1) \\
&= \textstyle\sum_{x_1} q(x_1) \left[\sum_{x_2} q(x_1|x_2) \left[\cdots \left[\sum_{x_t} q(x_t|x_{t-1}, \ldots, x_1)\right]\right]\right] \\
&= 1.
\end{aligned}$$

Thus we have

$$\mathbb{E}\left[\frac{\mathbb{I}_{\mathbf{X}\in\Sigma}}{q(\mathbf{X})}\right] = \sum_{\mathbf{X}\in\Sigma^*} \frac{\mathbb{I}_{\mathbf{X}\in\Sigma}}{q(\mathbf{X})} q(\mathbf{X}) = |\Sigma|, \tag{1.1.6}$$

where $\mathbb{I}_{\mathbf{X}\in\Sigma}$ is an indicator function for the set Σ. This implies that the estimator is unbiased.

Therefore, SIS procedure proceeds by simply sampling cell entries of the contingency table sequentially and terminates at the last cell such that the final distribution approximates the target distribution. It also uses the principle of importance sampling in estimating the total number of tables:

$$|\Sigma| = \sum_{\mathbf{X}\in\Sigma} \frac{1}{p(\mathbf{X})} p(\mathbf{X}) = \sum_{\mathbf{X}\in\Sigma^*} \frac{1}{p(\mathbf{X})} \frac{p(\mathbf{X})}{q(\mathbf{X})} q(\mathbf{X}),$$

where $\frac{p(\mathbf{X})}{q(\mathbf{X})}$ is called the importance sampling weight, and this means that sampling $\frac{1}{p(\mathbf{X})}$ from $p(\mathbf{X})$ is equivalent to sampling $\frac{1}{p(\mathbf{X})}\frac{p(\mathbf{X})}{q(\mathbf{X})} = \frac{1}{q(\mathbf{X})}$ from $q(\mathbf{X})$. In addition, because the tables are sampled separately, they are sampled independently and identically distributed (iid) from the proposal distribution.

Comparing with Monte Carlo Markov Chain (MCMC) approach with a **Markov basis (MB)** [19], there are two advantages of SIS procedure. First, SIS procedure does not require expensive or prohibitive pre-computations. In contrast, the computational problem of a MB can be hard. Recall the definition for a MB:

Definition 1.1.1. *[10] Define the* ***kernel (null space)*** *for matrix A as $ker_{\mathbb{Z}}(A) = \{\mathbf{X} \in \mathbb{Z}^t | A\mathbf{X} = 0\}$ and $\mathbf{m}$ is called a* ***Markov move*** *if $\mathbf{m} \in ker_{\mathbb{Z}}(A)$. A Markov basis M_A for A is a subset of the $ker_{\mathbb{Z}}(A)$ such that for each pair of vectors $\mathbf{u}, \mathbf{v} \in \mathbb{Z}^t_+$ with $A\mathbf{u} = A\mathbf{v}$, there is a sequence of Markov moves $\mathbf{m}_i \in M_A$, $i = 1, \ldots, k$, such that*

$$\mathbf{u} = \mathbf{v} + \sum_{i=1}^{k} \mathbf{m}_i, \ 0 \leq \mathbf{v} + \sum_{i=1}^{j} \mathbf{m}_i, \ j = 1, \ldots, k.$$

A method to compute Markov moves that connect all tables with given constrains was given in [19], but it cannot compute the moves in some large logistic regression examples. In fact, it was proved that the number of MB elements can be arbitrary large for three-way contingency tables with fixed two-way marginals [15]. Second, the SIS procedure is guaranteed to sample a table from the proposal distribution if there is no rejection, while in an MCMC approach the chain may take a long time to

converge to a stationary distribution in order to satisfy the independent condition, and what makes it worse is that the time complexity may be unknown.

1.1.2 SIS procedure with conditional Poisson (CP) distribution

In this section, we first explain why we need to develop a special SIS method only for zero-one contingency tables. Secondly, we review what is conditional Poisson (CP) distribution, and how to use it to generate a vector. Lastly, we introduce how to sample zero-one tables with linear constrains using SIS procedures with CP distribution.

The SIS procedure in Section 1.1.1 can also be used to sample zero-one tables. In order to do this, we need to define "slack" variables $\mathbf{Y}$ as $\mathbf{Y} = \mathbf{1_t} - \mathbf{X}$ so that we can write the set Σ in the form of $\Sigma = \{\mathbf{X} \in \mathbb{Z}^t \mid A'\mathbf{X}' = b', \mathbf{X}' \geq 0\}$, where $\mathbf{X}' = (\mathbf{X}^T, \mathbf{Y}^T)^T$, and A' and b' define the constrains which include both marginal sums and zero-one conditions (see more details in Section 1.3.1). Let's continue to use the simple example in the Section 1.1.1. To use the SIS procedure in Section 1.1.1 to sample a zero-one 2×3 table which has row sums $\mathbf{r} = (r_1, r_2)$ and column sums

$\mathbf{c} = (c_1, c_2, c_3)$, we define:

$$A' = \begin{pmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}, \quad \begin{aligned}
\mathbf{X} &= (x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}) \\
\mathbf{Y} &= (y_{11}, y_{12}, y_{13}, y_{21}, y_{22}, y_{23}) \\
\mathbf{X}' &= (\mathbf{X}^T, \mathbf{Y}^T)^T \\
b' &= (r_1, r_2, c_1, c_2, c_3, 1, 1, 1, 1, 1, 1)^T,
\end{aligned}$$

where x_{ij} is the entry of the zero-one 2×3 table in ith row and jth column. Thus the number of variables is doubled by adding the slack variables, and this can make the problem exponentially harder when the table is large.

In [9], Chen et al introduced a sequential importance sampling (SIS) procedure to sample zero-one two-way tables with given fixed marginal sums, i.e. row and column sums, via the conditional Poisson (CP) distribution. Compared with the SIS procedures in Section 1.1.1, it proceeds by sampling columns, but not cell entries, of the zero-one contingency table sequentially and terminates at the last column.

Before we go any further, the definition of the conditional Poisson distribution must be clarified.

Definition 1.1.2. *[9] Let*

$$Z = (Z_1, \ldots, Z_l)$$

be independent Bernoulli trials with probability of successes $p = (p_1, \ldots, p_l)$, where l

is the length of Z. Then the random variable

$$S_Z = Z_1 + \cdots + Z_l$$

has a Poisson–binomial distribution. The ***conditional Poisson (CP) distribution*** *is defined as the conditional distribution of Z given S_Z, i.e. $Z \mid S_Z$. Now let $w_k = p_k/(1-p_k)$, where $p_k \in (0,\, 1)$, be the "weight" of the kth cell. Then*

$$P(Z_1 = z_1, \ldots, Z_l = z_l | S_Z = l_0) \propto \prod_{k=1}^{l} w_k^{z_k}, \tag{1.1.7}$$

i.e. the conditional probability is proportional to the product of weights of those cells who have "1" as their entries.

Sampling a zero-one vector of length l means choosing l_0 among the l cells to have entry ones. There are $\binom{l}{l_0}$ many choices where the probability of picking each choice is calculated via the CP distribution. The details of this algorithm was introduced in [7]. Denote $[l] = \{1, 2, \ldots, l\}$ as the set of all cells in the vector, and l_0 of them need to be drawn one by one to have entry ones. Let $A_k \subset [l]$, be the set of selected cells after k cells are selected, $k = 0, \ldots, l_0$. Thus $A_0 = \emptyset$, and A_{l_0} is the set we want to obtain. By induction, all we need to show is how to get A_k from A_{k-1}. Define the complement sets $A_k^c = [l] \backslash A_k$, $k = 0, \ldots, l_0$. Assuming we have selected $k-1$ cells and stored them in A_{k-1}, then according to [7], the probability of choosing $j \in A_{k-1}^c$ to be the new selected cell is:

$$P(j, A_{k-1}^c) = \frac{w_j R(l_0 - k, A_{k-1}^c - j)}{(l_0 - k + 1) R(l_0 - k + 1, A_{k-1}^c)},$$

where

$$R(s, A) = \sum_{B \subset A, |B| = s} \left(\prod_{i \in B} w_i \right)$$

and the function $R(s, A)$ can be calculated using the recursive formula

$$R(s, A) = R(s, A \backslash \{s\}) + w_s R(s-1, A \backslash \{s\}).$$

Notice that the value $R(l_0, [l]) = \sum_{B\subset[l],|B|=l_0} \left(\prod_{i\in B} w_i\right)$ is exactly the normalizing constant for Equation (1.1.7).

For example, suppose we want to sample $Z = (Z_1, Z_2, Z_3, Z_4)$ given $S_Z = 2$ where the weights are w_1, w_2, w_3, w_4, respectively. We start with $A_0 = \emptyset$ and draw the first cell from a multinomial distribution with probabilities $P(j, [4])$, where $j = 1, .., 4$ and $[4] = \{1, 2, 3, 4\}$. Suppose the first cell is 2, then $A_1 = \{2\}$. Then we draw the second cell from a multinomial distribution with probabilities $P(j, \{1, 3, 4\})$, $j = 1, 3, 4$. Suppose the second cell is 3, then $A_2 = \{2, 3\}$, i.e. we obtain a sample $(0, 1, 1, 0)$ from the CP distribution. A useful trick is that when $l_0 > l/2$, sampling Z is equivalent to sampling $Z' = \mathbf{1}_l^T - Z$ given $S'_Z = l - l_0$ where weights $w'_k = 1/w_k$, $k = 1, \ldots, l$.

To apply the CP distribution to sampling zero-one two-way tables with fixed row sums and column sums, we can simply consider each column to be a random vector which follows a CP distribution where vector sums are the column sums and the weights can be determined by row sums [9, 8].

Theorem 1.1.3. *[9, Theorem 1] For the uniform distribution over all $m \times n$ zero-one tables with given row sums $r_1, \ldots, r_m$ and first column sum c_1, the marginal distribution of the first column is the same as the conditional distribution of Z given $S_Z = c_1$ with $p_i = r_i/n$.*

The idea of the proof has two steps. First, imaging that we randomly select r_i cells in the ith row to put entry ones, $i = 1, \ldots, m$. Because every choice for a single row is equally possible and rows are determined independently, the table we generate is sampled uniformly from the set of all zero-one $m \times n$ tables which have row sums $r_1, \ldots, r_m$, and the chance that the $(i, 1)$th cell has entry one in the specific table is $\binom{n-1}{r_i-1}/\binom{n}{r_i} = \frac{r_i}{n}$., i.e. p_i. Hence the first column can be considered as a vector of independent Bernoulli random variables with success probabilities $(p_1, \ldots, p_m)$. Second, we reject the table in the first step if its first column sum is not c_1, then the distribution of the first column becomes a CP distribution specified in the theorem.

Therefore, CP distribution is the desired marginal distribution of the first column in the zero-one two-way tables given the marginal sums.

Based on Theorem 1.1.3, an SIS procedure with CP distribution goes as following: first, sample the first column with CP distribution where weights are determined according to Theorem 1.1.3; second, remove the first column so that we have a subtable which contains the rest $n-1$ columns; third, consider the subtable to be a new table with updated row sums and column sums, and repeat the first two steps until only one column left, in which case the value of this column will be fixed. For table $\mathbf{X}$, denote the columns of the table as $\mathbf{x_1}, \ldots, \mathbf{x_n}$. Again by multiplication rule:

$$q(\mathbf{X} = (\mathbf{x_1}, \cdots, \mathbf{x_n})) = q(\mathbf{x_1})q(\mathbf{x_2}|\mathbf{x_1})q(\mathbf{x_3}|\mathbf{x_2}, \mathbf{x_1}) \cdots q(\mathbf{x_n}|\mathbf{x_{n-1}}, \ldots, \mathbf{x_1}).$$

Note that (1) every time before generating a column, we should check if there is any trivial cases, i.e. $\exists\ \frac{r_i}{n}$ (or $\frac{c_i}{m}$) $= 0$ (or 1). If there is, then we should fill the whole row (or column) with 0 (or 1), remove it and update the marginal sums; (2) every time after generating a column $\mathbf{x_j}$, we use Equation (1.1.7) to compute the probability that $\mathbf{x_j}$ takes the specific vector, i.e. the probability $q(\mathbf{x_j}|\mathbf{x_{j-1}}, \ldots, \mathbf{x_1})$, so that the probability of the whole table $\mathbf{X}$ can be obtained by the product of this series of probabilities of columns. Figure 1.1 gives an example of sampling a 3×4 zero-one table using SIS procedure via CP distribution given row sums $(2, 1, 3)$ and column sums $(2, 1, 1, 2)$.

The issue of rejection raises because the feasibility of the subtable is not considered when the previous column is sampled, so when there is no feasible solution for the subtable we will reject the sample $\mathbf{X}$ in process and record $\mathbb{I}_{\mathbf{X}\in\Sigma} = 0$ (see Equation (1.1.6)). An example of this type of rejection in sampling a 4×4 zero-one table with

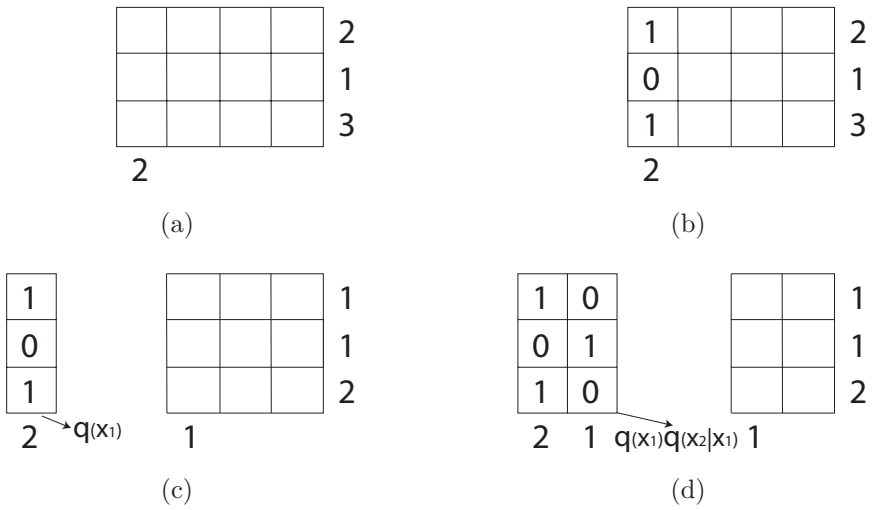
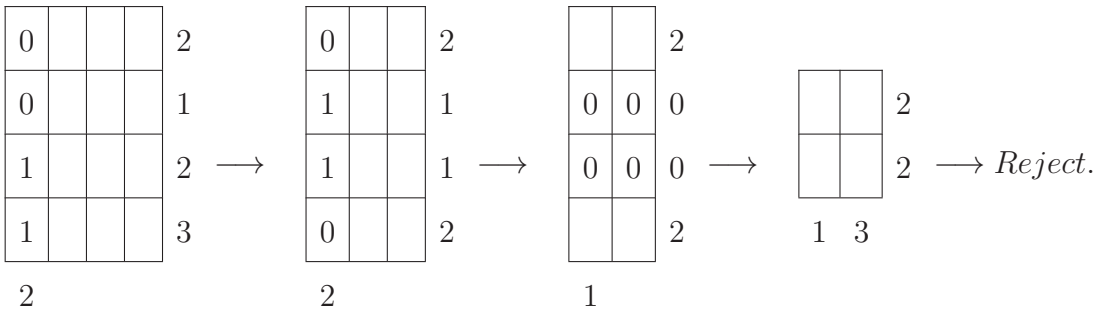


Figure 1.1: An example of sampling a 3×4 zero-one table using SIS via CP distribution

(a), start with the first column; (b), sample the first column with CP distribution, get $(1, 0, 1)$; (c), compute the probability of $\mathbf{x_1} = (1, 0, 1)$, remove $\mathbf{x_1}$, update the row sums for the 3×3 subtable and look at the first column of the subtable; (d), repeat (b) and (c) until a whole table is sampled.

row sums $(2, 1, 2, 3)$ and column sums $2, 2, 1, 3$ is given as below:



To deal with this issue, in [9], Chen et al figured out an improved SIS procedure with CP distribution for zero-one two-way tables, which is based on the sufficient and necessary condition provided by the Gale-Ryser Theorem [25, 45] and never has rejection. The Gale-Ryser Theorem will be stated after some definitions.

Definition 1.1.4. *[9, Definition 1] For any* $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$*, let* $x_{[1]} \geq \cdots \geq x_{[n]}$ *denote the components of* $\mathbf{x}$ *in decreasing order. For* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$*, we define* $\mathbf{x} \prec \mathbf{y}$ *if*

$$\sum_{i=1}^{k} x_{[i]} \leq \sum_{i=1}^{k} y_{[i]}, \ k = 1, \ldots, n-1, \ \textit{and} \ \sum_{i=1}^{n} x_{[i]} = \sum_{i=1}^{n} y_{[i]}.$$

When $\mathbf{x} \prec \mathbf{y}$, $\mathbf{x}$ *is said to be* ***majorized*** *by* $\mathbf{y}$ *(*$\mathbf{y}$ *majorizes* $\mathbf{x}$*).*

Definition 1.1.5. *[9, Definition 2] Let* $x_1, x_2 \ldots, x_n$ *be nonnegative integers, and define*

$$x_j^* = \#\{x_i : x_i \geq j\},\ j = 1, 2, \ldots$$

The sequence $x_1^*, x_2^*, x_3^*, \ldots$ *is said to be* ***conjugate*** *to* $x_1, x_2 \ldots, x_n$*. Note that the conjugate sequence* $\{x_i^*\}$ *is always non-increasing and is independent of the order of the* x_i*'s.*

Theorem 1.1.6 (Gale-Rayser Theorm)**.** *[25, 45] Let* $r_1, \ldots, r_m$ *be nonnegative integers not exceeding* n*, and* $c_1, \ldots, c_n$ *be nonnegative integers not exceeding* m*. A necessary and sufficient condition for the existence of an* $m \times n$ *zero-one table with row sums* $r_1, \ldots, r_m$ *and column sums* $c_1, \ldots, c_n$ *is that*

$$(c_1, \ldots, c_n) \prec (r_1^*, \ldots, r_m^*),\ \textit{or, equivalently,}\ (r_1, \ldots, r_m) \prec (c_1^*, \ldots, c_n^*).$$

In [8], Chen extended their SIS procedure to sampling zero-one two-way tables with given fixed marginal sums with structures, i.e., some cells are fixed to be zero or one. Since the structural ones can be converted to structural zeros simply by converting the marginal sums, we only discuss structural zeros for brevity. The cells which are structural zeros are usually denoted by "[0]", and we define:

$$\Omega = \{(i, j) : (i, j) \text{ is a structural zero}\}.$$

These structures are not limited in zero-one tables, but can appear in any contingency tables. And example is given in Table 1.2 to illustrate in what kind of cases we need to set up structures. The extended theorem is as following:

Theorem 1.1.7. *[8, Theorem 1] For the uniform distribution over all* $m \times n$ *zero-one tables with given row sums* $r_1, \ldots, r_m$*, first column sum* c_1*, and the set of structural zeros* Ω*, the marginal distribution of the first column is the same as the conditional*

Table 1.2: Different types of cancer separated by gender for Alaska in year 1989

Type of cancer	Female	Male	Total
Lung	38	90	128
Melanoma	15	15	30
Ovarian	18	[0]	18
Prostate	[0]	111	111
Stomach	0	5	5
Total	71	221	292

The structural zeros's are denoted by "[0]". For example, females cannot have prostate cancer so the corresponding cell is fixed to be 0, i.e. a structural zero.

distribution of Z *given* $S_Z = c_1$ *with* $p_i = I_{[(i,1)\notin\Omega]} r_i/(n - g_i)$ *where* g_i *is the number of structural zeros in the ith row.*

The strategy is straightforward. Take Figure 1.2 for example. When we sample

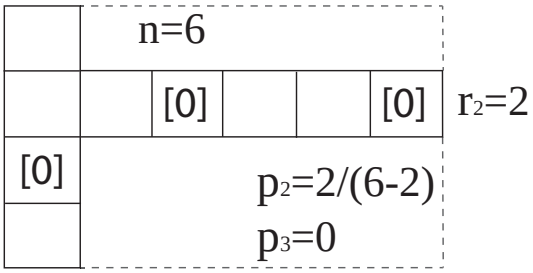


Figure 1.2: An example of sampling a 4×6 zero-one table with structural zeros

the first column of this 4×6 zero-one table, since there are two cells fixed to be 0 in the second row, we have to assign two ones into the four free cells equally randomly, and this means that the chance that the $(2,1)$th cell get entry one is $p_2 = 2/(6-2) = 2/4$. In the mean time, the chance that the $(3,1)$th cell get entry one $p_3 = 0$ because it is a structural zero.

In [8], Chen also tried to extend the Gale-Ryser Theorem to find a necessary and sufficient condition for the existence of zero-one two-way tables with given marginal sums and a fixed set of structural zeros so that they could design a corresponding

algorithm which never had rejections. But the corresponding theorem, [8, Theorem 2], was restricted to the special case that there is at most one structural zero in each row and each column, which usually is not true in practice.

1.1.3 SIS procedure in a algebraic geometric view

In this section, we will review the SIS procedure in Section 1.1.1 in an algebraic geometric view and point out the main implementation issue in this procedure – approximating the support of the marginal distribution of each cell. The notation defined in Section 1.1.1 will be adopted here.

In SIS procedure, we compute the lower bound l_i and upper bound u_i for x_i (see details in Section 1.3.1), and sample x_i from the interval of integers $[l_i, u_i]$, i.e. the sequence of integers $l_i, l_i+1, \ldots, u_i-1, u_i$, $i = 1, \ldots, t-1$. In this process, rejections can happen because for some cells, the supports of their marginal distributions are not intervals of integers (see Section 1.3.1 for the details about the existence of holes in the semigroups). In [10], Chen et al defined a property of the design matrix A with which these rejections can be avoided:

Definition 1.1.8. *[10, Definition 3.2] Define the projection operator $\pi_1 : \mathbb{Z}^k \to \mathbb{Z}$ by $\pi_1(z_1, \ldots, z_k) = z_1$. For $b \in \mathbb{Z}_+^r$ define $A^{-1}[b] := \{\mathbf{X} \in \mathbb{Z}_+^t : A\mathbf{X} = b\}$. Let $\mathbf{a}_1, \ldots, \mathbf{a}_t$ be the columns of A, and $A_i = (\mathbf{a}_i, \ldots, \mathbf{a}_t)$, $i = 1, \ldots, t$, be the submatrices of A that the first $i-1$ columns are removed. Then $A^{-1}[b]$ is said to have the* ***sequential interval property*** *if:*

- *$\pi_1(A^{-1}[b])$ is an interval of integers $[l_1, u_1]$, and*
- *for $i = 1, \ldots, t-1$: if $x_i \in \pi_1(A_i^{-1}[b - \mathbf{a}_1 x_1 - \ldots - \mathbf{a}_{i-1} x_{i-1}])$, then $\pi_1(A_{i+1}^{-1}[b - \mathbf{a}_1 x_1 - \ldots - \mathbf{a}_{i-1} x_{i-1} - \mathbf{a}_i x_i])$ is also an interval of integers $[l_{i+1}, u_{i+1}]$.*

Notice that with some orders of the cells A may have the sequential interval property and others may not, and it is clear that we can avoid the rejection because

of holes (see Section 1.3.1) as long as we can find one cell ordering $(x_1, \ldots, x_t)$ with which $A^{-1}[b]$ has the property. Hence finding good ordering of cells is important in SIS procedure. In the following content of this section, we are going to introduce the conditions given in [10] which can guarantee that $A^{-1}[b]$ has the sequential interval property.

We need to recall some definitions [10]. For a Markov move $\mathbf{m} \in ker_{\mathbb{Z}}(A)$, we define $\mathbf{m}^+ = max\{\mathbf{0}, \mathbf{m}\}$ and $\mathbf{m}^- = max\{\mathbf{0}, -\mathbf{m}\}$, which implies $\mathbf{m} = \mathbf{m}^+ - \mathbf{m}^-$. Define the polynomial ring $Q[y_1, \ldots, y_t]$ in indeterminates, i.e. polynomial variables, $y_1, \ldots, y_t$, one for each cell. Define the toric ideal

$$I_A := \langle \mathbf{y^u} - \mathbf{y^v} : A\mathbf{u} = A\mathbf{v} \rangle,$$

where $\mathbf{y^u} := y_1^{u_1} y_2^{u_2} \cdots y_t^{u_t}$ is the usual monomial notation for a nonnegative integer vector of exponents $\mathbf{u} = (u_1, \ldots, u_t)$. The way to connect a Markov move $\mathbf{m}$ to a polynomial is $\mathbf{y^{m^+}} - \mathbf{y^{m^-}}$, for example, the Markov move $(1, -1, -1, 1)'$ can be denoted as $y_1 y_4 - y_2 y_3$. An algebraic result given by [19, Theorem 3.1] says that a Markov basis always exists independently of the actual values of b, where $A\mathbf{X} = b$ defines the linear constraints. The following propositions give the conditions for the sequential interval property where lexicographic term order (lex order) is primarily used to order monomials.

Proposition 1.1.9. *[10, Proposition 3.1] Suppose a Markov basis $M_A = \{\pm\mathbf{m_1}, \ldots, \pm\mathbf{m_g}\}$ has the property that*

- *$G := \{\mathbf{y^{m_i^+}} - \mathbf{y^{m_i^-}}, i = 1, \ldots, g\}$ is a lex Gröbner basis with ordering $y_1 > y_2 > \cdots > y_t$ on indeterminates and*
- *suppose the elements of $G \cap Q[y_i, \ldots, y_t]$ are square-free in x_i for each i.*

Then $A^{-1}[b]$ has the sequential interval property for all b.

The converse proposition is also true.

Proposition 1.1.10. *[10, Proposition 3.2] Let A be a nonnegative integer matrix such that $A^{-1}[b]$ has the sequential interval property for all b. Then the reduced lex Gröbner basis G for I_A with ordering $y_1 > y_2 > \cdots > y_t$ on indeterminates has $G \cap Q[y_i, \ldots, y_t]$ square-free in x_i for all i.*

In some cases, the full Markov basis does not satisfy the required conditions that guarantee the sequential interval property ([10, Example 7.3] gives a 6-way table that is in this situation). Thus they also studied using the particular values of the margin constraints b so that a smaller and simpler connecting set, a Markov subbasis $M_{A,b}$ [10], may be allowed for this specific b. They worked out certain conditions for the Markov subbasis $M_{A,b}$ such that $A^{-1}[b]$ has the sequential interval property for the specific b. More details of this topic can be found in [10].

1.2 Model selection in Bayesian networks (BNs)

Bayesian networks (BNs), also known as belief networks, Bayes networks, Bayes(ian) models or probabilistic directed acyclic graphical models, find their applications in many areas, such as computational biology, bioinformatics (for example, gene regulatory networks, protein structure, gene expression analysis [23] learning epistasis from GWAS data sets [32]) and medicine [57]. BNs are a part of the family of probabilistic graphical models (GMs). These graphical structures represent information about probabilistic structures for a statistical model.

In order to define BNs precisely and explicitly, we will recall the basic notation and definitions in this section. Firstly, we give the definitions of conditional independence (CI) statements and CI models. Secondly, several types of graphs, including directed acyclic graphs (DAGs), and related concepts will be defined. Then we introduce the CI models induced by undirected graphs (UG), i.e. Markov networks, and by DAGs, i.e. BNs. Lastly, we parameterize the discrete BNs and talk about several properties of quality criterions that are used as score functions in model selection in BNs.

1.2.1 Conditional independence (CI) models

The notation and definitions about CI statements and CI models in this section can be found in [51, § 2.2.1 – § 2.2.3].

Let N be a set of random variables. A **disjoint triplet** over N is a triplet $\langle A,\ B \mid C\rangle$ of pairwise disjoint subsets of N. The class of all disjoint triplets over N is denoted by $\mathcal{T}(N)$.

Definition 1.2.1. *[51, § 2.2.1] A **conditional independence (CI) statement** over N is a statement of the form "A is conditionally independent of B given C" where $A,\ B,\ C \subseteq N$ are pairwise disjoint subsets of N. We can denote such a statement by $\langle A,\ B \mid C\rangle \in \mathcal{T}(N)$. Notice that a CI statement should always be understood with respect to a certain mathematical object $\mathbf{o}$ over N (for example, a probability measure over N, or a graph over N), in which sense we denote it by $A \perp\!\!\!\perp B \mid C\ [\mathbf{o}]$ where $[\mathbf{o}]$ is sometimes omitted if the omission does not result in confusion or hesitancy in reading.*

Definition 1.2.2. *[51, § 2.2.1] For any class $\mathcal{M} \subseteq \mathcal{T}(N)$ of disjoint triplets over N, if we define $\langle A,\ B \mid C\rangle \in \mathcal{M}$ as a CI statement with respect to $\mathcal{M}$, i.e. $A \perp\!\!\!\perp B \mid C\ [\mathcal{M}]$, then $\mathcal{M}$, which can be considered as a formalization of probabilistic relationships between variables in N, can be interpreted as a **conditional independence (CI) model**. We also use the same noun for the set of probability measures over N: $\mathbb{M} = \{P : A \perp\!\!\!\perp B \mid C\ [P]$, for $\forall\ \langle A,\ B \mid C\rangle \in \mathcal{M}\}$, which is also called **the statistical model of CI structure**.*

The conventional definition of conditional independence can be considered as a special case of Definition 1.2.1: for a probability measure P over N and pairwise disjoint subsets $A,\ B,\ C \subseteq N$, A is conditionally independent of B given C with respect to P, i.e. $A \perp\!\!\!\perp B \mid C\ [P]$, if and only if

$$P(A|BC) = P(A|C) \text{ for } A, B, C \text{ with } P(BC) > 0.$$

In addition, the **CI model induced by** P is $\mathcal{M}_P = \{\langle A,\ B \mid C\rangle \in \mathcal{M} : A \perp\!\!\!\perp B \mid C\ [P]\}$.

Definition 1.2.3. *[51, § 2.2.2] A subset $\mathcal{M} \subseteq \mathcal{T}(N)$ is called a* ***disjoint semi-graphoid*** *if for pairwise disjoint sets A, B, C, $D \subseteq N$ the following holds:*

1. ***triviality*** $A \perp\!\!\!\perp \emptyset \mid C\ [\mathcal{M}]$;
2. ***symmetry*** $A \perp\!\!\!\perp B \mid C\ [\mathcal{M}] \Longrightarrow B \perp\!\!\!\perp A \mid C\ [\mathcal{M}]$;
3. ***decomposition*** $A \perp\!\!\!\perp (B \cup D) \mid C\ [\mathcal{M}] \Longrightarrow A \perp\!\!\!\perp D \mid C\ [\mathcal{M}]$;
4. ***weak union*** $A \perp\!\!\!\perp (B \cup D) \mid C\ [\mathcal{M}] \Longrightarrow A \perp\!\!\!\perp B \mid (D \cup C)\ [\mathcal{M}]$;
5. ***contraction*** $A \perp\!\!\!\perp B \mid (D \cup C)\ [\mathcal{M}] \wedge A \perp\!\!\!\perp D \mid C\ [\mathcal{M}] \Longrightarrow A \perp\!\!\!\perp (B \cup D) \mid C\ [\mathcal{M}]$.

Notice here $A \perp\!\!\!\perp B \mid C\ [\mathcal{M}]$ means that $\langle A,\ B \mid C\rangle \in \mathcal{M}$.

The semi-graphoid properties above define the implication between valid CI statements that leads to the question whether certain CI statements are already implied by other CI statements. This question is known as the **CI implication problem** or the **CI inference problem**. By using these properties we are able to define a set of certain special CI statements, which are called elementary, such that they are sufficient and necessary for the existence of other statements.

Definition 1.2.4. *[51, § 2.2.3] An* ***elementary CI statement*** $A \perp\!\!\!\perp B \mid C\ [\mathbf{o}]$ *is an (elementary) triplet $\langle A,\ B \mid C\rangle$, where $A = \{a\}$ and $B = \{b\}$ are single elements in N.*

Lemma 1.2.5. *[51, Lemma 2.2] Suppose $\mathcal{M}$ is a disjoint semi-graphoid over N. $\forall\ \langle A,\ B \mid C\rangle \in \mathcal{T}(N)$, the CI statement $A \perp\!\!\!\perp B \mid C\ [\mathcal{M}]$ is valid if and only if:*

$$\forall a \in A,\ \forall b \in B,\ \forall D : C \subseteq D \subseteq (A \cup B \cup C)\backslash\{a, b\},\ \text{we have } a \perp\!\!\!\perp b \mid D\ [\mathcal{M}].$$

Lemma 1.2.6. *[51, Lemma 2.1] Every CI model $\mathcal{M}_P$ induced by a probability measure P over N is a disjoint semi-graphoid over N.*

1.2.2 Graphs

Intuitively speaking, CI models defined by graphs are called graphical models, and if the graph is a directed acyclic graph (DAG), then it is called a DAG model or Bayesian network (BN). In this section, we will show some well-defined classic graphs and some related concepts. One can find more details in [51, § A.3].

Definition 1.2.7. *[51, § A.3] A **graph** is specified by a non-empty finite set of nodes N and a set of edges consisting of pairs of distinct elements taken from N. Classic graphs admit only two basic types of edges. An **undirected edge** (or a **line**) over N is an unordered pair $\{a, b\}$ where $a, b \in N$, $a \neq b$. A **directed edge** (or an **arrow**) over N is an ordered pair (a, b) where $a, b \in N$, $a \neq b$.*

Definition 1.2.8. *[51, § A.3] A **graph with mixed edges** over N is given by a set of undirected edges $\mathcal{E}_{ud}$ and a set of directed edges $\mathcal{E}_d$ over N. Suppose $G = (N, \mathcal{E}_{ud}, \mathcal{E}_d)$ is a graph of this kind, then a pictorial representation of G can be naturally given by drawing "$a - b$", $\forall\ \{a, b\} \in \mathcal{E}_{ud}$, and drawing "$a \to b$", $\forall\ (a, b) \in \mathcal{E}_d$. $\forall$ disjoint $a, b \in N$, if either $a - b$ in G, $a \to b$ in G or $b \to a$ in G, then we briefly say $[a, b]$ is an **edge** in G. Now we can define the following graphs:*

- *a **hybrid graph** over N is a graph G which has no multiple edges, i.e. for an ordered pair of distinct nodes (a, b), $a, b \in N$, at most one of these three cases can occur: $a - b$, $a \to b$ or $b \to a$;*
- *an **undirected graph (UG)** is a graph containing only undirected edges, i.e. $\mathcal{E}_d = \emptyset$;*
- *a **directed graph** is a graph containing only directed edges, i.e. $\mathcal{E}_{ud} = \emptyset$;*

- *The **underlying graph (skeleton)** H of a graph $G = (N, \mathcal{E}_{ud}, \mathcal{E}_d)$ is an undirected graph over N such that $a - b$ in H if and only if $[a, b]$ is an edge in G;*

- *A **chain** for a hybrid graph G over N is a partition of N into an ordered sequence of non-empty disjoint subsets $B_1, \ldots, B_n$, $n \geq 1$ called **blocks** such that,*

  - *if $[a, b]$ is an edge in G with $a, b \in B_i$ then $a - b$, and*
  - *if $[a, b]$ is an edge in G with $a \in B_i$, $b \in B_j$, $i < j$ then $a \to b$.*

  *A **chain graph** is a hybrid graph which admits a chain;*

- *if $\emptyset \neq T \subseteq N$, then the **induced subgraph** of G for T is the graph $G_T = (T, \mathcal{E}_{ud}^T, \mathcal{E}_d^T)$ where $\mathcal{E}_{ud}^T$ ($\mathcal{E}_d^T$) is the set of those undirected (directed) edges over T which are also in $\mathcal{E}_{ud}$ ($\mathcal{E}_d^T$);*

- *a **complex** is an induced subgraph of a hybrid graph G for $T = \{a_1, \ldots, a_k\}$, $k \geq 3$ such that $d_1 \to d_2$, $d_i - d_{i+1}$ for $i = 2, \ldots, k-2$, $d_{k-1} \leftarrow d_k$ in G and no additional edge between any two distinct nodes of $\{d_1, \ldots, d_k\}$ exists in G.*

- *an **immorality** is an induced subgraph of a hybrid graph G for $T = \{a, b, c\}$ such that $a \to c$ in G and $b \to c$ in G while $[a, b]$ is not an edge in G.*

Remark 1.2.9. *Definition 1.2.8 implies:*

1. *undirected graphs are a subset of chain graphs: whenever there is only one block and all nodes belong to this block;*

2. *immoralities are complexes with $k = 3$, and the only type of complexes that can appear in directed graphs are immoralities.*

Definition 1.2.10. *[51, § A.3] A* ***route*** *from a node a to a node b (or between nodes a and b) in a graph G with mixed edges is a sequence of nodes $c_1, \ldots, c_n \in N$, $n \geq 1$ together with a sequence of edges $\epsilon_1, \ldots, \epsilon_{n-1} \in \mathcal{E}_{ud} \cup \mathcal{E}_d$ such that $a = c_1$, $b = c_n$ and ϵ_i is either $c_i - c_{i+1}$, $c_i \to c_{i+1}$ or $c_i \leftarrow c_{i+1}$ for $i = 1, \ldots, n-1$. A route is called to be* ***descending*** *if ϵ_i is either $c_i - c_{i+1}$, $c_i \to c_{i+1}$ for $i = 1, \ldots, n-1$. A* ***path*** *is a route in which $c_1, \ldots, c_n$ are distinct. A* ***cycle*** *is a route where $n \geq 3$, $c_1 = c_n$ and $c_1, \ldots, c_{n-1}$ are distinct such that, in the case $n = 3$, ϵ_2 is not a reverse copy of ϵ_1 (this implies that $a - b - a$, $a \to b \leftarrow a$ and $a \leftarrow b \to a$ are not cycles while $a - b \to a$ and $a \to b \to a$ are supposed to be cycles). A* ***directed cycle*** *is a cycle which is a descending route and at least one edge ϵ_i is directed.*

An ***acyclic directed graph****, which is also called* ***acyclic digraph*** *or* ***directed acyclic graph (DAG)****, over N is a directed graph over N without directed cycles.*

Remark 1.2.11. *A DAG can be equivalently introduced as a directed graph G whose nodes can be ordered in a sequence $a_1, \ldots, a_k$, $k \geq 1$ such that if $[a_i, a_j]$, $i < j$, is an edge in G then $a_i \to a_j$ in G. This also means that DAGs are chain graphs: every block has only one node and arrows are only allowed from block B_i to block B_j where $i < j$.*

Definition 1.2.12. *[51, § A.3] A node a is a* ***parent*** *of a node b in G, and dually b is a* ***child*** *of a, if $a \to b$ in G; a is an* ***ancestor*** *of b in G, and dually b is a* ***descendant*** *of a, if there exists a descending route (or equivalently a descending path) from a to b in G. For $b \in N$, the set of parents of b in G is denoted by $pa_G(b)$. For $A \subseteq N$, we define $an_G(A) = \{b \in N : \exists\ a \in A$, such that b is an ancestor of $a\}$.*

1.2.3 CI models induced by undirected graphs and acyclic directed graphs

For a graph G, suppose each node represents a random variable and each edge represents the probabilistic dependency among the random variables corresponding to the nodes adjacent to the edge [37], then G can be considered as a description of

CI structures, and a CI model induced by G can be defined as $\mathcal{M}_G = \{\langle A,\ B \mid C\rangle \in \mathcal{T}(N) : A \perp\!\!\!\perp B \mid C\ [G]\}$. We name the CI models which are induced by a graphs as **graphical models**. More names have been assigned to special graphical models: graphical models based on undirected graphs (UGs) are also known as **Markov networks**, and those based on acyclic directed graphs (DAGs) are called **DAG models** or **Bayesian Networks (BNs)**. The definitions of these models rely on the graphical criterions which answer the question that whether a certain CI statement is contained in the complete list of valid CI statements of a graph G, where different criterions should be defined for different types of graphs. In this section we will first introduce the graphical criterions for UGs and DAGs [51, § 3], and then an equivalent relation on chain graphs: Markov equivalence.

