An Information Geometric Approach to Increase Representational Power in Unsupervised Learning

Simon Luo

Supervisor: Prof. Fabio Ramos
Dr. Lamiae Azizi

School of Computer Science
Faculty of Engineering
The University of Sydney

A thesis submitted in partial fulfilment of requirements for the degree of
Doctor of Philosophy

July 2021

Statement of Originality

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes.

I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Signature*

Simon Luo
July 2021

Authorship Attribution Statement

- Chapter 3 of this thesis is published as **S. Luo**, M. Sugiyama, Bias-Variance Trade-Off in Hierarchical Probabilistic Models using Higher-Order Feature Interactions, Thirty-Third AAAI Conference on Artifical Intelligence, (AAAI-19). [65]

- Chapter 4 of this thesis is published as **S. Luo**, L. Azizi, M. Sugiyama, Hierarchical Probabilistic Model for Blind Source Separation via Legendre Transformation, The 37th Conference on Uncertainty in Artificial Intellegence (UAI2021). [61]

- Chapter 5 of this thesis is published as **S. Luo**, F. Zhou, L. Azizi, M. Sugiyama, Additive Poission Process: Learning Intensity of Higher-Order Interactions in Stochastic Processes. [66]

- Chapter 6 of this thesis is published as **S. Luo**, VW. Chu, J. Zhou, F. Chen, RK. Wong, RK. Wong, Multitask Learning for Sparse Failure Prediction, 23rd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2019). [62]

- Chapter 6 of this thesis is published as **S. Luo**, VW. Chu, J. Zhou, F. Chen, RK. Wong, RK. Wong, Multi-Task Learning by Hierarchical Dirichlet Mixture Model for Sparse Failure Prediction, International Journal of Data Science and Analytics (JDSA 2020). [63]

Acknowledgements

This thesis could not have been completed without the support of many people who were there for me throughout this journey.

Firstly I would like to thank my supervisors Fabio Ramos and Lamiae Azizi at The University of Sydney for their support for my research. Fabio, over the years, your strong leadership has built one of the largest machine learning research groups in the country. Working with this group has provided the one of the best opportunities to exchange research ideas and build friendships. Lamiae, the support and opportunities which you have provided me is in no doubt the reason why I am so successful in my research. Not only that you have provided me support to do my research, you provide the support and care to ensure that I am growing in all aspects of my life.

Next, I would like to extend my sincerest thanks to Mahito Sugiyama at The National Institute of Informatics. My two 6 month internship is no doubt has been a life-changing experience for me. It was my first experience working overseas and it has no doubt improved me as a person as a whole. My work ethics, approach to research and social ability has improved significantly after working with you. I am certain that any major achievements that I make in the future are in no doubt influenced by the years that I have worked with you.

I would also like to thank Victor Chu at Nanyang Technical University and Raymond Wong at The University of New South Wales for mentoring me at the beginning of my PhD. I would not have made it this far without your support. It is only with your guidance, support and direction that you have provided me that I was able to make it this far. The life advice which you both have provided me over the years has opened doors to many opportunities over the years. To this day, I still follow almost every advice you have given me over the years.

I would like to thank Fang Chen, Yang Wang, Jianlong Zhou and Zhidong Li from Data61, CSIRO for encouraging me to pursue a PhD. Jianlong and Zhidong, you two have given me the first taste of research when I was working at Data61. There are very few people who are willing to spend the time and patients to provide the close training and mentoring that you have given me over the years. You two have both trained and mentored me from almost knowing nothing at all to getting a PhD in machine learning. Fang and Yang, you two have built one of the most friendliest and most talented groups of researchers and engineers from such diverse backgrounds. Without a doubt that the interaction with your group has given me such a broad understanding of the field. Thinking about this group always gives me a feeling that we are all in one big family because of all the support and care we have provided for each other.

Tony Huang from University of Technology Sydney, you are one of the most supportive and understanding person I have met. It seems like whatever challenges that I have faced over the years. You seem to go out of your way to try and make things work for me. It is much appreciated that you have always looked out for me for all these years.

And thank you to all the interns, honours students, PhD students and Postdocs who have joined me on this journey from The Robot Learning and Reasoning group at the School of Computer Science, The University of Sydney, The Computational and Mathematical Intelligence Group at The School of Mathematics and Statistics, The University of Sydney, Data Driven Cities Group at Data61, CSIRO, Centre for Translational Data Science, The University of Sydney and Sugiyama Lab at The National Institute of Informatics. The journey would not have been as fun without you guys.

Lastly, I would like to thank Mick Boyle, Data61, CSIRO and The University of Sydney for providing the scholarship for me to pursue a PhD.

Abstract

Machine learning models increase their representational power by increasing the number of parameters in the model. The number of parameters in the model can be increased by introducing hidden nodes, higher-order interaction effects or by introducing new features into the model. In this thesis we study different approaches to increase the representational power in unsupervised machine learning models. We investigate the use of incidence algebra and information geometry to develop novel machine learning models to include higher-order interactions effects into the model. Incidence algebra provides a natural formulation for combinatorics by expressing it as a generative function and information geometry provides many theoretical guarantees in the model by projecting the problem onto a dually flat Riemannian structure for optimization. Combining the two techniques together formulates the information geometric formulation of the binary log-linear model. In this thesis we provide the following contributions: (1) we use the information geometric formulation of the binary log-linear model to formulate the higher-order Boltzmann machine (HBM) to compare the different behaviours when using hidden nodes and higher-order feature interactions to increase the representational power of the model. (2) We then apply the concepts learnt from this study to include higher-order interaction terms in Blind Source Separation (BSS) and (3) we create an efficient approach to estimate higher order functions in Poisson process. (4) We explore the possibility to use Bayesian non-parametrics to automatically reduce the number of higher-order interactions effects included in the model.

Table of contents

List of figures

List of tables

CHAPTER 1

Introduction

We introduce our thesis by first providing a motivation for our work. We then provide a brief overview of the contributions of this thesis. Lastly, we provide a brief outline of the organization of the remaining Chapters of the thesis.

1.1 Motivation

The representational power in a machine learning model describes the ability for a model to make the correct prediction. Additional parameters introduced models to increase its representational power. This can be done by either adding hidden nodes, introducing interaction effects between variables or adding more features to the model. However, selecting the ideal number of parameters in the model to minimize error in the model is still a difficult task. If there are too few parameters in the model, then the model will underfit and produce a biased prediction because it is unable to express the relations between features and the target outputs. If there are too many parameters, the model will overfit and produce a prediction with high variance because using a large number of parameters makes it sensitive to random noise. The balance between underfitting and overfitting is a classical problem in machine learning known as the bias-variance trade-off which is one of the fundamental challenges to reduce error in a machine learning model.

In this thesis, we investigate the inclusion of higher-order interaction effects in unsupervised learning models by using incidence algebra and information geometry. Incidence algebra provides a mathematical expression to write combinatorics as a generative function.

The interaction effects in machine learning models can be represented using a combinatoric function. Information geometry provides a geometric perspective to information theory which allows for efficient optimization techniques. Optimization in Information geometry is performed by projecting the graph structure onto a dually flat Riemannian manifold so that the discrete graph structure is represented as two continuous smooth convex functions. This projection onto a smooth manifold allows for efficient continuous optimization techniques such as gradient descent and natural gradient on a discrete structure. In this thesis we first study the difference between increasing the representation power of unsupervised models using hidden nodes and higher-order interactions effects. We then apply our technique of including higher-order feature interactions in blind source separation (BSS) and estimating higher-order functions in Poisson processes. Lastly, we explore using Bayesian non-parametrics to automatically select number of higher-order interactions effects included in the model.

1.2 Contributions

This section provides an overview of the contributions of this thesis.

1.2.1 Learning Higher-Order Interaction Effects

Chapter 3 investigates using incidence algebra to include higher-order interaction between features and parameters in the model. Incidence algebra can be used to formulate higher-order interaction effects as a generative function. The generative function provides a convenient framework to mathematically represent the higher-order interaction terms between features. We demonstrate the effectiveness of the formulation to include higher-order feature interaction effects by applying it to the Boltzmann machine. We then explore the use of continuous optimization algorithms for discrete structures using techniques in information geometry. Information geometry projects the discrete graph structure onto a dually flat Riemmanian manifold to use efficient optimization techniques such as gradient descent and natural gradient descent. Our formulation of the Boltzmann machine can be used as a base model to include higher-order interactions effects or to estimate high-dimensional functions as presented in Chapter 4 and Chapter 5.

1.2.2 Hidden Nodes vs Higher-Order Interactions Effects

The number of parameters in the model can be increased by either introducing hidden nodes or higher-order interaction effects. To this date, there has not been a study which

compares the behaviour of the two approaches. Chapter 3 first uses incidence algebra and information geometry to formulate the higher-order Boltzmann machine (HBM). We then perform an empirical experiment to study the behaviour of increasing the representation power of unsupervised learning models using higher-order feature interactions and hidden nodes by comparing the error in the model.

1.2.3 Higher-Order Interactions in Unsupervised Learning

Chapter 4 extends the information geometry formulation of the binary log-linear model to include higher-order interactions between source signals in blind source separation (BSS). The traditional approach to BSS uses techniques such as independent component analysis (ICA) or non-negative matrix factorization (NMF). Each of these approaches have their own limitations. ICA assumes that the transformation from the source signal to the received signal is an affine transformation. However, in practice, the mixing of the signal is much more complicated than an affine transformation. Another approach is to use NMF to separate the mixed signal into the latent source signal. However, the NMF is a degenerate approach and loses information in its reconstruction. Chapter 4 introduces a novel approach that is able to capture complex interactions in BSS by considering the higher-order interactions between signals without the loss of information.

1.2.4 Estimating Higher-Order Functions

Chapter 5 extends the information geometry formulation of the binary log-linear model introduced in Chapter 4 to estimate high-dimensional intensity function using the Poisson process respectively. In a multivariate Poisson process, it is difficult to estimate the intensity function of events with sparse observations or low event rate. Learning the intensity function directly from the observations is difficult because there are often too few joint samples or no samples to learn the joint intensity function. We propose a novel approach to estimate higher-order intensity function between stochastic processes by using lower dimensional projections of the intensity function.

1.2.5 Selecting Higher-Order Model-Parameters

Selecting the number of parameters in a model is challenging in any real-world applications where there are problems arising from data quality issues such as sparsity, noisy labels, missing data and sampling bias. A naive implementation of a Bayesian non-parametric approach will find it challenging to select the number of parameters in the model. Often for

these problems the parameters of the model are selected in consultation with domain experts. However, for a complex problem it may be difficult for the domain experts to provide the guidance to select parameters for the model because there are too many factors to consider. Chapter 6 investigates designing machine learning models to automatically learn the number of parameters in a model by using multi-task learning solution. The multi-task learning is performed by using a Bayesian non-parametric approach to create a model for agglomerative hierarchical clustering to learn the model parameters. The hierarchical cluster provides an framework which is able to automatically select the higher-order interaction effect terms to be included in the model.

1.3 Thesis Outline

The remainder of this thesis contains a background chapter, four contributing chapters followed by the conclusion. A brief outline of each chapter is given below.

Chapter 2 presents the background knowledge to understand the theoretical contribution in the four contributing chapters. The background chapter is written with the assumption that the reader has a university graduate level of understanding in machine learning, statistics, mathematics and computer science. The background chapter provides an introduction to the concepts used in information geometry and incidence algebra and its relation to a partially ordered set (poset) to make high-dimensional approximations.

The four contributing chapters are from Chapter 3 to Chapter 6 which follow the same basic structure, first an introduction which contains the motivation and related work, the methodology, then followed by the experimental results and the summary of the chapter. Chapter 3 investigates how to include higher-order features interactions into the Boltzmann machines by using incidence algebra and information geometry to formulate our proposed approach called the Higher-Order Boltzmann machine (HBM). We then apply our proposed approach to study the behaviour of the model when increasing the representation power of models using higher-order nodes and hidden nodes by comparing the HBM with the restricted Boltzmann machine (RBM). In the following chapters the HBM can be generalized as a log-linear model to formulate other machine learning models.

In the next two chapters, we demonstrate the capability of including higher-order interactions in other unsupervised learning models. We demonstrate the capability of the model to include higher-order interactions effects in Chapter 4 by applying the log-linear model to perform blind source separation (BSS) on signals with higher-order interactions effects in the mixed signal. We then demonstrate the capability of the model to approximate high-dimensional functions in Chapter 5 by formulating the Poisson process to learn the intensity

function of the higher-order interactions between stochastic processes. Both applications to BSS and Poisson processes have shown superior performance by including higher-order features in the model.

Finally, Chapter 6 explores using a Bayesian non-parametric approach for model selection by providing a framework for parameter selection by using techniques in transfer learning. It does this by using a hierarchy of Dirichlet process priors to formulate a model for agglomerative hierarchical clustering to learn a common latent distribution between datasets. The latent distribution is then used as a prior for the model to quantify uncertainty of the unknown parameters in the model. We demonstrate the superiority of our approach by applying our proposed approach to a real-world application to make predictions on critical component failure in large infrastructure networks.

This thesis is then closed in Chapter 7 with a review of the contributions and potential future research directions for applying incidence algebra and information geometry in machine learning.

CHAPTER 2

Background

In this chapter, we present the preliminary material for this thesis. We first define how we can increase the representational power in unsupervised learning. Then we introduce the background on the information geometric formulation of the exponential family, this is then followed by an introduction on incidence algebra to formulate higher-order interaction effects as a generative function and finally we introduce the log-linear model on a partial ordered set (poset).

2.1 Models for Unsupervised Learning

Unsupervised learning aims to find previously undetected patterns in a dataset without any pre-existing labels. Typically in unsupervised learning intends to infer an a priori probability distribution $p_X(\mathbf{x})$, where $\mathbf{x} = [x_1, \ldots, x_N]$. A large majority of the approaches assume independence between the features in $\mathbf{x}$ to simplify the inference algorithm which limits the representational power of the model. A model is able to increase its representational power by including the interaction terms which is usually represented by a multiplicative product between two or more of the features. An example of this by using additive models is,

$$\begin{aligned} \log p(\mathbf{x}) = \sum_i \theta^{(i)} x^{(i)} + \sum_{i<j} \theta^{(ij)} x^{(i)} x^{(j)} + \sum_{i<j<k} \theta^{(ijk)} x^{(i)} x^{(j)} x^{(k)} \\ + \ldots + \theta^{(1\ldots n)} x^{(1)} x^{(2)} \ldots x^{(n)} - \psi, \end{aligned} \tag{2.1}$$

where $\theta = (\theta^{(1)}, \ldots, \theta^{(12\ldots n)})$ is the parameter vector, ψ is the normalizer and $\eta = (\eta^{(1)}, \ldots, \eta^{(12\ldots n)})$ represents the expectation of variable combinations such that,

$$\begin{aligned} \eta^{(i)} &= \mathbb{E}\left[x^{(i)}\right] = p\left(x^{(i)} = 1\right) \\ \eta^{(ij)} &= \mathbb{E}\left[x^{(i)}x^{(j)}\right] = p\left(x^{(i)} = x^{(j)} = 1\right) \\ &\vdots \\ \eta^{(12\ldots n)} &= \mathbb{E}\left[x^{(1)}x^{(2)}\ldots x^{(n)}\right] = p\left(x^{(1)} = x^{(2)} = \ldots = x^{(n)} = 1\right) \end{aligned} \tag{2.2}$$

We are able to increase the representational power of the model by introducing more parameters into the model by using the multiplicative product to represent the higher-order interaction effects [25]. Our four contributing chapter explores increasing the representational power of unsupervised learning using Equation (2.1) and Equation (2.2). Chapter 3 formulates our proposed approach using information geometry to increase the representational power in the Boltzmann Machine. We then compare our approach to increasing the representational power by using hidden nodes. We then explore how we can use our formulation to include higher-order terms in blind source separation (BSS) in Chapter 4. Chapter 5 shows how higher-order terms can be used to approximate high dimensional functions in Poisson processes. Finally 6 explored how we could use Bayesian non-parametric techniques to automatically select the terms to include into the model.

We will first introduce the unsupervised learning models studied in this thesis. Then we will explore solutions found in information geometry to project the unsupervised learning problem onto a statistical manifolds. By using the information geometric projection is able advantage of many of theoretical guarantees found in the literature of information geometry. In the next section we will introduce how the exponential family can be represented from an information geometry perspective.

2.1.1 Boltzmann Machine

A *Boltzmann Machine* [1] is represented using an undirected graph $G = (V, E)$ with a vertex set $V = \{v_1, v_2, \ldots, v_n, h_1, h_2, \ldots, h_m\}$ and an edge set $E \subseteq \left\{\left\{x_i, x_j\right\} \mid x_i, x_j \in V\right\}$. Each vertex can be either "visible" or "hidden", where the visible node represents a direct observation from the dataset and the hidden node represents latent features detected in the model. The state of each vertex is represented by $\mathbf{x} = (x_1, x_2, \ldots, x_{n+m}) \in \{0,1\}^{n+m}$, which is a concatenation of $\mathbf{v} \in \{0,1\}^n$ and $\mathbf{h} \in \{0,1\}^m$. The generalized expression for the energy of the joint

configuration $(\mathbf{v}, \mathbf{h})$ of the network is defined by:

$$\Phi(\mathbf{x}; \mathbf{b}, \mathbf{w}) = -\sum_{i=1}^{n+m} x_i b_i - \sum_{i,j=1}^{n+m} x_i x_j w_{i,j}, \tag{2.3}$$

where the parameters $\mathbf{b} = (b_1, b_2, \ldots, b_{n+m})$ are the biases and $\mathbf{w} = (w_{1,2}, w_{1,3}, \ldots, w_{n+m-1,n+m})$ are the weights which are placed on the vertices and the edges respectively. If $\{i, j\} \notin E$ then $w_{i,j} = 0$ can be used to represent no edge connection between the two vertices. The probability of the current configuration of the entire Boltzmann Machine network G is given by:

$$p(\mathbf{x}; \mathbf{b}, \mathbf{w}) = \frac{1}{Z} \exp\left(-\Phi(\mathbf{x}; \mathbf{b}, \mathbf{w})\right), \tag{2.4}$$

where Z is the partition function given by:

$$Z = \sum_{\mathbf{x} \in \{0,1\}^{n+m}} \exp\left(-\Phi(\mathbf{x}; \mathbf{b}, \mathbf{w})\right), \tag{2.5}$$

which ensures $\sum_{\mathbf{x} \in \{0,1\}^{n+m}} p(\mathbf{x}; \mathbf{b}, \mathbf{w}) = 1$.

The Restricted Boltzmann Machine (RBM) is a BM in which the vertices form a bipartite graph. The two groups of nodes are known as the "visible layer" and the "hidden layer". The visible layer represents the direct observations on the dataset. An activation function is placed on the hidden nodes to model the interactions between features by the following equation:

$$p\left(h_j = 1 \middle| \mathbf{v}\right) = \sigma\left(b_j + \sum_i v_i w_{ij}\right),$$

where σ is the sigmoid function.

The RBM is one approach used to increase the representational power of machine learning models. In Chapter 3, we will look into create a novel framework to represent higher-order feature interactions to increase the representational power of the model. We then compare the characteristics of the two different approaches to increase the representational power of the model.

2.1.2 Independent Component Analysis

In this thesis we will study how we can include higher-order feature interactions in Inependent Component Analysis (ICA). Independent Component Analysis (ICA) [43] is the most popular

model used for Blind Source Separation (BSS). BSS is mathematically defined as a function f which is able to separate a set of *mixed signals* X into a set of *source signals* Z, i.e., $Z = f(X)$. However, it is generally considered to be impossible for the function f to recover the scale and the original order of the source signal. If we apply this limitation to the BSS problem, the problem is mathematically reduced to $Z \propto f(X)$.

We define ICA by letting $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_M], \mathbf{z}_m \in \mathbb{R}^N$ be a set of *source signals* and $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_M], \mathbf{x}_m \in \mathbb{R}^L$ be a set of *received signals*. The received signal $\mathbf{X}$ is an affine transformation of the source signal $\mathbf{Z}$ by a *mixing matrix* $\mathbf{A} \in \mathbb{R}^{L \times N} : \mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_N]$, written as

$$\mathbf{X} = \mathbf{A}\mathbf{Z}, \qquad x_{lm} = \sum_{n=1}^{N} a_{ln} z_{nm}, \tag{2.6}$$

where L is the number of received signals, N is the number of source signals and M is the sample size. The source signal can be recovered from $\mathbf{A}$ and $\mathbf{X}$

if $L \geq N$, while the source signal $\mathbf{Z}$ and the mixing matrix $\mathbf{A}$ are unknown in BSS. Our objective is to estimate the source signal $\mathbf{Z}$ by learning the parameters in $\mathbf{A}$ using the received signal $\mathbf{X}$ as the input to the model. The traditional formulation of ICA aims to learn the inverse of $\mathbf{A}$. Differently, our approach is able to learn the parameters of $\mathbf{A}$ directly.

However, there are several limitations in the ICA model. Firstly, we assume that the forward model to convert the source signal to the mixed signal is an affine transformation. However, this is usually an over simplification and the interaction between signals in the real-world is often more complex than an affine transformation. Secondly, the sign of the signal is not recovered in ICA. This could be problematic in many critical applications. Thirdly, ICA is non-convex, so different weight initialization will result in different model output. Our solution explored in Chapter 4 overcomes all these limitations in ICA.

2.1.3 Multivariate Poisson Processes

In the previous section, we introduced how we can formulate Independent Component Analysis (ICA) with higher-order interactions. In this section, we will show how we can estimate higher order functions in stochastic processes. A stochastic process is defined as a collection of random variables that is indexed by some mathematical set, so that each random variable of the stochastic process is uniquely associated with an element in the set. A point process is a stochastic process composed of a time series of binary events that occur in continuous time. They are used to describe a finite set of observations. For stochastic processes where the values are non-negative, integer and non-decreasing, we can

characterise them as a counting process [27]. One of the most prevalent type of point process and counting process is the Poisson process which is characterized by an arrival intensity λ. An inhomogeneous Poisson process is a general type of process, where the arrival intensity changes with time. The process with time-changing intensity $\lambda(t)$ is defined as a counting process $\mathbb{N}(t)$, which has an independent increment property. For all time $t \geq 0$ and changes in time $\delta \geq 0$, the probability p for the observations is given as,

$$\begin{aligned} p(\mathbb{N}(t+\delta) - \mathbb{N}(t) = 0) &= 1 - \delta\lambda(t) + o(\delta) \\ p(\mathbb{N}(t+\delta) - \mathbb{N}(t) = 1) &= \delta\lambda(t) + o(\delta) \\ p(\mathbb{N}(t+\delta) - \mathbb{N}(t) \geq 2) &= o(\delta) \end{aligned}$$

where $o(\cdot)$ denotes little-o notation. Given a realization of timestamps $\{t_i\}_{i=1}^N \subseteq [0,T]$ from an inhomogeneous Poisson process with intensity $\lambda(t)$. Each t_i is the time of occurrence for the i-th event and T is the observation duration for this process. The likelihood for the Poisson process is given by Daley and Vere-Jones [27],

$$p\left(\{t_i\}_{i=1}^N \mid \lambda(t)\right) = \exp\left[-\int_0^T \lambda(t)\,dt\right] \prod_{i=1}^N \lambda(t_i). \tag{2.7}$$

We define the functional prior on $\lambda(t)$ as,

$$\lambda(t) := g[f(t)] = \exp[f(t)]. \tag{2.8}$$

The function $g(\cdot)$ is a positive function to guarantee the non-negativity of intensity which we choose to be the exponential function, and our objective is to learn the function $f(t)$. The log-likelihood for the Poisson process with the functional prior can be described as follows:

$$\log p\left(\{t_i\}_{i=1}^N \mid \lambda(t)\right) = \sum_{i=1}^N f(t_i) - \int_0^T \exp[f(t)]\,dt.$$

The first term $\sum_{i=1}^N f(t_i)$ is the unnormalized intensity function. The second term $\int \exp[f(t)]dt$ is used to normalize the intensity function into the log-probability.

We generalize the likelihood function in Equation (2.7) to a multi-dimensional Poisson process

$$p\left(\{\mathbf{t}_i\}_{i=1}^N \mid \lambda(\mathbf{t})\right) = \exp\left[-\int \lambda(\mathbf{t})\,d\mathbf{t}\right] \prod_{i=1}^N \lambda(\mathbf{t}_i), \tag{2.9}$$

where $\mathbf{t} = [t^{(1)}, \ldots, t^{(D)}]$ and D is the number of dimensions. Then the log-likelihood of the multi-dimensional Poisson process with the functional prior is

$$\log p\left(\{\mathbf{t}_i\}_{i=1}^{N} \mid \lambda(\mathbf{t})\right) = \sum_{i=1}^{N} f(\mathbf{t}_i) - \int \exp[f(\mathbf{t})]\, d\mathbf{t}. \tag{2.10}$$

The objective of the multivariate Poisson process is to learn the joint intensity function of a multi-dimensional stochastic processes. This is a challenging problem because the event rate for the joint process can be extremely low by definition, leading to sparse observations. The multivariate majority of the multivariate Poisson process found in literature learns the conditional intensity function [113]. Differently, in Chapter 5 we look into estimating the joint intensity function in the Poisson Process which is considered as a more general solution. This form of the multivariate Poisson process is closely related to the Pitman-Yor process and the Poisson-Dirichlet Process which we will study in more details in Chapter 6 [20].

2.2 Information Geometric Formulation of the Exponential Family

In this section, we will present the derivation for the information geometric formulation of the exponential family. We first define two coordinate system, $\theta = \{\theta_1, \ldots, \theta_n\}$ and $\eta = \{\eta_1, \ldots, \eta_n\}$ to be its Legendre transformation (The full proof of this property is located in Section A.1) [9]. That is,

$$\theta = \nabla\varphi(\eta), \qquad \eta = \nabla\psi(\theta).$$

Where $\psi(\theta)$ represents the e-projection (exponential-projection) and $\varphi(\eta)$ represents the m-projection (mixture-projection). Using this notation the probability distribution for natural exponential family is written as [9],

$$p(x;\theta) = \exp\left[\sum_i \theta_i x_i - \psi(\theta)\right],$$

where, $\exp[\psi(\theta)]$ is the partition which which is uniquely determined by,

$$\psi(\theta) = \log \int \exp\left[\sum \theta_i x_i\right] d\mathbf{x}.$$

The coordinate systems θ becomes the natural parameter and η becomes the expectation parameter. In the following sections, we introduce incidence algebra to formulate a log-linear model on a poset which is a class of models in the exponential family.

2.3 Incidence Algebra

In this section we introduce the inclusion-exclusion principle, the zeta and the möbius function used in incidence algebra and the Möbius inversion formula.

2.3.1 Inclusion-Exclusion Principle

The inclusion-exclusion principle is a counting technique which generalizes the familiar methods of obtaining the number of elements in a union between finite sets. For a given finite set A, B and C. The union between the three finite set is given as,

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|. \tag{2.11}$$

This mathematical expression is illustrated graphically in Figure 2.1. By inspecting Equation (2.11), we can see that as the number of terms to compute the union grows combintorially with respect to the number of input sets.

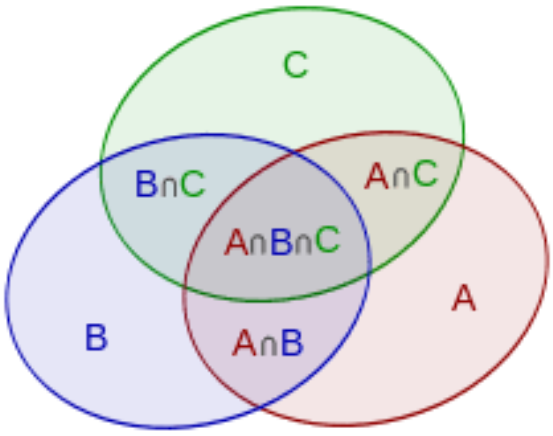


Fig. 2.1 An illustration of the inlcusion-exclusion principle using a Venn diagram of three sets.

Incidence algebra provides a natural construction of generating functions for the inclusion-exclusion principle [85]. Incidence algebra is based on the principles of enumeration to provide a powerful representation for a set of objects to be counted that is equipped with a

partial order structure. Here we introduce the partial order structure by letting $(\Omega, \leq)$ be a *partially ordered set* (*poset*) [36], where a *partial order* $\leq$ is the relation between elements in a set S. The *poset* must satisfy the following three properties for $\omega, \omega', \omega'' \in \Omega$: (1) $\omega \leq \omega$ (reflexivity), (2) $\omega \leq \omega', \omega' \leq \omega \Rightarrow \omega = \omega'$ (anti-symmetry), and (3) $\omega \leq \omega', \omega' \leq \omega'' \Rightarrow \omega \leq \omega''$ (transitivity). We assume that the set Ω is finite, where $\perp \in \Omega$ and $\perp \leq \omega, \forall \omega \in \Omega$. To be concise, we use Ω^+ to denote $\Omega \setminus \{\perp\}$. Then a discrete probability distribution can be treated as a partial order structure [85]. We first begin our formulation with a given interval $[a,b]$, let the functions $\mathbf{f}, \mathbf{g} \in \mathbb{R}^{|[a,b]|}$ be defined as follows,

$$\mathbf{f} = \begin{bmatrix} f(a,a) \\ \vdots \\ f(a,b) \end{bmatrix}, \qquad \mathbf{g} = \begin{bmatrix} g(a,a) \\ \vdots \\ g(b,a) \end{bmatrix}.$$

Then the addition and scalar multiplication are defined as follows,

$$\begin{aligned} (f+g)(a,b) &= f(a,b) + g(a,b), \\ (f*g)(a,b) &= \sum_{\substack{\omega \in \Omega \\ a \leq \omega \leq b}} f(a,\omega)\, g(\omega,b) = \mathbf{f}^T \mathbf{g}. \end{aligned}$$

Where ω is an entry in space Ω.

2.3.2 Möbius Inversion Formula

Incident algebra is used to formulate the *Möbius inversion formula* which provides a natural construction for a summation over an arbitrary locally finite partially ordered set. It is defined as,

$$\begin{aligned} g(\omega) &= \sum_{\substack{\omega' \in \Omega \\ \omega' \leq \omega}} \zeta(\omega', \omega) f(\omega') = \sum_{\substack{\omega' \in \Omega \\ \omega' \leq \omega}} f(\omega') \\ f(\omega) &= \sum_{\substack{\omega' \in \Omega \\ \omega' \leq \omega}} \mu(\omega', \omega) g(\omega') \end{aligned}$$

where the special elements *zeta function* $\zeta : \Omega \times \Omega \to \{0,1\}$ represents the integral function and is defined as,

$$\zeta(\omega', \omega) = \begin{cases} 1 & \text{if } \omega' \leq \omega, \\ 0 & \text{otherwise.} \end{cases}$$

and the *Möbius function* $\mu : \Omega \times \Omega \to \mathbb{Z}$ represents derivative and is convolutional inverse of the *zeta function* $\zeta\mu = \delta$ defined as,

$$\mu\left(\omega, \omega'\right) = \begin{cases} 1 & \text{if } \omega = \omega', \\ -\sum\limits_{\substack{\omega'' \in \Omega \\ \omega \leq \omega'' \leq \omega'}} \mu\left(\omega, \omega''\right) & \text{if } \omega < \omega', \\ 0 & \text{otherwise.} \end{cases}$$

In the following section we will apply the Möbius information formula to define the relationship between the natural parameter θ, probability p and the expectation parameter η.

2.4 Log-Linear Model on a Poset

We introduce the Log-linear model [2] proposed by Sugiyama et al. [97, 99]. by defining the log probability with respect to the parameters θ and the expectation η.

$$\log p\left(\omega\right) = \sum_{\omega' \in \Omega} \zeta\left(\omega', \omega\right) \theta\left(\omega'\right) = \sum_{\substack{\omega' \in \Omega \\ \omega' \leq \omega}} \theta\left(\omega'\right),$$
$$\eta\left(\omega\right) = \sum_{\omega' \in \Omega} \zeta\left(\omega, \omega'\right) p\left(\omega'\right) = \sum_{\substack{\omega' \in \Omega \\ \omega' \geq \omega}} p\left(\omega'\right).$$

Then from the *Möbius inversion formula* we have,

$$p\left(\omega\right) = \sum_{\omega' \in \Omega} \mu\left(\omega, \omega'\right) \eta\left(\omega'\right),$$
$$\theta\left(\omega\right) = \sum_{\omega' \in \Omega} \mu\left(\omega', \omega\right) \log p\left(\omega'\right).$$

We can see a clear intuition behind this formulation of the log-linear model. Naturally we can see if we would like to compute the expectation, we need to integrate the probability distribution and if we would like to obtain the parameters of the model we would like the take the derivative of the probability distribution which is represented by the zeta and the möbius function respectively.

These equations can be seen as the generalization of the *Log-linear model* that gives the probability $p(x)$ of an n-dimensional binary vector $\mathbf{x} = (x^{(1)}, x^{(2)}, \ldots, x^{(n)}) \in \{0, 1\}^n$ in Equation (2.1).

In the following section we will make the connection between the formulation of the log-linear model and information geometry. We will first show that the log-linear model is a dually flat structure and is a Riemannian structure induced by θ and η.

2.4.1 Dually Flat Structure

A dually flat manifold is a generalization of a Euclidean space, inheriting useful properties from it such as the Pythagorean theorem and coordinate systems. Here we present the proof presented in Sugiyama et al. [99] that the log-linear model is a dually flat Riemannian structure. To show this, we introduce a manifold Ω that is a dually flat Riemannian structure induced by two functions θ and η. We first define $\psi(\theta)$ as,

$$\psi(\theta) = -\theta(\perp) = -\log p(\perp)$$

which corresponds to the normalizer of p. It is a convex function because,

$$\psi(\theta) = \log \sum_{\omega \in \Omega} \exp \left[\sum_{\substack{\omega' \in \Omega \\ \perp \leq \omega' \leq \omega}} \theta(\omega') \right]. \tag{2.12}$$

The Legendre transformation of $\psi(\theta)$ is given as,

$$\varphi(\eta) = \max_{\theta'} \left[\theta' \eta - \psi(\theta') \right], \qquad \theta' \eta = \sum_{\omega \in \Omega^+} \theta'(\omega) \eta(\omega)$$

Then $\varphi(\eta)$ coincides with negative entropy.

