

# Nonparametric Latent Tree Graphical Models: Inference, Estimation, and Structure Learning

Le Song\*, Han Liu†, Ankur Parikh‡, and Eric Xing§

## Abstract

Tree structured graphical models are powerful at expressing long range or hierarchical dependency among many variables, and have been widely applied in different areas of computer science and statistics. However, existing methods for parameter estimation, inference, and structure learning mainly rely on the Gaussian or discrete assumptions, which are restrictive under many applications. In this paper, we propose new nonparametric methods based on reproducing kernel Hilbert space embeddings of distributions that can recover the latent tree structures, estimate the parameters, and perform inference for high dimensional continuous and non-Gaussian variables. The usefulness of the proposed methods are illustrated by thorough numerical results.

**Keyword:** Latent tree graphical models, reproducing kernel Hilbert space embedding of distributions, Latent variable models, parameter estimation, structure learning.

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Latent Tree Graphical Models</b>	<b>4</b>
<b>3</b>	<b>Overview of Our Technique</b>	<b>6</b>
<b>4</b>	<b>Kernel Density Estimator as Hilbert Space Embeddings</b>	<b>7</b>
4.1	Kernel density estimator . . . . .	7
4.2	A Hilbert space embedding view of KDE . . . . .	7
4.3	Tensor expression for KDE . . . . .	8
<b>5</b>	<b>Latent Tree Representation via Hilbert Space Embedding</b>	<b>8</b>

---

\*College of Computing, Georgia Institute of Technology, Atlanta, Georgia, 30332, USA; email: [lsong@cc.gatech.edu](mailto:lsong@cc.gatech.edu)

†Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ 08544, USA; e-mail: [hanliu@princeton.edu](mailto:hanliu@princeton.edu)

‡School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 15213, USA; email: [aparikh@cs.cmu.edu](mailto:aparikh@cs.cmu.edu)

§School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 15213, USA; email: [epxing@cs.cmu.edu](mailto:epxing@cs.cmu.edu)

<b>6</b>	<b>Inference on Latent Tree Graphical Models</b>	<b>9</b>
6.1	Message passing algorithm in density space . . . . .	9
6.2	Message passing algorithm using kernel embeddings . . . . .	11
<b>7</b>	<b>Observable Representation and Parameter Estimation</b>	<b>12</b>
7.1	Computation . . . . .	14
7.2	Example . . . . .	16
<b>8</b>	<b>Structure Learning</b>	<b>17</b>
8.1	Tree Metric and Pseudo-determinant for Non-Gaussian Variables . . . . .	17
8.2	Empirical Tree Metric Estimator and Its Concentration Property . . . . .	18
8.3	Structure Learning Algorithm . . . . .	20
<b>9</b>	<b>Experiments</b>	<b>22</b>
9.1	Synthetic data: density estimation . . . . .	23
9.2	Synthetic data: structure recovery. . . . .	23
9.3	Synthetic data: model selection. . . . .	24
9.4	Crime Dataset. . . . .	24
<b>10</b>	<b>Conclusions</b>	<b>26</b>
<b>A</b>	<b>Kernel Embedding of Distributions</b>	<b>26</b>
A.1	Population Definition . . . . .	27
A.2	Finite Sample Kernel Estimator . . . . .	28
<b>B</b>	<b>Kernel Embeddings of Conditional Distributions</b>	<b>29</b>
B.1	Population Definition . . . . .	29
B.2	Finite Sample Kernel Estimator . . . . .	30
<b>C</b>	<b>Hilbert Space Embeddings as Infinite Dimensional Higher Order Tensors</b>	<b>30</b>

# 1 Introduction

Modern data acquisition routinely produces massive amounts of high dimensional data with complex statistical dependency structures. Latent variable graphical models provide a succinct representation of such complex dependency structures by relating the observed variables to a set of latent ones. By defining a joint distribution over observed and latent variables, the marginal distribution of the observed variables can be obtained by integrating out the latent ones. This allows complex distributions over observed variables (e.g., clique models) to be expressed in terms of more tractable joint models (e.g., tree models) over the augmented variable space. Probabilistic graphical models with latent variables have been deployed successfully to a diverse range of problems such as in document analysis (Blei et al., 2002), social network modeling (Hoff et al., 2002), speech recognition (Rabiner and Juang, 1986) and bioinformatics (Clark, 1990).

In this paper, we focus on latent variable models where the latent structures are trees (we call it a “latent tree” for short). In these tree-structured graphical models, the leaves are the set of observed variables (e.g., taxa, pixels, words) while the internal nodes are hidden and intuitively

“represent” the common properties of their descendants (e.g., distinct ancestral species, objects in an image, latent semantics). This class of models strike a nice balance between their representation power (e.g., it can still model cliques) and the complexity of learning and inference processes on these structures (e.g., the message passing inference algorithm is exact on trees). In particular, we propose a unified nonparametric framework for addressing three key questions in tree-structured latent variable models:

- Estimating latent tree structures
- Estimate model parameters
- Conduct inference on the obtained tree graphical models.

In the current literature, the problem of estimating the structure of latent trees has largely been tackled by heuristic algorithms since exhaustively searching over the whole latent tree space is intractable. For instance, Zhang (2004) proposed a search heuristic for hierarchical latent class models by defining a series of local search operations and using the expectation-maximization (EM) algorithm to compute the likelihood of candidate structures. Harmeling and Williams (2010) proposed a greedy algorithm to learn binary trees by joining two nodes with a high mutual information and iteratively performing the EM algorithm to compute the mutual information among newly added hidden nodes. Alternatively, Heller and Ghahramani (2005) proposed the Bayesian hierarchical clustering method, which is an agglomerative clustering technique that merges clusters based on statistical hypothesis testing. Many other local search heuristics based on maximum parsimony and maximum likelihood methods can also be found from the phylogenetic community (Semple and Steel, 2003). Poon et al. (2010) propose a related model to latent trees called “pouch latent tree models” which allows for multiple observed variables to be placed in the same leaf (pouch) of the tree. Similar to the above methods, learning is done heuristically via a greedy BIC score search. However, none of these methods extends easily to the nonparametric setting since they require the data to be either discrete or at least the distributions have a parametric form such that likelihood based testing and estimation can be easily computed.

Besides these heuristic approaches, many methods with provable guarantees have also been proposed in the phylogenetic community (Erdős et al., 1999a,b; Mossel and Roch, 2006; Mossel, 2007; Roch, 2010; Mossel et al., 2011a,b; Gronau et al., 2008; Daskalakis et al., 2006, 2011; Anandkumar et al., 2011; Anandkumar and Valluvan, 2013). For instance, neighbor joining algorithm (Saitou et al., 1987) and recursive grouping algorithm (Choi et al., 2010) take a pairwise distance matrix between all observed variables as input and output an estimated tree graph by iteratively adding hidden nodes. While these methods are iterative, they have strong theoretical guarantees on structure recovery when the true distance matrix forms an additive tree metric, i.e., the distance between nodes  $s$  and  $t$  is equal to the sum of the distances of the edges along its path in a tree  $T$ . In this case these methods are guaranteed to recover the correct structure (binary for neighbor joining, but arbitrary branching for the recursive grouping method). However, a major challenge in applying these distance based methods to the continuous, nonparametric case is that it is not clear how to define a valid additive tree metric, and after recovering the latent tree structure, how to perform efficient parameter learning and probabilistic inference (e.g., calculating the marginal or conditional distributions).

Once the latent tree structure is correctly recovered, estimating the model parameters has predominantly relied on likelihood maximization and local search heuristics such as the EM algorithm (Dempster et al., 1977). Besides the problem of local minima, non-Gaussian statistical

features such as multimodality and skewness may pose additional challenge for EM. For instance, parametric models such as mixture of Gaussians may lead to an exponential blowup in terms of their representation during the inference stage of the EM algorithm. Further approximations are needed to make the computation tractable. These approximations make it even harder to analyze the theoretical properties of the obtained estimator. In addition, the EM algorithm in general requires many iterations to reach a pre-specified training precision.

In this paper, we all propose a method for learning the parameters of tree-structured latent variable models with continuous and non-Gaussian observation based on the concept of Hilbert space embedding of distributions (Smola et al., 2007a). The problems we try to address include: (1) How to estimate the structures of latent trees with strong theoretical guarantees; (2) How to perform local-minimum-free learning and inference based on the estimated tree structures, all in nonparametric setting. The main idea of our method is to exploit the spectral properties of the joint embedding (or covariance operators) in all the structure recovery, parameter learning, and probabilistic inference stages. For structure learning, we define a distance measure between pairwise variables based on singular value decomposition of covariance operators. This allows us to generalize existing distance based latent tree learning procedures such as neighbor joining (Saitou et al., 1987) and recursive grouping (Choi et al., 2010) to fully nonparametric settings. In this paper, we focus on the case where the set of observed variables (leaves) are continuous-valued and their joint distributions can not be easily characterized by a parametric family (e.g., Gaussian). Thus our approach fundamentally differs from almost all others, which mainly consider the case of discrete or Gaussian variables. Similar to their parametric counterparts, the obtained structure learning algorithms have strong theoretical guarantees. After tree structures have been recovered, we further exploit the principal singular vectors of the covariance operator to estimate the parameters of the latent variables. One advantage of our spectral algorithm is that it is local-minimum-free and hence amenable for further theoretical analysis. In particular, we will demonstrate the advantage of our method over existing approaches in both simulation and real data experiments.