Definition 1.2.13. *[51, § 3.1] Let $G = (N, \mathcal{E}_{ud})$ be an undirected graph and $\langle A,\ B \mid C\rangle \in \mathcal{T}(N)$. We say that C is a **separator** of A and B (or C separates A and B) in G if every route (equivalently every path) in G between $a \in A$ and $b \in B$ contains a node $c \in C$. The **separation criterion** says that $\langle A,\ B \mid C\rangle$ is **represented in** G, i.e. $A \perp\!\!\!\perp B \mid C\ [G]$, if and only if C is a separator of A and B. (See Figure 1.3 for an example.)*

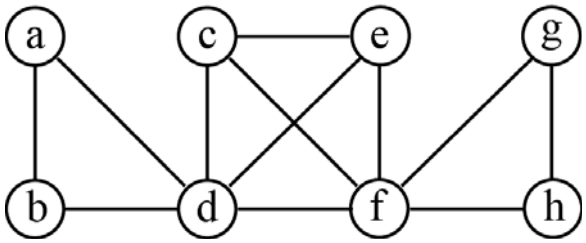


Figure 1.3: An example of separation criterion for undirected graph

Let $A = \{a, b\}$, $B = \{g, h\}$ and $C = \{d, f\}$. Since every path from A to B contains at least one node in C, we can say that C is a separator of A and B. By definition, $\langle A,\ B \mid C\rangle$ is represented in this UG according to separation criterion.

Definition 1.2.14. *[51, § 3.2] Let $G = (N, \mathcal{E}_d)$ be a DAG. The* ***moral graph*** *of G is an undirected graph which is obtained by two steps: first, add edges $a - b$ whenever a and b have a common child c; second, the moral graph is the skeleton of the resulting graph in the first step. Let $w : c_1, \ldots, c_n$, $n \geq 1$ be a route in G with edges $\epsilon_1, \ldots, \epsilon_{n-1}$. A node c_i is called a* ***collider*** *with respect to w if the edge ϵ_{i-1} is $c_{i-1} \to c_i$ and the edge ϵ_i is $c_i \leftarrow c_{i+1}$. We say w is* ***active*** *with respect to $C \subseteq N$ if: first, $\forall c_i$ which is a collider with respect to w, $c_i \in an_G(C)$; second, $\forall c_i$ which is not a collider with respect to w, $c_i \notin C$. If w is not active with respect to C, then we say w is* ***blocked*** *by C. Let $\langle A,\ B \mid C \rangle \in \mathcal{T}(N)$.*

- *Let H be the induced subgraph of G for $an_G(A \cup B \cup C) \cup (A \cup B \cup C)$. If C is a separator of A and B in the moral graph of H, then $\langle A,\ B \mid C \rangle$ is represented in G according to the* ***moralization criterion****.*

- *If every route from $a \in A$ to $b \in B$ is blocked by C, then $\langle A,\ B \mid C \rangle$ is represented in G according to the* ***d-separation criterion****.*

Remark 1.2.15. *Lauritzen et al showed in [38] that the moralization and the d-separation criterions for DAGs are equivalent. Another criterion appeared in [41] is a compromise between these two criterions, and we omit the details here. An example taken from [51, § 3.2] will be used to illustrate the two criterions.*

Example 1.2.16. *[51, § 3.2] Suppose a DAG G is given in Figure 1.4. Let $A = \{a\}$, $B = \{f\}$ and $C = \{c, d\}$. We want to see if $\langle A,\ B \mid C \rangle$ is represented in G.*

- ***moralization criterion.*** *Since $A \cup B \cup C = \{a, c, d, f\}$ and it has ancestor set $an_G(A \cup B \cup C) = \{a, b, d, e\}$, the induced subgraph is given in Figure 1.5(a) in which node g and all edges involved are removed. To build the moral graph for Figure 1.5(a), we first add an edge $[a, e]$ because they have a common child b, then we replace all directed edges with undirected edges. The resulting graph*

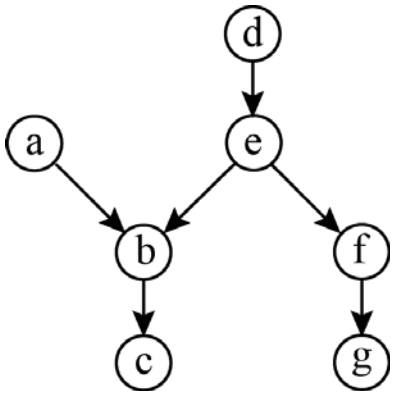
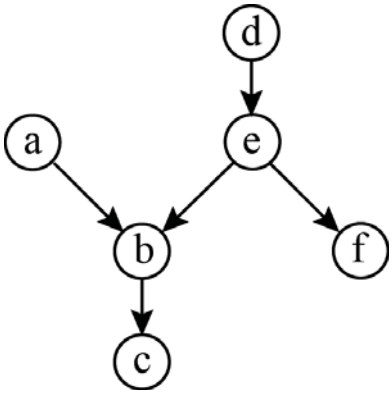


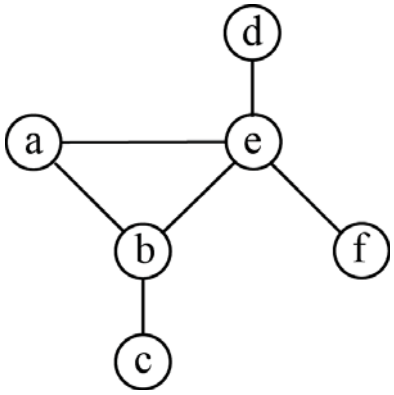
Figure 1.4: An example of acyclic directed graph to demonstrate graphical criterions

This DAG G has 7 nodes with $N = \{a, b, c, d, e, f, g\}$. Let $A = \{a\}$, $B = \{f\}$ and $C = \{c, d\}$. Consider $\langle A,\ B \mid C \rangle$ using moralization criterion and d-separation criterion.

is given in Figure 1.5(b). Now notice that $C = \{c, d\}$ is not a separator of $A = \{a\}$ and $B = \{f\}$ because we can find a path from a to f, $a - e - f$, that does not contain any node in C. Therefore, $\langle A,\ B \mid C \rangle$ is not represented in G according to moralization criterion.



(a) Induced subgraph of Figure 1.4



(b) The moral graph of Figure 1.5(a)

Figure 1.5: Graphs to illustrate moralization criterion for DAGs

- ***d-separation criterion.*** *Consider the route from a to f: $a \to b \leftarrow e \to f$. This route has one collider b which is in the ancestor set of C since $an_G(C) = \{a, b, d, e\}$. In addition, the other nodes in the route, a, e and f, do not belong*

to C. Hence, this route is active with respect to C, and this suggests that $\langle A,\ B \mid C\rangle$ is not represented in G according to d-separation criterion.

Graphical criterions for chain graphs are also available. The **moralization criterion for chain graphs** established by [36] and [24] is based on a definition of the moral graphs for chain graphs, and is a generalization of the moralization criterion for DAGs. An equivalent **c-separation criterion**, which generalizes the d-separation criterion for DAGs, was introduced in [5]. Some other criterions are also produced for other types of graphs. The details can be found in [51, § 3.3 – § 3.5].

Definition 1.2.17. *[51, § 3.1] Let P be a probability measure over N and G be a chain graph over N. Then P is called a* ***Markovian measure*** *with respect to G if*

$$A \perp\!\!\!\perp B \mid C\ [G] \Longrightarrow A \perp\!\!\!\perp B \mid C\ [P],\ \forall \langle A,\ B \mid C\rangle \in \mathcal{T}(N).$$

If, in addition, $A \perp\!\!\!\perp B \mid C\ [P]$ implies $A \perp\!\!\!\perp B \mid C\ [G]$, then we call P a ***perfectly Markovian measure****.*

Notice that if a Markovian measure P is not a perfectly Markovian measure, then it contains further valid CI statements that are not valid for the graph. In fact, there exist Markovian measures that are not representable by graphs, and this implies that the set of all graphical models is a strict (or proper) subset of all CI models. The following results have been done for the existence of perfectly Markovian measures:

- It was showed in [27, Theorem 11] that a perfectly Markovian discrete probability measure exists for every UG over N;

- Geiger and Pearl showed in [26] that a perfectly Markovian discrete probability measure exists for every DAG over N;

- the main result in [53] says that a perfectly Markovian positive discrete probability measure exists for every chain graph over N.

Definition 1.2.18. *[51, § 3.1] We say that two chain graphs G and H over N are* ***Markov equivalent*** *if the classes of Markovian measures with respect to G and H coincide, i.e. they induce the same conditional independence models. With this equivalence relation we can define equivalence classes to be the sets of the chain graphs where graphs in each class are Markov equivalent. We call them* ***Markov equivalence classes*** *of chain graphs.*

Remark 1.2.19. *The existence of perfectly Markovian measures for chain graphs implies that two chain graphs G and H are Markov equivalent if and only if $\mathcal{M}_G = \mathcal{M}_H$ [51, § 3.1]. There are a few results, which make it more intuitive to see if two graphs are Markov equivalent:*

- *two undirected graphs G and H are Markov equivalent if and only if $G = H$;*
- *two acyclic directed graphs G and H are Markov equivalent if and only if they have the same skeleton and the same immoralities [24];*
- *two chain graphs G and H are Markov equivalent if and only if they have the same skeleton and the same complexes [24].*

One should realize that a UG G is Markov equivalent with a DAG H if they have the same skeleton and H does not contain any immoralities.

In the rest of this section, we are going to study how to characterize and represent a Markov equivalence class of DAGs with a single graph [51, § 8.1].

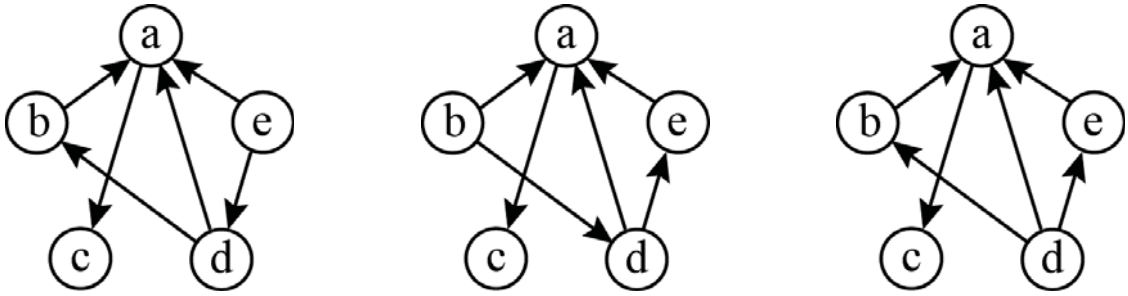
Definition 1.2.20. *[39, Definition 1.2.3] The* ***pattern*** *of a Markov equivalence class of DAGs is a hybrid graph having the same skeleton and the same immoralities as all DAGs in that class have. Given G in the class, we define $pat(G)$ as a pattern constructed from G.*

Definition 1.2.21. *[39, Definition 1.2.4] Consider a Markov equivalence class of DAGs. If a directed edge (i, j), $i \neq j$, is in every DAG in that class, then (i, j) is called a* ***protected*** *edge.*

Definition 1.2.22. *[39, Definition 1.2.5] The* ***essential graph*** *(or* ***completed pattern****) of a Markov equivalence class of DAGs is the pattern graph of that class in which all protected edges are directed.*

For a Markov equivalence class of DAGs, usually patterns are not unique, but the essential graph will be unique. Thus, essential graphs can serve as unique representatives of the equivalence classes of DAGs [3]. We have straightforward routines to construct $pat(G)$ and the essential graph from a DAG G, and also backwards (Example 1.2.23).

Example 1.2.23. *Consider the DAG G in Figure 1.6(a). Both G_1 in Figure 1.6(b) and G_2 in 1.6(c) are Markov equivalent with G because they have the exactly the same skeleton and immorality $b \to a \leftarrow e$.*



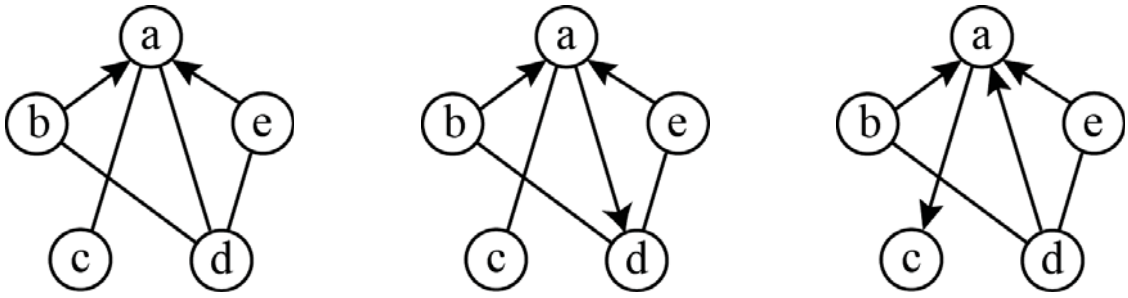
(a) Graph G (b) Graph G_1 (c) Graph G_2

Figure 1.6: A Markov equivalence class containing three graphs

- *To construct a pattern with respect to G, we keep all edges which are involved in the immorality, i.e. $b \to a$ and $a \leftarrow e$, directed, and convert all other edges to undirected edges. The resulting graph, given in Figure 1.7(a), is a pattern which*

contains the most undirected edges over all patterns. To reconstruct a DAG from a pattern, we simply add directions to all undirected edges while making sure no directed cycle or new immorality will be created. G_1 *and* G_2 *can be obtained from Figure 1.7(a) by this strategy. In fact,* G, G_1 *and* G_2 *are all DAGs we can find, so the corresponding Markov equivalence class is* $\{G, G_1, G_2\}$. *Notice Figure 1.7(b) is also a pattern, but we cannot reconstruct* G *with this pattern.*

- *Besides edges* $b \to a$ *and* $a \leftarrow e$ *involved in the immorality in* G, *we also find that edges* $a \to c$ *and* $a \leftarrow d$ *are contained in all DAGs in this Markov equivalence class. Thus these four edges are protected. Figure 1.7(c) gives the essential graph in which only the protected edges are directed. Similarly with patterns, to reconstruct a DAG from an essential graph, we add directions to all undirected edges while making sure no directed cycle or new immorality will be created.*



(a) A pattern (b) Another pattern (c) Essential graph

Figure 1.7: Patterns and the essential graph for the Markov equivalence class in Figure 1.6

Lemma 1.2.24. *[39, Corollary 1.2.6] Two DAGs* G *and* H *are Markov equivalent if and only if they have the same essential graph.*

1.2.4 Parameterization for discrete BNs and learning BNs using quality criterions

In this section, we focus on learning BNs in a discrete distribution framework with prescribed sample spaces, i.e. all probability measures on an arbitrary discrete sample space over N. We will introduce the parameterization for all discrete BNs, and some properties for quality criterions in learning BNs will be defined [51, § 8].

Given a set of random variables N, an element $i \in N$ can either be interpreted as a random variable or the corresponding node. We define $DAGs(N)$ as the collection of all DAGs over N. $X_N = \prod_{i \in N} X_i$ is a discrete joint sample space defined by a Cartesian product over X_i's, where X_i is a finite non-empty set which can be considered as the sample space of variable i, $i \in N$. $\forall A \subset N$, we can define $X_A = \prod_{i \in A} X_i$. Recall that the statistical model described by $G \in DAGs(N)$, $\mathbb{M}_G$, consists of the class of probability measures on X_N which are Markovian with respect to $G \in DAGs(N)$. **Data over** N, $DATA(N, d)$ with $d \in \mathbb{N}$, is a collection of all ordered sequences $\mathbf{x}^1, \ldots, \mathbf{x}^d$ where $\mathbf{x}^l \in \prod_{i \in N} X_i$ for $l = 1, \ldots, d$, i.e. the collection of all possible databases of length d. $\mathbf{x}^l$ is a vector which represents the lth observation.

Definition 1.2.25. *[51, § 8.2.1] Recall that $\forall P \in \mathbb{M}_G$ on X_N, P is uniquely determined by its density f. We can define the* ***marginal densities*** *of P:*

$$f_A(y) = \sum_{z \in X_{N \setminus A}} f(y, z), \text{ for } \emptyset \neq A \subset N,\ y \in X_A,$$

where $f_N \equiv f$, $f_\emptyset \equiv 1$ by convention. We can also define the ***conditional density*** *$f_{A|C}$ for disjoint A, $C \subseteq N$:*

$$f_{A|C}(x|z) = \begin{cases} \frac{f_{AC}(x,z)}{f_C(z)}, & if\ f_C(z) > 0 \\ 0, & if\ f_C(z) = 0 \end{cases} \quad for\ x \in X_A,\ z \in X_C.$$

Lemma 1.2.26 (Recursive Factorization)**.** *[38, Theorem 1] $P \in \mathbb{M}_G$ if and only if its density* ***recursively factorizes*** *with respect to G:*

$$f(\mathbf{x}) = \prod_{i \in N} f_{i|pa_G(i)}(x_i|x_{pa_G(i)}) \quad for\ every\ \mathbf{x} \in X_N.$$

Definition 1.2.27. *[51, § 8.2.1] Consider $G \in DAGs(N)$. $\forall i \in N$ and $\mathbf{x} = (x_i)_{i \in N} \in X_N$, we define x_A as the observed value for $A \subseteq N$ in $\mathbf{x}$,*

- *define $r(i) := |X_i| \geq 1$ as the number of possible values of random variable i, and $y_i^1, \ldots, y_i^{r(i)}$ is an ordering of elements of X_i, where y_i^k is the kth* ***node configuration*** *in the ordering, $k = 1, \ldots, r(i)$. $k(i, \mathbf{x})$ is the symbol for the unique k, $k \in \{1, \ldots, r(i)\}$, such that $y_i^k = x_i$;*

- *define $q(i, G) \equiv |X_{pa_G(i)}| = \prod_{l \in pa_G(i)} r(l) \geq 1$ as the number of* ***parent configurations*** *for random variable i where $q(i, G) = 1$ when $pa_G(i) = \emptyset$, and $z_i^1, \ldots, z_i^{q(i,G)}$ is an ordering of elements of $X_{pa_G(i)}$, where z_i^j is the jth* ***parent configuration*** *in the ordering, $j = 1, \ldots, q(i, G)$. $j(i, \mathbf{x})$ is the symbol for the unique j, $j \in \{1, \ldots, q(i, G)\}$, such that $z_i^j = x_{pa_G(i)}$, where $j = 1$ if $pa_G(i) = \emptyset$.*

Based on the recursive factorization (Lemma 1.2.26), a "standard" parameterization of $\mathbb{M}_G$ can be given by a set of parameters Θ_G which consists of vectors:

$$\begin{aligned} \boldsymbol{\theta} \equiv (\theta_{ijk})\ &where\ \theta_{ijk} \in [0, 1] \\ &for\ i \in N,\ j \in \{i, \ldots, q(i, G)\},\ k \in \{1, \ldots, r(i)\}, \\ &such\ that\ \sum_{k=1}^{r(i)} \theta_{ijk} = 1 for\ every\ i \in N,\ 1 \leq j \leq q(i, G), \end{aligned}$$

where every single θ_{ijk} can be interpreted as the value of the conditional density $f_{i|pa_G(i)}(y_i^k|z_i^j)$. Therefore for a specific $\boldsymbol{\theta} \in \Theta_G$, we have:

$$f^{\boldsymbol{\theta}}(\mathbf{x}) = \prod_{i \in N} \theta_{i\ j(i,\mathbf{x})\ k(i,\mathbf{x})} \quad for\ \mathbf{x} \in X_N.$$

Based on this parameterization, it is straightforward to define the ***numbers of configuration occurrences*** *in the database $D \in DATA(N,d)$ where D consists of d observations $\mathbf{x^1},\ldots,\mathbf{x^d} \in X_N$:*

$$\begin{aligned} d_{ij} &= |\{1 \le l \le d\};\ x^l_{pa_G(i)} = z_i^j|, \\ d_{ijk} &= |\{1 \le l \le d\};\ x^l_{\{i\}\cup pa_G(i)} = (y_i^k, z_i^j)|, \\ &\qquad for\ i \in N,\ j \in \{1,\ldots,q(i,G)\},\ k \in \{1,\ldots,r(i)\}, \\ d_{[x]} &= |\{1 \le l \le d\};\ x^l_A = x| \quad for\ \emptyset \ne A \subseteq N,\ x \in X_A, \end{aligned}$$

where $d_{i1} = d$ *if* $pa_G(i) = \emptyset$.

Remark 1.2.28. *Given $G \in DAGs(N)$ and $D \in DATA(N,d)$, the numbers of configuration occurrences can be considered as statistics used to estimate parameters:*

- $\frac{d_{ij}}{d}$ *is an estimator of* $f_{pa_G(i)}(z_i^j)$;
- $\frac{d_{ijk}}{d_{ij}} I_{d_{ij}>0} + \frac{1}{r(i)} I_{d_{ij}=0}$, *is an estimator of* $\theta_{ijk} = f_{i|pa_G(i)}(y_i^k|z_i^j)$ *[51, Lemma 8.1];*
- $\frac{d_{[x]}}{d}$ *is an estimator of* $f_A(x)$,

$\forall\ i \in N,\ j \in \{1,\ldots,q(i,G)\},\ k \in \{1,\ldots,r(i)\}$.

Example 1.2.29. *Figure 1.8 gives a DAG G over $N = \{a,b,c\}$. Suppose all three random variables are binary. Since $pa_G(a) = \emptyset$, $pa_G(b) = \{a\}$ and $pa_G(c) = \{a,b\}$, we have $f(\mathbf{x}) = f_a(x_a) f_{b|a}(x_b|x_a) f_{c|ab}(x_c|x_{ab})$, $\forall \mathbf{x} \in X_N$. Now take node*

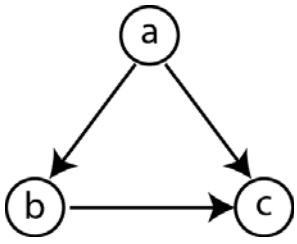


Figure 1.8: An example of parameterization and computing the numbers of configuration occurrences

c for example. We have $r(c) = |X_c| = |\{0,1\}| = 2$ *and* $q(c,G) = |X_{pa_G(c)}| = |\{(1,1),(1,0),(0,1),(0,0)\}| = 4$. *Fix an ordering of elements in* X_c, $(y_c^1, y_c^2) = (1,0)$, *and an ordering of elements in* X_{ab}, $(z_c^1, z_c^2, z_c^3, z_c^4) = (11,10,01,00)$. *Suppose* $D \in DATA(N,3)$ *has observations: (1, 0, 1), (0, 1, 1), and (1, 1, 0). Then* d_{c21} *is the number of observations in which* $x_{ab} = z_c^2 = (10)$ *and* $x_c = y_c^1 = 1$, *i.e.* $d_{c21} = 1$. *Thus the list of all* d_{cjk}, $j = 1,2,3,4$, $k = 1,2$, *are:*

$$d_{c11} = 0\ (1|11) \quad d_{c21} = 1\ (1|10) \quad d_{c31} = 1\ (1|01) \quad d_{c41} = 0\ (1|00)$$
$$d_{c12} = 1\ (0|11) \quad d_{c22} = 0\ (0|10) \quad d_{c32} = 0\ (0|01) \quad d_{c42} = 0\ (0|00).$$

Similarly, we can figure out the list of all d_{ajk}, $j = 1$, $k = 1,2$, *and* d_{bjk}, $j = 1,2$, $k = 1,2$*:*

$$d_{a11} = 2\ (1|\emptyset) \quad d_{b11} = 1\ (1|1) \quad d_{b21} = 1\ (1|0)$$
$$d_{a12} = 1\ (0|\emptyset) \quad d_{b12} = 1\ (0|1) \quad d_{b22} = 0\ (0|0).$$

On the other hand, the vector of parameters is $\boldsymbol{\theta} = (\theta_{a11}, \theta_{a12}, \theta_{b11}, \theta_{b21}, \theta_{b12}, \theta_{b22}, \theta_{c11}, \theta_{c21}, \theta_{c31}, \theta_{c41}, \theta_{c12}, \theta_{c22}, \theta_{c32}, \theta_{c42})$.

To conduct a model selection in BNs, we choose a **score function** which is a function measuring how good a certain BN structure given by a $G \in DAGs(N)$ fits to the given database $D \in DATA(N,d)$. We define a **quality criterion** as a score function $\mathcal{Q} : DAGs(N) \times DATA(N,d) \to \mathbb{R}$ assigning a real number $\mathcal{Q}(G,D)$ to a DAG G and a database D. For a given D and a BN structure determined by a G, the higher (or lower, depending on how criterion $\mathcal{Q}$ is defined) the value $\mathcal{Q}(G,D)$ is, the better the structure fits the data. Hence using a proper quality criterion is important in model selection in BN.

Definition 1.2.30. *[51, § 8.2.2] A quality criterion* $\mathcal{Q}$ *for learning DAG models is* ***score equivalent*** *if for every pair* G, $H \in DAGs(N)$ *and every* $D \in DATA(N,d)$*:*

$$\mathcal{Q}(G,D) = \mathcal{Q}(H,D) \text{ whenever } G \text{ and } H \text{ are Markov equivalent.}$$

This property is natural and necessary because if G and $H \in DAGs(N)$ are Markov equivalent, then the CI models induced by these two graphs coincide, i.e. $\mathbb{M}_G = \mathbb{M}_H$. Thus their scores for the same data should be equal.

Consider $D \in DATA(N, d)$, $D : x^1, \ldots, x^d$. For A, $\emptyset \neq A \subseteq N$, we call $D_A \in DATA(A, d) : x_A^1, \ldots, x_A^d$ a **projection of** D **onto** A [51, § 8.2.3].

Definition 1.2.31. *[51, § 8.2.3] A quality criterion $\mathcal{Q}$ for learning DAG models is* ***decomposable*** *if there exists a class of functions $q_{i|B} : DATA(\{i\} \cup B, d) \to \mathbb{R}$ where $i \in N$, $B \subseteq N \backslash \{i\}$, such that:*

$$\mathcal{Q}(G, D) = \sum_{i \in N} q_{i|pa_G(i)}(D_{i \cup pa_G(i)}),$$

for every $G \in DAGs(N)$ and every $D \in DATA(N, d)$. Notice here the functions $q_{i|B}$ do not depend on G.

This property means that the overall score can be decomposed into (i.e. can be written as the sum of) local scores where each local score only depends on one single node and its parents.

Definition 1.2.32. *[51, § 8.2.4] A quality criterion $\mathcal{Q}$ for learning DAG models is* ***regular*** *if there exists a class of functions $\mathbf{t}_A : DATA(A, d) \to \mathbb{R}$, $\emptyset \neq A \subseteq N$ and a constant $\mathbf{t}_\emptyset(D_\emptyset)$ depending on X_N and d, such that:*

$$\mathcal{Q}(G, D) = \sum_{i \in N} (\mathbf{t}_{i \cup pa_G(i)}(D_{i \cup pa_G(i)}) - \mathbf{t}_{pa_G(i)}(D_{pa_G(i)})),$$

for every $G \in DAGs(N)$ and every $D \in DATA(N, d)$.

If a quality criterion is regular, then it means that in addition to decomposable property, each local score of the corresponding node and its parents can be further decomposed into a difference between a score of the node and its parents and a score of the parents only.

Example 1.2.33. *Continue to use the graph G and database D in Example 1.2.29. D can be written in form of matrix:* $\begin{pmatrix} \mathbf{x^1} \\ \mathbf{x^2} \\ \mathbf{x^3} \end{pmatrix} = \begin{array}{c} \begin{matrix} a & b & c \end{matrix} \\ \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} \end{array}$. *Take node b for example. $D_{b\cup pa_G(b)} = D_{ab}$ is a submatrix of D which contains the first two columns of D, while function $q_{b|pa_G(b)}$ and $\mathbf{t}_{\{a,b\}}$ map $D_{b\cup pa_G(b)}$ to real numbers.*

Lemma 1.2.34. *[51, Lemma 8.3] Assume $r(i) \geq 2$, $\forall i \in N$. A quality criterion $\mathcal{Q}$ for learning DAG models is regular if and only if it is decomposable and score equivalent.*

It was proven in [51, Proposition 8.1 and Proposition 8.2] that the Akaike's information criterion (AIC) and the Bayesian information criterion (BIC), two quality criterions that are most frequently used, are regular criterions. In Section 1.3.2 we will discuss more about how these properties help us in learning BNs.

1.3 Polytopes arising from contingency tables and Bayesian networks

First of all, we need to recall some basic notation and definitions ([59] is used as a main reference). Let $\{\mathbf{x_1}, \ldots, \mathbf{x_k}\} \subset \mathbb{R}^d$ be a finite set of points. A point $\mathbf{x} \in \mathbb{R}^d$ is called a **convex combination** of $\{\mathbf{x_1}, \ldots, \mathbf{x_k}\}$ if it can be written as

$$\mathbf{x} = \sum_{i=1}^{k} \alpha_i \mathbf{x_i}, \text{ where } \sum_{i=1}^{k} \alpha_i = 1 \text{ and } \alpha_i \geq 0 \text{ for } i = 1, \ldots, k.$$

The following definitions of "convex" and "convex hull" can be found in [59, P3]. A point set $S \subseteq \mathbb{R}^d$ is **convex** if with any two points $\mathbf{x}$, $\mathbf{y} \in S$ it also contains the straight line segment $[\mathbf{x}, \mathbf{y}] = \{\lambda\mathbf{x} + (1-\lambda)\mathbf{y} : 0 \leq \lambda \leq 1\}$ between them. Clearly, every intersection of convex sets is convex. Thus for any $S \subseteq \mathbb{R}^d$, the **convex hull** of S can be defined as the "smallest" convex set containing S, which can be constructed

as the intersection of all convex sets that contain S:

$$conv(S) := \bigcap \{S' \subseteq \mathbb{R}^d : S \subseteq S', S' convex\}.$$

Note that this definition is equivalent to $conv(S) := \bigcup \{[\mathbf{x}, \mathbf{y}] : \mathbf{x}, \mathbf{y} \in S\}$.

Definition 1.3.1. *[59, Definition 0.1] A **convex polytope (polytope)** is the convex hull of a finite set of points in some $\mathbb{R}^d$.*

Remark 1.3.2. *[59, P4 and Theorem 1.1] A **polyhedron** is a set $\mathbf{P} \subseteq \mathbb{R}^d$ presented in the form:*

$$\mathbf{P} = \{\mathbf{x} \in \mathbb{R}^d : A\mathbf{x} \leq b\} \text{ for some } A \in \mathbb{R}^{r \times d},\ b \in \mathbb{R}^r,$$

*where $A\mathbf{x} \leq b$ is the system of inequalities that defines the polyhedron. A **polytope** can also be defined as a **bounded** polyhedron in the sense that it does not contain a ray $\{\mathbf{x} + t\mathbf{y} : t \geq 0\}$ for any $\mathbf{y} \neq \mathbf{0}$. This definition is equivalent with Definition 1.3.1 [59, Theorem 1.1]. Thus a polytope has two representations: the convex hull of a finite set of points, or bounded polyhedron that is defined by a system of inequalities.*

Definition 1.3.3. *[59, P3] Let S be a set in $\mathbb{R}^d$. The **affine hull** $aff(S)$ of S is the set of all **affine combinations** of elements of S, that is,*

$$aff(S) = \Big\{ \sum_{i=1}^{k} \alpha_i \mathbf{x_i} : k \in \mathbb{Z}^+, \mathbf{x_i} \in S, \alpha_i \in \mathbb{R}, \sum_{i=1}^{k} \alpha_i = 1 \Big\}.$$

Now the **dimension** of a polytope can be well defined as the dimension of its affine hull, and a **d-polytope** is a polytope of dimension d in some $\mathbb{R}^{d'}$ $(d' \geq d)$.

Definition 1.3.4. *[59, § 2.1, Definition 2.1 and Proposition 2.2] Let $\mathbf{P} \subseteq \mathbb{R}^d$ be a convex polytope. A linear inequality $w\mathbf{x} \leq w_0$ is **valid** for $\mathbf{P}$ if it is satisfied for all points $\mathbf{x} \in \mathbf{P}$. A **face** of $\mathbf{P}$ is any set of the form*

$$F = \mathbf{P} \cap \{\mathbf{x} \in \mathbb{R}^d : w\mathbf{x} = w_0\}$$

where $w\mathbf{x} \leq w_0$ is a valid inequality for $\mathbf{P}$. In this case w is called a ***cost vector*** *for face F.*

The ***dimension of a face*** *is the dimension of its affine hull: $dim(F) := dim(aff(F))$. A face of $\mathbf{P}$ is called a* ***proper face*** *if the corresponding cost vector is not an all-zero vector.*

The faces of dimensions 0, 1, $dim(\mathbf{P})-2$, and $dim(\mathbf{P})-1$ are called ***vertices***, ***edges***, ***ridges*** *and* ***facets***, *respectively. The set of all vertices of $\mathbf{P}$ is called the* ***vertex set***, *and is defined as $vert(\mathbf{P})$. An important fact is that every polytope is the convex hull of its vertices: $\mathbf{P} = conv(vert(\mathbf{P}))$. v^1, $v^2 \in vert(\mathbf{P})$ are called* ***neighbors*** *if they form an edge on $\mathbf{P}$.*

Remark 1.3.5. *Based on Definition 1.3.4, the following statements are trivial:*

- *v is a vertex of $\mathbf{P}$ if and only if $\exists$ a vector w^v such that $\forall v' \in vert(\mathbf{P})$, $w^v v' \leq w^v v$ where "=" holds if and only if $v' = v$. In fact, w^v is a cost vector for v;*
- *v^1 and v^2 form an edge on $\mathbf{P}$ if and only if $\exists$ a vector w^e such that $\forall v^3 \in vert(\mathbf{P})$, $w^e v^3 \leq w^e v^1 = w^e v^2$ where "=" holds if and only if $v^3 = v^1$ or $v^3 = v^2$. In fact, w^e is a cost vector for the edge formed by v^1 and v^2.*

Here are several special types of polytopes. A **d-simplex**, which is denoted by Δ_d, is a polytope of dimension d with $d+1$ vertices. A **d-dimensional simple polytope** is a d-polytope each of whose vertices are adjacent to exactly d edges (or facets), i.e. each vertex has exactly d neighbors. It is worth pointing out that all simplices are simple polytopes, and every pair of vertices of every simplex are neighbors. The **d-dimensional hypercube (d-cube)** is defined as $\mathbf{C}_d := conv\{\{+1, -1\}^d\}$, and is also a simple polytope.