Theorem 1 (Legendre Dual [99])**.** *The function*

$$\varphi(\eta) = \sum_{\omega \in \Omega} p(\omega) \log p(\omega). \tag{2.13}$$

is differentiable and a one-to-one transformation. This means φ is used as another coordinate system on the manifold M and it is connected to ψ by a Legendre transformation.

Since they are connected with each other by the Legendre transformation, they form a dual coordinate system $\nabla\psi(\theta)$ and $\nabla\varphi(\eta)$ of $\mathbb{S}$, which coincides with θ and η as follows.

Theorem 2 (Dual Coordinate System [99])**.** *A dual coordinate system can be defined by*

$$\nabla\psi(\theta) = \eta, \qquad \nabla\varphi(\eta) = \theta.$$

where the two coordinate systems are connected by the Legendre transformation.

The dually flat structure shows that the optimization problem can be represent the exponential family using two convex functions $\psi(\theta)$ and $\varphi(\eta)$, which represents the partition function and negative entropy respectively. These two functions are connected via the Legendre transformation. The two functions have a dualistic structure meaning that the parameters are defined locally instead of globally making it much more efficient for optimization. In the following section we will show that the log-linear model is a Riemannian structure which allows us to take advantage of many different optimization techniques.

2.4.2 Riemannian Structure

A metric tensor is a function which takes a pair of tangent vectors at a point on a surface of a manifold and produces a real number scalar that has properties of a dot product of vectors in Euclidean space. We present the proof that the proof shown by [99], that the Möbius inversion formula enables us to compute the Riemannian metric of Ω.

Theorem 3 (Riemannian Metric [99])**.** *The manifolid $(\Omega, G(\xi))$ is a Riemannian manifold with the Riemannian metric $G(\xi)$ such that for all $\omega, \omega', \in \Omega^+$,*

$$G_{\omega\omega'}(\xi) = \begin{cases} \sum_{\omega'' \in \Omega} \zeta(\omega, \omega'') \zeta(\omega', \omega'') p(\omega'') - \eta(\omega) \eta(\omega') & \text{if } \xi = \theta, \\ \sum_{\omega'' \in \Omega} \mu(\omega'', \omega) \mu(\omega'', \omega') p(\omega'')^{-2} & \text{if } \xi = \eta. \end{cases}$$

An affine connection ∇ is a differential operator defined on a manifold to allow us to define covaiant dervaitive of vector fields, parallel transpot of vectors on tagent planes along a smooth surve and geodesics. The Levi-Civita connection is a special case of the affine connection that is defined on a Riemannian manifold that has properties that are torsion free and parallel transport is an isometry (the inner product between tangent vectors are preserved).

Theorem 4 (Riemannian Connection [99])**.** *The Riemannian connection $\Gamma(\xi)$ on the manifold $(\Omega, g(\xi))$ is given in the following for all $\omega, \omega', \omega'' \in \Omega^+$,*

$$\Gamma_{\omega\omega'\omega''}(\xi) =$$
$$\begin{cases} \frac{1}{2} \sum_{\omega''' \in \Omega} [\zeta(\omega, \omega''') - \eta(\omega))(\zeta(\omega'', \omega''') - \eta(\omega'))(\zeta(\omega'', \omega''') - \eta(\omega'')] p(\omega'''), & \text{if } \xi = \theta, \\ -\frac{1}{2} \sum_{\omega''' \in \Omega} \mu(\omega''', \omega) \mu(\omega''', \omega') \mu(\omega''', \omega'') p(\omega''')^{-2}, & \text{if } \xi = \eta. \end{cases}$$

The Riemannian structure is a smooth manifold where the inner product is defined at each point. By projecting discrete structures onto a Riemannian structure it allows us to take advantage of continuous optimization techniques such as gradient descent, co-ordinate descent and natural gradient. In the following section, we will explain how optimization can be performed using the projection algorithm.

2.4.3 The Projection Algorithm

The projection algorithm is an iterative algorithm used to estimate the parameters in the model. The Bregman divergence is the canonical divergence that measures the difference between the two distribution P and Q on a dually flat manifold defined as,

$$\mathrm{D}[P,Q] = \psi(\theta_P) + \varphi(\eta_Q) - \theta_P \eta_Q$$

Substituting the equations from Theorem 1 $\varphi(\eta_Q) = \sum_{x \in S} q(\omega) \log q(\omega)$ and $\theta_P \eta_Q - \psi(\theta_P) = \sum_{x \in S} q(\omega) \log p(\omega)$ we get,

$$\mathrm{D}[P,Q] = \sum_{\omega \in \Omega} q(\omega) \log \frac{q(\omega)}{p(\omega)},$$

which coincides with the reverse Kullback-Leibler (KL) divergence.

The parameters in the log-linear model and the poset structure can be estimated using the projection algorithm. We define this by first letting $\mathcal{S}(\beta)$ be a submanifold of Ω such that

$$\mathcal{S}(\beta) = \{ P \in \Omega | \theta_P(\omega) = \beta(\omega) \text{ for all } \omega \in \mathrm{dom}(\beta) \},$$

specified by a function β with $\mathrm{dom}(\beta) \subseteq \Omega^+$. The projection of $P \in \Omega$ onto $\mathcal{S}(\beta)$ is called the m-projection which is defined as the distribution $P_\beta \in \mathcal{S}(\beta)$ as

$$\begin{cases} \theta_{P_\beta}(\omega) = \beta(\omega) & \text{if } \omega \in \mathrm{dom}(\beta), \\ \eta_{P_\beta}(\omega) = \eta_P(\omega) & \text{if } \omega \in \Omega^+ \setminus \mathrm{dom}(\beta), \end{cases}$$

is the minimizer of the KL divergence from P to $\mathcal{S}(\beta)$,

$$P_\beta = \underset{Q \in \mathcal{S}(\beta)}{\operatorname{argmin}} D_{\mathrm{KL}}(P,Q)$$

We can switch η and θ in the submanifold $\mathcal{S}(\beta)$ to obtain the *e*-projection

$$\begin{cases} \theta_{P_\beta}(\omega) = \theta_P(\omega) & \text{if } \omega \in \Omega^+ \setminus \operatorname{dom}(\beta), \\ \eta_{P_\beta}(\omega) = \beta(\omega) & \text{if } \omega \in \operatorname{dom}(\beta). \end{cases}$$

By alternating between the *e*-projection and *m*-projection, we are able to compute the gradient required to minimize the KL-divergence to estimate the parameters p, θ and η in the model.

2.5 Summary

In this chapter we presented the preliminary material for this thesis. We first showed how the representational power can be increased in unsupervised learning models. We then gave an overview of the unsupervised learning models studied in this thesis. We have also reviewed the information geometric formulation of the exponential family and how it can be extended to efficiently represent the inclusion-exclusion principle using properties in information geometry. The remainder of the thesis is structured as follows, Chapter 3 formulates our proposed approach using information geometry to increase the representational power in the Boltzmann Machine. We then compare our approach to increasing the representational power by using hidden nodes. We then explore how we can using our formulation to include higher-order terms in blind source separation (BSS) in Chapter 4. Chapter 5 shows how higher-order terms can be used to approximate high dimensional functions in Poisson processes. Finally 6 explored how we could use Bayesian non-parametric techniques to automatically select the terms to include into the model.

CHAPTER 3

Increasing Representational Power with Higher Order Feature Interactions

This Chapter investigates the differences in using hidden nodes and higher-order feature interactions to increase the representational power of machine learning models. We apply our approach to the Boltzmann machine as a representative example to compare the two approaches. In this Chapter we outline the formulation of the higher-order feature interactions. Chapter 4 and Chapter 5 uses concepts in this Chapter to include higher-order feature interactions in Blind Source Separation (BSS) and to an efficient estimation of higher-order functions in a multivariate Poisson process respectively.

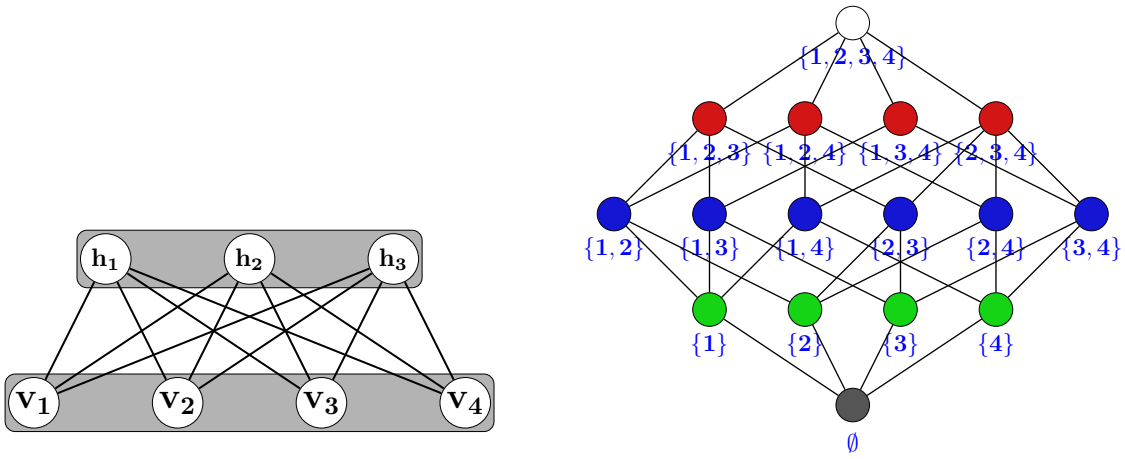
This Chapter is organized as follows: We first present an introduction on using higher-order feature interactions and hidden nodes in hierarchical models and the Boltzmann machine. We then provide the preliminaries for the Boltzmann machine, restricted Boltzmann machine (RBM) and the information geometric formulation of the binary log-linear model. The binary log-linear model can be represented visually as a partially ordered set (poset). We then present our proposed inference algorithm for the Higher-Order Boltzmann Machine which uses a combination of Gibbs sampling and Annealed Importance Sampling (AIS) in inference to overcome the computational and numerical problems encountered in training. Finally we present an empirical study to compare the different characteristics in using higher-order feature interactions and hidden nodes to increase the representational power of machine learning models.

3.1 Introduction to Hierarchical Machine Learning Models

Hierarchical machine learning models can be used to identify higher-order feature interactions. They include a wide range of models used in machine learning such as graphical models and deep learning because they can easily be generalized for many different applications. Hierarchical models are widely used as they can use a large number of parameters to create a high representative power for modeling interactions between features. However, turning towards the optimal model includes a classical machine learning problem known as the *bias-variance trade-off* [32]. Despite the prevalence of hierarchical models, the bias-variance trade-off for higher-order feature interactions have not been well studied.

In this Chapter, we study the differences in using *hidden layers* and *higher-order interactions* to achieve higher representation power in hierarchical models. In our study, we focus on the Boltzmann Machine (BM) [1], one of the fundamental machine learning models. The family of BMs has been used in a wide range of machine learning models including graphical models and deep learning. The Restricted Boltzmann Machine (RBM) [40] (Figure 3.1a) and the Higher-Order Boltzmann Machine (HBM) [94, 69] (Figure 3.1b) are fundamental models for learning higher-order feature interactions. However, the two models have different methods for achieving a higher representation power. The RBM is represented by a bipartite graph consisting of two groups, the "visible layer" and the "hidden layer". The visible layer represents the direct observations of the data, while the hidden layer is used to identify latent features. The bias and the edge weights between the visible nodes and hidden nodes are tuned to model the interaction between the features. The HBM is represented using a finite partially ordered set (poset) using a Hasse diagram [28, 36]. The poset is used to represent the outcome space of the HBM. A weight is placed on each of the nodes in the outcome space to model the higher-order feature interactions. From a theoretical perspective, both the RBM and HBM are capable models in modeling higher-order feature interactions [56].

In our study, we empirically perform a *bias-variance decomposition* for both the RBM and HBM to study the total errors of each model from the trade-off between bias and variance. For our study, we use Contrastive Divergence (CD) as the inferencing technique for the RBM. For the HBM, we use the recent information geometric formulation of the HBM by Sugiyama *et al.* [98, 99] presented in Chapter 2 and use gradient descent to maximize the likelihood. The generalized Pythagorean theorem from information geometry enables us to decompose the total error represented as the Kullback–Leibler (KL) divergence into bias and variance terms. Our analysis uses a synthetic dataset with varying features, samples and model complexity. Our contribution includes: 1) A proposal to use a combination of Gibbs sampling and Annealed Importance Sampling (AIS) in inference to overcome the



(a) (b)

Fig. 3.1 Example of Boltzmann machine modeling high-feature interactions. (a) A Restricted Boltzmann machine with a configuration of 4 visible nodes and 3 hidden nodes. (b) The outcome space of the Higher-Order Boltzmann machine with 4 visible nodes. The green, blue, red and white nodes show first, second, third and fourth order interactions respectively. The bottom node (black) is used to normalize the Boltzmann machine.

computational and numerical problems in training the HBM. 2) A study which compares the bias-variance trade-off in *hidden layers* and *higher-order interactions*.

Our results have shown that using *hidden layers* and *higher-order interactions* produce a similar error from the bias and *higher-order interactions* produce less variance for a smaller sample size. For larger datasets, the error from the bias is more dominant, therefore for sufficiently large datasets, *hidden layers* and *higher-order interactions* have a comparable error with a similar order of magnitude.

3.2 Preliminary Models

This section presents the hierarchical probabilistic models to analyze the error in modeling higher-order feature interactions. We first introduce the generic Boltzmann machine along with the Restricted Boltzmann Machine (RBM). We then present the information geometry formulation of the log-linear model of hierarchical probability distribution, which includes the family of Boltzmann machines.

3.2.1 Restricted Boltzmann Machine

Recall in section 2.1.1, we have introduced the Boltzmann Machine. Here we introduce the Restricted Boltzmann Machine (RBM) which is a class of BM in which the vertices form a bipartite graph. The two groups of nodes are known as the "visible layer" and the "hidden layer". The visible layer represents the direct observations on the dataset. An activation function is placed on the hidden nodes to model the interactions between features by the following equation:

$$p\left(h_j = 1 \middle| \mathbf{v}\right) = \sigma\left(b_j + \sum_i v_i w_{ij}\right),$$

where σ is the sigmoid function. We apply the efficient contrastive divergence technique to adjust the weights and biases to maximize the product of probabilities of generating a given dataset by the BM [39, 106]. The updates for the weights and the biases is given by:

$$\Delta\mathbf{W} = \varepsilon\left(\mathbf{v}\mathbf{h}^T - \mathbf{v}'\mathbf{h}'^T\right), \qquad \Delta\mathbf{b} = \varepsilon\left(\mathbf{x} - \mathbf{x}'\right).$$

Where $\mathbf{v}'$, $\mathbf{h}'$, $\mathbf{x}'$ represents the samples reconstructed from the model and ε represents the learning rate.

Additional hidden layers can be added to create the Deep Boltzmann Machine (DBM) to increase the representation power [87, 88]. However, this is not analyzed in our study due to the exponential increase in complexity to compute the partition function in Equation (**??**).

3.2.2 Information Geometric Formulation of the Log-Linear Model

The information geometric log-linear probabilistic model has been introduced by Amari *et al.* [4, 73, 74]. Further advances in the log-linear formulation by Sugiyama *et al.* [98] has enabled to analytically compute the Fisher information of parameters in hierarchical models. This formulation of the hierarchical model uses a partial order structure to represent the possible model outcomes.

Here we introduce the log-linear formulation introduced in Sugiyama *et al.* [98]. Let $(\Omega, \leq)$ be a *partially ordered set* (*poset*) [36], where a *partial order* $\leq$ is the relation between elements in a set S. The *poset* must satisfy the following three properties for $\omega, \omega', \omega'' \in \Omega$: (1) $\omega \leq \omega$ (reflexivity), (2) $\omega \leq \omega', \omega' \leq \omega \Rightarrow \omega = \omega'$ (anti-symmetry), and (3) $\omega \leq \omega', \omega' \leq \omega'' \Rightarrow \omega \leq \omega''$ (transitivity). We assume that the set Ω is finite, where $\perp \in \Omega$ and $\perp \leq \omega, \forall\omega \in \Omega$. To be concise, we use Ω^+ to denote $\Omega \setminus \{\perp\}$.

The *zeta function* $\zeta : \Omega \times \Omega \to \{0,1\}$ and the *Möbius function* $\mu : \Omega \times \Omega \to \mathbb{Z}$ are two functions used to construct the partial order structure. The *zeta function* is defined as:

$$\zeta\left(\omega', \omega\right) = \begin{cases} 1 & \text{if } \omega' \leq \omega, \\ 0 & \text{otherwise.} \end{cases}$$

The *Möbius function* μ is defined to be the convolution inverse of the *zeta function*, i.e:

$$\mu\left(\omega, \omega'\right) = \begin{cases} 1 & \text{if } \omega = \omega', \\ -\sum_{\substack{\omega'' \in \Omega \\ \omega \leq \omega'' < \omega}} \mu\left(\omega, \omega''\right) & \text{if } \omega < \omega', \\ 0 & \text{otherwise.} \end{cases}$$

The *log-linear model* on Ω provides a mapping of the discrete probability distribution to the structured outcome space $(\Omega, \leq)$. Let the probability distribution P denote a probability distribution that assigns a probability $p(\omega)$ for each $\omega \in \Omega$ while satisfying $\sum_{\omega \in \Omega} p(\omega) = 1$. Each probability $p(\omega)$ for $\omega \in \Omega$ is defined as:

$$\begin{aligned} \log p(\omega) &= \sum_{\omega' \in \Omega} \zeta\left(\omega', \omega\right) \theta(\omega) = \sum_{\omega' \leq \omega} \theta(\omega), \\ \theta(\omega) &= \sum_{\omega' \in \Omega} \mu\left(\omega', \omega\right) \log p\left(\omega'\right). \end{aligned} \tag{3.1}$$

$$\begin{aligned} \eta(\omega) &= \sum_{\omega' \in \Omega} \zeta\left(\omega, \omega'\right) p\left(\omega'\right), \\ p(\omega) &= \sum_{\omega' \in \Omega} \mu\left(\omega, \omega'\right) \eta\left(\omega'\right). \end{aligned} \tag{3.2}$$

Sugiyama *et al.* [99] has shown that the set of distributions $\Omega = \{P \mid 0 < p(\omega) < 1 \text{ and } \sum p(\omega) = 1\}$ always become a *dually flat Riemannian manifold*. This makes the two functions θ and η a dual coordinate system on Ω which is connected through the Legendre transformation.

3.3 Proposed Model

This section details the information geometric formulation of the Higher-Order Boltzmann Machine (HBM) and our proposed inference algorithm.

3.3.1 Higher-Order Boltzmann Machine

Higher order interactions in a Boltzmann machine are capable of modeling higher-order feature interactions. However, they are very rarely used in practice due to the high computational cost for inferencing and learning. The log-linear formulation of the Boltzmann machine provides an elegant representation of the outcome space. This formulation allows any parameters to be included or removed from Ω^+. For a given Boltzmann machine $S(B) = 2^V$ with $V = \{1, 2, \ldots, n\}$, the energy function of the kth order Boltzmann machine is defined as:

$$\Phi(\mathbf{x}; \mathbf{b}, \mathbf{w}) = -\sum_{i_1 \in V} b_{i_1} x_{i_1} - \sum_{i_1, i_2 \in V} w_{i_1 i_2} x_{i_1} x_{i_2} - \sum_{i_1, i_2, i_3 \in V} w_{i_1 i_2 i_3} x_{i_1} x_{i_2} x_{i_3} \tag{3.3}$$

$$- \cdots - \sum_{i_1, i_2, \ldots, i_k \in V} w_{i_1, i_2, \ldots, i_k} x_{i_1} x_{i_2} \ldots x_{i_k}, \tag{3.4}$$

Sugiyama *et al.* [98, 99] have shown that the log-linear model can be used to represent the family of Boltzmann Machines. A *submanifold* of Ω can be used to represent the set of Gibbs distribution of the BM B given by

$$\mathcal{S}(B) = \{P \in \Omega \mid \theta(\omega) = 0, \forall x \notin B\}. \tag{3.5}$$

The Gibbs distribution in Equation (**??**) directly corresponds to the log-linear model in Equation (3.1) by:

$$\begin{aligned} \log p(\omega) &= \sum_{\omega' \in B} \zeta(\omega', \omega)\, \theta(\omega') - \psi(\theta), \\ \psi(\theta) &= -\theta(\perp) = \log Z, \end{aligned} \tag{3.6}$$

where magnitude of $\theta(\omega)$ corresponds to the model parameters which model the order of interactions in the Boltzmann machine

$$B = \left\{\omega \in \Omega^+ \mid |x| = 1 \text{ or } x \in E\right\}, \tag{3.7}$$

that is; $\theta(\omega) = b_i$ if $|x| = 1$ and $\theta(\omega) = w_{x_{ij}}$ if $|x| = 2$. The log-linear formulation of the Boltzmann machine shown in Equation (3.6) can be extended to be a kth order Boltzmann machine by $B = \{\omega \in \Omega^+ \mid |x| \leq k\}$.

3.3.2 Inference Algorithm

The log-linear formulation of the Boltzmann machine can be trained by minimizing the KL (Kullback-Leibler) divergence to approximate a given empirical distribution $\hat{P}$:

$$\min_{P_B \in \mathcal{S}(B)} D_{\mathrm{KL}}\left(\hat{P}, P_B\right) = \min_{P_B \in \mathcal{S}(B)} \sum_{P_B \in \mathcal{S}(B)} \hat{p}\left(\omega\right) \log \frac{\hat{p}\left(\omega\right)}{p_B\left(\omega\right)}. \tag{3.8}$$

This is equivalent to maximizing the log-likelihood $L\left(P_B\right) = N \sum_{\omega \in \Omega} \hat{p}\left(\omega\right) \log p_B\left(\omega\right)$. The gradient is obtained as,

$$\begin{aligned}
\frac{\partial}{\partial \theta_B\left(\omega\right)} D_{\mathrm{KL}}\left(\hat{P}, P_B\right) &= \frac{\partial}{\partial \theta_B\left(\omega\right)} \sum_{\omega' \in \Omega} \hat{p}\left(\omega'\right) \log p_B\left(\omega'\right) \\
&= \frac{\partial}{\partial \theta_B\left(\omega\right)} \sum_{\omega' \in \Omega} \left(\hat{p}\left(\omega'\right) \sum_{\perp < \omega'' \leq \omega'} \theta_B\left(\omega''\right) \right) - \frac{\partial}{\partial \theta_B\left(\omega\right)} \psi\left(\theta_B\right) \sum_{\omega' \in \Omega} \hat{p}\left(\omega'\right) \\
&= \hat{\eta}\left(\omega\right) - \eta_B\left(\omega\right).
\end{aligned}$$

However, $\eta_B\left(\omega\right)$ is computationally expensive to compute because it requires to compute all values of P_B. We propose to use a combination of Gibbs sampling and Annealed Important Sampling (AIS) [76, 86] to approximate the distribution of P_B.

3.3.3 Gibbs sampling for η_B

Gibbs sampling [35] is a Markov Chain Monte Carlo (MCMC) algorithm which approximates a multivariate probability distribution. It fixes all the other model parameters and updates each of the model parameters one-by-one until convergence. MCMC can be applied directly to estimate the parameters θ and η. But in our approach we will use gradient to guide our optimisation. The equation for each MCMC step is given by,

$$P\left(\mathbf{x}_i = 1 | \mathbf{x}_{-i}; \theta\right) = \frac{P\left(\mathbf{x}; \theta\right)}{P\left(\mathbf{x}_{-i}; \theta\right)} \propto P\left(\mathbf{x}; \theta\right),$$

where $\mathbf{x}_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, , x_n)$. We apply Gibbs sampling to generate samples for $\mathbf{x} = (x_1, \ldots, x_n)$. By generating samples, we are able to approximate the un-normalized distribution denoted as f^*. The un-normalized probability distribution is proportional to the normalized distribution by a constant Z, i.e. $P = \frac{1}{Z} f^* \propto f^*$. We will later provide a solution using AIS to approximate the normalization constant Z. To update each $\mathbf{x}_i$, we use

the difference between the energy functions in each node.

$$\begin{aligned}\Delta\Phi_i(\mathbf{x};\theta) &= \Phi_i(\mathbf{x}_{x_i=0};\theta) - \Phi_i(\mathbf{x}_{x_i=1};\theta) \\ &= -C\log(P(\mathbf{x}_{x_i=0};\theta)) - (-C\log(P(\mathbf{x}_{x_i=1};\theta))) \\ &= C\log(P(\mathbf{x}_{x_i=1};\theta) - C\log(1-P(\mathbf{x}_{x_i=1};\theta))),\end{aligned}$$

where C represents the constant in the Boltzmann distribution. Rearranging the equation to solve for $P(\mathbf{x}_{x_i=1};\theta)$, we have

$$\begin{aligned}\exp\left(-\frac{\Delta\Phi_i(\mathbf{x};\theta)}{C}\right) &= \frac{1-P(\mathbf{x}_{x_i=1};\theta)}{P(\mathbf{x}_{x_i=1};\theta)}, \\ P(\mathbf{x}_{x_i=1};\theta) &= \frac{\exp(\Delta\Phi_i(\mathbf{x};\theta)/C)}{1+\exp(\Delta\Phi_i(\mathbf{x};\theta)/C)}.\end{aligned}$$

The term with the change in energy is calculated by:

$$\begin{aligned}\exp\left(\frac{\Delta\Phi_i(\mathbf{x};\theta)}{C}\right) &= \exp\left(\frac{1}{C}\left[\Phi_i(\mathbf{x}_{\mathbf{x}_i=0};\theta) - \Phi_i(\mathbf{x}_{\mathbf{x}_i=1};\theta)\right]\right) \\ &= \exp\left(-\log P(\mathbf{x}_{\mathbf{x}_i=0};\theta) - (-\log P(\mathbf{x}_{\mathbf{x}_i=1};\theta))\right) \\ &= \exp\left(\sum_{s\in S}\zeta(s,\mathbf{x}_{\mathbf{x}_i=1})\,\theta(s) - \sum_{s\in S}\zeta(s,\mathbf{x}_{\mathbf{x}_i=0})\,\theta(s)\right).\end{aligned}$$

A set of M samples can be generated to approximate η_B by using Equation (3.2) by empirically estimating the distribution of P_B^*. The overall run-time for each interaction for the Gibbs sampling step is $\mathcal{O}\left(M\left|\mathcal{S}(B)\right|^2\right)$.

3.3.4 Annealed Importance Sampling to Approximate P_B

We propose to use AIS [76, 86] to overcome the numerical problems in computing the normalization parameter in the partition function Z. By inspecting Equation (**??**), we can identify a number of problems in computing Z. Firstly, it is clear that the value of Z is extremely large because it takes the sum of the exponential of all energy functions. The large value of Z often creates numerical problems for most implementation. Secondly, computing the energy function $\Phi(x)$ for all nodes is extremely computationally expensive. AIS provides a solution to approximate the value of $\log(Z)$ without having to compute Z or evaluate the energy function $\Phi(x)$.

AIS approximates the normalization constant by tracking the gradual changes of an MCMC transition $T_k(\mathbf{x}_{n+1}|\mathbf{x}_n)$ operation such as Gibbs sampling. AIS uses a sequence of intermediate probability distributions to evaluate the importance weight $w_{\text{AIS}}^{(i)}$ which is an estimation of the ratio between the first and last distribution.

For our study, we use one of the most prevalent methods to generate a sequence of intermediate probability distributions for $k = 0, \ldots, K$ by using the following geometric property,

$$f_k(x) \propto f_0^*(x)^{1-\beta_k} f_k^*(x)^{\beta_k}$$

where $0 = \beta_0 < \beta_1 < \ldots < \beta_K = 1$. There have been several other more advanced techniques to model the path of the intermediate distributions [37]. However, this is not the focus of our study and can be subjected to further study in future work. The AIS weight w_{AIS} can be calculated by using

$$w_{\text{AIS}}^{(i)} = \frac{f_1^*(x_1)}{f_0^*(x_1)} \frac{f_2^*(x_2)}{f_1^*(x_2)} \cdots \frac{f_{K-1}^*(x_{K-1})}{f_{K-2}^*(x_{K-1})} \frac{f_K^*(x_K)}{f_{K-1}^*(x_K)},$$

where f^* denotes a function to calculate the un-normalized probability distribution. After completing M runs of AIS, the ratio of the first and final constant of the partition function can be estimated as

$$\frac{Z_K}{Z_0} \approx \frac{1}{M} \sum_{i=1}^{M} w_{\text{AIS}}^{(i)} = \hat{r}_{\text{AIS}} \tag{3.9}$$

Neal [76, 77] has theoretically shown that the $\text{Var}(\hat{r}_{\text{AIS}}) \propto 1/MK$. For practical implementations Equation (3.9) should be in log scale to avoid numerical problems. The standard form is shown here for conciseness. From Equation (3.9), the final $\log Z$ can be estimated without computing Z if Z_0 is known. Then $\log Z_0$ can be calculated efficiently if we initialize P_B uniformly, i.e. for HBM, $\theta(\perp) = -\log Z = N \log(2)$ and $\theta(\omega) = 0$, $\forall x \in \Omega^+$.

3.4 Experiments

Here we present the main results of this Chapter. This section presents the formulation of the *bias-variance decomposition*, the set-up of the experiment and the experimental results and discussion.

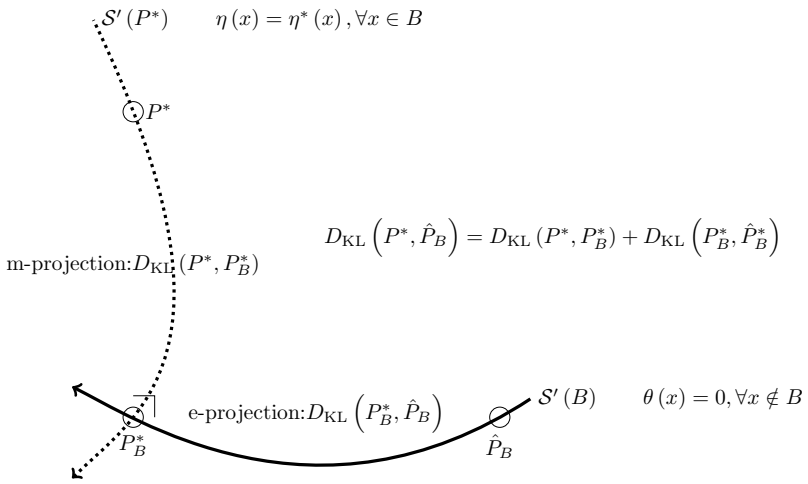
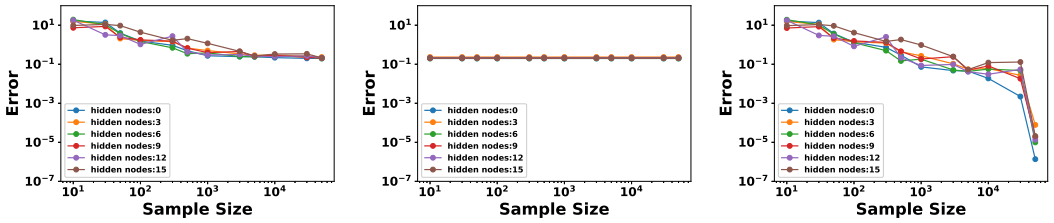


Fig. 3.2 An illustration of the decomposition of the bias and variance.



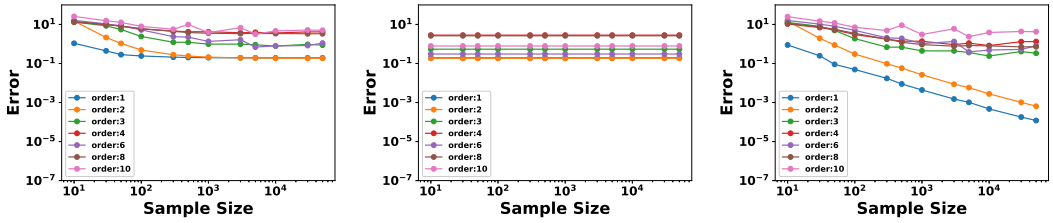
(a) Total Error (b) Bias (c) Variance

Fig. 3.3 Empirical evaluation of the error generated from the bias and variance of the RBM

3.4.1 Bias-Variance Decomposition

We use a *bias-variance decomposition* to compare the behavior of the higher-order feature interactions between the RBM and HBM by varying the complexity of the model. We focus on the expectation of the KL divergence $\mathbf{E}[D_{\mathrm{KL}}(P^*, \hat{P}_B)]$ from the true (unknown) distribution P^* to the maximum likelihood estimation (MLE) $\hat{P}_B$ of the empirical distribution $\hat{P}$ by a Boltzmann machine with the parameter set B. This term represents the total error in the model accumulated from the bias and variance in the model.