The rest of this paper is organized as follows. First, we will explain our terminology of latent tree graphical models in Section 2. In Section 3, we will provide a road map of our nonparametric framework which uses kernel embedding of distributions as key technique. In Section 4, we will present connection between kernel density estimation for tree-structured latent variable models and kernel embedding of the distributions of such models. Then in Section 5, we will first explain our nonparametric methods for inference and parameter learning. And in Section 6, a nonparametric method for structure learning, both based on the idea of Hilbert space embedding of distributions. Last, we will present our experimental results in synthetic and real datasets in Section 7. An introduction of the concept of Hilbert space embedding of distributions and conditional distributions is provided in Appendix A and B respectively. Furthermore, the connection between Hilbert space embedding of distributions and higher order tensors is presented in Section C instead.

## 2 Latent Tree Graphical Models

In this paper, we focus on latent variable models where the observed variables are continuous non-Gaussian. We also assume the conditional independence structures of the joint distribution of the observed and latent variables are fully specified by trees. We will use uppercase letters to denote random variables (e.g.,  $X_i$ ) and lowercase letters to denote the corresponding realizations (e.g.,  $x_i$ ). For notational simplicity, we assume that the domain  $\Omega$  of all variables are the same. Generalization

to the cases where the variables have different domains is straightforward. A latent tree model defines a joint distribution over a set of  $O$  observed variables, denoted by  $\mathcal{O} = \{X_1, \dots, X_O\}$ , and a set of  $H$  hidden variables, denoted by  $\mathcal{H} = \{X_{O+1}, \dots, X_{O+H}\}$ . The complete set of variables is denoted by  $\mathcal{X} = \mathcal{O} \cup \mathcal{H}$ . For simplicity, we assume that

- (A1) All observed variables have the same domain  $\mathcal{X}_{\mathcal{O}}$ , and all hidden variables are discrete and take  $k$  values from  $\mathcal{X}_{\mathcal{H}}$ . Furthermore, all observed variables are leaf nodes of the tree, and each hidden node in the tree has exactly 3 neighbors.

In this case, after we re-root the tree and redirect all the edges, for a node  $s$  (corresponding to the variable  $X_s$ ), we use  $\alpha_s$  to denote its sibling,  $\pi_s$  to denote its parent,  $\iota_s$  to denote its left child and  $\rho_s$  to denote its right child; the root node will have 3 children, we use  $\omega_s$  to denote the extra child of the root node. All the observed variables are leaves in the tree, and we will use  $\iota_s^*$ ,  $\rho_s^*$ ,  $\pi_s^*$  to denote an arbitrary observed variable which is found by tracing in the direction from node  $s$  to its left child  $\iota_s$ , right child  $\rho_s$ , and its parent  $\pi_s$  respectively. See Figure 1 for a summary of the notation.

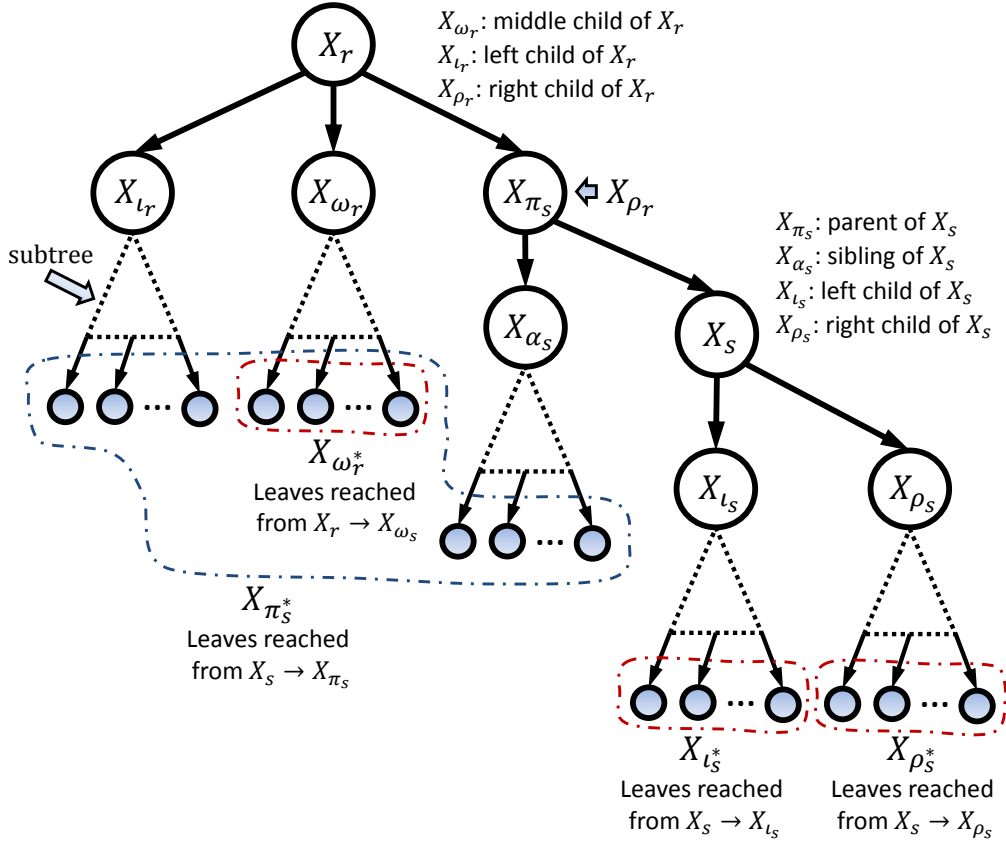


Figure 1: Illustration of a tree-structured latent variable model (or latent tree model). Open circle denotes hidden variables, and filled circles for observed ones. For a node  $s$ , we use  $\alpha_s$  to denote its sibling,  $\pi_s$  to denote its parent,  $\iota_s$  to denote its left child and  $\rho_s$  to denote its right child; the root node will have 3 children, we use  $\omega_s$  to denote the extra child.

The joint distribution of  $\mathcal{X}$  in a latent tree model is fully characterized by a set of conditional distributions. More specifically, we can select an arbitrary latent node in the tree as the root, and

reorient all edges away from the root. Then the set of conditional distributions between nodes and their parents  $p(X_i|X_{\pi_i})$  are sufficient to characterize the joint distribution (for the root node  $X_r$ , we set  $p(X_r|X_{\pi_r}) = p(X_r)$ ) and use  $p(\cdot)$  to refer to density in continuous case with

$$p(\mathcal{X}) = \prod_{i=1}^{O+H} p(X_i|X_{\pi_i}). \quad (1)$$

Compared to tree models which are defined solely on observed variables (e.g., models obtained from the Chow-Liu algorithm (Chow and Liu, 1968)), latent tree models encompass a much larger classes of models, allowing more flexibility in modeling observed variables. This is evident if we compute the marginal distribution of the observed variables by summing out the latent ones,

$$p(\mathcal{O}) = \sum_{\mathcal{H}} \prod_{i=1}^{O+H} p(X_i|X_{\pi_i}). \quad (2)$$

This expression leads to complicated conditional independence structures between observed variables depending on the tree structure. In other words, latent tree models allow complex distributions over observed variables (e.g., clique models) to be expressed in terms of more tractable joint models over the augmented variable space. This leads to a significant saving in model parametrization.

### 3 Overview of Our Technique

To address nonparametric problems in tree-structured latent variable models, we will use a crucial technique called Hilbert space embedding of distributions and conditional distributions (an introduction is provided in Appendix A and B). The key idea is that the joint density of the observed variables under a tree-structured latent variable model can be expressed using a Hilbert space embedding  $\mathcal{C}_{\mathcal{O}}$  which can be viewed as a higher order tensor (or multilinear operator). The density at point  $x_1, \dots, x_O$  can be evaluated by applying the multilinear operator on the feature-mapped data point, i.e.,  $\mathcal{C}_{\mathcal{O}} \bullet_1 \phi(x_1) \bullet_2 \dots \bullet_O \phi(x_O)$  (This operation means multiplying each side of the tensor  $\mathcal{C}_{\mathcal{O}}$  by feature vector  $\phi(x_1), \dots, \phi(x_O)$  respectively). Under this view, the information about the density is fully captured by the Hilbert space embedding  $\mathcal{C}_{\mathcal{O}}$ .