1.3.1 Connection between polytopes and SIS procedure

For a convex polyhedron, we also call the integer points inside the polyhedron the **lattice points**. Recall that in Section 1.1.1 we vectorize an arbitrary contingency table $\mathbf{X}$ as $\mathbf{X} = (x_1, \cdots, x_t)$, where t is the number of cells in $\mathbf{X}$. In another point of view $\mathbf{X}$ can also be considered as a lattice point in the Euclidean space $\mathbb{R}^t$, in which sense the set $\Sigma = \{\mathbf{X} \in \mathbb{Z}^t \mid A\mathbf{X} = b, \mathbf{X} \geq 0\}$ in Equation (1.1.5) is exactly the set of lattice points inside the polytope $\mathbf{P} = \{\mathbf{X} \in \mathbb{R}^t \mid A\mathbf{X} = b, \mathbf{X} \geq 0\}$. Thus a procedure of sampling a contingency table with given linear constraints also gives a method to sample over the set of of lattice points inside the corresponding polytope, and the problem of estimating the number of contingency tables is equivalent to estimating the number of lattice points inside the polytope.

In Section 1.1.1 we showed that in order to sample the table $\mathbf{X}$ sequentially by cells, we need to achieve the lower bound and upper bound of the support of marginal distributions $q(x_1)$ and $q(x_i|x_{i-1}, \ldots, x_2, x_1)$, $i = 2, \ldots, t$, and this problem can be converted to an optimization problem of a linear objective function over a feasible region that is defined by the linear constraints. Three techniques are available: linear programming / LP (lpSolve package in R), integer programming / IP (lpSolve package in R) and shuttle algorithm [22]. Take $q(x_1)$ for example. After assigning a proper objective function $\mathbf{c^T X}$, where $\mathbf{c^T} = (1, 0, \ldots, 0)$, the problem of computing lower bound l_1 and upper bound u_1 for x_1 can be either converted to LP problems:

$$l_1 = \lceil -\max_{\mathbf{X}\in\mathbf{P}}(-\mathbf{c})^{\mathbf{T}}\mathbf{X} \rceil = \lceil \min_{\mathbf{X}\in\mathbf{P}} x_1 \rceil, \text{ and } u_1 = \lfloor \max_{\mathbf{X}\in\mathbf{P}} \mathbf{c^T X} \rfloor = \lfloor \max_{\mathbf{X}\in\mathbf{P}} x_1 \rfloor, \tag{1.3.1}$$

where the floor function $\lfloor \cdot \rfloor$ maps a real number to the largest integer not greater than it and the ceiling function $\lceil \cdot \rceil$ maps a real number to the smallest integer not less than it, or converted to IP problems:

$$l_1 = -\max_{\mathbf{X}\in\Sigma}(-\mathbf{c})^{\mathbf{T}}\mathbf{X} = \min_{\mathbf{X}\in\Sigma} x_1, \text{ and } u_1 = \max_{\mathbf{X}\in\Sigma} \mathbf{c^T X} = \max_{\mathbf{X}\in\Sigma} x_1. \tag{1.3.2}$$

It is clear that IP gives the exact values of the bounds and LP gives an approximation of them. Although using LP to obtain bounds is much faster than using IP, we should be very careful about this because in some situations, like the case of transportation polytopes [14, 16], the bounds computed by LP can be very different with the ones computed by IP.

Notice that the form in Equation (1.3.1) is called the augmented form of LP problems and is important because LP problems must be converted into this form before being solved by the simplex algorithm. The form in Equation (1.3.2) is the standard form of IP problems. The reason that in Section 1.1.2 slack variables must be introduced in the system of linear equations is that the set Σ and polytope $\mathbf{P}$ must be written in the forms showed before so that the LP and IP problems which we need to solve will have the forms in Equations (1.3.1) and (1.3.2).

Algebraic geometry also has connection to the rejections in SIS procedures that is caused by holes in semigroups and to the sequential interval property introduced in Section 1.1.3. Let the column vectors of A be $\mathbf{a}_1, \ldots, \mathbf{a}_t$. Define the **semigroup** generated by $\mathbf{a}_i, \ldots, \mathbf{a}_t$:

$$Q_i = \{\mathbf{a}_i x_i + \ldots + \mathbf{a}_t x_t \mid x_1, \ldots, x_t \in \mathbb{Z}_+\},$$

the **cone** generated by $\mathbf{a}_i, \ldots, \mathbf{a}_t$:

$$K_i = \{\mathbf{a}_i x_i + \ldots + \mathbf{a}_t x_t \mid x_1, \ldots, x_t \in \mathbb{R}_+\},$$

and the **lattice** generated by $\mathbf{a}_i, \ldots, \mathbf{a}_t$:

$$L_i = \{\mathbf{a}_i x_i + \ldots + \mathbf{a}_t x_t \mid x_1, \ldots, x_t \in \mathbb{Z}\},$$

where $i = 1, \ldots, t$. The semigroup $Q_i^{sat} = K_i \cap L_i$ is called the **saturation** of the semigroup Q_i. Obviously $Q_i \subset Q_i^{sat}$. If they are equal, then we say Q_i is **saturated** (or **normal**), if not, then we define $H_i = Q_i^{sat} \backslash Q_i \neq \emptyset$ as the set of **holes** of the semigroup Q_i. Some examples of holes can be found in [55].

In SIS procedure, $\Sigma \neq \emptyset$ implies $b \in Q_1$. After x_i, $i = 1, \ldots, t-1$ is sampled, if there exist holes in the semigroup Q_{i+1} and $(b - \mathbf{a}_1 x_1 - \ldots - \mathbf{a}_i x_i) \in H_{i+1} \neq \emptyset$, then the remaining linear constraints no longer have feasible solution, which means we must reject the sample in process. If A has the sequential interval property introduced in Section 1.1.3 with respect to the ordering $x_1, x_2, \ldots, x_t$, then the sequence of semigroups defined above, $Q_1, Q_2, \ldots, Q_t$ are all saturated, i.e. we won't have any rejection because of the holes in semigroups.

1.3.2 Characteristic imset polytopes (cim-polytopes) for Bayesian networks

For a given set of random variables N, in general there are super exponentially many Markov equivalence classes over $DAGs(N)$. Hence the model selection in BNs, which proceeds with maximizing a quality criterion $\mathcal{Q}(G, D)$ over all possible Markov equivalence classes, is known to be an NP-hard problem [11, 42]. A basic idea of an algebraic and geometric approach in learning BNs is given in [51]: represent every BN structure by a vector which is uniquely determined. This yields a geometric understanding of learning BNs that new results and insights can be obtained from this point of view. In this section, we will first introduce the standard imsets [51, § 7.2], which are algebraic representations of classes of Markov equivalent DAGs. Then we will introduce another type of representations, the characteristic imsets [52], which we use in Chapter 3. Lastly, we will show how these representations can help us to formulate a model selection problem in BNs as an LP problem over a polytope.

Define an **imset** as a function $u : \mathcal{P}(N) \mapsto \mathbb{Z}$, where $\mathcal{P}(N) := \{T \mid T \subseteq N\}$ is the power set of N. A special imset $\delta : \mathcal{P}(N) \to \{0, 1\}$ with $\delta_T(S) = \begin{cases} 1 & \text{if } S = T, \\ 0 & \text{if } S \neq T, \end{cases}$ for $S \subseteq N$, is called the **identifier of a subset** T of N.

Definition 1.3.6. *[51, § 7.2.1] Given $G \in DAGs(N)$, the **standard imset** for G*

is an imset $u_G : \mathcal{P}(N) \to \mathbb{R}$ *with*

$$u_G = \delta_N - \delta_\emptyset + \sum_{i \in N} \Big\{ \delta_{pa_G(i)} - \delta_{\{i\} \cup pa_G(i)} \Big\}.$$

Corollary 1.3.7. *[51, Corollary 7.1] Let* G, $H \in DAGs(N)$. *Then* $\mathcal{M}_G = \mathcal{M}_H$ *if and only if* $u_G = u_H$.

Corollary 1.3.7 means that two graphs are Markov equivalent if and only if they have the same standard imsets. In this sense, we say standard imset is a **unique vector representative** for BN structures. On the other hand, the product formulas induced by standard imsets also characterize Markovian measures (Example 1.3.8).

Example 1.3.8. *A graph* $G \in DAGs(N)$ *is given in Figure 1.9, where* $N = \{a, b, c\}$. *The standard imset* u_G *is a vector of length* $|\mathcal{P}(N)| = 8$ *and its coordinates are* $u_G(T)$, $T \subseteq N$. *By Definition 1.3.6,* G *has standard imset* $u_G = \delta_{abc} - \delta_\emptyset + (\delta_b - \delta_{ab}) + (\delta_\emptyset - \delta_b) + (\delta_a - \delta_{ac})$, *i.e.*

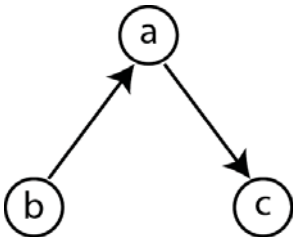


Figure 1.9: An example of constructing a standard imset

$$\begin{array}{c} \quad \emptyset \quad a \quad b \quad c \quad ab \quad ac \quad bc \quad abc \\ u_G = \Big(0, \quad 1, \quad 0, \quad 0, \quad \text{-}1, \quad \text{-}1, \quad 0, \quad 1 \Big) \end{array}$$

Now consider a product formulas induced by u_G*:*

$$\prod_{T \subseteq N,\ u_G(T) > 0} (f_T(x_T))^{u_G(T)} = \prod_{T \subseteq N,\ u_G(T) < 0} (f_T(x_T))^{u_G(T)}.$$

In this example we get:

$$f_a(x_a)f_{abc}(x_{abc}) = f_{ab}(x_{ab})f_{ac}(x_{ac}), \tag{1.3.3}$$

which implies the recursive factorization formula

$$f_{abc}(x_{abc}) = f_b(x_b)f_{a|b}(x_a|x_b)f_{c|a}(x_c|x_a).$$

Hence the Equation (1.3.3) *characterize the Markovian measures with respect to G:* $\mathbb{M}_G = \{P : P$ *is a probability measure over* N *which satisfies Equation* (1.3.3) $\}$.

In [52], Studený et al proposed another imset, the characteristic imset, which is an alternative vector representative of BN structures and can be obtained from the standard imset by an affine linear transformation.

Definition 1.3.9. *[52, Definition 1] For* $G \in DAGs(N)$*, the* ***characteristic imset*** c_G *for* G *is given by:*

$$c_G(T) = 1 - \sum_{S,T\subseteq S\subseteq N} u_G(S), \tag{1.3.4}$$

for $T \subseteq N$, $|T| \geq 2$.

The mapping in Equation (1.3.4) is invertible: we can obtain standard imsets from characteristic imsets by a Möbius inversion [4], which is also an affine linear transformation [52, Equation 4]:

$$u_G(S) = \sum_{T,S\subseteq T\subseteq N} (-1)^{|T\setminus S|} \cdot (1 - c_G(T))$$

for $S \subseteq N$, $|S| \geq 2$.

Theorem 1.3.10. *[52, Theorem 1] For* $G \in DAGs(N)$*, we have* $c_G(T) \in \{0, 1\}$ *for any* $T \subseteq N$, $|T| \geq 2$. *Moreover,* $c_G(T) = 1$ *if and only if there exists* $i \in T$ *with* $A\setminus\{i\} \subseteq pa_G(i)$.

Because of the linear transformations between characteristic imsets and standard imsets, it is clear that characteristic imsets are still unique vector representatives for BN structures. In addition, Theorem 1.3.10 showed that characteristic imsets are 0-1 vectors, and they are very intuitive in terms of graphs. Sometimes this theorem is referenced as the definition of characteristic imset.

Recall the regular criterions introduced in Section 1.2.4, Studený showed that regular criterions can be written as functions of standard imsets [51].

Theorem 1.3.11. *[51, Lemma 8.7] $\mathcal{Q}$ is regular, then there exists unique $s : DATA(N, d) \to \mathbb{R}$ and mapping $t : D \in DATA(N, d) \mapsto t_D \in \mathbb{R}^{\mathcal{P}(N)}$ s.t.:*

- $\forall T \subseteq N$ $|T| \leq 1$, $t_D(T) = 0$, $\forall D \in DATA(N, d)$,
- $\forall T \subseteq N$ $|T| \geq 2$, *mapping* $D \mapsto t_D(T)$ *depends on* D_T, *and* $\forall G \in DAGS(N)$ *and* $\forall D \in DATA(N, d)$, *the following holds:*

$$\mathcal{Q}(G, D) = s(D) - \langle t_D, u_G \rangle.$$

In [51, Proposition 8.4 and Corollary 8.6], Studený gave the formulas of $s(D)$ and t_D for criterions the maximized log-likelihood criterion (MML), AIC and BIC with the standard parameterization introduced in Section 1.2.4. More details about Theorem 1.3.11 can be found in [51, § 8.4.2].

Consider a class of graphs $\mathcal{G} \subseteq DAGs(N)$ that contains all graphs which we are interested in. We call the polytope $\mathbf{P}_{\mathcal{G},s} = conv\{u_G : G \in \mathcal{G}\}$ the **standard imset polytope** (or **sim-polytope**) for $\mathcal{G}$, and the polytope $\mathbf{P}_{\mathcal{G},c} = conv\{c_G : G \in \mathcal{G}\}$ the **characteristic imset polytope** (or **cim-polytope**) for $\mathcal{G}$. Then the only integer points in $\mathbf{P}_{\mathcal{G},s}$ and in $\mathbf{P}_{\mathcal{G},c}$, respectively, are their vertices [39, Lemma 2.1.4]. Moreover, we have $vert(\mathbf{P}_{\mathcal{G},s}) = \{u_G : G \in \mathcal{G}\}$ [54] and $vert(\mathbf{P}_{\mathcal{G},c}) = \{c_G : G \in \mathcal{G}\}$ ($\mathbf{P}_{\mathcal{G},c}$ is a truncation of a hypercube). Theorem 1.3.11 is remarkable in the sense that it formulates the problem of maximizing a regular criterion with a given data

$D \in DATA(N, d)$, i.e. $\max_{G \in \mathcal{G}} \mathcal{Q}(G, D)$, as an LP problem over $\mathbf{P}_{\mathcal{G},s}$: $\min_{\mathbf{x} \in \mathbf{P}_{\mathcal{G},s}} t_D^T \mathbf{x}$, where t_D is determined by D. This gives us a systematic way to find the best model with the optimality certificate rather than finding it by the brute-force search. Notice that this LP problem can be further converted to another LP problem over $\mathbf{P}_{\mathcal{G},c}$: $\min_{\mathbf{x} \in \mathbf{P}_{\mathcal{G},c}} r_D^T \mathbf{x}$, where r_D is a data vector revised from t_D with an affine linear transformation [39, Definition 2.1.5 and Lemma 2.1.6].

1.4 Main results and outline of the dissertation

This dissertation consists of two parts, and main results are summarized as following with respect to different parts.

- Estimating the number of zero-one multi-way tables via SIS procedures.
  - An SIS procedure with CP distribution is constructed for sampling zero-one three-way tables with fixed two-way marginals.

    The underlying model is the no three-way interaction model introduced in Section 2.1. Theorem 2.2.2 in Section 2.2 generalizes Theorem 1.1.3 and gives the marginal distribution of each column in a zero-one three-way table under the no three-way interaction model. The computational results for simulations (see Section 2.4) and Sampson's dataset (see Section 2.5) are based on the **R** code in Appendix.
  - An SIS procedure with CP distribution is constructed for sampling zero-one d-way tables ($d \geq 2$) with fixed $(d-1)$-way marginals.

    Theorem 2.3.2 in Section 2.3 further generalizes Theorem 2.2.2 and gives the marginal distribution of each column in a zero-one d-way table under the no d-way interaction model.
- The Characteristic Imset Polytopes for Bayesian Networks.

- A combinatorial description of all edges and the system of inequalities which defines all facets for $\mathbf{P}_{m,n}$, the characteristic imset polytopes for diagnosis models, are given in Section 3.2.

  Diagnosis models are defined in Section 3.1. Based on the properties of diagnosis models (see Section 3.1), we give a graphical description of all edges in $\mathbf{P}_{m,n}$, and show that $\mathbf{P}_{m,n}$ is a direct product of a sequence of simplices (see Section 3.2.1). Then we figure out the inequalities for all facets in $\mathbf{P}_{m,n}$ (see Section 3.2.2).

- A combinatorial description of all edges and the system of inequalities which defines all facets for $\mathbf{P}_{[n]}$, the characteristic imset polytopes for Bayesian networks with a fixed underlying ordering, are given in Section 3.3.

  Results are similar with Section 3.2. We show that $\mathbf{P}_{[n]}$ is also a direct product of a sequence of simplices, and all edges and facets can be computed based on this structure and the results in Section 3.2.

  A further generalization of these results for $\mathbf{P}_{\mathcal{G}_{[n],\Omega,c}}$, the characteristic imset polytopes for Bayesian networks with a fixed underlying ordering where some edges are forbidden, is discussed in Section 4.2.1.

In Chapter 4, we discuss the results in Chapter 2 and Chapter 3. We also talk about some open problems and future work on these two topics.

Chapter 2 Estimating the Number of Zero-One Multi-way Tables via Sequential Importance Sampling

Much work has been done on sampling multi-way contingency tables without zero-one constraints using SIS procedures. In [9], Chen et al introduced also an SIS procedure for sampling multi-way contingency tables without zero-one constraints, and in [10], Chen et al gave an excellent algebraic interpretation of precisely when an interval will equal the support of the marginal distribution using Markov basis (see Section 1.1.3). In [21], Dinwoodie and Chen used linear programming and sequential normal sampling to develop a new SIS procedure to sample a multi-way contingency table.

However, one cannot just simply apply these methods to sampling multi-way contingency tables with zero-one constraints. The reason is that we have to introduce "slack" variables to the system of the linear equations, which doubles the number of variables and makes the problem exponentially harder (see Sections 1.1.2 and 1.3.1). This is also why in [9] Chen et al developed an SIS procedure specifically for sampling zero-one two-way contingency tables (see Section 1.1.2). Therefore, we have to consider the problem of sampling zero-one multi-way contingency tables separate from the existing methods for sampling contingency tables without zero-one constraints.

In this chapter, we first introduce the model we consider, the no three-way interaction model, and explain why this model is important. Secondly, we generalize the SIS procedure on zero-one two-way tables (reviewed in Section 1.1.2) to an SIS procedure on zero-one three-way tables under the no three-way interaction model. In the third section we extend our method to zero-one d-way ($d \geq 2$) contingency tables under the no d-way interaction model, i.e., with fixed $d-1$ marginal sums. Then, we show some simulation results with our software (available in Appendix). Lastly, we

give some results based on a real dataset - Samson's monks data.

2.1 No three-way interaction model

Let $\mathbf{X} = (X_{ijk})$ of size (m, n, l), where $m, n, l \in \mathbb{N}$ and $\mathbb{N} = \{1, 2, \ldots, \}$, be a table of counts whose entries are independent Poisson random variables with expected frequencies $\{\mu_{ijk}\}$. Consider the generalized linear model,

$$log\mu_{ijk} = \lambda + \lambda_i^M + \lambda_j^N + \lambda_k^L + \lambda_{ij}^{MN} + \lambda_{ik}^{ML} + \lambda_{jk}^{NL} \tag{2.1.1}$$

for $i = 1, \ldots, m$, $j = 1, \ldots, n$, and $k = 1, \ldots, l$ where M, N, and L denote the nominal-scale factors. This model is called the **no three-way interaction model**.

Recall the definition of log-linear models for contingency tables (see Section 1.1.1), we should realize that the no three-way interaction model is a log-linear model:

- the sequence of constants $\mathbf{h}$ is a zero vector;
- the vector of parameters $\boldsymbol{\lambda} = (\lambda, \lambda_1^M, \ldots, \lambda_m^M, \lambda_1^N, \ldots, \lambda_{11}^{NL}, \ldots, \lambda_{nl}^{NL})$;
- the elements in matrix $\mathcal{A}$ can be figured out using Equation (2.1.1).

Define the **two-way marginals** as:

$$\begin{aligned} X_{+jk} &:= \textstyle\sum_{i=1}^{m} X_{ijk}, \ (j = 1, 2, \ldots, n, \ k = 1, 2, \ldots, l), \\ X_{i+k} &:= \textstyle\sum_{j=1}^{n} X_{ijk}, \ (i = 1, 2, \ldots, m, \ k = 1, 2, \ldots, l), \\ X_{ij+} &:= \textstyle\sum_{k=1}^{l} X_{ijk}, \ (i = 1, 2, \ldots, m, \ j = 1, 2, \ldots, n), \end{aligned} \tag{2.1.2}$$

then it is obvious that the one-way marginals and the total count can be written as linear combination of these two-way marginals:

$$X_{i++} := \sum_{j=1}^{n}\sum_{k=1}^{l} X_{ijk} = \sum_{j=1}^{n} X_{ij+} = \sum_{k=1}^{l} X_{i+k}, \ (i = 1, 2, \ldots, m),$$

$$X_{+j+} := \sum_{i=1}^{m}\sum_{k=1}^{l} X_{ijk} = \sum_{i=1}^{m} X_{ij+} = \sum_{k=1}^{l} X_{+jk}, \ (j = 1, 2, \ldots, n),$$

$$X_{++k} := \sum_{i=1}^{m}\sum_{j=1}^{n} X_{ijk} = \sum_{i=1}^{m} X_{i+k} = \sum_{j=1}^{n} X_{+jk},\ (k = 1, 2, \ldots, l),$$

$$X_{+++} := \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{l} X_{ijk} = \sum_{i=1}^{m}\sum_{j=1}^{n} X_{ij+} = \sum_{i=1}^{m}\sum_{k=1}^{l} X_{i+k} = \sum_{j=1}^{n}\sum_{k=1}^{l} X_{+jk}.$$

It is known that a choice of sufficient statistics of the no three-way interaction model are $\mathcal{A}\mathbf{X}$ (see Section 1.1.1), where it is straightforward to figure out that

$$\mathcal{A}\mathbf{X} = (X_{+++}, X_{1++}, \ldots, X_{m++}, X_{+1+}, \ldots, X_{+11}, \ldots, X_{+nl}).$$

As explained in Section 1.1.1, another choice of sufficient statistics for this model is: $A\mathbf{X} = (X_{11+}, \ldots, X_{mn+}, X_{1+1}, \ldots, X_{m+l}, X_{+11}, \ldots, X_{+nl})$, i.e. all two-way marginals.

In this dissertation we are going to focus on zero-one contingency tables. Thus we add additional constraints $X_{ijk} \in \{0, 1\}$, which give us $P(X_{ijk} = 1) = \frac{\mu_{ijk}}{1+\mu_{ijk}}$ and $P(X_{ijk} = 0) = \frac{1}{1+\mu_{ijk}}$, for $i = 1, \ldots, m$, $j = 1, \ldots, n$, and $k = 1, \ldots, l$, and therefore the probability of the whole table is:

$$P(\mathbf{X} \mid X_{ijk} \in \{0, 1\}, \mu_{ijk}, \forall\ i, j, k) = \prod_{i=1}^{m}\prod_{j=1}^{n}\prod_{k=1}^{l} \left(\frac{\mu_{ijk}}{1+\mu_{ijk}}\right)^{X_{ijk}} \left(\frac{1}{1+\mu_{ijk}}\right)^{1-X_{ijk}}.$$

We have showed in Section 1.1.1 that the conditional distribution of the table given the two-way marginals does not depend on the parameters, i.e. $P(\mathbf{X} \mid A\mathbf{X} = b, X_{ijk} \in \{0, 1\}, \mu_{ijk}, \forall\ i, j, k) = P(\mathbf{X} \mid A\mathbf{X} = b, X_{ijk} \in \{0, 1\}, \forall\ i, j, k)$, and we should also notice that with the zero-one constraints, the conditional likelihood function in Equation 1.1.4 becomes:

$$\begin{aligned}
\mathcal{L}_{\mathcal{A},\mathbf{h}}(\boldsymbol{\lambda} \mid \mathbf{X} \in \{0, 1\}^t, \mathcal{A}\mathbf{X} = \mathcal{b}) &= \frac{n_x! \prod_{j=1}^{t} (e^0)^{x_j}}{\sum_{\mathbf{Y}=(y_1,\ldots,y_t)\in\{0,1\}^t,\ \mathcal{A}\mathbf{Y}=\mathcal{b}} \frac{n_y!}{y_1!\cdots y_t!} \prod_{j=1}^{t} (e^0)^{y_j}} \\
&= \frac{n_x!}{\sum_{\mathbf{Y}=(y_1,\ldots,y_t)\in\{0,1\}^t,\ \mathcal{A}\mathbf{Y}=\mathcal{b}} n_y!} \\
&\propto n_x!,
\end{aligned}$$

which implies that it degenerates from the hypergeometric distribution to the uniform distribution.

The no three-way interaction model is particularly important because if we are able to count or estimate the number of tables under this model then this is equivalent to estimating the number of lattice points in any polytope [16, Theorem 1.1]. This means that if we can estimate the number of three-way zero-one tables under this model, then we can estimate the number of any zero-one tables with linear constraints by using De Loera and Onn's bijection mapping.

2.2 Sampling three-way zero-one tables with two-way marginal sums

We need to define notation for the three-way contingency tables. We call the two-way table $X_{\cdot\cdot k}$ with dimension $m \times n$ the kth layer of $\mathbf{X}$. We say the column of entries for the marginal $X_{i_0 j_0 +}$ of $\mathbf{X}$ is the (i_0, j_0)th column of $\mathbf{X}$ (equivalently we say (i_0, k_0)th column for the marginal $X_{i_0 + k_0}$ and (j_0, k_0)th column for the marginal $X_{+ j_0 k_0}$). Consider the (i_0, j_0)th column of the table $\mathbf{X}$ for some $i_0 \in \{1, \ldots, m\}$, $j_0 \in \{1, \ldots, n\}$ with the marginal $l_0 = X_{i_0 j_0 +}$. Also we let the other two marginal sums to be $r_k = X_{i_0 + k}$ and $c_k = X_{+ j_0 k}$. We intent to generate the (i_0, j_0)th column via CP distribution using formula (1.1.7) (see Section 1.1.2). In this formula, the weights $w_k = p_k/(1-p_k)$, $k = 1, 2, \ldots, l$, for each cell in this column should be decided by both r_k and c_k. To sample a zero-one three-way table $\mathbf{X}$ with given two-way marginals X_{ij+}, X_{i+k}, and X_{+jk} for $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, and $k = 1, 2, \ldots, l$, we sample the (i_0, j_0)th column of $\mathbf{X}$ for each $i_0 \in \{1, \ldots, m\}$, $j_0 \in \{1, \ldots, n\}$. Next, we are going to show that we should take:

$$p_k := \frac{r_k \cdot c_k}{r_k \cdot c_k + (n - r_k)(m - c_k)} \tag{2.2.1}$$

and thus

$$w_k = \frac{r_k \cdot c_k}{(n - r_k)(m - c_k)}. \tag{2.2.2}$$

Remark 2.2.1. *In the theorem below, we assume that we do not have the trivial cases, namely,* $1 \leq r_k \leq n-1$ *and* $1 \leq c_k \leq m-1$. *We will discuss the alternative cases later.*

Theorem 2.2.2. *For the uniform distribution over all* $m \times n \times l$ *zero-one tables with given marginals* $r_k = X_{i_0+k}$, $c_k = X_{+j_0k}$ *for* $k = 1, 2, \ldots, l$, *and a fixed marginal for the factor* L, l_0, *the marginal distribution of the fixed marginal* l_0 *is the same as the conditional distribution of* Z *defined by* (1.1.7) *given* $S_Z = l_0$ *with*

$$p_k := \frac{r_k \cdot c_k}{r_k \cdot c_k + (n - r_k)(m - c_k)}.$$

Proof. We start by giving an algorithm for generating tables uniformly from all $m \times n \times l$ zero-one tables with given marginals r_k, c_k for $k = 1, 2, \ldots, l$, and a fixed marginal for the factor L, l_0.

1. For $k = 1, \ldots, l$ consider the kth layer of $\mathbf{X}$, they are $m \times n$ tables. We randomly choose r_k positions in the (i_0, k)th column and c_k positions in the (j_0, k)th column, and put 1s in those positions. The choices of positions are independent across different layers.

2. Accept those tables with given column sum l_0.

It is easy to see that tables generated by this algorithm are uniformly distributed over all $m \times n \times l$ zero-one tables with given marginals r_k, c_k for $k = 1, 2, \ldots, l$, and a fixed marginal for the factor L, l_0 for the (i_0, j_0)th column of the table $\mathbf{X}$. We can derive the marginal distribution of the (i_0, j_0)th column of $\mathbf{X}$ based on this algorithm. At Step 1, we choose the cell at position $(i_0,\ j_0,\ 1)$ to put 1 in with the probability:

$$\frac{\binom{n-1}{r_1-1}\binom{m-1}{c_1-1}}{\binom{n-1}{r_1-1}\binom{m-1}{c_1-1} + \binom{n-1}{r_1}\binom{m-1}{c_1}} = \frac{r_1 \cdot c_1}{r_1 \cdot c_1 + (n - r_1)(m - c_1)}.$$

Because the choices of positions are independent across different layers, after Step 1 the marginal distribution of the (i_0, j_0)th column is the same as the distribution of Z

defined by (1.1.7) with

$$p_k = \frac{\binom{n-1}{r_k-1}\binom{m-1}{c_k-1}}{\binom{n-1}{r_k-1}\binom{m-1}{c_k-1} + \binom{n-1}{r_k}\binom{m-1}{c_k}} = \frac{r_k \cdot c_k}{r_k \cdot c_k + (n-r_k)(m-c_k)}.$$

Step 2 rejects the tables whose (i_0, j_0)th column sum is not l_0. This implies that after Step 2, the marginal distribution of the (i_0, j_0)th column is the same as the conditional distribution of Z defined by (1.1.7) with

$$p_k = \frac{r_k \cdot c_k}{r_k \cdot c_k + (n-r_k)(m-c_k)}.$$

□

Remark 2.2.3. *The sequential importance sampling via CP for sampling a two-way zero-one table defined in [9] is a special case of our SIS procedure. We can induce* p_k *defined in* (2.2.1) *and the weights defined in* (2.2.2) *to the weights for two-way zero-one contingency tables defined in [9]. Note that when we consider two-way zero-one contingency tables we have* $c_k = 1$ *for all* $k = 1, \ldots, l$ *and for all* $j_0 = 1, \ldots, n$ *(or* $r_k = 1$ *for all* $k = 1, \ldots, l$ *and for all* $i_0 = 1, \ldots, m$*), and* $m = 2$ *(or* $n = 2$*, respectively). Therefore when we consider the two-way zero-one tables we get*

$$p_k = \frac{r_k}{n},\ w_k = \frac{r_k}{n - r_k},$$

or respectively

$$p_k = \frac{c_k}{m},\ w_k = \frac{c_k}{m - c_k}.$$

We still need to extend Theorem 2.2.2 to deal with structural zeros. The reason is that even though no structural zero is assigned by users in the original table, during the intermediary steps of our SIS procedure via CP distribution on a three-way zero-one table, there will be some columns for the L factor with trivial cases. In that case we have to treat them as structural zeros in the kth layer for some $k \in \{1, \ldots, l\}$. We can use a strategy similar with the one in [8]. In the next theorem we are going to show that the probabilities for the distribution in (1.1.7) become as following:

$$p_k := \frac{r_k \cdot c_k}{r_k \cdot c_k + (n - r_k - g_k^{r_0})(m - c_k - g_k^{c_0})}, \tag{2.2.3}$$

where $g_k^{r_0}$ is the number of structural zeros in the (r_0, k)th column and $g_k^{c_0}$ is the number of structural zeros in the (c_0, k)th column. Thus we have weights:

$$w_k = \frac{r_k \cdot c_k}{(n - r_k - g_k^{r_0})(m - c_k - g_k^{c_0})}. \tag{2.2.4}$$

Theorem 2.2.4. *For the uniform distribution over all $m \times n \times l$ zero-one tables with structural zeros with given marginals $r_k = X_{i_0+k}$, $c_k = X_{+j_0k}$ for $k = 1, 2, \ldots, l$, and a fixed marginal for the factor L, l_0, the marginal distribution of the fixed marginal l_0 is the same as the conditional distribution of Z defined by* (1.1.7) *given $S_Z = l_0$ with*

$$p_k := \frac{r_k \cdot c_k}{r_k \cdot c_k + (n - r_k - g_k^{r_0})(m - c_k - g_k^{c_0})},$$

where $g_k^{r_0}$ is the number of structural zeros in the (r_0, k)th column and $g_k^{c_0}$ is the number of structural zeros in the (c_0, k)th column.

Proof. The proof is similar to the proof for Theorem 2.2.2, just replace the probability p_k with

$$p_k = \frac{\binom{n-1-g_k^{r_0}}{r_k-1}\binom{m-1-g_k^{c_0}}{c_k-1}}{\binom{n-1-g_k^{r_0}}{r_k-1}\binom{m-1-g_k^{c_0}}{c_k-1} + \binom{n-1-g_k^{r_0}}{r_k}\binom{m-1-g_k^{c_0}}{c_k}} = \frac{r_k \cdot c_k}{r_k \cdot c_k + (n - r_k - g_k^{r_0})(m - c_k - g_k^{c_0})}.$$

□

Remark 2.2.5. *The sequential importance sampling via CP for sampling a two-way zero-one table with structural zeros defined in Theorem 1 in [8] is a special case of our SIS. We can induce p_k defined in* (2.2.3) *and the weights defined in* (2.2.4) *to the weights for two-way zero-one contingency tables defined in [8]. Note that when we consider two-way zero-one contingency tables we have $c_k = 1$ for all $k = 1, \ldots, l$ and for all $j_0 = 1, \ldots, n$ (or $r_k = 1$ for all $k = 1, \ldots, l$ and for all $i_0 = 1, \ldots, m$), $m = 2$ (or $n = 2$, respectively), and $g_k^{c_0} = 0$ (or $g_k^{r_0}$, respectively). Therefore when we consider the two-way zero-one tables we get*

$$p_k = \frac{r_k}{n - g_k^{r_0}},\ w_k = \frac{r_k}{n - r_k - g_k^{r_0}},$$

or respectively

$$p_k = \frac{c_k}{m - g_k^{c_0}},\ w_k = \frac{c_k}{m - c_k - g_k^{c_0}}.$$

To end this section, we are going to give some algorithms of how to implement this method to sample a zero-one three-way table with fixed two-way marginals. The code is attached in Appendix. Notice that to modify our software in order to make it available for users to set up structures in the original table, one only needs to change the initial value of the table that stores all structures appearing during the intermediary steps and renewed during sampling. For the convenience of stating these algorithms, we say that the direction of (j_0, k_0)th column is the direction I, the direction of (i_0, k_0)th column is the direction J, and the direction of (i_0, j_0)th column is the direction K. Please look at Figure 2.1 to get a more intuitive view.

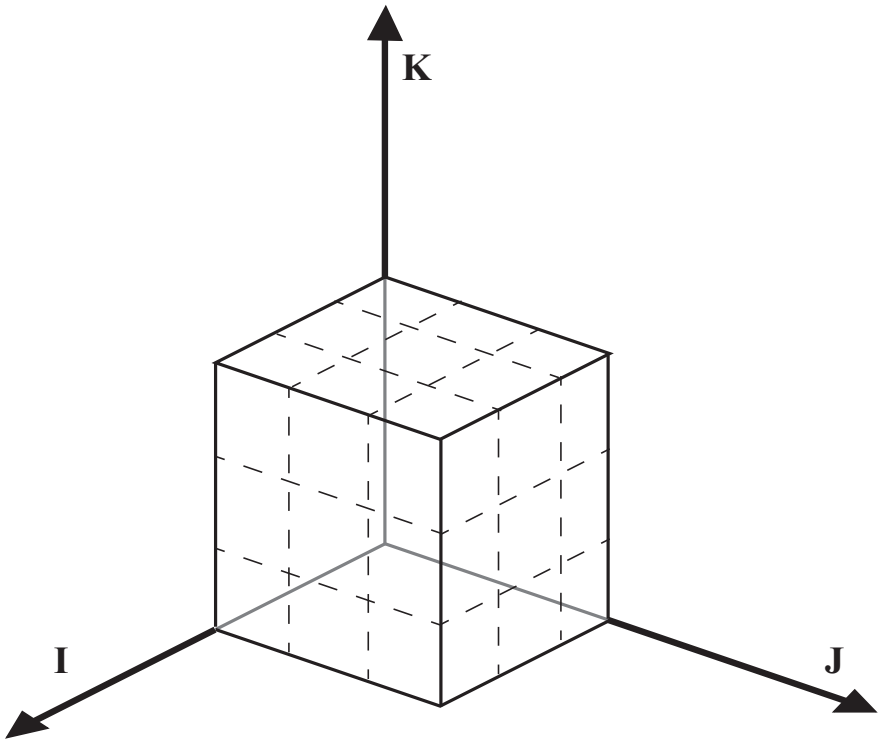


Figure 2.1: An example of a $3 \times 3 \times 3$ table.

