© 2017 by Yinghan Chen. All rights reserved.

SAMPLING FOR NETWORK MOTIF DETECTION AND ESTIMATION OF Q-MATRIX AND LEARNING TRAJECTORIES IN DINA MODEL

BY

YINGHAN CHEN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Statistics
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2017

Urbana, Illinois

Doctoral Committee:

Professor Yuguo Chen, Chair, Co-Director of Research
Associate Professor Steven Culpepper, Co-Director of Research
Professor Jeffrey Douglas
Assistant Professor Georgios Fellouris

Abstract

Monte Carlo methods provide tools to conduct statistical inference on models that are difficult or impossible to compute analytically and are widely used in many areas of statistical applications, such as bioinformatics and psychometrics. This thesis develops several sampling algorithms to address open issues in network analysis and educational assessments.

The first problem we investigate is network motif detection. Network motifs are substructures that appear significantly more often in the given network than in other random networks. Motif detection is crucial for discovering new characteristics in biological, developmental, and social networks. We propose a novel sequential importance sampling strategy to estimate subgraph frequencies and detect network motifs. The method is developed by sampling subgraphs sequentially node by node using a carefully chosen proposal distribution. The method generates subgraphs from a distribution close to uniform and performs better than competing methods. We apply the method to four real-world networks and demonstrate outstanding performance in practical examples.

The other two issues are related to educational measurement in psychometrics. Cognitive diagnosis models (CDMs) are partially ordered latent class models to classify students into skill mastery profiles. In educational assessment, these models help researchers analyze students' mastery of skills and learning process based on their responses to test items. The deterministic inputs, noisy "AND" gate model (DINA) is a popular psychometric model for cognitive diagnosis. We investigate the estimation of Q-matrix in DINA model. Q matrix is a binary matrix which maps the test item to its corresponding required attributes. We propose a Bayesian framework for estimating the DINA Q matrix. The proposed algo-

rithms ensure that the estimated Q matrices always satisfy the identifiability constraints. We present Monte Carlo simulations to support the accuracy of parameter recovery and apply our algorithms to Tatsuoka's fraction-subtraction dataset.

The last project is related to the recovery of learning process. The increasing presence of electronic and online learning resources presents challenges and opportunities for psychometric techniques that can assist in the measurement of abilities and even hasten their mastery. CDMs can assist in carefully navigating through the training and assessment of these skills in e-learning applications. We propose a class of CDMs for modeling changes in attributes, which we refer to as learning trajectories. We focus on the development of Bayesian procedures for estimating parameters of a first-order hidden Markov model and apply the developed model to a spatial rotation experimental intervention.

Table of Contents

Chapter 1 Introduction **1**

Chapter 2 Network Motif Detection **3**
2.1 Introduction 3
2.2 Subgraph Concentration 5
2.3 Sequential Importance Sampling of Subgraphs 7
2.4 Motif Detection 17
2.5 Applications 19
2.6 Discussion 30

Chapter 3 Q Matrix Estimation **32**
3.1 Introduction 32
3.2 Bayesian Model Formulation 34
3.3 Monte Carlo Simulation Studies 48
3.4 Application to Fraction-Subtraction Data 55
3.5 Discussion 59

Chapter 4 Learning trajectories in dynamic CDMs **62**
4.1 Introduction 62
4.2 Learning Trajectories 65
4.3 Modeling Learning Trajectories 68
4.4 Monte Carlo Simulation Study 76
4.5 Application to Spatial Rotation Skill Acquisition 78
4.6 Discussion 83

References **88**

Chapter 1

Introduction

A (0,1) matrix is an integer matrix whose elements are either 1 or 0. It appears in various kinds of settings and can be used to represent binary relationships, such as adjacency matrix of a graph, design matrix of specific tests or surveys, or binary relationships between two type of objects. In particular, this thesis considers two particular areas related to (0,1) matrices: 1) networks and 2) diagnostic assessments.

A network can be expressed by an adjacency matrix and the distributions of such matrix and its subgroups are of great interest, because they reveal the facts about networks with specific structures. However, the probabilistic features of substructures of the adjacency matrix are difficult to compute. When a true distribution is unknown or difficult to compute, Monte Carlo simulation provides an efficient tool to help conduct inference on the distribution of interest. For instance, sequential importance sampling algorithm develops a way to obtain statistics of the target distribution from a biased proposal distribution that is constructed sequentially. It is especially helpful when sampling from the original distribution is not feasible. In the research on network substructures, subgraph frequencies are of interest in network motif detection, however, sampling connected subgraphs uniformly is not feasible. Therefore, in Chapter 2 we propose a sequential importance sampling algorithm to draw subgraphs through connected nodes and estimate the subgraph frequencies from the proposed sampling method.

As for diagnostic assessment, (0,1) matrix, referred to as the Q-matrix, can represent the relationship between test items and latent attributes. The Q-matrix is used to design assessments and to diagnose subjects' skill profiles. However, the true design matrix and real

relations of subjects' skill acquisitions in an assessment usually remain unknown. Cognitive Diagnosis models (CDMs) are partially latent class models focusing on discrete-defined attributes, and they provide researchers efficient tools to diagnose the information regarding subjects' attributes. Inference on the item parameters and the latent classes of CDMs is also difficult to compute explicitly. When the target multi-dimensional distribution cannot be easily computed, Markov chain Monte Carlo methods are widely used to sample from the target distribution by constructing a Markov chain that converges to the stationary distribution. Therefore, we can compute the desired statistics from CDMs through Markov chain Monte Carlo methods. In Chapter 3, we propose a Bayesian framework for a popular Cognitive Diagnosis model, DINA model, and estimate the underlying Q-matrix and other item parameters through Gibbs and Metropolis sampling algorithms. In Chapter 4, we propose several models to interpret the transition of skill acquisitions in the longitudinal setting and estimate the transition matrix and all other item parameters through Gibbs sampling.

Chapter 2

Network Motif Detection

2.1 Introduction

In real-world networks, some patterns of subgraphs occur more frequently than would be expected in random networks. Milo et al. (2002) called these subgraphs network motifs. The frequency of these motifs contains important information about the structure and function of the network. For example, in regulatory networks, motifs describe the underlying characteristics of the transcription and translation processes, and in neural networks, motifs help interpret the operation of groups of neurons (Shen-Orr et al., 2002; Milo et al., 2002). Network motifs have also been found to be important in networks of signal transition, developmental transcription, and other types of networks (Milo et al., 2004).

To detect network motifs, both the frequency of occurrence of a subgraph in an observed network and the frequency of occurrence of the same subgraph in random networks need to be calculated. For determining subgraph frequency, an exhaustive enumeration algorithm was proposed by Milo et al. (2002) to count all subgraphs with k nodes. Wernicke (2006) also proposed an exhaustive enumeration algorithm that guarantees that every subgraph will be visited only once. Itzhack et al. (2007) proposed an exhaustive enumeration method which requires specification of the shapes of the subgraph patterns and counts the number of subgraphs using these shapes. Although the method of Itzhack et al. (2007) runs faster than previous enumeration methods for up to size 5 subgraphs, it struggles as the size of the subgraph becomes larger.

A more efficient enumeration method named *Kavosh* was proposed by Kashani et al.

(2009). This method detects network motifs by counting trees rooted at a particular node, and then removing that node after exhaustively enumerating all subgraphs containing that node. Later, Ribeiro and Silva (2010) created a new data structure called *G-Tries*, which improved the method efficiency at the cost of greater memory usage. Recently, Lin et al. (2015) employed Graphical Processing Units (GPU) to parallelize subgraph matching tasks, and the GPU-based enumeration runs faster than most CPU-based approaches. Wang et al. (2016) proposed a method to infer subgraph concentrations up to size 5 through enumerating subgraphs in a resampled graph from the given graph. The method does not require the knowledge of the entire graph and can be applied to graphs with missing information.

All the aforementioned methods exhaustively enumerate all possible subgraphs of a certain size in a given network, and can therefore be considered to be network-centric. Grochow and Kellis (2007) proposed a motif-centric technique that enumerates a single subgraph pattern in the network using symmetric-breaking. Motif-centric enumeration methods run faster than network-centric enumeration methods for counting a single target subgraph, but are inconvenient to apply to all possible subgraph patterns.

In addition to the exhaustive enumeration methods, subgraph frequencies can be estimated using randomly sampled subgraphs. Kashtan et al. (2004) proposed an importance sampling method to sample subgraph patterns sequentially by edge and estimate the concentrations of subgraph patterns in the network. Wernicke (2006) also proposed a uniform sampling method based on their exhaustive enumeration technique. A probability is assigned to each leaf of the enumeration tree to determine whether or not a leaf will be visited, so only a subset of all possible subgraphs are counted. Recently, Chen et al. (2016) proposed a random walk framework for the given graph and took consecutive steps of the random walk to generate subgraphs of any size. However, random walk-based methods require a tuning parameter to decide which subgraph relationship topology should be used, and usually it requires crawling the entire graph to construct the subgraph relation graphs.

For large networks, exhaustive enumeration methods are challenged by computing time

and memory cost, hence sampling methods are preferred. The importance sampling method of Kashtan et al. (2004) needs to compute the importance weights of all subgraphs and all potential permutations of edge sequences. The uniform sampling method of Wernicke (2006) is based on the exhaustive search tree and needs to complete the tree even if some branches are never visited. The random walk-based generator of Chen et al. (2016) requires the construction of subgraph relationship graphs, and it might generate invalid subgraphs. To improve the existing sampling methods, we develop a new sequential importance sampling method that samples subgraphs by nodes. The new proposal distribution is closer to the uniform distribution and the importance weight can be calculated easily with a simple formula.

This chapter is organized as follows. Section 2.2 gives the basic definitions of subgraph concentration. Section 2.3 presents the new sequential sampling algorithm. Section 2.4 discusses the motif detection strategy. Section 2.5 provides numerical results on several real networks. Section 2.6 provides concluding remarks.

2.2 Subgraph Concentration

In a directed or undirected graph $G(V, E)$, where V is the set of nodes and E is the set of edges, an induced subgraph with k nodes $H(V^*, E^*)$ is defined as a subset, V^*, of k nodes from V and a subset, E^*, of edges from E with both endpoints in V^*. Here we assume that there are no self-loops and at most two opposite directed edges between a pair of nodes. We refer to a connected, induced subgraph with k nodes as a size-k subgraph. Denote the set of all size-k subgraphs in G as $S_k(G)$. For example, the graph G in Figure 2.1 contains 7 nodes and 8 directed edges. Nodes {2,5,6} and the 3 directed edges connecting them form an induced subgraph of size 3. There are 8 size-3 subgraphs in this given graph, i.e., $|S_3(G)| = 8$, and they are {1,2,3}, {1,2,5}, {1,3,4}, {2,3,4}, {2,3,5}, {2,5,6}, {2,5,7}, and {5,6,7}.

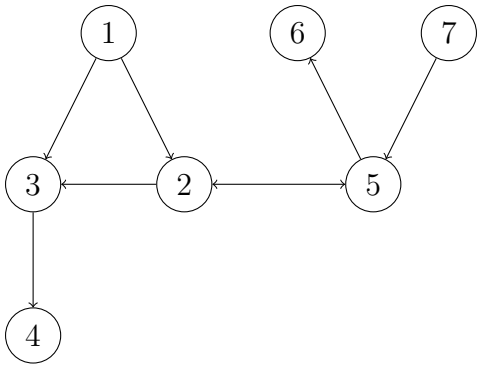


Figure 2.1: A graph with 7 nodes and 8 directed edges

If there exists a bijective mapping between a pair of subgraphs that preserves the adjacency of nodes and direction of edges, then the two subgraphs are topologically equivalent and differ only in the labeling of the nodes. For example, among the four size-3 subgraphs from the example in Figure 2.1 that are listed in Figure 2.2, subgraphs (a) and (b) are topologically equivalent because they have the same structure ignoring the node label. Subgraphs (c) and (d) are also topologically equivalent. However subgraphs (a) and (c) are not topologically equivalent because the direction of the edges cannot be preserved after any bijective mapping.

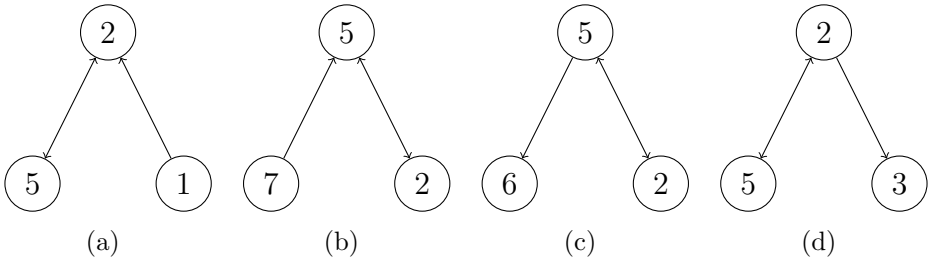


Figure 2.2: Illustration of isomorphic subgraphs

All topologically equivalent subgraphs belong to the same isomorphic class. Let I be the number of isomorphic classes for size-k subgraphs in G, and $S_k^i(G)$, $1 \leq i \leq I$, be the ith isomorphic class. For any induced size-k subgraph $H \in S_k(G)$, we have $H \in S_k^i(G) \subset S_k(G)$

for some i. The concentration of a subgraph isomorphic class in graph G is

$$\mathcal{C}_k^i(G) = \frac{\left|S_k^i(G)\right|}{\left|S_k(G)\right|} = \mathbf{E}_\pi\left[\mathbb{1}\{H \in S_k^i(G)\}\right], \tag{2.1}$$

where π is the uniform distribution over all possible size-k subgraphs, i.e., $\pi(H) = 1/|S_k(G)|$ for $H \in S_k(G)$. Network motifs are defined as subgraph patterns with a subgraph concentration $\mathcal{C}_k^i(G)$ significantly larger in the observed G than in random networks with the same degree sequence. The first task of detecting network motifs is therefore to estimate $\mathcal{C}_k^i(G)$, and then we will conduct significance test on the concentrations to find motifs.

2.3 Sequential Importance Sampling of Subgraphs

Simulating directly from the uniform distribution $\pi(H)$ is not easy. However, if we can simulate a subgraph H from a carefully chosen proposal distribution $q(\cdot)$, whose support includes $S_k(G)$, then the concentration $\mathcal{C}_k^i(G)$ can be written as

$$\mathcal{C}_k^i(G) = \sum_{H \in S_k(G)} \frac{\mathbb{1}\{H \in S_k^i(G)\}}{|S_k(G)|} = \sum_{H \in S_k(G)} \frac{\mathbb{1}\{H \in S_k^i(G)\}\pi(H)q(H)}{q(H)} = \mathbf{E}_q\left[\mathbb{1}\{H \in S_k^i(G)\}\frac{\pi(H)}{q(H)}\right], \tag{2.2}$$

and estimated by

$$\hat{\mathcal{C}}_k^i(G) = \frac{\sum_{j=1}^N \mathbb{1}\{H_j \in S_k^i(G)\}w_j}{\sum_{j=1}^N w_j} = \frac{\sum_{j:H_j \in S_k^i(G)} \pi(H_j)q(H_j)^{-1}}{\sum_{j:H_j \in S_k(G)} \pi(H_j)q(H_j)^{-1}} = \frac{\sum_{H_j \in S_k^i(G)} q(H_j)^{-1}}{\sum_{H_j \in S_k(G)} q(H_j)^{-1}}, \tag{2.3}$$

where $H_1, \ldots, H_N$ are independent and identically distributed (i.i.d.) samples from $q(\cdot)$, and $w_j = \pi(H_j)/q(H_j)$ is the importance weight.

The performance of the importance sampling method can be assessed using the *effective sample size*, $\mathrm{ESS} = N/(1 + \mathrm{cv}^2)$, where the *coefficient of variation (cv)* is

$$\mathrm{cv}^2 = \frac{\mathrm{var}_q(\pi(H)/q(H))}{E_q^2(\pi(H)/q(H))}. \tag{2.4}$$

The cv^2 can be estimated by its sample version. A small cv^2 indicates that we are sampling from a proposal distribution that is close to our desired target distribution π.

Our importance sampling procedure will be to sample a subgraph sequentially by leveraging the subgraph structure. We will sample node by node, conditional on the realization of the previous nodes, until a size-k subgraph is obtained.

To motivate the construction of our proposal distribution $q(\cdot)$, we begin by describing the process of sampling from the true uniform distribution $\pi(H)$ sequentially by node. Let $S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\} := \{H : H \in S_k(G),\ \{v_{i_1}, \ldots, v_{i_j}\} \subset H\}$ be the set of all size-k subgraphs containing nodes $v_{i_1}, \ldots, v_{i_j}$. Let V_c be the set of nodes that have already been sampled. Before any nodes have been sampled $V_c = \emptyset$. To sample a size-k subgraph from $\pi(H)$, the first node should be sampled with probability proportional to $|S_k(G) \cap \{v_i\}|$, $i = 1, \ldots, |V|$. Suppose the first node sampled is v_{i_1}, then $V_c = \{v_{i_1}\}$, and the second node should be sampled with probability proportional to $|S_k(G) \cap \{V_c \cup \{v_i\}\}|$, $i \neq i_1$. The procedure continues until a size-k subgraph is obtained, updating V_c after each node has been sampled. Suppose the nodes of the final size-k subgraph are sampled in the order $(v_{i_1}, \ldots, v_{i_k})$, then

$$
\begin{aligned}
& P(\text{a size-}k \text{ subgraph sampled in the order } (v_{i_1}, \ldots, v_{i_k})) \\
= \;& P(v_{i_1} \text{ is selected}) P(v_{i_2} \text{ is selected} | v_{i_1} \text{ is selected}) \cdots P(v_{i_k} \text{ is selected} | v_{i_1}, \ldots, v_{i_{k-1}} \text{ are selected}) \\
= \;& \frac{|S_k(G) \cap \{v_{i_1}\}|}{\sum_i |S_k(G) \cap \{v_i\}|} \times \frac{|S_k(G) \cap \{v_{i_1}, v_{i_2}\}|}{\sum_{i \neq i_1} |S_k(G) \cap \{v_{i_1}, v_i\}|} \times \cdots \times \frac{|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_{k-1}}, v_{i_k}\}|}{\sum_{i \neq i_1, \ldots, i_{k-1}} |S_k(G) \cap \{v_{i_1}, \ldots, v_{i_{k-1}}, v_i\}|} \\
= \;& \frac{|S_k \cap \{v_{i_1}\}|}{k \cdot |S_k(G)|} \times \frac{|S_k(G) \cap \{v_{i_1}, v_{i_2}\}|}{(k-1) \cdot |S_k(G) \cap \{v_{i_1}\}|} \times \cdots \times \frac{1}{|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_{k-1}}\}|} \\
= \;& \frac{1}{k! \cdot |S_k(G)|}. \qquad (2.5)
\end{aligned}
$$

The probability of sampling the first node v_{i_1} is $\frac{|S_k(G) \cap \{v_{i_1}\}|}{\sum_i |S_k(G) \cap \{v_i\}|}$, which equals to $\frac{|S_k(G) \cap \{v_{i_1}\}|}{k \cdot |S_k(G)|}$ because $\sum_{i=1}^{|V|} |S_k(G) \cap \{v_i\}| = k \cdot |S_k(G)|$. The probability for sampling other nodes can be derived similarly. Notice that the subgraph $\{v_{i_1}, \ldots, v_{i_k}\}$ could be obtained by sampling

these k nodes in any of the $k!$ orders, and each specific order will be sampled with probability $1/(k! \cdot |S_k(G)|)$, therefore, the probability of obtaining the subgraph $\{v_{i_1}, \ldots, v_{i_k}\}$ (without specifying the order) is

$$P(\text{sampling a size-}k \text{ subgraph } \{v_{i_1}, \ldots, v_{i_k}\}) = k! \times \frac{1}{k! \cdot |S_k(G)|} = \frac{1}{|S_k(G)|}. \tag{2.6}$$

So this procedure will yield subgraphs sampled from the uniform distribution $\pi(H)$.

The example in Figure 2.1 can help explain how the sequential procedure works. Suppose we are interested in size-3 subgraphs, and as mentioned before, $|S_3(G)| = 8$, then the target uniform distribution of sampling any size-3 subgraph from the given G is 1/8. The marginal probability of each node equals to the proportion of subgraphs containing it. So the first node will be sampled from $\{1, 2, 3, 4, 5, 6, 7\}$ with probability proportional to 3, 6, 4, 2, 5, 2, and 2 respectively. Suppose node 1 is selected, which occurs with probability 3/24, then the second node will be sampled from $\{2, 3, 4, 5, 6, 7\}$ with probability proportional to 2, 2, 1, 1, 0, and 0 respectively. Suppose node 2 is selected, which occurs with probability 2/6, then the third node will be sampled from $\{3, 4, 5, 6, 7\}$ with probability proportional to 1, 0, 1, 0, and 0 respectively. Suppose node 3 is selected, which occurs with probability 1/2. So the probability of sampling the subgraph $\{1, 2, 3\}$ in the order $(1, 2, 3)$ is $3/24 \times 2/6 \times 1/2 = 1/48$. It is easy to derive that the probability of sampling the subgraph $\{1, 2, 3\}$ in the order $(2, 1, 3)$, or any other order, is also 1/48. So the final probability of sampling the subgraph $\{1, 2, 3\}$ is $3! \times 1/48 = 1/8$, and the sample is drawn from the uniform distribution over $S_3(G)$.

Sampling subgraphs uniformly requires the quantity $|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\}|$ which is difficult to calculate in practice. Our importance sampling proposal distribution will be constructed by developing reasonable approximations to $|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\}|$.

2.3.1 Proposal distribution

Spanning tree

We use the spanning tree to approximate $|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\}|$. A spanning tree of a connected graph G is a subgraph of G that is a tree containing all nodes of G. Every connected graph G with m nodes $\{v_1, \ldots, v_m\}$ contains a spanning tree that is rooted at v_i, $1 \leq i \leq m$. The depth of a node in this spanning tree is defined as the number of edges from that node to the root v_i, i.e., the length of its path to the root. The level of a node is (1+ the depth). So the root node v_i has depth 0 and level 1. In this section we construct the spanning tree in the following way to avoiding over-counting the number of subgraphs containing specific nodes. We require that 1) a node is a child of v_i if and only if it is directly connected to v_i in the given graph G, and 2) every node is at the depth of its shortest path to v_i if multiple paths to v_i exist in the original graph G. This construction will allow us to derive an approximation of $|S_k(G) \cap \{v_i\}|$ by counting the number of k-node subtrees rooted at node v_i in the spanning tree.

For example, two spanning trees of the graph in Figure 2.1 are provided in Figure 2.3, with Figure 2.3(a) rooted at node 1 and Figure 2.3(b) rooted at node 2. The number of size-3 subgraphs containing node 1 in the original graph in Figure 2.1 is $|S_3(G) \cap \{1\}| = 3$, and we can approximate this number by the number of size-3 subtrees starting from node 1 in Figure 2.3(a) which is also 3 in this case. Similarly $|S_3(G) \cap \{2\}| = 6$ in Figure 2.1, and the number of size-3 subtrees starting from node 2 in Figure 2.3(b) is also 6. This approximation often works well in practice although it is not always exact. For example, if nodes 4 and 6 are connected in Figure 2.1, then the spanning tree rooted at node 2 is still the same as Figure 2.3(b), but we would miss subgraphs {2,3,4,6} and {2,4,5,6} when counting size-4 subtrees starting from node 2 in the spanning tree.

This procedure of counting subtrees in the spanning tree will be used to approximate $|S_k(G) \cap \{v_i\}|$ and to develop our proposal distribution for sequential importance sampling.

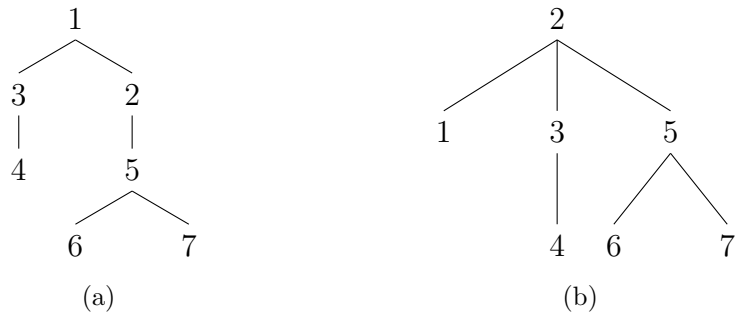


Figure 2.3: Two spanning trees of the graph in Figure 2.1

We propose a recursive formula for the approximate number of subtrees that ignores aspects of the subtree structure for computational simplicity. Moreover, in the special case of $k = 3$, we can construct a spanning tree where the exact number of such subtrees is the same as the number of corresponding subgraphs. We will discuss the special case in Section 2.3.3.

Recursive formula

Suppose the degree of node v_i is d_i in an undirected graph G, i.e., there are d_i nodes adjacent to node v_i. If the graph is directed, we refer d_i to the number of nodes adjacent to node v_i, ignoring direction of edges. Then in the spanning tree of G rooted at node v_i, there are d_i nodes at level 2 (see Figure 2.4). The set of all size-k subgraphs containing node v_i, $S_k(G) \cap \{v_i\}$, can be partitioned into subsets $S_k(G) \cap \{v_i^{(j)}\}$, $j = 1, \ldots, k-1$, where $S_k(G) \cap \{v_i^{(j)}\}$ is the set of all size-k subgraphs containing v_i with exactly j nodes directly connected to v_i. Clearly, $|S_k(G) \cap \{v_i\}| = \sum_{j=1}^{k-1} |S_k(G) \cap \{v_i^{(j)}\}|$. If a reasonable approximation $\mathcal{N}_k^{(j)}(d_i)$ to $|S_k(G) \cap \{v_i^{(j)}\}|$ can be obtained, then $|S_k(G) \cap \{v_i\}|$ can be approximated using $\mathcal{N}_k(d_i) = \sum_{j=1}^{k-1} \mathcal{N}_k^{(j)}(d_i)$. As discussed before we can choose $\mathcal{N}_k^{(j)}(d_i)$ to be the approximate number of subtrees containing node v_i with j nodes at depth 1 in the spanning tree of G rooted at v_i.

When $j = k-1$, subtrees of interest will have $k-1$ nodes at depth 1 (as shown in Figure 2.5(a)). There are $\binom{d_i}{k-1}$ such subtrees in the spanning tree, so $\mathcal{N}_k^{(k-1)}(d_i) = \binom{d_i}{k-1}$.

When $j = k-2$, the subtrees have $k-2$ nodes at depth 1 and 1 node at depth 2 (as

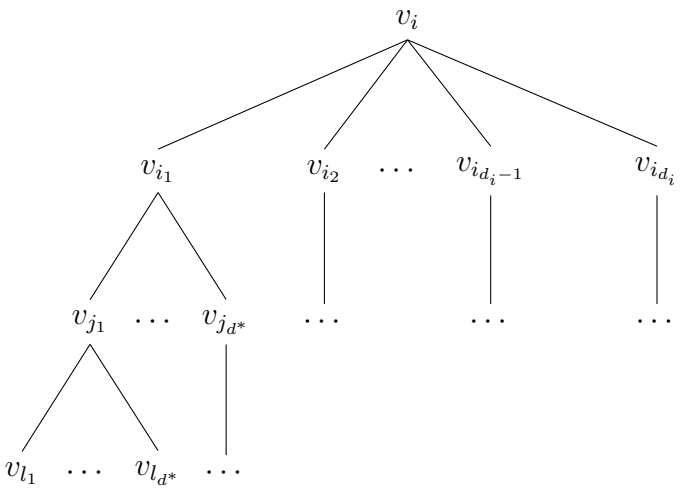
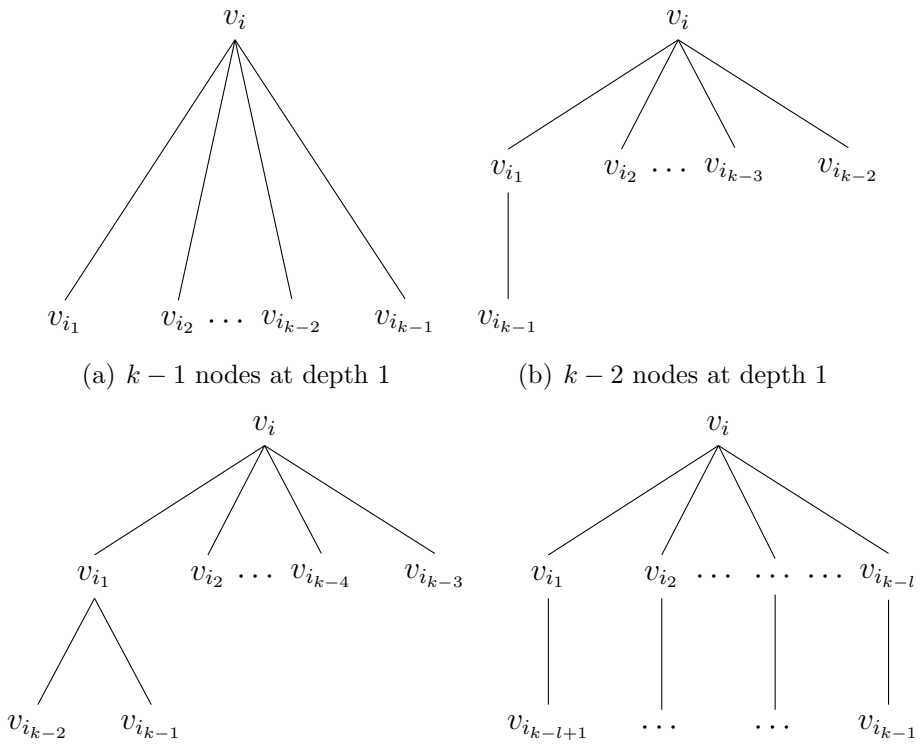


Figure 2.4: An illustration of the approximating spanning tree

shown in Figure 2.5(b)). There are $\binom{d_i}{k-2}$ ways to choose nodes at depth 1 and the number of candidate nodes at depth 2 equals to the number of children of the $k-2$ nodes at depth 1. Because each node may have a different number of children, to simplify the formula, we approximate the degree of each node (except v_i) by the average degree of the graph. In particular, denote by $\bar{d} = 2|E|/|V|$ the average of degree sequence. We use $\bar{d}$ to approximate the degree of all nodes but v_i, so all nodes except v_i will have approximately $d^* = \bar{d}-1$ nodes as their children. Figure 2.4 illustrates the approximate structure of a spanning tree in G. Based on this approximation, the number of candidate notes at depth 2 is approximately $(k-2)d^*$. So in total there are approximately $\binom{d_i}{k-2} \times (k-2)d^*$ subtrees from the rooted tree when $j = k-2$. i.e., $\mathcal{N}_k^{(k-2)}(d_i) = \binom{d_i}{k-2} \times (k-2)d^*$.