Furthermore, the presence of the latent variables endows further structures in the Hilbert space embedding  $\mathcal{C}_{\mathcal{O}}$ . That is  $\mathcal{C}_{\mathcal{O}}$  can be derived from a collection of simpler Hilbert of space embeddings  $\{\mathcal{C}\}$  each of which involves only two or three observed variables. This decomposition allows us to design computationally efficient algorithms for learning both the structure and parameters associated with the latent variable model, and carry out efficient inference based on the decomposed representation.

One characteristic of our algorithms for learning structure and parameters is that they are based on the spectral property, especially low rank property, of the Hilbert space embedding  $\mathcal{C}_{\mathcal{O}}$  due to the presence of latent variables. Our use of the spectral property of  $\mathcal{C}_{\mathcal{O}}$  results in algorithms which are based on simple linear algebraic operation (or kernel matrix operation) and yet equipped with provable guarantee.

In the following, we will describe our techniques in 5 section. First, we will connect kernel density estimation with Hilbert space embedding of distributions. Second, we will explain the decomposed representation of the joint embedding  $\mathcal{C}_{\mathcal{O}}$  of the tree-structured latent variable models. Third, we will use the decomposed form to design an efficient nonparametric inference algorithm.

Fourth, we will derive our nonparametric parameter learning algorithm. Last, we will design the nonparametric structure learning algorithm using the spectral property of the embedding.

## 4 Kernel Density Estimator as Hilbert Space Embeddings

In this section, we connect traditional kernel density estimators (KDE) (Rosenblatt, 1956; Parzen, 1962; Silverman, 1986; Wasserman, 2006) with Hilbert space embeddings of distributions (Smola et al., 2007a). These two sets of methods are closely related to each other by the fact that both are trying to model non-Gaussian distribution in a nonparametric fashion, and kernel functions play a key role in both methods. The difference is that the studies of Hilbert space embeddings focus more on injectively representing distributions in a function space, while kernel density estimations concern more about estimating the actual values of the density. By building the connections, our purpose is to show that KDE can benefit from the view of Hilbert space embedding. In particular, we will later show that the Hilbert space embedding framework allows us to exploit KDE to model latent tree structures.

### 4.1 Kernel density estimator

The kernel density estimator (KDE), also called Parzen window estimator, is a nonparametric method for estimating the density function  $p(x_1, \dots, x_O)$  for a set of continuous random variables  $\mathcal{O} = \{X_1, \dots, X_O\}$  from the domain  $\mathcal{X}_{\mathcal{O}}$ . Given a dataset  $\mathcal{D} = \{(x_1^i, \dots, x_O^i)\}_{i=1}^n$  drawn i.i.d. from  $p(x_1, \dots, x_O)$ , the KDE using product kernel is defined by

$$\hat{p}(x_1, \dots, x_O) = \frac{1}{n} \sum_{i=1}^n \prod_{j=1}^O K(x_j, x_j^i), \quad (3)$$

where  $K(x, x')$  is a kernel function. A commonly used kernel function, which we will focus on, is the Gaussian RBF kernel  $K(x, x') = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\|x - x'\|^2/2\sigma^2)$ . For the Gaussian RBF kernel, there exists a feature map

$$\phi(\cdot) : \mathbb{R} \mapsto \mathcal{F} \quad \text{such that } K(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}},$$

and the feature space has the reproducing property, *i.e.*, for all  $f \in \mathcal{F}$ ,  $f(x) = \langle f, \phi(x) \rangle_{\mathcal{F}}$ . Products of kernels are also kernels, which allows us to write  $\prod_{j=1}^O K(x_j, x_j')$  as a single inner product  $\left\langle \bigotimes_{j=1}^O \phi(x_j), \bigotimes_{j=1}^O \phi(x_j') \right\rangle_{\mathcal{F}^O}$ . Here  $\bigotimes_{j=1}^O (\cdot)$  denotes the outer product of  $O$  feature vectors which results in a rank-1 tensor of order  $O$ , and the inner product can be understood by analogy with the finite dimensional case: given  $x, y, z, x', y', z' \in \mathbb{R}^d$ ,  $(x^\top x')(y^\top y')(z^\top z') = \langle x \otimes y \otimes z, x' \otimes y' \otimes z' \rangle_{\mathbb{R}^{d^3}}$ .

### 4.2 A Hilbert space embedding view of KDE

To see the connection between KDE and Hilbert space embeddings of distributions, we compute the expected value of a KDE with respect to the random sample  $\mathcal{D}$ ,

$$\mathbb{E}_{\mathcal{D}} [\hat{p}(x_1, \dots, x_O)] = \mathbb{E}_{\mathcal{O}} \left[ \prod_{j=1}^O K(x_j, X_j) \right] = \langle \mathbb{E}_{\mathcal{O}} [\bigotimes_{j=1}^O \phi(X_j)], \bigotimes_{j=1}^O \phi(x_j) \rangle_{\mathcal{F}^O}, \quad (4)$$

where  $\mathcal{C}_\theta := \mathbb{E}_\theta \left[ \otimes_{j=1}^O \phi(X_j) \right]$  is called the Hilbert space embedding of the density  $p(x_1, \dots, x_O)$  with tensor features  $\otimes_{j=1}^O \phi(x_j)$  (Smola et al., 2007a). Furthermore, if we replace the embedding  $\mathcal{C}_\theta$  by its finite sample estimate  $\hat{\mathcal{C}}_\theta := \frac{1}{n} \sum_{i=1}^n \left( \otimes_{j=1}^O \phi(x_j^i) \right)$ , we recover the density estimator in (3).

### 4.3 Tensor expression for KDE

Using the tensor notation we can write equation (4) as

$$\langle \mathbb{E}_\theta \left[ \otimes_{j=1}^O \phi(X_j) \right], \otimes_{j=1}^O \phi(x_j) \rangle_{\mathcal{F}^O} = \mathcal{C}_\theta \bullet_1 \phi(x_1) \bullet_2 \dots \bullet_O \phi(x_O), \quad (5)$$

which means that the expected value of a KDE can be equivalently computed by first embedding the distributions into a tensor space and then multiply the embedding with the feature vectors of  $x_1, \dots, x_O$ . We note that it is not easy for traditional KDE to exploit the fact that the embedding  $\mathcal{C}_\theta$  may have low rank structure due to the latent tree structure. In the next section, we will show that we can exploit the latent tree structure and come up with a factorized estimator for  $\mathcal{C}_\theta$ .

## 5 Latent Tree Representation via Hilbert Space Embedding

When the conditional independent relation underlying a set of observed variables  $\mathcal{O} = \{X_1, \dots, X_O\}$  and hidden variables  $\mathcal{H} = \{X_{O+1}, \dots, X_{O+H}\}$  follows a latent tree structure, the joint embedding  $\mathcal{C}_\theta$  can factorize according to the tree structure. Instead of representing the density of the observed variables as a single Hilbert space embedding, we can represent it as a collection of Hilbert space embedding of conditional distributions, each of which involving only two variables. This is analogous to the factorization of a latent tree graphical model in equation (2), where we can represent such as a model using only the marginal distribution of the root node and the conditional distributions of variables with parent-child relations in the tree. Under a tree structure constraint, each variable has exactly one parent. In the nonparametric setting, we will instead use Hilbert space embedding of the marginal distributions and conditional distributions. More specifically,

- We will use the same kernel,  $L(x, x') = \langle \psi(x), \psi(x') \rangle_{\mathcal{G}}$  (with the associated RKHS  $\mathcal{G}$ ), for all discrete latent variables. In particular, each latent variable  $x$  can take up to  $k$  distinct values  $x \in \mathcal{X}_{\mathcal{H}} = \{1, \dots, k\}$ . The feature map

$$\psi(\cdot) : \{1, \dots, k\} \mapsto \mathcal{G} := \{0, 1\}^k \subset \mathbb{R}^k$$

such that  $\psi(x)$  is a length  $k$  vector with all entries equal to 0 except the  $x$ th entry take value 1. For  $x, x' \in \{1, \dots, k\}$ , their kernel function  $l(x, x')$  is computed as the Euclidean inner product of  $\psi(x)$  and  $\psi(x')$ .

- For each conditional distribution between observed continuous variable  $X_s$  and its parent  $X_{\pi_s}$ ,  $p(X_s|X_{\pi_s})$ , we also represent it as a conditional Hilbert space embedding

$$\mathcal{C}_{s|\pi_s} : \mathbb{R}^k \mapsto \mathcal{F}, \quad \text{such that} \quad \mathcal{C}_{s|\pi_s} \psi(X_{\pi_s}) := \mathbb{E}_{X_s|X_{\pi_s}} [\phi(X_s)].$$

One can think of this embedding as a second order tensor where one side is of infinite dimension but another side is of dimension  $k$ . Let the first mode of  $\mathcal{C}_{s|\pi_s}$  corresponds to variable  $X_s$ , then  $\mathcal{C}_{s|\pi_s} \psi(X_{\pi_s})$  can be equivalently written as  $\mathcal{C}_{s|\pi_s} \bullet_2 \psi(X_{\pi_s})$ . This operator will be used to integrate out observed variable  $X_s$  as we will see in later inference section.