Algorithm 2.2.6 (Store structures in the zero-one table)**.** *This algorithm stores the structures, including structural* 0*'s and structural* 1*'s, in the observed table* $\mathbf{x_0}$*. The output will be used to avoid trivial cases in sampling. The output* A *and* B *matrices both have the same dimension with* $\mathbf{x_0}$*. A cell in* A *will takes value* 1 *if the position is either structural zero or structural one, and* 0 *if neither. The matrix* B *is defined similarly with* A *but a cell takes value one only if the position is structural one. By converting structural* 1*'s to structural* 0*'s, we only need to consider sampling a table without structural* 1*'s, that is, a table with new marginal sums:* $X^*_{ij+} = X_{ij+} - \sum_{k=1}^{l} B_{ijk} = X_{ij+} - B_{ij+}$*,* $X^*_{i+k} = X_{i+k} - \sum_{j=1}^{n} B_{ijk} = X_{i+k} - B_{i+k}$*, and* $X^*_{+jk} = X_{+jk} - \sum_{i=1}^{m} B_{ijk} = X_{+jk} - B_{+jk}$ *for* $i = 1, 2, \ldots, m$*,* $j = 1, 2, \ldots, n$*, and* $k = 1, 2, \ldots, l$*.*

Input *The observed marginals* X_{ij+}*,* X_{i+k}*, and* X_{+jk} *for* $i = 1, 2, \ldots, m$*,* $j = 1, 2, \ldots, n$*, and* $k = 1, 2, \ldots, l$*.*

Output *Matrix* A *and* B*, new marginal sums* X^*_{ij+}*,* X^*_{i+k}*, and* X^*_{+jk} *for* $i = 1, 2, \ldots, m$*,* $j = 1, 2, \ldots, n$*, and* $k = 1, 2, \ldots, l$*.*

Algorithm

1. *Check all marginals in direction I. For* $i = 1, 2, \ldots, m$*:*

   If $X_{+jk} = 0$*,* $A_{i'jk} = 1$*, for all* $i' = 1, 2, \ldots, m$ *and* $A_{i'jk} = 0$*;*

   If $X_{+jk} = 1$*,* $A_{i'jk} = 1$ *and* $B_{i'jk} = 1$*, for all* $i' = 1, 2, \ldots, m$ *and* $A_{i'jk} = 0$*.*

2. *Check all marginals in direction J. For* $j = 1, 2, \ldots, n$*:*

   If $X_{i+k} = 0$*,* $A_{ij'k} = 1$*, for all* $j' = 1, 2, \ldots, n$ *and* $A_{ij'k} = 0$*;*

   If $X_{i+k} = 1$*,* $A_{ij'k} = 1$ *and* $B_{ij'k} = 1$*, for all* $j' = 1, 2, \ldots, n$ *and* $A_{ij'k} = 0$*.*

3. *Check all marginals in direction K. For* $k = 1, 2, \ldots, l$*:*

   If $X_{ij+} = 0$*,* $A_{ijk'} = 1$*, for all* $k' = 1, 2, \ldots, l$ *and* $A_{ijk'} = 0$*;*

   If $X_{ij+} = 1$*,* $A_{ijk'} = 1$ *and* $B_{ijk'} = 1$*, for all* $k' = 1, 2, \ldots, l$ *and* $A_{ijk'} = 0$*.*

4. *If any changes made in step (1), (2) or (3), come back to (1), else stop.*

5. *Compute new marginals:*

   $X^*_{ij+} = X_{ij+} - B_{ij+}$, $X^*_{i+k} = X_{i+k} - B_{i+k}$, *and* $X^*_{+jk} = X_{+jk} - B_{+jk}$ *for* $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, *and* $k = 1, 2, \ldots, l$.

Algorithm 2.2.7 (Generate a two-way table with given marginals)**.** *This algorithm is used to generate a slice (fixed i, the two-way table $X_{i..}$ with dimension $n \times l$ is called a* ***slice****) of the three-way table. The probability of the sampled slice will be computed, too.*

Input *Row sums r^*_j and column sums c^*_k, $j = 1, 2, \ldots, n$, and $k = 1, 2, \ldots, l$; structures A; marginal sums on direction I: X_{+jk} for $i = 1, 2, \ldots, m$.*

Output *A sampled table and its probability. Return* 0 *if the process fails.*

Algorithm

1. *Order all columns so that the column sums decreases.*

2. *Generate the column (along the direction K) with the largest sum, the weights used in CP distribution are computed by Equation* (2.2.4)*. Notice that each k relates to a specific cell in the column, r_k and c_k are the corresponding row sums in the direction J and I, respectively. $g_k^{r_0}$ and $g_k^{c_0}$ are the number of structures in the rows of the direction J and I, respectively. The probability of the generated column will be returned if the process succeeds, while* 0 *will be returned in this step if such a column does not exist.*

3. *Delete the generated column in step 2, and for the remaining subtable, do the following:*

   a) *If only one column is left, fill it with the corresponding marginals and go to step 4.*

b) If a) is not true, check all marginals to see if step 2 causes any new structures. We will be able to avoid trivial cases by doing this. Go back to step 1 with updated marginals and structures.

4. Return generated matrix and its CP probability. This matrix will be the corresponding slice in the three-way table. If the process fails, return 0.

Algorithm 2.2.8 (SIS with CP distribution for sampling a three-way zero-one table)**.** *We describe an algorithm to sample a three-way zero-one table* $\mathbf{X}$ *with given marginals* X_{ij+}, X_{i+k}, *and* X_{+jk} *for* $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, *and* $k = 1, 2, \ldots, l$ *via SIS with CP distribution.*

Input *The observed table* $\mathbf{x_0}$.

Output *The sampled table* $\mathbf{X}$.

Algorithm

1. Compute the two-way marginals for $\mathbf{X}$*:* X_{ij+}, X_{i+k}, *and* X_{+jk} *for* $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, *and* $k = 1, 2, \ldots, l$.

2. Run Algorithm 2.2.6 with the marginal sums computed in step 1. We will get the tables A, which stores the positions of all structures, and B, which only stores the positions of all structural 1*'s. It will also output the revised two-way marginals that we will use in the following steps.*

3. To sample a zero-one three-way table with the marginal sums obtained in step 2, do SIS:

a) Delete the slices in direction I ($X_{i..}$, $i = 1, \ldots, m$)*, the slices in direction J* ($X_{.j.}$, $j = 1, \ldots, n$)*, and the layers in direction K* ($X_{..k}$, $k = 1, \ldots, l$, $k = 1, \ldots, l$) *if they are completely filled by structures (i.e. they are completely fixed); consider the left-over subtable.*

b) Summing up the cells within all slices in direction I. Rearrange the slices by ordering their sums from the largest to the smallest.

c) Consider the slice in direction I with the largest sum and the positions for structural zeros in this slice, where these positions are stored in table A from Algorithm 2.2.7. Generate a sample for this slice and compute its probability. The algorithm will return 0 *if the sampling fails.*

d) Delete the generated slice in c), and for the remaining subtable, do the following:

i. if only one layer left, then fill it with the corresponding marginals and go to e);

ii. else, go back to step 2 with updated marginal sums.

e) Add the sampled three-way table with table B to retrieve the structural 1*'s.*

4. Return the table in e) and its probability, i.e. the same probability with the sampled table in d). Return 0 *if failed.*

2.3 Sampling d-way ($d \geq 2$) zero-one tables with $(d-1)$-way marginals

In this section we extend our results further to zero-one contingency tables under the no d-way ($d \in \mathbb{N}$ and $d > 3$) interaction model, i.e., with fixed $(d-1)$-way marginals. Let $\mathbf{X} = (X_{i_1 \ldots i_d})$ be a zero-one contingency table of size $(n_1 \times \cdots \times n_d)$, where $n_i \in \mathbb{N}$ for $i = 1, \ldots, d$. The sufficient statistics under the no d-way interaction model are

$$\begin{aligned} &X_{+i_2 \ldots i_d},\ X_{i_1+i_3 \ldots i_d},\ \ldots, X_{i_1 \ldots i_{d-1}+}, \\ &\text{for } i_1 = 1, \ldots, n_1,\ i_2 = 1, \ldots, n_2, \ldots, i_d = 1, \ldots, n_d, \end{aligned} \tag{2.3.1}$$

which are called the $(d-1)$**-way marginals**.

For each $i_1^0 \in \{1, \ldots, n_1\}, \ldots, i_{d-1}^0 \in \{1, \ldots, n_d\}$, we say the column of the entries for a marginal sum $X_{i_1 \ldots i_{j-1} + i_{j+1} \ldots i_d}$ the $(i_0, \ldots, i_{j-1}, i_{j+1}, \ldots, i_d)$th column of $\mathbf{X}$. For

each $i_1^0 \in \{1,\ldots,n_1\},\ldots,i_{d-1}^0 \in \{1,\ldots,n_{d-1}\}$, we consider the $(i_1^0,\ldots,i_{d-1}^0)$th column for the dth factor. Let $l_0 = X_{i_1^0,\ldots,i_{d-1}^0+}$. Let $r_k^j = X_{i_1^0\ldots i_{j-1}^0+i_{j+1}^0\ldots i_{d-1}^0 k}$ for fixed $k \in \{1,\ldots,n_d\}$. We are going to show the theorem that for sampling a zero-one d-way contingency table $\mathbf{X}$, the probabilities we should use in formula (1.1.7) are:

$$p_k := \frac{\prod_{j=1}^{d-1} r_k^j}{\prod_{j=1}^{d-1} r_k^j + \prod_{j=1}^{d-1}(n_j - r_k^j)}. \tag{2.3.2}$$

Remark 2.3.1. *We assume that we do not have trivial cases, namely,* $1 \le r_k^j \le n_j - 1$ *for* $k = 1,\ldots,n_j$ *and* $j = 1,\ldots,d$.

Theorem 2.3.2. *For the uniform distribution over all d-way zero-one contingency tables* $\mathbf{X} = (X_{i_1\ldots i_d})$ *of size* $(n_1 \times \cdots \times n_d)$, *where* $n_i \in \mathbb{N}$ *for* $i = 1,\ldots,d$ *with marginals* $l_0 = X_{i_1^0,\ldots,i_{d-1}^0+}$, *and* $r_k^j = X_{i_1^0\ldots i_{j-1}^0+i_{j+1}^0\ldots i_{d-1}^0 k}$ *for* $k \in \{1,\ldots,n_d\}$, *the marginal distribution of the fixed marginal* l_0 *is the same as the conditional distribution of* Z *defined by* (1.1.7) *given* $S_Z = l_0$ *with*

$$p_k := \frac{\prod_{j=1}^{d-1} r_k^j}{\prod_{j=1}^{d-1} r_k^j + \prod_{j=1}^{d-1}(n_j - r_k^j)}.$$

Proof. The proof is similar to the proof for Theorem 2.2.2, we just extend the same argument to a d-way zero-one table under the no d-way interaction model with the probability

$$p_k = \frac{\prod_{j=1}^{d-1} \binom{n_j-1}{r_k^j-1}}{\prod_{j=1}^{d-1} \binom{n_j-1}{r_k^j-1} + \prod_{j=1}^{d-1} \binom{n_j-1}{r_k^j}} = \frac{\prod_{j=1}^{d-1} r_k^j}{\prod_{j=1}^{d-1} r_k^j + \prod_{j=1}^{d-1}(n_j - r_k^j)}.$$

□

Similarly with the three-way case we have discussed before, even if no structural zero is assigned by user in the original table, during the intermediary steps of our SIS procedure via CP on a three-way zero-one table there will be some columns for the dth factor with trivial cases. In that case we have to treat them as structural zeros in the kth layer for some $k \in \{1,\ldots,l\}$. In the next theorem we are going to show

that the probabilities for the distribution in (1.1.7) become as follows:

$$p_k := \frac{\prod_{j=1}^{d-1} r_k^j}{\prod_{j=1}^{d-1} r_k^j + \prod_{j=1}^{d-1}(n_j - r_k^j - g_k^j)}. \tag{2.3.3}$$

where g_k^j is the number of structural zeros in the $(i_1^0, \ldots, i_{j-1}^0, i_{j+1}^0, \ldots, i_{d-1}^0, k)$th column of $\mathbf{X}$. Thus we have weights:

$$w_k = \frac{\prod_{j=1}^{d-1} r_k^j}{\prod_{j=1}^{d-1}(n_j - r_k^j - g_k^j)}. \tag{2.3.4}$$

Theorem 2.3.3. *For the uniform distribution over all d-way zero-one contingency tables* $\mathbf{X} = (X_{i_1 \ldots i_d})$ *of size* $(n_1 \times \cdots \times n_d)$*, where* $n_i \in \mathbb{N}$ *for* $i = 1, \ldots, d$ *with marginals* $l_0 = X_{i_1^0, \ldots, i_{d-1}^0 +}$*, and* $r_k^j = X_{i_1^0 \ldots i_{j-1}^0 + i_{j+1}^0 \ldots i_{d-1}^0 k}$ *for* $k \in \{1, \ldots, n_d\}$*, the marginal distribution of the fixed marginal* l_0 *is the same as the conditional distribution of* Z *defined by* (1.1.7) *given* $S_Z = l_0$ *with*

$$p_k := \frac{\prod_{j=1}^{d-1} r_k^j}{\prod_{j=1}^{d-1} r_k^j + \prod_{j=1}^{d-1}(n_j - r_k^j - g_k^j)}$$

where g_k^j *is the number of structural zeros in the* $(i_1^0, \ldots, i_{j-1}^0, i_{j+1}^0, \ldots, i_{d-1}^0 k)$*th column of* $\mathbf{X}$*.*

Proof. The proof is similar to the proof for Theorem 2.2.4, we just extend the same argument to a d-way zero-one table under the no d-way interaction model with the probability

$$p_k = \frac{\prod_{j=1}^{d-1} \binom{n_j - 1 - g_k^j}{r_k^j - 1}}{\prod_{j=1}^{d-1} \binom{n_j - 1 - g_k^j}{r_k^j - 1} + \prod_{j=1}^{d-1} \binom{n_j - 1 - g_k^j}{r_k^j}} = \frac{\prod_{j=1}^{d-1} r_k^j}{\prod_{j=1}^{d-1} r_k^j + \prod_{j=1}^{d-1}(n_j - r_k^j - g_k^j)}.$$

□

2.4 Computational examples of counting the total number of three-way tables with fixed two-way marginals

For our simulation study we use the software package **R** [56] in programming and use SIS procedure to estimate the total number of zero-one three-way tables with

fixed two-way marginals via sampling tables uniformly over Σ. The code can be found in Appendix. To compare with our estimators of numbers of tables, we count the exact numbers of tables via the software **LattE** [17] for small examples in this section (Examples (2.4.2) to (2.4.13)). When the contingency tables are large and/or the models are complicated, it is very difficult to obtain the exact number of tables. Thus we need a good measurement of accuracy for the estimated number of tables. In [9], they used the coefficient of variation (cv^2):

$$cv^2 = \frac{var_q\{p(\mathbf{X})/q(\mathbf{X})\}}{\mathbb{E}_q^2\{p(\mathbf{X})/q(\mathbf{X})\}}$$

which is equal to $var_q\{1/q(\mathbf{X})\}/\mathbb{E}_q^2\{1/q(\mathbf{X})\}$ for the problem of estimating the number of tables because the true distribution $p(\mathbf{X})$ is assumed to be the uniform distribution over Σ. The value of cv^2 is simply the chi-square distance between the two distributions p and q, which means the smaller it is, the closer the two distributions are. In [9] they estimated cv^2 by:

$$cv^2 \approx \frac{\sum_{i=1}^{\mathfrak{N}}\{1/q(\mathbf{X_i}) - \left[\sum_{j=1}^{\mathfrak{N}} 1/q(\mathbf{X_j})\right]/\mathfrak{N}\}^2/(\mathfrak{N}-1)}{\left\{\left[\sum_{j=1}^{\mathfrak{N}} 1/q(\mathbf{X_j})\right]/\mathfrak{N}\right\}^2},$$

where $\mathbf{X_1}, \ldots, \mathbf{X}_{\mathfrak{N}}$ are tables drawn iid from $q(\mathbf{X})$. When we have rejections, we compute the variance using only accepted tables. In this section and Section 4.1.1 we will also investigate relations of estimated number of tables with the exact numbers of tables and cv^2 when we have rejections.

In this section, we name the two-way marginals as following to avoid confusing: suppose we have an observed table $\mathbf{X} = (X_{ijk})_{m\times n\times l}$, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, and $k = 1, 2, \ldots, l$, then the two-way marginals can be computed using equations (2.1.2), i.e. by suming along the direction I, J and K, we are able to get the following three matrices: $si = (X_{+jk})_{n\times l}$, $sj = (X_{i+k})_{m\times l}$, and $sk = (X_{ij+})_{m\times n}$.

An interesting problem in mathematics is counting the number of semimagic cubes. In our examples we are going to estimate the number of **3-dimensional**

semimagic cubes, which are defined as $m \times n \times l$ contingency tables such that $m = n = l$ and all two-way marginals are equal, i.e. there exists a constant such that s, $1 \leq s \leq m-1$ and $X_{+jk} = X_{i+k} = X_{ij+} = s$, for $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, and $k = 1, 2, \ldots, l$.

Example 2.4.1 (The 3-dimensional Semimagic Cube)**.** *Suppose si, sj, and sk are all* 3×3 *matrices such that all cells of them take value* 1*, i.e.*

$$si = sj = sk = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} .$$

The exact number of tables is 12*. We took* 114.7 *seconds to run* $10,000$ *samples in the SIS procedure. The estimator was* 12*, and the acceptance rate was* 100%*. In fact, we realized that if the acceptance rate is* 100%*, then we can obtain a good estimation even when with a smaller sample size.*

We used **R** to produce more examples. Examples (2.4.2) to (2.4.13) are constructed by the same code but with different seeds, this means that the entries in the input matrix were generated by a pseudorandom number generator initialized by different numbers. The purpose of the usage of seeds is that we can regenerate the same pseudorandom zero-one tables repeatedly using the same seed so that our results can be tested and verified. **R** package "Rlab" is needed in the following code to use function "rbern".

```
seed = 6;
m = 3; n = 3; l = 4; prob = 0.8;
N = 1000; k = 200  # N: the sample size for SIS, i.e. \mathfrak{N}
set.seed(seed)
A = array( rbern(m*n*l, prob), c(m, n, l) )
outinfo = tabinfo(A)   # compute the two-way marginals
```

```
numtable(N, outinfo, k)   # estimate the total number of tables
```

The above code gives an example of how to produce an input table and estimate the total number of tables which have the same two-way marginals with the input table. In this specific example, table `A` is the input table with dimension $3 \times 3 \times 4$ and its entries are i.i.d. Bernoulli trials with success probability `prob = 0.8`. The number `N` is the sample size, i.e. the total number of tables we sample, including those that are rejected in the process. The number `k` is a parameter to control the printing of output. The functions `outinfo` and `numtable` can be found in Appendix. Notice that we can generate different examples simply by changing the setting of these parameters. For those examples in which the real number of tables cannot be computed by **LattE**, cv^2 will be used to measure how accurate our estimator is, it is defined as $\frac{Var}{Mean^2}$ and is introduced earlier in this section.

Example 2.4.2 (seed=6; m=3; n=3; l=4; prob=0.8)**.** *Suppose si, sj, and sk are as following, respectively:*

$$\begin{array}{|c|c|c|c|}\hline 2&2&2&2\\\hline 1&3&2&2\\\hline 2&3&3&2\\\hline\end{array}\,,\quad \begin{array}{|c|c|c|c|}\hline 2&3&2&2\\\hline 1&3&3&3\\\hline 2&2&2&1\\\hline\end{array}\,,\quad \begin{array}{|c|c|c|}\hline 3&3&3\\\hline 3&3&4\\\hline 2&2&3\\\hline\end{array}\,.$$

The real number of tables is 3*. The sample size was 1000 and the estimator was* 3.00762 *with* $cv^2 = 0.0708$*. The whole process took* 13.216 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.3 (seed=60; m=3; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are as following, respectively:*

$$\begin{array}{|c|c|c|c|}\hline 2&2&2&1\\\hline 1&1&1&0\\\hline 1&1&1&2\\\hline 1&1&2&3\\\hline\end{array}\,,\quad \begin{array}{|c|c|c|c|}\hline 3&3&2&1\\\hline 1&0&2&2\\\hline 1&2&2&3\\\hline\end{array}\,,\quad \begin{array}{|c|c|c|c|}\hline 3&2&2&2\\\hline 1&0&2&2\\\hline 3&1&1&3\\\hline\end{array}\,.$$

The real number of tables is 5. *The sample size was 1000 and the estimator was* 4.991026 *with* $cv^2 = 0.1335$. *The whole process took* 17.016 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.4 (seed=61; m=3; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are as following, respectively:*

1	2	2	1
0	1	1	2
1	0	2	1
0	1	3	2

,

1	2	3	2
1	1	2	3
0	1	3	1

,

3	1	1	3
1	2	2	2
2	1	1	1

.

The real number of tables is 8. *The sample size was 1000 and the estimator was 8.04964 with* $cv^2 = 0.2389$. *The whole process took* 16.446 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.5 (seed=240; m=4; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are as following, respectively:*

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

,

2	2	4	1
3	2	2	2
2	3	3	1
1	3	1	1

,

2	2	3	2
3	2	1	3
3	2	2	2
2	1	0	3

.

The real number of tables is 8. *The sample size was 1000 and the estimator was 8.039938 with* $cv^2 = 0.2857$. *The whole process took* 23.612 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.6 (seed=1240; m=4; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are*

as following, respectively:

2	3	2	3
1	2	3	2
2	2	3	2
3	2	3	2

,

1	4	1	3
4	2	4	2
1	2	4	3
2	1	2	1

,

2	2	2	3
3	3	3	3
3	2	2	3
2	1	2	1

.

The real number of tables is 28*. The sample size was 1000 and the estimator is* 26.89940 *with* $cv^2 = 1.0306$*. The whole process took* 29.067 *seconds (in* ***R****) with a* 100% *acceptance rate. For sample size 5000 the estimator becomes* 28.0917*, with* $cv^2 = 1.2070$*.*

Example 2.4.7 (seed=2240; m=4; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are as following, respectively:*

1	2	3	1
2	3	2	3
2	4	2	1
2	1	4	1

,

2	3	2	0
3	2	3	2
1	3	3	1
1	2	3	3

,

2	1	2	2
3	2	3	2
1	4	2	1
1	3	2	3

.

The real number of tables is 4*. The sample size was 1000 and the estimator was* 3.98125 *with* $cv^2 = 0.0960$*. The whole process took* 26.96 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.8 (seed=3340; m=4; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are as following, respectively:*

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

,

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

,

3	1	1	1
3	1	2	2
1	2	1	1
3	2	1	3

.

The real number of tables is 2. *The sample size was 1000 and the estimator was* 2 *with* $cv^2 = 0$. *The whole process took* 15.214 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.9 (seed=3440; m=4; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are as following, respectively:*

1	3	1	3
1	1	2	2
2	3	1	0
3	2	2	3

,

2	2	2	2
2	1	2	1
1	3	1	2
2	3	1	3

,

3	1	1	3
1	2	1	2
2	0	3	2
2	3	1	3

.

The real number of tables is 12. *The sample size was 1000 and the estimator was* 12.04838 *with* $cv^2 = 0.7819733$. *The whole process took* 27.074 *seconds (in* ***R****) with a* 85.9% *acceptance rate.*

Example 2.4.10 (seed=5440; m=4; n=4; l=4; prob=0.5)**.** *Suppose si, sj, and sk are as following, respectively:*

2	1	0	1
2	3	1	2
3	1	2	1
1	3	2	2

,

2	3	2	1
2	1	2	3
2	1	0	1
2	3	1	1

,

1	2	2	3
1	1	3	3
1	3	0	0
1	2	2	2

.

The real number of tables is 9. *The sample size was 1000 and the estimator was* 8.882672 *with* $cv^2 = 0.7701368$. *The whole process took* 30.171 *seconds (in* ***R****) with a* 100% *acceptance rate. Another result for the same sample size is: an estimator is* 8.521734, $cv^2 = 0.6695902$. *we can see that the latter has a slightly better* cv^2 *but a slightly worse estimator. We'll discuss more in Section 4.1.1.*

Example 2.4.11 (seed=122; m=4; n=4; l=5; prob=0.2)**.** *Suppose si, sj, and sk are as following, respectively:*

2	0	3	3	2
0	0	1	0	0
1	0	1	1	1
0	1	0	1	0

,

1	0	0	2	1
1	0	2	1	1
1	1	1	1	1
0	0	2	1	0

,

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

.

The real number of tables is 5*. The sample size was 1000 and the estimator was* 4.93625 *with* $cv^2 = 0.2035$*. The whole process took* 21.325 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.12 (seed=222; m=4; n=4; l=5; prob=0.2)**.** *Suppose si, sj, and sk are as following, respectively:*

1	0	1	1	1
2	1	0	1	2
0	1	1	1	0
1	1	1	1	1

,

2	1	0	0	2
1	2	1	2	1
1	0	1	1	1
0	0	1	1	0

,

2	3	0	0
1	3	2	1
0	0	1	3
1	0	0	1

.

The real number of tables is 2*. The sample size was 1000 and the estimator was* 2 *with* $cv^2 = 0$*. The whole process took* 19.064 *seconds (in* ***R****) with a* 100% *acceptance rate.*

Example 2.4.13 (seed=322; m=4; n=4; l=5; prob=0.2)**.** *Suppose si, sj, and sk are as following, respectively:*

1	1	1	1	1
1	1	1	1	1
1	2	0	0	1
2	0	1	1	2

,

0	0	1	1	0
1	0	1	0	1
2	2	0	1	2
2	2	1	1	2

,

0	2	0	0
1	0	0	2
1	3	1	2
3	0	3	2

.

The real number of tables is 5. *The sample size was 1000 and the estimator was* 4.992 *with* $cv^2 = 0.2179682$. *The whole process took* 23.25 *seconds (in* ***R****) with a* 85.2% *acceptance rate.*

Summary 2.4.14 (Summary of the results from Example (2.4.2) to Example (2.4.13))**.** *This is only a summary of main results of those examples in Table 2.1. For all results appear here we set the sample size* $1,000$. *We will discuss these results in Section 4.1.1.*

Table 2.1: Summary of Examples (2.4.2) - (2.4.13)

Dimension	Example	# tables	Estimation	cv^2	Acceptance rate
$3 \times 3 \times 4$	2.4.2	3	3.00762	0.0708	100%
$3 \times 4 \times 4$	2.4.3	5	4.991026	0.1335	100%
	2.4.4	8	8.04964	0.2389	100%
$4 \times 4 \times 4$	2.4.5	8	8.039938	0.2857	100%
	2.4.6	28	26.89940	1.0306	100%
	2.4.7	4	3.98125	0.0960	100%
	2.4.8	2	2	0	100%
	2.4.9	12	12.04838	0.7820	85.9%
	2.4.10	9	8.882672	0.7701	100%
$4 \times 4 \times 5$	2.4.11	5	4.93625	0.2035	100%
	2.4.12	2	2	0	100%
	2.4.13	5	4.992	0.2180	85.2%

Example 2.4.15 (Larger 3-dimensional Semimagic Cubes)**.** *In this example, we consider* $m \times n \times l$ *tables for* $m = n = l = 4, \ldots, 10$ *such that every two-way marginal equals to* 1. *The results are summarized in Table 2.2.*

Example 2.4.16 (Larger 3-dimensional Semimagic Cubes continues)**.** *In this example, we consider* $m \times n \times l$ *tables for* $m = n = l = 4, \ldots, 10$ *such that every two-way*

Table 2.2: A summary of computational results on $m \times m \times m$ semimagic cubes for $m = 4, \ldots, 10$

Dimension m	*# tables*	$\mathfrak{N}$	CPU time (sec)	Estimation	cv^2	Acceptance rate
4	576	1000	32.44	568.944	0.26	100%
		10000	324.18	571.1472	0.27	100%
5	161280	1000	60.39	161603.5	0.18	99%
		10000	605.45	161439.3	0.18	99.2%
6	812851200	1000	102.66	801634023	0.58	98.3%
		10000	1038.46	819177227	0.45	98.8%
7	6.14794e+13	1000	158.55	6.08928e+13	0.60	97%
		10000	1590.84	6.146227e+13	0.64	97.7%
8	1.08776e+20	1000	234.53	1.080208e+20	1.07	95.6%
		10000	2300.91	1.099627e+20	1.00	96.5%
9	5.52475e+27	1000	329.17	5.845308e+27	1.46	94%
		10000	3238.1	5.684428e+27	1.59	95.3%
10	9.98244e+36	1000	451.24	9.648942e+36	1.44	93.3%
		10000	4425.12	9.73486e+36	1.73	93.3%

All two-way marginals are equal to 1 in this example. The exact numbers for such $m \times m \times m$ semimagic cubes can be obtained using the number of all Latin squares of size m.

Table 2.3: An additional summary of computational results on $m \times m \times m$ semimagic cubes for $m = 4, \ldots, 10$.

Dimension m	s	CPU time (sec)	Estimation	cv^2	Acceptance rate
4	2	27.1	51810.36	0.66	97.7%
5	2	58.1	25196288574	1.69	97.5%
6	2	97.1	6.339628e+18	2.56	94.8%
	3	99.3	1.269398e+22	2.83	96.5%
7	2	150.85	1.437412e+30	4.76	93.1%
	3	166.68	2.365389e+38	25.33	96.7%
8	2	229.85	5.369437e+44	6.68	89.8%
	3	256.70	3.236556e+59	7.05	94.5%
	4	328.52	2.448923e+64	11.98	94.3%
9	2	319.32	4.416787e+62	8.93	85.7%
	3	376.67	7.871387e+85	15.23	91.6%
	4	549.73	2.422237e+97	14.00	93.4%
10	2	429.19	2.166449e+84	10.46	83.3%
	3	527.14	6.861123e+117	26.62	90%
	4	883.34	3.652694e+137	33.33	93.8%
	5	1439.50	1.315069e+144	46.2	91.3%

All two-way marginals are equal to s in each simulation. The sample size is $\mathfrak{N} = 1000$ in this example.

marginal equals to s, $1 \leq s \leq \frac{m}{2}$. *The results are summarized in Table 2.3. In this example, we set the sample size* $\mathfrak{N} = 1000$.

Example 2.4.17 (Bootstrap-t confidence intervals of Semimagic Cubes)**.** *As we can see in Table 2.3, generally speaking for fixed sample size,* cv^2 *becomes larger when the number of tables is larger, and in this case, the estimator we get via the SIS procedure varies greatly in different iterations. Therefore, we propose to compute a* $(1-\alpha)100\%$ *confidence interval for each estimator via a non-parametric bootstrap method. In Appendix, we will give a pseudo code of a non-parametric bootstrap method to get the* $(1-\alpha)100\%$ *confidence interval for* $|\Sigma|$*). See Table 2.4 for some results of Bootstrap-t* 95% *confidence intervals (*$\alpha = 0.05$*).*

Table 2.4: A summary of Bootstrap-t confidence intervals for the number of semimagic cubes.

Dim	s	Estimation			cv^2			Acceptance Rate
		$\widehat{\lvert\Sigma\rvert}$	Lower 95%	Upper 95%	$\widehat{cv^2}$	Lower 95%	Upper 95%	
7	2	1.306480e+30	1.156686e+30	1.468754e+30	3.442306	2.678507	4.199513	93.3%
	3	3.033551e+38	2.245910e+38	4.087225e+38	22.84399	8.651207	35.080408	96.2%
8	2	5.010225e+44	4.200752e+44	5.902405e+44	6.712335	4.539368	8.590578	90.4%
	3	2.902294e+59	2.389625e+59	3.484405e+59	9.047914	5.680128	12.797488	93.1%
	4	2.474874e+64	1.847911e+64	3.295986e+64	21.53559	5.384647	32.166086	94.6%
9	2	4.548401e+62	3.682882e+62	5.593370e+62	10.07973	4.886817	15.406899	87.1%
	3	9.702672e+85	7.189849e+85	1.250875e+86	18.65302	11.33462	23.77980	92.5%
	4	2.023034e+97	1.547951e+97	2.561084e+97	14.96126	10.20331	19.09515	92.2%
10	2	2.570344e+84	1.908609e+84	3.339243e+84	17.83684	9.785778	24.231544	84.8%
	3	8.68783e+117	5.92233e+117	1.22271e+118	29.67200	18.64549	37.64892	90.2%
	4	4.12634e+137	2.94789e+137	5.52727e+137	23.36831	15.32719	31.02614	92%
	5	1.54956e+144	9.85557e+143	2.24043e+144	39.06521	20.23674	53.60838	91.8%

Dimensions and marginals$s$ are defined same with Table 2.3. $\widehat{|\Sigma|}$ means the estimator of $|\Sigma|$ and $\widehat{cv^2}$ means the estimator of cv^2. The sample size for the SIS procedure is $\mathfrak{N} = 1000$ and the sample size for bootstraping is $B = 5000$. Only the simulations in which cv^2 are relatively large are involved.

2.5 Experiment with Sampson's data set

Sampson recorded the social interactions among a group of monks while he visited as an experimenter on vision. He collected numerous sociometric rankings [6, 46]. The data is organized as a $18 \times 18 \times 10$ table and one can find the full data sets at `http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/UciData.htm#sampson`.

In this dataset, each layer of 18×18 table represents a social relation between 18 monks at some time point. Most of the present data are retrospective, collected after the breakup occurred. They concern a period during which a new cohort entered the monastery near the end of the study but before the major conflict began. The exceptions are "liking" data gathered at three times: SAMPLK1 to SAMPLK3 - that reflect changes in group sentiment over time (SAMPLK3 was collected in the same wave as the data described below). In the data set four relations are coded, with separate matrices for positive and negative ties on the 10 relation: esteem (SAMPES) and disesteem (SAMPDES); liking (SAMPLK which are SAMPLK1 to SAMPLK3) and disliking (SAMPDLK); positive influence (SAMPIN) and negative influence (SAMPNIN); praise (SAMPPR) and blame (SAMPNPR). In the original data set they listed top three choices and recorded as ranks. However, we set these ranks as an indicator (i.e., if they are in the top three choices, then we set one and else, zero).

We ran the SIS procedure with $\mathfrak{N} = 100000$ and a bootstrap sample size $B = 50000$. The estimator was 1.705e+117 and its 95% confidence interval was [1.119e+117, 2.681e+119]. We also had $cv^2 = 621.4$ with its 95% confidence interval be [324.29, 2959.65]. The CPU time was 70442 seconds (around 20 hours). The acceptance rate was 3%. We will discuss these results in Section 4.1.1.

Chapter 3 The Characteristic Imset Polytopes for Bayesian Networks

In Section 1.3.2, we showed that the problem of learning Bayesian networks in a class of graphs $\mathcal{G}$ can be formulated as a LP problem over the corresponding characteristic imset polytope $\mathbf{P}_{\mathcal{G},c}$, which gives us a systematic way to find the best model with the optimality certificate rather than finding it by the brute-force search. In general, however, the dimension of $\mathbf{P}_{\mathcal{G},c}$, with the fixed set of nodes N, can be exponentially large (e.g. $dim(\mathbf{P}_{DAGs(N),c}) = 2^{|N|} - |N| - 1$) and there are double exponentially many vertices as well as facets of $\mathbf{P}_{\mathcal{G},c}$. Thus it is infeasible to optimize by software if $|N| > 6$ [39, Section 6.4.2]. In order to solve the LP problem for a larger $|N|$, we need to understand the structure of $\mathbf{P}_{\mathcal{G},c}$, such as combinatorial description of edges and facets of the polytope so that we might be able to apply a simplex method [48, Chapter 11] to find an optimal solution. However, in general, studying the structure of $\mathbf{P}_{\mathcal{G},c}$ is challenging because there are too many facets and too many edges of the polytope. Therefore, in this dissertation, we start with a particular family of BNs, namely **diagnosis models**, because the dimension of $\mathbf{P}_{\mathcal{G},c}$ for diagnosis models is dramatically reduced by prohibiting certain types of edges in the DAGs. These models are of particular interest because we can generalize our results in diagnosis models to a larger family of BNs (see Section 3.3 and Section 4.2.1): all BNs with the same **underlying ordering** of nodes.

In medical studies, researchers are often interested in probabilistic models in order to correctly diagnose a disease from a patient symptoms. The diagnoses models, also known as the Quick Medical Reference (QMR) diagnostic model, is introduced in [49] to diagnose a disease from a given set of symptoms of a patient (e.g. [40]). The DAGs that represent the diagnosis models are directed bipartite graph with two sets of nodes, one representing m diseases and one representing n symptoms, and set of

directed edges from nodes representing diseases to nodes representing symptoms (see Definition 3.1.1).

This chapter is organized as follows. In Section 3.1 we introduce notation and definitions for diagnosis models, and give some properties of characteristic imsets of these models. In Section 3.2, we show that the cim-polytopes of diagnosis models are directed product of simplices, and give a combinatorial description of edges and a expression of facets of the cim-polytopes. Then we generalize the results in Section 3.2 to a larger family of BN in Section 3.3.