The KL divergence for probabilities in the exponential family can be decomposed into bias and variance using its information geometric properties. We can calculate the true variance of the model by replacing $\hat{P}$ with P^* in Equation (3.8). The bias and variance can be



(a) Total Error (b) Bias (c) Variance

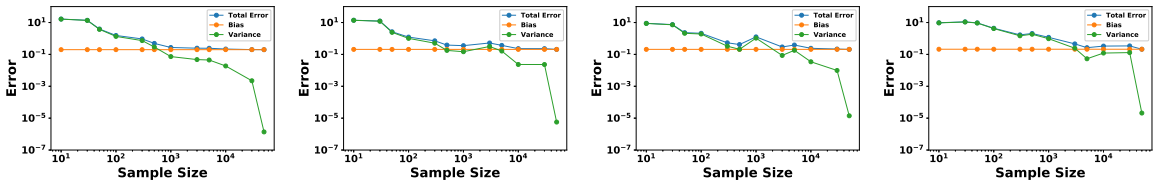
Fig. 3.4 Empirical evaluation of the error generated from the bias and variance for the HBM

separated into two components which are orthogonal as illustrated in Figure 3.2. Using this decomposition of the bias and variance, the total error in the model can be calculated using the *Generalized Pythagorean Theorem* [6],

$$\begin{aligned}\mathbf{E}\left[D_{\mathrm{KL}}\left(P^*,\hat{P}_B\right)\right] &= \mathbf{E}\left[D_{\mathrm{KL}}\left(P^*,P_B^*\right)\right]+\mathbf{E}\left[D_{\mathrm{KL}}\left(P_B^*,\hat{P}_B\right)\right]\\ &= D_{\mathrm{KL}}\left(P^*,P_B^*\right)+\mathbf{E}\left[D_{\mathrm{KL}}\left(P_B^*,\hat{P}_B\right)\right]\\ &= \underbrace{D_{\mathrm{KL}}\left(P^*,P_B^*\right)}_{\text{bias}}+\underbrace{\mathrm{var}\left(P_B^*,B\right)}_{\text{variance}}.\end{aligned}$$

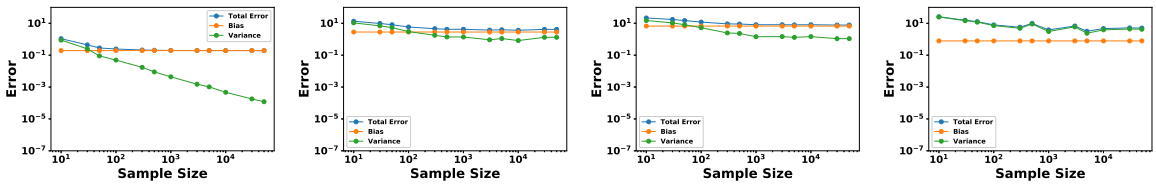
3.4.2 Experiment Setup

A synthetic dataset is generated to study the bias-variance trade-off in HBM and RBM. The synthetic data is created by drawing a random probability $[0,1]$ from a uniform distribution for each $P^* \in \mathcal{P}^*$ such that $\sum_{P^*\in\mathcal{P}^*} P^* = 1$, where $\mathcal{P}^*$ represents the set of probabilities for all possible feature combinations. A sample size of $N = \{\ 1\times 10,\ 3\times 10,\ 5\times 10,\ 1\times 10^2,\ 3\times 10^2,\ 5\times 10^2,\ 1\times 10^3,\ 3\times 10^3,\ 5\times 10^3,\ 1\times 10^4,\ 3\times 10^4,\ 5\times 10^4\ \}$ are drawn from the descrete probability distribution $\mathcal{P}^*$ using a multinomial. For each sample size N, we create 24 independent datasets to be used for both the HBM and RBM. The MLE of the HBM is calculated analytically by directly placing the true probability distribution P^* into the model. While, for the RBM, it is not analytically tractable to calculate the MLE, it is instead approximated by placing a dataset several orders of magnitude larger than the experimental dataset, in our case, we have generated a dataset with the sample size of 1×10^6 and have assumed this to be the MLE. Both the HBM and the RBM has been run using $10,000$ Gibbs samples, a learning rate of 0.1 and $10,000$ iterations.



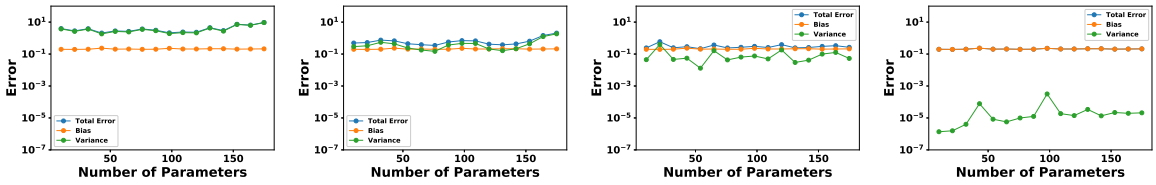
(a) 0 Hidden Nodes (b) 5 Hidden Nodes (c) 10 Hidden Nodes (d) 15 Hidden Nodes

Fig. 3.5 Empirical evaluation of the error generated from the bias and variance for varying hidden nodes in the RBM



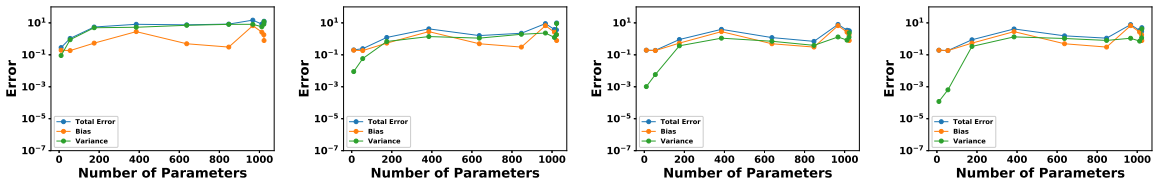
(a) Order 1 (b) Order 4 (c) Order 7 (d) Order 10

Fig. 3.6 Empirical evaluation of the error generated from the bias and variance for varying order of interactions in the HBM



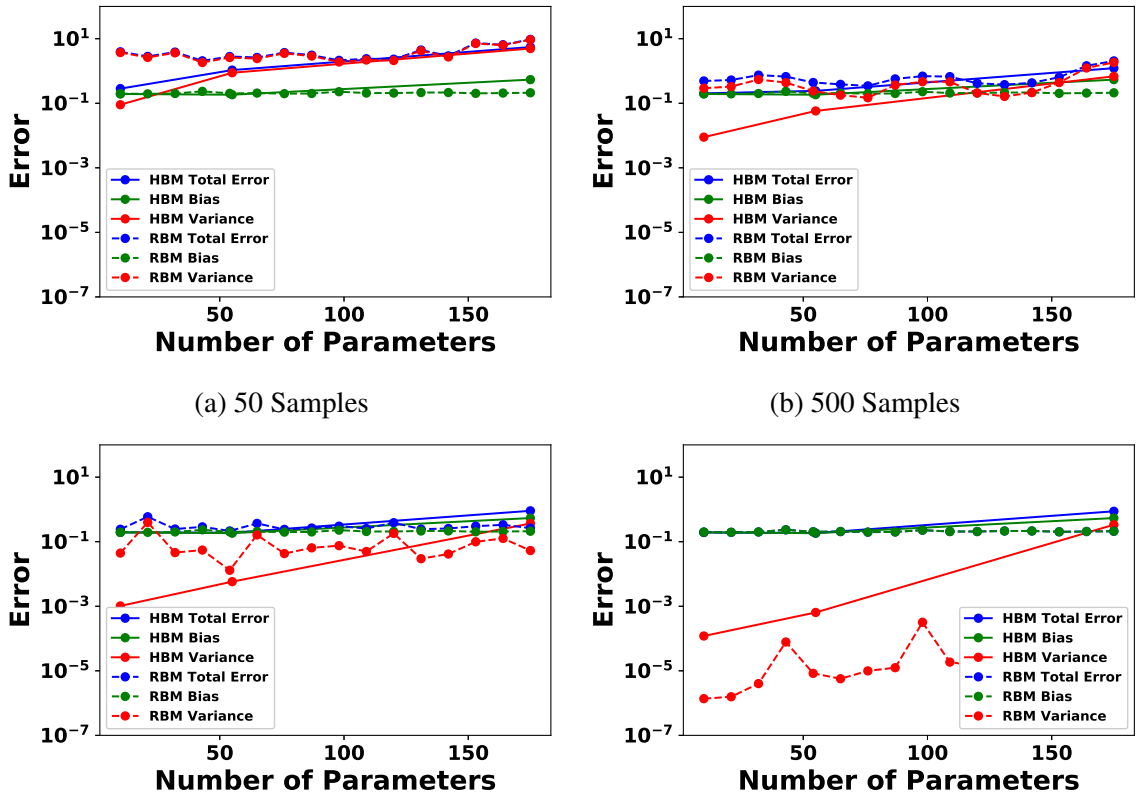
(a) 50 Samples (b) 500 Samples (c) 5000 Samples (d) 50000 Samples

Fig. 3.7 Empirical evaluation of the error generated from the bias and variance for varying sample size in the RBM



(a) 50 Samples (b) 500 Samples (c) 5000 Samples (d) 50000 Samples

Fig. 3.8 Empirical evaluation of the error generated from the bias and variance for varying sample size in the HBM



(a) 50 Samples

(b) 500 Samples

(c) 5000 Samples

(d) 50000 Samples

Fig. 3.9 Comparing empirical error in model for the HBM with RBM against the number of model parameters

3.4.3 Experiment Results

The empirical evaluation in Figure 3.3 and Figure 3.4 have shown that both RBM and HBM have shown similar trends, with a positive correlation between the number of model parameters and variance and an inverse relationship between the sample size and variance. Surprisingly, the bias does not show any clear correlation between the number of model parameters and the sample size.

The error generated from the variance is much more dominant for a smaller sample size and a larger number of model parameters. Comparing Figure 3.5 and Figure 3.6, the RBM has shown to be more effective at reducing the variance with a larger sample size. The importance sampling used to estimate the partition function in HBM may have led to the higher variance in empirical results for the HBM. Figure 3.7 and Figure 3.8 shows a positive correlation between the model parameters and variance. The HBM shows to have a larger correlation between the model parameters and variance.

The total error in the model is the sum of the bias and the variance. The total number of model parameters is a natural way to compare the total error generated by the RBM and HBM (i.e. $|\mathbf{b}| + |\mathbf{w}|$ for RBM and $|\mathcal{S}(B)|$ for HBM). Figure 3.9 shows that for small sample size, HBM has shown to have produced a lower error in the model. The higher error in the RBM is generated by the larger variance. For a larger sample size, the error from the bias is much more dominate. Since the bias in both the RBM and HBM is in a similar order of magnitude, both models have a total error in the same order of magnitude. RBM has shown to be much more effective at reducing the variance with a larger sample size, however, this does not reduce the total error significantly because it is several orders of magnitude smaller than the bias.

3.5 Summary

In this Chapter, we have first proposed using a combination of Gibbs sampling and importance sampling to overcome the computational issues in training the information geometric formulation of the higher-order Boltzmann machine (HBM). The experimental results have shown that our proposed approach is effective in estimating the probability distribution of the model. Our proposed approach has been compared with the RBM to compare using *hidden layers* and *higher-order interactions* to model higher-order feature interactions. Our experimental results have shown that both models have produced a total error with similar orders of magnitude and using *higher-order interactions* may be more effective at minimizing the variance for smaller sample size.

In the following Chapters, we use concepts that have been formulated in this Chapter to include higher-order feature interactions in Blind Source Separation (BSS) and to estimate higher-order functions in a multivariate Poisson process.

CHAPTER 4

Including Higher-Order Feature Interactions in Blind Source Separation

This Chapter presents the details of including higher-order feature interactions for a unsupervised learning problem known as Blind Source Separation (BSS). Our solution is to use the concepts presented in Chapter 3 to formulate a hierarchical probabilistic model for blind source separation (BSS) to increase the representational power of the model to capture the complex interactions. In our model, the source signal, received signal and the mixing signal are realized as different layers in our hierarchical sample space. The hierarchically structured sample space uses information geometry to provide theoretical guarantees to uniquely recover a set of source signals by minimizing the KL divergence from a set of mixed signals.

This Chapter is organized as follows: we first provide an introduction to Blind Source Separation (BSS), we then we outline our proposed information geometric approach to include higher-order feature interactions into Blind Source Separation (BSS). We then demonstrate our models capability with empirical experiments in image and signal separation.

4.1 Introduction to Blind Source Separation

The objective of *blind source separation* (BSS) is to identify a set of source signals from a set of multivariate mixed signals[1]. BSS is widely used for applications which are considered

[1]Mixed signals and received signals are used exchangeably throughout this Chapter.

to be the "cocktail party problem". Examples include image/signal processing [46], artifact removal in medical imaging [107], and electroencephalogram (EEG) signal separation [24].

Currently, there are a number of solutions for the BSS problem. The most widely used approaches are variations of principal component analysis (PCA) [81, 72] and independent component analysis (ICA) [23, 72]. However, they all have limitations with their approaches. PCA and its modern variations such as sparse PCA (SPCA) [114], non-linear PCA (NLPCA) [91], and Robust PCA [108] extract a specified number of components with the largest variance under an orthogonal constraint, which are composed of a linear combination of the variables. They create a set of uncorrelated orthogonal basis vectors that represent the source signal. The basis vectors with the N largest variance are called the principal components and is the output of the model. PCA has shown to be effective for many applications such as dimensionality reduction and feature extraction. However, for BSS, PCA makes the assumption that the source signals are orthogonal, which is often not the case in most practical applications.

Similarly, ICA also attempts to find the N components with the largest variance, but relaxes the orthogonality constraint. All variations of ICA such as infomax [12], FastICA [43] and JADE [21] separate a multivariate signal into additive subcomponents by maximizing statistical independence of each component. ICA assumes that each component is non-gaussian and the relationship between the source signal and the mixed signal is an affine transformation. In addition to these assumptions, ICA is sensitive to the initialization of the weights as the optimization is non-convex and is likely to converge to a local optimum.

Other potential methods which can perform BSS include non-negative matrix factorization (NMF) [57, 13], dictionary learning (DL) [79], and reconstruction ICA (RICA) [54]. NMF, DL and RICA are degenerate approaches to recover the source signal from the mixed signal. This means that these approaches are unable to restore the original mixed signal from the recovered signal. These approaches are more typically used for feature extraction. NMF factorizes a matrix into two matrices with nonnegative elements representing weights and features. The features extracted by NMF can be used to recover the source signal. More recently there are more advance techniques that uses Short-time Fourier transform (STFT) to transform the signal into the frequency domain to construct a spectrogram before applying NMF [89]. However, NMF does not maximize statistical independence which is required to completely separate the mixed signal into the source signal, and it is also sensitive to initialization as the optimization is non-convex. Due to the non-convexity, additional constraints or heuristics for weight initialization is often applied to NMF to achieve better results [30, 16]. DL can be thought of as a variation of the ICA approaches which requires an over-complete basis vector for the mixing matrix. DL may be advantageous because

additional constraints such as a positive code or a dictionary can be applied to the model. However, since it requires an over-complete basis vector, information may be lost when reconstructing the source signal. In addition, like all the other approaches, DL is also non-convex and it is sensitive to the initialization of the weights.

All previous approaches have limitations such as loss of information or non-convex optimization and require constraints or assumptions such as orthogonality and an affine transformation which are not ideal for BSS. In the following we introduce our approach to BSS by using an *information geometric formulation of the log-linear model*. Unlike the previous approaches, our proposed approach does not have the assumptions or limitations that they require. We provide a flexible solution by introducing a *hierarchical structure* between signals into our model, which allows us to treat interactions between signals that are more complex than an affine transformation. Our experimental results demonstrate that our hierarchical model leads to better separation of signals including complex interaction such as higher-order feature interactions [65] than existing methods. Our method solves a convex optimization problem, hence it always arrives at the globally optimal unique solution. Moreover, we theoretically show that it always minimizes the Kullback–Leibler (KL) divergence from a set of mixed signals to a set of source signals.

4.2 Formulation

BSS is mathematically defined as a function f which is able to separate a set of *received signals* X into a set of *source signals* Z, i.e., $Z = f(X)$. However, it is generally considered to be impossible for the function f to recover the scale and the original order of the source signal. If we apply this limitation to the BSS problem, the problem is mathematically reduced to $Z \propto f(X)$.

4.2.1 Log-Linear Model on Partially Ordered Set and Information Geometry

First we prepare the information geometric formulation of the *log-linear model* proposed by Sugiyama et al. [97, 100], which models a discrete probability distribution. Let Ω be the sample space of distributions. Sugiyama et al. [100] showed that if interactions between variables is represented as *partial orders* between elements in Ω; that is, if Ω is a partially ordered set (poset) [28], such interactions can be naturally treated by the log-linear model. The minimum requirement is that Ω is finite and the least element $\bot$ exists such that $\bot \leq \omega$

for all $\omega \in \Omega$, in which the partition function of the log-linear model is associated with $\perp$. We denote by $\Omega^+ = \Omega \setminus \{\perp\}$

The log-linear model gives probability $p(\omega) \in (0,1)$ to each element $\omega \in \Omega$ by

$$\log p(\omega;\theta) = \sum_{s\in\mathcal{S}, s\leq\omega} \theta(s) - \psi(\theta), \tag{4.1}$$

where $\mathcal{S} \subseteq \Omega$ is a predetermined parameter domain, and a parameter $\theta(s)$ is associated with each element $s \in \mathcal{S}$. The term $-\psi(\theta) = \theta(\perp)$ is the partition function which is uniquely determined from the parameters $(\theta(s))_{s\in\mathcal{S}}$,

$$\psi(\theta) = \log \sum_{\omega\in\Omega} \prod_{s\in\mathcal{S},\ s\leq\omega} \exp(\theta(s)) = -\theta(\perp). \tag{4.2}$$

It is known that Boltzmann machines is a special case of this log-linear model [101, 65] and also known that it belongs to the exponential family, where θ corresponds to natural parameters. The information geometric structure of the set of distributions, $\mathfrak{S} = \{p \mid 0 < p(\omega) < 1$ for all $\omega \in \Omega$ and $\sum_{\omega\in\Omega} p(\omega) = 1\}$, arises when we additionally introduce the *expectation* parameter given as

$$\eta(\omega) = \sum_{s\in\Omega, s\geq\omega} p(s;\theta), \tag{4.3}$$

Then the pair (θ, η) always becomes a pair of coordinate systems of the statistical manifold $\mathfrak{S}$, and they are orthogonal with each other as it is connected via *Legendre transformation* [6]. This orthogonality will be used in optimization of the model, where θ and η are jointly used to achieve minimization of the KL divergence [5].

4.2.2 Information Geometric Formulation of Blind Source Separation

Here we introduce our key technical contribution of this Chapter, a hierarchical structure for the sample space Ω of the log-linear model, to achieve BSS. We call this model *information geometric BSS* (IGBSS). Our idea is to construct the hierarchical structure of the three layers of BSS, the mixing layer, the source layer, and the received layer, into the sample space Ω and learn the joint representation on it using the log-linear model. The received layer and the source layer represent the input received signal and the output source signal of BSS, respectively, and the mixing layer encodes information of how to mix the source signal.

First, we treat a received (mixed) signal $\mathbf{X} \in \mathbb{R}^{L\times M}$ with the number L of received signals and the sample size M, which is an input to BSS, as an empirical distribution. To treat $\mathbf{X}$ as a probability mass function, we normalize it beforehand by dividing each

element by the sum of all elements; that is, an (input) empirical distribution $\hat{p}$ is obtained as $\hat{p}(x_{lm}) = x_{lm}/\sum_{l,m} x_{lm}$[2]. If the original input $\mathbf{X}$ contains negative values, an exponential kernel or min-max normalization can be used to obtain nonnegative values; that is, for the exponential kernel $\exp(x_{lm})/\sum_{l,m}\exp(x_{lm})$ and for min-max normalization $(x_{lm}+\varepsilon-\min(\mathbf{X}))/(\max(\mathbf{X})+\varepsilon-\min(\mathbf{X}))$, where ε is some arbitrary small value to avoid zero probability.

Next we implement three layers in the sample space Ω. We define $\Omega = \{\perp\} \cup \mathcal{A} \cup \mathcal{Z} \cup \mathcal{X}$ with $\mathcal{A} = \{a_{11}, \ldots, a_{LN}\}$, $\mathcal{Z} = \{z_{11}, \ldots, z_{MN}\}$, and $\mathcal{X} = \{x_{11}, \ldots, x_{LM}\}$, where N denotes the number of source signals. These sets $\mathcal{A}$, $\mathcal{Z}$, and $\mathcal{X}$ correspond to the mixing, source, and received layers, respectively. We always assume that the parameter domain $\mathcal{S} = \mathcal{A} \cup \mathcal{Z}$, meaning that mixing and source layers are used as parameters to represent distributions in our model. Here we introduce a partial order $\leq$ into our sample space Ω to construct a hierarchical structure of layers. Define

$$\begin{cases} a_{ij} \leq z_{kl} & \text{if } j = k \quad \text{for all } a_{ij} \in \mathcal{A} \text{ and } z_{kl} \in \mathcal{Z}, \\ z_{ij} \leq x_{kl} & \text{if } j = l \quad \text{for all } z_{ij} \in \mathcal{Z} \text{ and } x_{kl} \in \mathcal{X}. \end{cases} \tag{4.4}$$

The first condition encodes the structure such that the source layer is higher than the mixing layer, and the second condition encodes that the received layer is higher than the source layer. An example of our sample space with $L = M = N = 2$ is illustrated in Figure 4.1.

The log-linear model on our sample space Ω is given in Equation (4.1), and if we learn the joint distribution on Ω from a received signal $\mathbf{X}$, we will obtain probabilities on the source layer $p(z_{11}), \ldots, p(z_{MN})$, which represents normalized source signals. The rational of our approach is given as follows: The connections between each layer is structured so that the log-linear model performs a similar computation as a matrix multiplication between the mixing layer $\mathcal{A}$ and the source layer $\mathcal{Z}$ in the form of $\mathbf{X} = \mathbf{AZ}$, which is a typical approach employed by ICA. ICA models BSS as the matrix multiplication in which $\mathbf{X}$, $\mathbf{Z}$, and $\mathbf{A}$ correspond to the received signal, the source signal, and the mixing matrix, respectively, and assume that each x_{lm} is determined by $x_{lm} = \sum_{n=1}^{N} a_{ln} z_{nm}$, which means that $\mathbf{Z}$ is the affine transformed into $\mathbf{X}$. Similarly to ICA, our structure also ensures that each $p(x_{lm})$ is determined by $\theta(a_{ln})$ and $\theta(a_{ln})$, $n \in \{1, \ldots, N\}$, as we always have $a_{ln} \leq x_{lm}$ and $z_{nm} \leq x_{lm}$. Moreover, this formulation allows us to model more complex interaction, such as higher-order interactions, between signals if we additionally include partial order structure into $\mathcal{Z}$ and/or $\mathcal{A}$, which cannot be treated by a simple matrix multiplication.

According to the definition of the log-linear model in Equation (4.1), we can obtain each probability of the joint distribution P on Ω as follows.

[2]We abuse an entry x_{lm} of $\mathbf{X}$ and its corresponding state in Ω to avoid complicated notations.

Received Layer (Input Layer)

The received layer represents the input to the model. Probability $p(x)$ on the received layer $x \in \mathcal{X}$ is obtained as

$$\log p(x;\theta) = \sum_{z' \in \mathcal{Z}, z' \leq x} \theta(z') + \sum_{a' \in \mathcal{A}, a' \leq x} \theta(a') - \psi(\theta), \tag{4.5}$$

$$\eta(x;\theta) = \sum_{x' \in \mathcal{X},\ x' \geq x} p\left(x';\theta\right) = p(x;\theta). \tag{4.6}$$

Source Layer (Output Layer)

The next layer after the received layer is the source layer. This layer represents the model output. Probability $p(z)$ on the source layer for each $z \in \mathcal{Z}$ is given as

$$\log p(z;\theta) = \sum_{z' \in \mathcal{Z}, z' \leq z} \theta(z') + \sum_{a' \in \mathcal{A}, a' \leq z} \theta(a') - \psi(\theta) = \theta(z) + \sum_{a' \in \mathcal{A}, a' \leq z} \theta(a') - \psi(\theta), \tag{4.7}$$

$$\eta(z;\theta) = \sum_{x' \in \mathcal{X},\ x' \geq z} p\left(x';\theta\right) + \sum_{z' \in \mathcal{Z},\ z' \geq z} p\left(z';\theta\right) = \sum_{x' \in \mathcal{X},\ x' \geq z} p\left(x';\theta\right) + p(z;\theta). \tag{4.8}$$

Mixing Layer

The mixing layer is the first layer in the model. The mixing layer is the base layer to provide the connection between received signals and source signals. Probability $p(a)$ on this layer for each $a \in \mathcal{A}$ is given as

$$\log p(a;\theta) = \sum_{a' \in \mathcal{A}, a' \leq a} \theta(a') - \psi(\theta) = \theta(a) - \psi(\theta), \tag{4.9}$$

$$\eta(a;\theta) = \sum_{x' \in \mathcal{X},\ x' \geq a} p\left(x';\theta\right) + \sum_{z' \in \mathcal{Z},\ z' \geq a} p\left(z';\theta\right) + \sum_{a' \in \mathcal{A},\ a' \geq a} p\left(a';\theta\right) = \sum_{x' \in \mathcal{X},\ x' \geq a} p\left(x';\theta\right) + \sum_{z' \in \mathcal{Z},\ z' \geq a} p\left(z';\theta\right) + p(a;\theta). \tag{4.10}$$

The parameter values $\theta(a)$ in the mixing layer represents the degree of mixing between source signals. Hence they can be used to perform feature selection and feature extraction. For example, if $\theta(a) = 0$ in the extreme case, this means that this node a does not have any contribution to the source mixing.

Bottom Node

The bottom node $\perp$ provides the connections for all the elements in the node. The value $\theta(\perp)$ coincides with the negative of the partition function, i.e., $\psi(\theta) = -\theta(\perp)$. The partition

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}$$

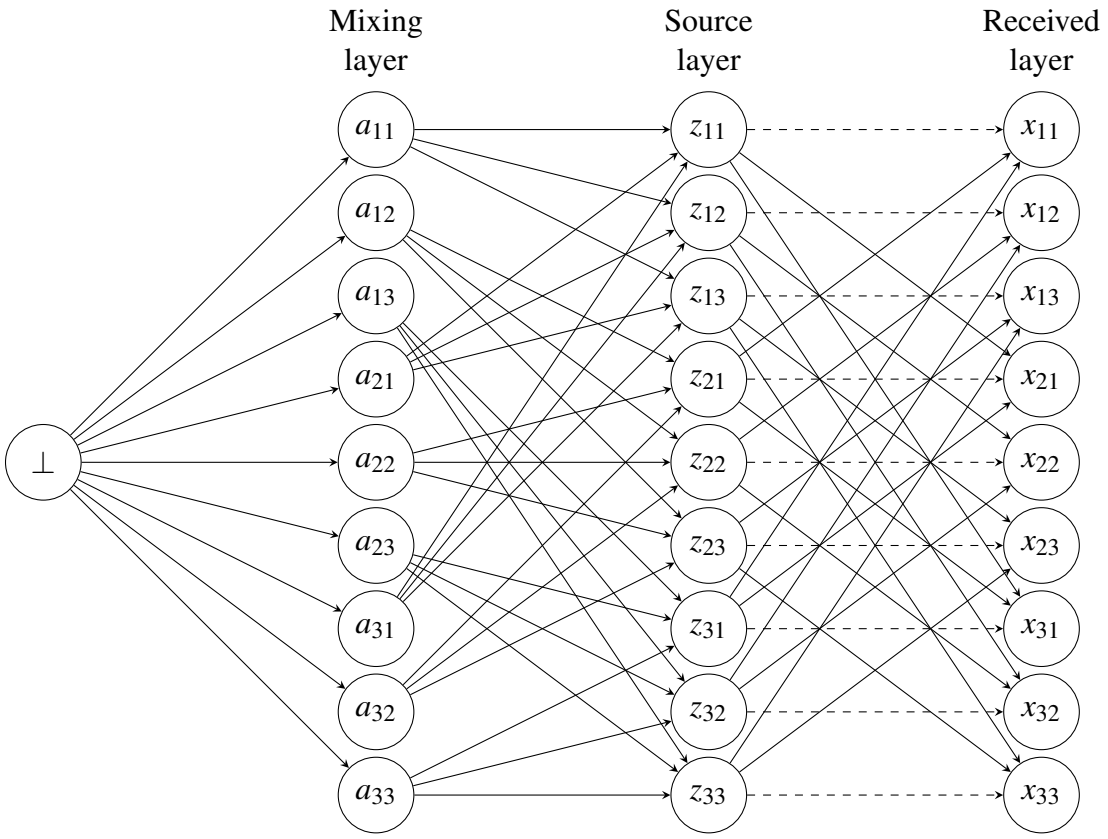


Fig. 4.1 An example of our sample space. The dashed line shows removed partial orders to allow for learning.

function ensures that the joint probability distribution on the entire space Ω sums to 1; that is, $\sum_{\omega\in\Omega} p(\omega) = 1$. Note that the partition function is not necessary when the model is applied to perform BSS, while we include it to theoretically provide the optimality of our model.

4.2.3 Optimization

We optimize the log-linear model by minimizing the KL divergence from an empirical distribution $\hat{p}$, which is determined by the received signal $\mathbf{X}$, to the model joint distribution p given by Equations (4.5), (4.7), and (4.9). Given a received signal X as an input dataset. In our empirical distribution, we only have observations on the received signal $\hat{p}(x)$ and do not have any observations for the values of the mixing and the source layers. Therefore we set

the empirical distribution $\hat{p}$ is defined as $\hat{p}(x_{lm}) = x_{lm}/\sum_{l,m} x_{lm}$ for all $x_{lm} \in \mathcal{X}$, $\hat{p}(z) = 0$ for all $z \in \mathcal{Z}$, and $\hat{p}(a) = 0$ for all $a \in \mathcal{A}$.

Our model is trained by minimizing the KL (Kullback-Leibler) divergence to approximate the given empirical distribution $\hat{p}$ given as: $\operatorname{argmin}_{p \in \mathfrak{S}} \mathrm{D}_{\mathrm{KL}}(\hat{p} \| p) = \operatorname{argmin}_{p \in \mathfrak{S}} \sum_{\omega \in \Omega} \hat{p}(\omega) \log \frac{\hat{p}(\omega)}{p(\omega)}$. Let us introduce two submanifolds of the set of distributions $\mathfrak{S}$ of our log-linear model. They are given as $\mathfrak{S}_\theta = \{p \in \mathfrak{S} \mid \theta(\omega) = 0, \forall \omega \notin \mathcal{S} \cup \{\bot\}\}$ with linear constraints on θ, called an *e-flat* submanifold, and $\mathfrak{S}_\eta = \{p \in \mathfrak{S} \mid \eta(\omega) = \hat{\eta}(\omega), \forall \omega \in S\}$ with linear constraints on η, called an *m-flat* submanifold. Remember that the parameter domain $\mathcal{S} = \mathcal{A} \cup \mathcal{Z}$. The intersection $\mathcal{S}_\theta \cap \mathcal{S}_\eta$ is always a singleton and it always minimizes the KL divergence [5, Theorem 3], that is, it is the globally optimal solution of our model.

Optimization is achieved by *e-projection*, which projects a probability distribution in $\mathfrak{S}_\theta$ onto $\mathfrak{S}_\eta$. The *e*-projection is always *convex optimization* as the *e*-flat submanifold $\mathcal{S}_\theta$ is convex with respect to θ [6]. We can therefore use the standard gradient descent strategy to optimize the log-linear model. The derivative of the KL divergence with respect to $\theta(s)$ is known to be the difference between expectation parameters η [100, Theorem 2]:

$$\frac{\partial}{\partial \theta(s)} D_{\mathrm{KL}}(\hat{p} \,\|\, p) = \eta(s) - \hat{\eta}(s) = \Delta\eta(s), \tag{4.11}$$

where η is defined in Equation (4.3), and the KL divergence $D_{\mathrm{KL}}(\hat{p} \| p)$ is minimized if and only if $\eta(s) = \hat{\eta}(s)$, $\forall s \in \mathcal{S}$. In our case of the structured sample space Ω, the gradient of the mixing layer is given as

$$\frac{\partial}{\partial \theta(a)} D_{KL}(\hat{p} \| p) = \eta(a) - \hat{\eta}(a) = \left[\sum_{x \in \mathcal{X}, x \geq a} p(x) + \sum_{z \in \mathcal{Z}, z \geq a} p(z) + p(a) \right] - \sum_{x \in \mathcal{X}, x \geq z} \hat{p}(x). \tag{4.12}$$

Likewise, the gradient of the source layer is given as

$$\frac{\partial}{\partial \theta(z)} D_{KL}(\hat{p} \| p) = \eta(z) - \hat{\eta}(z) = \left[\sum_{x \in \mathcal{X},\ x \geq z} p(x) + p(z) \right] - \sum_{x \in \mathcal{X},\ x \geq z} \hat{p}(x). \tag{4.13}$$

From our definition (Equation (4.4)) of Ω, we have $\eta(z_{kl}) = \eta(z_{k'l})$ for all $z_{kl}, z_{k'l} \in \mathcal{Z}$. Therefore all elements in the source layer will learn the same value. This problem can be avoided by removing some of partial orders between source and received layers. We propose to systematically remove the partial order $z_{ij} \leq x_{ij}$ for all $z_{ij} \in \mathcal{Z}$ and $x_{ij} \in \mathcal{X}$ to ensure $\eta(z_{kl}) \neq \eta(z_{k'l})$ (see Figure 4.1), while other strategies are possible as long as a node does not become fully disconnected, for example, random deletion of such orders. Using the

Algorithm 1 Information Geometric BSS

Function IGBSS($\hat{P}$, $\mathcal{S}$):
Initialize $(\theta(s))_{s\in\mathcal{S}}$ (randomly or $\theta(s) = 0$)
repeat
  Compute P using the current parameter $(\theta(s))_{s\in\mathcal{S}}$
  Compute $(\eta(\omega))_{s\in\mathcal{S}}$ from P
  $(\Delta\eta(\omega))_{\omega\in\mathcal{Z}} \leftarrow (\eta(\omega))_{\omega\in\mathcal{Z}} - (\hat{\eta}(\omega))_{\omega\in\mathcal{Z}}$
  $(\Delta\eta(\omega))_{\omega\in\mathcal{A}} \leftarrow (\eta(\omega))_{\omega\in\mathcal{A}} - (\hat{\eta}(\omega))_{\omega\in\mathcal{A}}$
  Compute the Fisher information matrix for source layer $\mathbf{G}_Z$ and the mixing layer $\mathbf{G}_A$
  $(\theta(\omega))_{\omega\in\mathcal{Z}} \leftarrow (\theta(\omega))_{\omega\in\mathcal{Z}} - \mathbf{G}_Z^{-1}(\Delta\eta(\omega))_{\omega\in\mathcal{Z}}$
  $(\theta(\omega))_{\omega\in\mathcal{A}} \leftarrow (\theta(\omega))_{\omega\in\mathcal{A}} - \mathbf{G}_A^{-1}(\Delta\eta(\omega))_{\omega\in\mathcal{A}}$
until convergence of $(\theta(s))_{s\in\mathcal{S}}$
End Function

above results, gradient descent can be directly applied for the convex optimization problem and can solve our *e*-projection. However, this may need a large number of iterations to reach convergence. To reduce the number of iterations, we propose to use *natural gradient* [7] which is a second-order optimization approach. The natural gradient is an instance of the *Bregman algorithm* applied to convex regions and is well known that it is able to converge to the global optimal solution [22]. In our optimization problem, we are also able to find the global optimal solution using natural gradient because the KL divergence is convex with respect to θ. Let $\mathcal{S} = \{s_1, s_2, \ldots, s_{|\mathcal{S}|}\}$. We formulate the natural gradient by vectorize θ and η such that $\theta = [\theta(s_1), \ldots, \theta(s_{|\mathcal{S}|})]^{\mathrm{T}}$ and $\eta = [\eta(s_1), \ldots, \eta(s_{|\mathcal{S}|})]^{\mathrm{T}}$. In each step, the current θ is updated to θ_{next} by the formula ,

$$\theta_{\mathrm{next}} = \theta - \mathbf{G}^{-1}\Delta\eta,$$

where $\mathbf{G} = (g_{ij}) \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ is the Fisher information matrix such that

$$g_{ij} = \frac{\partial\eta(s_i)}{\partial\theta(s_j)} = \mathbb{E}\left[\frac{\partial \log p(\omega)}{\partial\theta(s_i)}\frac{\partial \log p(\omega)}{\partial\theta(s_j)}\right] = \sum_{\omega\in\Omega,\ \omega\geq s_i,\ \omega\geq s_j} p(\omega) - \eta(s_j)\eta(s_j) \tag{4.14}$$

This is obtained from Theorem 3 in [100].