- For each conditional distribution between discrete latent variables  $X_s$  and its parent  $X_{\pi_s}$ ,  $p(X_s|X_{\pi_s})$ , we represent it as a conditional Hilbert space embedding

$$\mathcal{C}_{s^2|\pi_s} : \mathbb{R}^k \mapsto \mathbb{R}^k \times \mathbb{R}^k, \quad \text{such that} \quad \mathcal{C}_{s^2|\pi_s} \psi(X_{\pi_s}) := \mathbb{E}_{X_s|X_{\pi_s}} [\psi(X_s) \otimes \psi(X_s)]$$

One can also think of this embedding as a third order tensor where each side is of dimension  $k$ . Let the first two modes of  $\mathcal{C}_{s^2|\pi_s}$  correspond to variable  $X_s$ , then  $\mathcal{C}_{s^2|\pi_s} \psi(X_{\pi_s})$  can be equivalently written as  $\mathcal{C}_{s^2|\pi_s} \bullet_3 \psi(X_{\pi_s})$ . The reason why we need third order tensor here is because the latent variable has one parent and two children (Figure 1). This operator will be used to integrate out variable  $X_s$  as we will see in the inference section.

- Given the latent tree structure, we will choose an arbitrary latent variable as the root node  $r$ . Then we will represent the marginal distribution,  $p(X_r)$ , of the root node of the latent tree model as a Hilbert space embedding

$$\mathcal{C}_{r^3} := \mathbb{E}_{X_r} [\psi(x_r) \otimes \psi(x_r) \otimes \psi(x_r)] : \mathbb{R}^k \times \mathbb{R}^k \times \mathbb{R}^k \mapsto \mathbb{R}.$$

One can also think of this embedding as a third order tensor where each side is of dimension  $k$ . The reason why we need third order tensor here is because the root node has three children (Figure 1). In the factorization presented in (1),  $X_r$  will appear in the conditioning of three factors. The tensor  $\mathcal{C}_{r^3}$  will be used to integrate out the variable  $X_r$  as we will see in the inference section.

With the above representations, we will then be able to perform inference task on the latent tree graphical models, such as computing the marginal distribution of all observed variables by summing out all latent variables, or conditioning on the values of some observed variables to compute the distribution of other observed variables.

In the next section, we will first explain the inference (or query) task on a latent tree graphical model assuming that we are *given* the latent tree structures and the above mentioned conditional Hilbert space embedding representation of a latent tree graphical model. In this case, we can carry out the computation using a message passing algorithm. Next, we will discuss learning the Hilbert space embedding representation given information of the tree structure and the observed variables. Then, we will present our algorithm for discovering the tree structure of a latent variable model based on Hilbert space embedding of the joint distributions of pairs of observed variables.

## 6 Inference on Latent Tree Graphical Models

In this section, we assume the latent tree structure is known, i.e., for each variable  $X_s$ , we know its parent  $X_{\pi_s}$ . In addition, we assume all the conditional Hilbert space embedding of  $p(X_s|X_{\pi_s})$  is given. Our goal is to conduct inference on the tree. For this, we can carry out the computation using a message passing algorithm.

### 6.1 Message passing algorithm in density space

We will focus on computing the expected KDE,  $\mathbb{E}_{\mathcal{D}}[\hat{p}(x_1, \dots, x_d)] = \mathbb{E}_{\mathcal{D}} \left[ \prod_{j=1}^O K(x_j, X_j) \right]$ , for the observed variables in a latent tree graphical model. We will show that the computation can be carried out based on the representation we discussed in the last section. More specifically,

in a tree graphical model, we can carry out the computation efficiently via the message passing algorithm (Pearl, 2001):

- At a leaf node (corresponding to an observed variable), we pass the following message to its parent

$$m_s(\cdot) = \mathbb{E}_{X_s|X_{\pi_s}=\cdot}[K(x_s, X_s)] \in \mathbb{R}^k.$$

This step corresponds to evaluate the expected density at  $x_s$  for variable  $X_s$  given latent variable  $X_{\pi_s}$ .

- An internal latent variable aggregates incoming messages from its two children and then sends an outgoing message to its own parent

$$m_s(\cdot) = \mathbb{E}_{X_s|X_{\pi_s}=\cdot}[m_{\iota_s}(X_s)m_{\rho_s}(X_s)] \in \mathbb{R}^k.$$

This step corresponds to summing out a latent variable  $X_s$  by first multiplying together the two messages (or intermediate results),  $m_{\iota_s}$  and  $m_{\rho_s}$ , from its left child  $X_{\iota_s}$  and right child  $X_{\rho_s}$  respectively (Figure 1).

- Finally, at the root node, all incoming messages are aggregated

$$b_r = \mathbb{E}_{\mathcal{O}} \left[ \prod_{j=1}^O K(x_j, X_j) \right] = \mathbb{E}_{X_r} [m_{\iota_r}(X_r) m_{\rho_r}(X_r) m_{\omega_r}(X_r)] \in \mathbb{R}.$$

This step corresponds to summing out the latent variable  $X_r$  at the root of the tree by first multiplying together the three messages,  $m_{\iota_r}$ ,  $m_{\omega_r}$  and  $m_{\rho_r}$ , from its three children,  $X_{\iota_r}$ ,  $X_{\omega_r}$  and  $X_{\rho_r}$  respectively (Figure 1). The result  $b_r$  is also equal to the expected KDE in this case, i.e.,

$$b_r = \mathbb{E}_{\mathcal{D}}[\hat{p}(x_1, \dots, x_d)]$$

In the case where the domains the variable  $\mathcal{X}$  is discrete with small cardinality  $|\mathcal{X}|$ , or  $\mathcal{X}$  is continuous but the random variables are Gaussians (Weiss and Freeman, 2001), computing the above update step can be carried out efficiently. For general continuous domains or discrete domains where  $|\mathcal{X}|$  is too large to enumerate, however, the expectation in the updates becomes intractable to compute. This is the situation in our case where the observed continuous variables follow general distribution, and our graphical model is a mix of such continuous variables and discrete latent variables.

A number of approaches have been used to define message passing or belief propagation in higher dimensional spaces, and for more complex probability models. Minka (2001) proposes the expectation-propagation algorithm, where only certain moments of the messages are estimated. Unfortunately, this method does not address distributions that cannot be well-characterized by the first few moments. To address this problem, Sudderth et al. (2003) represent messages as mixtures of Gaussians, however the number of mixture components grows exponentially as the message is propagated: they alleviate this problem through subsampling. Ihler and McAllester (2009) propose a particle BP approach, where messages are expressed as functions of a distribution of particles at each node, and the expectation in the function update becomes sums over the particles.

Our algorithm deals with complex continuous variables in the message passing algorithm using a different approach. More specifically we use RKHS functions to express the messages  $m_s(\cdot)$ . As

a result of this representation, the messages  $m_s(\cdot)$  from an observed leaf nodes can be combined in a straightforward linear operation in feature space that implements the sum and product steps, producing a new message  $m_{s'}(\cdot)$  that remains an RKHS function.

## 6.2 Message passing algorithm using kernel embeddings

We now express the three message update operations in the previous section using the corresponding Hilbert space embeddings and linear operations in the Hilbert space (Song et al., 2010).