3.1 Diagnosis models and propositions of the corresponding characteristic imsets

In this section, we will first review the definition of diagnosis models. Then we will show that these models can lead to some properties of their characteristic imsets. Lastly, we will give two examples of the characteristic imsets and characteristic imset polytopes (cim-polytopes) of diagnosis models.

Definition 3.1.1. *A **diagnosis model** is a CI model induced by a **directed bipartite graph** $G \in DAGs(N)$ that can be described as following:*

- *its nodes $N = \{a_1, \ldots, a_m\} \cup \{b_1, \ldots, b_n\}$ can be divided into disjoint sets $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1, \ldots, b_n\}$; and*
- *if a directed edge $a \to b$ in G, then $a \in A$ and $b \in B$.*

An example of such directed bipartite graph is given in Figure 3.1.

The naming of the diagnosis models comes from an interpretation of the two sets of nodes: nodes in A can be interpreted as diseases, while nodes in B can be interpreted as symptoms, and every single edge can only be drawn from a disease to a symptom.

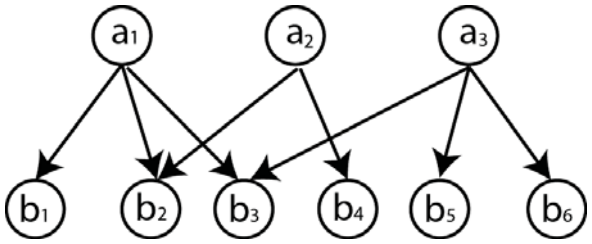


Figure 3.1: An example of a directed bipartite graph, $m = 3$, $n = 6$.

For fixed A and B, where $|A| = m$ and $|B| = n$, we define notation: $\mathcal{G}_{m,n} = \{$All possible directed bipartite graphs defined in Definition 3.1.1 based on A and $B\}$. We are going to study the properties of c_G, where $G \in \mathcal{G}_{m,n}$.

Proposition 3.1.2. *Fix $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1, \ldots, b_n\}$. Assume $G \in \mathcal{G}_{m,n}$ and $|N| = m + n > 2$. Then $c_G(T)$ is possible to take value 1 if and only if T has the form of $a_{i_1} \ldots a_{i_k} b_j$, where $1 \leq k \leq m$, $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, m\}$ and $j \in \{1, \ldots, n\}$.*

Proof. Notice that $\forall T \subseteq N$, $|T| \geq 2$, we can write T in the form of:

$$T = a_{i_1} \ldots a_{i_k} b_{j_1} \ldots b_{j_l}, \text{ where } \begin{array}{l} 0 \leq k \leq m, \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, m\}, \\ 0 \leq l \leq n, \{j_1, \ldots, j_l\} \subseteq \{1, \ldots, n\}, \\ k + l \geq 2. \end{array} \tag{3.1.1}$$

We need to prove that l can neither be 0 nor greater than 1, i.e. $l = 1$.

(a) If $l = 0$. $\forall\ s, t \in \{i_1, \ldots, i_k\}$, by Definition 3.1.1, $a_s \to a_t$ is not in G. This means $a_s \notin pa_G(a_t)$. Hence $\forall\ t \in \{i_1, \ldots, i_k\}$, $T \setminus \{a_t\} \nsubseteq pa_G(a_t)$. $c_G(T) = 0$.

(b) If $l > 1$. Similarly with above, by Definition 3.1.1, $\forall\ s', t' \in \{j_1, \ldots, j_l\}$, $b_{s'} \notin pa_G(b_{t'})$. Moreover, $\forall\ t \in \{i_1, \ldots, i_k\}$ and $t' \in \{j_1, \ldots, j_l\}$, $b_{t'} \notin pa_G(a_t)$. $c_G(T) = 0$.

□

Proposition 3.1.3. *Notation is adopted from Proposition 3.1.2. Suppose T has the form of $a_{i_1} \dots a_{i_k} b_j$, where $1 \leq k \leq m$, $\{i_1, \dots, i_k\} \subseteq \{1, \dots, m\}$ and $j \in \{1, \dots, n\}$, then $c_G(T) = \prod_{s=i_1,\dots,i_k} c_G(a_s b_j)$.*

Proof. Again by Definition 3.1.1, $\forall\ s, t \in \{i_1, \dots, i_k\}$, $a_s \notin pa_G(a_t)$. Therefore:

$$\begin{aligned} c_G(T) = 1 &\iff \{a_{i_1} \dots a_{i_k}\} \subseteq pa_G(b_j) \\ &\iff a_s \in pa_G(b_j),\ \forall s = i_1, \dots, i_k \\ &\iff c_G(a_s b_j) = 1,\ \forall s = i_1, \dots, i_k. \end{aligned} \tag{3.1.2}$$

Recall that $c_G(T)$ is binary. Thus $c_G(T) = \prod_{s=i_1,\dots,i_k} c_G(a_s b_j)$. □

Remark 3.1.4. *Proposition 3.1.3 implies that $\forall G \in \mathcal{G}_{m,n}$, c_G is determined by only $m \cdot n$ coordinates, $\{c_G(a_i b_j) : i = 1, \dots, m, j = 1, \dots, n\}$, i.e. the existence of directed edges $a_i \to b_j$, $i = 1, \dots, m$ and $j = 1, \dots, n$. Another way to see this property is that $\forall\ G \in \mathcal{G}_{m,n}$, G can be determined by $pa_G(b_j)$, $b_j \in B$. Thus if we consider a permutation of coordinates in c_G that corresponds to a permutation of T where T has the form in Proposition 3.1.2, then these coordinates can be broken into n parts:*

$$\underline{a_1 b_1, \dots, a_m b_1, \dots, a_1 \dots a_m b_1}, \underline{a_1 b_2, \dots, a_1 \dots a_m b_2}, \dots, \underline{a_1 b_n, \dots, a_1 \dots a_m b_n},$$

where the s-th part of coordinations $c_G(T)$, $T \in \{a_1 b_s, \dots, a_m b_s, a_1 a_2 b_s, \dots, a_1 \dots a_m b_s\}$ only depend on $pa_G(b_s)$, and different parts are completely irrelevant in the sense that $pa_G(b_s)$, $b_s \in B$, can be decided separately.

Proposition 3.1.5. *Fix m and n. The number of elements in $\mathcal{G}_{m,n}$ is 2^{mn}.*

Proof. This is trivial because of Remark 3.1.4 since there are mn possible edges that can be assigned: $a_i \to b_j$, where $i = 1, \dots, m$ and $j = 1, \dots, n$, and there are $\sum_{k=0}^{mn} \binom{mn}{k} = 2^{mn}$ many possible ways to assign the existence of these edges. □

Proposition 3.1.6. *Suppose $G \in \mathcal{G}_{m,n}$. The number of non-zero coordinates in c_G is at most $n \cdot (2^m - 1)$.*

Proof. This result is straightforward from Proposition 3.1.2 by counting the number of coordinates $c_G(T)$, where T has the form shown in Proposition 3.1.2. Note that when $|T| > m+1$, $\exists\, b_{j_1}, b_{j_2} \in \{1, \ldots, n\}$ *s.t.* $b_{j_1}, b_{j_2} \in T$, i.e. $c_G(T) = 0$ by Proposition 3.1.2. When $2 \leq |T| \leq m+1$, the number of coordinates of form $c_G(a_{i_1} \ldots a_{i_{|T|-1}} b_j)$, where $\{i_1, \ldots, i_{|T|-1}\} \subseteq \{1, \ldots, m\}$ and $j \in \{1, \ldots, n\}$, is $\binom{m}{|T|-1} \cdot n$. Hence the number of possible non-zero coordinates is:

$$\sum_{|T|=2}^{m+1} \binom{m}{|T|-1} \cdot n = n \cdot \sum_{k=1}^{m} \binom{m}{k} = n \cdot (2^m - 1).$$

□

For fixed m and n, consider the characteristic imset polytope for $\mathcal{G}_{m,n}$ (see the definition of cim-polytopes in Section 1.3.2) and let $\mathbf{P}_{m,n}$ be the cim-polytope: $\mathbf{P}_{m,n} := \mathbf{P}_{\mathcal{G}_{m,n},c}$. Proposition 3.1.6 implies that the dimension of $\mathbf{P}_{m,n}$ is at most $n \cdot (2^m - 1)$. We are going to show in Section 3.2 that the dimension of $\mathbf{P}_{m,n}$ is actually exactly $n \cdot (2^m - 1)$.

Before we end this section, we are going to show two examples of $\mathcal{G}_{m,n}$ and the characteristic imsets c_G, $G \in \mathcal{G}_{m,n}$. The coordinates of the characteristic imsets with the form in Proposition 3.1.2 will be ordered as the permutation showed in Remark 3.1.4, and we can observe in the examples that the other coordinates will be all zeroes.

Example 3.1.7 (Only One Disease)**.** *Let $A = \{a_1\}$ and $B = \{b_1, \ldots, b_n\}$. By Proposition 3.1.2, $c_G(T)$ will be zero if it doesn't have the form of $a_1 b_j$, $b_j \in \{1, \ldots, n\}$. Consider all combination of the existence of edges $a_1 \to b_j$, $b_j \in \{1, \ldots, n\}$, we can see that the cim-polytope $\mathbf{P}_{1,n}$ is the n-cube. A simple example of $n = 3$ is given here. The list of c_G, $\forall G \in \mathcal{G}_{1,3}$, is showed as a matrix. We can see that the last 8 columns*

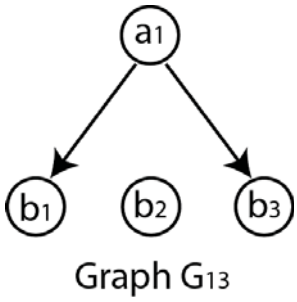


Figure 3.2: Graph G_{13} in $\mathcal{G}_{1,3}$

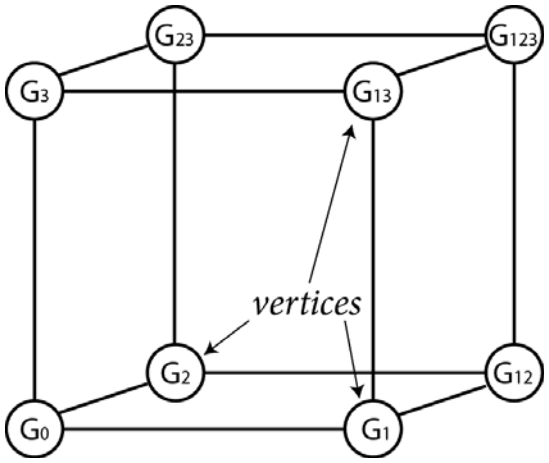


Figure 3.3: The characteristic imset polytope $\mathbf{P}_{1,3}$

in the matrix are all zeros.

$$\begin{array}{c} \\ \begin{pmatrix} c_{G_0} \\ c_{G_1} \\ c_{G_2} \\ c_{G_3} \\ c_{G_{12}} \\ c_{G_{23}} \\ c_{G_{13}} \\ c_{G_{123}} \end{pmatrix} \end{array} = \begin{array}{c} \begin{array}{ccccccccccc} T & a_1b_1 & a_1b_2 & a_1b_3 & b_1b_2 & b_2b_3 & b_1b_3 & a_1b_1b_2 & a_1b_2b_3 & a_1b_1b_3 & b_1b_2b_3 & a_1b_1b_2b_3 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Example 3.1.8 ($m = 2$, $n = 2$)**.** *We can encode the four possible edges as following: encode $a_1 \to b_1$ as 1, $a_2 \to b_1$ as 2, $a_1 \to b_2$ as 3, and $a_2 \to b_2$ as 4. An example of encoding the subscript is given by Figure 3.4. The list of c_G, $\forall G \in \mathcal{G}_{2,2}$, is showed as*

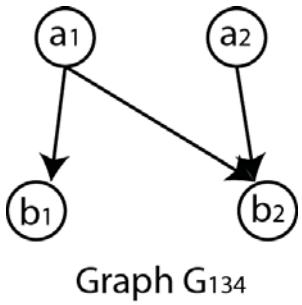


Figure 3.4: Graph G_{134} in $\mathcal{G}_{2,2}$

The subscript "134" means the existence of edges $a_1 \to b_1$, $a_1 \to b_2$, and $a_2 \to b_2$.

a matrix. We can see that the last 5 columns in the matrix are all zeros.

$$\begin{array}{c} \\ \begin{pmatrix} c_{G_0} \\ c_{G_1} \\ c_{G_2} \\ c_{G_3} \\ c_{G_4} \\ c_{G_{12}} \\ c_{G_{13}} \\ c_{G_{14}} \\ c_{G_{23}} \\ c_{G_{24}} \\ c_{G_{34}} \\ c_{G_{123}} \\ c_{G_{134}} \\ c_{G_{124}} \\ c_{G_{234}} \\ c_{G_{1234}} \end{pmatrix} \end{array} = \begin{array}{c} \begin{array}{cccccccccccc} T & a_1b_1 & a_2b_1 & a_1a_2b_1 & a_1b_2 & a_2b_2 & a_1a_2b_2 & a_1a_2 & b_1b_2 & a_1b_1b_2 & a_2b_1b_2 & a_1a_2b_1b_2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

3.2 The characteristic imset polytopes (cim-polytopes) for diagnosis models

3.2.1 Combinatorial description of edges on $\mathbf{P}_{m,n}$

Definition 3.2.1. *Consider a class of graphs $\mathcal{G}$. $\forall G, H \in \mathcal{G}$, G and H are called* ***neighbors*** *if c_G and c_H form an edge in $\mathbf{P}_{\mathcal{G},c}$, the characteristic imset polytope for $\mathcal{G}$.*

Lemma 3.2.2. *Fix m. Suppose G_1, $G_2 \in \mathcal{G}_{m,1}$ are arbitrary two distinct graphs in $\mathcal{G}_{m,1}$. Then G_1 and G_2 are neighbors, i.e. c_{G_1} and c_{G_2} form an edge in $\mathbf{P}_{m,1}$.*

Proof. Let $N = A \cup B$, where $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1\}$. By Remark 1.3.5, we need to prove: $\exists$ a cost vector w, such that $w \cdot c_{G_1} = w \cdot c_{G_2} > w \cdot c_{G_3}$, $\forall\ G_3 \in \mathcal{G}_{m,1}$ distinct with G_1 and G_2.

By Remark 3.1.4, G_1 and G_2 are determined by $pa_{G_1}(b_1)$ and $pa_{G_2}(b_1)$, respectively. We will discuss by two scenarios of $pa_{G_1}(b_1)$ and $pa_{G_2}(b_1)$: one is a subset of the other, and neither one is a subset of the other.

(1) One is a subset of the other. WLOG, suppose $pa_{G_1}(b_1) \subsetneq pa_{G_2}(b_1)$.

Define: $A_1 = pa_{G_1}(b_1)$, $A_2 = pa_{G_2}(b_1)$, $A_{2\setminus 1} = pa_{G_2}(b_1) \setminus pa_{G_1}(b_1)$, and $A_{comp} = (pa_{G_2}(b_1))^c$ (i.e. the complement set of $pa_{G_2}(b_1)$). Note that: $A_{2\setminus 1} \neq \emptyset$, A_1 and A_{comp} can be $\emptyset$; A_1, $A_{2\setminus 1}$ and A_{comp} is a partition of N.

Consider a function $w : \mathcal{P}(N) \mapsto \mathbb{R}$ where $w(T) = 0$ if $|T| < 2$. Then similar with imsets, w can also be considered as a vector, and we assume that the permutations of coordinates in w and in characteristic imsets coincide.

– If $|A_{2\setminus 1}| > 1$, we define w as:

$$w(T) = \begin{cases} c & \text{for } \ T = a_i b_j,\ a_i \in A_1 \\ -c & \text{for } \ T = a_i b_j,\ a_i \notin A_1 \\ |A_{2\setminus 1}| \cdot c & \text{for } \ T = A_{2\setminus 1} \cup \{b_1\} \\ 0 & \text{for } \ T \subset N, |T| > 2, \text{ and } T \neq A_{2\setminus 1} \cup \{b_1\} \end{cases}$$

where c is a positive number.

Then $\forall\ G_3 \in \mathcal{G}_{m,1}$, we have:

$$\begin{aligned} w \cdot c_{G_3} &= |A_1 \cap pa_{G_3}(b_1)| \cdot c - |pa_{G_3}(b_1) \setminus A_1| \cdot c + |A_{2\setminus 1}| \cdot c \cdot c_{G_3}(A_{2\setminus 1} \cup \{b_1\}) \\ &= |A_1 \cap pa_{G_3}(b_1)| \cdot c - |pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c - |pa_{G_3}(b_1) \cap A_{comp}| \cdot c \\ &\quad + |A_{2\setminus 1}| \cdot c \cdot c_{G_3}(A_{2\setminus 1} \cup \{b_1\}). \end{aligned}$$

In this equation:

* $|A_1 \cap pa_{G_3}(b_1)| \cdot c \leq |A_1| \cdot c$, where "=" holds if and only if $A_1 \subset pa_{G_3}(b_1)$;
* $-|pa_{G_3}(b_1) \cap A_{2\backslash 1}| \cdot c + |A_{2\backslash 1}| \cdot c \cdot c_{G_3}(A_{2\backslash 1} \cup \{b_1\}) \leq 0$, where "=" holds if and only if $pa_{G_3}(b_1) \cap A_{2\backslash 1} = \emptyset$ or $A_{2\backslash 1}$;
* $-|pa_{G_3}(b_1) \cap A_{comp}| \cdot c \leq 0$, where "=" holds if and only if $pa_{G_3}(b_1) \cap A_{comp} = \emptyset$.

Therefore, $w \cdot c_{G_3} \leq |A_1| \cdot c$, where "=" holds if and only if $G_3 = G_1$ or G_2.

– If $|A_{2\backslash 1}| = 1$, we let $A_{2\backslash 1} = \{a_q\}$, and define w as:

$$w(T) = \begin{cases} c & \text{for } T = a_i b_j,\ a_i \in A_1 \\ -c & \text{for } T = a_i b_j,\ a_i \notin A_2 \\ 0 & \text{for } T = a_q b_1 \\ 0 & \text{for } T \subset N, |T| > 2, \text{ and } T \neq A_{2\backslash 1} \cup \{b_1\} \end{cases}$$

where c is a positive number.

Then $\forall\ G_3 \in \mathcal{G}_{m,1}$, we have:

$$w \cdot c_{G_3} = |A_1 \cap pa_{G_3}(b_1)| \cdot c - |pa_{G_3}(b_1) \cap A_{comp}| \cdot c.$$

Again, in this equation:

* $|A_1 \cap pa_{G_3}(b_1)| \cdot c \leq |A_1| \cdot c$, where "=" holds if and only if $A_1 \subset pa_{G_3}(b_1)$;
* $-|pa_{G_3}(b_1) \cap A_{comp}| \cdot c \leq 0$, where "=" holds if and only if $pa_{G_3}(b_1) \cap A_{comp} = \emptyset$.

To satisfy the above two conditions, we must have $pa_{G_3}(b_1) = A_1$ or $(A_1 \cup a_q)$. Therefore, again, we have: $w \cdot c_{G_3} \leq |A_1| \cdot c$, where "=" holds if and only if $G_3 = G_1$ or G_2.

(2) Neither one is a subset of the other.

Define: $A_1 = pa_{G_1}(b_1)$, $A_2 = pa_{G_2}(b_1)$, $A_{1\cap 2} = pa_{G_1}(b_1) \cap pa_{G_2}(b_1)$, $A_{1\backslash 2} = pa_{G_1}(b_1) \backslash pa_{G_2}(b_1)$, $A_{2\backslash 1} = pa_{G_2}(b_1) \backslash pa_{G_1}(b_1)$, $A_{1\cup 2} = pa_{G_1}(b_1) \cup pa_{G_2}(b_1)$ and

$A_{comp} = (A_{1\cup 2})^c$. Note that: $A_{1\setminus 2}$, $A_{2\setminus 1} \neq \emptyset$, $A_{1\cap 2}$ and A_{comp} can be $\emptyset$; $A_{1\cap 2}$, $A_{1\setminus 2}$, $A_{2\setminus 1}$, and A_{comp} is a partition of N.

Consider a function w similar with part (1) that can also be considered as a vector such that the permutations of coordinates in w and in characteristic imsets coincide.

– If $|A_{1\setminus 2}| > 1$ and $|A_{2\setminus 1}| > 1$, we define w as:

$$w(T) = \begin{cases} c & \text{for} \quad T = a_i b_j,\ a_i \in A_{1\cap 2} \\ -c & \text{for} \quad T = a_i b_j,\ a_i \notin A_{1\cap 2} \\ -2c & \text{for} \quad T = A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\} \\ (|A_{1\setminus 2}| + 1) \cdot c & \text{for} \quad T = A_{1\setminus 2} \cup \{b_1\} \\ (|A_{2\setminus 1}| + 1) \cdot c & \text{for} \quad T = A_{2\setminus 1} \cup \{b_1\} \\ 0 & \text{for} \quad \text{other } T \subset N, |T| > 2 \end{cases}$$

where c is a positive number.

Then $\forall\ G_3 \in \mathcal{G}_{m,1}$, we have:

$$\begin{aligned} w \cdot c_{G_3} &= |pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c - |pa_{G_3}(b_1) \cap A_{1\setminus 2}| \cdot c \\ &\quad -|pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c - |pa_{G_3}(b_1) \cap A_{comp}| \cdot c \\ &\quad +(|A_{1\setminus 2}| + 1) \cdot c \cdot c_{G_3}(A_{1\setminus 2} \cup \{b_1\}) + (|A_{2\setminus 1}| + 1) \cdot c \cdot c_{G_3}(A_{2\setminus 1} \cup \{b_1\}) \\ &\quad -2c \cdot c_{G_3}(A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\}) \\ &= |pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c \\ &\quad -|pa_{G_3}(b_1) \cap A_{1\setminus 2}| \cdot c + (|A_{1\setminus 2}| + 1) \cdot c \cdot c_{G_3}(A_{1\setminus 2} \cup \{b_1\}) \\ &\quad -|pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c + (|A_{2\setminus 1}| + 1) \cdot c \cdot c_{G_3}(A_{2\setminus 1} \cup \{b_1\}) \\ &\quad -2c \cdot c_{G_3}(A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\}) \\ &\quad -|pa_{G_3}(b_1) \cap A_{comp}| \cdot c \end{aligned}$$

In this equation:

* $|pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c \leq |A_{1\cap 2}| \cdot c$, where "=" holds if and only if $A_{1\cap 2} \subset pa_{G_3}(b_1)$;

* $-|pa_{G_3}(b_1) \cap A_{1\backslash 2}| \cdot c + (|A_{1\backslash 2}| + 1) \cdot c \cdot c_{G_3}(A_{1\backslash 2} \cup \{b_1\}) \leq c$, where "=" holds if and only if $A_{1\backslash 2} \subset pa_{G_3}(b_1)$;
* $-|pa_{G_3}(b_1) \cap A_{2\backslash 1}| \cdot c + (|A_{2\backslash 1}| + 1) \cdot c \cdot c_{G_3}(A_{2\backslash 1} \cup \{b_1\}) \leq c$, where "=" holds if and only if $A_{2\backslash 1} \subset pa_{G_3}(b_1)$;
* $-2c \cdot c_{G_3}(A_{1\backslash 2} \cup A_{2\backslash 1} \cup \{b_1\}) \leq 0$, where "=" holds if and only if $(A_{1\backslash 2} \cup A_{2\backslash 1}) \nsubseteq pa_{G_3}(b_1)$;
* $-|pa_{G_3}(b_1) \cap A_{comp}| \cdot c \leq 0$, where "=" holds if and only if $pa_{G_3}(b_1) \cap A_{comp} = \emptyset$.

The above conditions cannot be satisfied simultaneously, but notice that:

* when $pa_{G_3}(b_1) = A_{1\cap 2}$, $w \cdot c_{G_3} = |A_{1\cap 2}| \cdot c + 0 + 0 + 0 + 0 = |A_{1\cap 2}| \cdot c$;
* when $pa_{G_3}(b_1) = A_1$, i.e. $G_3 = G_1$, $w \cdot c_{G_3} = |A_{1\cap 2}| \cdot c + c + 0 + 0 + 0 = (|A_{1\cap 2}| + 1) \cdot c$;
* when $pa_{G_3}(b_1) = A_2$, i.e. $G_3 = G_2$, $w \cdot c_{G_3} = |A_{1\cap 2}| \cdot c + 0 + c + 0 + 0 = (|A_{1\cap 2}| + 1) \cdot c$;
* when $pa_{G_3}(b_1) = A_{1\cup 2}$, $w \cdot c_{G_3} = |A_{1\cap 2}| \cdot c + c + c - 2c + 0 = |A_{1\cap 2}| \cdot c$.

Now it is obvious that $w \cdot c_{G_3} \leq (|A_{1\cap 2}| + 1) \cdot c$, where "=" holds if and only if $G_3 = G_1$ or G_2.

– If only one of $|A_{1\backslash 2}|$ and $|A_{2\backslash 1}|$ is 1. Suppose $|A_{1\backslash 2}| = 1$ and $|A_{2\backslash 1}| > 1$. We define w as:

$$w(T) = \begin{cases} c & \text{for} \quad T = a_i b_j,\ a_i \in A_1 \\ -c & \text{for} \quad T = a_i b_j,\ a_i \notin A_1 \\ -2c & \text{for} \quad T = A_{1\backslash 2} \cup A_{2\backslash 1} \cup \{b_1\} \\ (|A_{2\backslash 1}| + 1) \cdot c & \text{for} \quad T = A_{2\backslash 1} \cup \{b_1\} \\ 0 & \text{for} \quad \text{other } T \subset N, |T| > 2 \end{cases}$$

where c is a positive number.

Then $\forall\ G_3 \in \mathcal{G}_{m,1}$, we have:

$$\begin{aligned} w \cdot c_{G_3} &= |pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c + |pa_{G_3}(b_1) \cap A_{1\setminus 2}| \cdot c \\ &\quad -|pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c - |pa_{G_3}(b_1) \cap A_{comp}| \cdot c \\ &\quad +(|A_{2\setminus 1}| + 1) \cdot c \cdot c_{G_3}(A_{2\setminus 1} \cup \{b_1\}) - 2c \cdot c_{G_3}(A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\}) \\ &= |pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c \\ &\quad +|pa_{G_3}(b_1) \cap A_{1\setminus 2}| \cdot c \\ &\quad -|pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c + (|A_{2\setminus 1}| + 1) \cdot c \cdot c_{G_3}(A_{2\setminus 1} \cup \{b_1\}) \\ &\quad -2c \cdot c_{G_3}(A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\}) \\ &\quad -|pa_{G_3}(b_1) \cap A_{comp}| \cdot c. \end{aligned}$$

In this equation:

* $|pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c \leq |A_{1\cap 2}| \cdot c$, where "=" holds if and only if $A_{1\cap 2} \subset pa_{G_3}(b_1)$;
* $|pa_{G_3}(b_1) \cap A_{1\setminus 2}| \cdot c \leq c$, where "=" holds if and only if $A_{1\setminus 2} \subset pa_{G_3}(b_1)$;
* $-|pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c + (|A_{2\setminus 1}| + 1) \cdot c \cdot c_{G_3}(A_{2\setminus 1} \cup \{b_1\}) \leq c$, where "=" holds if and only if $A_{2\setminus 1} \subset pa_{G_3}(b_1)$;
* $-2c \cdot c_{G_3}(A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\}) \leq 0$, where "=" holds if and only if $(A_{1\setminus 2} \cup A_{2\setminus 1}) \nsubseteq pa_{G_3}(b_1)$;
* $-|pa_{G_3}(b_1) \cap A_{comp}| \cdot c \leq 0$, where "=" holds if and only if $pa_{G_3}(b_1) \cap A_{comp} = \emptyset$.

The above conditions cannot be satisfied simultaneously, but it is similar with the case of "$|A_{1\setminus 2}| > 1$ and $|A_{2\setminus 1}| > 1$" to show that $w \cdot c_{G_3} \leq (|A_{1\cap 2}| + 1) \cdot c$, where "=" holds if and only if $G_3 = G_1$ or G_2.

– If $|A_{1\setminus 2}| = |A_{2\setminus 1}| = 1$, we define w as:

$$w(T) = \begin{cases} c & \text{for} \quad T = a_i b_j,\ a_i \in A_{1\cup 2} \\ -c & \text{for} \quad T = a_i b_j,\ a_i \notin A_{1\cup 2} \\ -2c & \text{for} \quad T = A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\} \\ 0 & \text{for} \quad \text{other } T \subset N, |T| > 2 \end{cases}$$

where c is a positive number.

Then $\forall\ G_3 \in \mathcal{G}_{m,1}$, we have:

$$\begin{aligned} w \cdot c_{G_3} &= |pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c \\ &\quad + |pa_{G_3}(b_1) \cap A_{1\setminus 2}| \cdot c + |pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c \\ &\quad - 2c \cdot c_{G_3}(A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\}) \\ &\quad - |pa_{G_3}(b_1) \cap A_{comp}| \cdot c. \end{aligned}$$

In this equation:

* $|pa_{G_3}(b_1) \cap A_{1\cap 2}| \cdot c \leq |A_{1\cap 2}| \cdot c$, where "=" holds if and only if $A_{1\cap 2} \subset pa_{G_3}(b_1)$;
* $|pa_{G_3}(b_1) \cap A_{1\setminus 2}| \cdot c \leq c$, where "=" holds if and only if $A_{1\setminus 2} \subset pa_{G_3}(b_1)$;
* $|pa_{G_3}(b_1) \cap A_{2\setminus 1}| \cdot c \leq c$, where "=" holds if and only if $A_{2\setminus 1} \subset pa_{G_3}(b_1)$;
* $-2c \cdot c_{G_3}(A_{1\setminus 2} \cup A_{2\setminus 1} \cup \{b_1\}) \leq 0$, where "=" holds if and only if $(A_{1\setminus 2} \cup A_{2\setminus 1}) \nsubseteq pa_{G_3}(b_1)$;
* $-|pa_{G_3}(b_1) \cap A_{comp}| \cdot c \leq 0$, where "=" holds if and only if $pa_{G_3}(b_1) \cap A_{comp} = \emptyset$.

The above conditions cannot be satisfied simultaneously, but it is similar with the case of "$|A_{1\setminus 2}| > 1$ and $|A_{2\setminus 1}| > 1$" to show that: $w \cdot c_{G_3} \leq (|A_{1\cap 2}| + 1) \cdot c$, where "=" holds if and only if $G_3 = G_1$ or G_2.

□

Theorem 3.2.3. *Fix m and n. Two graphs, G_1, $G_2 \in \mathcal{G}_{m,n}$ are neighbors if and only if $\exists\ b_i \in B$ such that $pa_{G_1}(b_i) \neq pa_{G_2}(b_i)$ and $pa_{G_1}(b_j) = pa_{G_2}(b_j)$, $\forall\ b_j \in B$ and $b_j \neq b_i$, i.e. all nodes but one have exactly the same parent sets in G_1 and G_2.*

Proof. We will prove "if" and "only if" separately.

(1) Prove "if" part.

Suppose G_1, $G_2 \in \mathcal{G}_{m,n}$, and there exists $b_i \in B$ such that $pa_{G_1}(b_i) \neq pa_{G_2}(b_i)$ and $pa_{G_1}(b_j) = pa_{G_2}(b_j)$, $\forall\ b_j \in B$, $b_j \neq b_i$. We need to prove G_1 and G_2 are neighbors.

Consider an arbitrary graph $G_3 \in \mathcal{G}_{m,n}$. By Remark 1.3.5, we need to prove: $\exists$ a cost vector w such that $w \cdot c_{G_1} = w \cdot c_{G_2} \geq w \cdot c_{G_3}$, where "=" holds if and only if $G_3 = G_1$ or G_2.

Define the following graphs (a graphical example will be given in Remark 3.2.4):

- G_1', G_2', $G_3' \in \mathcal{G}_{m,1}$ with symptom $B_{m,1} = \{b_i\}$ such that $pa_{G_1'}(b_i) = pa_{G_1}(b_i)$, $pa_{G_2'}(b_i) = pa_{G_2}(b_i)$ and $pa_{G_3'}(b_i) = pa_{G_3}(b_i)$;
- G_0, $G_3'' \in \mathcal{G}_{m,(n-1)}$ with symptoms $B_{m,(n-1)} = B \backslash \{b_i\}$ such that $pa_{G_0}(b_j) = pa_{G_1}(b_j) = pa_{G_2}(b_j)$ and $pa_{G_3''}(b_j) = pa_{G_3}(b_j)$, $\forall\ b_j \in B_{m,(n-1)}$.

By Remark 3.1.4, with a proper permutation of coordinates, we can write the characteristic imsets of G_1, G_2 and G_3 in the form of:

$$\begin{aligned} c_{G_1} &= (c_{G_1'},\ c_{G_0}) \\ c_{G_2} &= (c_{G_2'},\ c_{G_0}) \\ c_{G_3} &= (c_{G_3'},\ c_{G_3''}) \end{aligned}$$

- By Lemma 3.2.2, G_1' and G_2' are neighbors, i.e. $\exists$ a cost vector w_1 such that $w_1 \cdot c_{G_1'} = w_1 \cdot c_{G_2'} \geq w_1 \cdot c_{G_3'}$, $\forall\ G_3' \in \mathcal{G}_{m,1}$, where "=" holds if and only if $G_3' = G_1'$ or G_2'.
- Since $c_{G_0} \in vert(\mathbf{P}_{\mathcal{G}_{m,(n-1),c}})$, $\exists$ a cost vector w_2 such that $w_2 \cdot c_{G_0} \geq w_2 \cdot c_{G_3''}$, $\forall\ G_3'' \in \mathcal{G}_{m,(n-1)}$, where "=" holds if and only if $G_3'' = G_0$.

Let $w = (w_1\ w_2)$. We have:

$$\begin{aligned} w \cdot c_{G_1} &= w_1 \cdot c_{G_1'} + w_2 \cdot c_{G_0} \\ &= w_1 \cdot c_{G_2'} + w_2 \cdot c_{G_0} \quad = \quad w \cdot c_{G_2} \\ &\geq w_1 \cdot c_{G_3'} + w_2 \cdot c_{G_3''} \quad = \quad w \cdot c_{G_3}, \end{aligned}$$

where "=" holds if and only if i) $G'_3 = G'_1$ or G'_2, and ii) $G'''_3 = G_0$, i.e. $G_3 = G_1$ or G_2.

(2) Prove "only if" part.

Suppose G_1, $G_2 \in \mathcal{G}_{m,n}$ are neighbors. i.e. $\exists$ a cost vector w such that $w \cdot c_{G_1} = w \cdot c_{G_2} > w \cdot c_G$, $\forall\ G \in \mathcal{G}_{m,n}$, $G \neq G_1$, G_2. We are going to prove this part by contradiction.

Suppose $\exists\ b_i$, $b_j \in B$ distinct, $pa_{G_1}(b_i) \neq pa_{G_2}(b_i)$ and $pa_{G_1}(b_j) \neq pa_{G_2}(b_j)$.

Define the following graphs (a graphical example will be given in Remark 3.2.4):

- G'_1, $G'_2 \in \mathcal{G}_{m,1}$ with symptom $B_{m,1} = \{b_i\}$ such that $pa_{G'_1}(b_i) = pa_{G_1}(b_i)$ and $pa_{G'_2}(b_i) = pa_{G_2}(b_i)$;
- G''_1, $G''_2 \in \mathcal{G}_{m,1}$ with symptom $B_{m,1} = \{b_j\}$ such that $pa_{G''_1}(b_j) = pa_{G_1}(b_j)$ and $pa_{G''_2}(b_j) = pa_{G_2}(b_j)$;
- $G'''_1, G'''_2 \in \mathcal{G}_{m,(n-2)}$ with symptoms $B_{m,(n-2)} = B \backslash \{b_i, b_j\}$ such that $pa_{G'''_1}(b_k) = pa_{G_1}(b_k)$ and $pa_{G'''_2}(b_k) = pa_{G_2}(b_k)$, $\forall\ b_k \in B_{m,(n-2)}$;
- $G_3 \in \mathcal{G}_{m,n}$ is all the same with G_1 but $pa_{G_3}(b_i) = pa_{G_2}(b_i)$;
- $G_4 \in \mathcal{G}_{m,n}$ is all the same with G_1 but $pa_{G_4}(b_j) = pa_{G_2}(b_j)$;
- $G_5 \in \mathcal{G}_{m,n}$ is all the same with G_2 but $pa_{G_5}(b_i) = pa_{G_1}(b_i)$ and $pa_{G_5}(b_j) = pa_{G_1}(b_j)$, notice that G_5 might be same with G_1.