More generally, when $j = k-l$ for $1 < l < k$, the subtrees of interest will have $k-l$ nodes at depth 1 (as shown in Figure 2.5(d)). There are $\binom{d_i}{k-l}$ ways to choose candidate nodes at depth 1. If the root node v_i and the $k-l$ nodes at depth 1 are collapsed to form a virtual node $\tilde{v}$, then $\tilde{v}$ has degree approximately $(k-l)d^*$ at the next depth, and the choice of the remaining $l-1$ nodes in this tree will be equivalent to choosing a l-node subtree from a spanning tree rooted at $\tilde{v}$ with degree $(k-l)d^*$. The number of such l-node subtrees can be approximated by $\mathcal{N}_l((k-l)d^*)$. Then the approximate number of subtrees when $j = k-l$



(a) $k-1$ nodes at depth 1 (b) $k-2$ nodes at depth 1

(c) $k-3$ nodes at depth 1 (d) $k-l$ nodes at depth 1

Figure 2.5: Illustration of size-k subtrees

is equal to $\binom{d_i}{k-l}\mathcal{N}_l((k-l)d^*)$, i.e., $\mathcal{N}_k^{(j)}(d_i) = \binom{d_i}{j}\mathcal{N}_{k-j}(jd^*)$.

Therefore the number of size-k connected subgraphs that contain node v_i of degree d_i can be approximated recursively by $\mathcal{N}_{k-1}(d), \cdots, \mathcal{N}_1(d)$:

$$|S_k(G) \cap \{v_i\}| \approx \mathcal{N}_k(d_i) = \sum_{j=1}^{k-1} \mathcal{N}_k^{(j)}(d_i) = \sum_{j=1}^{k-1} \binom{d_i}{j} \mathcal{N}_{k-j}(jd^*). \tag{2.7}$$

Obviously, $\mathcal{N}_1(d) = 1$, $\mathcal{N}_2(d) = d$, and $\mathcal{N}_3(d) = \binom{d}{2} + \binom{d}{1}\mathcal{N}_2(d^*) = \binom{d}{2} + dd^*$.

By using the idea of collapsing existing nodes as one virtual node, we can also approximate $|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\}|$ by counting rooted trees that are rooted at the virtual node, $\tilde{v}$, representing $\{v_{i_1}, \ldots, v_{i_j}\}$, i.e., $|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\}| = |S_{k-j+1}(G) \cap \{\tilde{v}\}|$, $1 < j < k$. The degree of the virtual node $\tilde{v}$ (after collapsing nodes $v_{i_1}, \ldots, v_{i_j}$) is $\tilde{d} = |\bigcup_{1 \le l \le j} V_{i_l} \setminus \{v_{i_1}, \ldots, v_{i_j}\}|$, where V_{i_l} is the set of nodes that are connected to v_{i_l}. We know the number of $(k-j+$

1)-node subtrees from the spanning tree rooted at the virtual node $\tilde{v}$ with degree $\tilde{d}$, i.e., $|S_{k-j+1}(G) \cap \{\tilde{v}\}|$, can be approximated by $\mathcal{N}_{k-j+1}(\tilde{d})$, so we could approximate the number of size-k subgraphs that contain nodes $\{v_{i_1}, \ldots, v_{i_j}\}$ in the same way as well, i.e.,

$$|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\}| \approx \mathcal{N}_{k-j+1}(\tilde{d}). \tag{2.8}$$

2.3.2 Sampling algorithm

We sample a size-k subgraph sequentially node by node using a proposal distribution close to the target uniform distribution. As shown in (2.5), the true probability of sampling node v_{i_j} given $v_{i_1}, \ldots, v_{i_{j-1}}$ have already been sampled is

$$|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_{j-1}}, v_{i_j}\}| / \sum_{i \neq i_1, \ldots, i_{j-1}} |S_k(G) \cap \{v_{i_1}, \ldots, v_{i_{j-1}}, v_i)\}|.$$

Although $|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_j}\}|$ is hard to obtain in practice, we use the approximation in (2.7) and (2.8) to construct our proposal to sample the nodes. As we sequentially sample the nodes, instead of considering all nodes that have not been sampled as the candidate for the next node v_{i_j}, we focus on the nodes that are connected to the set of sampled nodes $\{v_{i_1}, \ldots, v_{i_{j-1}}\}$. This is equivalent to approximating $|S_k(G) \cap \{v_{i_1}, \ldots, v_{i_{j-1}}, v_i\}|$ by 0 if v_i is not connected to $\{v_{i_1}, \ldots, v_{i_{j-1}}\}$. This approach will guarantee that the final size-k subgraph we obtain satisfies the connectedness requirement and avoid wasting samples. On the other hand, every connected size-k subgraph can be sampled in one or more orders so that the subgraphs obtained in the intermediate steps are always connected. So this modification of the algorithm will not miss any connected size-k subgraphs.

Our sampling procedure is described in Algorithm 1. Recall V_c is the set of sampled nodes. Let V_{ext} denote the set of candidate nodes that are connected to the sampled nodes in V_c. Let q denote the sampling probability of the nodes which can be computed sequentially.

Without loss of generality, assume the sampled size-k subgraph is $H = \{v_1, \ldots, v_k\}$.

Algorithm 1 Sampling algorithm for generating a size-k subgraph from $G(V, E)$

Initialize $V_c = \emptyset$, $V_{ext} = V$, $l = 0$, $q = 1$.
while $l < k$ **do**
  for all $v \in V_{ext}$ **do**
    $d(v)$ = number of nodes that are connected to v or V_c but not in $V_c \cup \{v\}$.
  end for
  Sample a node v_c from V_{ext} with probability proportional to $\mathcal{N}_{k-l}(d(v))$, $v \in V_{ext}$.
  Update the sequential sampling probability $q = q \times \frac{\mathcal{N}_{k-l}(d(v_c))}{\sum_{v \in V_{ext}} \mathcal{N}_{k-l}(d(v))}$.
  Update $V_c = V_c \cup \{v_c\}$; V_{ext} = set of nodes that are connected to V_c but not in V_c.
  $l = l + 1$.
end while

Suppose the k nodes are generated in the order $(v_1, \ldots, v_k)$. Then the algorithm outputs a size-k subgraph $H = \{v_1, \ldots, v_k\}$ and the associated probability $q(v_1, \ldots, v_k)$ of generating H in this particular order. As discussed in the beginning of Section 2.3, subgraph H could be obtained by sampling these k nodes in different orders, so the final probability of obtaining the subgraph H (without specifying the order) should sum over the probability of generating H in all possible orders, i.e.,

$$q(H) = \sum_{(i_1,\ldots,i_k) \in \mathcal{P}(1,\ldots,k)} q(v_{i_1}, \ldots, v_{i_k}), \tag{2.9}$$

where $\mathcal{P}(1, \ldots, k)$ is the set of all permutations of $(1, \ldots, k)$. In our algorithm, we sequentially add nodes that are connected to existing nodes, so it may not be feasible to generate the nodes in certain orders. For example, the subgraph $\{2, 5, 6\}$ in Figure 2.1 cannot be sampled in the order $(2, 6, 5)$ and $(6, 2, 5)$ because nodes 2 and 6 are not connected. Therefore the sum in (2.9) is only over the orders that are possible with Algorithm 1.

When considering the true uniform sampling of subgraphs in (2.5) and (2.6), each specific order of the nodes is sampled with the same probability. However in our sequential proposal, different orders of the nodes will be sampled with different probabilities. For example, the probabilities of sampling the subgraph $\{1, 2, 3\}$ in Figure 2.1 in the order $(1, 2, 3)$ and $(2, 1, 3)$ are $1/32$ and $1/36$, respectively. Therefore, the sum in (2.9) cannot be simplified to the

product of the number of possible orders and $q(v_1, \ldots, v_k)$.

Repeating the algorithm N times will give N i.i.d. subgraphs $H_1, \ldots, H_N$, and the sampling probabilities $q(H_i)$ can be computed using (2.9). Based on these samples and sampling probabilities, we can estimate the subgraph concentration using (2.3).

2.3.3 Special case for size-3 subgraphs

For size-3 subgraphs, the furthest depth is 2, so we can easily construct a spanning tree to represent every size-3 subgraph by one and only one 3-node subtree. Without loss of generality, assume the size-3 subgraph is $\{v_1, v_2, v_3\}$. The set of all size-3 subgraphs containing node v_1, $S_3(G) \cap \{v_1\}$, can be partitioned into subsets $S_3(G) \cap \{v_1^{(1)}\}$ and $S_3(G) \cap \{v_1^{(2)}\}$, where $S_3(G) \cap \{v_1^{(j)}\}$ is the set of all size-3 subgraphs containing v_1 with exactly j nodes directly connected to v_1. Figures 2.6(a) and (b) represent the subtree pattern of the subsets $S_3(G) \cap \{v_1^{(2)}\}$ and $S_3(G) \cap \{v_1^{(1)}\}$, respectively.

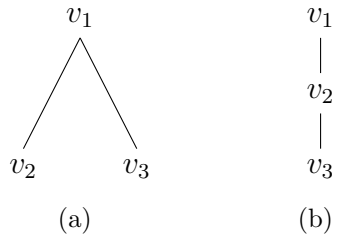


Figure 2.6: Possible patterns of 3-node trees rooted at v_1

Because of the relatively simple structure of size-3 subtrees, we can compute $|S_3(G) \cap \{v_1^{(1)}\}|$ and $|S_3(G) \cap \{v_1^{(2)}\}|$ exactly. Denote the set of all nodes that are connected to node v_1 as V_1, then $d_1 = |V_1|$ is the degree of v_1. There are $\binom{d_1}{2}$ subtrees of the pattern in Figure 2.6(a). For the patten in Figure 2.6(b), we can count the exact number of candidate nodes in depth 2 instead of approximating it by d^*. For any node v_j, $j \in V_1$, denote by $d_j^{(2)}$ the number of nodes that are connected to v_j, but not in $V_1 \cup \{v_1\}$. Then there are $\sum_{j \in V_1} d_j^{(2)}$ subtrees of the pattern in Figure 2.6(b). In total, the number of size-3 subtrees containing node v_1 should be $|S_3(G) \cap \{v_1\}| = \binom{d_1}{2} + \sum_{j \in V_1} d_j^{(2)}$. The first node should be sampled with

probability proportional to $|S_3(G) \cap \{v\}|$, $v \in V$.

After the first node v_1 has been sampled, we need to find the number of subgraphs that contain both v_1 and the next node v_2 we are going to choose. If we collapse v_1 and v_2 to one node $\tilde{v}$, then a 3-node subgraph containing both v_1 and v_2 is the same as a 2-node subgraph containing $\tilde{v}$. Therefore we can easily obtain $|S_3(G) \cap \{v_1, v_2\}| = |S_2(G) \cap \{\tilde{v}\}| = d_{1,2}$, where $d_{1,2} = \left|V_1 \cup V_2 \setminus \{v_1, v_2\}\right| = |V_{\tilde{v}}|$ and V_i is the set of nodes that are connected to v_i. Then the second node should be sampled with probability proportional to $|S_3(G) \cap \{v_1, v\}|$, $v \in V \setminus \{v_1\}$. For the last node v_3, since the number of size-3 subgraphs that contain v_1, v_2 and v_3 is just 1, the last node should be sampled uniformly from $V_1 \cup V_2 \setminus \{v_1, v_2\}$.

Using the exact number in the sampling procedure will guarantee that the probability of a size-3 subgraph sampled in the order (v_1, v_2, v_3) is $1/(3! \cdot |S_3(G))|$. This is uniform distribution if the order of the sample is considered. However, the probability of obtaining a subgraph $\{v_1, v_2, v_3\}$ without specifying the order, which involves a summation like (2.9), is not always $1/|S_3(G)|$ (i.e., not always the uniform distribution). The reason is because we may not be able to generate the subgraph in all 3! possible orders (see the discussion after (2.9)), and the number of possible orders may be different for different subgraphs.

2.4 Motif Detection

In order to detect network motifs, we need to do a significance test on the subgraph concentrations to determine whether a certain subgraph pattern appears significantly more frequently in the given network than in random networks with the same degree sequence. Milo et al. (2004) proposed the significance profile (SP) as a measure of statistical significance. SP is a normalized Z-score vector based on the number of subgraph patterns in the observed network and in random networks with the same degree sequence. Later, Kashtan et al. (2004) used normalized Z-score of subgraph frequencies to measure the statistical

significance. The Z-score is calculated based on a normal approximation,

$$Z = \frac{\mathcal{C}_k^i(G) - \bar{\mathcal{C}}_k^i(G_{random})}{\mathrm{sd}(\mathcal{C}_k^i(G_{random}))}, \tag{2.10}$$

where $\bar{\mathcal{C}}_k^i(G_{random})$ and $\mathrm{sd}(\mathcal{C}_k^i(G_{random}))$ are the mean and standard deviation of the randomized subgraph concentrations. How reliable this approximation is can be hard to evaluate in practice.

An alternative method is to compute the exact p-value, the probability of observing a subgraph concentration that is as large as or larger than what was seen in the observed network relative to random networks with the same degree sequence. A Metropolis-Hastings algorithm based on *switching* of edges may be used to generate random networks uniformly from the set of networks with a given degree sequence (Blitzstein and Diaconis, 2010). Denote $i-j$ as an edge between node i and node j. At each step of the Metropolis-Hastings algorithm, randomly choose two edges $x-y$ and $u-v$ from the current graph G with distinct nodes x, y, u, v. If there are no edges between the pair x and u and the pair y and v, then the Markov chain moves to a new state G' constructed by replacing the edges $x-y$ and $u-v$ in the current graph G by two new edges $x-u$ and $y-v$; otherwise, the Markov chain stays at the current graph G. We may take a sample G_i after every N_0 steps to reduce the dependence between the samples. Other ways for sampling networks with given degrees, such as sequential importance sampling, can be found in Blitzstein and Diaconis (2010) and Zhang and Chen (2013).

The exact p-value for a specific subgraph concentration $\mathcal{C}_k^i(G)$ can be estimated by

$$\frac{1}{N_1}\sum_{j=1}^{N_1} \mathbb{1}\{\mathcal{C}_k^i(G_j) \geq \mathcal{C}_k^i(G)\}, \tag{2.11}$$

where $G_1, \ldots, G_{N_1}$ are generated from the Metropolis-Hastings algorithm, and they are random networks with the same degree sequence as the given network G. Generally if $p < 0.01$

the subgraph pattern is considered to be a network motif. We will use the exact p-value to detect motifs in Section 2.5.

2.5 Applications

We apply Algorithm 1 on several real networks and demonstrate excellent performance. All data sets in the article are publicly available. The checking of isomorphic class is done using the *igraph* package of Csardi and Nepusz (2006). In motif detection, we throw away the first 1000 samples of the Metropolis-Hastings algorithm in Section 2.4 as the burn-in period, and keep every 100th sample afterwards to reduce the dependence between samples. Computation is performed on a Windows laptop with a 2.5 GHz processor and 8 GB RAM. The code for our algorithm is available in online supplementary materials.

2.5.1 *E. coli* network

We first examine an *E. coli* gene regulation network containing 423 nodes and 519 edges (Kashtan et al., 2004), see Figure 2.7. Each node represents an operon and a directed edge represents a source operon directly regulating a target operon (Shen-Orr et al., 2002). We compared three methods in estimating subgraph concentrations: the edge sampling method of Kashtan et al. (2004), the general node sampling algorithm proposed in Section 2.3.2, and the exact node sampling algorithm for size-3 subgraphs proposed in Section 2.3.3. For each algorithm, we generated $N = 500$ subgraphs to estimate subgraph concentration. Then the procedure is repeated 100 times to obtain 100 estimates. The mean of the 100 estimates and the standard error of the mean (in parentheses) are given in Tables 2.1, 2.2 and 2.3. The true values of the subgraph concentrations are reported in Kashtan et al. (2004) and they are shown in the tables as well.

Tables 2.1, 2.2 and 2.3 show the results for size-3, size-4 and size-5 subgraphs, respectively. All methods seem to give reasonable estimates for the subgraph concentrations. The node

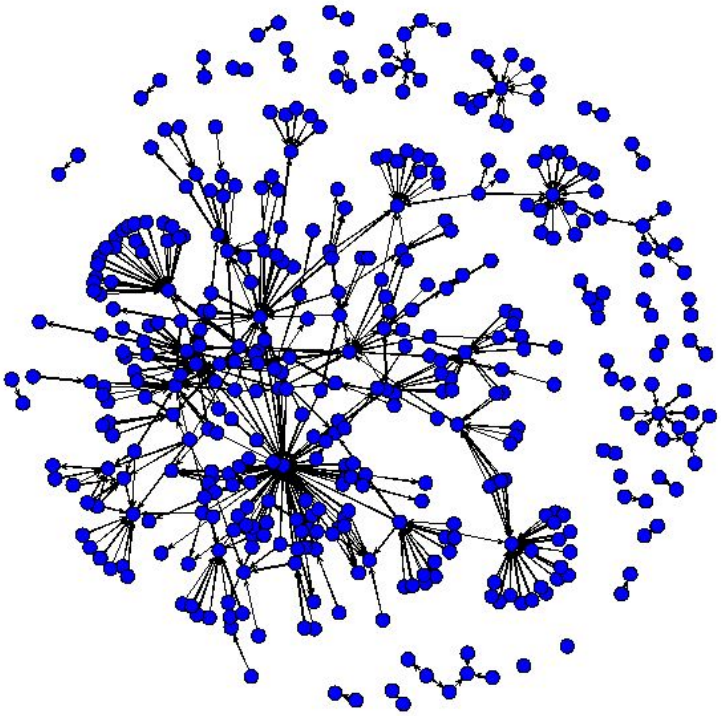


Figure 2.7: An *E. coli* network

sampling method gives smaller standard errors than the edge sampling method in most subgraph concentrations in the three tables. For size-3 subgraphs, the exact node sampling method gives the smallest standard errors for almost all subgraph concentrations in Table 2.1. A more direct way to compare edge sampling with node sampling is to look at the cv^2 of the importance weight, which measures the χ^2 distance between the proposal distribution and the target distribution (uniform in this case). Table 2.4 shows the cv^2 for subgraphs with different sizes. We can see that the cv^2 values for node sampling are very close to 0 for subgraphs with different sizes, and they are much smaller than the cv^2 for edge sampling. This indicates that the proposal distribution for node sampling, especially the exact node sampling of size-3 subgraphs, is very close to the uniform distribution. The cv^2 in general increases as the size of the subgraph increases.

Subgraph	True Concentration	Edge Sampling	Node Sampling (general)	Node Sampling (exact)
	91.760	91.713 (0.108)	91.624 (0.099)	91.827 (0.057)
	3.073	3.104 (0.057)	3.143 (0.067)	3.079 (0.056)
	4.364	4.385 (0.056)	4.432 (0.060)	4.297 (0.057)
	0.807*	0.798 (0.019)	0.801 (0.019)	0.797 (0.013)

Table 2.1: Concentrations of size-3 subgraphs in an *E. coli* network. The reported values are concentrations$\times 10^2$. Here * denotes subgraphs that are detected as motifs.

Subgraph	True Concentration	Edge Sampling	Node Sampling (general)
	833.824	819.0002 (5.672)	835.722 (1.287)
	6.138	6.046 (0.428)	6.319 (0.363)
	37.536	37.894 (0.835)	37.063 (0.768)
	97.112	98.507 (3.002)	96.231 (1.109)
	15.901	15.826 (1.072)	15.466 (1.737)
	2.491*	2.590 (0.235)	2.522 (0.120)
	0.608*	0.698 (0.104)	0.667 (0.097)

Table 2.2: Concentrations of size-4 subgraphs in an *E. coli* network. The reported values are concentrations$\times 10^3$. Here * denotes subgraphs that are detected as motifs. Only subgraphs with concentration greater than 2×10^{-3} and subgraphs that are detected as motifs are listed here.

We performed the significance test of frequency by comparing the subgraph frequencies of the observed network with 1000 randomized networks of the same degree sequence generated from the Metropolis-Hastings algorithm in Section 2.4. The significance level is chosen to be 0.01. Only one size-3 subgraph pattern (listed in Table 2.1) is detected as a network motif with the p-value for this pattern being 0 based on 1000 randomized networks. For size-4 subgraphs in Table 2.2, two subgraphs are found to be network motifs with the p-values for both patterns being 0. Four size-5 subgraphs are found to be significant and they are listed in Table 2.3.

Subgraph	True Concentration	Edge Sampling	Node Sampling (general)
	0.189	0.1640 (0.0501)	0.1728 (0.0308)
	0.014	0.0149 (0.0077)	0.0142 (0.0046)
	0.038	0.0359 (0.0028)	0.0421 (0.0125)
	0.013	0.0123 (0.0062)	0.0124 (0.0069)

Table 2.3: Concentrations of size-5 subgraphs in an *E. coli* network. The reported values are concentrations$\times 10^3$. Only subgraphs that are detected as motifs are listed here.

Subgraph Size	cv^2 for Edge Sampling	cv^2 for Node Sampling (general/exact)
3	1.1257	0.0216/0.003
4	3.0684	0.0540
5	5.2301	0.0568
6	5.0185	0.3342

Table 2.4: The cv^2 for different sampling methods on an *E. coli* network.

2.5.2 Electronic circuits network

The electronic circuits network we studied contains 512 nodes and 819 edges (Milo et al., 2004), see Figure 2.8. Nodes represent electronic components and edges represent current flows between components. Similar to the *E. coli* example, for each algorithm we generated $N = 1000$ subgraphs to estimate subgraph concentration, and then the procedure is repeated 100 times to obtain the mean of the 100 estimates and the standard error of the mean (in parentheses), as shown in Tables 2.5, 2.6 and 2.7. The true values of the subgraph concentrations are obtained through exhaustive enumerations in Wernicke (2006), and they are shown in the tables as well, except size-5 subgraphs for which the enumeration method is not feasible. We used the Metropolis-Hastings algorithm in Section 2.4 to detect network motifs.

Similar to what we observed in the *E. coli* example, Tables 2.5, 2.6 and 2.7 show that most of the standard errors for the node sampling method are smaller than that of the edge sampling method. This is especially so in Table 2.7 where node sampling always has a smaller standard error than edge sampling for those patterns with high frequencies. Table 2.8 shows that the cv^2 for node sampling for subgraphs with different sizes is much smaller

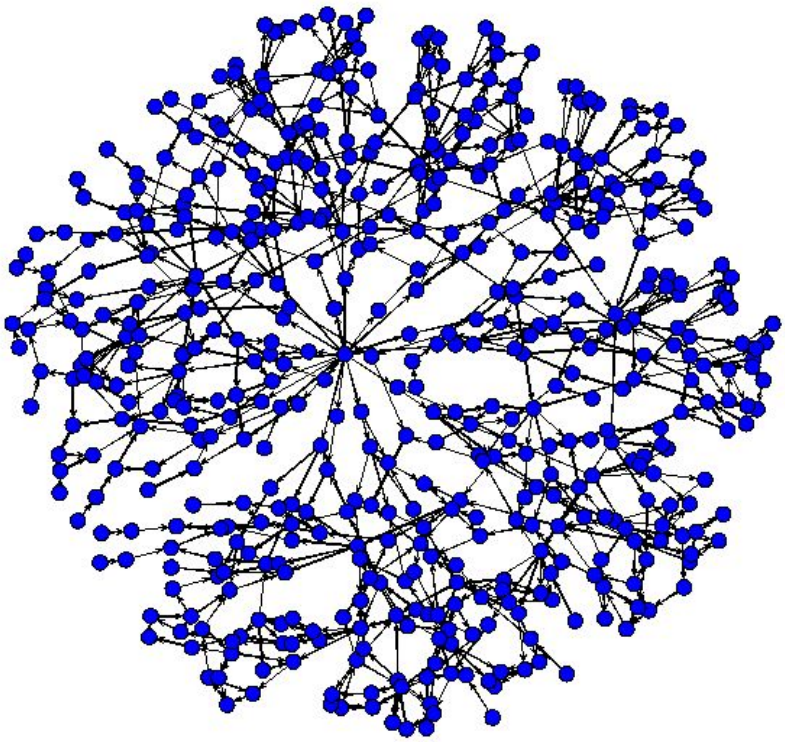


Figure 2.8: An electronic circuits network

than the cv^2 for edge sampling, indicating that the node sampling proposal is much closer to the target uniform distribution than the edge sampling proposal.

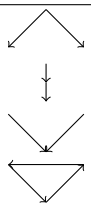
Subgraph	True Concentration	Edge Sampling	Node Sampling (general)	Node Sampling (exact)
	35.82	36.665 (0.257)	35.604 (0.212)	36.193 (0.189)
	45.81	44.932 (0.262)	45.802 (0.187)	45.511 (0.210)
	16.70	16.747 (0.168)	16.908 (0.135)	16.614 (0.156)
	1.67	1.656 (0.034)	1.686 (0.047)	1.682 (0.040)

Table 2.5: Concentrations of size-3 subgraphs in an electronic circuits network. The reported values are concentrations$\times 10^2$.

Subgraph	True Concentration	Edge Sampling	Node Sampling (general)
	1.505	1.617 (0.072)	1.636 (0.071)
	0.708	0.755 (0.039)	0.728 (0.049)
	0.226	0.245 (0.017)	0.255 (0.034)
	0.216	0.237 (0.028)	0.22 (0.026)
	0.079	0.092 (0.025)	0.067 (0.015)

Table 2.6: Concentrations of size-4 subgraphs in an electronic circuits network. The reported values are concentrations$\times 10^2$. Only subgraphs that are detected as motifs are listed here (significance level is 0.01).

Subgraph	Edge Sampling	Node Sampling (general)
	18.725 (1.031)	19.400 (0.458)
	12.299 (0.867)	14.838 (0.336)
	6.328 (0.570)	7.062 (0.327)
	4.227 (0.458)	4.365 (0.198)
	4.031 (0.323)	4.032 (0.242)
	3.608 (0.291)	3.642 (0.207)

Table 2.7: Concentrations of size-5 subgraphs in an electronic circuits network. The reported values are concentrations$\times 10^2$. Only subgraphs with frequency greater than 3% are listed here.

Subgraph Size	cv^2 for Edge Sampling	cv^2 for Node Sampling (general/exact)
3	0.3349	0.0400/0.0235
4	1.7302	0.3240
5	5.9873	0.5549
6	7.8054	0.8936

Table 2.8: The cv^2 for different sampling methods on an electronic circuits network.