Although the natural gradient requires much less iterations compared to the gradient descent, inverting a matrix is computationally expensive and has a complexity of $\mathcal{O}(|S|^3)$. In IGBSS, the entries for the cross-layer interaction between the mixing layer and the source layer are always $g_{az} = g_{za} = 0$ for all $a \in \mathcal{A}$ and $z \in \mathcal{Z}$. Hence they can be ignored in optimization. Therefore, to improve the efficiency of optimization, we can separate the

Table 4.1 Signal-to-Noise Ratio of the reconstructed data. (∗) Experimental results for Figure 4.2. (†) Experimental results for Figure 4.3. The ± shows the standard deviation after 40 runs. Each run contains a different set of images and a new randomly generated mixing matrix.

		Root Mean Squared Error (RMSE)				Signal-to-noise ratio (SNR) (units in dB)			
Exp.	Order	IGBSS	FastICA	DL	NMF	IGBSS	FastICA	DL	NMF
1	First*	**0.252 ± 0.000**	0.300 ± 0.089	0.394 ± 0.041	0.622 ± 0.000	**12.588 ± 0.000**	11.688 ± 4.829	6.810 ± 0.008	1.704 ± 0.000
	Second	**0.260 ± 0.000**	0.285 ± 0.096	0.441 ± 0.080	0.662 ± 0.000	10.729 ± 0.000	**12.353 ± 4.255**	0.526 ± 0.448	-3.426 ± 0.000
	Third†	**0.252 ± 0.000**	0.260 ± 0.111	0.362 ± 0.030	0.612 ± 0.000	12.588 ± 0.000	**12.922 ± 5.590**	1.471 ± 0.358	0.039 ± 0.000
2	First	**0.133 ± 0.000**	0.284 ± 0.064	0.474 ± 0.067	0.591 ± 0.000	**14.215 ± 0.000**	11.218 ± 1.964	2.098 ± 2.140	-0.940 ± 0.000
	Second	**0.256 ± 0.000**	0.263 ± 0.066	0.576 ± 0.008	0.684 ± 0.000	**10.612 ± 0.000**	11.986 ± 2.157	-1.589 ± 0.269	-3.675 ± 0.000
	Third	**0.282 ± 0.000**	0.239 ± 0.056	0.593 ± 0.007	0.665 ± 0.000	9.346 ± 0.000	**11.475 ± 2.145**	-2.274 ± 0.227	-4.073 ± 0.000
3	First	**0.155 ± 0.000**	0.699 ± 0.047	0.478 ± 0.121	0.628 ± 0.000	**11.285 ± 0.000**	10.785 ± 2.176	1.448 ± 4.249	0.628 ± 0.000
	Second	**0.200 ± 0.000**	0.280 ± 0.049	0.515 ± 0.007	0.709 ± 0.000	**10.862 ± 0.000**	10.171 ± 2.353	0.529 ± 0.228	-5.579 ± 0.000
	Third	**0.203 ± 0.000**	0.239 ± 0.056	0.536 ± 0.006	0.682 ± 0.000	**11.075 ± 0.000**	11.041 ± 2.708	-0.244 ± 0.185	-4.961 ± 0.000

update steps in the source layer and the mixing layer:

$$(\theta_{\text{next}}(\omega))_{\omega\in\mathcal{Z}} = (\theta(\omega))_{\omega\in\mathcal{Z}} - \mathbf{G}_Z^{-1}(\Delta\eta(\omega))_{\omega\in\mathcal{Z}}, \tag{4.15}$$

$$(\theta_{\text{next}}(\omega))_{\omega\in\mathcal{A}} = (\theta(\omega))_{\omega\in\mathcal{A}} - \mathbf{G}_A^{-1}(\Delta\eta(\omega))_{\omega\in\mathcal{A}}, \tag{4.16}$$

where $\mathbf{G}_Z^{-1}$ and $\mathbf{G}_A^{-1}$ are the Fisher information matrix for the source layer and mixing layer, respectively. They are constructed by assuming all other parameters are fixed. This approach reduces the complexity of the model to $\mathcal{O}(|\mathcal{Z}|^3+|\mathcal{A}|^3)$. In most practical cases, the size of $|\mathcal{Z}|$ is much greater than $|\mathcal{A}|$. This implies that the complexity of the model does not increase significantly if we include higher-order feature interactions. The full Algorithm using natural gradient is given in Algorithm 1. The complexity to compute $\mathcal{Q}$ in Algorithm 1 Line 4 is $\mathcal{O}(|\Omega||S|)$. The complexity to compute $\Delta\eta$ in Algorithm 1 Line 6 and Line 7 is $\mathcal{O}(|\mathcal{Z}|)+\mathcal{O}(\mathcal{A}) = \mathcal{O}(|S|)$. Therefore the total complexity of the model for each iteration is $\mathcal{O}(|\mathcal{Z}|^3+|\mathcal{A}|^3+|\Omega||S|)$.

4.3 Experiments

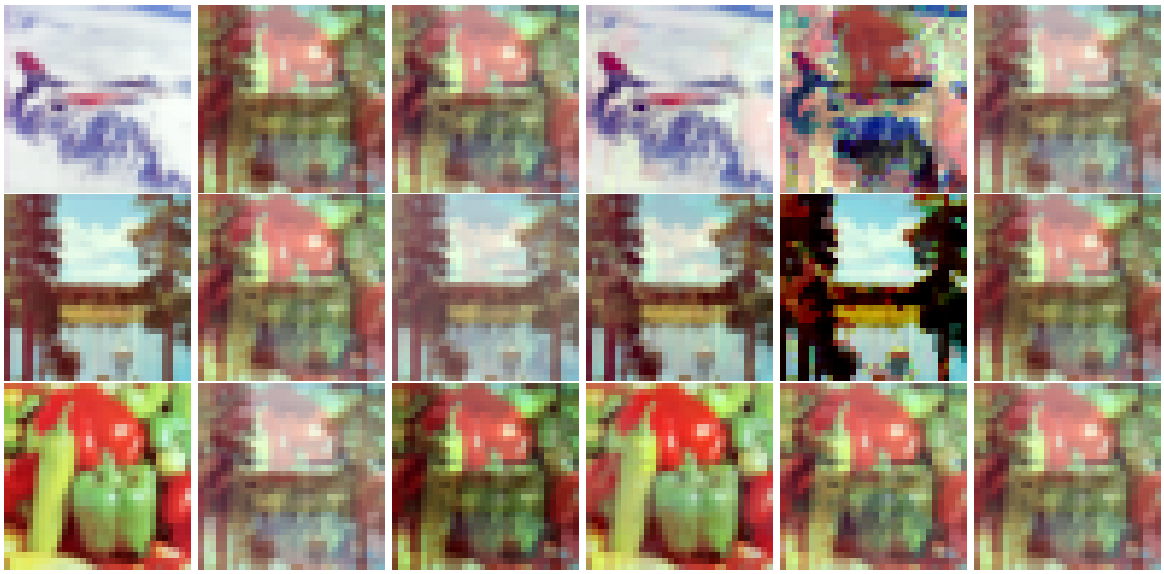
We empirically examine the effectiveness of IGBSS to perform BSS using real-world datasets for an affine transformation and higher-order interactions between signals. All experiments were run on CentOS Linux 7 with Intel Xeon CPU E5-2623 v4 and Nvidia QuadroGP100.

4.3.1 Blind Source Separation for Affine Transformations on Images

We demonstrate the effectiveness of our model. In our experiments, we use three benchmark images widely used in computer vision from the University of Southern California's Signal

(a) Source (b) Received (c) IGBSS (d) FastICA (e) NMF (f) DL

Fig. 4.2 First-order interaction experiment. Number of source signal: 3, order of interaction: 1, Number of mixed signal: 3.



(a) Source (b) Received (c) IGBSS (d) FastICA (e) NMF (f) DL

Fig. 4.3 Higher-order interaction experiment. Number of source signal: 3, order of interaction: 3, Number of mixed signal: 3.

and Image Processing Institute (USC-SIPI)[3], which include "airplane (F-16)", "lake" and "peppers". Each image is standardized to have 32x32 pixels with red, green and blue color channels with integer values between 0 and 255 to represent the intensity of each pixel. These images shown in Figure 4.2a are the *source signal* $\mathbf{Z}$ which are unknown to the model. They are only used as ground truth to evaluate the model output. The equation $\mathbf{X} = \mathbf{AZ}$ is used to generate the received signal $\mathbf{X}$ by randomly generating values for a mixing matrix $\mathbf{A}$ using the uniform distribution which generates real numbers between 1 and 6. The images are then rescaled to integer values within the range between 0 and 255. The received signal $\mathbf{X}$, which is the input to the model, is the three images shown in Figure 4.2b. The three images for the mixed signal may look visually similar, however, they are actually superposition of the source signal with different intensity. The objective of our model is to reconstruct the *source signal* $\mathbf{Z}$ without knowing the *mixing matrix* $\mathbf{A}$.

We compare our approach to FastICA with the log cosh function as the signal prior, DL with constraint for positive dictionary and positive code, and NMF with the coordinate descent solver and non-negative double singular value decomposition (NNDSVD) initialization [16] with zero values replaced with the mean of the input.

Since BSS is an unsupervised learning problem, the order of the signal is not recovered. We identify the corresponding signal by taking all permutations of the output and calculate the minimum euclidean distance with the ground truth. The permutation which returns the minimum error is considered the correct order of the image. The scale of the output is also not recovered, we have used min-max normalize the output of each model using

$$z_{\text{norm}} = \frac{z_{\text{out}} - \min(z_{\text{out}})}{\max(z_{\text{out}}) - \min(z_{\text{out}})}.$$

We visually inspect each of the model output and compare it to the ground truth. Our proposed approach IGBSS is able to recover majority of the "shape" of the source signal, while the intensity of each image appears to larger than the ground truth for all images. Small residuals of each image can be seen on the other images. For instance, in the airplane (F-16) image, there residuals from the lake image can be clearly seen. Compared to the reconstruction of IGBSS with FastICA, DL and NMF, IGBSS performs significantly better as all other approaches are unable to clearly separate the mixed signal. FastICA was unable to provide a reasonable reconstruction with 3 mixed signal. To overcome this limitation of FastICA, we randomly generated another column of the mixing matrix and append it to the current mixing matrix to create 4 mixed signals as an input to FastICA to recover a more reasonable signal.

[3] http://sipi.usc.edu/database/

The root mean square error (RMSE) of the Euclidean distance and the signal-to-noise ratio (SNR) between the reconstruction and the ground truth is calculated to quantify results of each method. The SNR is computed by

$$\text{SNR}_{dB} = 20\log_{10}\frac{z_{\text{norm}}}{|(z - z_{\text{norm}})|}.$$

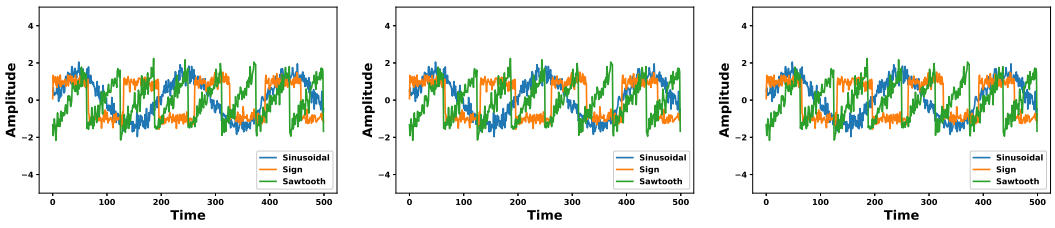
The full results are shown in Table 4.1 (top row for each experiment). In the table, we present three experiments with different RGB images from USC-SIPI dataset, for each experiment we generate a new mixing matrix, where the second and the third experiments uses images of "mandrill", "splash", "jelly beans" and "mandrill", "lake", "peppers", respectively. Ground truth and resulting images for second and third experiments are presented in Supplement. Our results clearly show that IGBSS is superior to other methods, that is, IGBSS has consistently produced the lowest RMSE error for every experiment. When looking at the SNR ratio, our model has produce the highest SNR for majority of the cases and is always able to recover the same result after each run.

4.3.2 Blind Source Separation with Higher-Order Feature Interactions

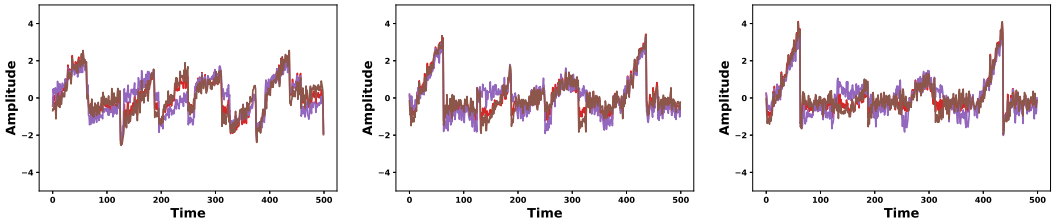
We demonstrate the ability of BSS for our model to include *higher-order feature interactions* in BSS. We use the same benchmark images in the standard BSS as the *source signal* $\mathbf{Z}$ for our experiment. We generate the higher-order feature interactions of the *received* signal by using the multiplicative product of the *source signal*. That is,

$$x_{lm} = \sum_{n=1}^{N} a_{ln} z_{nm} + \sum_{n=1}^{N} a_{ln} z_{nm} z_{nm'} + \sum_{n=1}^{N} a_{ln} z_{nm} z_{nm'} z_{nm''} + \cdots + \sum_{n=1}^{N} a_{ln} z_{nm} z_{nm'} \ldots z_{nm'\ldots'}.$$

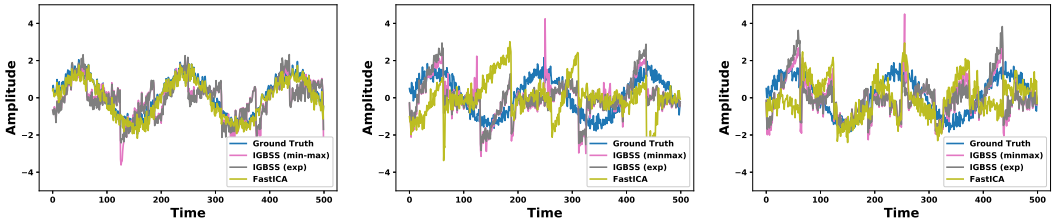
All the other known approaches take into account only first order interactions (that is, affine transformation) between features. Differently, our model can directly incorporate the higher-order features as we do not have any assumption of the affine transformation. Figure 4.3 shows experimental results for the higher-order feature experiment. Our approach IGBSS shows superior reconstruction of the source signal to other approaches. All the other approaches except for NMF is able to achieve reasonable reconstruction. NMF is able to recover the "shape" of the image, however, unlike IBSS, NMF is a degenerate approach, so it is unable to recover all color channels in the correct proportion, creating discoloring for the image which is clearly shown in the SNR values. Since the proportion of the intensity of the pixel is not recovered. In terms of the RMSE shown in Table 4.1, IGBSS again shows the best results for both second- and third-order interactions of signals across three experiments.



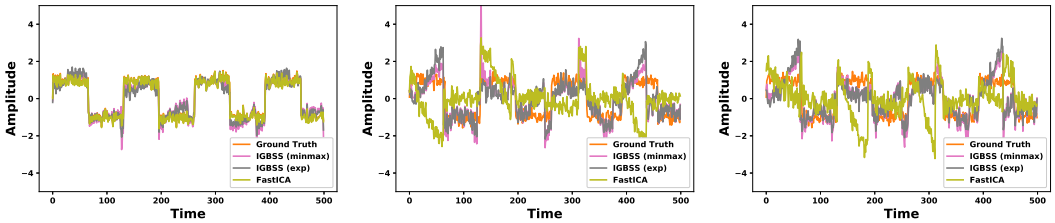
(a) Source signal (ground truth)



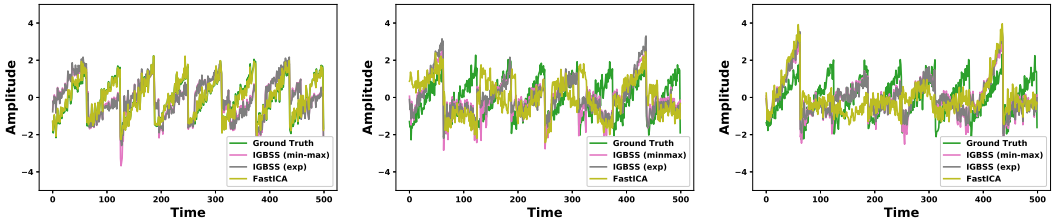
(b) Received signal (input)



(c) Recovery for sinusoidal function



(d) Recovery for sign function



(e) Recovery for sawtooth function

Fig. 4.4 Time Series Signal Experiment. First column represents first-order experiments, second column represents second-order experiments and third column represents third-order experiments.

Table 4.2 Quantitative results for time-series separation experiment. The $\pm$ shows the standard deviation after 40 runs.

(a) Root Mean Squared Error (RMSE)

Order	IGBSS (min-max)	IGBSS (exp)	FastICA
First	0.702 ± 0.000	0.703 ± 0.000	$\mathbf{0.414 \pm 0.286}$
Second	0.921 ± 0.000	$\mathbf{0.921 \pm 0.000}$	1.700 ± 0.167
Third	0.967 ± 0.000	$\mathbf{0.961 \pm 0.000}$	1.388 ± 0.178

(b) Signal-to-noise (SNR) (units in dB)

Order	IGBSS (min-max)	IGBSS (exp)	FastICA
First	3.596 ± 0.000	3.600 ± 0.000	$\mathbf{15.391 \pm 3.813}$
Second	$\mathbf{0.291 \pm 0.000}$	0.042 ± 0.000	-5.803 ± 1.124
Third	$\mathbf{0.340 \pm 0.000}$	0.128 ± 0.000	-3.427 ± 1.249

4.3.3 Higher-Order Interactions in Time Series Data Analysis

We demonstrate the effective of our model on time series data. In our experiments, we create three signals with 500 observations each using the sinusoidal function, sign function, and the sawtooth function. The synthetic data simulates typical signals from a wide range of applications including audio, medical and sensors. We randomly generate a mixing matrix by drawing from a uniform distribution with values between 0.5 and 2. In our experiment, we provide comparison of using both min-max normalization and exponential kernel as a pre-processing step and compare our approach with FastICA.

Experimental results are illustrated in Figure 4.4. These results show that IGBSS is superior to all the ICA approaches because it is able to recover both the shape of the signal and the sign of the signal, while all the other ICA approaches are only able to recover the shape of the signal and are unable to recover the sign of the signal. This means that ICA could recover a flipped signal. We have paired the recovered signal of ICA with the ground truth by finding the signal and sign with the lowest RMSE error. In any practical application, this is not possible for ICA because the latent signal is unknown. Through visual inspection, IGBSS is able to recover all visual signals with high accuracy, while FastICA is only able to recover the first-order interaction and it is unable to produce a reasonable recovery for second- and third-order interactions. In addition to our visual comparison, we have also performed a quantitative analysis on the experimental results using RMSE error with the ground truth. Results are shown in Table 4.2. FastICA has shown to have better performance for First-Order interactions. However, for second- and third-order SNR results for FastICA is unable to recover a reasonable signal because the noise is more dominant. IGBSS has

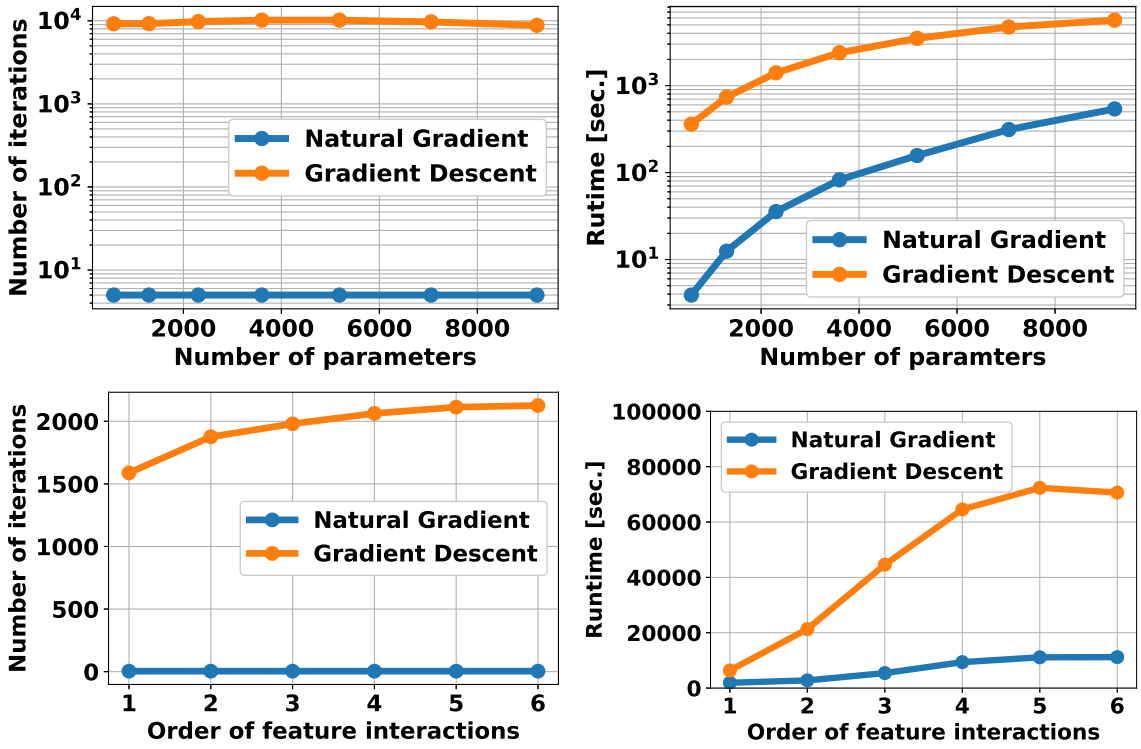


Fig. 4.5 Experimental analysis of the scalability of number of parameters and higher-order features in the model for both natural gradient approach and gradient descent

shown superior performance and is able to recover the signal for second- and third-order interactions with better scores for both RMSE and SNR.

4.3.4 Runtime Analysis

In our experiment, we used a learning rate of 1.0 for gradient descent. Although the time complexity for each iteration of natural gradient is $\mathcal{O}(|\mathcal{Z}|^3+|\mathcal{A}|^3+|\Omega||S|)$, which is larger than $\mathcal{O}(|\Omega||S|^2)$ for gradient descent, natural gradient is able to reach convergence faster because it is quadratic convergence and requires significantly less iterations compared to gradient descent, which linearly converges. Increasing the size of the input will increase the size of $|\Omega|$ only, while the number of parameters $|\mathcal{Z}|$, $|\mathbf{A}|$ remain this same. Since the complexity of natural gradient is linear with respect to the size $|\Omega|$ of the input, increasing the size of the input is unlikely to increase the runtime significantly. Our experimental analysis in Figure 4.5 supports this analysis: our model scales linearly for both natural gradient and gradient descent when increasing the order of interactions in our model. This is because for practical application it is unlikely that $|\mathcal{A}| > |\mathcal{Z}|$. The different between the runtime for natural gradient and gradient descent becomes larger as the order of interactions increased.

4.4 Summary

We have proposed a novel blind source separation (BSS) method, called *Information Geometric Blind Source Separation* (IGBSS). We have formulated our approach using the information geometric formulation of the log-linear model, which enables us to introduce a hierarchical structure into its sample space to achieve BSS. We have theoretically shown that IGBSS has desirable properties for BSS such as unique recover of source signals. It solves the convex optimization problem by minimizing the KL divergence from mixed signals to source signals to find the global optimal solution.

We have then experimentally shown that IGBSS recovers images and signals closer to the ground truth than independent component analysis (ICA), dictionary learning (DL) and non-negative matrix factorization (NMF). Thanks to the flexibility of the hierarchical structure, IGBSS is able to separate signals with complex interactions such as higher-order interactions. Our model is superior to all the other approach because it is a non-degenerate approach and is also able to recover the sign of the signal.

Our work has provided a new and novel formulation for BSS, which is supported by strong theoretical guarantees and experimental results to provide a better recovery of the source signal. Our approach is flexible and requires less assumptions than all alternative approaches, hence it can be applied to various real world applications such as medical imaging, signal processing, and image processing.

CHAPTER 5

Estimating Higher-Order Functions in Poisson Process

This chapter investigates using information geometric projects to formulate an efficient approach to estimating the higher-order intensity function in a multivariate Poisson process. A Poisson Process is a unsupervised machine learning model that is used for a series of discrete event where the average time between events is known, but the exact timing of events is random. This Chapter is to use concepts presented in Chapter 3 to estimate higher-order intensity functions in a multivariate Poisson processes by projecting observations into lower dimensional space.

This Chapter is organized as follows: we first provide an introduction to multivariate Poisson processes, we then outline the connection between Poisson processes and Generalized Additive Models (GAMs). This is then followed by our information geometric approach to estimate higher-order functions by using observations projected into lower dimensions. We then empirically demonstrate our models capability to estimate the higher-order intensity function.

5.1 Introduction

The Poisson process is a counting process used in a wide range of disciplines such as spatial-temporal sequential data in transportation [111, 112], finance [45], ecology [105], and the study of violent crimes [102] to model the arrival rate by learning an intensity function. For a given time interval, the integral of the intensity function represents the average number of events occurring in that interval.

The intensity function can be generalized to multiple dimensions. However, for most practical applications, learning the multi-dimensional intensity function becomes a challenge because of the sparse observations. Despite the recent advances of Poisson processes, most current Poisson process models are unable to learn the intensity function of a multi-dimensional Poisson process. Our research question is, "Are there any good ways of approximating the high dimensional intensity function?" Our proposed model, the *Additive Poisson Process* (APP), provides a novel solution to this problem.

Throughout this paper, we will use a running example in a spatial-temporal setting. Say we want to learn the intensity function for a taxi to pick up customers at a given time and location. For this setting, each event is multi-dimensional; that is, $(x_i, y_i, W_i, t_i)_{i=1}^N$, where a pair of x and y represents two spatial coordinates, W represents the day of the week, and t represents the time of the event. For any given location or time, we can expect at most a few pick-up events, which makes it difficult for any model to learn the low-valued intensity function. Figure 5.2b visualizes this problem. In this problem setup, if we would like to learn the intensity function at a given location (x, y) and day of the week W, the naïve approach would be to learn directly from the observation at (x, y, W). This would be extremely difficult because there could be very few events for a given location and day.

However, there is useful information in lower-dimensional space; for example, the marginalized observations at the location (x, y) across all days of the week, or on the day W at all locations. This information can be included into the model to improve the estimation of the joint intensity function. Using the information in lower-dimensional space provides a structured approach to include prior information based on the location or day of the week to improve the estimation of the joint intensity function. For example, a given location could be a shopping center or a hotel, where it is common for taxis to pick up passengers, and therefore we expect more passengers at this location. There could also be additional patterns that could be uncovered based on the day of the week. We can then use the observations of events to update our prior knowledge of the intensity function.

Previous approaches, such as kernel density estimation (KDE) [84], are able to learn the joint intensity function by using information in lower dimensions. However, KDE suffers from the curse of dimensionality, which means that it requires a large size of samples to build an accurate model. In addition, the complexity of the model expands exponentially with respect to the number of dimensions, which makes it infeasible to compute. Bayesian approaches, such as using a mixture of beta distributions with a Dirichlet prior [52] and Reproducing Kernel Hilbert Space (RKHS) [31], have been proposed to quantify the uncertainty for the intensity function. However, these approaches are often non-convex, making it

difficult to obtain the globally optimal solution. Besides, if observations are sparse, it is hard for these approaches to learn a reasonable intensity function.

In this paper, we propose a novel framework to learn the higher-order interaction effects of intensity functions in Poisson processes. Our model combines the techniques introduced by Luo and Sugiyama [65] to model higher-order interactions between Poisson processes and by Friedman and Stuetzle [34] in *generalized additive model*s to learn the joint intensity function using samples in a lower dimensional space. Our proposed approach is to decompose a multi-dimensional Poisson process into lower-dimensional representations. For example, we have points $(x_i)_{i=1}^N$ in the x-dimension, $(y_i)_{i=1}^N$ in the y-dimension, and $(t_i)_{i=1}^N$ in the time dimension. Such data in the lower-dimensional space can be used to improve the estimation of the joint intensity function. This is different from the traditional approaches where only the joint occurrence of events is used to learn the joint intensity.

We first show the connection between generalized additive models and Poisson processes, and then provide the connection between generalized additive models and the *log-linear model* [3], which has a well-established theoretical background in information geometry [6]. We draw parallels between the formulation of the generalized additive models and the binary log-linear model on a partially ordered set (poset) [100]. The learning process in our model is formulated as a convex optimization problem to arrive at a unique optimal solution using natural gradient, which minimizes the Kullback-Leibler (KL) divergence from the sample distribution in a lower-dimensional space to the distribution modeled by the learned intensity function. This connection provides remarkable properties to our model: the ability to learn higher-order intensity functions using lower-dimensional projections, thanks to the *Kolmogorov-Arnold representation theorem*. This property makes it advantageous to use our proposed approach for cases where there are no observations, missing samples, or low event rates. Our model is flexible because it can capture the interaction effects between events in a Poisson process as a partial order structure in the log-linear model and the parameters of the model are fully customizable to meet the requirements of the application. Our empirical results show that our model effectively uses samples projected onto a lower dimensional space to estimate the higher-order intensity function. More importantly, our model is also robust to various sample sizes.

5.2 Formulation

We start this section by introducing the technical background in the Poisson process and its extension to a multi-dimensional Poisson process. We then introduce the Generalized Additive Model (GAM) and its connection to the Poisson process. This is followed by

presenting our novel framework, called Additive Poisson Process (APP), which is our main technical contribution and has a tight link to the Poisson process modeled by GAMs. We show that the learning of APP can be achieved via convex optimization using natural gradient.

5.2.1 Poisson Process

The Poisson process is characterized by an intensity function λ:$\mathbb{R}^D \to \mathbb{R}$, where we assume D processes. An inhomogeneous Poisson process is an extension of a homogeneous Poisson process, where the arrival rate changes with time. The process with time-changing intensity $\lambda(t)$ is defined as a counting process $\mathbb{N}(t)$, which has an independent increment property. For any time $t \geq 0$ and infinitesimal interval $\delta \geq 0$, the probability of events count is

$$\begin{aligned} p(\mathbb{N}(t+\delta) - \mathbb{N}(t) = 0) &= 1 - \delta\lambda(t) + o(\delta) \\ p(\mathbb{N}(t+\delta) - \mathbb{N}(t) = 1) &= \delta\lambda(t) + o(\delta) \\ p(\mathbb{N}(t+\delta) - \mathbb{N}(t) \geq 2) &= o(\delta) \end{aligned}$$

where $o(\cdot)$ denotes little-o notation [27]. Given a realization of timestamps $\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N$ with $\mathbf{t}_i \in [0,T]^D$ from an inhomogeneous (multi-dimensional) a Poisson process can be used to learn the intensity function $\lambda(\mathbf{t})$ of the events. Each $\mathbf{t}_i$ is the time of occurrence for the i-th event across D dimensions and T is the observation duration. The likelihood for the Poisson process [27] is given by

$$p\left(\{\mathbf{t}_i\}_{i=1}^N \mid \lambda(\mathbf{t})\right) = \exp\left(-\int \lambda(\mathbf{t}) d\mathbf{t}\right) \prod_{i=1}^N \lambda(\mathbf{t}_i), \tag{5.1}$$

where $\mathbf{t} = [t^{(1)}, \ldots, t^{(D)}] \in \mathbb{R}^D$. We define the functional prior on $\lambda(\mathbf{t})$ as

$$\lambda(\mathbf{t}) := g(f(\mathbf{t})) = \exp(f(\mathbf{t})). \tag{5.2}$$

The function $g(\cdot)$ is a positive function to guarantee the non-negativity of the intensity which we choose to be the exponential function, and our objective is to learn the function $f(\cdot)$. The log-likelihood of the multi-dimensional Poisson process with the functional prior is described as

$$\log p\left(\{\mathbf{t}_i\}_{i=1}^N \mid \lambda(\mathbf{t})\right) = \sum_{i=1}^N f(\mathbf{t}_i) - \int \exp(f(\mathbf{t})) d\mathbf{t}. \tag{5.3}$$

In the following sections, we introduce the *generalized additive models* and propose to model it by the *log-linear model* to learn $f(\mathbf{t})$ and the normalizing term $(\int \exp(f(\mathbf{t}))\, d\mathbf{t})$.

5.2.2 Generalized Additive Model

In this section, we present the connection between Poisson processes with the Generalized Additive Model (GAM) proposed by Friedman and Stuetzle [34]. The GAM projects higher-dimensional features into lower-dimensional space to apply smoothing functions to build a restricted class of non-parametric regression models. GAM is less affected by the curse of dimensionality compared to directly using smoothing in a higher-dimensional space. For a given set of processes $J \subseteq [D] = \{1, \ldots, D\}$, the traditional GAM using one-dimensional projections is defined as

$$\log \lambda_J(\mathbf{t}) = \sum_{j \in J} f_j(t^{(j)}) - \beta_J.$$

with some smoothing function f_j.