- At a leaf node, we have  $m_s(\cdot) = \mathbb{E}_{X_s|X_{\pi_s}=\cdot}[K(x_s, X_s)]$ . Using the reproducing property of the kernel function,  $K(x_s, X_s) = \langle \phi(x_s), \phi(X_s) \rangle_{\mathcal{F}}$ , we have that

$$\mathbb{E}_{X_s|X_{\pi_s}=\cdot}[K(x_s, X_s)] = \mathbb{E}_{X_s|X_{\pi_s}=\cdot}[\langle \phi(x_s), \phi(X_s) \rangle_{\mathcal{F}}] = \langle \phi(x_s), \mathbb{E}_{X_s|X_{\pi_s}=\cdot}[\phi(X_s)] \rangle_{\mathcal{F}}.$$

Notice that  $\mathbb{E}_{X_s|X_{\pi_s}=\cdot}[\phi(X_s)]$  can be expressed using conditional embedding operator  $\mathcal{C}_{s|\pi_s}$ . Treating  $\mathcal{C}_{s|\pi_s}$  as a second order tensor results in the kernel embedding message update for leaf node

$$m_s(\cdot) = \mathcal{C}_{s|\pi_s} \bullet_1 \phi(x_s) \in \mathbb{R}^k. \quad (6)$$

- At internal nodes, we use a tensor product reproducing kernel Hilbert space  $\mathbb{R}^{2k} := \mathbb{R}^k \otimes \mathbb{R}^k$ , under which the product of incoming messages can be written as a single inner product,

$$m_{\iota_s}(X_s) m_{\rho_s}(X_s) = \langle m_{\iota_s}, \psi(X_s) \rangle_{\mathbb{R}^k} \langle m_{\rho_s}, \psi(X_s) \rangle_{\mathbb{R}^k} = \langle m_{\iota_s} \otimes m_{\rho_s}, \psi(X_s) \otimes \phi(X_s) \rangle_{\mathbb{R}^{2k}}.$$

Then the message update becomes

$$m_s(\cdot) = \mathbb{E}_{X_s|X_{\pi_s}=\cdot}[m_{\iota_s}(X_s) m_{\rho_s}(X_s)] = \langle m_{\iota_s} \otimes m_{\rho_s}, \mathbb{E}_{X_s|X_{\pi_s}=\cdot}[\phi(X_s) \otimes \phi(X_s)] \rangle_{\mathbb{R}^{2k}}$$

Notice that  $\mathbb{E}_{X_s|X_{\pi_s}=\cdot}[\phi(X_s) \otimes \phi(X_s)]$  can be expressed using conditional embedding operator  $\mathcal{C}_{s^2|\pi_s}$ . Treating  $\mathcal{C}_{s^2|\pi_s}$  as a third order tensor results in the kernel embedding message update for the internal nodes

$$m_s(\cdot) = \mathcal{C}_{s^2|\pi_s} \bullet_1 m_{\iota_s} \bullet_2 m_{\rho_s} \in \mathbb{R}^k. \quad (7)$$

- Finally, at the root nodes, we use the property of tensor product feature space  $\mathbb{R}^{3k} := \mathbb{R}^k \otimes \mathbb{R}^k \otimes \mathbb{R}^k$  and arrive at

$$\mathbb{E}_r[m_{\iota_r}(X_r) m_{\rho_r}(X_r) m_{\omega_r}(X_r)] = \langle m_{\iota_r} \otimes m_{\rho_r} \otimes m_{\omega_r}, \mathbb{E}_{X_r}[\phi(X_r) \otimes \phi(X_r) \otimes \phi(X_r)] \rangle_{\mathbb{R}^{3k}}$$

Notice that  $\mathbb{E}_{X_r}[\phi(X_r) \otimes \phi(X_r) \otimes \phi(X_r)]$  can be expressed using the kernel embedding  $\mathcal{C}_r^3$ . Treating  $\mathcal{C}_r^3$  as a third order tensor results in the kernel embedding message update for the root node

$$b_r = \mathcal{C}_r^3 \bullet_1 m_{\iota_r} \bullet_2 m_{\rho_r} \bullet_3 m_{\omega_r} \in \mathbb{R}. \quad (8)$$

We note that traditional kernel density estimator needs to maintain a tensor of order  $O$  involving all observed variables (i.e., Equation (5)). By exploiting the conditional independence structure of latent tree models, we only need to maintain tensors of much smaller orders. In particular, we only need to maintain tensors involving up to three variables (for each parent-child relation), then the density can be computed via message passing algorithms using these tensors of much smaller order. The overall inference algorithm is summarized in Algorithm 1.

---

**Algorithm 1** Inference

---

**In:** Latent tree structure, root node  $r$ , and the corresponding conditional embedding operators.

**Out:** Belief  $b_r$  at root.

```
1: Root the tree at node  $r$ , orient the edges in the tree pointing away from the root. The resulting
   directed acyclic graph (DAG) induces a topological ordering of the set of nodes  $\mathcal{V}$ .
2: for all  $s \in \mathcal{V}$  in reverse topological order do
3:   if  $s$  is the root  $r$  then
4:      $b_s = \mathcal{C}_{s^3} \bullet_1 m_{\iota_s} \bullet_2 m_{\rho_s} \bullet_3 m_{\omega_s}$ 
5:   else if  $s$  is an observed continuous variable taking value  $x_s$  then
6:      $m_s(\cdot) = \mathcal{C}_{s|\pi_s} \bullet_1 \phi(x_s)$ 
7:   else if  $s$  is an internal latent variable then
8:      $m_s(\cdot) = \mathcal{C}_{s^2|\pi_s} \bullet_1 m_{\iota_s} \bullet_2 m_{\rho_s}$ 
9:   end if
10: end for
```

---

## 7 Observable Representation and Parameter Estimation

In this section, we assume the latent tree structure is known, i.e., for each variable  $X_s$ , we know its parent  $X_{\pi_s}$ . Our goal is to estimate all the conditional embedding operators,  $\mathcal{C}_{s|\pi_s}$  or  $\mathcal{C}_{s^2|\pi_s}$ , associated with  $p(X_s|X_{\pi_s})$ , and the embedding,  $\mathcal{C}_{r^3}$ , associated with the root node.

From (6), (7) and (8), our key observation is that if we can recover the conditional embedding operators up to some invertible transformations, we will still be able to obtain the same final results for the message passing algorithm. More specifically,

- For each leaf node  $X_s$ , we define an invertible transformation  $T_s \in \mathbb{R}^{k \times k}$ , where the subscript is used to indicate that the transformation  $T_s$  is specific to node  $X_s$ . If we change to a different node  $X_{s'}$ , then the transformation  $T_{s'}$  will also change accordingly. Then we can transform the message,  $m_s$ , from a leaf node as

$$\tilde{m}_s(\cdot) = (\mathcal{C}_{s|\pi_s} \times_2 T_s) \bullet_1 \phi(x_s) = T_s \mathcal{C}_{s|\pi_s}^\top \phi(x_s), \quad (9)$$

where the tensor-matrix product notation  $\mathcal{C} \times_i T$  means that we multiply the  $i$ -th mode of  $\mathcal{C}$  with the columns of  $T$  (Kolda and Bader, 2009).

- For an internal node, we will introduce three invertible transformations  $T_s, T_{\iota_s}, T_{\rho_s} \in \mathbb{R}^{k \times k}$ . We note that in this case, we do not have complete freedom in choosing the invertible transformations:  $T_{\iota_s}$  and  $T_{\rho_s}$  are determined by the transformation chosen by the child node  $X_{\iota_s}$  and  $X_{\rho_s}$  of the current node  $X_s$ . Then the outgoing message,  $m_s$ , from this internal node becomes

$$\tilde{m}_s(\cdot) = (\mathcal{C}_{s^2|\pi_s} \times_1 T_{\iota_s}^{-1} \times_2 T_{\rho_s}^{-1} \times_3 T_s) \bullet_1 \tilde{m}_{\iota_s} \bullet_2 \tilde{m}_{\rho_s} \quad (10)$$

where we have defined  $\tilde{m}_{\iota_s} = T_{\iota_s} m_{\iota_s}$  and  $\tilde{m}_{\rho_s} = T_{\rho_s} m_{\rho_s}$ .

- Finally, at the root node, we also introduce three invertible transformations  $T_r, T_{\rho_r}, T_{\omega_r} \in \mathbb{R}^{k \times k}$  which are determined by the choice of transformations from its three child nodes. Then we obtain the final result

$$b_r = (\mathcal{C}_{r^3} \times_1 T_{\iota_r}^{-1} \times_2 T_{\rho_r}^{-1} \times_3 T_{\omega_r}^{-1}) \bullet_1 \tilde{m}_{\iota_r} \bullet_2 \tilde{m}_{\rho_r} \bullet_3 \tilde{m}_{\omega_r}, \quad (11)$$

where  $\tilde{m}_{\omega_r} = T_{\omega_r} m_{\omega_r}$ .

Basically, all the invertible transformations  $T$ 's cancel out with each other, and the final result  $b_r$  remains unchanged. However, these transformations provide us additional degrees of freedom for algorithm design: We can choose the invertible transforms more carefully so that the transformed representation can be recovered from observed quantities without the need for accessing the latent variables. Furthermore, we will show that these transformations  $T$ 's can be constructed from singular vectors  $U$  of cross covariance operator of certain pairs of observed variables.

To explain the idea, we will introduce a set of notation for relabeling the observed variables. Such relabeling is carried out with respect to a variable of interest and is determined by the relative positions of these variables to the focused variable. For instance, if the focused variable is  $X_s$ , then we will introduce relabeling notation,  $X_{\iota_s}^*$ ,  $X_{\rho_s}^*$  and  $X_{\pi_s}^*$ , as in Figure 1. Each notation denotes a set of variables, and their respective meanings are

- $X_{\iota_s}^*$  is the set of observed variables (or leaf nodes) which can be reached by following the direction of the edge from  $X_s$  to its left child  $X_{\iota_s}$ .
- $X_{\rho_s}^*$  is the set of observed variables which can be reached by following the direction of the edge from  $X_s$  to its right child  $X_{\rho_s}$ .
- $X_{\pi_s}^*$  is the set of observed variables which can be reached by following the reverse direction of the edge from  $X_s$  to its parent  $X_{\pi_s}$ .