2.5.3 *C. elegans* network

We examine a neuronal synaptic connection network of *C. elegans* with 280 nodes and 2170 edges (Achacoso and Yamamoto, 1992), see Figure 2.9. Each node represents a target muscle cell and each edge represents a synapse connection between a pair of neurons. This network is more dense than the last two examples. As in the last example, we generated $N = 1000$ subgraphs for each algorithm to estimate subgraph concentration, and then the

procedure is repeated 100 times to obtain the mean and its standard error (in parentheses), see Tables 2.9, 2.10 and 2.11. The true values of size-3 subgraph concentrations are reported in Kashtan et al. (2004) and shown in Table 2.9. The true values for size-4 and size-5 subgraph concentrations are not available. We also used the same Metropolis-Hastings algorithm to detect network motifs (significance level is 0.01).

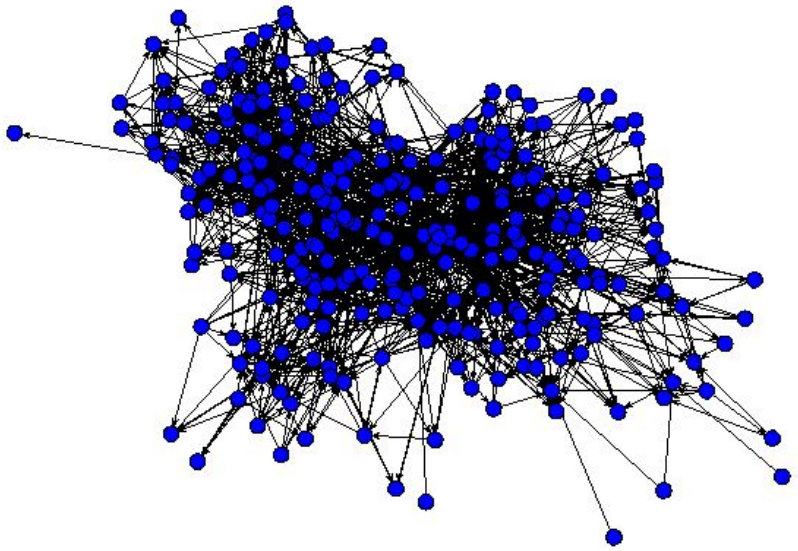


Figure 2.9: A *C.elegans* network

Table 2.9 lists size-3 subgraph patterns that are detected as motifs. We can see that node sampling tends to have a smaller standard error than edge sampling for the motif concentrations, and the exact node sampling has the smallest standard error among the three methods. Tables 2.10 and 2.11 list subgraph patterns with relatively high frequencies. In these two tables, node sampling gives smaller standard errors than edge sampling for all subgraph concentrations. The cv^2 in Table 2.12 further shows that node sampling is more efficient than edge sampling. Overall, the advantage of node sampling seems to be more obvious for this denser network.

Subgraph	True Concentration	Edge Sampling	Node Sampling (general)	Node Sampling (exact)
	33.678	33.354 (0.621)	33.512 (0.434)	33.730 (0.221)
	4.289	4.328 (0.168)	4.122 (0.163)	4.107 (0.097)
	1.370	1.186 (0.057)	1.565 (0.072)	1.400 (0.046)
	0.848	0.889 (0.051)	0.831 (0.051)	0.826 (0.039)
	0.487	0.551 (0.038)	0.445 (0.029)	0.527 (0.022)
	0.402	0.443 (0.031)	0.466 (0.053)	0.436 (0.022)
	0.043	0.042 (0.006)	0.039 (0.005)	0.039 (0.005)

Table 2.9: Concentrations of size-3 subgraphs in a *C. elegans* network. The reported values are concentrations$\times 10^2$. Only subgraphs that are detected as motifs are listed here.

Subgraph	Edge Sampling	Node Sampling (general)
	11.842 (0.547)	10.907 (0.313)
	10.438 (0.467)	10.853 (0.329)
	9.529 (0.410)	9.784 (0.321)
	9.038 (0.458)	8.942 (0.324)
	7.845 (0.344)	7.941 (0.314)
	6.497 (0.427)	6.692 (0.284)
	4.709 (0.297)	4.604 (0.204)
	2.957 (0.227)	3.210 (0.162)

Table 2.10: Concentrations of size-4 subgraphs in a *C. elegans* network. The reported values are concentrations$\times 10^2$. Only subgraphs with concentration greater than 3% are listed here.

Subgraph	Edge Sampling	Node Sampling (general)
	3.917 (0.344)	4.372 (0.256)
	3.773 (0.374)	3.834 (0.175)
	3.485 (0.298)	2.987 (0.184)
	2.778 (0.358)	2.935 (0.131)
	2.525 (0.366)	2.648 (0.158)

Table 2.11: Concentrations of size-5 subgraphs in a *C. elegans* network. The reported values are concentrations$\times 10^2$. Only subgraphs with concentration greater than 2% are listed here.

Subgraph Size	cv^2 for Edge Sampling	cv^2 for Node Sampling (general/exact)
3	0.4965	0.0346/0.0334
4	1.3404	0.0963
5	2.7872	0.2163
6	6.2931	0.4147

Table 2.12: The cv^2 for different sampling methods on a *C. elegans* network.

2.5.4 WWW data

The WWW dataset is a complex network of hyperlinks between webpages from the University of Notre Dame (Barabási and Albert, 1999). There are 325,729 nodes and 1,469,678 edges, where each node represents a webpage in the nd.edu domain, and each edge represents a link from a source page to a destination page. We generated $N = 50,000$ subgraphs for each algorithm to estimate subgraph concentration, and then the procedure is repeated 5 times to obtain the mean and its standard error (in parentheses), see Table 2.13. The true values of size-3 subgraph concentrations are reported in Kashtan et al. (2004) and also shown in Table 2.13. We used the same Metropolis-Hastings algorithm to detect network motifs (significance level is 0.01).

Table 2.13 lists size-3 subgraph patterns that are detected as motifs. Node sampling gives smaller standard errors in the majority of cases, especially for the motifs with relatively high frequencies. The cv^2 values in Table 2.14 show that node sampling is more effective than edge sampling. Although the cv^2 in general increases as the subgraph size increases for both methods, the cv^2 for node sampling is less sensitive to the subgraph size.

Subgraph	True Concentration	Edge Sampling	Node Sampling (general)
	1.74	1.615 (0.681)	1.644 (0.287)
	4.1	4.421 (1.358)	4.074 (1.169)
	1.08	1.064 (0.591)	1.093 (0.742)
	0.37	0.396 (0.138)	0.332 (0.231)
	23.6	23.805 (3.114)	24.13 (2.609)

Table 2.13: Concentrations of size-3 subgraphs in a WWW network. The reported values are concentrations$\times 10^3$. Only subgraphs that are detected as motifs are listed here.

Subgraph Size	cv^2 for Edge Sampling	cv^2 for Node Sampling (general)
3	11.4732	0.5863
4	33.6875	0.9712
5	47.4981	1.4492

Table 2.14: The cv^2 for different sampling methods on a WWW network.

2.5.5 Computing time and accuracy

We compare the running time of the edge sampling method (Kashtan et al., 2004) and our proposed node sampling method for the four real networks in the last sections with subgraph size varies from $k = 3$ to 6. Table 2.15 reports the time for generating 10,000 subgraphs for the first three networks and 100,000 subgraphs for the WWW network. For *E. coli* and electronic circuits networks, whose number of edges is close to the number of nodes, there is no dramatic difference between the running time of the two sampling methods. However, for *C. elegans* and the WWW networks with high edge density, the node sampling method is always faster than the edge sampling method. Also in general, node sampling becomes even more efficient than edge sampling as the subgraph size increases.

If we take into account the cv^2 values in earlier tables, then given the same amount of time, the effective sample size (ESS) of the node sampling method will be greater than the ESS of the edge sampling method. For example, node sampling for WWW network with $k = 5$ is about 3.4 times faster than edge sampling, and the ESS of node sampling is about 20 times larger than edge sampling based on cv^2 values in Table 2.14, so given the same amount of time, the ESS of node sampling is more than 60 times larger than the ESS of edge sampling. This shows the efficiency of our method, especially for sampling subgraphs of dense networks. The computing time for checking isomorphic class is not included in the running time in Table 2.15. The computational cost in this checking step can be reduced by applying recent computing techniques, such as parallel computing in Lin et al. (2015).

We also compare our node sampling method with the exact enumeration algorithm *FAN-*

Subgraph Size	*E. coli*		Electronic		*C. elegans*		WWW	
	Edge	Node	Edge	Node	Edge	Node	Edge	Node
3	26.97 s	27.91 s	41.76 s	44.91 s	60.28 s	27.23 s	2.1 h	56 m
4	27.35 s	28.71 s	62.27 s	65.52 s	137.64 s	40.61 s	6.9 h	2.1 h
5	29.67 s	29.98 s	86.50 s	78.55 s	434.92 s	76.45 s	12 h	3.5 h
6	32.14 s	32.72 s	151.24 s	129.41 s	663.41 s	89.61 s	-	-

Table 2.15: Running time for edge sampling and node sampling methods for sampling subgraphs and computing importance weights.

MOD implemented in C++ by Wernicke (2006). Table 2.16 reports the running time for the node sampling method to estimate subgraph concentrations and for *FANMOD* to complete exhaustive enumeration. Although the enumeration method seems to run fast on relatively small networks, it takes much longer time than node sampling for large networks with a lot of nodes and edges, such as the WWW network with subgraph size $k = 4$ and 5. It is worth mentioning that the checking of isomorphic class in node sampling is implemented in R, which is slower than C++, so the running time in Table 2.16 is not a fair comparison.

Subgraph Size	*E. coli*		Electronic		*C. elegans*		WWW	
	FANMOD	Node	*FANMOD*	Node	*FANMOD*	Node	*FANMOD*	Node
3	2 s	42 s	3 s	62s	3 s	52 s	17 m	2.1 h
4	2 s	57 s	5 s	81s	5 s	85 s	17.9 h	4.6 h
5	5 s	67 s	17 s	121 s	28 s	117 s	68 h	7.5 h
6	72 s	81 s	72 s	149 s	81 s	132 s	-	-

Table 2.16: Running time for the exact enumeration algorithm *FANMOD* and the node sampling method for estimating subgraph concentrations.

To assess the accuracy of our estimate of subgraph concentration based on node sampling, we compute the adjusted root mean squared error (RMSE) (Chen et al., 2016), defined as $\sqrt{\text{MSE}(\mathcal{C}_k^i)}/\mathcal{C}_k^i$. Figure 2.10 reports the RMSE of subgraph concentrations for *E. coli*, electronic circuits, and *C. elegans* networks with subgraph size $k = 4$, 5 and 6. The WWW network is not included because there are too many subgraphs to list in the figure. Figure 2.10 shows that our node sampling method estimates subgraph concentrations well with relatively small adjusted RMSEs. All adjusted RMSEs are less than 0.6, and most of them are less than 0.2 for electronic circuits and *C. elegans* networks.

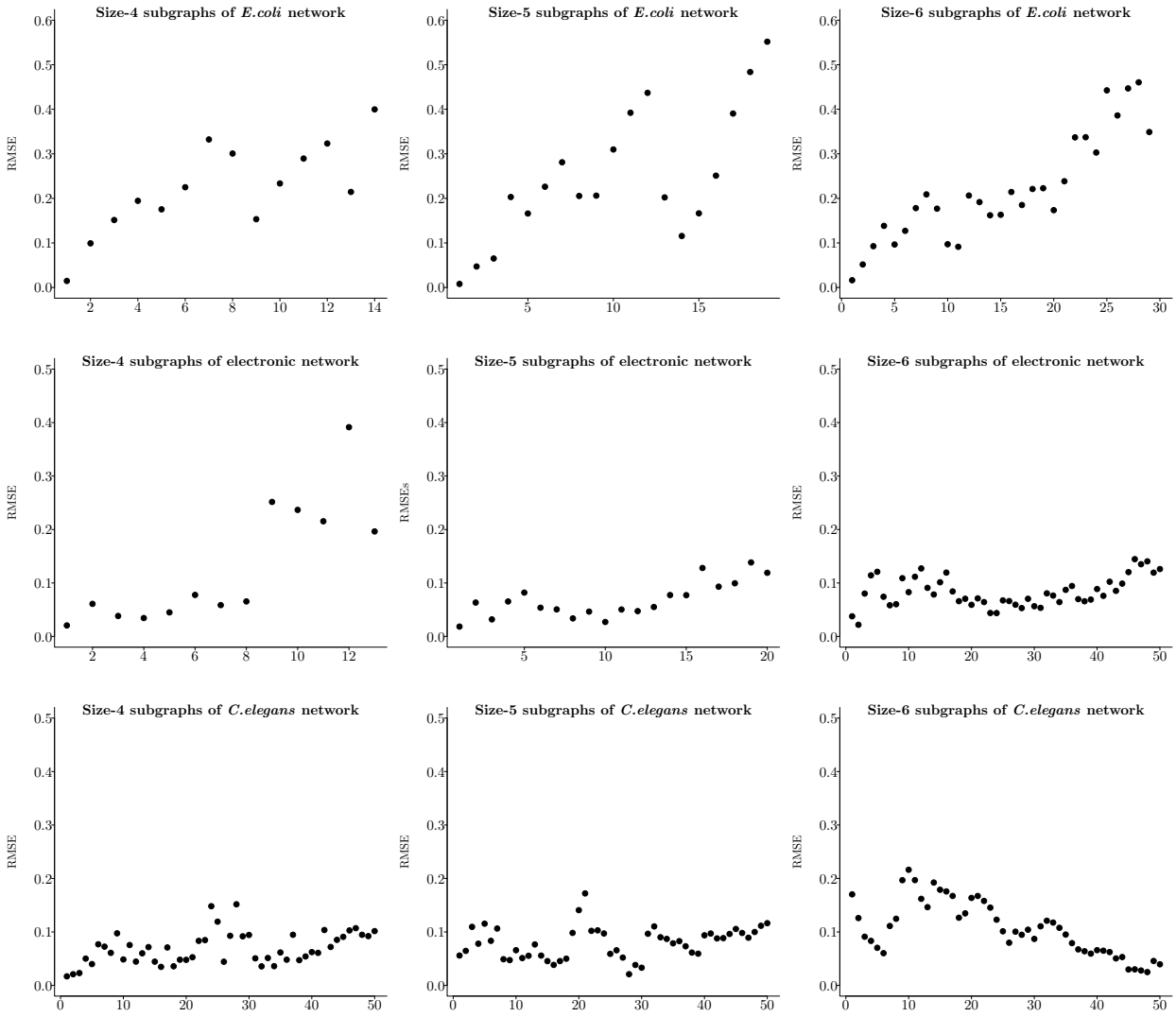


Figure 2.10: The adjusted RMSEs of subgraph concentrations for *E. coli*, electronic circuits, and *C. elegans* networks with subgraph size $k = 4$, 5 and 6. Subgraphs are listed along the X-axis in decreasing order with respect to the concentrations of the subgraph.

2.6 Discussion

We proposed a sequential importance sampling method to generate subgraphs from a given network and estimate the concentration of subgraph patterns. The algorithm samples subgraphs node by node, with the proposal distribution approximately proportional to the number of subgraphs containing a relevant set of nodes at each step. The proposal distribution is closer to the target uniform distribution than competing methods based on sampling

edges. For dense networks, node sampling outperforms edge sampling in terms of computation time as well. For the *C. elegans* network, the proposed node sampling method detected two additional motifs that were missed by the edge sampling method.

Accounting for the multiple ways to sample the same subgraph is an issue for both node and edge sampling methods. Both methods require calculating the probabilities of all possible order permutations of a sampled subgraph. For edge sampling method, the true probability of the sampled subgraph is the sum of the probabilities over all possible edge permutations. This number of permutations can be large as the subgraph becomes dense. The number of possible permutations of the edge sequence of a size-k subgraph is bounded above by $(k-1)!\binom{\binom{k}{2}}{k-1}$. For node sampling method, the number of all possible node permutations for a size-k subgraph is bounded above by $k!$. Therefore, the sampling weight is easier to compute for node sampling compared to edge sampling when the network is dense or the subgraph size is large.

One way to avoid the permutation of nodes or edges is to treat different orderings of the nodes or edges of a subgraph as different subgraphs. This results in a larger sample space, but it will be sufficient to count the number of permutations to obtain the concentrations. A similar idea has been used in Blitzstein and Diaconis (2010) to sample networks with given degrees. This approach saves time, however the cv^2 may become larger due to the larger sampling space. A new proposal distribution might be needed to ensure that the final proposal is close to the target distribution on this larger sample space.

Chapter 3

Q Matrix Estimation

3.1 Introduction

Cognitive diagnosis models (CDMs) were developed to provide educators and researchers with instructionally relevant assessments (Huff and Goodman, 2007; Leighton and Gierl, 2007). CDMs use cognitive theory within a psychometric framework to, "...explain achievement test performance by providing insight into whether it is students' understanding (or lack of it) or something else that is the primary cause of their performance" (Norris et al., 2007, p. 61). That is, CDMs are partially ordered latent class models that assist researchers in classifying students as either masters or non-masters on a collection of skills deemed important for success on educational tasks. A benefit of the CDM framework is that CDMs provide educators more detailed diagnostic information regarding student skills/attributes than is available with more broadly defined continuous traits in item response models.

The application of CDMs is dependent upon the availability of cognitive theory that specifies the skills and/or attributes necessary for success on a collection of tasks. In particular, let $j = 1, \ldots, J$ and $k = 1, \ldots, K$ index items and skills, respectively. Content expert knowledge is codified in a $J \times K$ $\mathbf{Q}$ matrix with individual elements denoted by $q_{jk} = 1$ if skill k is needed to respond to item j as a master and zero otherwise. Given a specified $\mathbf{Q}$, the CDM item and latent class parameters can be accurately estimated. In fact, CDMs have received broader application among educational and psychological researchers in large part due to the availability of computer code and software (de la Torre, 2009; Chiu and Köhn, 2016; Rupp, 2009; Rupp et al., 2010; Templin and Hoffman, 2013).

However, the application of CDMs is not without limitations and model parameters may be inaccurate in cases where researchers employ a misspecified $\mathbf{Q}$. For instance, Henson and Templin (2007) and Rupp and Templin (2008) provided Monte Carlo evidence that employing a misspecified $\mathbf{Q}$ results in inaccurate skill classification and biased item parameters.

Clearly, a fundamental problem for CDM research is the estimation of $\mathbf{Q}$ and prior research considered several general strategies for estimating $\mathbf{Q}$. First, several studies assumed partial knowledge of $\mathbf{Q}$ and developed procedures and algorithms for refining (Chiu, 2013; de la Torre, 2008; de la Torre and Chiu, 2016) and estimating (DeCarlo, 2012; Templin and Henson, 2006*a*) elements of $\mathbf{Q}$. For example, de la Torre and Chiu (2016) offer a validation procedure that first estimates item parameters from a provisional $\mathbf{Q}$ matrix and then updates elements of $\mathbf{Q}$ using a search algorithm. One limitation of the aforementioned strategy is that the partial knowledge of $\mathbf{Q}$ may be inaccurate, which could compromise the consistency of $\mathbf{Q}$ matrix refinement (Liu, 2015). Second, recent efforts considered Bayesian (Chung, 2014) and frequentist (Chen et al., 2015; Liu et al., 2012, 2013; Xiang, 2013) estimators of $\mathbf{Q}$. For instance, Chen et al. (2015) derived the necessary and sufficient identifiability conditions for the DINA and DINO models and applied regularization techniques to estimate $\mathbf{Q}$ within the framework of statistical model selection.

In this chapter we develop a Bayesian approach for estimating $\mathbf{Q}$ for the DINA model. The proposed Bayesian formulation in this study improves upon existing strategies for estimating $\mathbf{Q}$. First, the Bayesian estimator does not require partial knowledge of $\mathbf{Q}$ and is accordingly an exploratory technique similar to Chen et al. (2015). Second, the Bayesian approach discussed here offers an efficient Markov Chain Monte Carlo algorithm for estimating an identified $\mathbf{Q}$ for the DINA model. Specifically, the method of Chen et al. (2015) does not explicitly enforce the identifiability conditions. Chen et al.'s (2015) method performed well in simulation studies, however, the regularization techniques did not always produce identified $\mathbf{Q}$ matrices in empirical applications. This study proposes a Metropolis-Hastings (MH) within Gibbs algorithm to sample elements of $\mathbf{Q}$ from the space of identified $\mathbf{Q}$ matrices.

The remainder of this chapter includes four sections. The first section describes a Bayesian formulation for the DINA model and the MH sampler for $\mathbf{Q}$. Note that the proposed model can be directly extended to the DINO model, as well, given the duality of the DINA and DINO models (Xu and Zhang, 2016). The second section reports results from a Monte Carlo simulation study that examines the recoverability of $\mathbf{Q}$ for different sample sizes and population attribute distributions. The third section includes an application of the developed methodology to estimate $\mathbf{Q}$ for the classic Tatsuoka fraction-subtraction dataset (Tatsuoka, 1984, 2002). The last section provides concluding remarks and implications for future research.

3.2 Bayesian Model Formulation

3.2.1 Model Formulation

As introduced in the previous section, there are various CDMs to explain achievement test performance. This chapter focuses on the DINA model. Suppose there are N subjects, each of whom provides a binary response to J items. The model intuitively assumes that a subject's responses are stochastically related to the unobserved latent attributes, $\boldsymbol{\alpha} \in \{0,1\}^K$. The attribute profile for individual i is denoted as $\boldsymbol{\alpha}_i = (\alpha_{i1}, \cdots, \alpha_{iK})'$ where $\alpha_{ik} = 1$ means subject i has attribute k. We write the set of all possible configuration of attribute profile as $\mathcal{C}$ with cardinality $|\mathcal{C}| = 2^K$. Let Y_{ij} equal one for a correct response and zero otherwise. Furthermore, define $\boldsymbol{Y} = (\boldsymbol{Y}_1, \cdots, \boldsymbol{Y}_N)'$ as an $N \times J$ matrix of dichotomous random variables where $\boldsymbol{Y}_i = (Y_{i1}, \ldots, Y_{iJ})'$ is the J-dimensional vector of binary responses for individual i. Throughout the chapter Y refers to a random variable whereas as y denotes a realized value.

As noted above, the $\mathbf{Q}$ matrix catalogues the attributes needed to master each item. Note that $\mathbf{Q} = (\boldsymbol{q}_1, \cdots, \boldsymbol{q}_J)' = (\boldsymbol{Q}_1, \cdots, \boldsymbol{Q}_K)$ where $\boldsymbol{q}_j'$ is the j-th row of $\mathbf{Q}$ and the italicized $\boldsymbol{Q}_k$ denotes the k-th column. The DINA model specifies a conjunctive relationship between

latent attributes $\boldsymbol{\alpha}_i$ and observed responses Y_{ij} in the sense that success on an item occurs with highest probability for those students that possess all of the necessary attributes. The conjunctive nature of the DINA model is apparent with the ideal response for individual i on item j,

$$\eta_{ij} = \mathcal{I}\left(\alpha_{ik} \geq q_{jk} \text{ for all } k\right) = \mathcal{I}\left(\boldsymbol{\alpha}_i'\boldsymbol{q}_j = \boldsymbol{q}_j'\boldsymbol{q}_j\right), \tag{3.1}$$

where $\mathcal{I}(\cdot)$ denotes the indicator function. η_{ij} equals one if individual i possesses all of the necessary attributes and zero if at least one is missing. The item response function (IRF) for the DINA model is based upon the relationship between η_{ij} and Y_{ij} through the introduction of guessing and slipping parameters. That is, guessing is the probability of correctly answering item j when at least one attribute is lacking, $g_j = P\left(Y_{ij} = 1|\eta_{ij} = 0\right)$, and slipping is the probability of an incorrect response for individuals with all of the required attributes, $s_j = P\left(Y_{ij} = 0|\eta_{ij} = 1\right)$. A necessary constraint to ensure monotonicity of the IRF is $0 \leq g_j < 1 - s_j \leq 1$ (Junker and Sijtsma, 2001). Therefore, the probability that subject i correctly answers item j is,

$$P\left(Y_{ij} = 1|\boldsymbol{\alpha}_i, s_j, g_j, \boldsymbol{q}_j\right) = (1 - s_j)^{\eta_{ij}} g_j^{1-\eta_{ij}}. \tag{3.2}$$

The likelihood of observing $\boldsymbol{Y}$ under the assumption of conditional independence on the $N \times K$ matrix of attributes $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_N)'$ is,

$$p\left(\boldsymbol{Y}|\boldsymbol{s}, \boldsymbol{g}, \boldsymbol{\pi}, \mathbf{Q}\right) = \prod_{i=1}^{N} \sum_{\boldsymbol{\alpha}_c \in \{0,1\}^K} \pi_c \prod_{j=1}^{J} \left[(1 - s_j)^{\eta_{cj}} g_j^{1-\eta_{cj}}\right]^{y_{ij}} \left[s_j^{\eta_{cj}} (1 - g_j)^{1-\eta_{cj}}\right]^{1-y_{ij}}, \tag{3.3}$$

where $\eta_{cj} = \mathcal{I}\left(\boldsymbol{\alpha}_c'\boldsymbol{q}_j = \boldsymbol{q}_j'\boldsymbol{q}_j\right)$, $\boldsymbol{s} = (s_1, \ldots, s_J)'$, and $\boldsymbol{g} = (g_1, \ldots, g_J)'$. Furthermore, $\pi_c = P\left(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_c\right)$ is the marginal probability of the latent attribute configuration $\boldsymbol{\alpha}_c$, and the 2^K dimensional vector of latent class probabilities can be denoted by $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{2^K})'$.

The definitions for the DINA CDM imply the following Bayesian formulation,

$$y_{ij} \left| \boldsymbol{\alpha}_i, s_j, g_j, \boldsymbol{q}_j \right. \sim \text{ Bernoulli} \left(\left(1 - s_j\right)^{\eta_{ij}} g_j^{1-\eta_{ij}} \right) \tag{3.4}$$

$$\eta_{ij} = \mathcal{I}\left(\boldsymbol{\alpha}_i' \boldsymbol{q}_j = \boldsymbol{q}_j' \boldsymbol{q}_j \right)$$

$$p\left(\boldsymbol{\alpha}_i | \boldsymbol{\pi}\right) \propto \prod_{c=1}^{2^K} \pi_c^{\mathcal{I}(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_c)} \tag{3.5}$$

$$0 < \pi_c < 1, \ \sum_{c=1}^{C} \pi_c = 1$$

$$\boldsymbol{\pi} \sim \text{Dirichlet}\left(\boldsymbol{\delta}_0\right) \tag{3.6}$$

$$\boldsymbol{\delta}_0 = (\delta_{01}, \ldots, \delta_{0C})$$

$$p\left(s_j, g_j\right) \propto s_j^{\alpha_s - 1} \left(1 - s_j\right)^{\beta_s - 1} g_j^{\alpha_g - 1} \left(1 - g_j\right)^{\beta_g - 1} \mathcal{I}\left(0 \leq g_j < 1 - s_j \leq 1\right) \tag{3.7}$$

$$p\left(\mathbf{Q}\right) \propto \mathcal{I}\left(\mathbf{Q} \in \mathcal{Q}\right) \tag{3.8}$$

Equations (3.4) through (3.7) were discussed in prior research (Culpepper, 2015). In particular, (3.4) is the DINA model IRF, (3.5) is a categorical prior for $\boldsymbol{\alpha}_i$ conditioned upon $\boldsymbol{\pi}$, (3.6) is a conjugate Dirichlet prior for latent class probabilities $\boldsymbol{\pi}$, and (3.7) is a truncated Beta prior for item parameters that enforces the monotonicity restriction.

The innovation of this study is the specification of a prior for $\mathbf{Q}$, so that MCMC can be used to sample $\mathbf{Q}$ from the posterior distribution, which is defined as,

$$P(\mathbf{Q}|\boldsymbol{Y}, \boldsymbol{\alpha}, \boldsymbol{s}, \boldsymbol{g}) \propto P(\boldsymbol{Y}|\boldsymbol{\alpha}, \boldsymbol{s}, \boldsymbol{g}, \mathbf{Q}) \mathcal{I}(\mathbf{Q} \in \mathcal{Q}), \tag{3.9}$$

where

$$P(\boldsymbol{Y}|\boldsymbol{\alpha}, \boldsymbol{s}, \boldsymbol{g}, \mathbf{Q}) = \prod_{i=1}^{N} \prod_{j=1}^{J} \left(\left(1 - s_j\right)^{\eta_{ij}} g_j^{1-\eta_{ij}} \right)^{y_{ij}} \left(s_j^{\eta_{ij}} \left(1 - g_j\right)^{1-\eta_{ij}} \right)^{1-y_{ij}}. \tag{3.10}$$

Equation (3.8) specifies a uniform prior for $\mathbf{Q}$ in the space $\mathcal{Q}$ of identified models.