In this paper, we extend it to include higher-order interactions between features in GAM by introducing terms that represent the multiplicative product between events. The *k-th order GAM* is defined as

$$\begin{aligned} \log \lambda_J(\mathbf{t}) &= \sum_{j \in J} f_{\{j\}}(t^{(j)}) + \sum_{j_1, j_2 \in J} f_{\{j_1, j_2\}}(t^{(j_1)}, t^{(j_2)}) + \cdots + \sum_{j_1, \ldots, j_k \in J} f_{\{j_1, \ldots, j_k\}}(t^{(j_1)}, \ldots, t^{(j_k)}) - \beta_J \\ &= \sum_{I \subseteq J, |I| \leq k} f_I(\mathbf{t}^{(I)}) - \beta_J, \end{aligned} \tag{5.4}$$

where $\mathbf{t}^{(I)}$ denotes the subvector $(\mathbf{t}^{(j)})_{j \in I}$ of $\mathbf{t}$ with respect to $I \subseteq [D]$.

The definition of the additive model is in the same form as Equation (5.3). In particular, if we compare Equation (5.3) and (5.4), we can see that the smoothing function f in (5.3) corresponds to the right-hand side of (5.4).

Learning of a continuous function using lower-dimensional projections is well known because of the *Kolmogorov-Arnold representation theorem*, which states that:

Theorem 5 (Kolmogorov–Arnold Representation Theorem [18, 51])**.** *Any multivariate continuous function can be represented as a superposition of one–dimensional smooth functions; that is,* $f(t_1, \ldots, t_n) = \sum_{q=1}^{2n+1} f_q \left(\sum_{p=1}^{n} g_{q,p}(t_p) \right)$.

Braun [17] showed that the GAM is an approximation to the general form presented in Kolmogorov-Arnold representation theorem by replacing the range $q \in \{1, \ldots, 2n+1\}$

with $I \subseteq J$ and the inner function $g_{q,p}$ by the identity if $q = p$ and zero otherwise, yielding $f(\mathbf{t}) = \sum_{I \subseteq J} f_I(\mathbf{t}^{(I)})$.

Interestingly, the canonical form for additive models in Equation (5.4) can be rearranged to be in the same form as Kolmogorov-Arnold representation theorem. By letting $f(\mathbf{t}) = \sum_{I \subseteq J} f_I(\mathbf{t}^{(I)}) = g^{-1}(\lambda(\mathbf{t}))$ and $g(\cdot) = \exp(\cdot)$, we have

$$\lambda_J(\mathbf{t}) = \frac{1}{\exp(\beta_J)} \exp\left(\sum_{I \subseteq J} f_I\left(\mathbf{t}^{(I)}\right)\right) \propto \exp\left(\sum_{I \subseteq J} f_I\left(\mathbf{t}^{(I)}\right)\right), \tag{5.5}$$

where we assume $f_I(\mathbf{t}^{(I)}) = 0$ if $|I| > k$ for the k-th order model and $1/\exp(\beta_J)$ is the normalization term for the intensity function. Based on the Kolmogorov-Arnold representation theorem, generalized additive models are able to learn the intensity of the higher-order interaction between Poisson processes by using projections into lower dimensional spaces. The log-likelihood function for a kth-order model is obtained by substituting the Equation (5.4) into Equation (5.1),

$$\log p\left(\{\mathbf{t}\}_{i=1}^{N} | \lambda(\mathbf{t})\right) = \sum_{i=1}^{N} \exp\left(\sum_{I \subseteq J,\ |I| \leq k} f_I\left(\mathbf{t}^{(I)}\right)\right) - \beta', \tag{5.6}$$

where β' is a constant given by $\beta' = \int \lambda(\mathbf{t}) d\mathbf{t} + \sum_{I \subseteq J} \beta_J$. In the following section we will detail a log-linear formulation that efficiently maximizes this log-likelihood equation. In the following section we will propose a graph structure that arises from a partial order structure to estimate the parameters in equation 5.6.

5.2.3 Additive Poisson Process

We introduce our key technical contribution in this section. We introduce a log-linear formulation called the *additive Poisson process* to estimate the parameters for the higher-order interactions in equation 5.6. We begin by discretizing the time window $[0, T]$ into M bins and treat each bin as a natural number $\tau \in [M] = \{1, 2, \ldots, M\}$ for each process. We assume that M is predetermined by the user. First we introduce a structured space for the Poisson process to incorporate interactions between processes. Let $\Omega = \{ (J, \tau) | J \in 2^{[D]} \setminus \emptyset, \tau \in [M] \} \cup \{(\perp, 0)\}$. We define the *partial order* $\preceq$ [28] on Ω as

$$(J, \tau) \preceq (J', \tau') \iff J \subseteq J' \text{ and } \tau \leq \tau', \qquad \text{for each } \omega = (J, \tau), \omega' = (J', \tau') \in \Omega, \tag{5.7}$$

and $(\perp, 0) \preceq (J, \tau)$ for all $(J, \tau) \in \Omega$, which is illustrated in Figure 5.1. The relation $J \subseteq J'$ is used to model any-order interactions between Poisson processes [65] [6, Section 6.8.4]

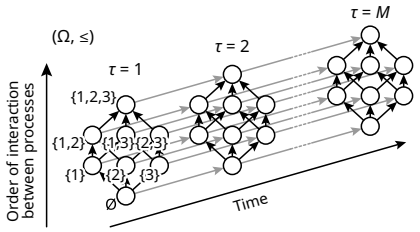
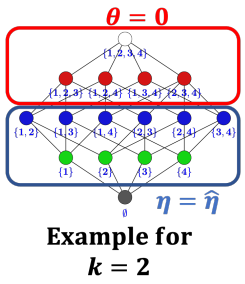
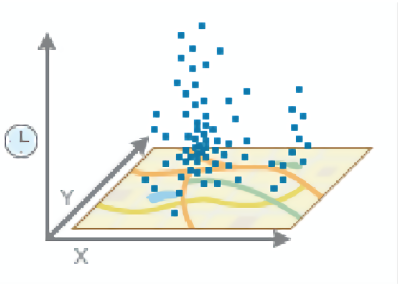


Fig. 5.1 Partial order structured sample space $(\Omega, \preceq)$ with $D = 3$. Each node represents a state and and the directed edge represents the direction of the partial ordering.



(a)



(b)

Fig. 5.2 (a) A visualisation of the truncated parameter space to approximate the joint intensity function with $D = 4$ and $k = 2$. (b) A visualization of the input datasets, where the blue points represent events with two spatial dimensions and one time dimension.

Algorithm 2 Additive Poisson Process (APP)

Function APP($\{\mathbf{t}_i\}_{i=1}^N$, $\mathcal{S}$, M, h):
Initialize Ω with the number M of bins
Apply Gaussian Kernel with bandwidth h on $\{\mathbf{t}_i\}_{i=1}^N$ to compute $\hat{p}$
Compute $\hat{\eta} = (\hat{\eta}_s)_{s\in\mathcal{S}}$ from $\hat{p}$
Initialize $\theta = (\theta_s)_{s\in\mathcal{S}}$ (randomly or $\theta_s = 0$)
repeat
    Compute p using the current $\theta = (\theta_s)_{s\in\mathcal{S}}$
    Compute $\eta = (\eta_s)_{s\in\mathcal{S}}$ from p
    $\Delta\eta \leftarrow \eta - \hat{\eta}$
    Compute the Fisher information matrix $\mathbf{G}$ using Equation (5.11)
    $\theta \leftarrow \theta - \mathbf{G}^{-1}\Delta\eta$
until convergence of $\theta = (\theta_s)_{s\in\mathcal{S}}$
End Function

and each τ in (J, τ) represents the auto-regressive component ("time") in our model with $\perp$ denoting the least element in the partial order structure. Each node ω in the poset graph structure[1] represents the state of the sample space and the arrows represent the partial order relationship between two nodes[2]; that is, if $\omega \to \omega'$, then $\omega \preceq \omega'$.

Intuitively, the greatest node in J (in Figure 5.1 it is $(\{1,2,3\}, \tau)$ for all $\tau \in [M]$) represents the multivariate Poisson process. Other nodes represent projections into lower-dimensional

[1] In information geometry, this is a *hypergraph* which has a *simplicial complex* structure Ay et al. [10, Section 2.9].

[2] This graph structure should not be confused with the graph structure studied in graphical models, where the nodes typically represent a random variable and the arrows represent the relationship between the two random variables.

space that correspond to the marginalized observations; for example, $\{\{1\},\{2\},\{3\}\}$ and $\{\{1,2\},\{1,3\},\{2,3\}\}$ represent the first- and second-order processes. Using our example in the introduction, where we wanted to estimate the intensity function of a pick-up event of a taxi, $\{1\},\{2\},\{3\}$ represents (x,y), the two spatial dimensions, and the day of the week W and τ could represent the time dimension in each day. We can then update our prior knowledge to model the second-order intensity function by using observations of the second order events. For example, $\{1,2\},\{1,3\},\{2,3\}$ represents an event occurring at $\{x,y\}$, $\{x,W\}$, and $\{y,W\}$. We can then continue this process to an arbitrary order of interactions. Later on in this section we introduce the mathematics to estimate the higher-order function using a restricted number of lower-dimensional projections.

The domain of τ can be generalized from $[M]$ to $[M]^D$ to take different time stamps into account, while in the following we assume that observed time stamps are always the same across processes for simplicity. Our experiments in the next section demonstrate that we can still accurately estimate the density of processes. Our model can be applied not only to time-series data, but to any sequential data.

On any set equipped with a partial order, we can introduce a *log-linear model* [100]. Let us assume that a parameter domain $\mathcal{S} \subseteq \Omega$ is given. For a partially ordered set $(\Omega, \preceq)$, the log-linear model with parameters $(\theta_s)_{s\in\mathcal{S}}$ is introduced as,

$$\log p(\omega;\theta) = \sum_{s\in\mathcal{S}} \mathbf{1}_{[s\preceq\omega]}\theta_s - \psi(\theta), \tag{5.8}$$

for each $\omega \in \Omega$, where $\mathbf{1}_{[\cdot]} = 1$ if the statement in $[\cdot]$ is true and 0 otherwise, and $\psi(\theta) \in \mathbb{R}$ is the partition function uniquely obtained as,

$$\psi(\theta) = \log \sum_{\omega\in\Omega} \exp\left(\sum_{s\in\mathcal{S}} \mathbf{1}_{[s\preceq\omega]}\theta_s\right) = -\theta_{(\perp,0)}.$$

A special case of this formulation coincides with the density function of the *Boltzmann machines* [101, 65].

Here there is a clear correspondence between the log-linear formulation and that in the form of Kolmogorov-Arnold representation theorem in Equation (5.5) if we rewrite Equation (5.8) as

$$p(\omega;\theta) = \frac{1}{\exp\psi(\theta)} \exp\left(\sum_{s\in\mathcal{S}} \mathbf{1}_{[s\preceq\omega]}\theta_s\right) \propto \exp\left(\sum_{s\in\mathcal{S}} \mathbf{1}_{[s\preceq\omega]}\theta_s\right). \tag{5.9}$$

We call this model with $(\Omega, \preceq)$ defined in Equation (5.7) the additive Poisson process, which represents the intensity λ as the joint distribution across all possible states. The intensity λ of the multi-dimensional Poisson process given via the GAM in Equation (5.5) is fully

modeled (parameterized) by Equation (5.8) and each intensity $f_I(\cdot)$ is obtained as $\theta_{(I,\cdot)}$. To consider the k-th order model, we consistently use the parameter domain $\mathcal{S}$, given as,

$$\mathcal{S} = \{(J,\tau) \in \Omega \mid |J| \leq k\},$$

where k is an input parameter to the model that specifies the upper bound of the order of interactions. This means that $\theta_s = 0$ for all $s \notin \mathcal{S}$. Note that our model is well-defined for any subset $\mathcal{S} \subseteq \Omega$ and the user can use an arbitrary domain in applications. A visualization of the truncated parameter space is shown in Figure 5.2a.

For a given J and each bin τ with $\omega = (J,\tau)$, the empirical probability $\hat{p}(\omega)$, which corresponds to the input observation, is given as

$$\hat{p}(\omega) = \frac{1}{Z}\sum_{I \subseteq J} \sigma_I(\tau), \qquad Z = \sum_{\omega \in \Omega} \hat{p}(\omega), \qquad \sigma_I(\tau) := \frac{1}{N h_I}\sum_{i=1}^{N} K\left(\frac{\tau^{(I)} - \mathbf{t}_i^{(I)}}{h_I}\right) \tag{5.10}$$

for each discretized state $\omega = (J,\tau)$, where $\tau = (\tau,\ldots,\tau) \in \mathbb{R}^D$. The function σ_I performs smoothing on time stamps $\mathbf{t}_1,\ldots,\mathbf{t}_N$, which is the kernel smoother proposed by Buja et al. [19]. The function K is a kernel and h_I is the bandwidth for each projection $I \subseteq [D]$. We use the Gaussian kernel as K to ensure that probability is always nonzero, meaning that the definition of the kernel smoother coincides with the kernel estimator of the intensity function proposed by Schäbe [90].

5.2.4 Optimization

Given an empirical distribution $\hat{p}$ defined in Equation (5.10), the task is to learn the parameter $(\theta_s)_{s\in\mathcal{S}}$ such that the distribution via the log-linear model in Equation (5.8) is as close to $\hat{p}$ as much as possible. Let us define $\mathfrak{S}_\mathcal{S} = \{p \mid \theta_s = 0 \text{ if } s \notin \mathcal{S}\}$, which is the set of distributions that can be represented by the log-linear model using the parameter domain $\mathcal{S}$. Then the objective function is given as $\min_{p\in\mathfrak{S}_\mathcal{S}} D_{\mathrm{KL}}(\hat{p},p)$, where $D_{\mathrm{KL}}(\hat{p},p) = \sum_{\omega\in\Omega}\hat{p}\log(\hat{p}/p)$ is the KL divergence from $\hat{p}$ to p. In this optimization, let p^* be the learned distribution from the sample with infinitely large sample size and let p be the learned distribution for each sample. Then we can lower bound the uncertainty (variance) $\mathbb{E}[D_{\mathrm{KL}}(p^*,p)]$ by $|\mathcal{S}|/2N$ [11].

Thanks to the well-developed theory of *information geometry* [6] for the log-linear model [8], it is known that this problem can be solved by *e-projection*, which coincides with the maximum likelihood estimation and is always *convex optimization* [6, Chapter 2.8.3]. The gradient with respect to each parameter θ_s is obtained by $(\partial/\partial\theta_s)D_{\mathrm{KL}}(\hat{p},p) = \eta_s - \hat{\eta}_s$, where $\eta_s = \sum_{\omega\in\Omega}\mathbf{1}_{[\omega\succeq s]}p(\omega)$. The value η_s is known as the expectation parameter [100]

and $\hat{\eta}_s$ is obtained by replacing p with $\hat{p}$ in the above equation. If $\hat{\eta}_s = 0$ for some $s \in \mathcal{S}$, we remove s from $\mathcal{S}$ to ensure that the model is well-defined.

Let $\mathcal{S} = \{s_1, \ldots, s_{|\mathcal{S}|}\}$ and $\theta = [\theta_{s_1}, \ldots, \theta_{s_{|\mathcal{S}|}}]^T$, $\eta = [\eta_{s_1}, \ldots, \eta_{s_{|\mathcal{S}|}}]^T$. We can always use the *natural gradient* [7] as the closed form solution of the Fisher information matrix is always available [100]. The update step is $\theta_{\text{next}} = \theta - \mathbf{G}^{-1}(\eta - \hat{\eta})$, where the Fisher information matrix $\mathbf{G}$ is obtained as

$$g_{ij} = \frac{\partial}{\partial \theta_{s_i} \partial \theta_{s_j}} D_{\text{KL}}(\hat{p}, p) = \sum_{\omega \in \Omega} \mathbf{1}_{[\omega \succeq s_i]} \mathbf{1}_{[\omega \succeq s_j]} p(\omega) - \eta_{s_i} \eta_{s_j}. \tag{5.11}$$

Theoretically the Fisher information matrix is numerically stable to perform a matrix inversion. However, computationally, floating point errors may cause the matrix to become indefinite. To overcome this issue, a small positive value is added along the main diagonal of the matrix. This technique is known as jitter and it is used in areas like Gaussian processes to ensure that the covariance matrix is computationally positive semi-definite [75].

The pseudocode for APP is shown in Algorithm 2. The time complexity of computing line 7 is $\mathcal{O}(|\Omega||\mathcal{S}|)$. This means when implementing the model using gradient descent, the time complexity of the model is $\mathcal{O}(|\Omega||\mathcal{S}|^2)$ to update the parameters in $\mathcal{S}$ for each iteration. For natural gradient, the cost of inverting the Fisher information matrix G is $\mathcal{O}(|\mathcal{S}|^3)$; therefore, the time complexity to update the parameters in $\mathcal{S}$ is $\mathcal{O}(|\mathcal{S}|^3 + |\Omega||\mathcal{S}|)$ for each iteration. The time complexity for natural gradient is significantly higher higher because of the requirement to invert the fisher information matrix; if the number of parameters is small, it is more efficient to use natural gradient because it requires significantly fewer iterations. However, if the number of parameters is large, it is more efficient to use gradient descent.

5.3 Experiments

We perform experiments using two-dimensional synthetic data, higher-dimensional synthetic data, and real-world data to evaluate the performance of our proposed approach.

5.3.1 Experimental Setup

Our code is implemented in Python 3.7.5 with NumPy version 1.8.2 and the experiments are run on Ubuntu 18.04 LTS with an Intel i7-8700 6c/12t with 16GB of memory. In experiments with synthetic data, we simulate random events using Equation (5.1). We generate an intensity function using a mixture of Gaussians, where the mean is drawn from a uniform distribution and the covariance is drawn from an inverted Wishart distribution. The intensity function

is then the density function multiplied by the sample size. The synthetic data is generated by directly drawing a sample from the probability density function. An arbitrary number of samples is drawn from the mixture of Gaussians. We then run our models and compare with Kernel Density Estimation (KDE) [84], an inhomogeneous Poisson process whose intensity is estimated by a reproducing kernel Hilbert space formulation (RKHS) [31], and a Dirichlet process mixture of Beta distributions (DP-beta) [52]. The hyper-parameters M and h in our proposed model are selected using grid search and cross-validation. For situations where a validation set is not available, then h could be selected using a rule of thumb approach such as Scott's Rule [93] and M could be selected empirically from the input data by computing the time interval of the joint observation.

5.3.2 Bandwidth Sensitivity Analysis

Our first experiment is to demonstrate the ability for our proposed model to learn an intensity function from samples. We generate a Bernoulli process with probably of $p = 0.1$ to generate samples for every 1 seconds for 100 seconds to create a toy problem for our model. This experiment is to observe the behaviour of varying the bandwidth in our model. In Figure 5.3a, we observe that applying no kernel, we learn the deltas of each individual observation. When we apply a Gaussian kernel, the output of the model for the intensity function is much more smooth. Increasing the bandwidth of the kernel will provide a wider and much smoother function. Between the 60 seconds and 80 seconds mark, it can be seen when two observations have overlapping kernels, the intensity function becomes larger in magnitude.

5.3.3 One Dimensional Poisson Process

A one dimensional experiment is simulated using Ogata's thinning algorithm [78]. We generate two experiments use the standard sinusoidal benchmark intensity function with a frequency of 20π. The dense experiment has troughs with 0 intensity and peaks at 201 and the sparse experiment has troughs with 0 intensity and peaks at 2. Figure 5.3d shows the experimental results of the dense case, our model has no problem learning the intensity function. We compare our results using KL divergence between the underlying intensity function used to generate the samples to the intensity function generated by the model. Figure 5.3b shows that the optimal bandwidth is $h = 1$.

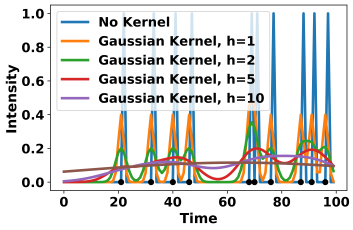
Algorithm 3 Thinning Algorithm for non-homogenous Poisson Process

Function Thinning Algorithm ($\lambda(t)$, T):
$n = m = 0$, $t_0 = s_0 = 0$, $\bar{\lambda} = \sup_{0 \le t \le T} \lambda(t)$
repeat
  $u \sim \text{uniform}(0,1)$
  $w = -\frac{1}{\bar{\lambda}} \ln u$ $\{w \sim \text{exponential}(\bar{\lambda})\}$
  $s_{m+1} = s_m + w$
  $D \sim \text{uniform}(0,1)$
  if $D \le \frac{\lambda(s_{m+1})}{\bar{\lambda}}$ **then**
    $t_{n+1} = s_{m+1}$
    $n = n + 1$
  else
    $m = m + 1$
  end if
  if $t_n \le T$ **then**
    return $\{t_k\}_{k=1,2,\ldots,n}$
  else
    return $\{t_k\}_{k=1,2,\ldots,n-1}$
  end if
until $s_m \le T$
End Function

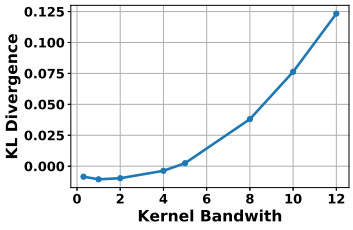
5.3.4 Experiments on Two-Dimensional Processes

For our experiment, we use 20 Gaussian components and simulate a dense case with 100,000 observations and a sparse case with 1,000 observations within the time frame of 10 seconds. We consider that a joint event occurs if the two events occur 0.1 seconds apart. Figure 5.4a and Figure 5.4b compare the KL divergence between the first- and second-order models and plots in Figure 5.5 are the corresponding intensity functions. In the first-order processes, both first- and second-order models have the same performance. This is expected, as both of the models can treat first-order interactions and are able to learn the empirical intensity function exactly, which is the superposition of the one-dimensional projection of the Gaussian kernels on each observation. For the second-order process, the second-order model performs better than the first-order model because it is able to directly learn the intensity function from the projection onto the two-dimensional space. In contrast, the first-order model must approximate the second-order process using the observations from the first-order processes. In the sparse case, the second-order model performs better when the correct bandwidth is selected.

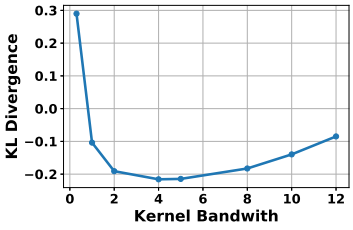
Table 5.1 compares our approach APP with other state-of-the-art approaches. APP performs the best for first-order processes in both the sparse and dense experiments. Exper-



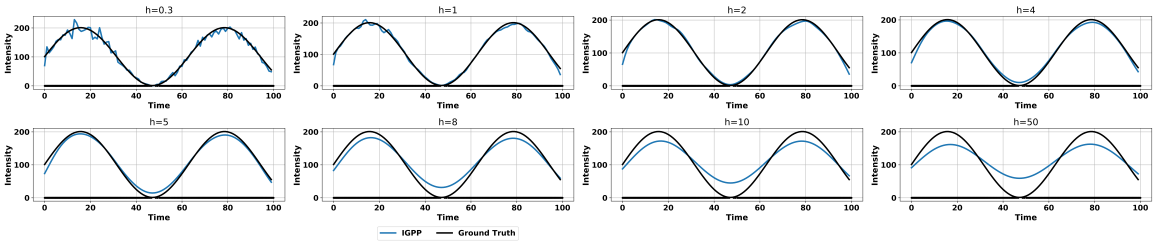
(a) Toy Example



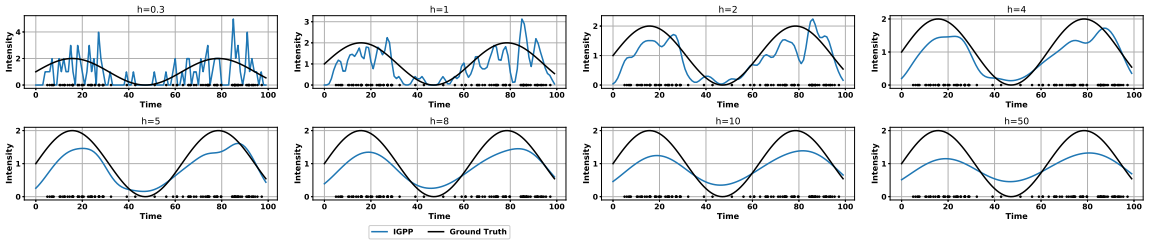
(b) KL divergence of dense experiment



(c) KL divergence of sparse experiment



(d) Ogata's thinning algorithm with high intensity



(e) Ogata's thinning algorithm with low intensity

Fig. 5.3 One dimensional experiments

iments for RKHS and DP-beta were unable to complete running within two days for the dense experiment. In the second-order process, our approach was outperformed by KDE, while the second-order APP was able to outperform both RKHS and DP-beta process for both sparse and dense experiments. Figures 5.4a and 5.4b show that KDE is sensitive to changes in bandwidth, which means that, for any practical implementation of the model, second-order APP with a less sensitive bandwidth is more likely to learn a more accurate intensity function when the ground truth is unknown.

5.3.5 Experiments on Higher-Dimensional Processes

We generate a fourth-order process to simulate the behavior of the model in higher dimensions. The model is generalizable to higher dimensions, but it is difficult to demonstrate results for processes higher than fourth order. For our experiment, we generate an intensity function

Table 5.1 The lowest KL divergence from the ground truth distribution to the obtained distribution on two types of single processes ([1] and [2]) and joint process of them ([1,2]). APP-# represents the order of the Additive Poisson Process. Missing values mean that the computation did not finish within two days.

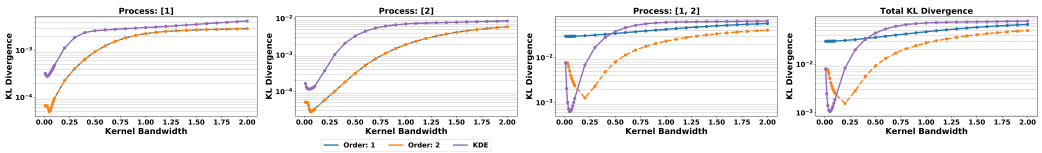
	Process	APP-1	APP-2	KDE	RKHS	DP-beta
Dense	[1]	**4.98e-5**	**4.98e-5**	2.81e-4	-	-
	[2]	**2.83e-5**	**2.83e-5**	1.17e-4	-	-
	[1,2]	2.98e-2	1.27e-3	**6.33e-4**	4.09e-2	4.54e-2
Sparse	[1]	**7.26e-4**	**7.26e-4**	8.83e-4	1.96e-2	2.62e-3
	[2]	**2.28e-4**	**2.28e-4**	2.76e-4	2.35e-3	2.49e-3
	[1,2]	2.88e-2	1.77e-2	**3.67e-3**	1.84e-2	3.68e-2

Table 5.2 Negative test log-likelihood for the New York Taxi data. Single processes ([T] and [W]) and joint process of them ([T,W]). APP-# represents the order of the Additive Poisson Process.

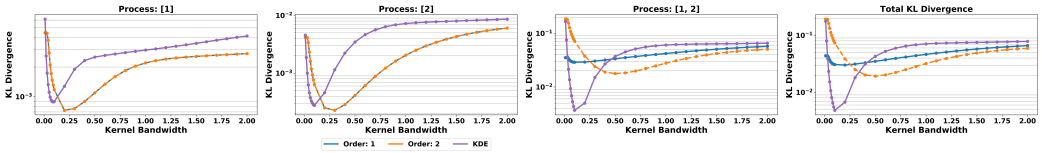
	Process	APP-1	APP-2	KDE	RKHS	DP-beta
Jan	[T]	714.07	714.07	**713.77**	728.13	731.01
	[W]	745.60	745.60	**745.23**	853.42	790.04
	[T,W]	249.60	**246.05**	380.22	259.29	260.30
Feb	[T]	**713.43**	**713.43**	755.71	795.61	765.76
	[W]	**738.66**	**738.66**	773.65	811.34	792.10
	[T,W]	328.84	**244.21**	307.86	334.31	326.52
Mar	[T]	**716.72**	**716.72**	733.74	755.48	741.28
	[W]	**738.06**	**738.06**	816.99	853.33	832.43
	[T,W]	291.20	**246.19**	289.69	328.47	300.36

using 50 Gaussian components and draw a sample with the size of 10^7 for the dense case and that with the size of 10^5 for the sparse case. We consider the joint event to be the time frame of 0.1 seconds.

We were not able to run comparison experiments with other models because they are unable to learn when there are no or few joint observations in third- and fourth-order processes. In addition, the time complexity is too high to learn from joint observations in first- and second-order processes because all the other models have their time complexity proportional to the number of observations. The time complexity for KDE is $\mathcal{O}(N^D)$ for the dimensionality with D, while DP-beta is $\mathcal{O}(N^2K)$, where K is the number of clusters, and RKHS is $\mathcal{O}(N^2)$ for each iteration with respect to the sample size N, where DP-beta and RKHS are applied directly on the joint observation as they cannot use the projections in lower-dimensional



(a) Dense observations $N = 100,000$.



(b) Sparse observations, $N = 1000$.

Fig. 5.4 KL Divergence for second-order Poisson process. The order of the model (color of the line) represents the k-th order model, i.e., $k = 1$ (blue) and $k = 2$ (orange).

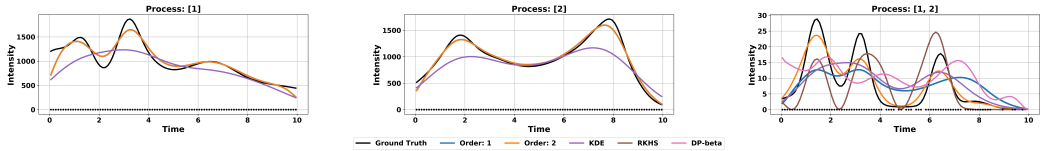
space. KDE is able to make an estimation of the intensity function using projections in lower-dimensional space, but it was too computationally expensive to complete running the experiment. By contrast, our model is more efficient because the time complexity is proportional to the number of bins in our model. The time complexity of APP for each iteration is $\mathcal{O}(|\Omega||\mathcal{S}|)$, where $|\Omega| = M^D$ and $|\mathcal{S}| = \sum_{c=1}^{k} \binom{D}{c}$. Our model scales combinatorially with respect to the number of dimensions. However, this is unavoidable for any model that directly takes into account the high-order interactions. For practical applications, the number of dimensions D and the order of the model k is often small, making it feasible to compute.

In Figure 5.6a, we observe similar behavior in the model, where the first-order processes fit precisely to the empirical distribution generated by the Gaussian kernels. The third-order model is able to period better on the fourth-order process. This is because the observation shown in Figure 5.7a is largely sparse and learning from the observations directly may overfit. A lower-dimensional approximation is able to provide a better result in the third-order model. Similar trends can be seen in the sparse case, as shown in Figure 5.6b, where a second-order model is able to produce better estimation in third- and fourth-order processes. The observations are extremely sparse, as seen in Figure 5.7b, where there are only a few observations or no observations at all to learn the intensity function.

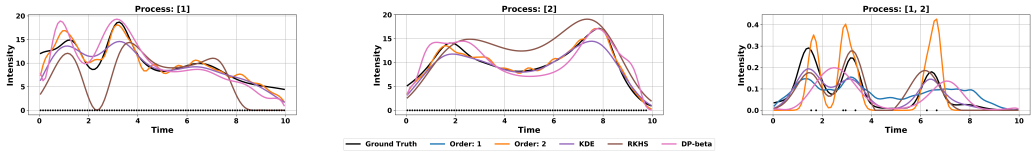
5.3.6 Uncovering Common Patterns in the New York Taxi Dataset

We demonstrate the capability of our model on the 2016 Green Taxi Trip dataset[3], which is a open source dataset with a CC0: Public Domain licences. We are interested in finding the common pick-up patterns across Tuesdays and Wednesdays. We define a common pick-up

[3] https://data.cityofnewyork.us/Transportation/2016-Green-Taxi-Trip-Data/hvrh-b6nb



(a) Dense observations, $N = 1000$, $h = 0.4$.



(b) Sparse observations, $N = 100,000$, $h = 0.2$.

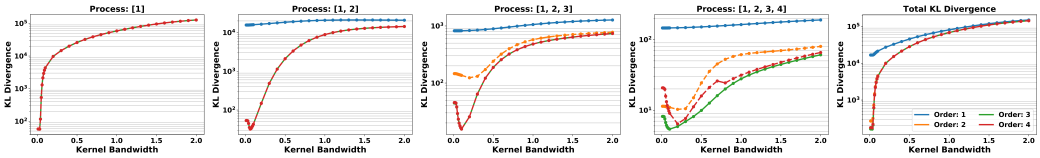
Fig. 5.5 Intensity function of two dimensional processes. Dots represent observations. Left: Represents marginalized observation of the first dimension. Middle: Represents marginalized observation of the second dimension. Right: The joint observation of dimension 1 and 2. The order of the model (color of the line) represents the k-th order model, i.e., $k = 1$ (blue) and $k = 2$ (orange).

time to be within 1-minute intervals of each other between the two days. We have chosen to learn an intensity function using the Poisson process for Tuesday and Wednesday and a joint process for both of them. The joint process uncovers the common pick-up patterns between the two days. We have selected to use the first two Tuesdays and Wednesdays in January 2016 as our training and validation sets and the Tuesday and Wednesday of the third week of January 2016 as our testing set. We repeat the same experiment for February and March.

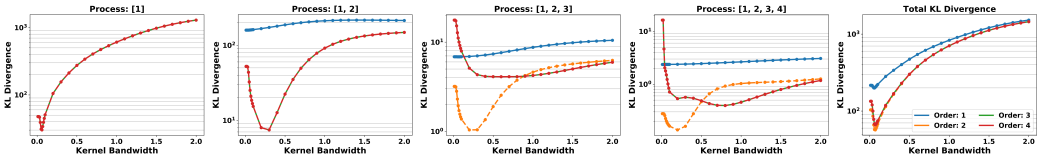
We show our results in Table 5.2, where we use the negative test log-likelihood as an evaluation measure. APP-2 has consistently outperformed all the other approaches for the joint process between Tuesday and Wednesday. In addition, for the individual process, APP-1 and -2 also showed the best result for February and March. These results demonstrate the effectiveness of our model in capturing higher-order interactions between processes, which is difficult for the other existing approaches.

5.4 Summary

We have proposed a novel framework, called *Additive Poisson Process* (APP), to learn the intensity function of the higher-order interaction between Poisson processes using observations projected into lower-dimensional spaces. We formulated our proposed model using the *log-linear model* and optimize it using information geometric structure of the distribution space. We drew parallels between our proposed model and *generalized additive model* and



(a) Dense observations, $N = 10^7$.