Similarly with part (1), with a proper permutation of coordinates, we can write the characteristic imsets of G_1, G_2, G_3, G_4 and G_5 in the following form:

$$
\begin{aligned}
c_{G_1} &= (c_{G'_1},\ c_{G''_1},\ c_{G'''_1}) \\
c_{G_2} &= (c_{G'_2},\ c_{G''_2},\ c_{G'''_2}) \\
c_{G_3} &= (c_{G'_2},\ c_{G''_1},\ c_{G'''_1}) \\
c_{G_4} &= (c_{G'_1},\ c_{G''_2},\ c_{G'''_1}) \\
c_{G_5} &= (c_{G'_1},\ c_{G''_1},\ c_{G'''_2})
\end{aligned}
$$

With the same permutation of coordinates, w can be written as $w = (w_1\ w_2\ w_3)$. Thus we have:

- $G_3 \neq G_1$ or G_2, which implies:

$$\begin{aligned} & w \cdot c_{G_1} &=& w_1 \cdot c_{G_1'} + w_2 \cdot c_{G_1''} + w_3 \cdot c_{G_1'''} \\ > & w \cdot c_{G_3} &=& w_1 \cdot c_{G_2'} + w_2 \cdot c_{G_1''} + w_3 \cdot c_{G_1'''} \\ \Longrightarrow & w_1 \cdot c_{G_1'} &>& w_1 \cdot c_{G_2'}; \end{aligned}$$

- $G_4 \neq G_1$ or G_2, which implies:

$$\begin{aligned} & w \cdot c_{G_1} &=& w_1 \cdot c_{G_1'} + w_2 \cdot c_{G_1''} + w_3 \cdot c_{G_1'''} \\ > & w \cdot c_{G_4} &=& w_1 \cdot c_{G_1'} + w_2 \cdot c_{G_2''} + w_3 \cdot c_{G_1'''} \\ \Longrightarrow & w_2 \cdot c_{G_1''} &>& w_2 \cdot c_{G_2''}. \end{aligned}$$

There is a contradiction:

$$\begin{aligned} w \cdot c_{G_2} &= w_1 \cdot c_{G_2'} + w_2 \cdot c_{G_2''} + w_3 \cdot c_{G_2'''} \\ &< w_1 \cdot c_{G_1'} + w_2 \cdot c_{G_1''} + w_3 \cdot c_{G_2'''} \quad = w \cdot c_{G_5} \\ \Longrightarrow\ w \cdot c_{G_2} &< w \cdot c_{G_5}. \end{aligned}$$

Therefore G_1 and G_2 cannot be neighbors.

□

Remark 3.2.4. *Two graphical examples will be given for a more intuitive view of the proof of Theorem 3.2.3.*

- *Part (1), the proof of "if" statement. In Figure 3.5, $m = 4$, $n = 3$ and $b_i = b_1$.*

- *Part (2), the proof of "only if" statement. In Figure 3.6, $m = 4$, $n = 3$, $b_i = b_1$ and $b_j = b_2$.*

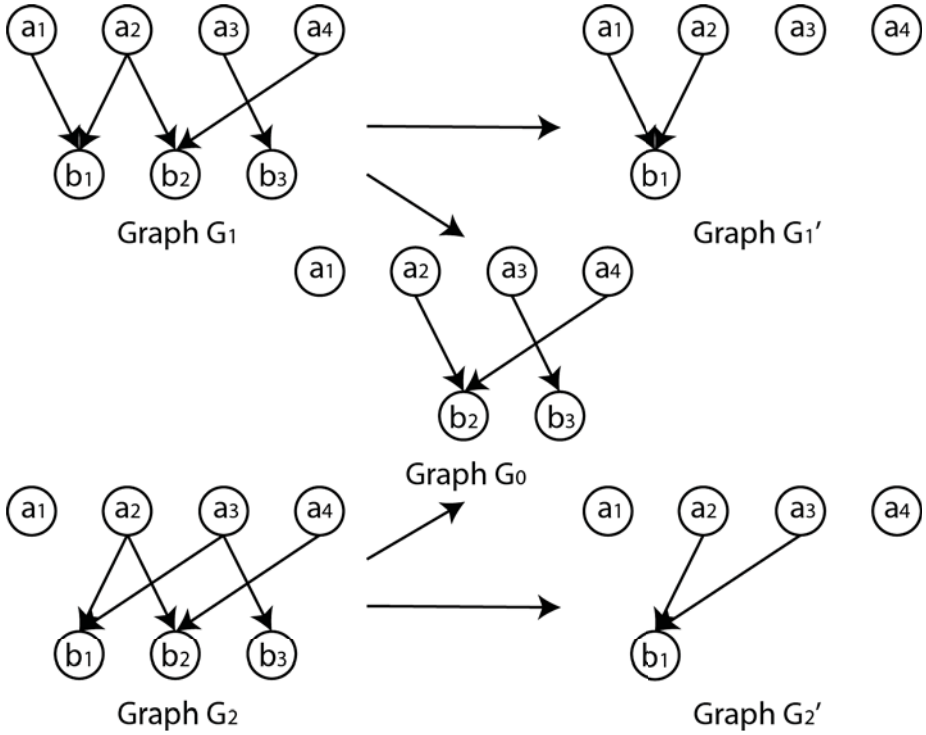


Figure 3.5: An example for the proof of Theorem 3.2.3, part (1)

3.2.2 $\mathbf{P}_{m,n}$ is a direct product of simplices

Theorem 3.2.5. *Fix m and n. For an arbitrary $G \in \mathcal{G}_{m,n}$, G has $n \cdot (2^m - 1)$ many neighbors.*

Proof. By Theorem 3.2.3, $\forall H \in \mathcal{G}_{m,n}$, G and H are neighbors if and only if: $\exists\, b_k \in B$ such that $pa_G(b_k) \neq pa_H(b_k)$ and $pa_G(b_j) = pa_H(b_j)$, $\forall\ b_j \in B$ and $b_j \neq b_k$.

Now fix $b_i \in B$. Define graphs:

- G', $H' \in \mathcal{G}_{m,1}$ with symptom $B_{m,1} = \{b_i\}$ such that $pa_{G'}(b_i) = pa_G(b_i)$ and $pa_{H'}(b_i) = pa_H(b_i)$;
- G'', $H'' \in \mathcal{G}_{m,(n-1)}$ with symptoms $B_{m,(n-1)} = B \backslash \{b_i\}$ such that $pa_{G''}(b_j) = pa_G(b_j)$ and $pa_{H''}(b_j) = pa_H(b_j)$, $\forall\ b_j \in B_{m,(n-1)}$.

Since G and H are neighbors and $G' \neq H'$ will lead to $G'' = H''$, and by Proposition

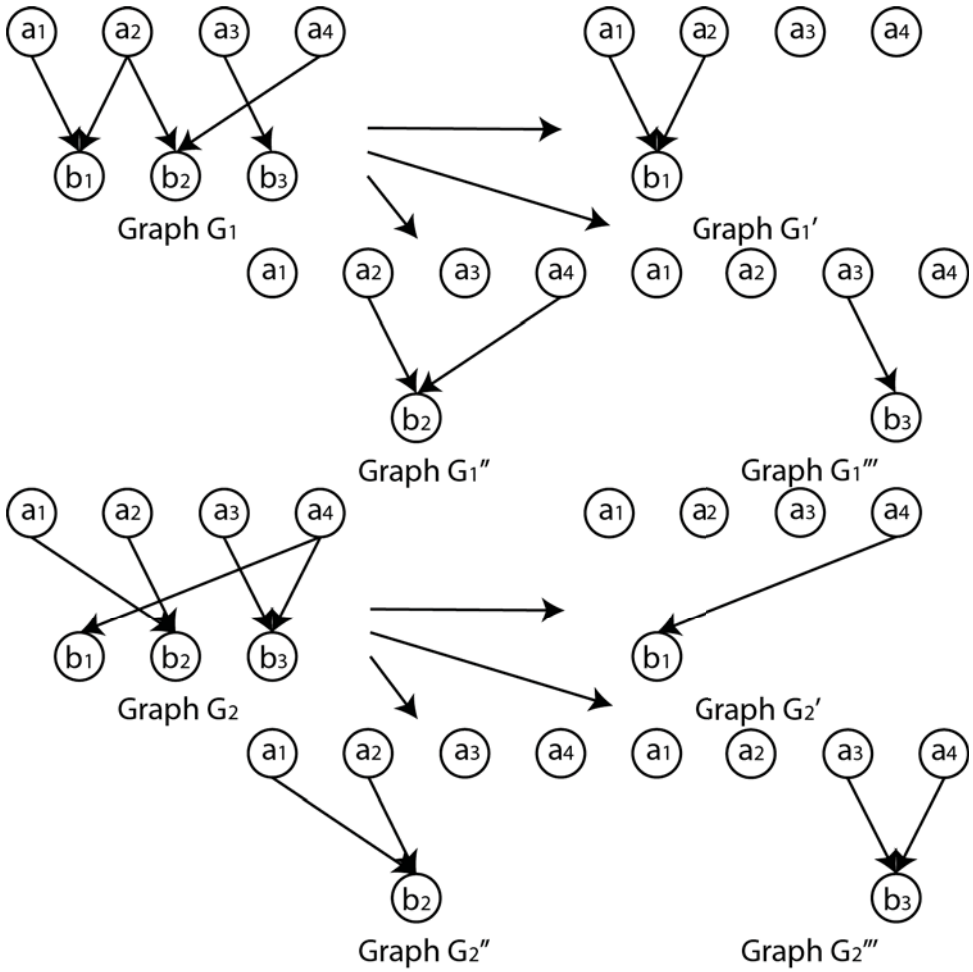


Figure 3.6: An example for the proof of Theorem 3.2.3, part (2)

3.1.5 there are 2^m graphs in $\mathcal{G}_{m,1}$, there are $2^m - 1$ different choices of H's, and each corresponds to a different neighbor of G.

We can use the same strategy for every $b_i \in B$, i.e. we can find $2^m - 1$ neighbors from each fixed $b_i \in B$. It is easy to see that these neighbors are all distinct: if H_1, H_2 are all the same with G but $pa_G(b_i) \neq pa_{H_1}(b_i)$ and $pa_G(b_j) \neq pa_{H_2}(b_j)$, where b_i, $b_j \in B$ are distinct, then this implies $pa_{H_2}(b_i) = pa_G(b_i) \neq pa_{H_1}(b_i)$, i.e. H_1 and H_2 are different. Therefore the total number of neighbors for G is: $n \cdot (2^m - 1)$. □

Remark 3.2.6. *Theorem 3.2.5 implies that every vertex of* $\mathbf{P}_{m,1}$ *has* $(2^m - 1)$ *neighbors. Since* $|vert(\mathbf{P}_{m,1})| = 2^m$ *(by Proposition 3.1.5),* $\mathbf{P}_{m,1}$ *is a simplex with dimen-*

sion $(2^m - 1)$, *i.e.* $\mathbf{P}_{m,1} = \Delta_{2^m-1}$.

Theorem 3.2.7. $\mathbf{P}_{m,n}$ *is the direct product of* n *many* Δ_{2^m-1}, *i.e.*

$$\mathbf{P}_{m,n} = \underbrace{\Delta_{2^m-1} \times \Delta_{2^m-1} \times \cdots \times \Delta_{2^m-1}}_{n \text{ many}}.$$

And the i_{th} *simplex is* $\mathbf{P}_{m,1}$ *with the same diseases* A *and only one symptom* $\{b_i\}$.

Proof. Fix m, we are going to prove the equality by induction on n.

- $n = 1$. See Remark 3.2.6;
- Fix $q \in \mathbb{Z}^+$. Suppose the equality holds for $\mathbf{P}_{m,n}$, $\forall\ n < q$, then we need to prove that it also holds for $\mathbf{P}_{m,q}$. Recall that for $\mathcal{G}_{m,q}$, the symptoms are: $B = \{b_1, b_2, \ldots, b_q\}$.

  First, we need to prove: $\mathbf{P}_{m,q} \subseteq \mathbf{P}_{m,q-1} \times \mathbf{P}_{m,1}$.

  Similarly with the proof of Theorem 3.2.3, $\forall\ G \in \mathcal{G}_{m,q}$, we define graphs:

  - $G' \in \mathcal{G}_{m,(q-1)}$ with symptoms $B_{m,(q-1)} = B \backslash \{b_q\}$ such that $pa_{G'}(b_i) = pa_G(b_i)$, $\forall\ b_i \in B_{m,(q-1)}$. This implies $c_{G'} \in \mathbf{P}_{m,q-1}$;
  - $G'' \in \mathcal{G}_{m,1}$ with symptom $B_{m,1} = \{b_q\}$ such that $pa_{G''}(b_q) = pa_G(b_q)$. This implies $c_{G''} \in \mathbf{P}_{m,1}$.

  With a proper permutation of coordinates, we can write c_G in the form of:

  $$c_G = (c_{G'}, c_{G''}).$$

  Recall that $vert(\mathbf{P}_{m,q}) = \{c_G : G \in \mathcal{G}_{m,q}\}$, so $\forall x \in \mathbf{P}_{m,q}$, with the same permutation of coordinates, we have:

  $$x = \sum_{G \in \mathcal{G}_{m,q}} \alpha_G c_G = (\sum_{G \in \mathcal{G}_{m,q}} \alpha_G c_{G'}\ ,\ \sum_{G \in \mathcal{G}_{m,q}} \alpha_G c_{G''}) \tag{3.2.1}$$

  where $0 \leq \alpha_G \leq 1$, $\forall G \in \mathcal{G}_{m,q}$ and $\sum\limits_{G \in \mathcal{G}_{m,q}} \alpha_G = 1$.

Note that $\sum_{G\in\mathcal{G}_{m,q}} \alpha_G c_{G'} \in \mathbf{P}_{m,q-1}$ and $\sum_{G\in\mathcal{G}_{m,q}} \alpha_G c_{G''} \in \mathbf{P}_{m,1}$, Equation (3.2.1) implies $x \in \mathbf{P}_{m,q-1} \times \mathbf{P}_{m,1}$. Hence:

$$\mathbf{P}_{m,q} \subseteq \mathbf{P}_{m,q-1} \times \mathbf{P}_{m,1}.$$

Second, we need to prove: $\mathbf{P}_{m,q-1} \times \mathbf{P}_{m,1} \subseteq \mathbf{P}_{m,q}$.

Let $\mathcal{G}_{m,q-1}$ has symptoms $B_{m,(q-1)} = B\backslash\{b_q\}$ and $\mathcal{G}_{m,1}$ has symptom $B_{m,1} = \{b_q\}$. $\forall\ G' \in \mathcal{G}_{m,(q-1)}$ and $G'' \in \mathcal{G}_{m,1}$, we can define $G \in \mathcal{G}_{m,q}$ such that $pa_G(b_i) = pa_{G'}(b_i)$, $\forall\ b_i \in B_{m,(q-1)}$, and $pa_G(b_q) = pa_{G''}(b_q)$. c_G has the form of $c_G = (c_{G'}, c_{G''})$.

$\forall\ x \in \mathbf{P}_{m,q-1} \times \mathbf{P}_{m,1}$, x can be written as:

$$\begin{aligned} x &= \Big(\sum_{G'\in\mathcal{G}_{m,q-1}} \beta_{G'} c_{G'}\ ,\ \sum_{G''\in\mathcal{G}_{m,1}} \gamma_{G''} c_{G''}\Big) = \sum_{G'\in\mathcal{G}_{m,q-1}} \sum_{G''\in\mathcal{G}_{m,1}} \beta_{G'}\gamma_{G''}(c_{G'}\ ,\ c_{G''}) \\ &= \sum_{G'\in\mathcal{G}_{m,q-1}} \sum_{G''\in\mathcal{G}_{m,1}} (\beta_{G'}\gamma_{G''})\ c_G\ , \end{aligned}$$

where $0 \le \beta_{G'}, \gamma_{G''} \le 1$, $\forall G' \in \mathcal{G}_{m,q-1}$, $\forall G'' \in \mathcal{G}_{m,1}$, and $\sum_{G'\in\mathcal{G}_{m,q-1}} \beta_{G'} = 1$, $\sum_{G''\in\mathcal{G}_{m,1}} \gamma_{G''} = 1$. Note that

$$\sum_{G'\in\mathcal{G}_{m,q-1}} \sum_{G''\in\mathcal{G}_{m,1}} (\beta_{G'}\gamma_{G''}) = \sum_{G'\in\mathcal{G}_{m,q-1}} \beta_{G'}\Big(\sum_{G''\in\mathcal{G}_{m,1}} \gamma_{G''}\Big) = \sum_{G'\in\mathcal{G}_{m,q-1}} \beta_{G'} = 1,$$

which leads to $x \in \mathbf{P}_{m,q}$. Hence:

$$\mathbf{P}_{m,q-1} \times \mathbf{P}_{m,1} \subseteq \mathbf{P}_{m,q}.$$

Therefore,

$$\mathbf{P}_{m,q} = \mathbf{P}_{m,q-1}\times\mathbf{P}_{m,1} = \underbrace{\Delta_{2^m-1} \times \cdots \times \Delta_{2^m-1}}_{\text{q-1 many}} \times \Delta_{2^m-1} = \underbrace{\Delta_{2^m-1} \times \cdots \times \Delta_{2^m-1}}_{\text{q many}}.$$

□

Theorem 3.2.7 implies that $\mathbf{P}_{m,n}$ is a simple polytope with dimension $n \cdot (2^m - 1)$. In Appendix, we will give another proof which use linear algebra to show that $\mathbf{P}_{m,n}$ is simple and obtain its dimension.

3.2.3 Expression of facets of $\mathbf{P}_{m,n}$

Based on Theorem 3.2.7, we are going to show the expression of facets of $\mathbf{P}_{m,n}$ using the following lemma:

Lemma 3.2.8. *[59] Suppose* $\mathbf{P}$ *is the direct product of simplices* $\Delta_{\alpha_1}, \ldots, \Delta_{\alpha_k}$*. Then every facet of* $\mathbf{P}$ *has the form of* $\Delta_{\alpha_1} \times \ldots \times \Delta_{\alpha_{i-1}} \times F_{\alpha_i} \times \Delta_{\alpha_{i+1}} \times \ldots \times \Delta_{\alpha_k}$*, where* F_{α_i} *is a facet of* Δ_{α_i}*.*

Remark 3.2.9. *Lemma 3.2.8 implies that in order to study the facets of a direct product of simplices, we can simply study the facets of each simplex. As by Theorem 3.2.7,* $\mathbf{P}_{m,n}$ *is a direct product of* n *many* $\mathbf{P}_{m,1}$*, our problem is simplified as studying the facets of* $\mathbf{P}_{m,1}$*. Thus we assume* $B = \{b_1\}$ *in the following content of this section.*

Assume $A = \{a_1, \ldots, a_m\}$ and $B = \{b_1\}$. By Proposition 3.1.6, the vertices of $\mathbf{P}_{m,1}$ has at most $2^m - 1$ many non-zero coordinates. We define the indeterminates, i.e. variables, $\{x_s,\ s \subseteq A,\ s \neq \emptyset\}$, where one indeterminate x_s for each coordinate $c_G(s \cup \{b_1\})$ in the characteristic imset c_G, $G \in \mathcal{G}_{m,1}$. Define the vector of indeterminates $x = \{x_s,\ s \subseteq A,\ s \neq \emptyset\}$. Suppose $A_m x \leq b_m$ is the system of inequalities that defines $\mathbf{P}_{m,1}$. We can define a $2^m \times 2^m$ matrix: $D_m = [b_m| - A_m]$. Denote the elements in D_m by $(d_{st})_{s \subseteq A, t \subseteq A}$ so that we can rewrite the system of inequalities as: $d_{s\emptyset} + \sum_{t \subseteq A, t \neq \emptyset} d_{st} x_t \geq 0,\ s \subseteq A$. Then we have the expression of 2^m facets of $\mathbf{P}_{m,1}$ as following:

$$F_s = \mathbf{P}_{m,1} \cap \{x : d_{s\emptyset} + \sum_{t \subseteq A, t \neq \emptyset} d_{st} x_t = 0\},\ s \subseteq A,$$

where the elements d_{st}, $s, t \subseteq A$ can be obtained using Theorem 3.2.10.

Theorem 3.2.10. *The elements in matrix* D_m *satisfies:*

- $d_{st} \neq 0$ *if and only if* $s \subseteq t$*;*
- *if* $s \subseteq t$*, then* $d_{st} = (-1)^{|t|-|s|}$*.*

This implies that $\mathbf{P}_{m,1}$ *has* 2^m *facets:*

$$F_s = \mathbf{P}_{m,1} \cap \{x : d_{s\emptyset} + \sum_{t\subseteq A, t\neq\emptyset} d_{st}x_t = 0\},\ s \subseteq A.$$

What's more, $\forall\ s \subseteq A$, $vert(\mathbf{P}_{m,1})\backslash\{c_{G_s}\} \subset F_s$, *where* $pa_{G_s}(b_1) = s$.

Proof. For convenience, let $x_\emptyset \equiv 1$. $\forall s \subseteq A$, let $d_{s\cdot} = (d_{st})_{t\subseteq A}$ be the corresponding row of D_m, and G_s be the graph in $\mathcal{G}_{m,1}$ such that $pa_{G_s}(b_1) = s$. Now we can rewrite the system of inequalities as:

$$\sum_{t\subseteq A} d_{st}x_t = d_{s\cdot}(1\ x)^T \geq 0,\ \text{for}\ \forall\ s \subseteq A.$$

We are going to prove that $\forall s \subseteq A$, we can find $2^m - 1$ vertices on F_s that are linearly independent, and this implies that F_s is a facet of $\mathbf{P}_{m,1}$. In fact, we will prove that: $\{c_{G_{s'}},\ s' \subseteq A,\ s' \neq s\} \subset F_s$ and $c_{G_s} \notin F_s$, i.e. $d_{s\cdot}(1\ c_{G_{s'}})^T = 0,\ \forall\ s' \subseteq A,\ s' \neq s$ and $d_{s\cdot}(1\ c_{G_s})^T > 0$.

Notice that $\forall t \subseteq A$, $c_{G_{s'}}(t \cup \{b_1\}) \neq 0$ if and only if $t \subseteq pa_{c_{G_{s'}}}(b_1) = s'$, and $d_{st} \neq 0$ if and only if $s \subseteq t$. So:

$$d_{s\cdot}(1\ c_{G_{s'}})^T = d_{s\emptyset} + \sum_{t\subseteq A,\ t\neq\emptyset} d_{st}c_{G_{s'}}(t \cup \{b_1\}) = d_{s\emptyset} + \sum_{s\subseteq t\subseteq s',\ t\neq\emptyset} d_{st} = \sum_{s\subseteq t\subseteq s'} d_{st}.$$

- If $s = s'$, then $d_{s\cdot}(1\ c_{G_{s'}})^T = d_{ss} = 1 > 0$;
- If $s \subsetneq s'$, then $d_{s\cdot}(1\ c_{G_{s'}})^T = \sum_{s\subseteq t\subseteq s'} (-1)^{|t|-|s|} = \sum_{t'\subseteq s'\backslash s} (-1)^{|t'|} = 0$;
- If $s \not\subseteq s'$, then $d_{s\cdot}(1\ c_{G_{s'}})^T = 0$.

□

Example 3.2.11 (Facets of $\mathbf{P}_{2,1}$)**.** *Notation adopted from Theorem 3.2.10. Fix* $m = 2$

and $n = 1$. All characteristic imsets are given as a matrix:

$$\begin{pmatrix} c_{G_0} \\ c_{G_1} \\ c_{G_2} \\ c_{G_{12}} \end{pmatrix} = \begin{array}{c} T \\ \\ \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{ccc} a_1b_1 & a_2b_1 & a_1a_2b_1 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$

The matrix $D_2 = [b_2| - A_2]$:

$$D_2 = \begin{array}{c} s\backslash t \\ \emptyset \\ a_1 \\ a_2 \\ a_1a_2 \end{array} \begin{array}{c} \begin{array}{cccc} \emptyset & a_1 & a_2 & a_1a_2 \end{array} \\ \begin{pmatrix} 1 & -1 & -1 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

The system of inequalities that defines $\mathbf{P}_{2,1}$:

$$\begin{array}{c} s\backslash t \\ \emptyset \\ a_1 \\ a_2 \\ a_1a_2 \end{array} \begin{array}{c} \begin{array}{ccccc} \emptyset & a_1 & a_2 & a_1a_2 & \end{array} \\ \begin{pmatrix} 1 & -x_{a_1} & -x_{a_2} & +x_{a_1a_2} & \geq 0 \\ & x_{a_1} & & -x_{a_1a_2} & \geq 0 \\ & & x_{a_2} & -x_{a_1a_2} & \geq 0 \\ & & & x_{a_1a_2} & \geq 0 \end{pmatrix} \end{array}$$

Vertices c_{G_0}, c_{G_1} and $c_{G_{12}}$ are in the facet F_{a_2} while c_{G_2} is not (see Figure 3.7).

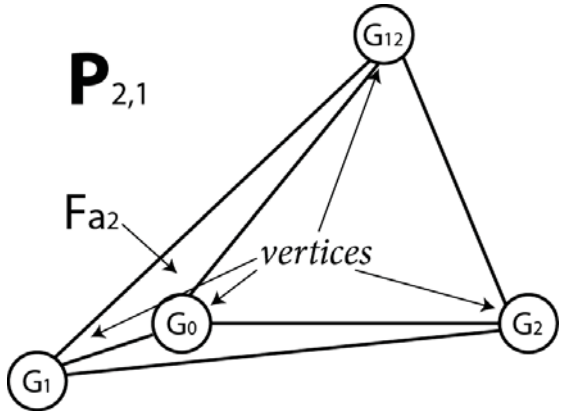


Figure 3.7: The facets and vertices of $\mathbf{P}_{2,1}$

3.3 The characteristic imset polytopes (cim-polytopes) for Bayesian networks

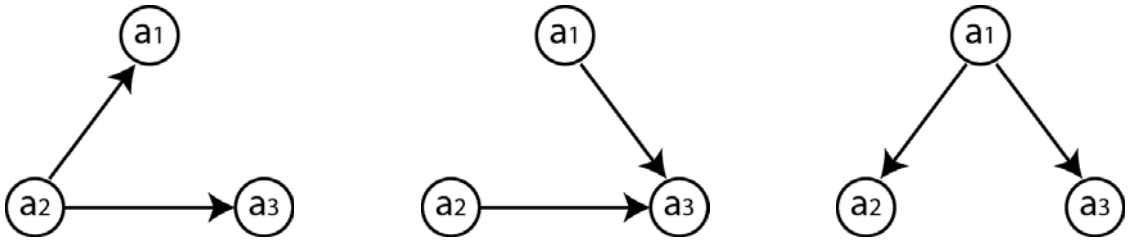
The results in Section 3.2 are limited to diagnosis models. In this section, we will generalize the results to all Bayesian networks with the same underlying order. Similarly with Section 3.2, we will also give the combinatorial description of edges on the cim-polytopes, and prove that these cim-polytopes are also direct produce of simplices. The expression of facets of these cim-polytopes can be obtained, too.

For a set of random variables $N = \{a_1, \ldots, a_n\}$, where now n is the total number of nodes in N. According to Remark 1.2.11, $\forall G \in DAGs(N)$, there exists an **underlying ordering** over N, $[n]_G = (a_{[1]}, \ldots, a_{[n]})$, such that if $[a_{[i]}, a_{[j]}]$, $i < j$, is an edge in G, then $a_{[i]} \to a_{[j]}$ in G. In this section, we will focus on the class of graphs which share a specific underlying ordering $[n]$, i.e. $\mathcal{G}_{[n]} = \{G \in DAGs(N) : [n]_G = [n]\}$, and its characteristic imset polytope $\mathbf{P}_{[n]} = \mathbf{P}_{\mathcal{G}_{[n]},c}$.

Example 3.3.1 (Underlying ordering of graphs)**.** *Let $N = \{a_1, a_2, a_3\}$. Consider an ordering over N, $[n] = (a_2, a_1, a_3)$, i.e. $a_{[1]} = a_2$, $a_{[2]} = a_1$ and $a_{[3]} = a_3$. Then $\forall G \in \mathcal{G}_{[n]}$, the only type of directed edges allowed in G are $a_{[i]} \to a_{[j]}$, where $i < j$. For instance, $a_2 \to a_1$ is allowed while $a_1 \to a_2$ is not. Thus graph G_1 in Figure 3.8(a) and graph G_2 in Figure 3.8(b) are both in $\mathcal{G}_{[n]}$. Graph G_3 in Figure 3.8(c) is not in $\mathcal{G}_{[n]}$ since it has arrow $a_1 \to a_2$, and the underlying ordering for G_3, i.e. $[n]_{G_3}$, can either be (a_1, a_2, a_3) or (a_1, a_3, a_2).*

Remark 3.3.2. *For a specific ordering $[n]$ and an arbitrary $G \in \mathcal{G}_{[n]}$, we have the following proposition that is similar with Proposition 3.1.3.*

- *$\forall T \subseteq N$, $|T| = k \geq 2$, we can order the elements in T according to $[n]$ and write T in the form of $a_{[i_1]}a_{[i_2]}\ldots a_{[i_k]}$ where $i_1 < i_2 < \cdots < i_k$. Then $c_G(T) = \prod_{s=i_1,\ldots,i_{k-1}} c_G(a_{[s]}a_{[i_k]})$. This property means that the whole c_G is determined*



(a) G_1 (b) G_2 (c) G_3

Figure 3.8: Three graphs to illustrate the underlying ordering of graphs

by $\binom{n}{2}$ *coordinates,* $\{c_G(a_{[i]}a_{[j]}),\ i < j\}$*, which can also be interpreted as the existence of the directed edges* $a_{[i]} \to a_{[j]}$*,* $i < j$*.*

Another way to see this property is that $\forall\ G \in \mathcal{G}_{[n]}$*,* G *can be determined by* $pa_G(a_{[i]})$*,* $i = 2, \ldots, n$ *since* $pa_G(a_{[1]}) = \emptyset$*. Similarly with Remark 3.1.4, we can consider a permutation of coordinates in* c_G *that corresponds to a permutation of* T*, then these coordinates can be broken into* $n-1$ *parts:*

$$\underline{(12)}, \underline{(13), (23), (123)}, \underline{(14), (24), \ldots, (1234)}, \ldots, \underline{(1n), (2n), \ldots, ((n-1)n), \ldots, (12\ldots n)}$$

where $(i_1 \ldots i_k)$ *stands for* $T = a_{[i_1]}a_{[i_2]} \ldots a_{[i_k]}$*,* $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$*. The* k*-th part of the coordinations,* $\{c_G(T)$*:* $a_{[j]} \notin T,\ \forall j > k\}$ *only depend on* $pa_G(a_{[k]})$*, and different parts are completely irrelevant in the sense that* $pa_G(a_{[k]})$*,* $a_{[k]} \in N$*, can be decided separately.*

Theorem 3.3.3. *Suppose* $n \geq 2$*.* $\mathbf{P}_{[n]}$ *is a direct product of a sequence of simplices:*

$$\mathbf{P}_{[n]} = \underbrace{\Delta_{2^1-1} \times \Delta_{2^2-1} \times \cdots \times \Delta_{2^{n-1}-1}}_{\text{n-1 simplices}},$$

where the i_{th} *simplex* Δ_{2^i-1} *is the same with the cim-polytope for diagnosis models,* $\mathbf{P}_{i,1}$*, with diseases* $A = \{a_{[1]}, \ldots, a_{[i]}\}$ *and one symptom* $\{a_{[i+1]}\}$*.*

Proof. We are going to prove the equality by induction on n. Since $n \geq 2$, we start the induction from $n = 2$.

- $n = 2$. It is obvious since there are only two vertices in $\mathbf{P}_{[n]}$: (1) and (0). So $\mathbf{P}_{[n]}$ is a line segment which is a simplex of dimension 1, i.e. $\mathbf{P}_{[n]} = \Delta_1$.

- Fix $q \in \mathbb{Z}_+$. Suppose the equality holds for $\mathbf{P}_{[n]}$, $\forall\ n < q$, and we need to prove that it also holds for $\mathbf{P}_{[q]}$. Define notation $N_{[k]} = \{a_{[1]}, \ldots, a_{[k]}\}$ for $k = 1, \ldots, q$.

  First, we want to prove: $\mathbf{P}_{[q]} \subseteq \mathbf{P}_{[q-1]} \times \Delta_{2^{q-1}-1}$.

  $\forall\ G \in \mathcal{G}_{[q]}$, we can define graphs:

  - G' is the induced subgraph of G for $N_{[q-1]}$, which implies $c_{G'} \in \mathbf{P}_{[q-1]}$;
  - G'' is a graph over N such that the only edges in G'' are $a_{[i]} \to a_{[q]}$, where $a_{[i]} \in pa_G(a_{[q]})$. Consider a diagnosis model where $N_{[q-1]}$ is the set of diseases and $a_{[q]}$ is the symptom, then we can see that $c_{G''} \in \mathbf{P}_{q-1,1} = \Delta_{2^{q-1}-1}$.

  Now, with a proper permutation of coordinates (see Remark 3.3.2), we can write c_G in the form of:

$$c_G = (c_{G'} \quad c_{G''}).$$

  Since $vert(\mathbf{P}_{[q]}) = \{c_G : G \in \mathcal{G}_{[q]}\}$, $\forall\ x \in \mathbf{P}_{[q]}$, with the same permutation of coordinates, we have:

$$x = \sum_{G \in \mathcal{G}_{[q]}} \alpha_G c_G = (\sum_{G \in \mathcal{G}_{[q]}} \alpha_G c_{G'}\ ,\ \sum_{G \in \mathcal{G}_{[q]}} \alpha_G c_{G''}), \tag{3.3.1}$$

  where $0 \leq \alpha_G \leq 1$, $\forall\ G \in \mathcal{G}_{[q]}$ and $\sum\limits_{G \in \mathcal{G}_{[q]}} \alpha_G = 1$.

  Notice that $\sum\limits_{G \in \mathcal{G}_{[q]}} \alpha_G c_{G'} \in \mathbf{P}_{[q-1]}$ and $\sum\limits_{G \in \mathcal{G}_{[q]}} \alpha_G c_{G''} \in \Delta_{2^{q-1}-1}$, Equation (3.3.1) implies $x \in \mathbf{P}_{[q-1]} \times \Delta_{2^{q-1}-1}$. Hence:

$$\mathbf{P}_{[q]} \subseteq \mathbf{P}_{[q-1]} \times \Delta_{2^{q-1}-1}.$$

  Second, we want to prove: $\mathbf{P}_{[q-1]} \times \Delta_{2^{q-1}-1} \subseteq \mathbf{P}_{[q]}$.

Let $\mathcal{G}_{[q-1]}$ has nodes $N_{[q-1]}$, and $\mathcal{G}_{q-1,1}$ has diseases $N_{[q-1]}$ and symptom $a_{[q]}$. $\forall\ G' \in \mathcal{G}_{[q-1]}$ and $G'' \in \mathcal{G}_{q-1,1}$, we can define $G \in \mathcal{G}_{[q]}$ by extending G' as following: add a node $a_{[q]}$ and edges $(a_{[i]}, a_{[q]})$, $\forall a_{[i]} \in pa_{G''}(a_{[q]})$, to G'. We can write c_G in the form of $c_G = (c_{G'}\ \ c_{G''})$.

$\forall x \in \mathbf{P}_{[q-1]} \times \Delta_{2^{q-1}-1}$, x can be written as:

$$\begin{aligned} x &= (\sum_{G' \in \mathcal{G}_{[q-1]}} \beta_{G'} c_{G'}\ ,\ \sum_{G'' \in \mathcal{G}_{q-1,1}} \gamma_{G''} c_{G''}) = \sum_{G' \in \mathcal{G}_{[q-1]}} \sum_{G'' \in \mathcal{G}_{q-1,1}} \beta_{G'} \gamma_{G''} (c_{G'}\ ,\ c_{G''}) \\ &= \sum_{G' \in \mathcal{G}_{[q-1]}} \sum_{G'' \in \mathcal{G}_{q-1,1}} (\beta_{G'} \gamma_{G''})\ c_G\ \ , \end{aligned}$$

where $0 \leq \beta_{G'},\ \gamma_{G''} \leq 1$, $\forall G' \in \mathcal{G}_{[q-1]}$, $\forall G'' \in \mathcal{G}_{q-1,1}$, and $\sum_{G' \in \mathcal{G}_{[q-1]}} \beta_{G'} = 1$, $\sum_{G'' \in \mathcal{G}_{q-1,1}} \gamma_{G''} = 1$.