Chen et al. (2015) proved that $\mathbf{Q} \in \mathcal{Q}$ for the DINA model and is identified if the

following additional conditions are met:

1. For a $J \times J$ permutation matrix $\mathbf{P}$, $\mathbf{Q}$ can be expressed as,

$$\mathbf{PQ} = \begin{bmatrix} \mathbf{I}_K \\ \mathbf{I}_K \\ \widetilde{\mathbf{Q}} \end{bmatrix}, \tag{3.11}$$

   where $\mathbf{I}_K$ is a $K \times K$ identity matrix and $\widetilde{\mathbf{Q}}$ is a $(J - 2K) \times K$ sub-matrix of $\mathbf{Q}$ with column k denoted by $\widetilde{\boldsymbol{Q}}_k$.

2. Each skill loads onto at least three items, which implies $\boldsymbol{Q}_k'\mathbf{1}_J \geq 3$ where $\mathbf{1}_J$ is a J dimensional vector of ones. Similarly, if c_k is the kth column margin for $\widetilde{\mathbf{Q}}$ (i.e., $c_k = \widetilde{\boldsymbol{Q}}_k'\mathbf{1}_{J-2K}$) this condition is equivalent to $c_k > 0$.

3. Each item loads onto at least one skill such that $\boldsymbol{q}_j'\mathbf{1}_K > 0$.

3.2.2 Identifiability and Consistency

Our exploratory procedures for estimating the DINA $\boldsymbol{Q}$ enforce Chen et al.'s (2015) identifiability conditions. Accordingly, as noted by an anonymous reviewer, our procedures assume that the true $\boldsymbol{Q}$ satisfies the identifiability conditions. It is possible that the true $\boldsymbol{Q}$ for a collection of items does not satisfy Chen et al.'s (2015) identifiability conditions and it might be desirable to use an exploratory method, such as Chung's (2014) unconstrained approach, that freely estimates $\boldsymbol{Q}$ to determine whether $\boldsymbol{Q}$ in fact satisfies the identifiability conditions. However, Chen et al. (2015) showed that a consistent estimator exists for $\boldsymbol{Q}$ only if the aforementioned identifiability conditions are satisfied. Consequently, our Bayesian estimator is consistent if the true $\boldsymbol{Q}$ satisfies the identifiability conditions. In contrast, there is no known consistent estimator if the true $\boldsymbol{Q}$ is outside the space of identified $\boldsymbol{Q}$ matrices.

Restricting $\boldsymbol{Q}$ to the identified space is also important for estimating other model pa-

rameters, such as guessing and slipping probabilities. If the model is identified the item parameters can be estimated by computing averages in the posterior because there is a single mode (up to a column permutation of $\boldsymbol{Q}$) with a one-to-one mapping between values of the item parameters and the corresponding value of the likelihood function. In contrast, if the model parameters are not identified it is possible that two different sets of values for the item parameters correspond to the same likelihood (i.e., different modes for different values of the item parameters). In this case there is no longer a clear strategy for estimating the item parameters because a posterior average would be computed across multiple modes.

3.2.3 Sampling Candidates of $\boldsymbol{Q} \in \mathcal{Q}$

$\boldsymbol{Q}$ can be sampled in a manner that satisfies the aforementioned identifiability restrictions. That is, the viable candidates for $\boldsymbol{Q}$, $\boldsymbol{Q}_*$, are in the set:

$$\boldsymbol{Q}_* \in \mathcal{Q} = \left\{ \boldsymbol{Q} : \boldsymbol{Q}_k' \mathbf{1}_J \geq 3 \; \forall \; k, \; \boldsymbol{q}_j' \mathbf{1}_K > 0 \; \forall \; j, \; (\mathbf{P}\boldsymbol{Q})' = \left[\mathbf{I}_K, \mathbf{I}_K, \widetilde{\boldsymbol{Q}}' \right] \right\}. \tag{3.12}$$

This section describes three approaches for drawing candidates $\boldsymbol{Q}_*$ for a MH sampler.

Independence Sampler

The first strategy is to construct independent samples of $\boldsymbol{Q}$ by generating a random $\mathbf{PA}$ matrix and then randomly permuting it with $\mathbf{P}'$ such that $\boldsymbol{Q}_* = \mathbf{P}'(\mathbf{PA})$. Specifically, consider the following steps to draw independent candidates $\boldsymbol{Q}_*$:

1. Fix the first $2K$ rows of $\mathbf{PA}$ to be $[\mathbf{I}_K, \mathbf{I}_K]'$.
2. Draw a sample of $\widetilde{\boldsymbol{Q}}$ that satisfies $\widetilde{\boldsymbol{Q}}_k' \mathbf{1}_{J-2K} > 0$ and $\boldsymbol{q}_j' \mathbf{1}_K > 0$:
   (a) For column k where $k = 1, \cdots, K-1$, sample c_k uniformly in $[1, J-2K]$.
   (b) Then sample $\widetilde{\boldsymbol{Q}}_k$ uniformly from one of the $\binom{J-2K}{c_k}$ configurations.

(c) For the last column K, count the number of rows that have all zero elements excluding the last column, denoted as l, fix the elements of the l rows in the last column to be 1.

(d) Sample c_K uniformly in $\big[\max(1, l), J - 2K\big]$.

(e) Sample $\widetilde{\boldsymbol{Q}}_K$ uniformly from one of the $\binom{J-2K-l}{c_K-l}$ configurations with the l elements fixed in 2(c).

3. Draw a $J \times J$ permutation matrix $\mathbf{P} = \big(\mathbf{e}_{i_1}, \ldots, \mathbf{e}_{i_J}\big)$ where $(i_1, \ldots, i_J)$ is a random permutation of $(1, \cdots, J)$ and $\mathbf{e}_i$ is the vector of

$$\mathbf{e}_i(l) = \begin{cases} 0 & l \neq i \\ 1 & l = i \end{cases} . \tag{3.13}$$

4. Set the candidate as $\boldsymbol{Q}_* = \mathbf{P}'(\mathbf{PA})$.

Dependence Sampler #1 (DS1)

The independence sampler described above ensures that each $\boldsymbol{Q}_*$ will be independent of the previous draw. A disadvantage of using an independence sampler is that it has a relatively low acceptance rate and likely requires more iterations to explore the posterior distribution in the MH steps. To introduce some dependence, we propose a random walk movement from the current state:

1. Initialize with any identified $\boldsymbol{Q}^{(0)} \in \mathcal{Q}$.

2. Update candidate $\boldsymbol{Q}^{(t)}$ at iteration t under identifiability constraints:

(a) Randomly select a column, say the k-th column.

(b) Let k_1 be the number of rows that have a single 1 at column k. Fix the 1's of those k_1 elements in the k-th column.

(c) For every $i \neq k, i = 1, \ldots, K$, let l_i denote the number of rows that equal to 1 at the i-th column and equal to 0 at every column other than column i and column k. Because of the identifiability constraints, $l_i \geq 2$. Randomly select 2 elements from those l_i rows in the k-th column and fix them as 0.

(d) There are $J - k_1 - 2(K-1)$ remaining elements in $\boldsymbol{Q}_k$ that are not fixed in 2(b) or 2(c) and the column sum c_k should be at least 3. Sample $\boldsymbol{Q}_{*k}$ uniformly from one of the $\sum_{c=\max\{3,k_1\}}^{J-2(K-1)} \binom{J-k_1-2(K-1)}{c-k_1}$ configurations for the remaining elements in $\boldsymbol{Q}_{k,B}$.

Dependence Sampler #2 (DS2)

Preliminary Monte Carlo studies suggested that the aforementioned dependence sampler accepts candidates with a low probability. In practice, we can adapt the range of each movement to adjust the acceptance rate, that is to say, we can update a subset of size B in a selected column:

1. Initialize with any identified $\boldsymbol{Q}^{(0)} \in \mathcal{Q}$.

2. Update candidate $\boldsymbol{Q}^{(t)}$ under identifiability constraints:

   (a) Randomly select one column and a subset of size B within it, say $\boldsymbol{Q}_{k,B}$ in the k-th column.

   (b) Let k_1 be the number of rows within $\boldsymbol{Q}_{k,B}$ that have a 1 at column k and zeros in the other columns. Fix the k_1 elements of those rows as 1 in $\boldsymbol{Q}_{k,B}$.

   (c) For every $i \neq k, i = 1, \ldots, K$, let l_i be the number of rows outside $\boldsymbol{Q}_{k,B}$ that have a single 1 at the i-th column; let b_i be the number of rows in $\boldsymbol{Q}_{k,B}$ that equal to 1 at the i-th column and equal to 0 for all columns other than columns i and k. If $l_i < 2$, randomly select $2 - l_i$ elements of the b_i rows in $\boldsymbol{Q}_{k,B}$ and fix as 0.

(d) Let $0 \leq k_0 \leq \min\{B - k_1, 2K - 2\}$ be the number of elements in $\boldsymbol{Q}_{k,B}$ that are fixed as 0 in 2(c).

(e) Let m be the number of elements equal to 1 in the k-th column outside $\boldsymbol{Q}_{k,B}$.

(f) There are $B - k_0 - k_1$ remaining elements in $\boldsymbol{Q}_{k,B}$ that are not fixed in 2(b) or 2(c) and the column sum c should be at least 3. Sample $\boldsymbol{Q}_{*k,B}$ uniformly from one of the $\sum_{c=\max\{3,m+k_1\}}^{B-k_0+m} \binom{B-k_0-k_1}{c-m-k_1}$ configurations for the remaining elements in $\boldsymbol{Q}_{k,B}$.

Example. We provide a step-by-step illustration of DS2 for steps 2(a) to 2(g) with a simple example using the following matrix $\boldsymbol{Q} \in \mathcal{Q}$:

$$\boldsymbol{Q} = \begin{pmatrix} 1 & \mathbf{0}(\mathrm{I}) & 0 \\ 0 & \mathbf{1}(\mathrm{II}) & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & \mathbf{0}(\mathrm{III}) & 1 \\ 0 & \mathbf{1}(\mathrm{IV}) & 0 \\ 0 & \mathbf{0}(\mathrm{V}) & 1 \\ 1 & 1 & 0 \end{pmatrix}, \quad \boldsymbol{Q}_* = \begin{pmatrix} 1 & \mathbf{0} & 0 \\ 0 & \mathbf{0} & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & \mathbf{1} & 1 \\ 0 & \mathbf{1} & 0 \\ 0 & \mathbf{0} & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

Consider the following steps to perform the DS2:

1. Suppose $\boldsymbol{Q}_{2,5}$ in the 2nd column is randomly selected to be updated. Note that entries of $\boldsymbol{Q}_{2,5}$ are indexed by Roman numerals.

2. Entry (IV) in $\boldsymbol{Q}_{2,5}$ must be fixed to 1 according to 2(b), which implies that $k_1 = 1$.

3. Examining rows of $\boldsymbol{Q}$ outside of $\boldsymbol{Q}_{2,5}$ shows that the third and fifth rows have a single one in the first and third columns, respectively, so $l_1 = 1$ and $l_3 = 1$. The first row of $\boldsymbol{Q}$ is the only row within $\boldsymbol{Q}_{2,5}$ where there is a 1 in the first column and a zero in the

third, which implies $b_1 = 1$. Similarly, $b_3 = 2$ given that the second and eighth rows of $\boldsymbol{Q}$ in $\boldsymbol{Q}_{2,5}$ have a 1 in column three and zero in column one. $l_1 = 1 < 2$ and $b_1 = 1$ implies that entry (I) is fixed to 0. $l_3 = 1$ and $b_3 = 2$, so one of entry (II) or (V) is randomly set to 0. For purposes of illustration suppose entry (V) is randomly set to 0.

4. In the previous step, entry (I) and (V) were fixed to 0, so $k_0 = 2$.

5. $m = 2$ for the ones in the fourth and last position of column two.

6. The column sum c needs to be at least 3. Based on the fact that $m + k_1 = 3$, the possible configurations for entries (II) and (III) are $(0,0), (0,1), (1,0)$ and $(1,1)$. In this case we sample the configuration with entries (II) and (III) equal to 0 and 1, respectively.

3.2.4 Irreducibility and Symmetry of Proposed Transition Function

A necessary condition for MH sampling is that the Markov chain is irreducible, in that any $\boldsymbol{Q} \in \mathcal{Q}$ can be reached from any other state. It is trivial to show that the independence sampler can propose every possible $\boldsymbol{Q} \in \mathcal{Q}$. For dependence samplers, we prove the irreducibility of the proposal that updates only one element at a time. This would imply irreducibility of both DS1 and DS2, as shown in the remarks after the proof.

Theorem 1. *When $J - 2K \geq 2$, the dependence sampler #2 with $B = 1$ can reach every possible candidate $\boldsymbol{Q}_* \in \mathcal{Q}$ in finite steps, regardless of the current state $\boldsymbol{Q}^{(t)}$ at time t.*

Proof. It is obvious the dependence sampler #2 ensures the candidate $\boldsymbol{Q}$ satisfies the identifiablility constraints. When $B = 1$, DS2 is equivalent to updating a randomly chosen element in $\boldsymbol{Q}^{(t)}$. If the selected element is 1 from the row vector $\boldsymbol{e}_k$, it remains unchanged in step 2(b); if it is 0 from the row vector $\boldsymbol{e}_k$, it remains unchanged in step 2(c) if there are only 2 $\boldsymbol{e}_k$ in $\boldsymbol{Q}$. Alternatively, the selected element can be updated to 1 later in 2(f) if there

are more than 2 $\boldsymbol{e}_k$ in $\boldsymbol{Q}$. If the selected element is 1 in a column with column sum equal to 3, it remains unchanged due to step 2(f). For all other positions, the selected element can be updated to either 1 or 0.

Because any $\boldsymbol{Q} \in \mathcal{Q}$ can be expressed as $\boldsymbol{Q} = \mathbf{P}' \left[\mathbf{I}_K, \mathbf{I}_K, \widetilde{\boldsymbol{Q}}'\right]'$, if we can show that any $\boldsymbol{Q}_* = \left[\mathbf{I}_K, \mathbf{I}_K, \widetilde{\boldsymbol{Q}}'_*\right]'$ can be reached from $\boldsymbol{Q}^{(t)} = \left[\mathbf{I}_K, \mathbf{I}_K, \widetilde{\boldsymbol{Q}}'\right]'$ and that any row permutation is allowed, then the irreducibility is proved. We first claim that if current state $\boldsymbol{Q}^{(t)} = \left[\mathbf{I}_K, \mathbf{I}_K, \widetilde{\boldsymbol{Q}}'\right]'$, any candidate $\boldsymbol{Q}_* = \left[\mathbf{I}_K, \mathbf{I}_K, \widetilde{\boldsymbol{Q}}'_*\right]'$ can be reached by the proposed transition strategy. With the additional condition that $J - 2K \geq 2$, we can choose to update elements in one row of $\widetilde{\boldsymbol{Q}}_*$, say row l, sequentially to all 1's in finite steps. This is always feasible because if row $l = \boldsymbol{e}_k$, there exist two other $\boldsymbol{e}_k$ rows in $\boldsymbol{Q}^{(t)}$, then 0's in row l can be updated to 1 in step 2(f); if row l is in any other form, 0's in it can be updated to 1. Since the new row l ensures every column sum is at least 3, the elements in the remaining rows of $\widetilde{\boldsymbol{Q}}$ can be updated sequentially to any possible configuration. We can choose to update them to be the same as the corresponding rows in $\widetilde{\boldsymbol{Q}}_*$. After that, row l can also be updated sequentially to be the same as row l in $\widetilde{\boldsymbol{Q}}_*$, so that $\widetilde{\boldsymbol{Q}}_*$ satisfying identifiability constraints 2 and 3 can be reached. This indicates that the Markov chain can reach all possible candidates where the position of the two identity matrices remains the same as the current state.

It is left to show that any random row permutation can be achieved by this transition, and it suffices to show that any pair of rows i and j can exchange. Without loss of generality, let the current state $\boldsymbol{Q}^{(t)} = \left[\mathbf{I}_K, \mathbf{I}_K, \widetilde{\boldsymbol{Q}}'\right]'$, and we need to prove that for any i and j, $i \neq j$, candidate $\boldsymbol{Q}_*$ of rows i and j swapped can be achieved through the transition procedure. If both $i, j > 2K$, i.e., they are rows in $\widetilde{\mathbf{Q}}$, based on the argument above, $\boldsymbol{Q}_*$ can be reached.

Now consider the case where $i \leq 2K, j > 2K$, i.e., one row is in $[\mathbf{I}_K, \mathbf{I}_K]'$ and one is in $\widetilde{\boldsymbol{Q}}$. Suppose row $i = \boldsymbol{e}_k$ that has a single 1 in the k-th column, and row j could be any other non-zero row vector. There exists another row $m = \boldsymbol{e}_k$ from the identity matrix, $m \neq i, j$. We can start the transition by updating another row l from $\widetilde{\boldsymbol{Q}}$, $l \neq j$, sequentially to be a vector of all 1's. The new row l ensures that all column sum is at least 3, and then row j can

be updated to $\boldsymbol{e}_k$ sequentially. If the original row j equals to 0 in column k, we can choose to update the elements in other columns of row i to be the same as the original row j, and update 1 in column k of row i to be 0 at last. Any 0 in row i can be updated to 1 because there exist two other rows m and j of form $\boldsymbol{e}_k$. If the original row j equals to 1 in column k, we can update the remaining elements in row i sequentially to elements of the original row j, thus we have switched the rows i and j, and then we can update row l back to its original configuration.

If both $i, j \leq 2K$, we can find another row l from $\widetilde{\boldsymbol{Q}}$. As noted above, we can switch row i with row l, switch row j with the current row i, and then switch the current row j with the current row l, thus we have switched rows i and j. □

DS2 with $B > 1$ can reach every possible state with similar procedures. Every transition of updating 1 element at a time is a special transition of DS2 with size B, where we choose B elements but just update one of them. Therefore any state can be sequentially achieved as shown in Theorem 1 through DS2 with size $B > 1$. Similarly, updating a single element is also a special transition of DS1. Therefore, DS1 and DS2 guarantee that the Markov chain is irreducible.

When $J - 2K = 1$, the only possible form of $\boldsymbol{Q} \in \mathcal{Q}$ is that $\boldsymbol{Q} = \mathbf{P}\left[\mathbf{I}_K, \mathbf{I}_K, \mathbf{1}_K\right]'$, where $\mathbf{P}$ is a $J \times J$ permutation matrix. To sample random candidates from $\mathcal{Q}$, we can simply switch two rows in every transition so as to draw dependent samples.

Furthermore, we will show that the transition proposals DS1 and DS2 are both symmetric,

Theorem 2. *DS2 proposes a symmetric transition from current $\boldsymbol{Q}^{(t)}$ to any proposed candidate $\boldsymbol{Q}_*$, i.e., $T(\boldsymbol{Q}_*, \boldsymbol{Q}^{(t)}) = T(\boldsymbol{Q}^{(t)}, \boldsymbol{Q}_*)$ where $T(x, y)$ is the transition probability from state x to y.*

Proof. It is obvious that for $\boldsymbol{Q}_* = \boldsymbol{Q}^{(t)}$, $T(\boldsymbol{Q}_*, \boldsymbol{Q}^{(t)}) = T(\boldsymbol{Q}^{(t)}, \boldsymbol{Q}_*)$ always holds. We just need to prove the equation still holds when $\boldsymbol{Q}_* \neq \boldsymbol{Q}^{(t)}$.

When $B = 1$, if $\boldsymbol{Q}_* \neq \boldsymbol{Q}^{(t)}$, then the element selected cannot be any of the following three cases: 1) a 1 from the row vector $\boldsymbol{e}_k$, 2) a 1 from the column with column sum exactly 3, and 3) a 0 from the row vector $\boldsymbol{e}_k$ when there are only two $\boldsymbol{e}_k$'s in $\boldsymbol{Q}$, because for all these cases the selected element would remain unchanged based on DS2. Other than the three cases, the selected block can have two configurations: 0 or 1, then $T(\boldsymbol{Q}^{(t)}, \boldsymbol{Q}_*) = \frac{1}{2} = T(\boldsymbol{Q}_*, \boldsymbol{Q}^{(t)})$ holds.

For $B > 1$, let p_i denote a specific path of updating steps in DS2 that changes $\boldsymbol{Q}^{(t)}$ to $\boldsymbol{Q}_*$. Then the transition probability of a given path p_i based on steps 2(a), 2(c), and 2(f) should be,

$$T(\boldsymbol{Q}^{(t)}, \boldsymbol{Q}_* | p_i) = \overbrace{\frac{1}{K\binom{J}{B}}}^{2(a)} \times \overbrace{\frac{1}{\prod_{i \neq k} \binom{b_i}{2-l_i}^{\mathcal{I}(l_i<2)}}}^{2(c)} \times \overbrace{\frac{1}{\sum_{c=\max\{3, m+k_1\}}^{B-k_0+m} \binom{B-k_0-k_1}{c-m-k_1}}}^{2(f)}. \tag{3.14}$$

When the block is chosen in step 2(a), k_1 in step 2(b) is determined, and b_i and l_i in step 2(c) are also determined because the number positions need to be fixed as 0 or 1 remain the same for the given block. So k_1, b_i, l_i and k_0 are all determined by the block, and m is also determined because m is the number of 1's outside the block. Therefore the transition probability (3.14) remains the same for all the paths from $\boldsymbol{Q}^{(t)}$ to $\boldsymbol{Q}_*$.

For every path p_i changing $\boldsymbol{Q}^{(t)}$ to $\boldsymbol{Q}_*$ with the chosen block B_i and chosen positions in steps 2(b) and 2(c), there is a corresponding path q_i that chooses the same block and positions and changes $\boldsymbol{Q}_*$ to $\boldsymbol{Q}^{(t)}$. Based on the argument above, the transition probability of path q_i is the same as (3.14), i.e., $T(\boldsymbol{Q}^{(t)}, \boldsymbol{Q}_* | p_i) = T(\boldsymbol{Q}_*, \boldsymbol{Q}^{(t)} | q_i)$. So the transition probability of DS2 satisfies:

$$T(\boldsymbol{Q}_*, \boldsymbol{Q}^{(t)}) = \sum_{\substack{q_i \in \{\text{all paths from} \\ \boldsymbol{Q}_* \text{ to } \boldsymbol{Q}^{(t)}\}}} T(\boldsymbol{Q}_*, \boldsymbol{Q}^{(t)} | q_i) = \sum_{\substack{p_i \in \{\text{all paths from} \\ \boldsymbol{Q}^{(t)} \text{ to } \boldsymbol{Q}_*\}}} T(\boldsymbol{Q}^{(t)}, \boldsymbol{Q}_*, | p_i) = T(\boldsymbol{Q}^{(t)}, \boldsymbol{Q}_*) \tag{3.15}$$

□

3.2.5 Metropolis Sampling for $\boldsymbol{Q}$

The proposed transition functions to generate matrix $\boldsymbol{Q}$ are irreducible and symmetric, which implies Metropolis sampling can be implemented to draw $\boldsymbol{Q}$ from the posterior $p(\boldsymbol{Q}|\boldsymbol{Y}, \boldsymbol{s}, \boldsymbol{g}, \boldsymbol{\alpha})$. Given the current iteration at time t:

1. Draw a candidate $\boldsymbol{Q}_*$ through DS2.

2. Draw $U \sim Uniform(0, 1)$ and update

$$\boldsymbol{Q}^{(t)} = \begin{cases} \boldsymbol{Q}_* & \text{if } U \leq r(\boldsymbol{Q}^{(t-1)}, \boldsymbol{Q}_*) \\ \boldsymbol{Q}^{(t-1)} & \text{otherwise} \end{cases}$$

   where $r(\boldsymbol{Q}^{(t-1)}, \boldsymbol{Q}_*) = \min\left\{1, \frac{p(\boldsymbol{Q}_*|\boldsymbol{Y},\boldsymbol{s},\boldsymbol{g},\boldsymbol{\alpha})}{p(\boldsymbol{Q}^{(t-1)}|\boldsymbol{Y},\boldsymbol{s},\boldsymbol{g},\boldsymbol{\alpha})}\right\}$.

Because of the ratio, we can use the posteriors with unknown normalizing constant to update $\boldsymbol{Q}$.

Furthermore, a Metropolis within Gibbs algorithm can be implemented to sample the parameters $\boldsymbol{Q}$, $\boldsymbol{s}$, $\boldsymbol{g}$, $\boldsymbol{\alpha}$, and $\boldsymbol{\pi}$ from the posterior distribution. The method is described as the following:

1. Randomly initialize $\boldsymbol{\alpha}^{(0)}$, $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{Q}^{(0)} \in \mathcal{Q}$, $\boldsymbol{g}^{(0)}$ and $\boldsymbol{s}^{(0)}$ such that $0 \leq g_j^{(0)} \leq 1 - s_j^{(0)} \leq 1$ for $j = 1, \cdots, J$.

2. At iteration t, update $\boldsymbol{s}^{(t)}$, $\boldsymbol{g}^{(t)}$, $\boldsymbol{\alpha}^{(t)}$, $\boldsymbol{\pi}^{(t)}$, and $\boldsymbol{Q}^{(t)}$:

   (a) Update $\boldsymbol{g}^{(t)} \sim p\left(\boldsymbol{g}|\boldsymbol{Y}, \boldsymbol{s}^{(t-1)}, \boldsymbol{\alpha}^{(t-1)}, \boldsymbol{Q}^{(t-1)}\right)$.

   (b) Update $\boldsymbol{s}^{(t)} \sim p\left(\boldsymbol{s}|\boldsymbol{Y}, \boldsymbol{g}^{(t)}, \boldsymbol{\alpha}^{(t-1)}, \boldsymbol{Q}^{(t-1)}\right)$.

   (c) Update $\boldsymbol{\alpha}^{(t)} \sim p\left(\boldsymbol{\alpha}|\boldsymbol{Y}, \boldsymbol{s}^{(t)}, \boldsymbol{g}^{(t)}, \boldsymbol{\pi}^{(t-1)}, \boldsymbol{Q}^{(t-1)}\right)$.

(d) Update $\boldsymbol{\pi}^{(t)} \sim p\left(\boldsymbol{\pi}|\boldsymbol{\alpha}^{(t)}\right)$.

(e) Update $\boldsymbol{Q}^{(t)}$ using Metropolis steps as aforementioned.

Note that steps 2(a) through 2(d) are Gibbs updates with full conditional distributions discussed in Culpepper (2015).

3.2.6 Constrained Gibbs Sampling for $\boldsymbol{Q}$

Theorem 1 has proved that updating a single element in $\boldsymbol{Q}$ through DS2 ensures the irreducibility of the Markov chain. Based on the sampling strategy of DS2 with special case of $B = 1$, we propose an alternative Gibbs sampling algorithm for $\boldsymbol{Q}$ subject to identifiability constraints. Given the current iteration at time t the constrained Gibbs sampling algorithm proceeds as the following:

1. Update $\boldsymbol{s}^{(t)}$, $\boldsymbol{g}^{(t)}$, $\boldsymbol{\alpha}^{(t)}$, $\boldsymbol{\pi}^{(t)}$ through Gibbs sampling same as aforementioned.

2. Update $q_{jk}^{(t)}$, $j = 1, \ldots, J$, $k = 1, \ldots, K$ through Gibbs sampling:

   (a) If $q_{jk}^{(t-1)}$ is in one of the following three positions: 1) a 1 from the row vector $\boldsymbol{e}_k$, 2) a 1 from the column with column sum exactly 3, and 3) a 0 from the row vector $\boldsymbol{e}_k$ where there are only two $\boldsymbol{e}_k$'s in the current $\boldsymbol{Q}$, $q_{jk}^{(t)}$ remains the same as $q_{jk}^{(t-1)}$.

   (b) Otherwise, $q_{jk}^{(t-1)}$ can be updated to 0 or 1. Sample $q_{jk}^{(t)} = i$ with probability proportional to $p(\boldsymbol{Y}|\boldsymbol{s}^{(t)}, \boldsymbol{g}^{(t)}, \boldsymbol{\alpha}^{(t)}, \boldsymbol{Q}_{new}^{(t)}, q_{jk}^{(t)} = i, \boldsymbol{Q}_{old}^{(t-1)}), i = 0, 1$, where $\boldsymbol{Q}_{new}^{(t)}$ refers to the elements of the current $\boldsymbol{Q}$ that have already been updated, and $\boldsymbol{Q}_{old}^{(t-1)}$ refers to the elements of the current $\boldsymbol{Q}$ that have not been updated.

As shown in Theorem 1, the constrained Gibbs sampling ensures that the Markov chain is irreducible and the sampled $\boldsymbol{Q}$ always satisfies identifiability constraints.