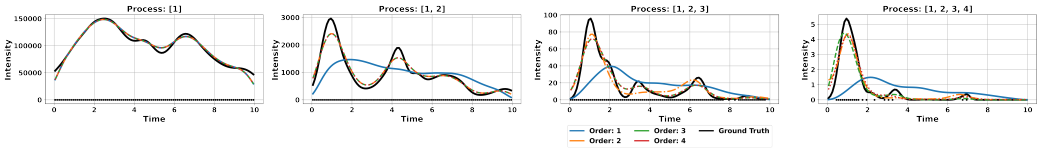


(b) Sparse observations, $N = 10^5$.

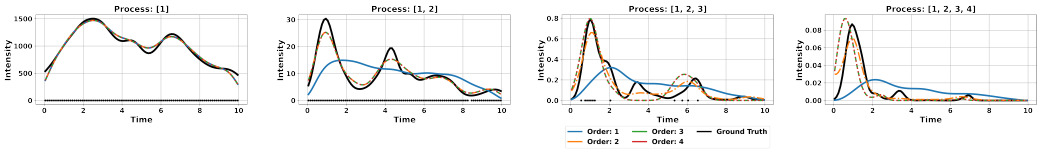
Fig. 5.6 KL Divergence for fourth-order Poisson process. We selected four representative examples for our experimental results, full results available in the appendix. The line color signifies the order of the model, i.e., $k = 1$ (blue), $k = 2$ (orange), $k = 3$ (green) and $k = 4$ (red).

showed the ability to learn from lower dimensional projections via the *Kolmogorov-Arnold representation theorem*.

Our empirical results show the superiority of our method when learning the higher-order interactions between Poisson processes and when there are no or extremely sparse joint observations. Our model is also robust to varying sample sizes. Our approach provides a novel formulation to learn the joint intensity function which typically has extremely low intensity. There is enormous potential to apply APP to real-world applications, where higher order interaction effects need to be modeled such as transportation, finance, ecology, and violent crimes.



(a) Dense observations, $N = 10^7$.



(b) Sparse observations, $N = 10^5$.

Fig. 5.7 Intensity function of higher dimensional processes. Dots represent observations. We have selected four representative examples for our experimental results, full results available in the appendix. The order of the model (color of the line) represents the k-th order model, i.e., $k = 1$ (blue), $k = 2$ (orange), $k = 3$ (green) and $k = 4$ (red).

CHAPTER 6

Automatically Selecting Higher-Order Features Using Dirichlet Process Priors

Chapter 3 explored the bias-variance trade-off when increasing the representational power of machine models by introducing hidden nodes and higher-order feature interactions. In this Chapter, we address the issue of model selection by exploring the feasibility to automatically select the number of parameters in the model by using techniques in Bayesian non-parametrics. We then demonstrate our proposed techniques to automatically select the number of parameters in the model on infrastructure failure prediction to demonstrate the capability of our approach in a real-world application.

In this Chapter, we first introduce the challenges faced in our application in infrastructure failure prediction. We then present the related work and the preliminary models found in literature. This is then followed by our proposed approach Hierarchical Dirichlet Process Mixture of Hierarchical Beta process which is used as automatic feature selection for complex real-world problems. We then demonstrate the capability of our proposed approach in real-world applications to make failure predictions in a water supply network.

6.1 Introduction to Infrastructure Failure Prediction

Sparsity and noisy labels are undesirable characters which occur in many real-world datasets. They are known to degrade the prediction performance of machine learning models. Sparsity may arise intrinsically and/or extrinsically. For instance, high dimensionality of the feature

space or label space, and a small number of observations are their respective examples. It is easy to say that we can improve a model by providing more training data, but there are many situations in which it is infeasible to do so. For example, there is not much historical data of infrastructure failure available. Previously, clustering technique were used to gain additional information from data points with similar characteristics. However, sparsity and noisy labels degrade the quality of clusters which in turn also degrades the quality of predictions [60]. Indeed, sparse prediction and noisy label problems are difficult to overcome with a single dataset because there are simply not enough data to build a reliable model. Previous solutions have been relying on the assistance from domain experts and using strong assumptions in model construction [59]. Differently, we propose a multi-task learning solution to automatically learn the model parameters for a group of *tasks* from multiple *domains*. They can then be applied to a failure prediction model, such as *hierarchical beta process* (HBP) [104].

HBP is a *Bayesian non-parametric model* (BNP) which is well suited to make sparse predictions by using information obtained from identified clusters. However, HBP model itself in its vanilla form is unable to assign data points to a cluster. In recent time, there have been proposals to apply Dirichlet prior to create a *generalised hierarchical beta process* (G-HBP) [38, 60] to automatically learn the model parameters. However, this method does not overcome the sparsity issue because it is difficult to produce low variance clusters in a sparse dataset. Moreover, it is also difficult to select the model parameters for each *task*. Distinguishably, we propose a solution which uses a BNP approach to automatically select the model parameters, which were previously selected by domain experts manually, by learning the *task* for a subset of data. Specifically, we adapt *hierarchical Dirichlet process* (HDP) [103] to form a multi-task learning framework to simultaneously learn the model parameters and prediction *tasks*. Our proposed model is highly flexible which can be used as a stand-alone transfer learning model for clustering sparse datasets. Alternatively, it can be connected to HBP to form our proposed *hierarchical Dirichlet process mixture of hierarchical beta process* (HDP-HBP) to learn the model parameters for a group of *tasks* to make sparse predictions.

The data sparsity commonly occurs in infrastructure failure predictions [59]. The robustness of an infrastructure heavily depends on its components and there are many factors which could contribute to the stability of the components. Some of the factors also exhibit complex feature interactions. To account for all these factors, datasets for infrastructure failure prediction tend to have a large feature space. Combined with a low failure rate in critical infrastructure, it is often very difficult to obtain sufficient samples to use traditional machine learning approaches for failure predictions.

In our experiments, we applied our proposed model to make water pipe failure predictions to demonstrate the capability of our proposed model in performing multi-task learning to improve sparse predictions. Any improvement to prediction accuracy in infrastructure preventive maintenance can have a material financial impact. Notably, American Society for Civil Engineers estimated that the United States needs to invest $3.6 trillion by 2020 to maintain its infrastructure [53]. Infrastructure failures could lead to significant social and economic cost in a city, where a single failure could cost more than 10 times the maintenance cost. Prioritising the correct water pipes to repair with a limited budget could have huge financial savings by preventing financial loss from water pipe failure. Nevertheless, the datasets which are available to make water pipe failure predictions are extremely sparse because the feature space is large. Moreover, there is only a very short observation period available and critical water main (CWM) have a low failure rate of about 0.5%. However, despite the low failure rate, the failure of a single CWM could lead to major disruptions and heavy financial losses in a society.

Predicting failure of CWM is very difficult to do because CWM components only makes up 10% of a water pipe network. Due to low failure event rate and the small number of such components in each network, it is very difficult to train a model with a single dataset without taking any strong assumptions. Currently, most of these critical infrastructures are repaired reactively. Our proposed multi-task learning method provides a solution by using additional datasets supplied by other utility companies with different feature spaces to automatically learn model parameters and *tasks* to make predictions for critical components. Note that a *task* in our context of study is defined to be a group of pipes with the same subset of features (i.e., a cluster). This subset of features is created by a Dirichlet Process (DP) to cluster on features with similar failure rates. It is defined to be a *task* because they will have the same model-parameters and hyper-parameters, but different observations when making the failure prediction.

In this paper, we extend the theoretical analysis and provide additional empirical verifications to our recent work on multi-task learning for sparse failure prediction [62]. We organise the rest of the paper as follows, Section 6.2 discusses related work in infrastructure failure prediction and multi-task learning in hierarchical models. Section 6.4.1 outlines our proposed model hierarchical Dirichlet process mixture model (HDPMM) for multi-task learning. We combine HDPMM with our failure prediction model hierarchical beta process (HBP) to form our sparse failure prediction model hierarchical Dirichlet process mixture of hierarchical beta process (HDP-HBP). Section 6.5 provides the analysis for our experimental results.

6.2 Related Work

This section provides an overview of the related work in infrastructure failure prediction and in multi-task learning models using Bayesian non-parametric approach. Below is a summary of some real-world implementations of failure predictions applying in different applications.

6.2.1 Empirical Bayes Method

A Bayesian approach to better handle sparsity and noisy label problems allows us to incorporate prior knowledge that have been obtained elsewhere to improve our model. The model can also be constructed in hierarchical form and to approximate a fully Bayesian model by placing prior distributions on the unknown parameters which required to be tuned. In the traditional Bayesian approach, these parameters are often selected by domain experts or tuned by machine learning experts via cross validation. However, with highly complex interactions, sparse samples, noisy labels and sampling bias, parameter selection through domain experts are no longer viable options. Empirical Bayes method provides a solution to approximate the parameters of a fully Bayesian model in Bayesian hierarchical models by using cross validation [14]. We propose an empirical Bayesian solution using multi-task learning by incorporating data obtained from other *domains* to assist in parameter selection for the model.

6.2.2 Multi-Task Learning in Hierarchical BNP Models

Multi-task learning is a special case of transfer learning, where the goal is to improve the prediction accuracy by solving multiple learning *tasks* at the same time. The improvement in performance is made by finding and exploiting commonalities across different *tasks* by relaxing the independent and identically distributed (iid) assumption [80]. This can be achieved through regularization or learning a common parameter in the model [110].

Hierarchical models are a natural way to formulate a multi-task learning framework. Hierarchical models are able to apply a constraint to ensure that the distribution between the *domains* learns a marginal probability distribution from a common parameter or prior [92]. For example, multi-task learning in Gaussian process (GP) a common kernel is learned to share information between *domains* [15]. Dirichlet Process (DP) and Beta Process (BP) have also been explored in multi-task learning by using it as a prior in hierarchical Bayesian models [38, 109].

DPs and BPs are Bayesian non-parametric (BNP) models. DPs have been explored in a number of areas such as Bayesian non-parametric mixture models, document analysis [103]

and DNA sequence analysis [42]. While BPs have been explored extensively in survival analysis [41] and factor analysis. They have shown to be highly flexible models which allow the number of parameters to be driven by the structure of the dataset. Combining BNP models with hierarchical models allow a model to have the flexibility to learn extremely complex datasets.

Multi-task learning in DP and BP attempts to learn the parameters for a group of common *tasks*. The parameters learnt by the DP and BP can then be applied to the prediction model to perform each *task*. We use the multi-task learning techniques previously used in BNP and apply it to HBP to formulate a solution which is able to simultaneously learn a group of *tasks* along with the model parameters for failure prediction.

6.3 Learning Parameters via Bayesian Non-Parametrics

In this section we introduce two Bayesian non-parametric approaches, the beta process and the Dirichlet process which we use to formulate a flexible solutions to learn the number of parameters in the model based on a given dataset. We first introduce the stick breaking process for both the Dirichlet process and the beta process, we then introduce the Chinese restaurant process (CRP), which provides a more efficient realization of the Dirichlet process.

6.3.1 Stick-Breaking Process

The *stick breaking process* are realizations of the *Dirichlet Process* (DP) and *Beta Process* (BP) so that it can be practically implemented. The stick-breaking process uses an analogy where a unit length stick is broken in a way that it resembles a DP or a BP.

Dirichlet Process

The *stick-breaking process* can represent a DP, $G \sim \mathrm{DP}(\alpha, H)$, where H is the base measure and α is the concentration parameter [95]. The stick is broken using a beta distribution and is assigned to an atom from the base distribution. The remaining part of the stick is used to continue the process. The algebraic expression for this process is given by

$$\begin{aligned} \pi'_k|\alpha_0, G_0 &\sim \mathrm{Beta}(1, \alpha_0), & \phi_k|\alpha_0, G_0 &\sim G_0, \\ \pi_k &= \pi'_k \prod_{l=1}^{k-1} (1-\pi'_l), & G &= \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k}, \end{aligned} \tag{6.1}$$

where δ is the indicator function, ϕ represents the location, π are independent random variables and k represents each cluster assignment. Often in literature, the notation $\pi \sim \text{GEM}(\alpha_0)$ is used to denote the stick-breaking realization of the DP. However, the stick-breaking realization of DP is not computationally tractable because it is an infinite process and requires to be truncated to be implemented. Later in Section 6.3.2, we introduce the Chinese Restaurant Process (CRP) which is a more efficient realization of the DP.

Beta Process

The BP, $H \sim \text{BP}(c, B_0)$ is defined as a Lévy process over a general space Ω with a random measure B_0. The BP can also be derived from the stick-breaking process. BP uses the part of the stick which has been broken off to continue the process. This process is complementary to the stick-breaking realization of the DP. This process can be represented as

$$\begin{aligned} H(\omega) &= \sum_i \pi_i \delta_{\omega_i}(\omega), \\ \pi_i &\sim \text{Beta}(cq_i, c(1-q_i)), \\ \omega_i &\sim B_0, \end{aligned} \tag{6.2}$$

where c is the concentration parameter, $q_i \in [0,1]$ is the mean parameter and B_0 is the discrete form of the base measure defined by $B_0 = \sum_i q_i \delta_{\omega_i}$. The beta process can be used as a natural prior to the Bernoulli process to make prediction on binary data.

6.3.2 Chinese Restaurant Process

The *Chinese restaurant process* (CRP) is a perspective on DP which uses the *Pólya urn scheme* representation. The name CRP comes from the analogy used to explain the model. Imagine a restaurant which has tables that can support an infinite number of customers. When a customer (i.e., a new data point) enters the restaurant (i.e., the *domain*), the probability for the new customer to be assigned to a table (i.e., the cluster) is proportional to the number of customers sitting on that particular table. Since each table attracts a new customer proportional to the number of customers on the table. The equation for CRP is given by

$$\begin{aligned} P(z_i = a | z_1, \ldots, z_{i-1}) &= \text{CRP}(\gamma) \\ &= \begin{cases} \frac{n_a}{N-1+\gamma} & \text{If } k \leq K \\ \frac{\gamma}{N-1+\gamma} & \text{Otherwise,} \end{cases} \end{aligned} \tag{6.3}$$

where n_a is the number of data points in each table, k is the cluster index in K, N is the total number of customers in the restaurant and γ is the hyper-parameter controlling the probability of the new customer sitting on a new table.

The CRP can be constructed in hierarchy of each other to create *hierarchical Dirichlet process* (HDP) from a CRP perspective, this is a Chinese restaurant franchise (CRF) [103]. Explaining HDP with the same analogy as CRP, each customer enters restaurant franchise with a shared menu (i.e., a list of dishes). At each table, one dish (i.e., model parameter) is ordered from the menu and is shared among all customers who sit at that table. Each restaurant can serve the same dish at different table, the same dish can be served at different restaurants.

Both CRP and CRF are often used as priors for clustering and mixture models because the exchangeability property provides a computational efficient method to solve DP.

6.4 Our Proposed Model for Sparse Predictions

This section presents our proposed model *Hierarchical Dirichlet Process Mixture Model* (HDPMM) for model-parameter transfer. We then combine HDPMM with our failure prediction model HDP to formulate our infrastructure failure prediction model *Hierarchical Dirichlet Process Mixture of Hierarchical Beta Process* (HDP-HBP).

6.4.1 A Flexible Transfer Learning Framework

We propose to use *hierarchical Dirichlet process mixture model* (HDPMM) to perform the clustering to group across *domains*. The aim of the HDPMM framework is to perform cross *domain* clustering to find a better representation of the structure of the failure patterns in a given network. This framework allows more data to learn the failure characteristics of each cluster, to reduce the variance in learning the clusters. When using the clustering information for prediction, the empirical expectation of each cluster should be closer to the true expectation. This means that data points with low number of failure counts and low number of observations are able to leverage on clusters with high number of failure counts and high number of observations.

HDPMM is constructed from *Dirichlet process* (DP), where an explicit representation of a draw is represented by $G = \sum_{k=1}^{\infty} \beta_k \delta_{\phi_k}$, where ϕ_k is an independent random variable, δ_{ϕ_k} is an atom located at ϕ_k and β_k is the weight. This representation of DP becomes a flexible transfer learning framework which is constructed by a HDPMM. The equations are given as

follows

$$\begin{aligned} G_0|\alpha_0, H &\sim \mathrm{DP}(\alpha_0, H), \\ G_{\mathcal{D}}|\gamma_{\mathcal{D}}, G_0 &\sim \mathrm{DP}(\gamma_{\mathcal{D}}, G_0), \\ z_{\mathcal{D}i}|G_d &\sim G_{\mathcal{D}}, \\ x_{\mathcal{D}i}|z_{\mathcal{D}i} &\sim \mathrm{F}(x_{\mathcal{D}i}, z_{\mathcal{D}i}), \end{aligned} \tag{6.4}$$

where $\mathcal{D}$ represents *domain* and z is an integer representing the cluster assignment. In these equations, F is a likelihood function and G_0 is the global base measure and $G_{\mathcal{D}}$ is the base measure for each *domain*. By inspecting the equation, H is a continuous base measure for the hyper layer, so that G_0 can draw discrete samples. The samples of G_0 can be used as a guide to set the same ϕ_k for $G_{\mathcal{D}}$. A plate diagram of the model is shown in Figure 6.1b.

6.4.2 Problem Definition Based on HBP Model

We formulate our model based on infrastructure failure prediction. Our example is based on the water supply network infrastructure which is critical for any city. We start with a given water pipe dataset, $X = \{x_{\ell j} \in \{0,1\}\}$ represents a sparse and noisy dataset with recorded failure history for all water pipe ℓ in year j. When a failure occurs, only a small section where the failure has occurred of the water pipe is repaired. The entire water pipe is not replaced with a new asset (i.e., newly repaired pipe does not mean that it has a lower chance of failure).

For CWM, the failure rate of these water pipes is extremely low (on average less than 0.5% per year). Therefore an assumption has been made that the failures are independent from each other, where the event of a failure is drawn from a Bernoulli process $x_{\ell j} \sim \mathrm{BerP}(\pi_{\ell j})$. $\pi_{\ell j}$ represents the predicted failure rate on a beta process which is a natural prior to the Bernoulli process. The plate notation of the Hierarchical Beta Process (HBP) is shown in Figure 6.1a and the algebraic formulation is given as follows

$$\begin{aligned} q_{kj} &\sim \mathrm{Beta}(c_0 q_0, c_0(1-q_0)), \\ \pi_{\ell j} &\sim \mathrm{Beta}\left(c_{kj} q_{kj}, c_{kj}\left(1-q_{kj}\right)\right), \\ x_{\ell j} &\sim \mathrm{Ber}\left(\pi_{\ell j}\right), \end{aligned} \tag{6.5}$$

where c_0 (concentration) and q_0 (mean) are the hyper-parameters of the upper-layer of the beta process. c_{kj} and q_{kj} are the latent parameters for the middle layer beta process. Each k represents a cluster of features, where each feature combination represents a group of pipes with the same features.

Making a prediction directly from the beta-Bernoulli process $P\left(\pi_{\ell j}\middle|x_{\ell j}\right)$ will have a high bias for sparse observations. HBP overcomes this issue by making a prediction for $\pi_{\ell j}$ using the clustering information and the individual observation to reduce the bias in the prediction, i.e., $P\left(\pi_{\ell j}\middle|q_{kj}X, c_0, q_0\right)$. However, HBP does not provide a method to learn q_{kj}. Previous approaches such as self-taught clustering [26], flexible grouping approach [64] and stochastic block models [48] are all prone to learning a bias in the model parameter q_{kj} in sparse datasets.

Our proposed approach attempts to reduce the bias using multi-task learning by learning q_{kj} from $P\left(q_{kj}\middle|X_{\mathcal{T}}, \{X_{\mathcal{S}}\}, c_0, q_0\right)$, where $\mathcal{T}$ represents the target (sparse) *domain* and $\mathcal{S}$ represents the source *domain*. An assumption used in multi-task learning is applied to the marginal probability distribution to enforce that the source and target *domains* are equal. Each *task* learnt in the model share a common parameter q_{kj}. The next section will discuss how we learn q_{kj} using a multi-task learning framework to reduce the bias in the model by learning the model parameters for a group of *tasks*.

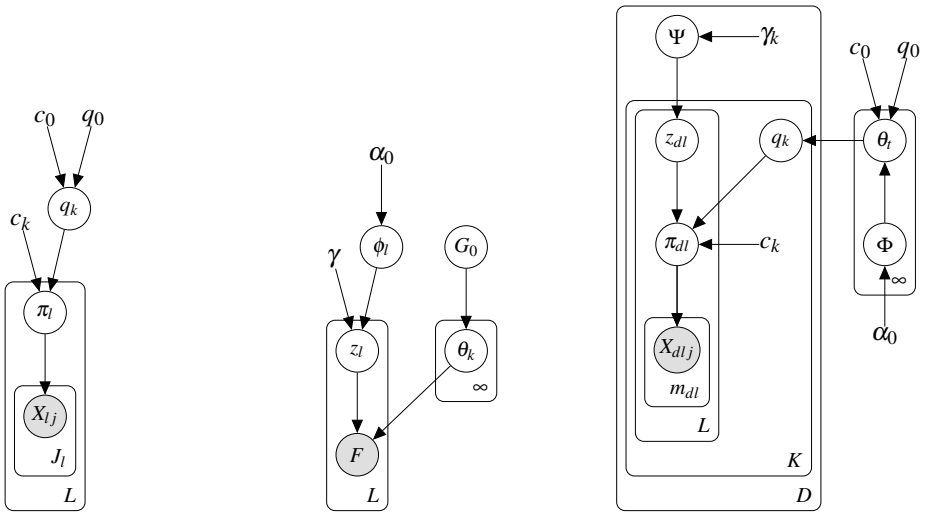


Fig. 6.1 Plate diagrams: (a) Hierarchical beta process (HBP), (b) Hierarchical Dirichlet process mixture model constructed using Chinese restaurant process (CRP), and (c) Our proposed hierarchical Dirichlet process mixture of hierarchical beta process (HDP-HBP).

6.4.3 Sharing Parameter Estimation with HDP-HBP model

We propose to use *hierarchical Dirichlet process mixture model* (HDPMM) to perform the clustering to group across *domains* and use the HBP as the mixture component to make the failure prediction. The aim of HDPMM framework is to perform cross *domain* clustering to

find a better representation of the structure of the failure patterns in a given network, where

$$\begin{aligned}
\Theta &\sim \text{Beta}\left(c_0 q_0, c_0\left(1-q_0\right)\right), \qquad \Theta = \{\theta_{1\ldots\infty}\}; \\
\Phi &\sim \text{DP}\left(\alpha_0\right), \qquad \Phi = \{\phi_{1\ldots\infty}\} \\
t &\sim \text{Mul}\left(\Phi\right); \\
\Psi_d &\sim \text{DP}\left(\gamma_d\right), \qquad \Psi_d \sim \left\{\psi_{d,1\ldots\infty}\right\}; \\
z_{d\ell} &\sim \text{Mul}\left(\psi_d\right); \\
q_{z_{dl}} &= \theta_t, \\
\pi_{d\ell} &\sim \text{Beta}\left(c_{z_{d\ell}} q_{z_{d\ell}}, c_{z_{d\ell}}\left(1-q_{z_{d\ell}}\right)\right), \\
X_{d\ell j} &\sim \text{Ber}\left(\pi_{d\ell}\right).
\end{aligned} \tag{6.6}$$

Formulating the problem in this way allows HDP-HBP to have the ability to learn different *tasks* from *domains* with different feature space $\mathfrak{X}$. In this formulation, each table in the CRP represents a *task*. Each *task* to learn its own model parameter q_k for its prediction function. The dishes from the global menu provides a constraint for sparse data to prevent assignment to new tables. Each *domain* has a parameter c_k which controls contribution between the beta-Bernoulli process and the group level from q_k. This allows parameter c_k to be used to control the variance of the clustering.

We will use the analogy being used to explain the CRP's representation of HDP (discussed in section 6.3.2) to explain how the information is transferred as follows. Our approach treats the denser *domain* as a restaurant which can assign new tables and add new dishes to the global menu shared amongst the restaurants. Both the dense and sparse *domains* are able to assign new dishes into the global menu. This gives the opportunity for both sparse and dense *domains* to create new clusters. The amount of clusters created by each *domain* is controlled by the concentration parameter. For this approach it is not required to know in advance which *domain* is sparse and which *domain* is dense.

6.4.4 Inference Algorithm

The objective of the inference algorithm is a combination of both clustering and estimating the latent feature rate in the target dataset. This can be seen as a maximum a posteriori (MAP) estimation problem, i.e., $P\left(\mathbf{z}, \mathbf{q} | \mathbf{X}_{\mathcal{T}}, \{\mathbf{X}_{\mathcal{S}}\}, c_0, q_0, \alpha_0, c_k, f\left(\cdot\right)\right)$. Gibbs sampling is used to

update the upper layer of the Dirichlet process (dish level), i.e.,

$$\begin{aligned} P\left(q_k=\theta_t|\mathbf{X}_{\mathcal{T}},\{\mathbf{X}_{\mathcal{S}}\},\alpha_0,-q_k,\Theta\right) &= \mathrm{CRP}\left(\alpha_0\right)\frac{P\left(x_{d\mathbf{l}},x_{d,-\mathbf{l}}\middle|q_t,c_t\right)}{P\left(x_{-\mathbf{l}}|q_t,c_t\right)} \\ &= \mathrm{CRP}\left(\alpha_0\right)\prod_{\ell\in\mathbf{l}}P\left(x_{d\mathbf{l}}|q_t,c_t\right) \end{aligned} \tag{6.7}$$

where $\mathbf{l}$ are a set of data points belonging to the same cluster. The mean of each cluster q_k is then determined by

$$P\left(z_{d\mathbf{l}}=k|q_{z_k},X_{\mathcal{T}},\{X_{\mathcal{S}}\},\mathbf{fl}_k,-\mathbf{z}_{d\mathbf{l}}\right)=\mathrm{CRP}\left(\gamma_k\right)\prod_{\mathbf{l}:z_{d\mathbf{l}}=k}P\left(x_{d\mathbf{l}}|q_k,c_k\right). \tag{6.8}$$

The likelihood functions $P\left(x_{\mathbf{l}}|q_k,c_k\right)$ and $P\left(x_{\mathbf{l}}|\theta,c_k\right)$ are easily solved because beta-Bernoulli are conjugate priors.

$$\begin{aligned} f\left(X_{d\ell,1\ldots m}\right) &= \int P\left(X_{d\ell,1\ldots m}|\pi_\ell\right)P\left(\pi_\ell|q_{z_{d\ell}},c_{z_{d\ell}},z_{d\ell}\right)d\pi_\ell \\ &= \int\prod_{j=1}^{m}P(X_{d\ell j}|\pi_\ell)P\left(\pi_\ell|q_{z_{d\ell}},c_{z_{d\ell}},z_{d\ell}\right)d\pi_\ell \\ &= \frac{\Gamma\left(c_{z_{d\ell}}\right)\Gamma\left(c_{z_{d\ell}}q_{z_{d\ell}}+\sum_{j=1}^{m}X_{d\ell j}\right)\Gamma\left(c_{z_{d\ell}}\left(1-q_{z_{d\ell}}\right)+m-\sum_{j=1}^{m}x_{d\ell j}\right)}{\Gamma\left(c_{z_{dl}}q_{z_{d\ell}}\right)\Gamma\left(c_{z_{d\ell}}\left(1-q_{z_{d\ell}}\right)\right)\Gamma\left(c_{z_{d\ell}}+m\right)} \end{aligned} \tag{6.9}$$

The value of q_k can be approximated by

$$P\left(q_k\middle|c_k,\delta_{z_{d\ell=k}},X_{d\ell,1\ldots m}\right)\sim\mathrm{Beta}\left(c_0q_0+\sum_{\ell}s_{d\ell},c_0\left(1-q_0\right)+\sum_{\ell}\sum_{t}^{m-s_{d\ell}-1}\frac{c_k}{c_k+t}\right). \tag{6.10}$$

An approximated Gibbs sampling technique outlined in Li *et al.* [59] is used to improve computational efficiency. The method uses a Taylor series expansion to approximate the posterior distribution

$$\begin{aligned} &P\left(q_k\middle|c_k,\delta_{z_{d\ell=k}},X_{d,\ell,1\ldots m}\right) \\ &\propto P\left(q_k,\delta_{z_{d\ell=k}},X_{d,\ell,1\ldots m}\middle|c_k\right) \\ &= P\left(q_k\right)P\left(\left\{X_{d,\ell,1\ldots m}\right\}_{\theta_{d\ell}=k}\middle|q_k,c_k,\{\theta_{d\ell}\}=k\right) \\ &\approx \frac{\Gamma\left(c_0\right)q_k^{c_0q_0-1}\left(1-q_k\right)^{c_0(1-q_0)-1}}{\Gamma\left(c_0q_0\right)\Gamma\left(c_0\left(1-q_0\right)\right)}\prod_{d,\ell,\theta_{dl}=k}\left[\frac{\left(c_kq_k\right)^{s_{dl}}\prod_{t=0}^{m-s_{d\ell}-1}\left(c_k\left(1-q_k\right)+t\right)}{\prod_{t=0}^{m-1}\left(c_k+t\right)}\right]. \end{aligned} \tag{6.11}$$

It is based on the sparse nature of the dataset, where $s_{d\ell} = \sum_{d,j} X_{d,\ell,j}$ and m is the number of observations.

$$P\left(q_k \middle| c_k, \delta_{z_{d\ell=k}}, X_{d\ell,1\ldots m}\right) \propto q_k^{c_0 q_0 - 1}\left(1-q_k\right)^{c_0(1-q_0)-1} \prod_{d,\ell,\theta_{d\ell}=k}\left[\left(\frac{c_k q_k}{c_k+m-1}\right)^{s_{d\ell}}\left(1-q_k\right)^{\sum_{t=0}^{m-s_{d\ell}-1}\frac{c_k}{c_k+t}}\right]. \tag{6.12}$$

By inspecting the results in Equation (6.12), the posterior distribution of q_k is a beta distribution, i.e.,

$$P\left(q_k \middle| c_k, \delta_{z_{d\ell=k}}, X_{d\ell,1\ldots m}\right) \sim \text{Beta}\left(c_0 q_0 + \sum_{\ell} s_{d\ell}, c_0\left(1-q_0\right) + \sum_{\ell}\sum_{t}^{m-s_{d\ell}-1}\frac{c_k}{c_k+t}\right). \tag{6.13}$$

Finally π can be sampled directly from its conditional distribution by using the group index generated by $\theta_{d\ell}$ as follows

$$P\left(\pi_{d\ell} \middle| q_{z_{d\ell}}, z_{d\ell}, c_{z_{d\ell}}, X_{d\ell,1\ldots m}\right) \sim \text{Beta}\left(c_{z_{d\ell}} q_{z_{d\ell}} + s_{d\ell}, c_{z_{d\ell}}\left(1-q_{z_{d\ell}}\right) + m - s_{d\ell}\right). \tag{6.14}$$

All of the updating steps outlined above are repeated iteratively until convergence has been reached. The inference algorithm above does not outline a method to select the parameters for c_k.

From Equation (6.14), the variance of the beta distribution is given by

$$\begin{aligned} \tilde{\alpha} &= c_{z_{d\ell}} q_{z_{d\ell}} + s_{d\ell} > 0, \\ \tilde{\beta} &= c_{z_{d\ell}}\left(1-q_{z_{d\ell}}\right) + m - s_{d\ell} = c_{z_{d\ell}} + m - \alpha > 0, \end{aligned} \tag{6.15}$$

where, $c_{z_{d\ell}} > 0$, $0 < q_{z_{d\ell}} < 1$, $s_{d\ell} \geq 0$, $m > 0$ and $m \geq s$. The variance of the beta distribution is given by

$$\text{var}\left[\pi\right] = \frac{\tilde{\alpha}\tilde{\beta}}{\left(\tilde{\alpha}+\tilde{\beta}\right)^2\left(\tilde{\alpha}+\tilde{\beta}+1\right)} \tag{6.16}$$

Taking the first derivative to find the gradient of the variance with respect to β,

$$\begin{aligned} \frac{d\left(\text{var}\left[\pi\right]\right)}{d\tilde{\beta}} &= \frac{\tilde{\alpha}\left(\tilde{\alpha}\left(\tilde{\alpha}+1\right) - \tilde{\beta}\left(\tilde{\alpha}+2\tilde{\beta}+1\right)\right)}{\left(\tilde{\alpha}+\tilde{\beta}\right)^3\left(\tilde{\alpha}+\tilde{\beta}+1\right)^2} \\ &\propto \tilde{\alpha}\left(\tilde{\alpha}+1\right) - \tilde{\beta}\left(\tilde{\alpha}+2\tilde{\beta}+1\right). \end{aligned} \tag{6.17}$$

Since $\frac{d\tilde{\alpha}}{dm} = 0$, and $\frac{d\tilde{\beta}}{dm} = 1$, therefore

$$\begin{aligned}\frac{d\left(\mathrm{var}\left[\pi\right]\right)}{dm} &= \frac{d\left(\mathrm{var}\left[\pi\right]\right)}{d\tilde{\alpha}}\frac{d\tilde{\alpha}}{dm} + \frac{d\left(\mathrm{var}\left[\pi\right]\right)}{d\tilde{\beta}}\frac{d\tilde{\beta}}{dm} \\ &= \frac{d\left(\mathrm{var}\left[\pi\right]\right)}{d\tilde{\beta}}.\end{aligned} \tag{6.18}$$

Substituting $\tilde{\alpha}$ and $\tilde{\beta}$ and finding the maximum of $\frac{d(\mathrm{var}[\pi])}{dm}$ with respect to m,

$$\max_m \left[\frac{d\left(\mathrm{var}\left[\pi\right]\right)}{dm}\right] = \frac{1}{4}\left(q_{z_{d\ell}} + s - c_{z_{d\ell}} - 1\right). \tag{6.19}$$

By inspecting Equation (6.19), if $c_{z_{d\ell}} \geq s$, then the variance is monotonically decreasing as the number of data points increases because $0 < q_{z_{d\ell}} < 1$.

From this result a reasonable choice for the concentration parameter should be $c_{z_{d\ell}} \geq m$. The value to select for $c_{z_{d\ell}}$ is non-trivial and is subjected to further research in future work. For our experiment, the value of $c_{z_{d\ell}}$ is fixed to m as $s \leq m$ to ensure that there is a decreasing variance with an increasing number of observations.

6.5 Experiments

This section presents the results of the experiments for our proposed model HDPMM. Our experimental results is comprised of three sections. We first design a synthetic experiment to observe the sensitivity of the parameters in our model. We then demonstrate that our proposed approach outperforms all other heuristics previously used for HBP. Then finally, we demonstrate the capability of our proposed HDP-HBP model by comparing it with other state-of-the-art techniques in failure prediction, transfer learning and survival analysis.