Notice that

$$\sum_{G' \in \mathcal{G}_{[q-1]}} \sum_{G'' \in \mathcal{G}_{q-1,1}} (\beta_{G'} \gamma_{G''}) = \sum_{G' \in \mathcal{G}_{[q-1]}} \beta_{G'} (\sum_{G'' \in \mathcal{G}_{q-1,1}} \gamma_{G''}) = \sum_{G' \in \mathcal{G}_{[q-1]}} \beta_{G'} = 1.$$

This leads to $x \in \mathbf{P}_{[q]}$. Hence:

$$\mathbf{P}_{[q-1]} \times \Delta_{2^{q-1}-1} = \mathbf{P}_{[q-1]} \times \mathbf{P}_{q,1} \subseteq \mathbf{P}_{[q]}.$$

By induction on n, we finish the proof by:

$$\mathbf{P}_{[q]} = \mathbf{P}_{[q-1]} \times \mathbf{P}_{q-1,1} = (\Delta_{2^1-1} \times \cdots \times \Delta_{2^{q-2}-1}) \times \Delta_{2^{q-1}-1} = \Delta_{2^1-1} \times \cdots \times \Delta_{2^{q-1}-1}.$$

□

Remark 3.3.4. *Two immediate results from Theorem 3.3.3 are:*

- *the dimension of* $\mathbf{P}_{[n]}$ *is* $2^n - (n+1)$*, and it is a simple polytope;*
- *the expression of facets of* $\mathbf{P}_{[n]}$ *can be obtained by Lemma 3.2.8 and Theorem 3.2.10.*

Remark 3.3.5. *Note that the equality in Theorem 3.3.3 is actually* $\mathbf{P}_{[n]} = \Delta_{2^0-1} \times \Delta_{2^1-1} \times \Delta_{2^2-1} \times \cdots \times \Delta_{2^{n-1}-1}$, *where* Δ_{2^0-1} *is omitted as it has dimension 0 (a point). Theorem 3.3.3 and its proof also imply that* $\forall x \in \mathbf{P}_{[n]}$, $x \in vert(\mathbf{P}_{[n]})$ *if and only if with the permutation of coordinates in Remark 3.3.2,* x *can be written in the form of* $x = (v_1, v_2, \ldots, v_{n-1})$, *where* v_i *is the vertex of* Δ_{2^i-1}, $i = 1, \ldots, n-1$. *Suppose* $x = c_G$, $G \in \mathcal{G}_{[n]}$, *then* $v_i = c_{G_i}$, *where* G_i *is in* $\mathcal{G}_{i,1}$ *with diseases* $N_{[i]}$ *and symptom* $a_{[i+1]}$, $i = 1, \ldots, n-1$, *and* $pa_{G_i}(a_{[i+1]}) = pa_G(a_{[i+1]})$.

The following theorem will be stated in two forms which are equivalent by Theorem 3.3.3 and Lemma 3.2.2.

Theorem 3.3.6. *Fix an underlying ordering* $[n]$ *over* N.

- *(From the view of graph theory.) Two graphs,* G_1, $G_2 \in \mathcal{G}_{[n]}$ *are neighbors in* $\mathcal{G}_{[n]}$ *if and only if:* $\exists\ a_{[i]} \in N$ *such that* $pa_{G_1}(a_{[i]}) \neq pa_{G_2}(a_{[i]})$ *and* $pa_{G_1}(a_{[j]}) = pa_{G_2}(a_{[j]})$, $\forall\ a_{[j]} \in N$ *and* $a_{[j]} \neq a_{[i]}$, *i.e., all nodes but one have exactly the same parent sets in both* G_1 *and* G_2.

- *(From the view of polyhedral geometry.)* $\forall\ \mathbf{x} \in \mathbf{P}_{[n]}$, $\mathbf{x}$ *is on an edge of* $\mathbf{P}_{[n]}$ *if and only if with the permutation of coordinates showed in Remark 3.3.2* $\mathbf{x}$ *can be written in the form of* $\mathbf{x} = (v_1, \ldots, v_{i-1}, e_i, v_{i+1}, \ldots, v_{n-1})$, *where* e_i *belongs to an edge on* Δ_{2^i-1}, $i \in \{1, \ldots, n-1\}$, *and* $v_j \in vert(\Delta_{2^j-1})$, $j \in \{1, \ldots, n-1\} \backslash \{i\}$.

Proof. The proof from the view of graph theory will be very similar with the proof of Theorem 3.2.3, so we are going to give a proof from the view of polyhedral geometry, i.e. prove that: "$\exists$ vertices of v^1, $v^2 \in \mathbf{P}_{[n]}$ such that $\mathbf{x} = \beta v^1 + (1-\beta)v^2$ where $0 \leq \beta \leq 1$, and v^1, v^2 form an edge in $\mathbf{P}_{[n]}$" if and only if "$\mathbf{x}$ can be written in the form of $\mathbf{x} = (v_1, \ldots, v_{i-1}, e_i, v_{i+1}, \ldots, v_{n-1})$, $i \in \{1, \ldots, n-1\}$".

We will prove "if" and "only if" separately.

(1) Prove "if" part.

Suppose $\mathbf{x}$ has the form $\mathbf{x} = (v_1, \ldots, v_{i-1}, e_i, v_{i+1}, \ldots, v_{n-1})$.

Since e_i belongs to an edge on Δ_{2^i-1}, we can find two vertices v_i^1, $v_i^2 \in \Delta_{2^i-1}$ which form this edge, and this implies $e_i = \beta v_i^1 + (1 - \beta v_i^2)$, $0 \leq \beta \leq 1$. Suppose the cost vector for this edge is w_i^e, then for any $v_i^3 \in vert(\Delta_{2^i-1})$, $w_i^e v_i^3 \leq w_i^e v_i^1 = w_i^e v_i^2$, where "=" holds if and only if $v_i^3 = v_i^1$ or $v_i^3 = v_i^2$.

We can also find w_j^v which is a cost vector for vertex v_j in Δ_{2^j-1}, $j \in \{1, \ldots, n-1\}\backslash\{i\}$. Still, we have: $\forall v_j^3 \in vert(\Delta_{2^j-1})$, $w_j^v v_j^3 \leq w_j^v v_j$, where "=" holds if and only if $v_j^3 = v_j$.

Now let $v^1 = (v_1, \ldots, v_{i-1}, v_i^1, v_{i+1}, \ldots, v_{n-1})$, $v^2 = (v_1, \ldots, v_{i-1}, v_i^2, v_{i+1}, \ldots, v_{n-1})$ and $w = (w_1^v, \ldots, w_{i-1}^v, w_i^e, w_{i+1}^v, \ldots, w_{n-1}^v)$. Obviously $\mathbf{x} = \beta v^1 + (1 - \beta)v^2$, where $0 \leq \beta \leq 1$. In addition, $\forall v^3 = (v_1^3, \ldots, v_{n-1}^3) \in vert(\mathbf{P}_{[n]})$, we have:

$$\begin{aligned} wv^3 = w_i^e v_i^3 + \sum_{j=1,\ j\neq i}^{n-1} w_j^v v_j^3 \ &\leq\ w_i^e v_i^1 + \sum_{j=1,\ j\neq i}^{n-1} w_j^v v_j \ =\ wv^1 \\ &=\ w_i^e v_i^2 + \sum_{j=1,\ j\neq i}^{n-1} w_j^v v_j \ =\ wv^2, \end{aligned}$$

where "=" holds if and only if $v^3 = v^1$ or $v^3 = v^2$, i.e. v^1 and v^2 form an edge on $\mathbf{P}_{[n]}$.

(2) Prove "only if" part.

Suppose $\exists\ v^1 = (v_1^1, \ldots v_{n-1}^1)$, $v^2 = (v_1^2, \ldots v_{n-1}^2) \in vert(\mathbf{P}_{[n]})$ such that $\mathbf{x} = \beta v^1 + (1-\beta)v^2$ where $0 \leq \beta \leq 1$, and v^1, v^2 form an edge in $\mathbf{P}_{[n]}$. If we can prove that $\exists\, i \in \{1, \ldots, n-1\}$ such that $v_i^1 \neq v_i^2$ and $v_j^1 = v_j^2$, $\forall\ j \in \{1, \ldots, n-1\}\backslash\{i\}$, then $\mathbf{x}$ has the form $\mathbf{x} = (v_1, \ldots, v_{i-1}, e_i, v_{i+1}, \ldots, v_{n-1})$, where e_i is on the edge of Δ_{2^i-1} formed by v_i^1 and v_i^2. We are going to prove this statement by contradiction.

Suppose $\exists\ i, j \in \{1, \ldots, n-1\}$ distinct such that $v_i^1 \neq v_i^2$ and $v_j^1 \neq v_j^2$, but v^1 and v^2 still form an edge on $\mathbf{P}_{[n]}$. Let $w = (w_1, \ldots, w_{n-1})$ be the cost vector for

this edge, i.e. $\forall v^3 = (v_1^3, \ldots v_{n-1}^3) \in vert(\mathbf{P}_{[n]})$, $wv^3 \leq wv^1 = wv^2$ where "=" holds if and only if $v^3 = v^1$ or $v^3 = v^2$.

- If we set v^3 as following: $v_i^3 = v_i^2$, $v_k^3 = v_k^1$, $\forall\ k \in \{1, \ldots, n-1\}\backslash\{i\}$. Obviously $v^3 \neq v^1$ and $v^3 \neq v^2$. Thus:

$$\begin{aligned} wv^3 = w_i v_i^2 + \sum_{k=1,\ k\neq i}^{n-1} w_k v_k^1 \ &< \ wv^1 = \sum_{k=1}^{n-1} w_k v_k^1 = w_i v_i^1 + \sum_{k=1,\ k\neq i}^{n-1} w_k v_k^1 \\ \Longrightarrow \qquad w_i v_i^2 \ &< \ w_i v_i^1. \end{aligned}$$

- If we set v^3 as following: $v_j^3 = v_j^2$, $v_k^3 = v_k^1$, $\forall\ k \in \{1, \ldots, n-1\}\backslash\{j\}$. Obviously $v^3 \neq v^1$ and $v^3 \neq v^2$. Thus:

$$\begin{aligned} wv^3 = w_j v_j^2 + \sum_{k=1,\ k\neq j}^{n-1} w_k v_k^1 \ &< \ wv^1 = \sum_{k=1}^{n-1} w_k v_k^1 = w_j v_j^1 + \sum_{k=1,\ k\neq j}^{n-1} w_k v_k^1 \\ \Longrightarrow \qquad w_j v_j^2 \ &< \ w_j v_j^1. \end{aligned}$$

Now we set v^3 as following: $v_i^3 = v_i^1$, $v_j^3 = v_j^1$, $v_k^3 = v_k^2$, $\forall\, k \in \{1, \ldots, n-1\}\backslash\{i, j\}$. Then we have:

$$wv^3 = w_i v_i^1 + w_j v_j^1 + \sum_{k=1,\ k\neq i,j}^{n-1} w_k v_k^2 > w_i v_i^2 + w_j v_j^2 + \sum_{k=1,\ k\neq i,j}^{n-1} w_k v_k^2 = \sum_{k=1}^{n-1} w_k v_k^2 = wv^2,$$

i.e. $wv^3 > wv^2$, which is a contradiction with our assumption.

□

Chapter 4 Discussion and Future Work

4.1 Discussion and future work for SIS for zero-one three-way tables with fixed two-way marginals

4.1.1 Discussion on the computational results in Chapter 2

In this dissertation we do not have a sufficient and necessary condition for the existence of the three-way zero-one table so we cannot avoid rejection. However, since the SIS procedure gives an unbiased estimator, we may only need a small sample size as long as it converges. For example, the sample size is fixed to be 1000 in Table 2.1 since most estimators (the column "Estimation" in Table 2.1 in Table 2.1, i.e. $\widehat{|\Sigma|}$) are exactly the same as the true numbers of tables (the column "# tables", i.e. Σ). Also note that the acceptance rate does not depend on a sample size. Thus, it would be interesting to investigate the convergence rate of the SIS procedure with CP for zero-one three-way tables.

It seems that the convergence rate is slower when we have a "large" table, where "large" means in terms of $|\Sigma|$ rather than its dimension, i.e., the number of cells. A large value of $\widehat{|\Sigma|}$ usually corresponds to a larger cv^2, and this often comes with large variations of $\widehat{|\Sigma|}$ and cv^2, i.e. $\widehat{|\Sigma|}$ and cv^2 obtained from different iterations can vary much. For example, we ran six iterations for the $8 \times 8 \times 8$ semimagic cube with all two-way marginals equal to 3 (see Table 2.3 for Example 2.4.17): three iterations of 1000 and three iterations of 5000. The results for the former are: $\widehat{|\Sigma|} = 3.24e{+}59$ with $cv^2 = 7.05$; $\widehat{|\Sigma|} = 2.90e + 59$ with $cv^2 = 9.05$; and $\widehat{|\Sigma|} = 3.88e + 59$ with $cv^2 = 55.59$. The results for the latter are: $\widehat{|\Sigma|} = 3.36e + 59$ with $cv^2 = 25.88$; $\widehat{|\Sigma|} = 3.39e + 59$ with $cv^2 = 18.64$; and $\widehat{|\Sigma|} = 4.92e + 59$ with $cv^2 = 461.60$. We can see that: 1) in general, a large cv^2 would most possibly point to an unreliable estimator; and

2) cv^2 is not necessarily smaller when the sample size increases, but with a larger sample, the estimator $\widehat{|\Sigma|}$ will be more stable if cv^2 is not inflated compared with other iterations with the same sample size. Although we have the issue of large cv^2 when $|\Sigma|$ is large, fortunately, the estimation of number of tables seems to be still reliable and the computational time seems to be still reasonable if the acceptance rate is still high. Thus, for a fixed sample size, when one finds a large $\widehat{|\Sigma|}$ or a large cv^2 (especially a large cv^2), we recommend to apply several iterations and pick the one with a relatively small cv^2 (we do not necessarily choose the one with the smallest cv^2 because a small improvement in cv^2 does not necessarily mean a better estimator (see Example 2.4.10)). Take the three iterations of 1000 for example. We first exclude the one with $cv^2 = 55.59$ since this cv^2 is too large compared with the other two. Then we can choose either result from the rest two iterations. For reference, Table 2.4 gives the bootstrap-t confidence intervals (see details in Appendix) for semimagic cubes with $m = n = l = 7, \ldots, 10$ in Example 2.4.17. Bootstrap-t confidence intervals will be more useful if cv^2 is not very small, but if cv^2 is too large, then another iteration with a smaller cv^2 will preferable to produce a more informative and reliable confidence interval.

For the experiment with Sampson's data set, we observed a very low acceptance rate compared with experimental studies on simulated data sets. We investigated why this happens and found two possible reasons: first, it seems that our sampling works better when the success rates of cells are balanced, i.e. $P(X_{ijk} = 1), \forall i, j, k$, are close to each other; second, a bigger table size might be unfavorable for acceptance rate. Simulations show that the acceptance rates can be very low when we have a large table with unbalanced success rates of cells: a simulation of a $10 \times 10 \times 10$ table with unbalanced success rates of cells has acceptance rate only 40%, and it decreases to only 1% for a $18 \times 18 \times 10$ table. On the other hand, a large cv^2, which reflects a large variation in $\widehat{|\Sigma|}$, can also cause problem. We noticed that among the

1000 sampled tables, there are a few of them with extremely small probabilities that resemble outliers and may cause a large cv^2. These "outliers" can make the results very unstable: Table 1 in Appendix gives the results with and without 7 "outliers", we can see that the cv^2 without the "outliers" is much smaller than the one with "outliers".

4.1.2 Open problems and future work on SIS procedures

1. The trial distribution $q(\cdot)$ for sampling contingency tables is designed to approximate the target distribution $p(\cdot)$. Setting the target distribution to be the uniform distribution performs much better than hypergeometric distribution in estimating the total number of tables, while setting the target distribution to be the hypergeometric distribution is more preferable in goodness-of-fit tests (see Section 1.1.1). However, in general, sampling according to a hypergeometric distribution is more difficult than according to a uniform distribution because the marginal distributions for the hypergeometric distribution are not trivial except in very small examples. In [10], they proposed a "hypergeometric sampling method", in which the marginal distribution $q(x_i|x_{i-1}, \ldots, x_1)$ is assumed to be the hypergeometric distribution over $[l_i, u_i]$, where l_i and u_i are the lower and upper bounds of the support of the marginal distribution, $i = 1, \ldots, t$. This method gives a reasonable marginal approximation and works nicely for some non-sparse tables. But for sparse tables, it fails to give proper p-values. Therefore, how to find a better approximation of the marginal function for the hypergeometric distribution in sparse table case is still an open problem.

2. In Section 4.1.1, we showed that low acceptance rates will lead to less reliable $\widehat{|\Sigma|}$ and larger variation in the estimators. In [9], the Gale–Ryser Theorem (see Section 1.1.2) was used to obtain an SIS procedure with no rejection for two-way zero-one tables. An generalization of this theorem for three-way contingency tables is given

in [34]. Although we can not apply it directly to produce an SIS procedure with no rejection for three-way tables because we naturally have structural zeros and trivial cases in a process of sampling one table, it is interesting to generalize the results in [34] to contingency tables with structural zeros.

3. At the end of Section 4.1.1, we showed that cv^2 can be reduced remarkably by removing several "outliers". However, new issues come up because it is not clear whether it is reasonable to remove "outliers", i.e. whether the result is still reliable after removing them, and if the result is still reliable, then how to decide the cutoff of "outliers".

4.2 Discussion and future work for the characteristic imset polytopes for Bayesian networks

4.2.1 Discussion on the results in Chapter 3 and its connection with the K2 algorithm for learning Bayesian networks

Using similar strategy, the results in Section 3.3 can be further generalized: with fixed underlying ordering of nodes $[n]$ and sets of nodes $\Omega = \{\Omega_i,\ i = 2, \ldots, n\}$ such that $\Omega_i \subseteq \{a_{[1]}, \ldots, a_{[i-1]}\}$, if we define the class of graphs $\mathcal{G}_{[n],\Omega} = \{G \in DAGs(N) : [n]_G = [n],\ pa_G(a_{[i]}) \subseteq \Omega_i,\ i = 2, \ldots, n\}$, then the cim-polytope $\mathbf{P}_{\mathcal{G}_{[n],\Omega},c}$ is a direct product of a sequence of simplices:

$$\mathbf{P}_{\mathcal{G}_{[n],\Omega},c} = \Delta_{2^{|\Omega_2|}-1} \times \Delta_{2^{|\Omega_3|}-1} \times \ldots \times \Delta_{2^{|\Omega_n|}-1}, \tag{4.2.1}$$

where the i-th simplex $\Delta_{2^{|\Omega_{i+1}|}-1}$ is the same with the cim-polytope for diagnosis models, $\mathbf{P}_{|\Omega_{i+1}|,1}$, with diseases $A = \Omega_{i+1}$ and one symptom $a_{[i+1]}$. It is obvious that the cim-polytope for diagnosis models, $\mathbf{P}_{m,n}$, is a special case of $\mathbf{P}_{\mathcal{G}_{[n],\Omega},c}$: the underlying ordering of nodes is $(a_1, \ldots, a_m, b_1, \ldots, b_n)$ (the ordering is not unique in the sense that the order of two diseases or two symptoms can exchange), $\Omega_i = \emptyset$, $i = 1, \ldots, m$, and $\Omega_i = \{a_1, \ldots, a_m\}$, $i = m+1, \ldots, m+n$.

Once the cim-polytope can be written as a direct product of a sequences of simplices, we are able to find the optimal BN structure by maximizing a target function in each simplex (see Section 1.3.2): given data $D \in DATA(N, d)$,

$$\max_{G \in \mathcal{G}_{[n],\Omega}} \mathcal{Q}(G, D) \Longrightarrow \min_{\mathbf{x} \in \mathbf{P}_{\mathcal{G}_{[n],\Omega},c}} r_D^T \mathbf{x} = \sum_{i=2}^{n} \min_{\mathbf{x_i} \in \Delta_{2^{|\Omega_i|}-1}} r_{D,i}^T \mathbf{x_i}, \tag{4.2.2}$$

where $\mathbf{x_i}$ contains the coordinates $\{T \subseteq \Omega_i \cup \{a_{[i]}\} : |T| \geq 2, a_{[i]} \in T, a_{[j]} \notin T, \forall j > i\}$ in $\mathbf{x}$, and the coordinates of $r_{D,i}^T$ matches the coordinates of $\mathbf{x_i}$. This implies that we can find the optimal parent sets of $a_{[i]}$, $i = 2, \ldots, n$, sequentially until we obtain the whole BN structure, which will be exactly the optimal BN structure in $\mathcal{G}_{[n],\Omega}$.

Equation (4.2.2) gives a polyhedral geometric insight of the K2 algorithm [13], which is a well-known heuristic method in learning Bayesian networks. Recall that in K2 algorithm, an ordering on the nodes is also fixed and parent sets of $a_{[i]}$, $i = 2, \ldots, n$, are also determined sequentially. However, in order to find the optimal BN, Equation (4.2.2) claims that we need to find $G_i \in \mathcal{G}_{|\Omega_i|,1}$ such that $r_{D,i}^T c_{G_i} = \min_{\mathbf{x_i} \in \Delta_{2^{|\Omega_i|}-1}} r_{D,i}^T \mathbf{x_i}$, while the K2 algorithm obtain each parent set $pa_G(a_{[i]})$ by adding nodes to $\emptyset$ stepwisely (or removing nodes from $\{a_{[1]}, \ldots, a_{[i-1]}\}$ stepwisely), which cannot guarantee that the resulting parent sets are optimal (see Example 4.2.1 for a counter-example).

Example 4.2.1. *Consider $\mathcal{G}_{3,1}$. The characteristic imsets of all possible graphs in $\mathcal{G}_{3,1}$ is listed as a matrix:*

$$\begin{array}{c} \\ \begin{pmatrix} c_{G_0} \\ c_{G_1} \\ c_{G_2} \\ c_{G_3} \\ c_{G_{12}} \\ c_{G_{23}} \\ c_{G_{13}} \\ c_{G_{123}} \end{pmatrix} \end{array} = \begin{array}{c} \begin{matrix} T & a_1b_1 & a_2b_1 & a_3b_1 & a_1a_2b_1 & a_1a_3b_1 & a_2a_3b_1 & a_1a_2a_3b_1 \end{matrix} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

We are going to give counter-examples that the resulting BN of the K2 algorithm is not the optimal solution.

- *Forward selection, i.e. each parent set* $pa_G(_{[i]})$ *is obtained by adding nodes to* $\emptyset$ *stepwisely. Suppose* $r_D^T = (-1, -2, -1, -3, -10, -4, 20)$ *which satisfies* $r_D^T c_{G_{13}} = -12 < r_D^T c_G$, $\forall G \in \mathcal{G}_{3,1}$, $G \neq G_{13}$, *i.e. the optimal graph is* G_{13}. *In K2 algorithm, we start from* $pa_G(b_1) = \emptyset$. *Next,* a_2 *is added to* $pa_G(b_1)$ *because* $r_D^T c_{G_2} = -2 < r_D^T c_{G_1} = r_D^T c_{G_3} = -1$. *Then* a_3 *is added to* $pa_G(b_1)$ *because* $r_D^T c_{G_{23}} = -7 < r_D^T c_{G_{12}} = -6$. *Procedure ends here because* $r_D^T c_{G_{23}} = -7 < r_D^T c_{G_{123}} = -1$. *The graph chosen by K2 algorithm,* G_{23}, *is not the optimal graph.*

- *Backward selection, i.e. each parent set* $pa_G(a_{[i]})$ *is obtained by removing nodes from* $\{a_{[1]}, \ldots, a_{[i-1]}\}$ *stepwisely. Suppose* $r_D^T = (-3, -1, -1, 3, 3, 0, 10)$ *which satisfies* $r_D^T c_{G_1} = -3 < r_D^T c_G$, $\forall G \in \mathcal{G}_{3,1}$, $G \neq G_1$, *i.e. the optimal graph is* G_1. *In K2 algorithm, we start from* $pa_G(b_1) = \{a_1, a_2, a_3\}$. *Next,* a_1 *is removed from* $pa_G(b_1)$ *because* $r_D^T c_{G_{23}} = -2 < r_D^T c_{G_{12}} = r_D^T c_{G_{13}} = -1$. *Procedure ends here because* $r_D^T c_{G_{23}} = -2 < r_D^T c_{G_2} = r_D^T c_{G_3} = -1$. *The graph chosen by K2 algorithm,* G_{23}, *is not the optimal graph.*

4.2.2 Open problems and future work on characteristic imset polytopes of Bayesian networks

1. $\mathbf{P}_{\mathcal{G}_{[n],\Omega},c}$ is define in Section 4.2.1. Consider a vertex $v \in vert(\mathbf{P}_{\mathcal{G}_{[n],\Omega},c})$. A **normal cone** at v is a cone (see Section 1.3.1) generated by the normal vectors of all facets that contain v. In fact, the normal cone at v is the set of all cost vectors for vertex v (see Definition 1.3.4). The **normal fan** of $\mathbf{P}_{\mathcal{G}_{[n],\Omega},c}$ is the union of normal cones for all vertices of $\mathbf{P}_{\mathcal{G}_{[n],\Omega},c}$. We want to compute the normal fan of $\mathbf{P}_{\mathcal{G}_{[n],\Omega},c}$ so that we can analyze sensitivity of the quality criterions and data.

2. All work in Chapter 3 is theoretical. Although we have simplified our problem of learning BNs to LP problems over each simplex (see Equation (4.2.2)) in the direct produce showed in Theorem 3.3.3 and Equation (4.2.1), and have described all edges and facets of these simplices (see Section 3.2), if the number of nodes is large, we still have to face the possibility that the procedure of searching the optimal solutions in each simplex can be very time-consuming. In this sense, simulations and analysis on real datasets are very important to compare the solution and time complexity of our method with other existing classifiers [58]. On the other hand, we also need to study on the misspecification problem of our method via simulations, i.e. how our method performs when the underlying ordering of nodes is misspecified and how sensitive the assumed underlying ordering is to the results.

3. Consider a class of BNs $\mathcal{G}$ we are interested in. In practice, sometimes some BNs in $\mathcal{G}$ are preferable than others, in which case larger prior probabilities can be assigned to these BNs to actualize the trend of choosing these models, or sometimes we are more interested in the existence of some directed edges than others. However, it is not trivial to carry out this information in our method. Two possible ways can be considered as candidates. First, we may think about putting weights to the coordinates of the data vector r_D^T in Equation (4.2.2). Second, we may consider the class of graphs where some edges are forbidden and some edges are fixed, i.e. given a set of forbidden edges $\mathcal{E}_N$ and a set of fixed edges $\mathcal{E}_Y$, consider the structure of cim-polytope for $\mathcal{G} = \{G \in DAGs(N) : \forall \epsilon \in \mathcal{E}_N, \epsilon \text{ is not in } G, \forall \epsilon' \in \mathcal{E}_Y, \epsilon' \text{ is in } G\}$.

4. We are also interested in the structure of cim-polytopes for other types of BNs. Example are: the cim-polytope for all trees over N, the cim-polytope for all BNs over N where an upper bound on the number of parents for each node is fixed, and so on.

5. This dissertation focuses on the case that all random variables in N are finite random variables. It is still an open problem that how to generalize our method to the case that some or all of the random variables in N are continuous random variables.

Appendix

A.1 Non-parametric bootstrap method to compute confidence intervals for SIS procedure

In this section we will explain how to use a non-parametric bootstrap method to get the $(1-\alpha)100\%$ confidence interval for $|\Sigma|$. Notice that the bootstrap sample size is denoted by B, and see Chapter 2 for the notation.

(1) Drawing pseudo dataset.

- **Concept** In an SIS procedure with sample size $\mathfrak{N}$, we get a sequence of random tables $\mathbf{X_1}, \ldots, \mathbf{X}_{\mathfrak{N}}$. Define $\mathbf{Y_i} = \frac{\mathbb{I}_{\mathbf{X_i} \in \Sigma}}{q(\mathbf{X_i})}$, $i = 1, \ldots, \mathfrak{N}$, where $q(\mathbf{X})$ is the trial distribution, then $\mathbf{Y_1}, \ldots, \mathbf{Y}_{\mathfrak{N}}$ form a sequence of i.i.d random variables. This means that we can consider the empirical distribution of $\mathbf{Y_i}$, which is nonparametric maximum likelihood estimator of the real distribution of $\mathbf{Y_i}$ (since $\mathbf{Y_i}$ can only take finitely many values, the empirical distribution is in fact the maximum likelihood estimator of the real distribution). We can draw a pseudo sample $\mathbf{Y}^*_{\mathbf{1}}, \ldots, \mathbf{Y}^*_{\mathfrak{N}}$ from the empirical distribution.
- **Algorithm** Use the SIS procedure introduced in Chapter 2 to sample $\mathfrak{N}$ tables $\mathbf{X_1}, \ldots, \mathbf{X}_{\mathfrak{N}}$. If $\mathbf{X_i}$ is sampled successfully, then $\mathbb{I}_{\mathbf{X_i} \in \Sigma} = 1$ and $q(\mathbf{X_i})$ will be outputted, else $\mathbb{I}_{\mathbf{X_i} \in \Sigma} = 0$. Thus we can compute the values of $\mathbf{Y_i} = \frac{\mathbb{I}_{\mathbf{X_i} \in \Sigma}}{q(\mathbf{X_i})}$, $i = 1, \ldots, \mathfrak{N}$, and draw $\mathfrak{N}$ elements from this sequence with replacement.

(2) One Bootstrap replication.

- **Concept** Consider the pseudo sample $\mathbf{Y}_{\mathbf{1}}^*, \ldots, \mathbf{Y}_{\mathfrak{N}}^*$ as a "new" sample from the empirical distribution. Then the cumulative distribution function (CDF) of $\widehat{\theta}^* = T(\mathbf{Y}_{\mathbf{1}}^*, \ldots, \mathbf{Y}_{\mathfrak{N}}^*)$ is a consistent estimator of the CDF of $\widehat{\theta} = T(\mathbf{Y_1}, \ldots, \mathbf{Y}_{\mathfrak{N}})$. In this dissertation we consider the estimators for $|\Sigma|$,

$$\widehat{|\Sigma|} = \widehat{\theta_1} = T_1(\mathbf{Y_1}, \ldots, \mathbf{Y}_{\mathfrak{N}}) = \frac{1}{\mathfrak{N}} \sum_{i=1}^{\mathfrak{N}} \mathbf{Y_i},$$

and cv^2,

$$\widehat{cv^2} = \widehat{\theta_2} = T_2(\mathbf{Y_1}, \ldots, \mathbf{Y}_{\mathfrak{N}}) = \frac{\sum_{i=1}^{\mathfrak{N}} \{\mathbf{Y_i} - \left[\sum_{j=1}^{\mathfrak{N}} \mathbf{Y_j}\right]/\mathfrak{N}\}^2/(\mathfrak{N}-1)}{\{\left[\sum_{j=1}^{\mathfrak{N}} \mathbf{Y_j}\right]/\mathfrak{N}\}^2}.$$

- **Algorithm** Consider the pseudo sample $\mathbf{Y}_{\mathbf{1}}^*, \ldots, \mathbf{Y}_{\mathfrak{N}}^*$ as a sample from the SIS procedure and compute the first bootstrap replication

$$\widehat{|\Sigma|}^{*1} = \frac{1}{\mathfrak{N}} \sum_{i=1}^{\mathfrak{N}} \mathbf{Y_i^*} \text{ and } \widehat{cv^2}^{*1} = cv^2 of(\mathbf{Y}_{\mathbf{1}}^*, \ldots, \mathbf{Y}_{\mathfrak{N}}^*).$$

(3) Bootstrap-t Confidence Interval.

- **Concept** Repeat step (1) and step (2) until we get B Bootstrap replications: $\widehat{\theta}_i^{*1}, \ldots, \widehat{\theta}_i^{*B}$, $i = 1, 2$. Because the empirical distribution of $\widehat{\theta}_i^*$ is the nonparametric maximum likelihood estimator of CDF of $\widehat{\theta}_i^*$ and the latter is a consistent estimator of the CDF of $\widehat{\theta}_i$, we can use the $(\frac{\alpha}{2})100_{th}$ and $(1-\frac{\alpha}{2})100_{th}$ percentiles of the empirical distribution as the lower and upper bounds of the confidence interval.

- **Algorithm** Repeat step (1) and step (2) for B times. For $\{\widehat{|\Sigma|}^{*1}, \ldots, \widehat{|\Sigma|}^{*B}\}$, for $0 < a < 1$, define $\widehat{|\Sigma|}^*_{(a)}$ as the $100a_{th}$ percentile of the list of values. Then bootstrap-t $(1-\alpha)100\%$ confidence interval of $\widehat{|\Sigma|}$ is $\left[\widehat{|\Sigma|}^*_{(\alpha/2)}, \widehat{|\Sigma|}^*_{(1-\alpha/2)}\right]$. Similarly we can get the confidence interval for $\widehat{cv^2}$.

Table 1: Compare results for Sampson's dataset with/without 7 "outliers".

	Estimation			cv^2			
"Outliers"	$\widehat{\lvert\Sigma\rvert}$	Lower 95%	Upper 95%	$\widehat{cv^2}$	Lower 95%	Upper 95%	Acceptance Rate
With	1.313089e+117	4.771677e+116	2.391368e+117	392.6767	230.4170	711.2878	2.803%
Without	1.932762e+116	9.973317e+115	3.154941e+116	226.8825	124.2770	336.6534	2.796%

The sample size for SIS procedure is $\mathfrak{N} = 100000$ and the sample size for bootstrapping is $B = 50000$. The cutoff for "outliers" of number of tables is 4e+120, i.e. samples with $\mathbf{Y_i} = \frac{\mathbb{I}_{\mathbf{X_i} \in \Sigma}}{q(\mathbf{X_i})} > 4 \times 10^{120}$ are removed. The 7 "outliers" are: 2.83e+121, 1.93e+121, 3.00e+121, 1.07e+121, 9.08e+120, 4.66e+120, 1.00e+121.

A.2 Manual for the R code to sample and estimate the number of zero-one three-way tables with given two-way marginals

The software is implemented in **R**. It can be either used to sample a zero-one three-way table with given two-way marginals via the SIS procedure introduced in Chapter 2, or used to estimate the number of such tables. Note that this software needs minor modification to allow the existence of structures in the observed table. We are going to give the syntaxes and examples for the two main functions, `genbin` and `numtable`.

- **Function** `genbin`.

Description This function is used to sample a zero-one three-way table with given two-way marginals via the SIS procedure introduced in Chapter 2.

Usage
```
genbin(outinfo, output = T)
genbin(outinfo = tabinfo(x0), output = T)
```

Arguments `outinfo`: a list of three matrices that present the fixed two-way marginals. These three matrices are denoted as si, sj and sk (see Section 2.4). This list can either be given by users directly, or be computed through an observed table `x0`. Function `tabinfo` is available to compute the two-way marginals: `outinfo = tabinfo(x0)`.

`output`: logical; if TRUE (default), the output will a list consist of `A`, `logcpr` and `ntable`, otherwise the output will be a list that only includes `ntable`.

Output `A`: a zero-one three-way table sampled by SIS procedure that satisfies the given two-way marginals. Only appear if the sampling succeeds.

`logcpr`: the logarithm of $q(\mathtt{A})$, where $q(\cdot)$ is the trial distribution and `A` is the sampled table. Only appear if the sampling succeeds.

`ntable`: the value of $1/q(\mathtt{A})$, which can be considered as the estimator of

$|\Sigma|$, $|\widehat{\Sigma}|$, based on this single sample. Only appear if the sampling succeeds.

If the sample is rejected, then the output will be a number `0`.

Example Sample a 3-dimensional semimagic cube in Example 2.4.1.

```
si = sj = sk = matrix(c(1, 1, 1, 1, 1, 1, 1, 1, 1), 3, 3)
outinfo = list(si = si, sj = sj, sk = sk)
genbin(outinfo)
```

The output of the above code:

```
$A
, , 1
[,1] [,2] [,3]
[1,]    0    0    1
[2,]    0    1    0
[3,]    1    0    0
, , 2
[,1] [,2] [,3]
[1,]    0    1    0
[2,]    1    0    0
[3,]    0    0    1
, , 3
[,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    0    1
[3,]    0    1    0
$logcpr
[1] -2.484907
$ntable
```

```
[1] 12
```

- **Function** `numtable`.

Description This function is used to estimate the number of zero-one three-way tables with given two-way marginals via the SIS procedure introduced in Chapter 2.

Usage
```
numtable(N = 1000, outinfo, knotprint=50)
numtable(N = 1000, outinfo = tabinfo(x0), knotprint=50)
```

Arguments `N`: the number of samples produced by SIS procedure, including those which are rejected.

`outinfo`: a list of three matrices that present the fixed two-way marginals. These three matrices are denoted as si, sj and sk (see Section 2.4). This list can either be given by users directly, or be computed through an observed table `x0`. Function `tabinfo` is available to compute the two-way marginals: `outinfo = tabinfo(x0)`.

`knotprint`: a number of samples to print a note to the screen. The purpose of this argument is giving users the information about how many samples have been finished so that users can estimate how much time left to end the process.

Output `NumofTables`: the estimator of $|\Sigma|$, $\widehat{|\Sigma|}$, based on the `N` samples.

`cv2`: the estimator of cv^2, $\widehat{cv^2}$, which is a measurement of accuracy for $\widehat{|\Sigma|}$ (see Section 2.4).

`acceptance`: the acceptance rate of the `N` sampled tables, which is the ratio of the number of accepted tables to `N`.