3.2.7 Computing $\hat{\boldsymbol{Q}}$, Column Permuting

In the Monte Carlo study, the estimated $\hat{\boldsymbol{Q}}$ is taken as the mode of the sampled $\boldsymbol{Q}$ matrices for posterior inference. In order to find the mode, we use a bijective mapping from binary sequence to integer number (e.g., see Chung, 2014; von Davier, 2014), and transform the column vectors of sampled $\boldsymbol{Q}$ into integer numbers. For instance, $\boldsymbol{v}'\boldsymbol{Q}_k$ is a bijection for $\boldsymbol{v}' = \left(2^{J-1}, 2^{J-2}, \ldots, 2, 1\right)$. Every $\boldsymbol{Q}$ can be recorded as a K dimensional integer vector. We permute the vector in a decreasing order so that column permutation of $\boldsymbol{Q}$ is allowed, and count the occurrence of each vector. The estimated $\hat{\boldsymbol{Q}}$ corresponds to a vector that appears most, and it satisfies the identifiability constraints. Note that finding the posterior mode in this manner eliminates the problem of columns permuting when summarizing $\hat{\boldsymbol{Q}}$.

3.3 Monte Carlo Simulation Studies

This section reports results from a Monte Carlo simulation study on the relative performance of the proposed constrained MCMC procedures for estimating $\boldsymbol{Q}$ with Chung's (2014) unconstrained Gibbs sampler.

3.3.1 Overview

The Monte Carlo simulation study examined recovery of $\boldsymbol{Q}$ for different sample sizes (i.e., $N = 500$, 1,000, 2,000, and 4,000), numbers of attributes (i.e., $K = 3$, 4), and correlations among the latent attributes (i.e., $\rho = 0$, 0.05, 0.15, and 0.25). The simulation study specified

the true unknown $\boldsymbol{Q}$ matrices, $\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$

$$\boldsymbol{Q}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \boldsymbol{Q}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}$$

which both satisfy the aforementioned identifiability constraints. For the $\rho = 0$ case attribute profiles were generated uniformly from the 2^K configurations. For cases where $\rho > 0$ attribute dependence is introduced using the method of Chiu et al. (2009). Suppose $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_K)$ follows multivariate normal distribution $N(\mathbf{0}, \boldsymbol{\Sigma})$ where the covariance matrix $\boldsymbol{\Sigma}$ has unit variance and a common correlation ρ:

$$\boldsymbol{\Sigma} = (1 - \rho)\mathbf{I}_K + \rho\mathbf{1}\mathbf{1}'.$$

We consider the situation where $\rho = 0.05, 0.15$ and 0.25 and the attribute α for each subject is given by

$$\alpha_k = \begin{cases} 1 & \text{if } \theta_k \geq 0 \\ 0 & \text{otherwise} \end{cases}, \ k = 1, \ldots, K.$$

We generated 100 independent simulated datasets for each condition to assess the performance of three MCMC methods. The independence sampler and the DS1 sampler mixed more slowly than the DS2 sampler with acceptance rates for $\boldsymbol{Q}$ less than 0.5% and 5%, respectively. In contrast, the acceptance rate for DS2 proposal was between 18% - 25%. In preliminary investigations we considered $B = K$ and $B = 2K$ and the simulation study and application employ $B = 2K$ rather than $B = K$ given the acceptance rate was higher and approximately 20%.

The simulation study also examined the performance of the constrained Gibbs sampler with block size of $B = 1$ and Chung's (2014) unconstrained Gibbs sampler. In all tables of results, "MH" refers to our method of Metropolis sampling DS2 with a block size of $B = 2K$, "CGibbs" refers to our developed constrained Gibbs sampler that sequentially updates each element of $\boldsymbol{Q}$, and "Gibbs" refers to the unconstrained sampler. For each replication we ran a Markov chain of length 30,000 with a burn-in length of 15,000 for all MCMC methods. The starting value for $\boldsymbol{s}$, $\boldsymbol{g}$, $\boldsymbol{\pi}$ are sampled from the prior distributions, and the starting value of $\boldsymbol{Q}$ is randomly drawn from the sample space $\mathcal{Q}$. Each replication recorded the $\boldsymbol{Q}$ that maximized the posterior mode to compute the overall accuracy as well as entry-wise accuracy of the MCMC methods.

For the diagnostic of convergence, we used Geweke's diagnostic (Geweke, 1992) on -2 times log of the posterior density to determine the burn-in period (Cowles and Carlin, 1996). For iterations from 15,000 to 30,000 in chains from MH and CGibbs sampler, the reported Z-scores are -0.218 and -0.628 respectively for $K = 3, N = 4,000$, and -1.36 and -0.705 respectively for $K = 4, N = 4,000$, indicating that there is no evidence against the convergence of the joint posterior.

3.3.2 Results

Table 3.1: Summary of Recovery Rate of $\boldsymbol{Q}$ by Sample Size (N), Number of Attributes (K), and Attribute Dependence (ρ) for the Metropolis-Hasting (MH), constrained Gibbs (CGibbs), and unconstrained Gibbs (Gibbs) Samplers.

K	N	ρ	$\hat{\boldsymbol{Q}} = \boldsymbol{Q}$			Entry-wise Average		
			MH	CGibbs	Gibbs	MH	CGibbs	Gibbs
3	500	0	91	95	94	95.89	98.22	97.12
3	1000	0	94	99	95	97.59	99.61	97.93
3	2000	0	96	92	90	98.52	97.33	95.11
3	4000	0	98	88	91	99.56	96.33	95.42
3	500	0.05	87	99	82	93.25	99.63	92.96
3	1000	0.05	92	99	90	94.67	99.64	94.41
3	2000	0.05	93	95	90	97.89	98.52	95.18
3	4000	0.05	94	88	88	98.78	96.33	95.20
3	500	0.15	93	99	91	97.42	99.63	95.44
3	1000	0.15	95	99	92	98.51	99.67	95.89
3	2000	0.15	96	95	94	99.29	98.10	97.65
3	4000	0.15	95	91	89	99.35	97.39	94.04
3	500	0.25	92	99	90	98.70	99.98	94.55
3	1000	0.25	96	98	94	99.42	99.65	96.20
3	2000	0.25	96	94	93	99.92	98.10	95.83
3	4000	0.25	97	89	91	99.88	95.57	95.76
4	500	0	60	97	59	91.42	99.92	89.81
4	1000	0	67	91	67	93.11	97.29	92.24
4	2000	0	76	79	73	94.52	94.54	93.96
4	4000	0	87	51	82	95.50	87.02	95.25
4	500	0.05	37	98	40	82.00	99.13	83.39
4	1000	0.05	52	94	58	88.57	98.36	89.87
4	2000	0.05	48	90	53	88.02	97.44	89.28
4	4000	0.05	53	53	51	89.50	89.62	88.94
4	500	0.15	34	96	40	81.61	99.43	83.09
4	1000	0.15	44	88	60	84.87	96.83	92.00
4	2000	0.15	55	90	53	89.13	97.17	88.97
4	4000	0.15	56	74	52	89.92	91.64	89.19
4	500	0.25	35	97	37	81.78	99.58	82.94
4	1000	0.25	43	96	58	84.67	98.57	90.24
4	2000	0.25	55	85	54	89.87	95.64	89.80
4	4000	0.25	55	79	51	90.09	94.07	89.30

Note. $\hat{\boldsymbol{Q}} = \boldsymbol{Q}$ refers to the number of correctly recovered $\boldsymbol{Q}$ out of 100 replications. Entry-wise average refers to average percentage of correctly recovered entries of $\boldsymbol{Q}$.

Table 3.1 reports the accuracy of the three MCMC methods (i.e., MH, CGibbs, and Gibbs) across N, ρ, and K. Specifically, Table 3.1 presents the number of replications where the Bayesian methods correctly recovered all elements of $\boldsymbol{Q}$ (i.e., $\hat{\boldsymbol{Q}} = \boldsymbol{Q}$) in addition to the average entry-wise accuracy, which was computed as the percentage of correctly estimated entries. In general, our methods (i.e., MH and CGibbs) successfully estimated $\boldsymbol{Q}$ and demonstrated greater accuracy than the unconstrained Gibbs sampler.

The CGibbs approach tended to provide more accurate estimates of $\boldsymbol{Q}$ for smaller sample sizes (i.e., $N = 500$ and 1,000) whereas the MH and Gibbs samplers were less accurate for smaller sample sizes and the difference in recovery rates was more pronounced for $K = 4$. It is important to note that the three Bayesian estimators outperformed Chen et al.'s (2015) L_1 penalty approach for sample sizes of 500 and 1,000. For example, for the $\rho = 0$ and $N = 500$ case Chen et al.'s (2015) L_1 approach correctly recovered 38 out of 100 $\boldsymbol{Q}$ matrices for $K = 3$ and 20 $\boldsymbol{Q}$ matrices for $K = 4$. In contrast, all three Bayesian methods recovered more than 90% of $\boldsymbol{Q}$ matrices for $K = 3$ and the CGibbs approach recovered 97 out of 100 $\boldsymbol{Q}$ matrices when $K = 4$.

Sample size impacted the performance of the Bayesian methods. In fact, Chen et al.'s (2015) L_1 outperformed the Bayesian methods for sample sizes of $N = 2,000$ and $4,000$. The CGibbs approach tended to be less accurate when N = 4,000; however, the proposed MH sampler outperformed the unconstrained Gibbs sampler in these cases. Table 3.1 provides evidence that $\boldsymbol{Q}$ was most difficult to recover when N = 4,000 and $K = 4$. Both the MH and Gibbs samplers became less accurate as ρ increased. For $K = 4$ and $\rho > 0$ cases, the constrained Gibbs sampler consistently reports the best recovery rates for all correlation settings.

Figures 3.1 and 3.2 report the mean squared error (MSE) of the estimated guessing and slipping item parameters. The constrained Gibbs sampler tends to have the smallest MSEs among the three Bayesian algorithms, and the unconstrained Gibbs sampler has the largest MSEs. For instance, Figure 3.1 reports MSE for the $\rho = 0$ case and shows that the MH

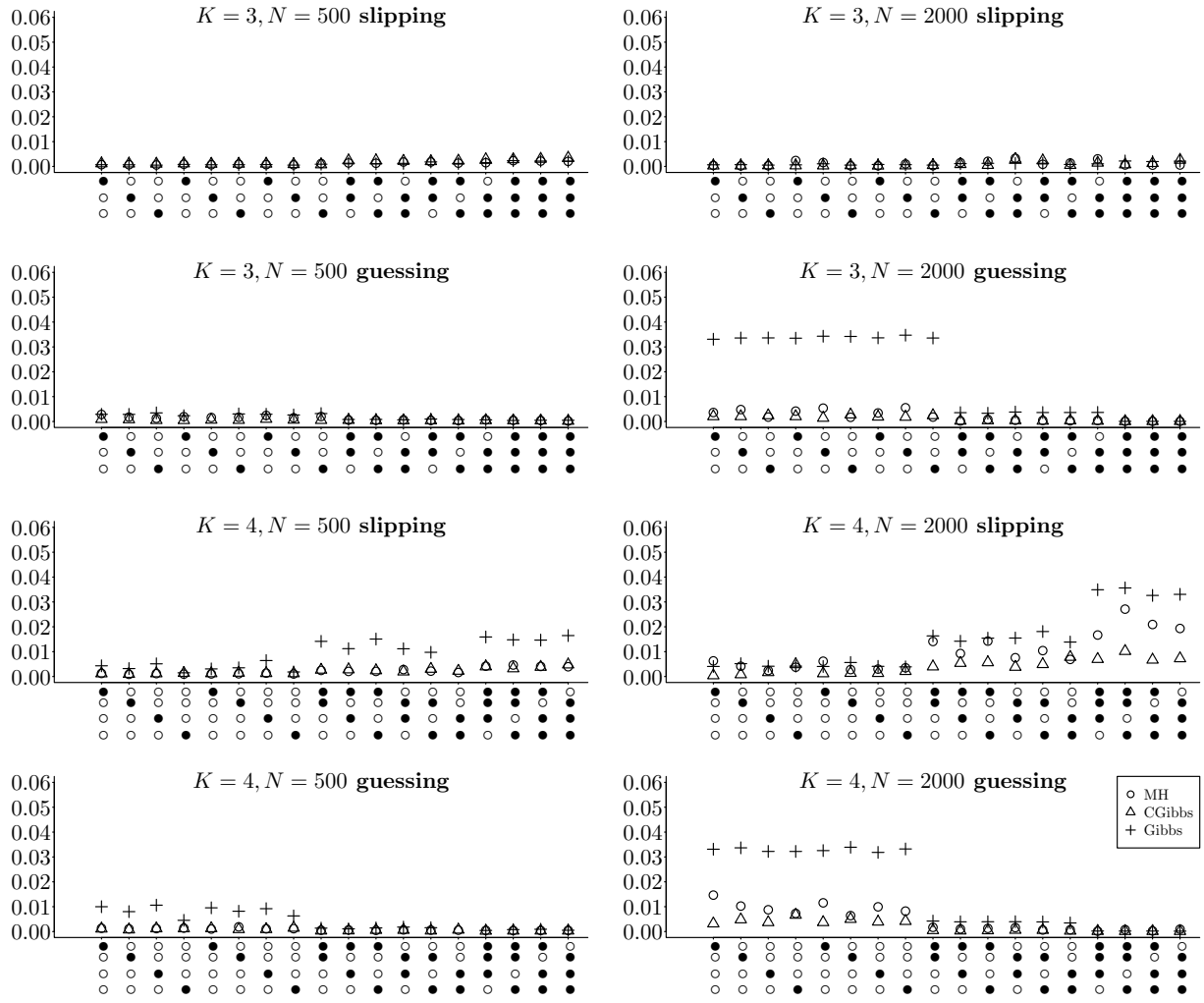


Figure 3.1: Mean squared errors for slipping and guessing parameters for independent attributes with $K = 3, 4$, and $N = 500, 2000$. The x-axis labels refer to the corresponding entries in $\boldsymbol{Q}$, $\circ$ refers to 0 and $\bullet$ refers to 1. The legend at the last plot is the same for all plots.

and CGibbs algorithms produced more accurate item parameter estimates than the unconstrained Gibbs method. For $N = 2{,}000$ and $K = 3$, the MSEs of guessing parameters for items requiring single attributes are larger for the unconstrained Gibbs than items requiring multiple attributes. Additionally, the unconstrained Gibbs has larger MSE for slipping parameters in the $K = 4$ case. Furthermore, Figure 3.2 reports the item parameter MSEs for $\rho = 0.05$ and 0.25. Similar to the independent case, the MSEs for the unconstrained

sampler are larger than for the constrained MH and CGibbs methods. One explanation for the difference in recovery of item parameters can be attributed to the fact that our methods restricted $\boldsymbol{Q}$ to be identifiable whereas the unconstrained sampler can visit values for $\boldsymbol{Q}$ where the item parameters are not identified.

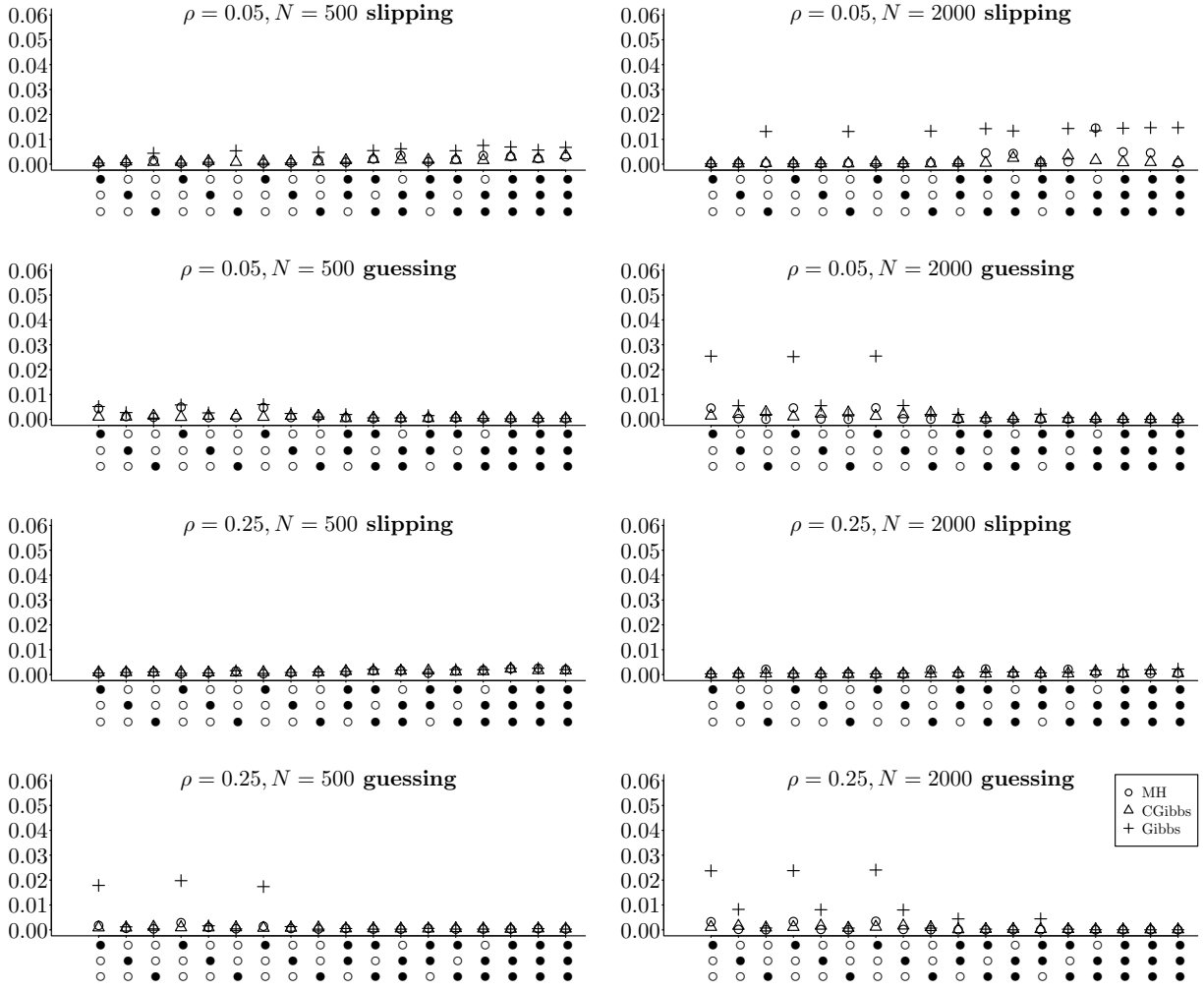


Figure 3.2: Mean squared error for slipping and guessing parameters for dependent attributes with $K = 3$, $\rho = 0.05, 0.25$, and $N = 500, 2000$. The x-axis labels refer to the corresponding entries in $\boldsymbol{Q}$, $\circ$ refers to 0 and $\bullet$ refers to 1. The legend at the last plot is the same for all plots.

3.4 Application to Fraction-Subtraction Data

In this section we compare our methods and the unconstrained Gibbs sampler (Chung, 2014) in an application to estimate $\boldsymbol{Q}$ for Tatsuoka's Fraction-Subtraction data set (Tatsuoka, 2002, 1984). The data set contains responses to $J = 20$ items from $N = 536$ middle school students, and has been widely analyzed (e.g., Chen et al., 2015; DeCarlo, 2012; de la Torre, 2008; de la Torre and Douglas, 2004, 2008; Mislevy and Wilson, 1996). In fact, Table 3.2 presents an expert derived $\boldsymbol{Q}$ matrix with the following eight skills:

1. Convert a whole number to fraction,
2. Separate a whole number from fraction,
3. Simplify before subtraction,
4. Find a common denominator,
5. Borrow from the whole number part,
6. Column borrow to subtract the second numerator from the first
7. Subtract numerators
8. Reduce answers to simplest form.

In Table 3.2 almost all of the items require Attribute 7, and several attributes are required at the same time, indicating that 8 attributes might be too many for 20 items. Hence, this section reports an estimate of $\boldsymbol{Q}$ for $K = 3$ and 4.

Table 3.3 presents the estimated $\hat{\boldsymbol{Q}}$, slipping, and guessing parameters from both the MH and unconstrained Gibbs methods for $K = 3$. Note that $\hat{\boldsymbol{Q}}$ for the constrained Gibbs sampler was the same as reported in Table 3.3 for the MH sampler. Most entries of the estimated $\hat{\boldsymbol{Q}}$ are the same for the constrained and unconstrained methods; however, an important difference is that the unconstrained Gibbs sampler did not yield a $\hat{\boldsymbol{Q}}$ that satisfies

Table 3.2: Expert Specified Fraction-Subtraction Data $\boldsymbol{Q}$ Matrix with 8 attributes

	1	2	3	4	5	6	7	8
$\frac{5}{3} - \frac{3}{4}$	0	0	0	1	0	1	1	0
$\frac{3}{4} - \frac{3}{8}$	0	0	0	1	0	0	1	0
$\frac{5}{6} - \frac{1}{9}$	0	0	0	1	0	0	1	0
$3\frac{1}{2} - 2\frac{3}{2}$	0	1	1	0	1	0	1	0
$4\frac{3}{5} - 3\frac{4}{10}$	0	1	0	1	0	0	1	1
$\frac{6}{7} - \frac{4}{7}$	0	0	0	0	0	0	1	0
$3 - 2\frac{1}{5}$	1	1	0	0	0	0	1	0
$\frac{2}{3} - \frac{2}{3}$	0	0	0	0	0	0	1	0
$3\frac{7}{8} - 2$	0	1	0	0	0	0	0	0
$4\frac{4}{12} - 2\frac{7}{12}$	0	1	0	0	1	0	1	1
$4\frac{1}{3} - 2\frac{4}{3}$	0	1	0	0	1	0	1	0
$\frac{11}{8} - \frac{1}{8}$	0	0	0	0	0	0	1	1
$3\frac{3}{8} - 2\frac{5}{6}$	0	1	0	1	1	0	1	0
$3\frac{4}{5} - 3\frac{2}{5}$	0	1	0	0	0	0	1	0
$2 - \frac{1}{3}$	1	0	0	0	0	0	1	0
$4\frac{5}{7} - 1\frac{4}{7}$	0	1	0	0	0	0	1	0
$7\frac{3}{5} - 2\frac{4}{5}$	0	1	0	0	1	0	1	0
$4\frac{1}{10} - 2\frac{8}{10}$	0	1	0	0	1	1	1	0
$4 - 1\frac{4}{3}$	1	1	1	0	1	0	1	0
$4\frac{1}{3} - 1\frac{5}{3}$	0	1	1	0	1	0	1	0

Table 3.3: Estimated $\boldsymbol{Q}$, Slipping ($\hat{\boldsymbol{s}}$), and Guessing ($\hat{\boldsymbol{g}}$) parameters for $K = 3$ from the Constrained Metropolis-Hastings (MH) and Unconstrained Gibbs Samplers.

	MH					Unconstrained Gibbs				
Item	$\hat{\boldsymbol{Q}}$			$\hat{\boldsymbol{s}}$	$\hat{\boldsymbol{g}}$	$\hat{\boldsymbol{Q}}$			$\hat{\boldsymbol{s}}$	$\hat{\boldsymbol{g}}$
$\frac{5}{3} - \frac{3}{4}$	1	0	0	0.14	0.04	1	0	0	0.14	0.04
$\frac{3}{4} - \frac{3}{8}$	1	0	0	0.07	0.05	1	0	0	0.07	0.05
$\frac{5}{6} - \frac{1}{9}$	1	0	0	0.14	0.01	1	0	0	0.14	0.01
$3\frac{1}{2} - 2\frac{3}{2}$	0	1	0	0.13	0.21	0	1	1	0.12	0.18
$4\frac{3}{5} - 3\frac{4}{10}$	1	0	1	0.21	0.30	1	0	1	0.22	0.31
$\frac{6}{7} - \frac{4}{7}$	0	0	1	0.04	0.31	0	0	1	0.04	0.30
$3 - 2\frac{1}{5}$	1	0	1	0.24	0.05	1	0	0	0.35	0.03
$\frac{2}{3} - \frac{2}{3}$	1	1	0	0.05	0.58	1	1	0	0.06	0.58
$3\frac{7}{8} - 2$	0	0	1	0.17	0.34	0	0	1	0.25	0.34
$4\frac{4}{12} - 2\frac{7}{12}$	0	1	0	0.23	0.03	0	1	1	0.23	0.03
$4\frac{1}{3} - 2\frac{4}{3}$	0	1	1	0.08	0.07	0	1	1	0.07	0.06
$1\frac{1}{8} - \frac{1}{8}$	0	0	1	0.09	0.21	0	0	1	0.09	0.20
$3\frac{3}{8} - 2\frac{5}{6}$	1	1	1	0.33	0.02	1	1	1	0.35	0.02
$3\frac{4}{5} - 3\frac{2}{5}$	0	0	1	0.07	0.10	0	0	1	0.07	0.09
$2 - \frac{1}{3}$	1	0	1	0.17	0.11	1	0	1	0.25	0.06
$4\frac{5}{7} - 1\frac{4}{7}$	0	0	1	0.11	0.13	0	0	1	0.11	0.12
$7\frac{3}{5} - 2\frac{4}{5}$	0	1	1	0.14	0.05	0	1	1	0.14	0.04
$4\frac{1}{10} - 2\frac{8}{10}$	0	1	1	0.16	0.13	0	1	1	0.16	0.13
$4 - 1\frac{4}{3}$	1	1	1	0.32	0.03	1	1	1	0.33	0.03
$4\frac{1}{3} - 1\frac{5}{3}$	0	1	1	0.19	0.02	0	1	1	0.18	0.02

Note. $\hat{\boldsymbol{Q}}$ estimated from CGibbs is the same as the MH algorithm above.

Table 3.4: Estimated $\boldsymbol{Q}$, Slipping ($\hat{\boldsymbol{s}}$), and Guessing ($\hat{\boldsymbol{g}}$) parameters for $K = 4$ from the Constrained Metropolis-Hastings (MH) and Unconstrained Gibbs Samplers.

	MH						Gibbs					
Item	$\hat{\boldsymbol{Q}}$				$\hat{\boldsymbol{s}}$	$\hat{\boldsymbol{g}}$	$\hat{\boldsymbol{Q}}$				$\hat{\boldsymbol{s}}$	$\hat{\boldsymbol{g}}$
$\frac{5}{3} - \frac{3}{4}$	1	0	0	0	0.11	0.04	1	0	0	0	0.12	0.04
$\frac{3}{4} - \frac{3}{8}$	1	0	0	0	0.04	0.05	1	0	0	0	0.04	0.04
$\frac{5}{6} - \frac{1}{9}$	1	0	0	0	0.12	0.01	1	0	0	0	0.12	0.01
$3\frac{1}{2} - 2\frac{3}{2}$	0	1	0	0	0.12	0.15	0	1	0	0	0.12	0.16
$4\frac{3}{5} - 3\frac{4}{10}$	1	0	1	0	0.16	0.34	1	0	1	0	0.21	0.31
$\frac{6}{7} - \frac{4}{7}$	0	0	1	0	0.04	0.30	0	0	1	0	0.04	0.28
$3 - 2\frac{1}{5}$	0	0	0	1	0.24	0.02	0	0	0	1	0.20	0.01
$\frac{2}{3} - \frac{2}{3}$	0	0	1	0	0.06	0.54	0	1	0	0	0.06	0.56
$3\frac{7}{8} - 2$	0	0	1	0	0.17	0.40	0	0	1	0	0.25	0.33
$4\frac{4}{12} - 2\frac{7}{12}$	0	1	0	0	0.23	0.03	0	1	1	0	0.23	0.03
$4\frac{1}{3} - 2\frac{4}{3}$	0	1	1	0	0.08	0.07	0	1	1	0	0.08	0.07
$1\frac{1}{8} - \frac{1}{8}$	0	0	1	0	0.09	0.21	0	0	1	0	0.09	0.17
$3\frac{3}{8} - 2\frac{5}{6}$	1	1	1	1	0.33	0.02	1	1	1	0	0.34	0.02
$3\frac{4}{5} - 3\frac{2}{5}$	0	0	1	0	0.07	0.10	0	0	1	0	0.07	0.06
$2 - \frac{1}{3}$	0	0	0	1	0.11	0.04	0	0	1	1	0.11	0.04
$4\frac{5}{7} - 1\frac{4}{7}$	0	0	1	0	0.11	0.13	0	0	1	0	0.12	0.12
$7\frac{3}{5} - 2\frac{4}{5}$	0	1	1	0	0.14	0.06	0	1	1	0	0.14	0.05
$4\frac{1}{10} - 2\frac{8}{10}$	0	1	1	1	0.20	0.13	0	1	1	0	0.16	0.13
$4 - 1\frac{4}{3}$	0	1	1	1	0.32	0.03	0	1	1	1	0.26	0.03
$4\frac{1}{3} - 1\frac{5}{3}$	0	1	1	0	0.19	0.02	0	1	1	0	0.19	0.02

Note. $\hat{\boldsymbol{Q}}$ estimated from CGibbs is the same as the MH algorithm above.

the identifiability constraints. By looking at items requiring only one attribute, we can interpret the three attributes as (1) finding common denominator, (2) borrowing from the integer part and (3) applying subtraction for integer and fraction parts separately. For items $3\frac{1}{2} - 2\frac{3}{2}$ and $4\frac{4}{12} - 2\frac{7}{12}$ our method suggests that these items only require the second attribute whereas the unconstrained sampler suggests two attributes are needed for success. Similarly, our method suggests the item $3 - 2\frac{1}{5}$ requires the first and last attributes whereas the unconstrained sampler estimated only a single attribute. The estimated slipping and guessing parameters intuitively correspond to the easiness of each item. For instance, item $\frac{2}{3} - \frac{2}{3}$ and item $3\frac{7}{8} - 2$ have large guessing parameters, and item $3\frac{3}{8} - 2\frac{5}{6}$ and $4 - 1\frac{4}{3}$ have large slipping parameters. Furthermore, most items requiring a single attribute tend to have smaller guessing parameters. Table 3.3 shows that the differences in estimates for $\boldsymbol{Q}$ translates into differences in item parameters between the two methods. For instance, the estimated slipping parameters of item $3 - 2\frac{1}{5}$, $3\frac{7}{8} - 2$, and $2 - \frac{1}{3}$ significantly differ between the two methods.