6.5.1 Synthetic Data

For our synthetic data experiment, we attempt to create a typical dataset from a utility company to make infrastructure failure prediction. We create a synthetic infrastructure dataset with categorical features that has 512 unique features combinations and assign a true failure rate $P_{\text{true\_fr}}$ to each of these features using a beta distribution. Another beta distribution to assign the true distribution on unique features $P_{\text{true\_f}}$. Samples are then from $P_{\text{true\_f}}$ and then draw the samples for the observations for each feature sample from $P_{\text{true\_fr}}$. A dense *domain* with sample size $N = 10^6$ is also created using the same marginal probability distribution.

6.5.2 Real-world Implementations of Failure Prediction

There have been many implementations to improve water pipe maintenance in the past 40 years. The first known implementation is in Calgary, Alberta in Canada, where they have used a Poisson point process to assist in scheduling maintenance [96]. However, survival analysis is not an effective long-term solution to the scheduling maintenance problem because once a component has been replaced, it is no longer considered in the model. After that, a number of physical based models have been used across Canada and in Adelaide, Australia [70]. They have been attempted to model the physical interactions between the soil pressure, traffic loading, frost loads, operational pressure and third party inferences. These approaches have proven to be effective; however, it is often that the needed data are either unavailable or they are very costly to acquire [50].

More recently, statistical approaches have been implemented by a research group in Sydney, NSW, Australia [58]. While, the city of Syracuse, New York, United States of America, has proposed some solutions to use machine learning models to proactively repair the critical components [53]. The machine learning solutions are able to effectively find failure patterns with limited amount of data. However, tuning the parameters for these machine learning models are challenging. Our study also aims to address this issue.

6.5.3 Failure Predictions for Sparse Data

In the past, there have been a number of physical models [49, 68] developed to assist in preventative maintenance. These models work by modelling the physical interactions to prioritise water pipes with a higher probability of failure for manual inspection. Some examples of the physical interactions are the interaction between the soil and the material, the age of the pipe, effects from the weather such as frost loading, soil pressure and the water supply demand. Building physical models to model real world-interactions have shown to be a very effective method to accurately predict component failure. However, it requires a lot of data which is often infeasible or extremely costly to collect. For more details, a comprehensive review of methods to model structural deterioration of water mains can be found in [50, 70].

A variety of statistical models have also been explored to make failure prediction. Statistical models aim to make the best possible prediction using the available data. The earliest work using statistical models found in literature uses a Poisson point process to schedule the optimal time for the water main to be repaired or replaced by forecasting the number of breaks in the existing network [96]. Later on, more advanced survival analysis techniques have been used to allow multi-variant analysis to assist in scheduling maintenance, such as

the Cox proportional hazard model [29, 60] and Weibull models [44]. Survival analysis is often used to assist in scheduling maintenance for components with a higher risk of failure. However, survival analysis may not be well suited for this *task* because, in practice, only a small section of the component is repaired instead of replacing the entire asset. Since the majority of the component has not changed, a survival analysis approach may not be the best option because parts of the original characteristics still remains for the repaired component.

Traditional machine learning techniques such as gradient boosted decision trees [33] have shown to be effective when there is availability to high-quality data [53]. Traditional machine learning approaches use features from historical studies such as pipe material, pipe age, soil type, toxicity level of the soil, geological composition, and topography as potential factors that affect the lifespan of water mains [47, 55, 67, 71, 82, 96]. However, currently, it is often difficult to obtain a high-quality dataset to make predictions. Oftentimes, the dataset will contain missing data, noisy labels, few training examples and have a short observation period making it difficult to apply the traditional machine learning approaches.

More recent formulations of infrastructure failure prediction attempt to estimate the latent failure rate given the historical failure records for each asset. Unlike the approaches used in survival analysis, this approach does not assume a 'life' for each asset. But instead, these models groups assets using features to estimate the failure rate. An example is the hierarchical beta process [59, 104], where the failure rates are estimated by using a combination of individual observations for each pipe and the average group observations. This approach is a more effective solution for lower data quality such as missing data and short observation period. However, it requires a heuristic for grouping water pipes with similar characteristics together.

When the latent failure model was originally proposed, a heuristic provided by domain experts was used to group water pipes with similar characteristics together based on the intrinsic pipe failures [59]. There have been some proposals to use Bayesian non-parametric clustering as a heuristic to automatically group the water pipes for HBP, such as the Dirichlet Process Mixture Model (DPMM) [103] and the Pitman-Yor process [83]. These methods adopt a two-step approach which has separate models for clustering and failure prediction. Subsequently, more advanced techniques have been employed to combine the clustering heuristic with the failure prediction model by using stochastic block models [64] and DPs [60]. However, none of these methods are able to make accurate predictions for extremely sparse components, such as a critical water main, where there are only very few components for any water pipe network. Therefore, applying machine learning models on a single dataset may not be effective as there is not enough data to make an effective prediction. Our proposed

approach aims at resolving this issue by automatically selecting the model-parameters by clustering across *domains* to learn a common *task* for the failure prediction model.

We create scenarios using a beta distribution with parameters ($\alpha = 0.5$, $\beta = 10$), ($\alpha = 0.5$, $\beta = 5$) and ($\alpha = 0.5$, $\beta = 1$) to simulate the failure observations in a water pipe network. These synthetic datasets correspond with water pipes with a mean failure rate of 4.76%, 9.09% and 33.3%. For each sample size, we create 20 independent datasets and take the mean value of the mean averaged error (MAE). The result in Figure 6.2b has shown that multi-task learning is able to produce a lower error prediction compared to DPMHBP for the single *task*. As the size of the dataset increases, the error of just using a single *domain* reduces to the same order of magnitude as using multi-task learning. This is because for large sample size, the dataset is no longer considered sparse, therefore there is less of an improvement when using multi-task learning. However, as we increase the failure rate, the irreducible error in multi-task learning becomes more dominant, therefore for higher failure rates, using a single *domain* may produce better results as observed in Figure 6.2d. The synthetic results have shown that using the multi-task learning approach is only effective for the special case of sparse predictions.

A sample size of $N = 10^4$ to study the sensitivity of the concentration parameter α_0, γ_k and c_k. For this experiment, we set $\alpha_0 = \gamma_k$ for simplicity and all values of c_k are equal. The parameters are then normalised by dividing c_k by the number of observations and α_0 and γ_k are normalised by dividing with the number of features. Figure 6.2a is the experimental results for the sensitivity of the hyper-parameters. We observed that the hyper-parameters in HDP-HBP are non-convex, with α with values of 0.01 and 0.1 producing results closest to the global optimal. While $\alpha = 100$ may be easier to tune the hyper-parameter c_k. This is because when α increases, there are few number of data points in each cluster, therefore the value of c_k becomes less relevant as it controls the contribution between the beta-Bernoulli process and the group level for each domain. However, for lower values of α the values of c_k is much more critical to make a correct prediction.

Table 6.1 Summary of pipe network and pipe failure data

Region	Type	No. Pipes	No. Failures	Avg. Failure Per Year	Total Length[m]
Region A	RWM	23 926	3 782	0.99%	879 050
	CWM	2 945	182	0.39%	229 915
Region B	RWM	62 089	10 717	1.08%	3 878 255
	CWM	7 119	435	0.38%	671 884
Region C	RWM	45 030	10 719	1.49%	2 593 412
	CWM	5 001	606	0.76%	477 094

Table 6.2 Comparing Area Under Curve (AUC) for heuristic approaches.

		Dense-to-Sparse				
		HDP-HBP	DPMHBP	HBP(Expert)	HBP(DPMM)	HBP(HPY)
AUC	Region A	**67.00%**	62.45%	62.38%	62.45%	60.90%
	Region B	**89.46%**	84.10%	86.62%	55.02%	54.00%
	Region C	**87.81%**	86.93%	87.24%	86.38%	85.78%

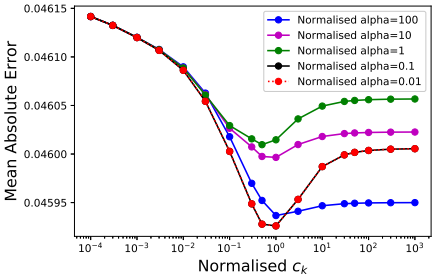
Table 6.3 Comparing Area Under Curve (AUC) for alternative approaches.

		Dense-to-Sparse					
		HDP-HBP	DPMHBP	HBP	Cox	SVM	Boost
AUC	Region A	**67.00%**	62.45%	62.38%	60.94%	61.07%	66.19%
	Region B	**89.46%**	84.10%	86.62%	63.33%	77.91%	63.95%
	Region C	87.81%	86.93%	87.24%	75.96%	82.98%	**89.92%**

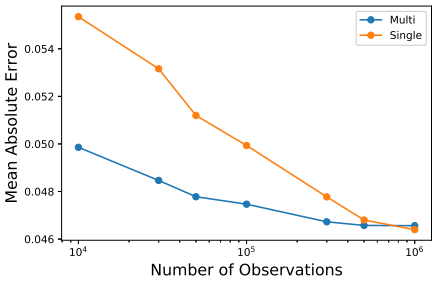
6.5.4 Case Study: Water Pipe Failure Prediction

The data collected for each metropolitan area includes the water pipe network data and water pipe failure data. Three sets of data are used and referred to as Region A (a central business district, population:210,000, area:25km^2), Region B (a prominent suburb, population:230,000, area:80km^2) and Region C (a large rural city, population:205,000, area:685km^2). The water pipe network data contains all pipe information in the network. This consists of a unique pipe ID to identify the pipe, pipe attributes and location. The pipe attributes represent intrinsic features belonging to each pipe. These include the year the pipe is laid, protective coating, diameter, material and methods to join pipe segments together. These particular features are factors which may be correlated to the failure rate. Additionally, there are also external factors such as, tree canopy converge, soil corrosion, water supply demand and temperature which are also identified as external factors which may be correlated with the failure rate. The failure data is a time series data which contains the pipe ID, failure dates and the failure location. A new record is added to the dataset each time a failure has been observed in the water supply network.

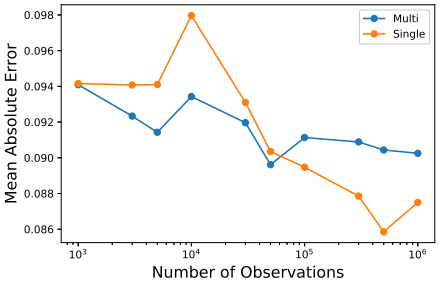
For the dataset collected for the three metropolitan areas, the pipes are laid between years 1884 - 2011. These pipes are often split into two main categories, reticulation water main (RWM) and critical water main (CWM). The categories of these pipes are defined by domain experts according to the impact of each water main, such as the financial cost of compensation and the social effect during an event of a failure. The ratio of these pipes for each region is summarized in Table 6.1. As the role of RWM and CWM is different, it is expected that the failure behaviour of RWM and CWM is also different. Therefore, RWM



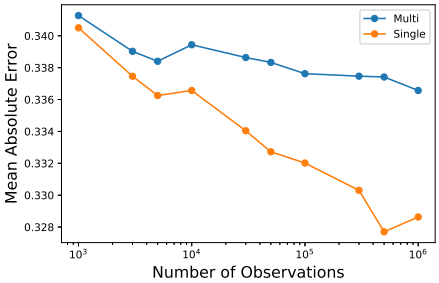
(a) Sensitive of the hyper-parameters for a sample size of $N = 10^4$.



(b) Beta distribution with parameters $\alpha = 0.5, \beta = 10$. Mean failure rate 4.76%.

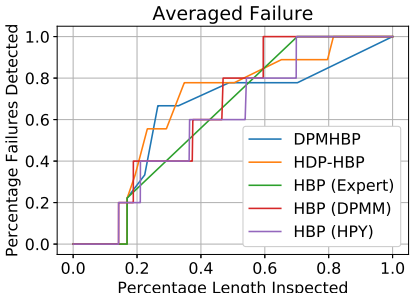


(c) Beta distribution with parameters $\alpha = 0.5, \beta = 5$. Mean failure rate 9.09%.

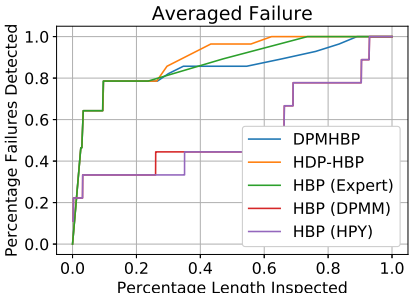


(d) Beta distribution with parameters $\alpha = 0.5, \beta = 1$. Mean failure rate 33.3%.

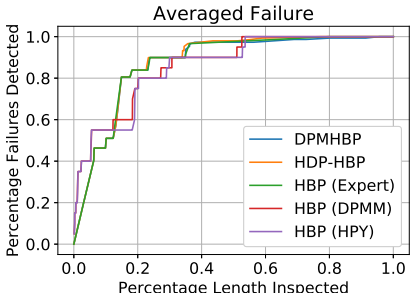
Fig. 6.2 Synthetic experiment results for varying sparsity ratio.



(a) Region A

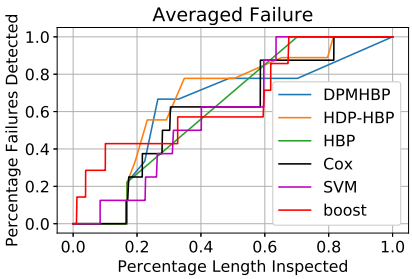


(b) Region B

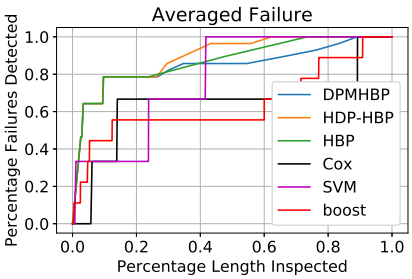


(c) Region C

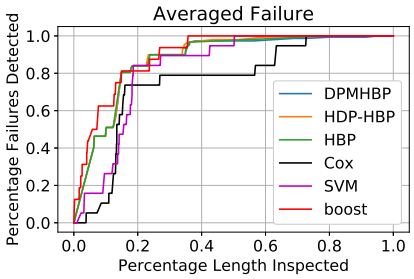
Fig. 6.3 Comparison of failure prediction results for each region in 2012 for heuristic methods.



(a) Region A



(b) Region B



(c) Region C

Fig. 6.4 Comparison of failure prediction results for each region in 2012 with other state-of-the-art approaches.

and CWM are considered as separate *domains*. Quite typically, the water supply network is comprised of approximately 10% CWM and 90% RWM. The average failure rate each year for the CWM across the entire network is approximately 0.5%. The failure history recorded has an observation period of 16 years spanning from 1997 to 2012 which is very short given the failure rate. Due to a combination of these factors the dataset used to make failure predictions on water pipe is extremely sparse.

Our experiments focus on a dense-to-sparse transfer scenario. The dense-to-sparse scenario attempts to improve the learning performance by transferring information from RWM to CWM. RWM is considerably denser compared to CWM as there are fewer pipes and failures observed, which makes it much more difficult for a model to use a single CWM dataset to learn the failure patterns as there are fewer training examples. We demonstrate the ability for HDP-HBP to use the information from RWM and CWM to improve the failure prediction performance.

In the following sections we first compare our approaches with other potential heuristics for HBP such as Dirichlet process mixture model (DPMM) and the Hierarchical Pitman-Yor (HPY) process. We then compare our approach with other failure prediction methods used in survival analysis, transfer learning and traditional machine learning approach. They are Cox proportional hazard model, transfer learning transfer adaptive boosting (boost) and support vector machine (SVM), respectively.

Comparing Heuristic Methods for HBP

In this section we present the experimental results on different heuristics for HBP grouping. In Figure 6.3, we compare our HDP-HBP approach with Dirichlet Process Mixture of Hierarchical Beta Process (DPMHBP), heuristics provided by domain experts, and clustering by Dirichlet Process Mixture Model (DPMM) [103] and Hierarchical Pitman-Yor process (HPY) process [83]. We quantitatively compare these results in Table 6.2 by calculate the Area Under Curve (AUC). HDP-HBP clearly outperform all other approaches.

Comparison with Alternative Approaches

This section presents the experimental results to compare our HDP-HBP approach with alternative approaches for sparse failure prediction. We observe in Figure 6.4 that the experimental results show that the behavior of HBP based models are similar on the left-hand side due to the contribution from the beta-Bernoulli process. If there is a high number of observations of failure for the water pipe, these water pipes are generally ranked with the

highest probability of failure in each cluster. As a result, on the left-hand side all variants of HBP model, HBP, DPMHBP and HDP-HBP have very similar failure prediction performance.

For the HBP approach, we have used a heuristic provided by domain experts for grouping. Our experimental results have shown that the grouping by domain experts have typically performed better than using two-stage approach for clustering and failure predictions, e.g. DPMM or HPY process. To avoid clutter on our experimental results, we have only included the heuristic provided by the domain expert produced the best result.

In Figure 6.4, Region A has shown to have detected very little failure across all HBP models. Manually inspecting these water pipes in the dataset, these water pipes have historically had a high failure rate. Towards the right-hand side, these pipes generally have either no observations or very few observations of failure during its observation period. In this region, our proposed model is shown to perform much better by providing better clustering and parameter selection. In Region A and Region B, HDP-HBP has shown that it is superior in learning the model parameters for each group of *tasks*. The AUC can be used to quantify the average model performance. The AUC for each model is shown in Table 6.3. HDP-HBP clearly outperforms all other models, with the exception of Region C, where transfer adaptive boosting (Boost) have slightly outperformed our approach. However, in Region A and Region B, HDP-HBP have outperform Boost by a large margin.

6.6 Summary

We have proposed a novel multi-task learning solution by *Bayesian non-parametric* formulation. We have constructed a model which can automatically learn the model parameters for a group of *tasks* to improve failure prediction accuracy on sparse datasets. Most importantly, our proposed model achieve the improvement by leveraging the data obtained from other *domains* with different feature space via empirical Bayes. The cross *domain* clustering method learns a common latent structure which is then used as the model parameters for failure prediction for each *task*. We have demonstrated how cross *domain* clustering can be obtained by using *hierarchical Dirichlet process mixture model* (HDPMM) and then illustrated how it can be integrated with *hierarchical beta process* (HBP) by only two separate steps. We have then shown that this approach can be further improved by combining clustering and prediction models together to create a new framework, namely *hierarchical Dirichlet process mixture of hierarchical beta process* (HDP-HBP). Experimental results confirmed that our proposed approach is able to outperform other known heuristics which have been used for grouping in HBP. In additional, our proposed learning and prediction framework has been able to improve upon previous state-of-the-art methods with higher prediction accuracy. It offers an improved

capability for utility companies to proactively inspect critical infrastructure components with sparse failure history instead of repairing failures reactively. Notably, the improvement to prediction accuracy in infrastructure preventive maintenance can provide significant financial savings in a city and reduce adverse social impact from unwanted failures.

This Chapter has demonstrated the feasibility of using Bayesian non-parametrics to automatically select the number of parameters in the model for complex real-world problems. Currently the models proposed in Chapter 3, Chapter 4 and Chapter 5 is unable to perform feature selection to reduce the error caused by an increased number of parameters. An interesting direction for future work is to incorporate these Bayesian non-parametric techniques to the information geometric formulation of the binary log-linear model to automatically select the useful parameters in the model.

CHAPTER 7

Conclusion

7.1 Contributions

This thesis has made contributions in furthering the understanding of increasing the representational power in unsupervised learning. In summary, these are:

1. In Chapter 3 we have presented a novel framework to allow higher-order feature interactions to be included into hierarchical models. The model uses an information geometric perspective to formulate the binary log-linear model. The binary log-linear model has been used to formulate the Higher-Order Boltzmann Machine (HBM) as a representative example to include higher-order features in machine learning models. The higher-order interaction is formulated using incidence algebra which provides a provides a mathematical expression for a generative function for combinatorics. The HBM can be used to include higher-order interactions into the model or to estimate high dimensional functions.

2. Chapter 3 proposed the information geometric formulation of the HBM for efficient inference algorithm of discrete structures by projecting it onto a dually flat Riemannian manifold. This means that the discrete structure can be represented as two continuous convex functions so that gradient descent or natural gradient can be applied for optimization. This is much more efficient and accurate compared to the previous approaches such as MCMC, contrastive divergence or variational inference.

3. Chapter 3 provides the first study on the different characteristics in increasing the representational power in machine learning models when using hidden nodes and higher-order feature interactions. Our study is performed by using a bias-variance decomposition to determine the error in the model. We have found that using higher-order features have lower model error when fitting sparse data.

4. Chapter 4 uses the concepts shown in Chapter 3 to formulate a novel framework for blind source separation (BSS) to include higher-order interaction between a set of source signals. We do this by designing a hierarchical structure by using the concepts presented in Chapter 3, where the source signal, received signal and the mixing matric are realized in different layers. We transform our hierarchical structure by projecting it onto a dually flat Riemannian manifold to take advantage of the theoretical guarantees found in information geometry. Our proposed framework has highly desirable properties for BSS such as including more complex interactions between the set of source signal and formulating it as a convex optimization problem where our solution always arrives at a unique solution.

5. Chapter 5 uses the concepts shown in Chapter 3 to provide an efficient solution to estimate higher-order intensity function in Poisson processes. Our solution is design a hierarchical structure to include the higher-order interaction terms to estimate the higher-order intensity function in the multivariate Poisson process. Our structure projections the samples into lower dimensional space to estimate the higher-order functions. Our results are back by strong theoretical guarantees in information geometry and the empirical results have demonstrated many advantages over current state-of-the-art techniques such the ability to estimate the joint intensity function with extremely low event rate and sparse observations.

6. Chapter 6 explores techniques for automatic select the number of parameters in the model. The proposed framework Hierarchical Dirichlet Process Mixture of Hierarchical beta Process (HDP-HBP) demonstrated the ability for Bayesian non-parametrics to automatically select the higher-order interaction terms to include into the model. The proposed approach works effectively in real-world application with poor data quality such as sparsity, noisy labels and missing data. But further work is required to include automatic parameter selection in our information geometric formulations.

7.2 Future Work

This thesis has laid out the foundations to understand to increase the representational power in unsupervised learning. There are still many open ended research questions which can still be explored for the information geometric formulation of the binary log-linear model. Here are some interesting future research directions which may be worth exploring.

1. Feature selection and model selection – Our proposed approach is to use a truncated number of parameters based on the order of interactions to avoid combinatorial explosion. However, not all the models included are important for the model. Reducing the number of parameters in the model will reduce the noise and variance in the model. We have demonstrated the ability for Bayesian non-parametric models to automatically select the number of parameters in the model. Currently, our model does not include any approach to select the most important features. Combining Bayesian non-parametrics to automatically select the number of parameters will improve both the performance of the model and its scalability.

2. Efficient scalable optimization technique – we have proposed to use gradient descent and natural gradient for optimization. These techniques are extremely efficient, however do not scale well with respect to the number of parameters. Natural gradient is able to reach the global optimal in just a few iterations, however, to invert the fisher information matrix, the model scales cubically. On the other hand, gradient descent is much more efficient in terms of the run-time complexity as it scales quadratically, but requires more iterations to converge. For all applications we have applied the log-linear model, our empirical results have shown that natural gradient is more efficient for optimization in our applications. However, this may not be the case for applications which requires a large number of model parameters. Currently, for models with a large number of parameters the most efficient optimization to use is gradient descent. It would be desirable to design more efficient optimization techniques similar to natural gradient which are more scalable to use for large parameter space.

3. Introducing hidden nodes to the binary log-linear model – Currently our formulation provides a convenient approach to include higher-order interactions effects into the model. Our model could be extended to include hidden nodes to incorporate latent factors into the model. The behaviour of using both hidden nodes and higher-order interactions effects is still not very well explored in machine learning and could potential have many highly desirable properties in many applications.

4. Real-world applications – The models proposed for blind source separation (BSS) and Poisson processes have only been applied to toy examples to demonstrate its capabilities from a theoretical perspective. It would be interesting to apply these models to real-world applications. An example of an application for IGBSS are electroencephalography (EEG) and magnetoencephalography (MEG). EEG and MEG use sensors to monitor electrical activity of the brain. However, there are often other noises which create artefacts in the signal such as blinking, eye muscle movement, facial muscle and cardiac signals which are mixed into the recorded signal. Multivariate Poisson processes have many real-world applications which could be explored, such as infrastructure failure prediction, disease modeling and in transportation. A simple example for a multivariate Poisson process in infrastructure failure prediction is the following. There are two safety mechanisms to prevent a failure of a component in an infrastructure network. The chances of a single safety mechanism failing is common, however, it is very rare for both failure mechanisms to fail at the same time. Additive Poisson Process (APP) is able to estimate the intensity function for both safety mechanisms failing at the same time leading to the failure of the component to make a more accurate prediction.

References

[1] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1987). A learning algorithm for Boltzmann machines. In *Readings in Computer Vision*, pages 522–533. Elsevier.

[2] Agresti, A. (2003). *Categorical data analysis*, volume 482. John Wiley & Sons.

[3] Agresti, A. (2012). *Categorical Data Analysis*. Wiley, 3 edition.

[4] Amari, S. (2001a). Information geometry on hierarchy of probability distributions. *IEEE Transactions on Information Theory*, 47(5):1701–1711.

[5] Amari, S. (2009). Information geometry and its applications: Convex function and dually flat manifold. In Nielsen, F., editor, *Emerging Trends in Visual Computing: LIX Fall Colloquium, ETVC 2008, Revised Invited Papers*, pages 75–102. Springer.

[6] Amari, S. (2016a). *Information Geometry and Its Applications*. Springer.

[7] Amari, S.-I. (1998). Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276.

[8] Amari, S.-I. (2001b). Information geometry on hierarchy of probability distributions. *IEEE Transactions on Information Theory*, 47(5):1701–1711.

[9] Amari, S.-i. (2016b). *Information geometry and its applications*, volume 194. Springer.

[10] Ay, N., Gibilisco, P., and Matus, F. (2018). *Information Geometry and Its Applications*. Springer.

[11] Barron, A. and Hengartner, N. (1998). Information theory and superefficiency. *The Annals of Statistics*, 26(5):1800–1825.

[12] Bell, A. J. and Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural computation*, 7(6):1129–1159.

[13] Berne, O., Joblin, C., Deville, Y., Smith, J., Rapacioli, M., Bernard, J., Thomas, J., Reach, W., and Abergel, A. (2007). Analysis of the emission of very small dust particles from spitzer spectro-imagery data using blind signal separation methods. *Astronomy & Astrophysics*, 469(2):575–586.

[14] Bishop, C. M. et al. (1995). *Neural networks for pattern recognition*. Oxford university press.

[15] Bonilla, E. V., Chai, K. M. A., and Williams, C. K. (2007). Multi-task gaussian process prediction. In *NIPs*, volume 20, pages 153–160.

[16] Boutsidis, C. and Gallopoulos, E. (2008). Svd based initialization: A head start for nonnegative matrix factorization. *Pattern recognition*, 41(4):1350–1362.

[17] Braun, J. (2009). *An application of Kolmogorov's superposition theorem to function reconstruction in higher dimensions*. PhD thesis, Universitäts-und Landesbibliothek Bonn.

[18] Braun, J. and Griebel, M. (2009). On a constructive proof of kolmogorov's superposition theorem. *Constructive approximation*, 30(3):653.

[19] Buja, A., Hastie, T., Tibshirani, R., et al. (1989). Linear smoothers and additive models. *The Annals of Statistics*, 17(2):453–510.

[20] Buntine, W. and Hutter, M. (2010). A bayesian view of the poisson-dirichlet process. *arXiv preprint arXiv:1007.0296*.

[21] Cardoso, J.-F. (1999). High-order contrasts for independent component analysis. *Neural computation*, 11(1):157–192.

[22] Censor, Y. and Lent, A. (1981). An iterative row-action method for interval convex programming. *Journal of Optimization theory and Applications*, 34(3):321–353.

[23] Comon, P. (1994). Independent component analysis, a new concept? *Signal processing*, 36(3):287–314.

[24] Congedo, M., Gouy-Pailler, C., and Jutten, C. (2008). On the blind source separation of human electroencephalogram by approximate joint diagonalization of second order statistics. *Clinical Neurophysiology*, 119(12):2677–2686.

[25] Cox, D. R. (1984). Interaction. *International Statistical Review/Revue Internationale de Statistique*, pages 1–24.

[26] Dai, W., Yang, Q., Xue, G.-R., and Yu, Y. (2008). Self-taught clustering. In *Proceedings of the 25th international conference on Machine learning*, pages 200–207. ACM.

[27] Daley, D. J. and Vere-Jones, D. (2007). *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media.

[28] Davey, B. A. and Priestley, H. A. (2002). *Introduction to Lattices and Order*. Cambridge University Press.

[29] David, C. R. et al. (1972). Regression models and life tables (with discussion). *Journal of the Royal Statistical Society*, 34:187–220.

[30] Ding, C. H., Li, T., and Jordan, M. I. (2008). Convex and semi-nonnegative matrix factorizations. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):45–55.

[31] Flaxman, S., Teh, Y. W., Sejdinovic, D., et al. (2017). Poisson intensity estimation with reproducing kernels. *Electronic Journal of Statistics*, 11(2):5081–5104.

[32] Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The Elements of Statistical Learning*. Springer.

[33] Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378.

[34] Friedman, J. H. and Stuetzle, W. (1981). Projection pursuit regression. *Journal of the American statistical Association*, 76(376):817–823.

[35] Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741.

[36] Gierz, G., Hofmann, K. H., Keimel, K., Lawson, J. D., Mislove, M., and Scott, D. S. (2003). *Continuous Lattices and Comains*, volume 93. Cambridge University Press.

[37] Grosse, R. B., Maddison, C. J., and Salakhutdinov, R. R. (2013). Annealing between distributions by averaging moments. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2769–2777.

[38] Gupta, S., Phung, D., and Venkatesh, S. (2013). Factorial multi-task learning: a bayesian nonparametric approach. In *International conference on machine learning*, pages 657–665.

[39] Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800.

[40] Hinton, G. E. (2012). A practical guide to training restricted Boltzmann machines. In *Neural Networks: Tricks of the Trade*, pages 599–619. Springer.

[41] Hjort, N. L. et al. (1990). Nonparametric bayes estimators based on beta processes in models for life history data. *The Annals of Statistics*, 18(3):1259–1294.

[42] Huelsenbeck, J. P., Jain, S., Frost, S. W., and Pond, S. L. K. (2006). A dirichlet process model for detecting positive selection in protein-coding dna sequences. *Proceedings of the National Academy of Sciences*, 103(16):6263–6268.

[43] Hyvärinen, A. and Oja, E. (2000). Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430.

[44] Ibrahim, J. G., Chen, M.-H., and Sinha, D. (2005). *Bayesian survival analysis*. Wiley Online Library.

[45] Ilalan, D. (2016). A poisson process with random intensity for modeling financial stability. *The Spanish Review of Financial Economics*, 14(2):43–50.

[46] Isomura, T. and Toyoizumi, T. (2016). A local learning rule for independent component analysis. *Scientific reports*, 6:28073.

[47] Kabir, G., Tesfamariam, S., and Sadiq, R. (2015). Predicting water main failures using bayesian model averaging and survival modelling approach. *Reliability Engineering & System Safety*, 142:498–514.

[48] Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., and Ueda, N. (2006). Learning systems of concepts with an infinite relational model. In *AAAI*, volume 3, page 5.

[49] Kettler, A. and Goulter, I. (1985). An analysis of pipe breakage in urban water distribution networks. *Canadian Journal of Civil Engineering*, 12(2):286–293.

[50] Kleiner, Y. and Rajani, B. (2001). Comprehensive review of structural deterioration of water mains: statistical models. *Urban water*, 3(3):131–150.

[51] Kolmogorov, A. N. (1957). On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. In *Doklady Akademii Nauk*, volume 114, pages 953–956. Russian Academy of Sciences.

[52] Kottas, A. (2006). Dirichlet process mixtures of beta distributions, with applications to density and intensity estimation. In *Workshop on Learning with Nonparametric Bayesian Methods, 23rd International Conference on Machine Learning (ICML)*, volume 47.

[53] Kumar, A., Rizvi, S. A. A., Brooks, B., Vanderveld, R. A., Wilson, K. H., Kenney, C., Edelstein, S., Finch, A., Maxwell, A., Zuckerbraun, J., et al. (2018). Using machine learning to assess the risk of and prevent water main breaks. *arXiv preprint arXiv:1805.03597*.

[54] Le, Q. V., Karpenko, A., Ngiam, J., and Ng, A. Y. (2011). Ica with reconstruction cost for efficient overcomplete feature learning. In *Advances in neural information processing systems*, pages 1017–1025.

[55] Le Gat, Y. and Eisenbeis, P. (2000). Using maintenance records to forecast failures in water networks. *Urban Water*, 2(3):173–181.

[56] Le Roux, N. and Bengio, Y. (2008). Representational power of restricted Boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649.

[57] Lee, D. D. and Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562.

[58] Li, B., Zhang, B., Li, Z., Wang, Y., Chen, F., and Vitanage, D. (2015). Prioritising water pipes for condition assessment with data analytics.

[59] Li, Z., Zhang, B., Wang, Y., Chen, F., Taib, R., Whiffin, V., and Wang, Y. (2014). Water pipe condition assessment: a hierarchical beta process approach for sparse incident data. *Machine learning*, 95(1):11–26.

[60] Lin, P., Zhang, B., Wang, Y., Li, Z., Li, B., Wang, Y., and Chen, F. (2015). Data driven water pipe failure prediction: A bayesian nonparametric approach. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 193–202. ACM.

[61] Luo, S., Azizi, L., and Sugiyama, M. (2021). Hierarchical probabilistic model for blind source separation via legendre transformation. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021)*, Virtual Event.

[62] Luo, S., Chu, V. W., Li, Z., Wang, Y., Zhou, J., Chen, F., and Wong, R. K. (2019). Multitask learning for sparse failure prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 3–14. Springer.

[63] Luo, S., Chu, V. W., Li, Z., Wang, Y., Zhou, J., Chen, F., and Wong, R. K. (2020a). Multi-task learning by hierarchical dirichlet mixture model for sparse failure prediction. *International Journal of Data Science and Analytics*, pages 1–15.

[64] Luo, S., Chu, V. W., Zhou, J., Chen, F., Wong, R. K., and Huang, W. (2017). A multivariate clustering approach for infrastructure failure predictions. In *Big Data (BigData Congress), 2017 IEEE International Congress on*, pages 274–281. IEEE.