In later use of these sets, each time we will typically select one variable from each set. The exact identity of the selected variable is not important. Thus, for simplicity of notation, we will also use,  $X_{\iota_s}^*$ ,  $X_{\rho_s}^*$  and  $X_{\pi_s}^*$ , to denote those selected variables. For the root node  $X_r$ , we also introduce notation  $X_{\omega_r}^*$  which denotes the additional set of variables reached by following the direction of the edge from  $X_r$  to its middle child  $X_{\omega_r}$ .

Now we consider the simple case for the transformed message  $\tilde{m}_s$  from the leaf node (Equation 9). We first find  $X_{\pi_s}^*$ , and compute the embedding,  $\mathcal{C}_{s\pi_s^*}$ , of the joint density of  $X_s$  and  $X_{\pi_s}^*$ . Let  $U_{s\pi_s^*}$  be the left singular vectors of  $\mathcal{C}_{s\pi_s^*}$  corresponding to the top  $k$  singular values. We note that the number,  $k$ , of singular vectors needs to be the same as the number of possible value that the latent variables can take; Such choice ensures that some intermediate matrices are invertible. Hence the same notation,  $k$ , is used for both cases. Then we construct the transformation  $T_s = (\mathcal{C}_{s|\pi_s}^\top U_{s\pi_s^*})^{-1}$ , and we have

$$\tilde{m}_s = (\mathcal{C}_{s|\pi_s} \times_2 (\mathcal{C}_{s|\pi_s}^\top U_{s\pi_s^*})^{-1}) \bullet_1 \phi(x_s) = (\mathcal{C}_{s|\pi_s}^\top U_{s\pi_s^*})^{-1} \mathcal{C}_{s|\pi_s}^\top \phi(x_s).$$

Then, to remove the dependency of the expression on latent variable  $X_{\pi_s}$ , we multiply both sides by  $\mathcal{C}_{\alpha_s s} U_{s\pi_s^*}$  ( $X_{\alpha_s}$  is the sibling node of  $X_s$  as in Figure 1) and obtain

$$\begin{aligned} \mathcal{C}_{\alpha_s s} U_{s\pi_s^*} \tilde{m}_s &= \mathcal{C}_{\alpha_s s} U_{s\pi_s^*} (\mathcal{C}_{s|\pi_s}^\top U_{s\pi_s^*})^{-1} \mathcal{C}_{s|\pi_s}^\top \phi(x_s) \\ &= \mathcal{C}_{\alpha_s|\pi_s} \mathcal{C}_{\pi_s \pi_s} \mathcal{C}_{s|\pi_s}^\top U_{s\pi_s^*} (\mathcal{C}_{s|\pi_s}^\top U_{s\pi_s^*})^{-1} \mathcal{C}_{s|\pi_s}^\top \phi(x_s) \\ &= \mathcal{C}_{\alpha_s|\pi_s} \mathcal{C}_{\pi_s \pi_s} \mathcal{C}_{s|\pi_s}^\top \phi(x_s) \\ &= \mathcal{C}_{\alpha_s s} \phi(x_s) \end{aligned} \tag{12}$$

where in the second and third equalities we have used Hilbert space embedding expression for the relation  $p(X_{\alpha_s}, X_s) = \sum_{x_{\pi_s}} p(X_{\alpha_s}|x_{\pi_s})p(x_{\pi_s})p(X_s|x_{\pi_s})$ , i.e.,  $\mathcal{C}_{\alpha_s s} = \mathcal{C}_{\alpha_s|\pi_s} \mathcal{C}_{\pi_s \pi_s} \mathcal{C}_{s|\pi_s}^\top$ . Finally, based on the derivation in equation (12), we have that

$$\tilde{m}_s = (\mathcal{C}_{\alpha_s s} U_{s\pi_s^*})^\dagger \mathcal{C}_{\alpha_s s} \phi(x_s),$$

which depends only on information of observed variables. Define  $\tilde{\mathcal{C}}_{s|\pi_s} := \mathcal{C}_{s|\pi_s} \times_2 T_s = (\mathcal{C}_{\alpha_s s} U_{s\pi_s^*})^\dagger \mathcal{C}_{\alpha_s s}$  as a new and transformed representation for the operator  $\mathcal{C}_{s|\pi_s}$ . Then one can replace  $\mathcal{C}_{s|\pi_s}$  by  $\tilde{\mathcal{C}}_{s|\pi_s}$  in Algorithm 1 and define a message passing algorithm in the transformed space as we will see later.

The general pattern of the derivation in (12) is that we can relate the transformed latent quantity to the observed quantities by choosing appropriate invertible transformations. Similar strategy can be applied to  $\tilde{\mathcal{C}}_{s^2|\pi_s} := \mathcal{C}_{s^2|\pi_s} \times_1 T_{\iota_s}^{-1} \times_2 T_{\rho_s}^{-1} \times_3 T_s$  in the internal message update, and to  $\tilde{\mathcal{C}}_{r^3} := \mathcal{C}_{r^3} \times_1 T_{\iota_s}^{-1} \times_2 T_{\rho_s}^{-1} \times_3 T_{\omega_r}^{-1}$  in the update at the root (for more details, see (Parikh et al., 2011)). We summarize the results below

**Theorem 1.** *The transformed quantities involving latent variables can be computed via observed quantities using the following formulas*

- For observed variables,  $\tilde{\mathcal{C}}_{s|\pi_s} = (\mathcal{C}_{\alpha_s s} U_{s\pi_s^*})^\dagger \mathcal{C}_{\alpha_s s}$ .
- For latent variables,  $\tilde{\mathcal{C}}_{s^2|\pi_s} = \mathcal{C}_{\iota_s^* \rho_s^* \pi_s^*} \times_1 U_{\iota_s^* \pi_s^*}^\top \times_2 U_{\rho_s^* \pi_s^*}^\top \times_3 (\mathcal{C}_{\pi_s^* \iota_s^*} U_{\iota_s^* \pi_s^*})^\dagger$ .
- For the root node,  $\tilde{\mathcal{C}}_{r^3} = \mathcal{C}_{\iota_r^* \rho_r^* \omega_r^*} \times_1 U_{\iota_r^* \rho_r^*}^\top \times_2 U_{\rho_r^* \omega_r^*}^\top \times_3 U_{\omega_r^* \iota_r^*}^\top$ .

Then the overall message passing algorithm can be expressed purely based on embeddings of observed variables, and it is summarized in Algorithm 2

## 7.1 Computation

When we use the factorization of Hilbert space embeddings from Theorem 1, and the message passing algorithm in Algorithm 2 for density estimation, it leads us to a very efficient algorithm for computing the expected kernel density  $\mathbb{E}_{\mathcal{D}}[\hat{p}(x_1, \dots, x_O)] = \mathbb{E}_{\mathcal{D}} \left[ \prod_{j=1}^O K(x_j, X_j) \right]$ . The main computational cost only involves a sequence of singular value decompositions of the embedding of joint distributions (or pairwise cross-covariance operators). Once the transformed quantities are obtained, we can then use them in the message passing algorithm to obtain the final belief.

Given a sample  $\mathcal{D} = \{(x_1^i, \dots, x_O^i)\}_{i=1}^n$  drawn i.i.d. from  $p(x_1, \dots, x_O)$ , the spectral algorithm for latent tree graphical models proceeds by first performing a “thin” SVD of the sample covariance operators. For instance, for two variables  $X_s$  and  $X_t$ , we denote the feature matrices by  $\Upsilon = (\phi(x_s^1), \dots, \phi(x_s^n))$  and  $\Phi = (\phi(x_t^1), \dots, \phi(x_t^n))$ , and estimate  $\hat{\mathcal{C}}_{ts} = \frac{1}{n} \Phi \Upsilon^\top$ . Then the left singular vector  $v = \Phi \alpha$  ( $\alpha \in \mathbb{R}^n$ ) can be estimated as follows

$$\Phi \Upsilon^\top \Upsilon \Phi^\top v = \beta v \Leftrightarrow \mathbf{L} \mathbf{K} \mathbf{L} \alpha = \beta \mathbf{L} \alpha \quad (\beta \in \mathbb{R}), \quad (13)$$

where  $\mathbf{K} = \Upsilon^\top \Upsilon$  and  $\mathbf{L} = \Phi^\top \Phi$  are the kernel matrices, and  $\alpha$  is the generalized eigenvector. After normalization, we have  $v = \Phi \alpha / \sqrt{\alpha^\top \mathbf{L} \alpha}$ . Then the  $U$  is the column concatenation of the top  $k$  left singular vectors, i.e.,  $\hat{U} = (v_1, \dots, v_k)$ . In practice, fast computation of the kernel