Table 3.4 presents the results for $K = 4$. Again, constrained Gibbs sampling estimated the same $\hat{\boldsymbol{Q}}$ as MH sampling and the results for MH and unconstrained Gibbs sampling are similar. Our method identifies item $3 - 2\frac{1}{5}$ and $2 - \frac{1}{3}$ as testing the same single attribute whereas the unconstrained approach indicates the latter item involves more attributes. The estimated slipping parameters of the item $3 - 2\frac{1}{5}$ and $3\frac{7}{8} - 2$ still differ here.

3.5 Discussion

The fundamental building block of a cognitive diagnosis model is the $\mathbf{Q}$ matrix. However, estimation of this discrete parameter has been the most elusive statistical challenge. While many models have been developed for cognitive diagnosis, with challenging parameter estimation issues of their own, $\mathbf{Q}$ has traditionally been taken as known. This is limiting in practice, because it requires expert opinion and any two experts are likely to disagree on

many entries, or even on the number of attributes. Statistical methods for exploring the structure of $\mathbf{Q}$ are desirable for much the same reason that exploratory factor analysis has proven useful. Specifying the nonzero loadings in confirmatory factor analysis is essentially the same as specifying $\mathbf{Q}$, but exploratory factor analysis has become more widely used because it allows for a wider scope of possibilities than confirmatory factor analysis. However, the discrete nature of $\mathbf{Q}$, and the reliance upon Boolean algebra rather than linear algebra, has made estimation of $\mathbf{Q}$ a difficult problem.

The introduction of a consistent estimator of $\mathbf{Q}$ in Liu et al. (2013) offered hope for solving this problem, and recent years have seen several advances in estimation of $\mathbf{Q}$. The Bayesian approach of this research has borrowed from the identifiability theory of Chen et al. (2015), in order to implement a fully Bayesian approach capable of exploring all possible $\mathbf{Q}$ matrices. By enforcing these constraints as well as constraints on permutations of columns, a trivial source of unidentifiability, the number of possible $\mathbf{Q}$ matrices is reduced from $2^{J\times K}$, but is still enormous. Much of this research has concerned efficient implementation of MCMC to quickly search the space of identifiable models in order to provide a quality estimator in a practical amount of time. The sample sizes and test lengths we have studied proved quite manageable, and accurate estimates of all model parameters, including $\mathbf{Q}$, could be obtained. An ongoing question will be the scope of the method, and how the computational complexity changes with N, J and especially K.

An issue worthy of further investigation is how to best summarize the posterior distribution of the discrete $\mathbf{Q}$. The simulations considered estimation of $\mathbf{Q}$ by taking the most frequent value observed in the Markov chain. This proved effective, but may be problematic as K increases, considering the exponentially increasing number of candidates. The concern is that a tremendously long chain might be required when using the mode as the estimator in a parameter space of such great cardinality. Another promising method is to simply take the entrywise modes to comprise the estimator and additional research is needed. Another fruitful topic for future research will be extensions to models beyond the DINA. The MCMC

algorithms for other models should be similar, and the main difference will be to discern what restrictions are needed for Bayesian formulations of other models to be identifiable.

The value of exploratory methods for determining $\mathbf{Q}$ will ultimately be decided through applications. This research provides an alternative to relying on experts for specifying $\mathbf{Q}$. However, the ideal scenario would be to compare $\mathbf{Q}$ matrices developed by experts with those developed statistically, in order to reach compromises that allow both a good model fit and a useful interpretation. It is also quite conceivable that a portion of $\mathbf{Q}$ could be determined by experts, which would greatly enhance the interpretation and reduce the space to explore statistically. A special case of this would encompass the problem of introducing new items to an item bank. If $\mathbf{Q}$ is known for a subset of the items with an appropriate span, then calibration of the remainder of $\mathbf{Q}$ and other item parameters could readily be accomplished.

Chapter 4

Learning trajectories in dynamic CDMs

4.1 Introduction

The increasing presence of electronic and online learning resources presents challenges and opportunities for psychometric techniques that can assist in the measurement of abilities and even hasten their mastery. Cognitive diagnosis models (CDMs) are ideal for tracking many fine-grained skills that comprise a domain, and can assist in carefully navigating through the training and assessment of these skills in e-learning applications. We consider the coupling of models for skill acquisition with item response models in cognitive diagnosis. Modeling learning is a natural direction for skills diagnosis with the increased presence of online training and interventions designed to promote skill acquisition.

Traditional research in CDMs has assumed that the vector of latent skills under diagnosis is static throughout an assessment. However, e-learning and intelligent tutoring systems provide opportunities to provide training interspersed with assessment, so it is more accurate to think of the latent attribute vector dynamically. Only recently have CDMs been considered for changing attribute vectors, and this has mostly been in the longitudinal data setting (Kaya and Leite, 2017; Li et al., 2015), in which considerable time could expire between two assessments within which the latent trait is assumed static. Li et al. (2015) used the DINA (Junker and Sijtsma, 2001) model together with a transition model as a means of measuring the effects of an educational intervention. Through simulation studies Kaya and Leite (2017) studied such transition models for longitudinal applications using both the DINA and the DINO (Templin and Henson, 2006*b*) models.

In e-learning settings one might reasonably expect that learning could take place between items or between brief learning modules with little separation in time. Studer (2012) proposed the Parameter Driven Process for Change method which indirectly tracks learning by assessing a student's membership in a small number of possible latent states. The number of latent states would be much fewer than the 2^K attribute profiles that exist when K binary skills are being assessed, and the K components of the attribute profile are modeled as independent given the latent state. These latent states essentially serve as higher-order latent variables. Some states have higher prevalence for each component and learning is tracked less specifically by considering membership in the latent states rather than charting the change in the attribute profile itself. In this way fewer transition probabilities are needed to model transitions, but membership in the different latent states has a bearing on changing posterior probabilities for the full attribute pattern as it changes over time. Studer (2012) also considers the case in which there are 2^K latent states in the transition model, one for each attribute pattern, and refers to this as Knowledge Tracing plus CDM, though no specific formulation or examples are provided. Recently Wang et al. (2017) considered a full transition model for the entire attribute pattern to address the individualized transition patterns by using a set of latent and observed covariates, such as a general learning ability and practice effects, and applied it to a study of learning spatial reasoning skills in which an intervention was administered between short blocks of items. Recent learning research in CDMs has also considered sequential statistical methods for detection of learning (Ye et al., 2016), in which methods for detection of learning with minimal delay were introduced. More accurate models for learning could enhance detection, enabling one to more efficiently navigate through a list of skills to be learned while avoiding false detections.

In the data mining field, the method of Knowledge Tracing (Corbett and Anderson, 1994) has emerged as a popular technique for modeling learning, usually in the setting of intelligent tutoring systems. Knowledge Tracing bears much resemblance to learning models for CDM, but traditionally focuses on one attribute at a time. In fact, Studer (2012) shows

that Knowledge Tracing is mathematically equivalent to an extension of the NIDA (Junker and Sijtsma, 2001; Maris, 1999) model for multiple time points, with the restriction that each item depends on a single skill. However, recent extensions to Knowledge Tracing have been introduced that incorporate multiple skills at once and allow for different item parameters (Xu and Mostow, 2012; González-Brenes and Mostow, 2013; González-Brenes et al., 2014). As the body of research on dynamic latent class modeling grows, the parallel tracks of CDM and Knowledge Tracing can be expected to produce ideas that can benefit both methodologies at once.

The purpose of this chapter is to propose a class of dynamic models for tracing changes in attributes, which we refer to as learning trajectories. We establish a Bayesian Modeling framework to recover learning trajectory. We evaluate the proposed model through a Monte Carlo simulation and a real data application. In practical settings researchers may have a pool of items to administer over time to evaluate learning trajectories. There is no prior research on how to design such tests to ensure accurate model parameter recovery. Accordingly, our Monte Carlo simulation includes conditions to evaluate different test designs for studies of learning trajectories and the results can provide some guidelines on how to design an efficient test to guarantee the accuracy of model parameters and an estimated learning transition matrix. The real data application demonstrates how to use the proposed model to capture students' learning trajectories in a spatial rotation experiment with learning interventions. The estimated transition matrix has implications on how students learn those skills and this information can be used to refine learning interventions to improve the speed at which skills are mastered.

This chapter is organized as follows. Section 4.3 presents a Bayesian model formulation for estimating CDMs. Section 4.4 reports results from a Monte Carlo study that is designed to assess parameter recovery of a first-order hidden Markov model (FOHM). Section 4.5 presents an application of a FOHM model to a spatial rotation experimental intervention. The last section provides concluding remarks.

4.2 Learning Trajectories

This section considers the circumstance where the latent attributes may change over time. Let α_{ikt} be one if individual i $(i = 1, \ldots, N)$ possesses attribute k $(k = 1, 1 \ldots, K)$ at time t $(t = 1, \ldots, T)$. Define a trajectory of attribute profiles for individual i as $\boldsymbol{\alpha}_i = (\boldsymbol{\alpha}_{i1}, \ldots, \boldsymbol{\alpha}_{iT})'$ where $\boldsymbol{\alpha}_{it} = (\alpha_{i1t}, \ldots, \alpha_{iKt})'$ represents student i's attribute profile at time t. Let $j = 1, \ldots, J_t$ index the number of items administered at time t and define the total number of items as $J = \sum_{t=1}^{T} J_t$. Let $\boldsymbol{Q}_t$ be the $J_t \times K$ $\boldsymbol{Q}$ matrix at time t with zeros and ones that denote the skills needed to correctly answer the J_t items at time t.

Let the total set of attribute vectors be $\mathcal{A} = \left\{\boldsymbol{\alpha} : \boldsymbol{\alpha} \in \{0,1\}^K\right\}$ so that $\boldsymbol{\alpha}_c \in \mathcal{A}$ denotes one of the possible 2^K attribute vectors. Define the set of attribute profiles with k mastered skills as $\mathcal{A}_k = \left\{\boldsymbol{\alpha} : \boldsymbol{\alpha} \in \{0,1\}^K, \|\boldsymbol{\alpha}\|_1 = k\right\}$ and note that $\mathcal{A} = \cup_{k=1}^{K} \mathcal{A}_k$. Recall that the cardinality of $\mathcal{A}_k$ is the number of attribute profiles in $\mathcal{A}_k$ and is denoted by $|\mathcal{A}_k| = \binom{K}{k}$ and the total number of attribute profiles is $|\mathcal{A}| = \sum_{k=0}^{K} \binom{K}{k} = 2^K$.

4.2.1 Unrestricted Learning Trajectories

Definition 1: *In the absence of restrictions on the discrete latent class space the set of learning trajectories at time t is recursively defined as,*

$$\mathcal{A}^{(t)} = \mathcal{A}^{t-1} \times \mathcal{A} = \mathcal{A}^t. \tag{4.1}$$

Note the parentheses in the superscript are dropped and $\mathcal{A}^t$ is used to denote the set of learning trajectories at time t. The definition implies the set of attribute trajectories for time $t = 1$ is $\mathcal{A}^{(1)} = \mathcal{A} = \mathcal{A}^1$. Similarly, the set of attribute trajectory classes for time $t = 2$ is $\mathcal{A}^{(2)} = \mathcal{A}^1 \times \mathcal{A} = \mathcal{A}^2$. That is, students can theoretically switch from the initial states in $\mathcal{A}^1$ to any attribute class in $\mathcal{A}$ after receiving feedback and/or instruction. Rules concerning Cartesian products imply the cardinality of $\mathcal{A}^T$ is the product of sets is $\left|\mathcal{A}^T\right| = 2^{KT}$; i.e.,

learning trajectories grow exponentially with time and the number of attributes.

Example 1: *Consider $\mathcal{A}^T$ for $K = 2$ and $T = 2$.* If $K = T = 2$, $\mathcal{A}^1 = \{00, 10, 01, 11\}$ and $\mathcal{A}^2$ includes 16 learning trajectories:

$$\mathcal{A}^2 = \mathcal{A}^1 \times \mathcal{A} = \{\, (00,00)\,, (00,10)\,, (00,01)\,, (00,11)\,, (10,00)\,, (10,10)\,, (10,01)\,, (10,11)\,,$$
$$(01,00)\,, (01,10)\,, (01,01)\,, (01,11)\,, (11,00)\,, (11,10)\,, (11,01)\,, (11,11)\,\}\,.$$

□

4.2.2 Non-Decreasing Learning Trajectories

Clearly, the discrete latent space grows exponentially as t increases and the set $\mathcal{A}^t$ can be restricted to study a subset of the 2^{KT} learning trajectories. For instance, in education a reasonable assumption is that students acquire skills over time in a non-decreasing fashion. Stated differently, learning can be assumed to be an absorbing state where mastered attributes cannot be unlearned. This subsection compares the set of non-decreasing learning trajectories with $\mathcal{A}^t$.

Definition 2: *Non-decreasing learning and skill acquisition trajectories are defined as,*

$$\mathcal{A}^t_+ = \mathcal{A}^t \cap \{\boldsymbol{\alpha} : \boldsymbol{\alpha}_t \succcurlyeq \boldsymbol{\alpha}_{t-1},\ \forall\, t > 1\} \tag{4.2}$$

That is, $\mathcal{A}^t_+$ is the set of learning trajectories that are non-decreasing. It is straightforward to find the cardinality of $\mathcal{A}^t_+$ by counting the number of ways of picking the change point (i.e., the point where α_{ikt} changes from zero to one). For a given attribute, there are T possibilities for the change point if learning occurs and one possibility if the attribute remains a zero over time for a total of $T+1$ outcomes. Consequently, the number of non-decreasing change points for K attributes is $\left|\mathcal{A}^T_+\right| = (T+1)^K$.

Example 2: *Revisiting Example 1 with $K = 2$ and $T = 2$.* For $K = 2$ and $T = 2$ there are a total of nine non-decreasing learning trajectories within $\mathcal{A}_+^2$. The set of non-decreasing learning trajectories, $\mathcal{A}_+^2$ includes the following elements,

$$\mathcal{A}_+^t = \left\{(00,00),(00,10),(00,01),(00,11),(10,10),(10,11),(01,01),(01,11),(11,11)\right\}$$

□

Table 4.1: Cardinality of Unrestricted and Non-Decreasing Learning Trajectories for Values of K and T

	$\left\lvert\mathcal{A}_+^T\right\rvert = (T+1)^K$				$\left\lvert\mathcal{A}^T\right\rvert = 2^{KT}$			
T	$K=1$	$K=2$	$K=3$	$K=4$	$K=1$	$K=2$	$K=3$	$K=4$
1	2	4	8	16	2	4	8	16
2	3	9	27	81	4	16	64	256
3	4	16	64	256	8	64	512	4096
4	5	25	125	625	16	256	4096	65536
5	6	36	216	1296	32	1024	32768	1048576
6	7	49	343	2401	64	4096	262144	16777216

Any models of student learning trajectories necessarily must propose a manner in which students transition between attribute classes over time. One limitation for practice is that the number of learning trajectories grows exponentially with t, which implies that larger datasets would be needed to accurately model the share of the population in any given learning trajectory. The learning assumption significantly reduces the number of possible learning trajectories. For purpose of demonstration Table 4.1 reports the cardinality of $\mathcal{A}_+^T$ and $\mathcal{A}^T$ for different values of T and K. The values in Table 4.1 show that even for the case where $K = 2$ and $T = 6$ that there are 4096 unrestricted learning trajectories, which could pose challenges in real applications. In contrast, the learning assumption reduces the number of learning trajectories by a factor of nearly 100 from 4096 to 49. An important observation from Table 4.1 is that the number of unrestricted learning trajectories grows quickly even for a modest $K = 4$ with a total of 2^{24} possible trajectories. Although there are relatively fewer non-decreasing learning trajectories for $K = 4$ and $T = 6$ researchers

would need substantial data to estimate probabilities of membership in one of the possible 2401 classes.

4.3 Modeling Learning Trajectories

This section discusses strategies for modeling unrestricted and non-decreasing learning trajectories. Let $\boldsymbol{\alpha}_l' = (\boldsymbol{\alpha}_{l1}, \ldots, \boldsymbol{\alpha}_{lT})$ be a learning trajectory such that $\boldsymbol{\alpha}_l \in \mathcal{A}^T$. Let Y_{ijt} denote the random variable for person i to item j at time t and let y_{ijt} represent the observed value. Let $\boldsymbol{y}_i' = (\boldsymbol{y}_{i1}, \ldots, \boldsymbol{y}_{iT})$ denote vectors of observations over time of the random variables $\boldsymbol{Y}_i' = (\boldsymbol{Y}_{i1}, \ldots, \boldsymbol{Y}_{iT})$ with $\boldsymbol{y}_{it}' = (y_{i1t}, \ldots, y_{iJ_tt})$ and $\boldsymbol{Y}_{it}' = (Y_{i1t}, \ldots, Y_{iJ_tt})$. Let the collection of item parameters over time be denoted by $\boldsymbol{\zeta}' = (\boldsymbol{\zeta}_1, \ldots \boldsymbol{\zeta}_T)$ with $\boldsymbol{\zeta}_t' = (\boldsymbol{\zeta}_{1t}, \ldots, \boldsymbol{\zeta}_{J_tt})$ and $\boldsymbol{\zeta}_{jt}$ denotes the item parameters for item j at time t.

The probability of a correct response on item j at time t for an individual with learning trajectory l is $P\left(Y_{ijt} = 1 | \boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{lt}, \boldsymbol{\zeta}_{jt}\right)$. Assuming that responses are independent given $\boldsymbol{\alpha}_i$, the probability of observing subject i's responses to items $j = 1, \ldots, J_t$ at time t conditioned upon $\boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{lt}$ and the item parameters $\boldsymbol{\zeta}_t$ is,

$$p\left(\boldsymbol{Y}_{it} | \boldsymbol{\alpha}_{lt}, \boldsymbol{\zeta}_t\right) = \prod_{j=1}^{J_t} \left[P\left(Y_{ijt} = 1 | \boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{lt}, \boldsymbol{\zeta}_{jt}\right)\right]^{y_{ijt}} \left[1 - P\left(Y_{ijt} = 1 | \boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{lt}, \boldsymbol{\zeta}_{jt}\right)\right]^{1-y_{ijt}}. \tag{4.3}$$

Assuming item responses are independent over time given the learning trajectory of attribute profiles and item parameters implies that the likelihood for individual i is,

$$p\left(\boldsymbol{Y}_i | \boldsymbol{\alpha}_l, \boldsymbol{\zeta}\right) = \prod_{t=1}^{T} p\left(\boldsymbol{Y}_{it} | \boldsymbol{\alpha}_{lt}, \boldsymbol{\zeta}_t\right). \tag{4.4}$$

The posterior probability that $\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l$ given the observed responses and item parameters is proportional to the product of $p\left(\boldsymbol{Y}_i | \boldsymbol{\alpha}_l, \boldsymbol{\zeta}\right)$ and $p\left(\boldsymbol{\alpha}_l\right)$ where $p\left(\boldsymbol{\alpha}_l\right)$ is the prior probability that individual i has learning trajectory l. Certainly, there are many choices for $p\left(\boldsymbol{\alpha}_l\right)$. The

following discussion describes several models for $\boldsymbol{\alpha}_i$ that differ in terms of complexity and the number of parameters.

4.3.1 Unstructured Trajectories

Consider the case where individuals can freely move through attribute classes over time (i.e., losing an attribute is possible). In this case, as noted above there are 2^{KT} possible learning trajectories $\boldsymbol{\alpha}_l$. Let $\pi_l = P\left(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l \in \mathcal{A}^T\right)$ denote the structural probability of having learning trajectory l characterized by starting at $\boldsymbol{\alpha}_{l1}$, proceeding through $\boldsymbol{\alpha}_{lt}$ for $1 < t < T$, and terminating at $\boldsymbol{\alpha}_{lT}$. Let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{2^{KT}})'$ be the 2^{KT} vector of unstructured learning trajectory profiles, so the categorical prior for $\boldsymbol{\alpha}_i$ is,

$$p\left(\boldsymbol{\alpha}_i \,|\boldsymbol{\pi}\right) = \prod_{l=1}^{2^{KT}} \pi_l^{\mathcal{I}(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l)}. \tag{4.5}$$

The posterior probability that individual i's followed learning trajectory $\boldsymbol{\alpha}_l$ conditioned upon the data, item parameters, and structural parameters in the unstructured case is,

$$p\left(\boldsymbol{\alpha}_i|\boldsymbol{Y}_i, \boldsymbol{\zeta}, \boldsymbol{\pi}\right) \propto p\left(\boldsymbol{Y}_i|\boldsymbol{\alpha}_l, \boldsymbol{\zeta}\right) \left(\prod_{l=1}^{2^{KT}} \pi_l^{\mathcal{I}(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l)}\right). \tag{4.6}$$

DINA Bayesian Model Formulation

The prior for the general unstructured model for learning trajectories can be included as a prior for any CDM Bayesian formulation. Consider the case of the DINA model for the item response function, $P\left(Y_{ijt} = 1|\boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{lt}, \boldsymbol{\zeta}_{jt}\right)$. Also, let item parameters be invariant over time (i.e., $s_{jt} = s_j$ and $g_{jt} = g_j$), and suppose $\boldsymbol{Q}$ is known. A Bayesian model for estimating

the general unstructured model for the DINA CDM is,

$$Y_{ijt} \left| \boldsymbol{\alpha}_{it}, s_j, g_j \right. \sim \text{ Bernoulli} \left(\left(1 - s_j\right)^{\eta_{ijt}} g_j^{1-\eta_{ijt}} \right), \ \eta_{ijt} = \prod_{k=1}^{K} \alpha_{ikt}^{q_{jk}} \tag{4.7}$$

$$p\left(\boldsymbol{\alpha}_i | \boldsymbol{\pi}\right) \propto \prod_{l=1}^{2^{KT}} \pi_l^{\mathcal{I}(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l)}, \ 0 < \pi_l < 1, \ \sum_{l=1}^{2^{KT}} \pi_l = 1 \tag{4.8}$$

$$\boldsymbol{\pi} \sim \text{Dirichlet}\left(\boldsymbol{\delta}_0\right), \ \boldsymbol{\delta}_0 = \left(\delta_{01}, \ldots, \delta_{0,2^{KT}}\right) \tag{4.9}$$

$$p\left(s_j, g_j\right) \propto s_j^{\alpha_s - 1} \left(1 - s_j\right)^{\beta_s - 1} g_j^{\alpha_g - 1} \left(1 - g_j\right)^{\beta_g - 1} \mathcal{I}\left(0 \leq g_j < 1 - s_j \leq 1\right) \tag{4.10}$$

Equation ((4.9)) is a conjugate Dirichlet prior for the unrestricted learning trajectory probabilities.

4.3.2 Non-Decreasing Learning Trajectories

As noted above in Table 4.1, using the prior in (4.6) would require substantial observations for even modest T and K to accurately recover parameters. One way to reduce the computational burden in educational contexts is to assume learning trajectories are non-decreasing. As noted above, there are $(T+1)^K$ possible non-decreasing attribute trajectories. One strategy is to specify the prior for $\boldsymbol{\alpha}_i$ as categorical over the $(T+1)^K$ non-decreasing trajectories,

$$p\left(\boldsymbol{\alpha}_i \left| \boldsymbol{\pi}_+ \right.\right) = \prod_{l=1}^{(T+1)^K} \pi_{+l}^{\mathcal{I}(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l)} \propto \mathcal{I}\left(\mathbf{0}_K \preceq \boldsymbol{\alpha}_{i1} \preceq \boldsymbol{\alpha}_{i2} \preceq \cdots \preceq \boldsymbol{\alpha}_{iT} \preceq \mathbf{1}_K\right) \prod_{l=1}^{2^{KT}} \pi_l^{\mathcal{I}(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l)} \tag{4.11}$$

where $\pi_{+l} = P\left(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l \in \mathcal{A}_+^T\right)$ is the probability of being in the non-decreasing attribute profile l and $\boldsymbol{\alpha}_{i,t-1} \preceq \boldsymbol{\alpha}_{it}$ if $\alpha_{ik,t-1} \leq \alpha_{ikt}$ for $k = 1, \ldots, K$. The posterior probability that individual i follows non-decreasing attribute trajectory $\boldsymbol{\alpha}_l$ is,

$$p\left(\boldsymbol{\alpha}_i | \boldsymbol{Y}_i, \boldsymbol{\zeta}, \boldsymbol{\pi}_+\right) \propto p\left(\boldsymbol{Y}_i | \boldsymbol{\alpha}_l, \boldsymbol{\zeta}\right) \left(\prod_{l=1}^{(T+1)^K} \pi_{+l}^{\mathcal{I}(\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_l)} \right) \tag{4.12}$$

DINA Bayesian Model Formulation

Consider the DINA model IRF for the non-decreasing attribute trajectories with parameters for item j invariant over time (i.e., $s_{jt} = s_j$ and $g_{jt} = g_j$) and $\boldsymbol{Q}$ is known. A Bayesian model for estimating the non-decreasing model is obtained by updating the prior for $\boldsymbol{\alpha}_i$ by replacing (4.6) with (4.11) as,

$$p\left(\boldsymbol{\alpha}_i|\boldsymbol{\pi}\right) \propto \prod_{l=1}^{\left|\mathcal{A}_+^T\right|} \pi_{+l}^{\mathcal{I}(\boldsymbol{\alpha}_i=\boldsymbol{\alpha}_l)},\ 0 < \pi_{+l} < 1,\ \sum_{l=1}^{\left|\mathcal{A}_+^T\right|} \pi_{+l} = 1 \tag{4.13}$$

$$\boldsymbol{\pi} \sim \text{Dirichlet}\left(\boldsymbol{\delta}_0\right),\ \boldsymbol{\delta}_0 = \left(\delta_{01}, \ldots, \delta_{0,\left|\mathcal{A}_+^T\right|}\right) \tag{4.14}$$

4.3.3 First-Order Hidden Markov Model

Table 4.1 shows that the cardinality for the set of unrestricted and non-decreasing learning trajectory sets may be too large for datasets observed in typical educational research studies. An alternative is to use a more parsimonious approximation to larger sets of the unrestricted and non-decreasing learning trajectories.

One approximation as employed in prior research (e.g., see Kaya and Leite, 2017, for an example with $t = 2$) is to consider a hidden Markov model with first-order transition probabilities, $\boldsymbol{\Omega}$. That is, we can let $\boldsymbol{\pi}_1$ denote initial class membership probabilities at time $t = 1$ and let $\boldsymbol{\Omega}$ be a $2^K \times 2^K$ matrix of first-order transition probabilities between classes. Specifically, the elements of $\boldsymbol{\Omega}$ denote the chance of transitioning from class c to c' between any two time periods. Let elements of $\boldsymbol{\Omega}$ be denoted as $\omega_{c'|c} = p\left(\boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{c'}|\boldsymbol{\alpha}_{i,t-1} = \boldsymbol{\alpha}_c\right)$ for all t. The prior for $\boldsymbol{\alpha}_i$ under an unrestricted first-order hidden Markov (FOHM) model for

$t \geq 2$ is,

$$p\left(\boldsymbol{\alpha}_i \left| \boldsymbol{\pi}_1, \boldsymbol{\Omega}\right.\right) = p\left(\boldsymbol{\alpha}_{i1} \left| \boldsymbol{\pi}_1\right.\right) \prod_{t=2}^{T} p\left(\boldsymbol{\alpha}_{it} \left| \boldsymbol{\alpha}_{i,t-1}, \boldsymbol{\Omega}\right.\right)$$
$$= \left(\prod_{\boldsymbol{\alpha}_c \in \mathcal{A}} \pi_{1c}^{\mathcal{I}(\boldsymbol{\alpha}_{i1} = \boldsymbol{\alpha}_c)} \right) \prod_{t=2}^{T} \left(\prod_{\boldsymbol{\alpha}_{c'} \in \mathcal{A}} \omega_{c'|c}^{\mathcal{I}\left(\boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{c'}\right)} \right)^{\mathcal{I}\left(\boldsymbol{\alpha}_{i,t-1} = \boldsymbol{\alpha}_c\right)}. \tag{4.15}$$

The unrestricted FOHM requires significantly fewer parameters to be estimated than both $\mathcal{A}^T$ and $\mathcal{A}_+^T$. In fact, there are $2^K + 4^K$ total parameters (i.e., 2^K elements in $\boldsymbol{\pi}_1$ and 4^K elements in $\boldsymbol{\Omega}$). A clear advantage of the FOHM is that the model imposes some structure to reduce the parameters to a set that can be more easily recovered in applied educational settings. Furthermore, the number of parameters for the FOHM does not grow with T. However, it should be emphasized that the FOHM is an approximation and may not capture the true underlying nuance found in the unstructured and non-decreasing learning trajectories $\mathcal{A}^T$ and $\mathcal{A}_+^T$.