[65] Luo, S. and Sugiyama, M. (2019). Bias-variance trade-off in hierarchical probabilistic models using higher-order feature interactions. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI-19)*, volume 33, pages 4488–4495, Hawaii, USA.

[66] Luo, S., Zhou, F., Azizi, L., and Sugiyama, M. (2020b). Additive poisson process: Learning intensity of higher-order interaction in stochastic processes. *arXiv preprint arXiv:2006.08982*.

[67] Mailhot, A., Pelletier, G., Noël, J.-F., and Villeneuve, J.-P. (2000). Modeling the evolution of the structural state of water pipe networks with brief recorded pipe break histories: Methodology and application. *Water Resources Research*, 36(10):3053–3062.

[68] Mavin, K. (1996). *Predicting the failure performance of individual water mains*. Urban Water Research Association of Australia.

[69] Min, M. R., Ning, X., Cheng, C., and Gerstein, M. (2014). Interpretable sparse high-order Boltzmann machines. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 614–622.

[70] Misiūnas, D. (2008). *Failure monitoring and asset condition assessment in water supply systems*. Vilniaus Gedimino technikos universitetas.

[71] Morris Jr, R. (1967). Principal causes and remedies of water main breaks. *Journal-American Water Works Association*, 59(7):782–798.

[72] Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

[73] Nakahara, H. and Amari, S. (2002). Information-geometric measure for neural spikes. *Neural Computation*, 14(10):2269–2316.

[74] Nakahara, H., Amari, S., and Richmond, B. J. (2006). A comparison of descriptive models of a single spike train by information-geometric measure. *Neural Computation*, 18(3):545–568.

[75] Neal, R. (1999). Regression and classification using gaussian process priors. *Bayesian statistics 6*, pages 475–501.

[76] Neal, R. M. (2001). Annealed importance sampling. *Statistics and Computing*, 11(2):125–139.

[77] Neal, R. M. (2005). Estimating ratios of normalizing constants using linked importance sampling. *arXiv:math/0511216*.

[78] Ogata, Y. (1981). On lewis' simulation method for point processes. *IEEE transactions on information theory*, 27(1):23–31.

[79] Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325.

[80] Pan, S. J. and Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

[81] Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

[82] Pelletier, G., Mailhot, A., and Villeneuve, J.-P. (2003). Modeling water pipe breaks—three case studies. *Journal of water resources planning and management*, 129(2):115–123.

[83] Pitman, J., Yor, M., et al. (1997). The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25(2):855–900.

[84] Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, pages 832–837.

[85] Rota, G.-C. (1964). On the foundations of combinatorial theory i. theory of möbius functions. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 2(4):340–368.

[86] Salakhutdinov, R. (2008). Learning and evaluating Boltzmann machines. *Technical Report UTML TR 2008-002, Department of Computer Science, University of Toronto*.

[87] Salakhutdinov, R. and Hinton, G. E. (2009). Deep Boltzmann machines. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 448–455.

[88] Salakhutdinov, R. and Hinton, G. E. (2012). An efficient learning procedure for deep Boltzmann machines. *Neural Computation*, 24(8):1967–2006.

[89] Sawada, H., Ono, N., Kameoka, H., Kitamura, D., and Saruwatari, H. (2019). A review of blind source separation methods: two converging routes to ILRMA originating from ICA and NMF. *APSIPA Transactions on Signal and Information Processing*, 8.

[90] Schäbe, H. (1993). Nonparametric estimation of intensities of nonhomogeneous poisson processes. *Statistical Papers*, 34(1):113–131.

[91] Scholz, M., Kaplan, F., Guy, C. L., Kopka, J., and Selbig, J. (2005). Non-linear pca: a missing data approach. *Bioinformatics*, 21(20):3887–3895.

[92] Schwaighofer, A., Tresp, V., and Yu, K. (2005). Learning gaussian process kernels via hierarchical bayes. In *Advances in Neural Information Processing Systems*, pages 1209–1216.

[93] Scott, D. W. (2015). *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

[94] Sejnowski, T. J. (1986). Higher-order Boltzmann machines. In *AIP Conference Proceedings*, volume 151, pages 398–403. AIP.

[95] Sethuraman, J. (1994). A constructive definition of dirichlet priors. *Statistica sinica*, pages 639–650.

[96] Shamir, U., Howard, C., et al. (1979). An analytical approach to scheduling pipe replacement. *Journal-American Water Works Association*, 71(5):248–258.

[97] Sugiyama, M., Nakahara, H., and Tsuda, K. (2016a). Information decomposition on structured space. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 575–579, Barcelona, Spain.

[98] Sugiyama, M., Nakahara, H., and Tsuda, K. (2016b). Information decomposition on structured space. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 575–579. IEEE.

[99] Sugiyama, M., Nakahara, H., and Tsuda, K. (2017a). Tensor balancing on statistical manifold. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3270–3279. JMLR. org.

[100] Sugiyama, M., Nakahara, H., and Tsuda, K. (2017b). Tensor balancing on statistical manifold. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 3270–3279, Sydney, Australia.

[101] Sugiyama, M., Nakahara, H., and Tsuda, K. (2018). Legendre decomposition for tensors. In *Advances in Neural Information Processing Systems 31*, pages 8825–8835, Montréal, Canada.

[102] Taddy, M. A. (2010). Autoregressive mixture models for dynamic spatial poisson processes: Application to tracking intensity of violent crime. *Journal of the American Statistical Association*, 105(492):1403–1417.

[103] Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

[104] Thibaux, R. and Jordan, M. I. (2007). Hierarchical beta processes and the indian buffet process. In *AISTATS*, volume 2, pages 564–571.

[105] Thompson, H. (1955). Spatial point processes, with applications to ecology. *Biometrika*, 42(1/2):102–115.

[106] Tieleman, T. (2008). Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 1064–1071.

[107] Vigário, R., Jousmäki, V., Hämäläinen, M., Hari, R., and Oja, E. (1998). Independent component analysis for identification of artifacts in magnetoencephalographic recordings. In *Advances in neural information processing systems*, pages 229–235.

[108] Xu, H., Caramanis, C., and Sanghavi, S. (2010). Robust pca via outlier pursuit. In *Advances in Neural Information Processing Systems*, pages 2496–2504.

[109] Xue, Y., Liao, X., Carin, L., and Krishnapuram, B. (2007). Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8(Jan):35–63.

[110] Zhang, Y. and Yang, Q. (2017). A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*.

[111] Zhou, F., Li, Z., Fan, X., Wang, Y., Sowmya, A., and Chen, F. (2018). A refined MISD algorithm based on Gaussian process regression. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 584–596. Springer.

[112] Zhou, F., Luo, S., Li, Z., Fan, X., Wang, Y., Sowmya, A., and Chen, F. (2021). Efficient em-variational inference for nonparametric hawkes process. *Statistics and Computing*, 31(4):1–11.

[113] Zocher, M. (2006). *Multivariate counting processes*. Techn. Univ., Inst. für Mathematische Stochastik.

[114] Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286.

APPENDIX A

Appendix

A.1 Information Geometric Formulation of the Exponential Family

In this section, we will present the derivation for the information geometric formulation of the exponential family. We first introduce two coordinate systems $\xi = (\xi_1, \ldots, \xi_n)$ and $\zeta = (\zeta_1, \ldots, \zeta_n)$ so that there exists a one-to-one correspondence between them and have the relations,

$$\begin{aligned} \xi &= \mathbf{f}(\zeta_1, \ldots, \zeta_n) \\ \zeta &= \mathbf{f}^{-1}(\xi_1, \ldots, \xi_n) \end{aligned} \tag{A.1}$$

where $\mathbf{f}$ and $\mathbf{f}^{-1}$ are coordinate transformations that is invertible and differentiable. The transformation between ξ and ξ^* is one-to-one and differentiable. This transformation is known as the Legendre transformation and has a dualistic structure for the two coupled coordinate system ξ and ξ^*.

Theorem 6 (Legendre Transformation [9])**.** *The gradient*

$$\xi^* = \nabla \psi(\xi), \tag{A.2}$$

is differentiable and a one-to-one transformation. This means that ξ^* is used as another coordinate system on the manifold M and it is connected to ξ by the Legendre Transformation.

Proof. The Legendre transformation has a dualistic structure concerning the two coupled coordinate system ξ and ξ^*. To show this, a new function ξ^* is defined as,

$$\psi^*\left(\xi^*\right) = \xi\xi^* - \psi\left(\xi\right), \tag{A.3}$$

where,

$$\xi\xi^* = \sum_i \xi_i \xi_i^*.$$

By differentiating Equation A.3 with respect to ξ^*, we have,

$$\nabla\psi^*\left(\xi^*\right) = \xi + \frac{\partial\xi}{\partial\xi^*}\xi^* - \nabla\psi(\xi)$$

The last two terms cancel out because of Equation A.2, then we have,

$$\nabla\psi^*\left(\xi^*\right) = \xi.$$

This is a dualistic structure because we have,

$$\xi^* = \nabla\psi\left(\xi\right), \qquad \xi = \nabla\psi\left(\xi^*\right).$$

ψ^* is called the Legendre dual of ψ. because it satisfies,

$$\psi^*\left(\xi^*\right) = \max_{\xi'}\left\{\,\xi'\xi^* - \psi\left(\xi'\right)\,\right\},$$

which is the definition of ψ^*. □

Let x be a discrete random variable taking values on $X = \{\,0,1,\ldots,n\,\}$. We then define a probability distribution $p(x)$ in a probability vector $\mathbf{p} = (p_0, p_1, \ldots, p_n)$ with the restriction $\sum_{i=0}^{n} p_i = 1$ and $p_i > 0$. The set of all probability distributions $\mathbf{p}$ forms an n-dimensional manifold. Its coordinate system is given by,

$$\xi = (p_1, \ldots, p_n),$$

where, p_0 is uniquely determined by,

$$p_0 = 1 - \sum \xi_i.$$

The manifold is a n-dimensional probability simplex S_n. Then a probability distribution of x is

$$p(x,\xi) = \sum_{i=1}^{n} \xi_i \delta_i(x) + p_0(\xi)\,\delta_0(x)$$

We define another coordinate system $\theta = \{\theta_1, \ldots, \theta_n\}$ by,

$$\theta_i = \log \frac{p_i}{p_0} \qquad \text{for all } i = 1, \ldots, n,$$

which will be a more convenient notation to define our equations in the future. We then define another coordinate system as,

$$\eta = \nabla \psi(\theta).$$

Using this notation the probability distribution for natural exponential family is written as [9],

$$p(x;\theta) = \exp\left[\sum_i \theta_i x_i - \psi(\theta)\right],$$

where, $\exp[\psi(\theta)]$ is the partition which which is uniquely determined by,

$$\psi(\theta) = \log \int \exp\left[\sum \theta_i x_i\right] d\mathbf{x}.$$

The coordinate systems θ becomes the natural parameter and η becomes the expectation parameter. In the following sections, we introduce incidence algebra to formulate a log-linear model on a poset which is a class of models in the exponential family.

A.2 Information Geometric Proofs for the Log-Linear Model

A.2.1 Proof for Dually Flat Structure

Proof for Legendre Dual

Theorem 7 (Legendre Dual [99])**.** *The gradient*

$$\varphi(\eta) = \sum_{\omega \in \Omega} p(\omega) \log p(\omega). \tag{A.4}$$

is differentiable and a one-to-one transformation. This means φ is used as another coordinate system on the manifold M and it is connected to ψ by a Legendre transformation.

Proof.

$$\theta' \eta = \sum_{\omega \in \Omega^+} \left[\sum_{\substack{\omega' \in \Omega \\ \perp < \omega' \leq \omega}} \mu(\omega', \omega) \log p'(\omega') \sum_{\omega' \geq \omega} p(\omega') \right] = \sum_{\omega \in \Omega^+} p(\omega) \left(\log p'(\omega) - \log p'(\perp) \right)$$

Thus it holds that

$$\theta' \eta - \psi(\theta') = \sum_{\omega \in \Omega} p(\omega) \log p'(\omega).$$

Hence it is maximized with $p(\omega) = p'(\omega)$. □

Proof for Dual Coordinate System

Theorem 8 (Dual Coordinate System [99])**.** *A dual coordinate system can be defined by*

$$\nabla \psi(\theta) = \eta, \qquad \nabla \varphi(\eta) = \theta$$

where the two coordinate systems are connected by the Legendre transformation.

Proof.

$$\frac{\partial \psi(\theta)}{\partial \eta(\omega)} = \frac{\sum_{\omega'' \in \Omega} \exp\left[\sum_{\substack{\omega' \in \Omega \\ \perp < \omega' \leq \omega''}} \theta(\omega')\right]}{\sum_{\omega'' \in \Omega} \exp\left[\sum_{\substack{\omega' \in \Omega \\ \perp < \omega' \leq \omega''}} \theta(\omega')\right]} = \sum_{\substack{\omega' \in \Omega \\ \omega' \geq \omega}} p(\omega') = \eta(\omega),$$

$$\frac{\partial \varphi(\eta)}{\partial \eta(\omega)} = \frac{\partial}{\partial \eta(\omega)} \left[\theta \eta - \psi(\theta)\right] = \theta(\omega).$$

orthogonality of θ and η as,

$$\begin{aligned}
&\mathbb{E}\left[\frac{\partial}{\partial \theta(\omega)} \log p(\omega'') \frac{\partial}{\partial \eta(\omega')} \log p(\omega'')\right] \\
=&\sum_{\omega'' \in \Omega} \left[p(\omega'') \frac{\partial}{\partial \theta(\omega)} \sum_{\omega''' \in \Omega} \zeta(\omega''', \omega'') \theta(\omega''') \frac{\partial}{\partial \eta(\omega')} \log \left[\sum_{\omega''' \in \Omega} \mu(\omega'', \omega''') \eta(\omega''') \right] \right], \\
=&\sum_{\omega'' \in \Omega} \left[p(\omega'') \left(\zeta(\omega, \omega'')\right) - \eta(\omega) \frac{\mu(\omega'', \omega')}{p(\omega'')} \right] \\
=&\sum_{\omega'' \in \Omega} \zeta(\omega, \omega'') \mu(\omega'', \omega') = \delta_{\omega\omega'}
\end{aligned}$$

□

A.2.2 Proof for Riemannian Structure

Theorem 9 (Riemannian Metric [99])**.** *The manifolid* $(\Omega, G(\xi))$ *is a Riemannian manifold with the Riemannian metric* $G(\xi)$ *such that for all* $\omega, \omega', \in \Omega^+$*,*

$$G_{\omega\omega'}(\xi) = \begin{cases} \sum_{\omega'' \in \Omega} \zeta(\omega, \omega'') \zeta(\omega', \omega'') p(\omega'') - \eta(\omega) \eta(\omega') & \text{if } \xi = \theta, \\ \sum_{\omega'' \in \Omega} \mu(\omega'', \omega) \mu(\omega'', \omega') p(\omega'')^{-2} & \text{if } \xi = \eta. \end{cases}$$

Proof. Since the Riemannian metric is defined as

$$G(\theta) = \nabla\nabla\psi(\theta), \qquad G(\eta) = \nabla\nabla\varphi(\eta),$$

when $\xi = \theta$ we have

$$\begin{aligned} G_{\omega\omega'}(\theta) &= \frac{\partial^2}{\partial\theta(\omega)\,\partial\theta(\omega')}\psi(\theta) = \frac{\partial}{\partial\theta(\omega)}\eta(\omega') \\ &= \frac{\partial}{\partial\theta(\omega)}\sum_{\omega''\in\Omega}\zeta(\omega',\omega'')\exp\left[\sum_{\substack{\omega'''\in\Omega \\ \perp<\omega'''\leq\omega''}}\theta(\omega''') - \psi(\theta)\right] \\ &= \sum_{\omega''\in\Omega}\zeta(\omega,\omega'')\,\zeta(\omega',\omega'')\,p(\omega'') - \eta(\omega)\,\eta(\omega'). \end{aligned}$$

When $\xi = \eta$, it follows that

$$\begin{aligned} G_{\omega\omega'}(\eta) &= \frac{\partial^2}{\partial\eta(\omega)\,\partial\eta(\omega')}\varphi(\eta) = \frac{\partial}{\partial\eta(\omega)}\theta(\omega') \\ &= \frac{\partial}{\partial\eta(\omega)}\sum_{\substack{\omega''\in\Omega \\ \omega''\leq\omega'}}\mu(\omega'',\omega')\log p(\omega'') \\ &= \frac{\partial}{\partial\eta(\omega)}\sum_{\substack{\omega''\in\Omega \\ \omega''\leq\omega'}}\mu(\omega'',\omega')\log\left[\sum_{\substack{\omega'''\in\Omega \\ \omega'''\geq\omega''}}\mu(\omega'',\omega''')\,\eta(\omega''')\right] \\ &= \sum_{\omega''\in\Omega}\frac{\mu(\omega'',\omega)\,\mu(\omega'',\omega')}{\sum_{\substack{\omega'''\in\Omega \\ \omega'''\geq\omega''}}\mu(\omega'',\omega''')\,\eta(\omega''')} \\ &= \sum_{\substack{\omega''\in\Omega \\ \omega''\in\Omega}}\mu(\omega'',\omega)\,\mu(\omega'',\omega')\,p(\omega'')^{-1} \end{aligned} \tag{A.5}$$

Since $G(\xi)$ coincides with Fisher information matrix

$$\begin{aligned} \mathbb{E}\left[\frac{\partial}{\partial\theta(\omega)}\log p(\omega'')\,\frac{\partial}{\partial\theta(\omega')}\log p(\omega'')\right] &= G_{\omega\omega'}(\theta), \\ \mathbb{E}\left[\frac{\partial}{\partial\eta(\omega)}\log p(\omega'')\,\frac{\partial}{\partial\eta(\omega')}\log p(\omega'')\right] &= G_{\omega\omega'}(\eta). \end{aligned}$$

Then the Riemannian (Levi-Chivita) connection $\Gamma(\xi)$ with respect to ξ is defined as

$$\Gamma_{\omega\omega'\omega''}(\xi) = \frac{1}{2}\left[\frac{\partial G_{\omega'\omega''}(\xi)}{\partial\xi(\omega)} + \frac{\partial G_{\omega\omega''}(\xi)}{\partial\xi(\omega')} - \frac{G_{\omega\omega'}(\xi)}{\partial\xi(\omega'')}\right]$$

for all $\omega, \omega', \omega'' \in \Omega^+$, can be analytically obtained. □

Theorem 10 (Riemannian Connection [99])**.** *The Riemannian connection* $\Gamma(\xi)$ *on the manifold* $(\Omega, g(\xi))$ *is given in the following for all* $\omega, \omega', \omega'' \in \Omega^+$*,*

$$\Gamma_{\omega\omega'\omega''}(\xi) =$$
$$\begin{cases} \frac{1}{2}\sum\limits_{\omega'''\in\Omega}\left[\zeta\left(\omega,\omega'''\right)-\eta\left(\omega\right)\right)\left(\zeta\left(\omega'',\omega'''\right)-\eta\left(\omega'\right)\right)\left(\zeta\left(\omega'',\omega'''\right)-\eta\left(\omega''\right)\right]p\left(\omega'''\right), & \text{if } \xi=\theta, \\ -\frac{1}{2}\sum\limits_{\omega'''\in\Omega}\mu\left(\omega''',\omega\right)\mu\left(\omega''',\omega'\right)\mu\left(\omega''',\omega''\right)p\left(\omega'''\right)^{-2}, & \text{if } \xi=\eta. \end{cases}$$

Proof. for all $\omega, \omega', \omega'' \in \Omega$

$$\frac{\partial G_{\omega'\omega''}(\theta)}{\partial\theta\left(\omega\right)} = \frac{\partial}{\partial\theta\left(\omega\right)}\sum_{\omega'''\in\Omega}\zeta\left(\omega',\omega'''\right)\zeta\left(\omega'',\omega'''\right)p\left(\omega'''\right) - \frac{\partial}{\partial\theta\left(\omega\right)}\eta\left(\omega'\right)\eta\left(\omega''\right),$$

where,

$$\begin{aligned} &\frac{\partial}{\partial\theta\left(\omega\right)}\sum_{\omega'''\in\Omega}\zeta\left(\omega',\omega'''\right)\zeta\left(\omega'',\omega'''\right)p\left(\omega'''\right) \\ =&\frac{\partial}{\partial\theta\left(\omega\right)}\sum_{\omega'''\in\Omega}\zeta\left(\omega,\omega'''\right)\zeta\left(\omega',\omega'''\right)\zeta\left(\omega'',\omega'''\right)\exp\left[\sum_{\substack{\omega''''\in\Omega \\ \perp<\omega''''\leq\omega'''}}\theta\left(\omega''''\right)-\psi\left(\theta\right)\right] \\ =&\sum_{\omega'''\in\Omega}\zeta\left(\omega,\omega'''\right)\zeta\left(\omega',\omega'''\right)\zeta\left(\omega'',\omega'''\right)p\left(\omega'''\right)-\eta\left(\omega\right)\sum_{\omega'''\in\Omega}\zeta\left(\omega',\omega'''\right)\zeta\left(\omega'',\omega'''\right)p\left(\omega'''\right) \end{aligned}$$

and

$$\begin{aligned} &\frac{\partial}{\partial\theta\left(\omega\right)}\eta\left(\omega'\right)\eta\left(\omega''\right) \\ =&\frac{\partial\eta\left(\omega'\right)}{\partial\theta\left(\omega\right)}\eta\left(\omega''\right)+\frac{\partial\eta\left(\omega''\right)}{\partial\theta\left(\omega\right)}\eta\left(\omega'\right) \\ =&\eta\left(\omega''\right)\sum_{\omega'''\in\Omega}\zeta\left(\omega,\omega'''\right)\zeta\left(\omega',\omega'''\right)p\left(\omega'''\right) \\ &+\eta\left(\omega'\right)\sum_{\omega'''\in\Omega}\zeta\left(\omega,\omega'''\right)\zeta\left(\omega'',\omega'''\right)p\left(\omega'''\right)-2\eta\left(\omega\right)\eta\left(\omega'\right) \end{aligned}$$

It follows that,

$$\frac{\partial G_{\omega'\omega''}(\theta)}{\partial\theta\left(\omega\right)} = \sum_{\omega'''\in\Omega}\left(\zeta\left(\omega,\omega'''\right)-\eta\left(\omega\right)\right)\left(\zeta\left(\omega',\omega'''\right)-\eta\left(\omega'\right)\right)\left(\zeta\left(\omega'',\omega'''\right)-\eta\left(\omega''\right)\right)p\left(\omega'''\right).$$

On the other hand,

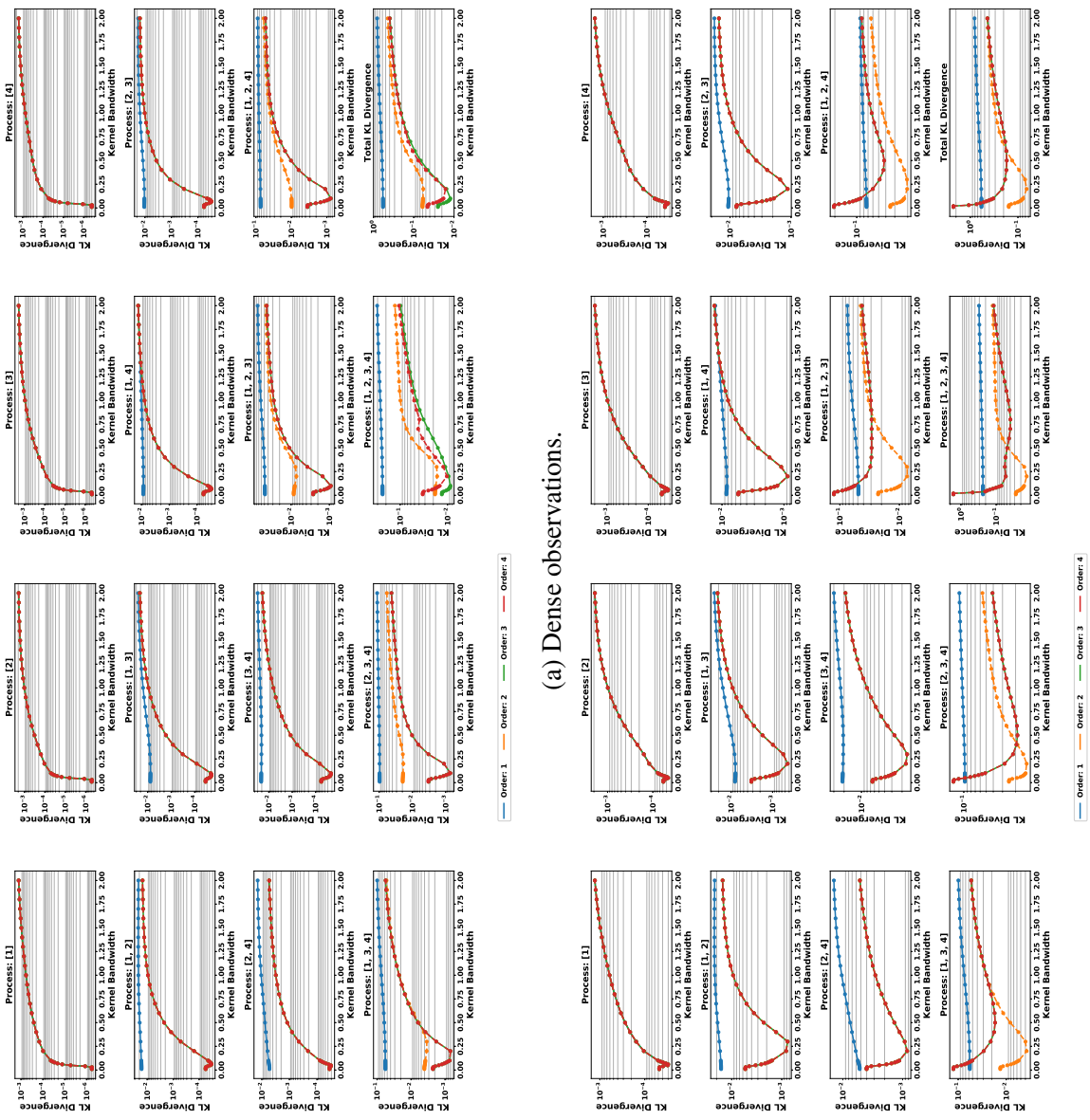
$$\begin{aligned}
\frac{\partial G_{\omega'\omega''}(\theta)}{\partial \eta(\omega)} &= \frac{\partial}{\partial \eta(\omega)} \sum_{\omega''' \in \Omega} \mu\left(\omega''', \omega'\right) \mu\left(\omega''', \omega''\right) p\left(\omega'''\right)^{-1} \\
&= \frac{\partial}{\partial \eta(\omega)} \sum_{\omega''' \in \Omega} \mu\left(\omega''', \omega'\right) \mu\left(\omega''', \omega''\right) \left[\sum_{\substack{\omega'''' \in \Omega \\ \omega'''' \geq \omega'''}} \mu\left(\omega''', \omega''''\right) \eta\left(\omega'''\right) \right] \\
&= -\sum_{\omega''' \in \Omega} \mu\left(\omega''', \omega\right) \mu\left(\omega''', \omega'\right) \mu\left(\omega''', \omega''\right) \left[\sum_{\substack{\omega'''' \in \Omega \\ \omega'''' \geq \omega'''}} \mu\left(\omega''', \omega''''\right) \eta\left(\omega'''\right) \right]^{-2} \\
&= \frac{\partial}{\partial \eta(\omega)} \sum_{\omega''' \in \Omega} \mu\left(\omega''', \omega'\right) \mu\left(\omega''', \omega'\right) \mu\left(\omega''', \omega''\right) p\left(\omega'''\right)^{-2}
\end{aligned}$$

Therefore, from the definition of $\Gamma(\xi)$, it follows that,

$$\begin{aligned}
\Gamma_{\omega\omega'\omega''}(\theta) &= \frac{1}{2}\left[\frac{\partial G_{\omega'\omega''}(\theta)}{\partial \theta(\omega)} + \frac{\partial G_{\omega\omega''}(\theta)}{\partial \theta(\omega')} - \frac{\partial G_{\omega\omega'}(\theta)}{\partial \theta(\omega'')}\right] \\
&= \frac{1}{2} \sum_{\omega''' \in \Omega} \left(\zeta\left(\omega''', \omega\right) - \eta(\omega)\right) \left(\zeta\left(\omega''', \omega'\right) - \eta\left(\omega'\right)\right) \left(\zeta\left(\omega''', \omega''\right) - \eta\left(\omega''\right)\right) p\left(\omega'''\right), \\
\Gamma_{\omega\omega'\omega''}(\eta) &= \frac{1}{2}\left[\frac{\partial G_{\omega'\omega''}(\eta)}{\partial \eta(\omega)} + \frac{\partial G_{\omega\omega''}(\eta)}{\partial \eta(\omega')} - \frac{\partial G_{\omega\omega'}(\eta)}{\partial \eta(\omega'')}\right], \\
&= -\frac{1}{2} \sum_{\omega''' \in \Omega} \mu\left(\omega''', \omega\right) \mu\left(\omega''', \omega'\right) \mu\left(\omega''', \omega'\right) p\left(\omega'''\right)^{-2}.
\end{aligned}$$

□

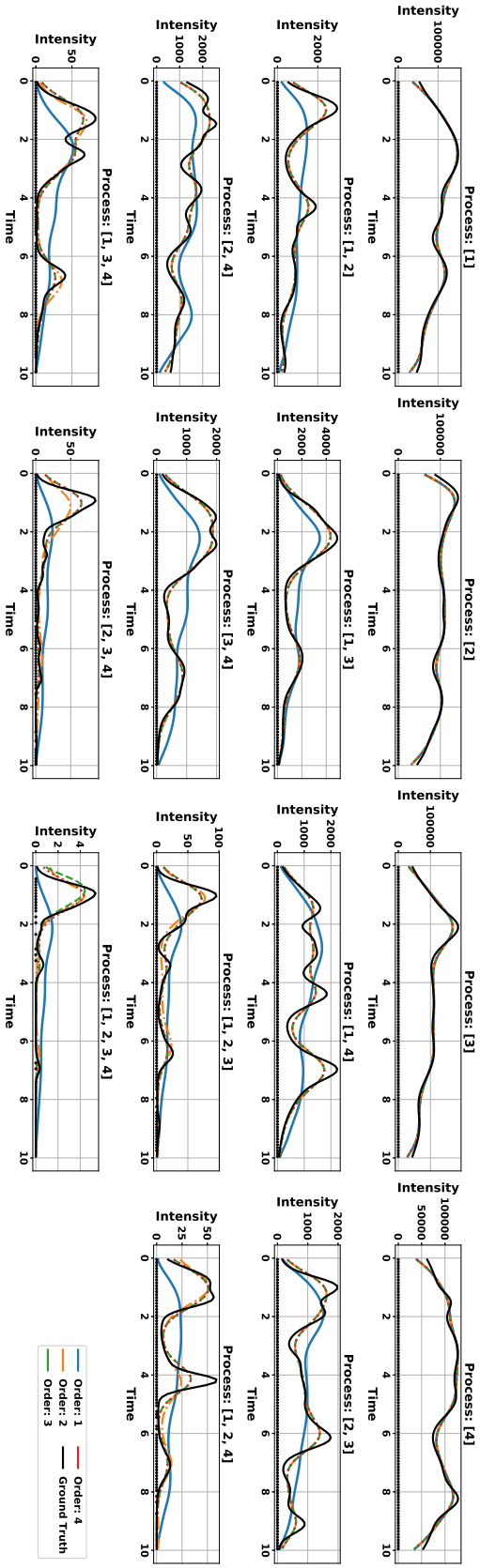
A.3 Full Results for Additive Poisson Process Experiments



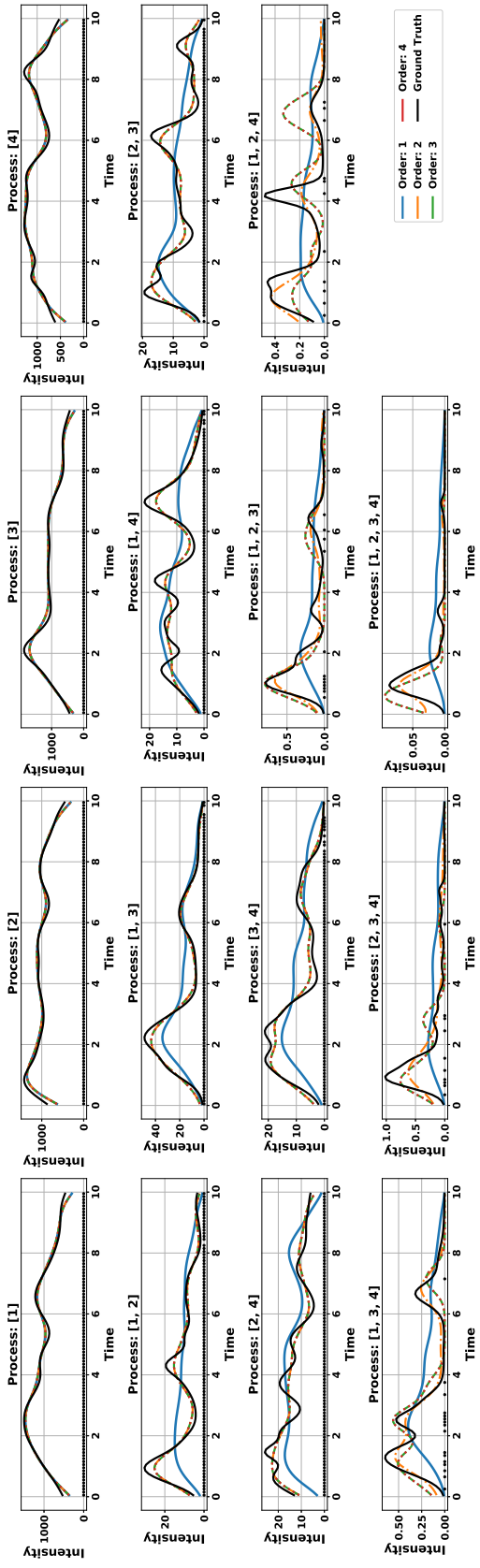
(a) Dense observations.

(b) Sparse observations.

Fig. A.1 KL Divergence for four-order Poisson process.



(a) Dense observations.



(a) Sparse observations.

Fig. A.3 Intensity function of higher dimensional processes. Dots represent observations.