---

**Algorithm 2** Inference using observable representations

---

**In:** Latent tree structure, root node  $r$ , and the corresponding conditional embedding operators.

**Out:** Belief  $b_r$  at root.

- 1: Root the tree at node  $r$ , orient the edges in the tree pointing away from the root. The resulting directed acyclic graph (DAG) induces a topological ordering of the set of nodes  $\mathcal{V}$ .
  - 2: **for all**  $s \in \mathcal{V}$  in reverse topological order **do**
  - 3:   **if**  $s$  is the root  $r$  **then**
  - 4:     Find  $X_{\iota_r^*}$ ,  $X_{\omega_r^*}$  and  $X_{\rho_r^*}$ .
  - 5:     Compute embeddings  $\mathcal{C}_{\iota_r^* \rho_r^* \omega_r^*}$ ,  $\mathcal{C}_{\iota_r^* \rho_r^*}$ ,  $\mathcal{C}_{\rho_r^* \omega_r^*}$  and  $\mathcal{C}_{\omega_r^* \iota_r^*}$ .
  - 6:     Compute the leading  $k$  left singular vectors  $U_{\iota_r^* \rho_r^*}$ ,  $U_{\rho_r^* \omega_r^*}$  and  $U_{\omega_r^* \iota_r^*}$  of the embeddings  $\mathcal{C}_{\iota_r^* \rho_r^*}$ ,  $\mathcal{C}_{\rho_r^* \omega_r^*}$  and  $\mathcal{C}_{\omega_r^* \iota_r^*}$  respectively.
  - 7:      $\tilde{\mathcal{C}}_{r3} = \mathcal{C}_{\iota_r^* \rho_r^* \omega_r^*} \times_1 U_{\iota_r^* \rho_r^*}^\top \times_2 U_{\rho_r^* \omega_r^*}^\top \times_3 U_{\omega_r^* \iota_r^*}^\top$ .
  - 8:      $b_r = \tilde{\mathcal{C}}_{r3} \bullet_1 \tilde{m}_{\iota_r} \bullet_2 \tilde{m}_{\rho_r} \bullet_3 \tilde{m}_{\omega_r}$
  - 9:   **else if**  $s$  is an observed continuous variable taking value  $x_s$  **then**
  - 10:     Find  $X_{\pi_s^*}$  and  $X_{\alpha_s}$ .
  - 11:     Compute embeddings  $\mathcal{C}_{s\pi_s^*}$  and  $\mathcal{C}_{\alpha_s s}$ .
  - 12:     Compute the leading  $k$  left singular vectors  $U_{s\pi_s^*}$  of the embedding  $\mathcal{C}_{s\pi_s^*}$ .
  - 13:      $\tilde{\mathcal{C}}_{s|\pi_s} = (\mathcal{C}_{\alpha_s s} U_{s\pi_s^*})^\dagger \mathcal{C}_{\alpha_s s}$
  - 14:      $\tilde{m}_s(\cdot) = \tilde{\mathcal{C}}_{s|\pi_s} \bullet_1 \phi(x_s)$
  - 15:   **else if**  $s$  is an internal latent variable **then**
  - 16:     Find  $X_{\iota_s^*}$ ,  $X_{\rho_s^*}$  and  $X_{\pi_s^*}$ .
  - 17:     Compute embeddings  $\mathcal{C}_{\iota_s^* \rho_s^* \pi_s^*}$ ,  $\mathcal{C}_{\iota_s^* \pi_s^*}$ ,  $\mathcal{C}_{\rho_s^* \pi_s^*}$  and  $\mathcal{C}_{\pi_s^* \iota_s^*}$ .
  - 18:     Compute the leading  $k$  left singular vectors  $U_{\iota_s^* \pi_s^*}$ ,  $U_{\rho_s^* \pi_s^*}$  of the embeddings  $\mathcal{C}_{\iota_s^* \pi_s^*}$  and  $\mathcal{C}_{\rho_s^* \pi_s^*}$  respectively.
  - 19:      $\tilde{\mathcal{C}}_{s^2|\pi_s} = \mathcal{C}_{\iota_s^* \rho_s^* \pi_s^*} \times_1 U_{\iota_s^* \pi_s^*}^\top \times_2 U_{\rho_s^* \pi_s^*}^\top \times_3 (\mathcal{C}_{\pi_s^* \iota_s^*} U_{\iota_s^* \pi_s^*})^\dagger$
  - 20:      $\tilde{m}_s(\cdot) = \tilde{\mathcal{C}}_{s^2|\pi_s} \bullet_1 \tilde{m}_{\iota_s} \bullet_2 \tilde{m}_{\rho_s}$
  - 21:   **end if**
  - 22: **end for**
- 

SVD can be carried out by first performing an incomplete Cholesky decomposition of the kernel matrices (Shawe-Taylor and Cristianini, 2004). If we let  $A := (\alpha_1, \dots, \alpha_k) \in \mathbb{R}^{n \times k}$  be the column concatenation of the  $k$  top  $\alpha_i$ , and  $D := \text{diag}((\alpha_1^\top \mathbf{L} \alpha_1)^{-1/2}, \dots, (\alpha_k^\top \mathbf{L} \alpha_k)^{-1/2}) \in \mathbb{R}^{k \times k}$ , we can concisely express  $\hat{U} = \Phi A D$ .

When the number of samples is large, we will use incomplete Cholesky decomposition for kernel matrices to further speed up the computation. In this case the kernel matrix  $\mathbf{K}$  for samples from a variable  $X$  can be factorized as  $\mathbf{K} = R_X^\top R_X$ , where  $R_X \in \mathbb{R}^{r \times n}$  comes from Cholesky decomposition with  $r \ll n$ . Basically, with incomplete Cholesky decomposition, everything goes back to finite dimensional operations.

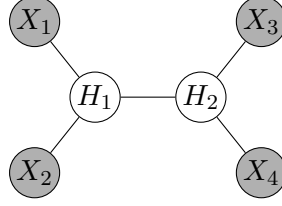


Figure 2: A tree-structured latent variable model with 4 observed variable  $X_1, X_2, X_3, X_4$ , connected via two hidden variables  $H_1, H_2$ .

## 7.2 Example

We can factorize the embedding of the marginal distributions of the four observed variables in the latent tree structure in Figure 2 as

$$\begin{aligned}\mathcal{C}_{X_1 X_2 X_3 X_4} &= \mathbb{E}_{X_1 X_2 X_3 X_4} [\phi(X_1) \otimes \phi(X_2) \otimes \phi(X_3) \otimes \phi(X_4)] \\ &= (\mathcal{C}_{H_1^3} \times_1 \mathcal{C}_{X_1|H_1} \times_2 \mathcal{C}_{X_2|H_1}) \times_3 (\mathcal{C}_{H_2^2|H_1} \times_1 \mathcal{C}_{X_3|H_2} \times_2 \mathcal{C}_{X_4|H_2}).\end{aligned}$$

Then the term  $\mathcal{C}_{H_1^3}$  can be transformed as

$$\begin{aligned}\mathcal{C}_{H_1^3} \times_1 S_{X_1} \times_2 S_{X_2} \times_3 S_{X_3} &= \mathcal{C}_{H_1^3} \times_1 (U_{X_1}^\top \mathcal{C}_{X_1|H_1}) \times_2 (U_{X_2}^\top \mathcal{C}_{X_2|H_1}) \times_3 (U_{X_3}^\top \mathcal{C}_{X_3|H_1}) \\ &= \mathcal{C}_{X_1 X_2 X_3} \times_1 U_{X_1}^\top \times_2 U_{X_2}^\top \times_3 U_{X_3}^\top.\end{aligned}$$

The term  $\mathcal{C}_{H_2^2|H_1}$  can be transformed as

$$\begin{aligned}\mathcal{C}_{H_2^2|H_1} \times_3 (\mathcal{C}_{X_1|H_1} \mathcal{C}_{H_1^2} S_{X_3}^\top S_{X_3}^{-\top}) \times_1 S_{X_3} \times_2 S_{X_4} \\ &= \mathcal{C}_{H_2^2|H_1} \times_3 (\mathcal{C}_{X_1 X_3} U_{X_3} S_{X_3}^{-\top}) \times_1 (U_{X_3}^\top \mathcal{C}_{X_3|H_2}) \times_2 (U_{X_4}^\top \mathcal{C}_{X_4|H_2}) \\ &= \mathcal{C}_{H_2^2|H_1} \times_3 (\mathcal{C}_{X_1 X_3} U_{X_3} S_{X_3}^{-\top}) \times_1 (U_{X_3}^\top \mathcal{C}_{X_3|H_2}) \times_2 (U_{X_4}^\top \mathcal{C}_{X_4|H_2}) \\ &= \mathcal{C}_{X_3 X_4 X_1} \times_1 U_{X_3}^\top \times_2 U_{X_4}^\top.\end{aligned}$$