Example Estimate the number of zero-one $3\times3\times4$ tables with the two-way marginals given in Example 2.4.2.

```
seed = 6;
```

```
m = 3; n = 3; l = 4; prob = 0.8;
N = 1000; k = 200
set.seed(seed)
A = array( rbern(m*n*l, prob), c(m, n, l) )
numtable(N = 1000, outinfo = tabinfo(A), k = 200)
```

The output of the above code:

```
Finished  200  tables
Finished  400  tables
Finished  600  tables
Finished  800  tables
Finished  1000  tables
$NumofTables
[1] 3.005
$cv2
[1] 0.1116811
$acceptance
[1] 1
```

A.3 R code to sample and estimate the number of zero-one three-way tables with given two-way marginals

The code is available at `http://www.polytopes.net/code/CP/`.

```
u=1
printDebug <- 0 # 0 = no printing, 1 = print debug information
preclearcheck <- 0 # whether do preclear2way check
myPrint <- function(myStr) {
    if (printDebug==1)    {
        print(myStr)
    }
}
cp <- function(p, c) # all elements in p are in (0,1) {
m = length(p); w = p/(1-p)^u; # m may not be num of rows
Z=rep(0,m)
rest=1:m; done=NULL;
if(c==0) return(list(Z=Z,done=done,logcpr=0))
if(m==c) return(list(Z=rep(1,m),done=rest,logcpr=0))
if(m<c)  return(0)    ###fail, should back, use is.list to judge
if(c>m/2) {
outcp=cp(1-p,m-c); Z=rep(1,m)-outcp$Z;
done=(1:m)[-outcp$done]
return(list(Z=Z,done=done,logcpr=outcp$logcpr))
}
#only left 0<c<=m/2
while(length(done)<c) {
outd=drawone(w,rest,done,c)
```

```
ik=outd$ik
done=c(done,ik); rest=rest[rest!=ik] #k=length(done)
if(length(done)==1) denompr=outd$invconst
}
#compute cp prob
Z[done]=1
lognumpr=sum(Z*log(w))
logcpr=lognumpr-log(denompr) #log(cp prob)

return(list(Z=Z,done=done,logcpr=logcpr))
}
Rfunc <- function(s, A, w)
{ #A is subset of {1,...,m}, w=(w1,...,wm)
lA=length(A)
if(lA<s)
{print("Invalid R function"); return(0)}
if(s==0) return(1)
if(s==1) return(sum(w[A]))
if(s==lA) return(prod(w[A]))
RsA=Rfunc(s,A[-1],w)+w[A[1]]*Rfunc(s-1,A[-1],w)
return(RsA)
}
drawone <- function(w, rest, done, c) {
lenr=length(rest); lend=length(done)
Pj=rep(0,lenr)
up=w[rest[1]]*Rfunc(c-lend-1,rest[-1],w)
downR=Rfunc(c-lend,rest[-1],w)+up
```

```
Pj[1]=up/((c-lend)*downR)
if(lenr>1)
{
for(i in 2:lenr)
{
up=w[rest[i]]*Rfunc(c-lend-1,rest[-i],w)
Pj[i]=up/((c-lend)*downR)
}
}
ik=sample(rest,1,prob=Pj)
return(list(ik=ik,invconst=downR))
}
tabinfo <- function(x0) {
judge=1
if(sum(x0<0)) judge=0
if(sum((x0==0),(x0==1))<length(x0)) judge=0
si=apply(x0,c(2,3),sum)  # sum, only 1st index not fixed
sj=apply(x0,c(1,3),sum)
sk=apply(x0,c(1,2),sum)
return(list(si=si,sj=sj,sk=sk,judge=judge))
}
onecol <- function(si,rs,cs,m=m,n=length(cs),l=length(rs),strucA)
{ # no trivial case but may has structrual 0, generate first col
if(cs[1]==0) return(list(vec=rep(0,l),logcpr=0))
ck=si[1,]
rk=rs
vec=rep(-1,l)
```

```
struc0=which(strucA[1,1,]==1)
vec[struc0]=0
if(length(struc0)==l) {
if(sum(vec)!=cs[1]) return(0) # not feasible
else return(list(vec=vec,logcpr=0))
}
left=(1:l)
if(length(struc0)>0) left=left[-struc0]
if(length(left)<cs[1])  {
return(0) # not feasible for binary
}
grk=apply(strucA[1,,],2,sum)
gck=apply(strucA[,1,],2,sum)
t1=n-grk[left]-rk[left]
t2=m-gck[left]-ck[left]
if(sum(t1<=0)+sum(t2<=0)>0) return(0) else{
p=rk[left]*ck[left]/(rk[left]*ck[left]+t1*t2)
# cat("p=",p," rk=",rk," ck=",ck," m=",m,"\n")
outcp=cp(p,cs[1])
vec[left]=outcp$Z
logcpr=outcp$logcpr
return(list(vec=vec,logcpr=logcpr))
}
}
firstgencol <- function(cs,m) # firstgen2 in 11th-GR-ver2.R
{
j=which(cs==max(cs))[1]
```

```
return(gen=j)
}
# Find layer with largest sum (most 1's)
firstgenlay <- function(sj) # first layer to generated {
layersum=apply(sj,1,sum)
mlayer=max(layersum)
i=which(layersum==mlayer)[1]
return(gen=i)
}
# si is not used in this function at all.
# This function uses the structure 0 information,
# and the row and column sum to check if there are
# trivial rows or columns.
clear2way <- function(si,rs,cs,m,n,l,strucA) {
    myPrint(sprintf("clear2way: m=%f, n=%f, l=%f",m,n,l))
    myPrint(si)
    myPrint(rs)
    myPrint(cs)
    myPrint(sprintf("clear2way: sum(strucA)=%d",sum(strucA)))
A2 <- t(strucA[1,,])
B2 <- matrix(0,l,n)
clearr=NULL; clearc=NULL
stop=0
while(stop==0) {
        myPrint("clear2way: Loop:")
        #print(A2)
        #print(B2)
```

```
stop=1
A2rs=apply(A2,1,sum)
B2rs=apply(B2,1,sum)
        myPrint(sprintf("clear2way: A2rs"))
        myPrint(A2rs)
        myPrint(sprintf("clear2way: B2rs"))
        myPrint(B2rs)
pr=(rs-B2rs)/(n-A2rs)
        myPrint(sprintf("clear2way: pr"))
        myPrint(pr)
for(k in 1:l) {
if((rs-B2rs)[k]!=0 || (n-A2rs)[k]!=0) {
if(pr[k]<0 || pr[k]>1)
                {
                    myPrint(sprintf("clear2way: Row sum [%d]
probability not in [0,1]",k))
                    return(0)
                }
if(pr[k]==1)
B2[k,][which(A2[k,]==0)]=1 # B2 records the structure [1]
if(pr[k]==0 || pr[k]==1) {
                    myPrint(sprintf("clear2way: Row sum [%d] value
in {0,1}.",k))
A2[k,]=1
clearr=c(clearr,k)
stop=0
}
```

```
}
}
        myPrint("clear2way: Loop (cs check):")
        #print(A2)
        #print(B2)
A2cs=apply(A2,2,sum)
B2cs=apply(B2,2,sum)
        myPrint(sprintf("clear2way: A2cs"))
        myPrint(A2cs)
        myPrint(sprintf("clear2way: B2cs"))
        myPrint(B2cs)
pc=(cs-B2cs)/(1-A2cs)
        myPrint(sprintf("clear2way: pc"))
        myPrint(pc)
for(j in 1:n) {
if((cs-B2cs)[j]!=0 || (1-A2cs)[j]!=0) {
if(pc[j]<0 || pc[j]>1)
                {
                    myPrint(sprintf("clear2way: Col sum [%d]
probability not in [0,1]",j))
                    return(0)
                }
if(pc[j]==1)
                    B2[,j][which(A2[,j]==0)]=1
if(pc[j]==0 || pc[j]==1) {
                    myPrint(sprintf("clear2way: Col sum [%d] value
in {0,1}.",j))
```

```
A2[,j]=1
clearc=c(clearc,j)
stop=0
}
}
}
}
leftr=(1:l); leftc=(1:n)
if(length(clearr)>0) leftr=leftr[-clearr]
if(length(clearc)>0) leftc=leftc[-clearc]
change=1
if(length(clearr)==0 && length(clearc)==0) {
change=0
myPrint(sprintf("clear2way: change = %d,sum(X) = %d",change,sum(B2)))
return(list(X=B2,continue=1,change=change))
}
strucA[1,,]=t(A2)
if(length(leftr)==0 || length(leftc)==0) {
myPrint(sprintf("clear2way: change = %d,sum(X) = %d",change,sum(B2)))
return(list(X=B2,continue=0))
} else
    myPrint(sprintf("clear2way: change = %d",change))
return(list(X=B2,leftr=leftr,leftc=leftc,strucA=strucA,continue=1,
change=change))
}
twoway <- function(si, rs, cs, m, n=length(cs), l=length(rs), strucA){
    myPrint(sprintf("twoway: m=%f, n=%f, l=%f",m,n,l))
```

```
    myPrint(si)
    myPrint(rs)
    myPrint(cs)
    myPrint(sprintf("twoway: sum(strucA) = %d",sum(strucA)))
if(n==1) return(list(X=rs,logcpr=0))
if(l==1) return(list(X=cs,logcpr=0))
X <- matrix(-1, l, n)
p <- rs/n
pcol <- cs/l
logcpr=0
#If any p>1 or p<0, already unfeasible
badrow=c(which(p<0),which(p>1))
badcol=c(which(pcol<0),which(pcol>1))
if(length(badrow)+length(badcol)>0)
    { myPrint(sprintf("twoway: length(badrow)+length(badcol)>0. %f
+ %f > 0. Returning 0",length(badrow),length(badcol)))
        myPrint(p)
        myPrint(pcol)
        return(0)
    }
#initialize the structures
strucA1=t(strucA[1,,])
for(i in 1:l) {
for(j in 1:n) {
if(strucA1[i,j]==1)
            {
                myPrint(sprintf("twoway: Setting X[%d,%d]=0",i,j))
```

```
                X[i,j]=0
            }
}
}
# Maybe comment out this since clear2way is better.
# In fact, this section could lead to bugs as Jing pointed out.
# Eg. if the row sum is equal to the number of cells,
# this code will fill in the entire row with 1's. However,
# this is potentially a problem since there may be
# structure 0's which implies the row is infeasible!
# BEGIN ___
# fill those with row p=0 or 1
indp0<- which(p==0)
indp1<- which(p==1)
if((length(indp0)>0 || length(indp1)>0) && preclearcheck == 1) {
        myPrint("twoway: Some row probs 0 or 1")
leave <- which((p>0)*(p<1)>0)
X[indp0,] <- 0
X[indp1,] <- 1
if(length(leave)==0)      {
            myPrint ("twoway: No other row probabilities in (0,1).
Returning.")
            return(list(X=X,logcpr=0))
        }
        else {
            myPrint ("twoway: Still some row probabilities in (0,1).
Calling twoway.")
```

```
out2w1=twoway(si=si[,leave],rs=rs[leave],cs=cs-length(indp1),
  m=m,strucA=strucA[,,leave])
if(!is.list(out2w1))
            {
            myPrint ("twoway: Function returned empty list.")
            return(0)
            }
            else {
X[leave, ] <- out2w1$X;
logcpr=logcpr+out2w1$logcpr
return(list(X=X,logcpr=logcpr))
}
}
}
# END ___
else {

# fill those with col pcol=0 or 1
cindp0<- which(pcol==0)
cindp1<- which(pcol==1)
if((length(cindp0)>0 || length(cindp1)>0) && preclearcheck == 1) {
            myPrint("twoway: Some col probs 0 or 1")
cleave <- which((pcol>0)*(pcol<1)>0)
X[,cindp0] <- 0
X[,cindp1] <- 1
if(length(cleave)==0)
            {
```

```
                myPrint ("twoway: No other probabilities in (0,1).
Returning.")
                return(list(X=X,logcpr=0))
            }
            else {
out2w2=twoway(si=si[cleave,],rs=rs-length(cindp1),cs=cs[cleave],
  m=m,strucA=strucA[,cleave,])
if(!is.list(out2w2))
                {
                    myPrint ("twoway: Function twoway (col) returned
empty list.")
                    return(0)
                }
                else {
X[,cleave] <- out2w2$X;
logcpr=logcpr+out2w2$logcpr
return(list(X=X,logcpr=logcpr))
}
}
}
# left only cases with p, 0<p<1, and pcol, 0<pcol<1
else {
            myPrint ("twoway: No p entries 0 or 1")
outc2w=clear2way(si,rs,cs,m,n,l,strucA)
if(!is.list(outc2w))
            {
                myPrint("twoway: clear2way returned empty list.
```

```
Returning.")
                return(0)
            }
            else {
if(outc2w$continue==0)  return(list(X=outc2w$X,logcpr=0))
else {
if(outc2w$change) {
X=outc2w$X; leftc=outc2w$leftc; leftr=outc2w$leftr
#si=si[leftc,leftr]
strucA=outc2w$strucA[,leftc,leftr]
                        myPrint ("twoway: outc2w$change non-zero.
Calling twoway.")
                        myPrint ("#\\___//\\___//#\\___//#")
                        myPrint ("t(X)=")
                        myPrint (t(X))
                        Xrs=apply(X,1,sum)
                        Xcs=apply(X,2,sum)
                        myPrint("Xrs=")
                        myPrint(Xrs)
                        myPrint("Xcs=")
                        myPrint(Xcs)
out2w4=twoway(si=si[leftc,leftr],rs=(rs - Xrs)[leftr],cs=(cs - Xcs)
[leftc],m=m,strucA=strucA)
if(!is.list(out2w4))    return(0)
                    else {
X[leftr,leftc]=out2w4$X
logcpr=logcpr+out2w4$logcpr
```

```
return(list(X=X,logcpr=logcpr))
}
}
else {
genj=firstgencol(cs,l)
neworderc=1:n; neworderc[1]=genj; neworderc[genj]=1
tempcs=cs[neworderc]; # pretend that the jth col is the 1st col
myPrint(sprintf("twoway: Calling onecol. genj = %d",genj))
outonecol <- onecol(si[neworderc,], rs, tempcs, m=m,
strucA=strucA[,neworderc,])
                        myPrint("twoway: outonecol$vec = ")
                        myPrint(outonecol$vec)
if(!is.list(outonecol))     return(0)
                        else {
X[,genj] <- outonecol$vec
logcpr=logcpr+outonecol$logcpr
myPrint ("twoway: outc2w$change zero. Calling twoway.")
out2w3=twoway(si=si[-genj,],rs=rs-X[,genj],cs=cs[-genj],m=m,
strucA=strucA[,-genj,])
if(!is.list(out2w3))      return(0)
                            else {
X[,-genj] <- out2w3$X
logcpr=logcpr+out2w3$logcpr
return(list(X=X,logcpr=logcpr))
}
}
}
```

```
}
}
}
}
    myPrint("twoway: Reached end of function. Is this possible?")
return(list(X=X,logcpr=logcpr))
}
# Using marginals, return the structure 0's and structure 1's
# Recalculate marginals by subtracting structure 1's
# In the end, we will add the B matrix below
# A matrix: 1 means SOME structure there at that position
# B matrix: 1 means structure 1 at that position (if 1 in A),
# 0 means structure 0.
strucarray <- function(si, sj, sk, m=dim(sk)[1], n=dim(si)[1], l=dim
(si)[2]) {
A <- array(0,c(m,n,l)) #store all structures
B <- array(0,c(m,n,l)) #only struc 1
stop=0
while(stop==0) {
stop=1
Asi=apply(A,c(2,3),sum)
Bsi=apply(B,c(2,3),sum)
pi <- (si-Bsi)/(m-Asi)
for(j in 1:n) {
for(k in 1:l) {
if((si-Bsi)[j,k]!=0 || (m-Asi)[j,k]!=0) {
if(pi[j,k]<0 || pi[j,k]>1) return(0)
```

```
if(pi[j,k]==1) B[,j,k][which(A[,j,k]==0)] <-1
if(pi[j,k]==0 || pi[j,k]==1)
{A[,j,k] <- 1; stop=0}
}
}
}
Asj=apply(A,c(1,3),sum)
Bsj=apply(B,c(1,3),sum)
pj <- (sj-Bsj)/(n-Asj)
for(i in 1:m) {
for(k in 1:l) {
if((sj-Bsj)[i,k]!=0 || (n-Asj)[i,k]!=0) {
if(pj[i,k]<0 || pj[i,k]>1) return(0)
if(pj[i,k]==1) B[i,,k][which(A[i,,k]==0)] <-1
if(pj[i,k]==0 || pj[i,k]==1)
{A[i,,k] <- 1; stop=0}
}
}
}
Ask=apply(A,c(1,2),sum)
Bsk=apply(B,c(1,2),sum)
pk <- (sk-Bsk)/(l-Ask)
for(i in 1:m) {
for(j in 1:n) {
if((sk-Bsk)[i,j]!=0 || (l-Ask)[i,j]!=0) {
if(pk[i,j]<0 || pk[i,j]>1) return(0)
if(pk[i,j]==1) B[i,j,][which(A[i,j,]==0)] <-1
```

```
if(pk[i,j]==0 || pk[i,j]==1)
{A[i,j,] <- 1; stop=0}
}
}
}
}
newsi=si-apply(B,c(2,3),sum)
newsj=sj-apply(B,c(1,3),sum)
newsk=sk-apply(B,c(1,2),sum)
return(list(A=A,B=B,si=newsi,sj=newsj,sk=newsk))
}
# Single three way table.
# si = X_+jk,  sj = X_i+k, sk = X_ij+
threeway <- function(si, sj, sk, m=dim(sk)[1], n=dim(si)[1], l=dim
(si)[2]) {
    myPrint(sprintf("threeway: m=%f, n=%f, l=%f",m,n,l))
if(m==1) return(list(A=si,logcpr=0))
if(n==1) return(list(A=sj,logcpr=0))
if(l==1) return(list(A=sk,logcpr=0))
A <- array(-1,c(m,n,l))
logcpr <- 0
outsa=strucarray(si,sj,sk)
if(!is.list(outsa))     {
        myPrint("threeway: Function strucarray returned empty list.
Returning 0.")
        return(0)
    }
```

```
#if the whole table is made of structures
outsaA=outsa$A
if(sum(outsaA)==m*n*l)    {
        myPrint ("threeway: Entire tables is structure. Returning.")
        return(list(A=outsa$B,logcpr=0))
    }
    myPrint(sprintf("threeway: Number of structures %d", sum(outsaA)))
si=outsa$si
sj=outsa$sj
sk=outsa$sk
for(i in 1:m) {
for(j in 1:n) {
for(k in 1:l) {
if(outsaA[i,j,k]==1) A[i,j,k]=0 #add B later
}
}
}
geni=firstgenlay(sj)
    myPrint(sprintf("threeway: geni=%d",geni))
    #if the whole layer is made of structures
if(sum(outsa$A[geni,,])==n*l) {
A[geni,,]=0
out3way0=threeway(si,sj[-geni,],sk[-geni,],m=m-1)
# sj[-geni,] removes the geni element in the vector
if(!is.list(out3way0))        {
            myPrint ("threeway: Function threeway (all structs) return
empty list. Returning 0.")
```

```
            return(0)
        }
        else {
            A[-geni,,]=out3way0$A
            logcpr=logcpr+out3way0$logcpr
            return(list(A=A+outsa$B,logcpr=logcpr))
}
 }
     else {
rs=sj[geni,]; cs=sk[geni,]
neworderl=1:m; neworderl[1]=geni; neworderl[geni]=1
out2way=twoway(si,rs,cs,m,strucA=outsa$A[neworderl,,])
if(!is.list(out2way))         {
            myPrint ("threeway: Function twoway returned empty list.
Returning 0.")
            return(0)
        }
        else {
A[geni,,]=t(out2way$X);
logcpr=logcpr+out2way$logcpr
out3way=threeway(si-A[geni,,],sj[-geni,],sk[-geni,],m=m-1)
if(!is.list(out3way))
            {
                myPrint ("threeway: Function threeway returned empty
list. Returning 0.")
                return(0)
            }
```

```
            else {
A[-geni,,]=out3way$A
logcpr=logcpr+out3way$logcpr
return(list(A=A+outsa$B,logcpr=logcpr))
}
}
 }
}
genbin <- function(outinfo, output=T) {
si=outinfo$si; sj=outinfo$sj; sk=outinfo$sk
out3way=threeway(si,sj,sk)
if(is.list(out3way)) {
A <- out3way$A
outti <- tabinfo(A)
check=checkbin(A, si, sj, sk) # Checks the rows and column sums
} else return(0);
if(!check) return(0);
logcpr=out3way$logcpr;
ntable=1/(exp(logcpr))
if(output) {
return(list(A=A,logcpr=logcpr,ntable=ntable));
}
else return(list(ntable=ntable))
}
checkbin <- function(A, si, sj, sk) {
check=1;
outti=tabinfo(A)
```

```
if(outti$judge==0) check=0
j1=sum((si-outti$si)^2)+sum((sj-outti$sj)^2)+sum((sk-outti$sk)^2)
if(j1>0) check=0
return(check)
}
numtable <- function(N=1000,outinfo,knotprint=50) {
success=NULL
vecntable=rep(0,N)
for(i in 1:N) {
outgb=genbin(outinfo,output=F)
if(is.list(outgb)) {
vecntable[i]=outgb$ntable
success=c(success,i)
}
if(i%%knotprint==0) cat("Finished ",i," tables\n");
}
aventable=mean(vecntable) # suggested by Dr. Chen, should be unbiased
varntable=var(vecntable[success])
aventable2=mean(vecntable[success])
acceptance=length(success)/N
return(list(NumofTables=aventable,cv2=varntable/(aventable2)^2,
acceptance=acceptance))
}
```

B.1 Additional theorems and proofs in Chapter 3

This section will provide some additional theorems and proofs for Section 3.2. Recall that in Section 3.2, we first proved that $\mathbf{P}_{m,1}$ is a simplex Δ_{2^m-1}, and then we proved that $\mathbf{P}_{m,n}$ is a direct product of n many Δ_{2^m-1}, which implies that $\mathbf{P}_{m,n}$ is a simple polytope with dimension $n \cdot (2^m - 1)$. In this section, we are going to show another flow to prove the results in Section 3.2.

First, we will use linear algebra to show that $\mathbf{P}_{m,n}$ has dimension $n \cdot (2^m - 1)$. We adopt the notation from Section 3.2. Given N, by Proposition 3.1.2 and Proposition 3.1.6, we can define $\mathcal{S}_{m,n}$ as the support of $\{c_G : G \in \mathcal{G}_{m,n}\}$, i.e.:

$$\mathcal{S}_{m,n} = \{T : \exists\ G \in \mathcal{G}_{m,n} \text{ such that } c_G(T) = 1\} \subset \mathcal{P}(N),$$

where $\mathcal{P}(N)$ is the power set of N.

Theorem 4.2.2. *Fix m and n. The dimension of $\mathbf{P}_{m,n}$ is exactly $n \cdot (2^m - 1)$.*

Proof. Similar with imsets, we can consider the standard basis $\mathbf{e_T}$, $T \subset N$, as functions $\mathbf{e_T} : \mathcal{P}(N) \mapsto \mathbb{Z}$ such that $\forall\ T_0 \subset N$, $\mathbf{e_T}(T_0) = 1$ if $T_0 = T$, and 0 otherwise. Each $\mathbf{e_T}$ can also be considered as a vector with coordinates $T_0 \subset N$.

It is obvious that: 1) $\{c_G,\ G \in \mathcal{G}_{m,n}\} \subset \mathbb{R}^{2^{m+n}-(m+n+1)}$; 2) $\{\mathbf{e_T}, T \in \mathcal{S}_{m,n}\}$ is a basis of $\mathbb{R}^{n\cdot(2^m-1)}$ that is embedded in $\mathbb{R}^{2^{m+n}-(m+n+1)}$ (Proposition 3.1.6); and 3)$\{c_G,\ G \in \mathcal{G}_{m,n}\}$ can be written as a linear combination of $\{\mathbf{e_T}, T \in \mathcal{S}_{m,n}\}$. We are going to prove that $\{\mathbf{e_T}, T \in \mathcal{S}_{m,n}\}$ can be expressed as a linear combination of $\{c_G,\ G \in \mathcal{G}_{m,n}\}$. Notice that $\{\mathbf{e_T}, T \in \mathcal{S}_{m,n}\}$ is equivalent with $\{\mathbf{e_T}, T \subset N$ and T has the form of $a_{i_1} \dots a_{i_k} b_j$, where $1 \leq k \leq m$, $\{i_1, \dots, i_k\} \subseteq \{1, \dots, m\}$ and $j \in \{1, \dots, n\}\}$ (Proposition 3.1.2), we can prove the statement by induction on $|T|$.

- When $|T| = 2$ (i.e. $k = 1$), i.e. $T = a_i b_j$, where $a_i \in A$ and $b_j \in B$, we know $c_G = \mathbf{e_T}$, where $G \in \mathcal{G}_{m,n}$ has only one edge $a_i \to b_j$.

- Suppose $\forall\, T$, T has the form in Proposition 3.1.2 and $|T| \leq k$, $\mathbf{e_T}$ can be written as a linear combination of $\{c_G,\ G \in \mathcal{G}_{m,n}\}$. Now consider $T_k = a_{i_1} \dots a_{i_k} b_j$, where $\{i_1, \dots, i_k\} \subseteq \{1, \dots, m\}$ and $j \in \{1, \dots, n\}$.

  Let $G \in \mathcal{G}_{m,n}$ have k edges: $a_{i_l} \to b_j$, $l = 1 \dots k$. Then:

$$\mathbf{e_{T_k}} = c_G - \sum_{T_a \subset \{a_{i_1}, \dots, a_{i_k}\}, 0 < |T_a| < k} \mathbf{e_{T_a \cup \{b_j\}}}.$$

  Since $\forall\ T_a \subset \{a_{i_1}, \dots, a_{i_k}\}$, $0 < |T_a| < k$ (i.e. $T_a \subsetneq \{a_{i_1}, \dots, a_{i_k}\}$), $|T_a \cup b_j| \leq k$, $\mathbf{e_{T_a \cup b_j}}$ can be expressed as a linear combination of $\{c_G,\ G \in \mathcal{G}_{m,n}\}$. Therefore, $\mathbf{e_{T_k}}$ can be written as a linear combination of $\{c_G,\ G \in \mathcal{G}_{m,n}\}$.

□

A special case of $n = 1$ in Theorem 4.2.2 and Proposition 3.1.5 claims that $\mathbf{P}_{m,1}$ has 2^m vertices and dimension $2^m - 1$. This directly lead to Corollary 4.2.3.

Corollary 4.2.3. *Fix m, $\mathbf{P}_{m,1}$ is a simplex with dimension $2^m - 1$, i.e. $\mathbf{P}_{m,1} = \Delta_{2^m - 1}$.*

Lemma 3.2.2 is an immediate result of Corollary 4.2.3, while Theorem 3.2.5 and Theorem 3.2.7 can be obtained based on Lemma 3.2.2 and Corollary 4.2.3 using the same proofs in Section 3.2. It is worth mentioning that Theorem 4.2.2 and Theorem 3.2.5 imply that $\mathbf{P}_{m,n}$ is a simple polytope with dimension $n \cdot (2^m - 1)$ because the number of neighbors for each vertex equals to the dimension of the polytope. In 2000, V. Kaibel and M. Wolff proved that a zero-one polytope is simple if and only if it equals to a direct product of zero-one simplices [33]. Recall that characteristic imset polytopes are zero-one polytopes (Theorem 1.3.10), we are able to conclude that $\mathbf{P}_{m,n}$ is a direct product of zero-one simplices [33]. Our progress is that we proved a even strong result in Theorem 3.2.7 with an intuitive graphical interpretation of each simplex in the direct product.

Bibliography

[1] A. Agresti. Exact inference for categorical data: recent advances an continuing controversies. Statistics in Medicine, 20:2709–2722, 2001.

[2] A. Agresti. Categorical Data Analysis. Wiley, second edition, 2002.

[3] S. A. Andersson, D. Madigan, and M. D. Perlman. A characterization of markov equivalence classes for acyclic digraphs. Annals of Statistics, 25:505–541, 1997.

[4] E. A. Bender and J. R. Goldman. On the application of mobius inversion in combinatorial analysis. The American Mathematical Monthly, 82:789–803, 1975.

[5] R. R. Bouckaert and M. Studený. Chain graphs: semantics and expressiveness, in symbolic and quantitative approaches to reasoning and uncertainty (c. froidevaux, j. kohlas eds.). Lecture Notes in Artificial Intelligence 946, pages 67–76. Springer Verlag, 1995.

[6] R. Breiger, S. Boorman, and P. Arabie. An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. Journal of Mathematical Psychology, 12:328–383, 1975.

[7] X. H. Chen, A. P. Dempster, and J. S. Liu. Weighted finite population sampling to maximize entropy. Biometrika, 81:457–469, 1994.

[8] Y. Chen. Conditional inference on tables with structural zeros. Journal of Computational and Graphical Statistics, 16(2):445–467, 2007.

[9] Y. Chen, P. Diaconis, S. Holmes, and J. S. Liu. Sequential monte carlo methods for statistical analysis of tables. J. Amer. Statist. Assoc., 100:109–120, 2005.

[10] Y. Chen, I. H. Dinwoodie, and S. Sullivant. Sequential importance sampling for multiway tables. The Annals of Statistics, 34(1):523–545, 2006.

[11] D. M. Chickering. Learning bayesian networks is np-complete. In Learning from Data: Artificial Intelligence and Statistics V, pages 121–130. Springer-Verlag, 1996.

[12] R. R. Cook and J. F. Quinn. The influence of colonization in nested species subsets. Oecologia, 102:413–424, 1995.

[13] G. F. Cooper. A bayesian method for the induction of probabilistic networks from data. Machine Learning, 9:309–347, 1992.

[14] J. De Loera and S. Onn. All rational polytopes are transportation polytopes and all polytopal integer sets are contingency tables. In Tenth International Conference in Integer Programming and Combinatorial Optimization, pages 338–351. Springer, 2004.

[15] J. De Loera and S. Onn. Markov bases of three-way tables are arbitrarily complicated. J. Symb. Comput., 41(2):173–181, 2005.

[16] J. De Loera and S. Onn. All linear and integer programs are slim 3-way transportation programs. SIAM Journal on Optimization, 17:806–821, 2006.

[17] J. A. De Loera, D. Haws, R. Hemmecke, P. Huggins, J. Tauzer, and R. Yoshida. LattE, version 1.2. Available from URL `http://www.math.ucdavis.edu/~latte/`, 2005.

[18] P. Diaconis and B. Efron. Testing for independence in a two-way table: New interpretations of the chi-square statistic (with discussion). Annals of Statistics, 13:845–913, 1985.

[19] P. Diaconis and B. Sturmfels. Algebraic methods for sampling from conditional distributions. Annals of Statistics, 26:363–397, 1998.

[20] I. H. Dinwoodie. Polynomials for classification trees and applications, 2008.

[21] I. H. Dinwoodie and Y. Chen. Sampling large tables with constraints. Statistica Sinica, 21:1591–1609, 2011.

[22] A. Dobra and S.E. Fienberg. The generalized shuttle algorithm, 2008.

[23] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using bayesian networks to analyze expression data. Journal of Computational Biology, 7:601–620, 2000.

[24] M. Frydenberg. The chain graph markov property. Scandinavian Journal of Statistics, 17:333–353, 1990.

[25] D. Gale. A theorem on flows in networks. Pacific Journal of Mathematics, 7:1073–1082, 1957.

[26] D. Geiger and J. Pearl. On the logic of causal models. Uncertainty in Artificial Intelligence 4 (R. D. Shachter, T. S. Lewitt, L. N. Kanal, J. F. Lemmer eds.), pages 3–14. North-Holland, 1990.

[27] D. Geiger and J. Pearl. Logical and algorithmic properties of conditional independence and graphical models. Annals of Statistics, 21:2001–2021, 1993.

[28] L. A. Goodman. Association models and the bivariate normal for contingency tables with ordered categories. Biometrika, 68:347–355, 1981.

[29] S. J. Haberman. A warning on the use of chi-squared statistics with frequency tables with small expected cell counts. J. Amer. Statist. Assoc., 83:555–560, 1988.

[30] H. Hara, A. Takemura, and Yoshida R. A markov basis for conditional test of common diagonal effect in quasi-independence model for square contingency tables. Computational Statistics and Data Analysis, 53:1006–1014, 2009.

[31] M. Huber. Fast perfect sampling from linear extensions. Discrete Mathematics, 306:420–428, 2006.

[32] X. Jiang, R.E. Neapolitan, M.M. Barmada, and S. Visweswaran. Learning genetic epistasis using bayesian network scoring criteria. BMC Bioinformatics, 12(89), 2011.

[33] V. Kaibel and M. Wolff. Simple 0/1-polytope. Europ. J. Combinatorics, 21:139–144, 2000.

[34] H. K. Kim and J. Y. Lee. Criteria of valid line sum arrays for multidimensional matrices, 2013.

[35] D. Krackhardt. Cognitive social structures. Social Networks, 9:109–134, 1987.

[36] S. L. Lauritzen. Mixed graphical association models. Scandinavian Journal of Statistics, 16:273–306, 1989.

[37] S. L. Lauritzen. Graphical Models. Clarendon Press, 1996.

[38] S. L. Lauritzen, A. P. Dawid, B. N. Larsen, and H. G. Leimer. Independence properties of directed markov fields. Networks, 20:491–505, 1990.

[39] Silvia Lindner. Discrete optimisation in machine learning - learning of Bayesian network structures and conditional independence implication. PhD thesis, Technische Universität München, 2012.

[40] P.J.F. Lucas. Bayesian model-based diagnosis. International Journal of Approximate Reasoning, 27(2):99–119, 2001.

[41] J. L. Massey. Causal interpretation of random variables (in russian). Problemy Peredachi Informatsii, 32:112–116, 1996.

[42] Hemmecke R., Lindner S., and Studený M. Learning restricted bayesian network structures. 2010.

[43] G. Rasch. Probabilistic models for some intelligence and attainment tests. Copenhagen: Danish Institute for Eucational Research, 1960.

[44] A. Roberts and L. Stone. Island-sharing by archipelago species. Oecologia, 83:560–567, 1990.

[45] H. J. Ryser. Combinatorial properties of matrices of zeros and ones. The Canadian journal of mathematics, 9:371–377, 1957.

[46] S. Sampson. Crisis in a cloister. unpublished doctoral dissertation, 1969.

[47] J. G. Sanderson. Testing ecological patterns. American Scientist, 88:332–339, 2000.

[48] A. Schrijver. Theory of Linear and Integer Programming. Wiley, 1998.

[49] M. A. Shwe, D. E. Heckerman, M. Henrion, H. P. Lehmann, and G. F. Cooper. Probabilistic diagnosis using a reformulation of the internist-1/qmr knowledge base: I. the probabilistic model and inference algorithms. Methods of Information in Medicine, 30:241–255, 1991.

[50] T. A. B. Snijders. Enumeration and simulation methods for $0-1$ matriceswith given marginals. Psychometrika, 56:397–417, 1991.

[51] M. Studený. Probabilistic Conditional Independence Structures. Springer Verlag, 2005.

[52] M. Studený, R. Hemmecke, and S. Lindner. Characteristic imset: a simple algebraic representative of a bayesian network structure. In Proceedings of the 5th European Workshop on Probabilistic Graphical Models, pages 257–264, 2010.

[53] M. Studený and R. R. Rouckaert. On chain graph models for description of conditional independence structures. Annals of Statistics, 26:1434–1495, 1998.

[54] M. Studený, J. Vomlel, and R. Hemmecke. A geometric view on learning bayesian network structures. International Journal of Approximate Reasoning, 51(5):573–586, 2010.

[55] A. Takemura and R. Yoshida. A generalization of the integer linear infeasibility problem. Discrete Optimization, 5:36–52, 2008.

[56] R Project Team. R project. GNU software. Available at http://www.r-project.org/, 2011.

[57] J. Uebersax. Pgenetic counseling and cancer risk modeling: An application of bayes nets. marbella. Spain: Ravenpack International, 2004.

[58] J. Vomlel, H. Kružík, P. T ˚ uma, J. Přeček, and M. Hutyra. Machine learning methods for mortality prediction in patients with st elevation myocardial infarction. Proceedings of WUPES 2012, pages 204–213, 2012.

[59] G. M. Ziegler. Lectures on Polytopes. Springer Verlag, New York, New York, 1994.

[1] [2] [3] [4] [6] [7] [9] [10] [8] [11] [12] [13] [14] [15] [16] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28]
[29] [30] [42] [31] [32] [33] [34] [35] [36] [38] [37] [39] [41]
[43] [44] [45] [46] [47] [48] [49] [50] [5] [53] [51] [52] [54] [55] [58] [57] [59]

Vita

Jing Xi

University of Kentucky Department of Statistics

Place of Birth: Nan-Chang, Jiang-Xi Province, China

Education

Master of Science in Statistics, University of Kentucky, 2011

Bachelor of Science in Statistics, University of Science and Technology of China, 2008

Employment

Research Assistant *Spring 2012 – Fall 2013*

CPH (College of Public Health) Consulting Lab, University of Kentucky

Research Assistant *Spring 2011 – Fall 2011*

Department of Statistics, University of Kentucky

Teaching Assistant *Fall 2008 – Fall 2010*

Department of Statistics, University of Kentucky

Selected Publications

---

1. Xi, J., Yoshida R. and Haws, D. (2012). Estimating the number of zero-one multi-way tables via sequential importance sampling. Annals of the Institute of Statistical Mathematics (AISM), In-Press
2. Xi, J., Yoshida R. and Hemmecke, R. (2012). The characteristic imset polytope for diagnosis models and a generalization. Preprint, Available at `http:`

//arxiv.org/abs/1206.0406.