The full conditional distribution for $\boldsymbol{\alpha}_{it}$ depends upon t for the FOHM. For attributes profiles at time t such that $1 < t < T$, $\boldsymbol{\alpha}_{it}$ is conditioned upon the adjacent attributes $\boldsymbol{\alpha}_{i,t-1}$ and $\boldsymbol{\alpha}_{i,t+1}$. Accordingly, the full conditional probability that $\boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_{lt}$ given that $\boldsymbol{\alpha}_{i,t-1} = \boldsymbol{\alpha}_{c,t-1}$ and $\boldsymbol{\alpha}_{i,t+1} = \boldsymbol{\alpha}_{c',t+1}$ is:

$$p\left(\boldsymbol{\alpha}_{lt} | \boldsymbol{Y}_i, \boldsymbol{\zeta}, \boldsymbol{\Omega}, \boldsymbol{\alpha}_{c,t-1}, \boldsymbol{\alpha}_{c',t+1}\right) \propto p\left(\boldsymbol{Y}_i | \boldsymbol{\alpha}_l, \boldsymbol{\zeta}\right) p\left(\boldsymbol{\alpha}_{lt} | \boldsymbol{\alpha}_{c,t-1}, \boldsymbol{\Omega}\right) p\left(\boldsymbol{\alpha}_{c',t+1} | \boldsymbol{\alpha}_{lt}, \boldsymbol{\Omega}\right)$$
$$\propto p\left(\boldsymbol{Y}_{it} | \boldsymbol{\alpha}_{lt}, \boldsymbol{\zeta}_t\right) \omega_{l|c}\, \omega_{c'|l} \tag{4.16}$$

where $\omega_{l|c} = p\left(\boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_l | \boldsymbol{\alpha}_{i,t-1} = \boldsymbol{\alpha}_c\right)$ and $\omega_{c'|l} = p\left(\boldsymbol{\alpha}_{i,t+1} = \boldsymbol{\alpha}_{c'} | \boldsymbol{\alpha}_{it} = \boldsymbol{\alpha}_l\right)$ are first-order transition probabilities. At time $t = 1$ the conditional probability that $\boldsymbol{\alpha}_{i1} = \boldsymbol{\alpha}_{l1}$ given the item responses and parameters and $\boldsymbol{\alpha}_{i2} = \boldsymbol{\alpha}_{c2}$ is,

$$p\left(\boldsymbol{\alpha}_{l1} | \boldsymbol{Y}_i, \boldsymbol{\zeta}, \boldsymbol{\Omega}, \boldsymbol{\alpha}_{c2}\right) \propto p\left(\boldsymbol{Y}_{i1} | \boldsymbol{\alpha}_{l1}, \boldsymbol{\zeta}_1\right) \pi_l\, \omega_{c|l}, \tag{4.17}$$

where $\pi_l = p(\boldsymbol{\alpha}_{i1} = \boldsymbol{\alpha}_{l1})$ is the baseline prior probability. Similarly, for time $t = T$ the conditional probability that $\boldsymbol{\alpha}_{iT} = \boldsymbol{\alpha}_{lT}$ given $\boldsymbol{Y}_i$, $\boldsymbol{\zeta}$, and $\boldsymbol{\alpha}_{i,T-1} = \boldsymbol{\alpha}_{c,T-1}$ is,

$$p\left(\boldsymbol{\alpha}_{lT}|\boldsymbol{Y}_i, \boldsymbol{\zeta}, \boldsymbol{\Omega}, \boldsymbol{\alpha}_{c,T-1}\right) \propto p\left(\boldsymbol{Y}_{iT}|\boldsymbol{\alpha}_{lT}, \boldsymbol{\zeta}_T\right) \omega_{l|c}. \tag{4.18}$$

Given the priors $p(\boldsymbol{\omega}_c)$ and $p(\boldsymbol{\pi}_1)$, the full conditional distribution for $\boldsymbol{\Omega}$ and $\boldsymbol{\pi}$ can be derived as:

$$\begin{aligned} p\left(\boldsymbol{\Omega}, \boldsymbol{\pi}_1 \mid \boldsymbol{\alpha}, \boldsymbol{\zeta}\right) &\propto \prod_{l=1}^{2^K} \prod_{c=1}^{2^K} \left(p\left(\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_c \mid \boldsymbol{\alpha}_{t-1} = \boldsymbol{\alpha}_l, \boldsymbol{\omega}_l\right) p(\omega_{c|l})\right)^{\mathcal{I}(t>1)} \left(\pi_{1,l} p(\pi_{i,l})\right)^{\mathcal{I}(t=1)} \\ &\propto \prod_{l=1}^{2^K} \prod_{c=1}^{2^K} \omega_{c|l}^{\tilde{N}_{c|l}} \pi_{1,l}^{\tilde{N}_{0,l}} p\left(\boldsymbol{\omega}_l\right) p\left(\boldsymbol{\pi}_1\right) \\ &\propto \prod_{l=1}^{2^K} \prod_{c=1}^{2^K} \omega_{c|l}^{\tilde{N}_{c|l}+\delta_{l,c}} \pi_{1,l}^{\tilde{N}_{0,l}+\delta_{0,l}} \end{aligned} \tag{4.19}$$

where $\tilde{N}_{c|l}$ is the number of subjects that have attribute profile $\boldsymbol{\alpha}_l$ at time t and change into $\boldsymbol{\alpha}_c$ at time $t+1$, $t = 1, \ldots, T-1$, and $\tilde{N}_{0,l}$ is the number of subjects that initially have attribute profile $\boldsymbol{\alpha}_l$. Let Dirichlet distributions be the conjugate priors for $\boldsymbol{\pi}_1$ and each $\boldsymbol{\omega}_c, c = 1, \ldots, 2^K$, i.e.,

$$\boldsymbol{\pi}_1 = (\pi_{1,1}, \ldots, \pi_{1,2^K}) \sim \text{Dirichlet}(\boldsymbol{\delta}_0),\ \boldsymbol{\delta}_0 = \left(\delta_{0,1}, \ldots, \delta_{0,2^K}\right) \tag{4.20}$$

$$\boldsymbol{\omega}_c = (\omega_{1|c}, \ldots, \omega_{2^K|c}) \sim \text{Dirichlet}(\boldsymbol{\delta}_c),\ \boldsymbol{\delta}_c = \left(\delta_{1|c}, \ldots, \delta_{2^K|c}\right) \tag{4.21}$$

then as shown in (4.19) the posterior for $\boldsymbol{\pi}_1$ will be Dirichlet$(\boldsymbol{\delta}_0 + \tilde{\boldsymbol{N}}_0)$, where $\tilde{\boldsymbol{N}}_0 = (\tilde{N}_{0,1}, \ldots, \tilde{N}_{0,2^K})$, and posteriors for $\boldsymbol{\omega}_c$ will be Dirichlet$(\boldsymbol{\delta}_c + \tilde{\boldsymbol{N}}_c)$, where $\tilde{\boldsymbol{N}}_c = (\tilde{N}_{1|c}, \ldots, \tilde{N}_{2^K|c})$.

4.3.4 First-Order Hidden Markov Model, Non-Decreasing Attributes

The previous subsection introduced the unrestricted FOHM and there may be substantive areas where it is reasonable to assume attributes are non-decreasing over time. A non-decreasing FOHM can also be formulated that restricts transitions to non-decreasing states. Specifically, let $\boldsymbol{\Omega}_+$ include zeros for elements of $\boldsymbol{\Omega}$ that correspond with losing an attribute. In this case the prior for $\boldsymbol{\alpha}_i$ becomes,

$$p\left(\boldsymbol{\alpha}_i \left| \boldsymbol{\pi}_1, \boldsymbol{\Omega}_+ \right.\right) = p\left(\boldsymbol{\alpha}_{i1} \left| \boldsymbol{\pi}_1 \right.\right) \prod_{t=2}^{T} p\left(\boldsymbol{\alpha}_{it} \left| \boldsymbol{\alpha}_{i,t-1}, \boldsymbol{\Omega}_+ \right.\right)$$

$$\propto \prod_{\boldsymbol{\alpha}_c \in \mathcal{A}} \pi_{1c}^{\mathcal{I}(\boldsymbol{\alpha}_{i1}=\boldsymbol{\alpha}_c)} \prod_{t=2}^{T} \left(\prod_{\alpha_{c'} \succeq \boldsymbol{\alpha}_{i,t-1}} \omega_{c'|c}^{\mathcal{I}\left(\boldsymbol{\alpha}_{it}=\boldsymbol{\alpha}_{c'}\right)} \right)^{\mathcal{I}\left(\boldsymbol{\alpha}_{i,t-1}=\boldsymbol{\alpha}_c\right)} . \tag{4.22}$$

The non-decreasing FOHM includes fewer parameters to estimate than the unrestricted FOHM. In cases where attributes cannot be unlearned it is possible to show that $\boldsymbol{\Omega}_+$ includes a total of 3^K transition probabilities. The reduction in parameters between the unrestricted and non-decreasing FOHMs is nontrivial. For $K = 1, 2, 3, 4$, $\boldsymbol{\Omega}$ for the unrestricted FOHM includes 4^1, 4^2, 4^3, and 4^4 elements, respectively, in comparison to 3^1, 3^2, 3^3, and 3^4 for $\boldsymbol{\Omega}_+$ of the non-decreasing FOHM.

The full conditional distributions for $\boldsymbol{\alpha}_{it}$ under the non-decreasing FOHM have the same form as (4.16) through (4.18) for the unrestricted FOHM. The only modification is to recognize that some of the transition probabilities are zero.

4.3.5 Higher-Order First-Order Hidden Markov Model

A parsimonious alternative is to model transition probabilities for each attribute and introduce dependence in transition by conditioning on a higher-order factor, θ. Another version of a higher-order FOHM specifies a prior for each α_{ikt} conditional on a higher-order learning

trait θ for the probability of transitioning and retaining an attribute,

$$\omega_{1|0,ik} = p\left(\alpha_{ikt} = 1|\alpha_{ik,t-1} = 0, \theta_i, \boldsymbol{\gamma}_0\right) \tag{4.23}$$

$$\omega_{1|1,ik} = p\left(\alpha_{ikt} = 1|\alpha_{ik,t-1} = 1, \theta_i, \boldsymbol{\gamma}_1\right) \tag{4.24}$$

where $\boldsymbol{\gamma}_0$ and $\boldsymbol{\gamma}_1$ are possibly vectors of parameters used to model the transition probabilities as a function of θ. It is important to recognize the role of θ in the higher-order FOHM. The higher-order factor captures individual differences in learning rates and is included to model dependence among transitions in attributes over time.

One option for modeling the transition probabilities for attribute k is to, for example, set $\omega_{1|0,ik} = \Psi\left(\gamma_{00k} + \gamma_{01k}\theta_i\right)$ where $\Psi\left(\cdot\right)$ is a cumulative distribution function. In fact, Wang et al. (2017) provided an example of setting Ψ as a logistic function.

The distribution of α_{ikt} given $\alpha_{ik,t-1}$, θ and the transition model parameters, is,

$$p\left(\alpha_{ikt}|\alpha_{ik,t-1}, \theta_i, \boldsymbol{\gamma}_0, \boldsymbol{\gamma}_1\right) = \left(\omega_{1|1,ik}^{\alpha_{ikt}}\left(1 - \omega_{1|1,ik}\right)^{1-\alpha_{ikt}}\right)^{\alpha_{ik,t-1}}\left(\omega_{1|0,ik}^{\alpha_{ikt}}\left(1 - \omega_{1|0,ik}\right)^{1-\alpha_{ikt}}\right)^{1-\alpha_{ik,t-1}}. \tag{4.25}$$

If attribute transitions are independent given θ, the prior probability for $\boldsymbol{\alpha}_i$ conditioned upon initial class membership $\boldsymbol{\pi}_1$ and transition parameters, $\boldsymbol{\gamma}_0$ and $\boldsymbol{\gamma}_1$, is,

$$p\left(\boldsymbol{\alpha}_i\,|\boldsymbol{\pi}_1, \boldsymbol{\gamma}_0, \boldsymbol{\gamma}_1\right) = \prod_{\boldsymbol{\alpha}_c \in \mathcal{A}} \pi_{1c}^{\mathcal{I}(\boldsymbol{\alpha}_{i1} = \boldsymbol{\alpha}_c)} \int \left(\prod_{t=2}^{T}\prod_{k=1}^{K} p\left(\alpha_{ikt}|\alpha_{ik,t-1}, \theta_i\right)\right) p\left(\theta_i\right) d\theta_i \tag{4.26}$$

where $p\left(\theta_i\right)$ is a prior distribution for θ.

The aforementioned discussion of the higher-order FOHM allows for individuals to lose attributes when transition from on time to the next. The learning assumption can easily be enforced by setting $\omega_{1|1,ik} = 1$ and $\alpha_{ikt} \succeq \alpha_{ik,t-1}$ for all k and t.

4.4 Monte Carlo Simulation Study

4.4.1 Overview

This section reports Monte Carlo simulation studies evaluating the recovery of the transition matrix and item parameters based on the first-order hidden Markov model for non-decreasing learning trajectories. We illustrate our model by considering $K = 4$ attributes, $T = 5$ time points and $J = 50$ items with slipping parameters equal to 0.2 and guessing parameters equal to 0.3. The transition matrix is set to be the same as the estimated transition probabilities from the real data of the next section (see Table 4.6). In this simulated test, 50 items are assigned into 5 blocks with 10 items in each block, and the $\boldsymbol{Q}$ matrix in each block is complete (Chiu et al., 2009) for model identification purposes. Wang et al. (2017) conjectured that the item positions may cause biased item parameter estimation because the students' latent attribute profiles are assumed to change with time while the DINA model parameters are static. In this case the items in the later stage of the test may not receive the sufficient exposure to different latent classes, causing an inaccurate estimation for the corresponding model parameters. In order to investigate whether there is an order effect of the item positions to model parameter estimation, we considered two test designs. The first mimicked the experimental design in Wang et al. (2017) where the blocks of items are counterbalanced allowing each item to be positioned throughout all test stages to guarantee items will be exposed evenly at different time points, and we refer it as a "balanced" design from now on. The second was an unbalanced design where each examinee receives the same order of blocks during the test. For each test design, three levels of sample sizes $N = 500$, 2500, 5000 were considered and for each level 50 independent datasets were generated to assess the performance. We used a Markov chain of 20000 iterations with a burn-in of 10000 iterations and evaluated the convergence of Gibbs samplers using the multivariate potential scale reduction factor (MPSRF) $\hat{R}$ (Brooks and Gelman, 1998) for chain length of 10000. Examinees' initial attribute profiles were simulated from the multivariate normal threshold

model (e.g., see Chiu et al., 2009), and both independent structure and dependent structure with $\rho = 0.5$ of the initial attributes are considered in the simulation. The true distribution of initial attribute profiles is shown in Table 4.2.

Table 4.2: True distribution of initial attribute profiles

$\boldsymbol{\alpha}_c$	0000	0001	0010	0011	0100	0101	0110	0111
$\rho = 0$	0.038	0.010	0.026	0.006	0.058	0.014	0.038	0.010
$\rho = 0.5$	0.122	0.002	0.012	0.001	0.044	0.003	0.013	0.002
$\boldsymbol{\alpha}_c$	1000	1001	1010	1011	1100	1101	1110	1111
$\rho = 0$	0.154	0.038	0.102	0.026	0.230	0.058	0.154	0.038
$\rho = 0.5$	0.178	0.013	0.058	0.012	0.194	0.044	0.179	0.122

4.4.2 Results

Parameter convergence was evaluated using the MPSRF based on five independent Markov chains of length 10000 starting from random initial values. The results suggest the Markov chain converges to the posterior distribution after 4000 iterations, which indicates that a chain-length of 20000 iterations with a burn-in of 10000 is reasonable for this simulation study.

Figures 4.1 and 4.2 show that our method can accurately estimate the slipping and guessing parameters. When the sample size is small, results using the balanced design outperform the results under the unbalanced design, and the difference between the two test designs diminishes as sample size grows. Figure 4.3 presents the deviance of estimated probabilities for initial attribute classes. The bias becomes smaller as the sample size grows, and results from the balanced design tend to have smaller bias than the unbalanced design. Figure 4.4 presents the deviance of estimated transition probabilities. There are 81 transition probabilities and the last one is always 1 because of the non-decreasing constraint. Again, the accuracy of the estimated transition probabilities improves with the increase of the sample size. In summary, our simulation results indicate it's better to use the counterbalanced design for relatively small sample sizes to ensure accurate recovery of the model parameters.

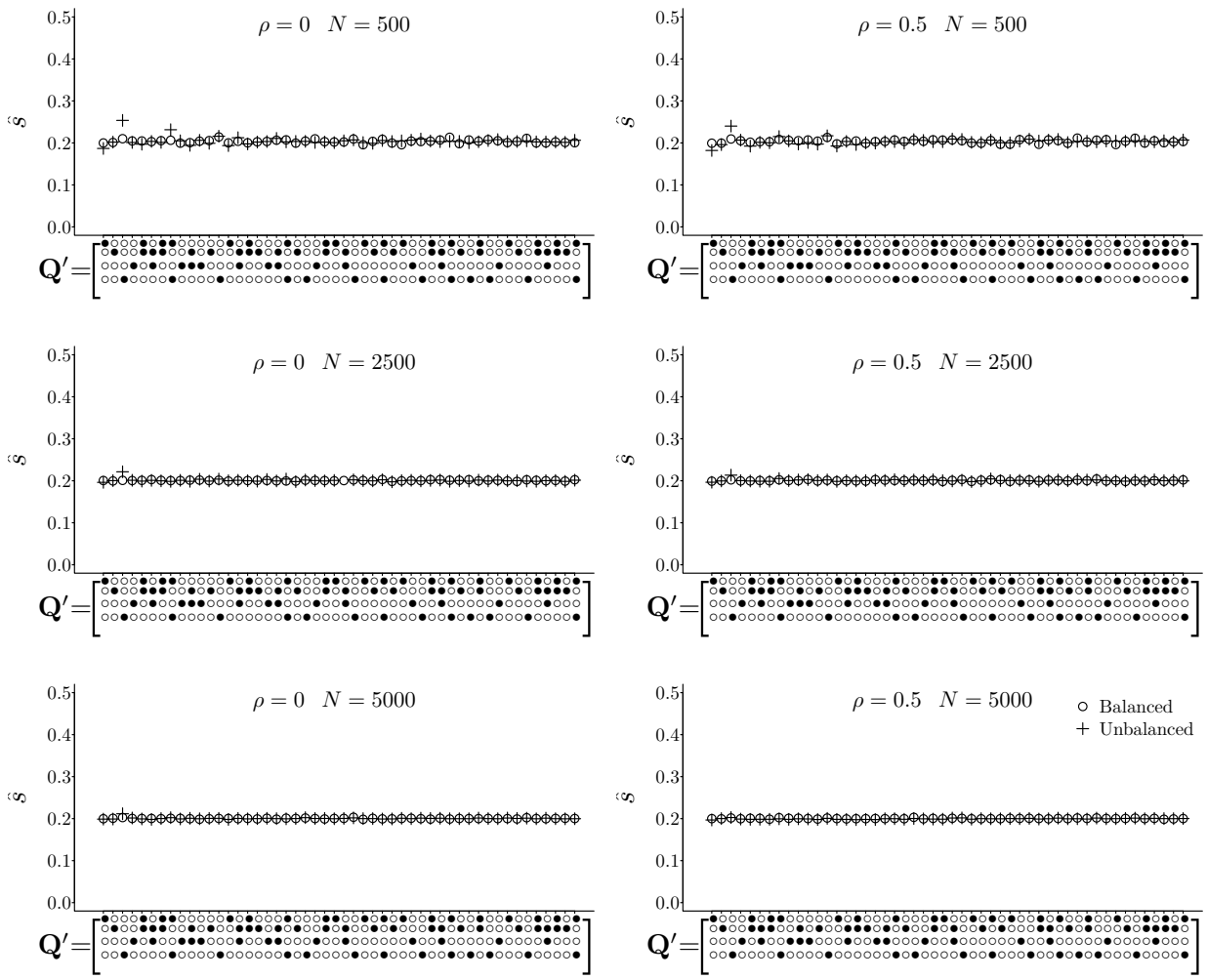


Figure 4.1: Estimated slipping parameters for different settings. Results are based upon 50 replications.

4.5 Application to Spatial Rotation Skill Acquisition

In this section, we apply the first-order hidden Markov Model with non-decreasing attributes assumption to model students' learning of spatial skills. The spatial reasoning questions were developed based on the Revised Purdue Spatial Visualization Test (PSVT-R) (Yoon, 2011). Wang et al. (2017) extended this test to incorporate a training tool which is designed to cause learning of rotation tasks. Specifically, participants first answered the questions in a test block then proceeded to a learning block in which they can receive feedback on

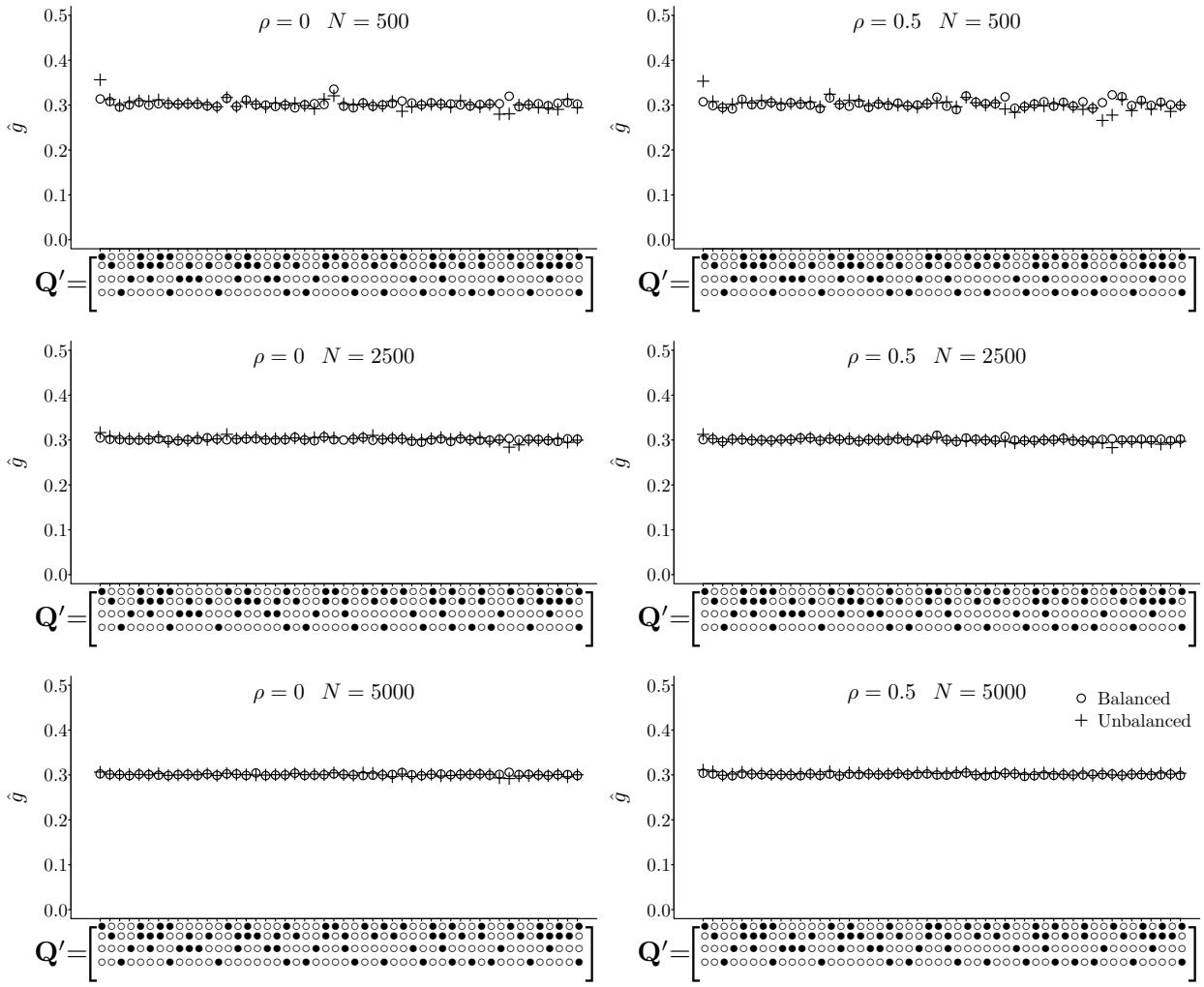


Figure 4.2: Estimated guessing parameters for different settings. Results are based upon 50 replications.

their responses in the previous test block and learn the corresponding spatial rotation tasks through a learning intervention. The whole test consisted of 5 test blocks each containing 10 questions, followed by a learning intervention block. The items in the test block included a rotated object and subjects were presented a new object and must determine which of the 5 options corresponding to the rotated version. Four mental rotation skills were identified 1) 90^0 x-axis, 2) 90^0 y-axis,3) 180^0 x-axis, and 4) 180^0 y-axis. All items included x- and y-axis rotations with objects of varying complexity. The 10 questions were assembled with similar structures but were different for each block. In order to eliminate the order effect on

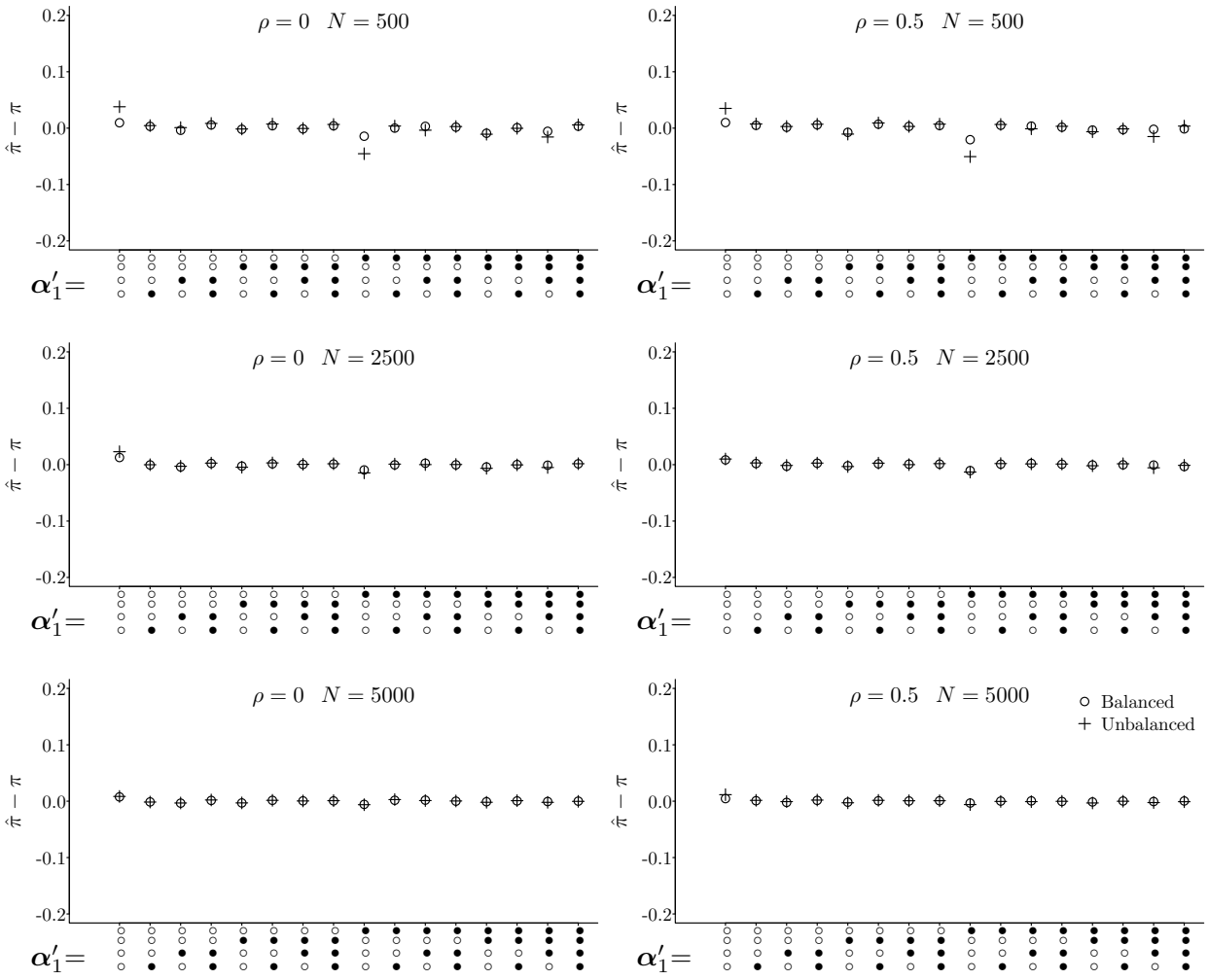


Figure 4.3: Deviance of estimated prior for initial attribute profile class probabilities

the estimation of the item parameters, a counterbalanced test condition was applied. Five versions of test were developed by rotating the five test blocks as the first block in the whole test process. During the experiment, those 5 tests were randomly assigned to participants to guarantee that different test blocks can have an equal chance to be the first block among the participants. The detailed experiment design can be referred to Wang et al. (2017).

Responses from 351 individuals to those 50 questions were collected from this experiment, and the first-order hidden Markov model with non-decreasing attributes assumption and DINA Bayesian Model formulation was applied to estimate students' learning trajecto-

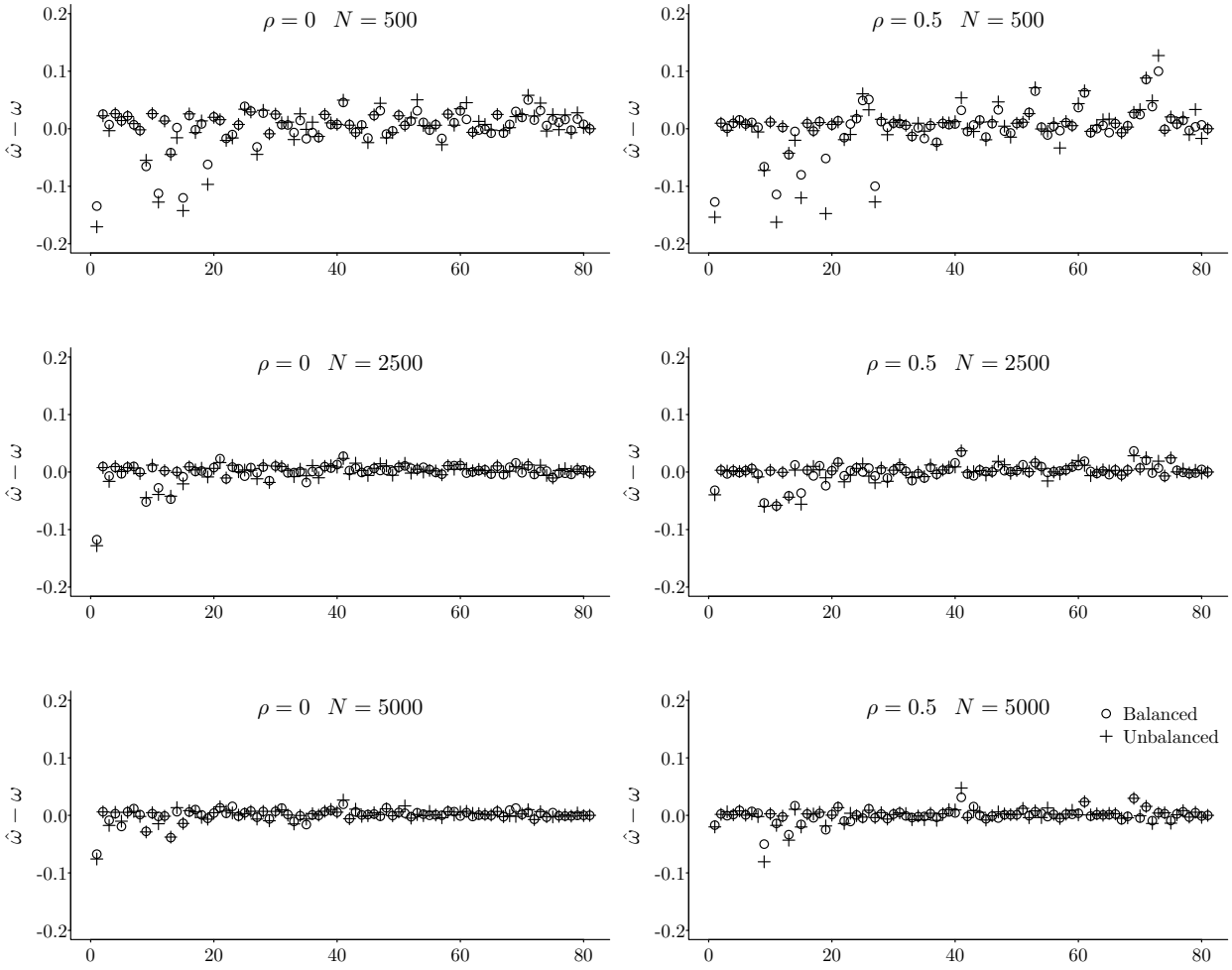


Figure 4.4: Estimated transition probabilities.

ries. The 5 test blocks in the test represented 5 different time points. Based on previous simulation results, we used the chain of 30000 iterations with a burn-in of 15000 to ensure the convergence of the Markov chain.

Table 3 documents the estimates of the DINA item parameters. The results are quite consistent with the estimates from Wang et al. (2017). That is most of the items have relatively large guessing parameters. This result might be due to the simple shapes for some items and the many distractors that can be easily eliminated from consideration. We summarized students' learning trajectories in terms of the estimated mastery rate for each of

the four skills over time, the frequencies of the number of mastered skills at each time point and the transition matrix, which are presented in Table 4, Table 5 and Table 6 respectively. From those results, we can observe an increase of mastery rates for each skill with time and the transition pattern of different latent classes at each time point.

The transition probability matrix explicitly gives the probabilities of remaining in the same stage or learning more skills from time t to $t+1$, which reveals the learning pattern of current mastered skills. The learning process can be inferred by interpreting the entries in the estimated transition matrix. For example, in Table 6, the ninth row of the estimated transition matrix presents the transition probability for current state of attribute profile $(1, 0, 0, 0)$ to another latent class. It indicates that if a student has mastered a single skill on 90° x-axis, he or she tends to master more skills next time with probability around 0.87 and the most likely skill to be mastered next is 90° y-axis or 180° x-axis. On the other hand, the fifth row shows the transition probability for current state of acquisition on the single skill of 90° y-axis. The chance for mastering more skills next time is around 0.67, which is lower compared to mastery of 90° x-axis, and the next mastered skill with this current latent state will most likely to be 180° y-axis. The second and third rows also imply that students who have mastered skills on 180° x-axis or y-axis will most likely master 90° x-axis or y-axis next time. This in fact also indicates certain hierarchical structure among those four latent skills, mastering the 90° might be prerequisite for mastering 180°.

The distribution of the initial latent classes at time point one in the transition matrix are slightly different from those estimated from Wang et al. (2017). This might due to the different assumptions applied to the transition matrix. The first-order hidden Markov model assumes the time-homogeneity of the transition matrix, that is the transition probability of different latent classes are the same at different points of time, and are the same for each individual. However, the transition probabilities in Wang et al. (2017) depend on different covariates, which could make it different for different individuals and at different points of time. In this sense, FOHM with non-decreasing learning trajectories assumption is a reduced

model from Wang et al. (2017), and it's worth investigating the model comparison in the future to select the most appropriate model for a given data set.

Table 4.3: Item parameter estimates for Spatial Rotation Data

	parameters			parameters	
Item	$\hat{s}_j$	$\hat{g}_j$	Item	$\hat{s}_j$	$\hat{g}_j$
1	0.023	0.811	26	0.174	0.323
2	0.037	0.749	27	0.176	0.553
3	0.109	0.562	28	0.018	0.883
4	0.027	0.618	29	0.041	0.794
5	0.136	0.487	30	0.259	0.357
6	0.519	0.229	31	0.098	0.461
7	0.058	0.601	32	0.031	0.522
8	0.159	0.425	33	0.027	0.723
9	0.017	0.848	34	0.151	0.613
10	0.140	0.578	35	0.123	0.384
11	0.054	0.620	36	0.141	0.320
12	0.056	0.754	37	0.379	0.327
13	0.027	0.527	38	0.056	0.751
14	0.040	0.819	39	0.165	0.345
15	0.181	0.315	40	0.135	0.500
16	0.122	0.547	41	0.055	0.569
17	0.112	0.324	42	0.110	0.468
18	0.056	0.639	43	0.056	0.785
19	0.155	0.325	44	0.027	0.659
20	0.132	0.370	45	0.123	0.461
21	0.059	0.645	46	0.059	0.468
22	0.063	0.533	47	0.230	0.287
23	0.115	0.446	48	0.447	0.301
24	0.117	0.433	49	0.018	0.727
25	0.018	0.805	50	0.069	0.479

Note. Results are based on the chain of length 30,000 and burnin 15,000.

4.6 Discussion

Online education is becoming ubiquitous and when coupled with intelligent tutoring systems and sequenced training modules, it provides an opportunity to systematically train subjects on a long list of prespecified attributes while keeping careful track of progress. This is an

Table 4.4: Skill Mastery Rate Over Time

Skills	Time point 1	Time point 2	Time point 3	Time point 4	Time point 5
90° x-axis	0.530	0.607	0.632	0.652	0.670
90° y-axis	0.618	0.678	0.712	0.749	0.781
180° x-axis	0.473	0.536	0.575	0.621	0.670
180° y-axis	0.459	0.521	0.547	0.601	0.661

Note. Results are based on the chain of length 30,000 and burnin 15,000.

Table 4.5: Number of Skill Mastered Over Time

Number of Skills	Time point 1	Time point 2	Time point 3	Time point 4	Time point 5
0	121	99	93	81	70
1	37	28	23	23	23
2	33	41	36	32	26
3	13	20	25	26	27
4	147	163	174	189	205

Note. Results are based on the chain of length 30,000 and burnin 15,000.

ideal setting for utilizing latent class models for cognitive diagnosis. However, the static models that have typically been considered are not adequate when training and learning are the goals, which suggests that latent class models for learning in cognitive diagnosis will be a fruitful and fertile area for research and applications. Here we lay out some of the possibilities for modeling learning trajectories in cognitive diagnosis, starting with the most general model with 2^{KT} parameters, and then a more parsimonious model for non-decreasing trajectories with $(T+1)^K$ parameters. In most applications this will be too many to calibrate, and we further restrict the model to arrive at the FOHM, with 3^K parameters to estimate in the case of nondecreasing trajectories. This model is quite flexible, and is practical when large samples are available, when K is relatively small, or when a long list of attributes can be partitioned into small subgroups of attributes that are studied together and can be modeled separately with distinct FOHMs. In order to further reduce the parameters of transition models for learning, covariates may be used, and this is a wide-open area for future research. Covariates may involve demographic variables, measures of the amount of practice one has done, dummy variables for interventions, and many other predictors of learning. To address the heterogeneity of learning rates across subjects in a population, continuous

random effects may be used, which can be interpreted as latent variables representing the construct of general learning ability as studied by Wang et al. (2017). Measuring such a construct may be of intrinsic interest, beyond simply serving to address the heterogeneity.

The directions for extending static cognitive diagnosis models to include the dynamics of learning are numerous, but should not outrace the empirical data on learning by much. There is a need for more applied research to gather data applicable to learning research, which will afford the opportunity to investigate the particular models and covariates for learning that provide the best fits and yield the most accurate and useful predictions. The spatial reasoning example in our real data study was gathered carefully in a laboratory according to a designed study, and illustrates how a model such as the FOHM can be used in practice. The accuracy of learning transition probability estimates was limited by the modest sample size. However, the MCMC algorithm ran efficiently with these data and indicated convergence to the posterior distribution of the parameters. The model complexity of the FOHM is difficult to compare to the higher-order model with covariates of Wang et al. (2017). The number of structural parameters is much higher in the FOHM, but the higher-order model includes a latent learning ability parameter for each individual, resulting in more actual transition probabilities. Nevertheless, these two models yielded results that were relatively consistent.

A common aspect of the FOHM and the models of Wang et al. (2017) is that of nondecreasing trajectories. Yet another direction to consider in latent class models for learning, is how to build relatively parsimonious models that allow for traits to be learned and unlearned. This modeling of retention may prove especially useful in long range longitudinal studies more than in the relatively short time duration of the spatial reasoning training and assessment.

In our study we assumed the item parameters were invariant over time. It is unclear whether fixing item parameters over time is a necessary condition for identifying the model parameters. Additional theoretical research is needed to establish general model identifiability conditions.

Models for learning will necessarily be more complicated than static models, because they combine the measurement model with a transition model. Computational aspects of fitting these models will be a critical area of research. In the simulation study and real data analysis the DINA model was used. When attributes are so clearly defined, such as the particular rotation operations in the data example that must be applied conjunctively, the DINA model can fit quite adequately as shown by Wang et al. (2017). However, in applications with less clear attributes, perhaps leading to some Q-matrix misspecification, more general CDMs will prove useful. Coupling the FOHM or another learning trajectory model with a more general measurement model (e.g., see Henson et al., 2009; Xu, 2017) can present computational challenges and the need for research and software development. Another area of research that will play a vital role is assessment of model fit. The dynamic aspect of this model presents another dimension of fit assessment than is required by static models, and techniques that target particular aspects of misfit will be more useful than global measures of fit. The general method of posterior predictive checking (Sinharay et al., 2006; Sinharay, 2006; Sinharay and Almond, 2007) allows one to study residuals that pertain to a particular aspect of fit, such as the measurement model or the transition model, and there is a need for research into the residuals that best evaluate the fit of each. As more data are collected through online learning, new questions and applications will arrive that will require additional methodological research in this emerging area of psychometrics.

Table 4.6: Estimated prior for starting attributes and transition matrix

	$\boldsymbol{\alpha}_c$	$\hat{\boldsymbol{\pi}}_1$	$\hat{\omega}_{1\|i}$	$\hat{\omega}_{2\|i}$	$\hat{\omega}_{3\|i}$	$\hat{\omega}_{4\|i}$	$\hat{\omega}_{5\|i}$	$\hat{\omega}_{6\|i}$	$\hat{\omega}_{7\|i}$	$\hat{\omega}_{8\|i}$	$\hat{\omega}_{9\|i}$	$\hat{\omega}_{10\|i}$	$\hat{\omega}_{11\|i}$	$\hat{\omega}_{12\|i}$	$\hat{\omega}_{13\|i}$	$\hat{\omega}_{14\|i}$	$\hat{\omega}_{15\|i}$	$\hat{\omega}_{16\|i}$
1	0000	0.31	0.84	0.01	0.01	0.01	0.02	0.02	0.01	0.01	0.01	0.01	0.02	0.01	0.02	0.01	0.01	0.01
2	0001	0.02		0.13		0.11		0.18		0.10		0.11		0.12		0.12		0.13
3	0010	0.01			0.12	0.11			0.13	0.13			0.16	0.11			0.12	0.11
4	0011	0.01				0.31				0.25				0.21				0.23
5	0100	0.05					0.37	0.15	0.11	0.08					0.09	0.07	0.07	0.07
6	0101	0.02						0.72		0.12						0.09		0.06
7	0110	0.02							0.50	0.17							0.18	0.15
8	0111	0.01								0.78								0.22
9	1000	0.03									0.13	0.09	0.17	0.10	0.19	0.10	0.12	0.10
10	1001	0.01										0.22		0.25		0.29		0.24
11	1010	0.02											0.73	0.08			0.12	0.07
12	1011	0.02												0.43				0.57
13	1100	0.05													0.72	0.08	0.09	0.12
14	1101	0.02														0.47		0.53
15	1110	0.03															0.55	0.45
16	1111	0.38																1.00

Note. Results are based on the chain of length 30,000 with a burnin of 15,000.
The skills represented by $\boldsymbol{\alpha}_c$ are labeled in the order of 90^0 x-axis, 90^0 y-axis, 180^0 x-axis and 180^0 y-axis.
$\omega_{j|i}$ refers to the transition probability from the i^{th} attribute profile to the j^{th} attribute profile.

References

Achacoso, T., and Yamamoto, W. (1992), *AY's Neuroanatomy of C. elegans for Computation*, Boca Raton, FL: CRC Press.

Barabási, A.-L., and Albert, R. (1999), "Emergence of scaling in random networks," *Science*, 286(5439), 509–512.

Blitzstein, J., and Diaconis, P. (2010), "A sequential importance sampling algorithm for generating random graphs with prescribed degrees," *Internet Mathematics*, 6, 489–522.

Brooks, S. P., and Gelman, A. (1998), "General methods for monitoring convergence of iterative simulations," *Journal of Computational and Graphical Statistics*, 7(4), 434–455.

Chen, X., Li, Y., Wang, P., and Lui, J. C. (2016), "A general framework for estimating graphlet statistics via random walk," *Proceedings of the VLDB Endowment*, 10(3), 253–264.

Chen, Y., Liu, J., Xu, G., and Ying, Z. (2015), "Statistical analysis of Q-matrix based diagnostic classification models," *Journal of the American Statistical Association*, 110(510), 850–866.

Chiu, C.-Y. (2013), "Statistical refinement of the Q-matrix in cognitive diagnosis," *Applied Psychological Measurement*, 37(8), 598–618.

Chiu, C.-Y., Douglas, J. A., and Li, X. (2009), "Cluster analysis for cognitive diagnosis: Theory and applications," *Psychometrika*, 74(4), 633–665.

Chiu, C.-Y., and Köhn, H.-F. (2016), "The reduced RUM as a logit model: Parameterization and constraints," *Psychometrika*, 81(2), 350–370.

Chung, M.-t. (2014), Estimating the Q-matrix for cognitive diagnosis models in a Bayesian framework, PhD thesis, Columbia University.

Corbett, A. T., and Anderson, J. R. (1994), "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User Modeling and User-adapted Interaction*, 4(4), 253–278.

Cowles, M. K., and Carlin, B. P. (1996), "Markov chain Monte Carlo convergence diagnostics: a comparative review," *Journal of the American Statistical Association*, 91(434), 883–904.

Csardi, G., and Nepusz, T. (2006), "The igraph software package for complex network research," *InterJournal*, Complex Systems, 1695.

Culpepper, S. A. (2015), "Bayesian estimation of the DINA model with Gibbs sampling," *Journal of Educational and Behavioral Statistics*, 40(5), 454–476.

de la Torre, J. (2008), "An empirically based method of Q-matrix validation for the DINA model: Development and applications," *Journal of Educational Measurement*, 45(4), 343–362.

de la Torre, J. (2009), Estimation code for the G-DINA model,, in *Presentation at the meeting of the American Educational Research Association, San Diego, CA*.

de la Torre, J., and Chiu, C.-Y. (2016), "A general method of empirical Q-matrix validation," *Psychometrika*, 81(2), 253–273.

de la Torre, J., and Douglas, J. A. (2004), "Higher-order latent trait models for cognitive diagnosis," *Psychometrika*, 69(3), 333–353.

de la Torre, J., and Douglas, J. A. (2008), "Model evaluation and multiple strategies in cognitive diagnosis: An analysis of fraction subtraction data," *Psychometrika*, 73(4), 595–624.

DeCarlo, L. T. (2012), "Recognizing uncertainty in the Q-matrix via a Bayesian extension of the DINA model," *Applied Psychological Measurement*, 36(6), 447–468.

Geweke, J. (1992), "Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments," *Bayesian Stat.*, 4, 169–188.

González-Brenes, J., Huang, Y., and Brusilovsky, P. (2014), General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge,, in *The 7th International Conference on Educational Data Mining*, University of Pittsburgh, pp. 84–91.

González-Brenes, J. P., and Mostow, J. (2013), What and when do students learn? Fully data-driven joint estimation of cognitive and student models,, in *Proceedings of the 6th International Conference on Educational Data Mining*, pp. 236–240.

Grochow, J. A., and Kellis, M. (2007), Network motif discovery using subgraph enumeration and symmetry-breaking,, in *Research in Computational Molecular Biology*, Springer, pp. 92–106.

Henson, R., and Templin, J. (2007), Importance of Q-matrix construction and its effects cognitive diagnosis model results,, in *Annual Meeting of the National Council on Measurement in Education, Chicago, IL*.

Henson, R., Templin, J., and Willse, J. (2009), "Defining a family of cognitive diagnosis models using log-linear models with latent variables," *Psychometrika*, 74(2), 191–210.

Huff, K., and Goodman, D. P. (2007), "The demand for cognitive diagnostic assessment," in *Cognitive Diagnostic Assessment for Education*, eds. J. P. Leighton, and M. J. Gierl Cambridge University Press, pp. 19–60.

Itzhack, R., Mogilevski, Y., and Louzoun, Y. (2007), "An optimal algorithm for counting network motifs," *Physica A: Statistical Mechanics and Its Applications*, 381, 482–490.

Junker, B. W., and Sijtsma, K. (2001), "Cognitive assessment models with few assumptions, and connections with nonparametric item response theory," *Applied Psychological Measurement*, 25(3), 258–272.

Kashani, Z. R., Ahrabian, H., Elahi, E., Nowzari-Dalini, A., Ansari, E. S., Asadi, S., Mohammadi, S., Schreiber, F., and Masoudi-Nejad, A. (2009), "Kavosh: a new algorithm for finding network motifs," *BMC Bioinformatics*, 10(1), 318.

Kashtan, N., Itzkovitz, S., Milo, R., and Alon, U. (2004), "Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs," *Bioinformatics*, 20(11), 1746–1758.

Kaya, Y., and Leite, W. L. (2017), "Assessing change in latent skills across time with longitudinal cognitive diagnosis modeling: An evaluation of model performance," *Educational and Psychological Measurement*, 77(3), 369–388.

Leighton, J. P., and Gierl, M. J. (2007), "Why cognitive diagnostic assessment," in *Cognitive Diagnostic Assessment for Education*, eds. J. P. Leighton, and M. J. Gierl Cambridge University Press, pp. 3–18.

Li, F., Cohen, A., Bottge, B., and Templin, J. (2015), "A latent transition analysis model for assessing change in cognitive skills," *Educational and Psychological Measurement*, 76(2), 181–204.

Lin, W., Xiao, X., Xie, X., and Li, X.-L. (2015), Network motif discovery: A GPU approach,, in *Proceedings of the 31st International Conference on Data Engineering*, IEEE, pp. 831–842.

Liu, J. (2015), "On the consistency of Q-matrix estimation: A commentary," *Psychometrika*, DOI: 10.1007/s11336-015-9487-4.

Liu, J., Xu, G., and Ying, Z. (2012), "Data-driven learning of Q-matrix," *Applied Psychological Measurement*, 36(7), 548–564.

Liu, J., Xu, G., and Ying, Z. (2013), "Theory of the self-learning Q-matrix," *Bernoulli*, 19(5A), 1790.

Maris, E. (1999), "Estimating multiple classification latent class models," *Psychometrika*, 64(2), 187–212.

Milo, R., Itzkovitz, S., Kashtan, N., Levitt, R., Shen-Orr, S., Ayzenshtat, I., Sheffer, M., and Alon, U. (2004), "Superfamilies of evolved and designed networks," *Science*, 303(5663), 1538–1542.

Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., and Alon, U. (2002), "Network motifs: simple building blocks of complex networks," *Science*, 298(5594), 824–827.

Mislevy, R. J., and Wilson, M. (1996), "Marginal maximum likelihood estimation for a psychometric model of discontinuous development," *Psychometrika*, 61(1), 41–71.

Norris, S. P., Macnab, J. S., and Phillips, L. M. (2007), "Cognitive Modeling of Performance on Diagnostic Achievement Tests," in *Cognitive Diagnostic Assessment for Education*, eds. J. P. Leighton, and M. J. Gierl Cambridge University Press, pp. 61–84.

Ribeiro, P., and Silva, F. (2010), G-Tries: an efficient data structure for discovering network motifs,, in *Proceedings of the 2010 ACM Symposium on Applied Computing*, pp. 1559–1566.

Rupp, A. A. (2009), Software for calibrating diagnostic classification models: An overview of the current state-of-the-art,, in *Symposium conducted at the meeting of the American Educational Research Association, San Diego, CA*.

Rupp, A. A., and Templin, J. L. (2008), "The effects of Q-matrix misspecification on parameter estimates and classification accuracy in the DINA model," *Educational and Psychological Measurement*, 68(1), 78–96.

Rupp, A. A., Templin, J. L., and Henson, R. A. (2010), *Diagnostic Measurement: Theory, Methods, and Applications*, New York, NY: Guilford Press.

Shen-Orr, S. S., Milo, R., Mangan, S., and Alon, U. (2002), "Network motifs in the transcriptional regulation network of Escherichia coli," *Nature Genetics*, 31(1), 64–68.

Sinharay, S. (2006), "Model diagnostics for Bayesian networks," *Journal of Educational and Behavioral Statistics*, 31(1), 1–33.

Sinharay, S., and Almond, R. G. (2007), "Assessing fit of cognitive diagnostic models a case study," *Educational and Psychological Measurement*, 67(2), 239–257.

Sinharay, S., Johnson, M. S., and Stern, H. S. (2006), "Posterior predictive assessment of item response theory models," *Applied Psychological Measurement*, 30(4), 298–321.

Studer, C. (2012), Incorporating learning over time into the cognitive assessment framework, PhD thesis, Carnegie Mellon University.

Tatsuoka, C. (2002), "Data analytic methods for latent partially ordered classification models," *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 51(3), 337–350.

Tatsuoka, K. K. (1984), *Analysis of Errors in Fraction Addition and Subtraction Problems*, Champaign, IL: Computer-based Education Research Laboratory, University of Illinois.

Templin, J. L., and Henson, R. A. (2006*a*), A Bayesian method for incorporating uncertainty into Q-matrix estimation in skills assessment,, in *Symposium conducted at the meeting of the American Educational Research Association, San Diego, CA*.

Templin, J. L., and Henson, R. A. (2006*b*), "Measurement of psychological disorders using cognitive diagnosis models.," *Psychological Methods*, 11(3), 287.

Templin, J. L., and Hoffman, L. (2013), "Obtaining diagnostic classification model estimates using Mplus," *Educational Measurement: Issues and Practice*, 32(2), 37–50.

von Davier, M. (2014), "The DINA model as a constrained general diagnostic model: Two variants of a model equivalency," *British Journal of Mathematical and Statistical Psychology*, 67(1), 49–71.

Wang, P., Lui, J. C., Towsley, D., and Zhao, J. (2016), Minfer: A method of inferring motif statistics from sampled edges,, in *Proceedings of the 32nd International Conference on Data Engineering*, IEEE, pp. 1050–1061.

Wang, S., Yang, Y., Culpepper, S., and Douglas, J. (2017), "Tracking skill acquisition with cognitive diagnosis models: Application to spatial rotation skills," *Manuscript submitted for publication*, .

Wernicke, S. (2006), "Efficient detection of network motifs," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(4), 347–359.

Xiang, R. (2013), Nonlinear penalized estimation of true Q-Matrix in cognitive diagnostic models, PhD thesis, Columbia University.

Xu, G. (2017), "Identifiability of restricted latent class models with binary response," *Annals of Statistics*, 45, 675–707.

Xu, G., and Zhang, S. (2016), "Identifiability of diagnostic classification models," *Psychometrika*, 81(3), 625–649.

Xu, Y., and Mostow, J. (2012), Comparison of methods to trace multiple subskills: Is LR-DBN best?,, in *Proceedings of the 5th International Conference on Educational Data Mining*, pp. 41–48.

Ye, S., Fellouris, G., Culpepper, S., and Douglas, J. (2016), "Sequential detection of learning in cognitive diagnosis," *British Journal of Mathematical and Statistical Psychology*, 69, 139–158.

Yoon, S. Y. (2011), Psychometric properties of the Revised Purdue Spatial Visualization Tests: Visualization of rotations (the Revised PSVT-R), PhD thesis, Purdue University.

Zhang, J., and Chen, Y. (2013), "Sampling for conditional inference on network data," *Journal of the American Statistical Association*, 108, 1295–1307.