We therefore have

$$\mathcal{C}_{H_2^2|H_1} \times_3 S_{X_3}^{-\top} \times_1 S_{X_3} \times_2 S_{X_4} = \mathcal{C}_{X_3 X_4 X_1} \times_1 U_{X_3}^\top \times_2 U_{X_4}^\top \times_3 (\mathcal{C}_{X_1 X_3} U_{X_3})^\dagger$$

and the term  $\mathcal{C}_{X_1|H_1}$  can be transformed as

$$\mathcal{C}_{X_1|H_1} S_{X_1}^{-1} = \mathcal{C}_{X_1 X_2} (U_{X_1}^\top \mathcal{C}_{X_1 X_2})^\dagger.$$

Then the computation of  $U$  using the approximated covariance matrix can be carried out as follows

$$\frac{1}{n^2} (R_{X_1} R_{X_2}^\top) (R_{X_2} R_{X_1}^\top) U_{X_1} = U_{X_1} \Gamma_1 \quad (14)$$

$$\frac{1}{n^2} (R_{X_2} R_{X_1}^\top) (R_{X_1} R_{X_2}^\top) U_{X_2} = U_{X_2} \Gamma_2 \quad (15)$$

$$\frac{1}{n^2} (R_{X_3} R_{X_1}^\top) (R_{X_1} R_{X_3}^\top) U_{X_3} = U_{X_3} \Gamma_3 \quad (16)$$

$$\frac{1}{n^2} (R_{X_4} R_{X_1}^\top) (R_{X_1} R_{X_4}^\top) U_{X_4} = U_{X_4} \Gamma_4, \quad (17)$$



where  $\Gamma$  is a diagonal matrix of the corresponding eigenvalues. Furthermore, the transformed parameters of the latent tree graphical model can also be obtained from the Cholesky factor. For example

$$\mathcal{C}_{X_1 X_2} (U_{X_1}^\top \mathcal{C}_{X_1 X_2})^\dagger \approx \frac{1}{n} R_{X_1} R_{X_2}^\top (U_{X_1}^\top \frac{1}{n} R_{X_1} R_{X_2}^\top)^\dagger. \quad (18)$$

To evaluate a new point, we first obtain the incomplete Cholesky decomposition for the new point  $r_{X_1}, r_{X_2}, r_{X_3}, r_{X_4}$ . Then we obtain the density estimation by

$$\mathcal{C}_{X_1 X_2 X_3 X_4} \bullet_1 r_{X_1} \bullet_2 r_{X_2} \bullet_3 r_{X_3} \bullet_4 r_{X_4} \quad (19)$$

which can be computed by message passing algorithm.

## 8 Structure Learning

In the last section we focuses on estimating the Hilbert space embedding under the assumption that the structure of the latent tree is known. In this section, we focus on learning latent tree structure based observational data. We exploit a distance-based method for iteratively constructing latent trees which takes a distance matrix between all pairs of observed variables as input and outputs an estimated tree structure by iteratively adding hidden nodes. We provide theoretical guarantees of the structure learning algorithm. We start with some definitions.

### 8.1 Tree Metric and Pseudo-determinant for Non-Gaussian Variables

One concept that plays a pivotal role in our latent tree structure learning algorithm is tree metric. More specifically, if the joint probability distribution  $p(\mathcal{X})$  has a latent tree structure, then a distance measure  $d_{st}$  between an arbitrary pair of variables  $X_s$  and  $X_t$  is called tree metric if it satisfies the path additive condition:  $d_{st} = \sum_{(u,v) \in \text{Path}(s,t)} d_{uv}$ . The formal definition of tree metric is as follows.

**Definition 2** (Tree Metric). *Let  $\mathcal{V}$  be a set of nodes. A metric  $(\mathcal{V}, d)$  is a tree metric if there is a tree  $\mathcal{T}$  with non-negative edge lengths whose nodes contain  $\mathcal{V}$  such that for every  $s, t \in \mathcal{V}$ , we have that  $d_{st}$  is equal to the sum of the lengths of the edges on the unique  $(s, t)$  path in the tree  $\mathcal{T}$ . Tree metrics are also called additive metrics.*

For discrete and Gaussian variables, tree metric can be defined via the determinant  $|\cdot|$

$$d_{st} = -\log |\mathcal{C}_{st}| + \frac{1}{2} \log |\mathcal{C}_{ss}| + \frac{1}{2} \log |\mathcal{C}_{tt}|, \quad (20)$$

where  $\mathcal{C}_{st}$  is the joint probability matrix in the discrete case and the covariance matrix in the Gaussian case;  $\mathcal{C}_{ss}$  is the diagonalized marginal probability matrix in the discrete case and marginal variance matrix in the Gaussian case. However, the definition of tree metric in (20) is restrictive since it requires all discrete variables to have the same number of states and all Gaussian variables to have the same dimension. For our case where the observed variables are continuous and non-Gaussian. The problem with the above definition is that determinant is only defined for square and non-singular matrices. Therefore, the definition in (20) is not suitable. To overcome this drawback, we define a tree metric based on pseudo-determinant which works for more general operators. First, we list an extra assumption.

(A2) Let  $\sigma_k(C_{st})$  be the  $k$ th singular value of  $\mathcal{C}$ , we assume that  $\sigma_{\min} := \inf_{s,t} \sigma_k(C_{st}) > 0$ .

Under the condition that the value  $k$  in Assumption (A1) is known, we define the pseudo-determinant of a covariance operator  $\mathcal{C}$  as the product of all its non-zero singular values, i.e.,

$$|\mathcal{C}|_* = \prod_{i=1}^k \sigma_i(\mathcal{C}).$$

We then generalize the definition of tree metric in (20) from the covariance matrix setting to the more general covariance operator setting. In particular, we define a nonparametric distance metric between two variables  $s$  and  $t$  as

$$d_{st} = -\log |\mathcal{C}_{st}|_* + \frac{1}{2} \log |\mathcal{C}_{ss}|_* + \frac{1}{2} \log |\mathcal{C}_{tt}|_*. \quad (21)$$

The following theorem shows that (21) is a valid tree metric.

**Theorem 3.** *The distance define in equation (21) is a tree metric.*

*Proof.* We prove this by induction on the path length. We first show that the additive property holds for a path  $X_s - X_u - X_t$  which only involves a single hidden variable  $X_u$ . For this, we exploit the relationship between eigenvalues and singular values and have

$$d_{st} = -\frac{1}{2} \log |\mathcal{C}_{st}\mathcal{C}_{st}^\top|_* + \frac{1}{4} \log |\mathcal{C}_{ss}\mathcal{C}_{ss}^\top|_* + \frac{1}{4} \log |\mathcal{C}_{tt}\mathcal{C}_{tt}^\top|. \quad (22)$$

Furthermore, using the Markov property, we factorize  $|\mathcal{C}_{st}\mathcal{C}_{st}^\top|_*$  into  $|\mathcal{C}_{s|u}\mathcal{C}_{uu}\mathcal{C}_{t|u}^\top\mathcal{C}_{t|u}\mathcal{C}_{uu}\mathcal{C}_{s|u}^\top|_*$ . According to the Sylvester's determinant theorem, the latter is equal to  $|\mathcal{C}_{s|u}^\top\mathcal{C}_{s|u}\mathcal{C}_{uu}\mathcal{C}_{t|u}^\top\mathcal{C}_{t|u}\mathcal{C}_{uu}|_*$  by flipping  $\mathcal{C}_{s|u}^\top$  to the front. By introducing two copies of  $|\mathcal{C}_{uu}|$  and rearranging the terms, we have

$$|\mathcal{C}_{st}\mathcal{C}_{st}^\top|_* = \frac{|\mathcal{C}_{s|u}\mathcal{C}_{uu}\mathcal{C}_{s|u}^\top|_* |\mathcal{C}_{t|u}\mathcal{C}_{uu}\mathcal{C}_{t|u}^\top|_*}{|\mathcal{C}_{uu}| |\mathcal{C}_{uu}|} = \frac{|\mathcal{C}_{su}\mathcal{C}_{su}^\top|_* |\mathcal{C}_{tu}\mathcal{C}_{tu}^\top|_*}{|\mathcal{C}_{uu}\mathcal{C}_{uu}|_*}. \quad (23)$$

We then plug this into (22) and get

$$\begin{aligned} d_{st} &= -\frac{1}{2} \log |\mathcal{C}_{su}\mathcal{C}_{su}^\top|_* - \frac{1}{2} \log |\mathcal{C}_{tu}\mathcal{C}_{tu}^\top|_* + \frac{1}{2} \log |\mathcal{C}_{uu}\mathcal{C}_{uu}^\top|_* + \frac{1}{4} \log |\mathcal{C}_{ss}\mathcal{C}_{ss}^\top|_* + \frac{1}{4} \log |\mathcal{C}_{tt}\mathcal{C}_{tt}^\top|_* \\ &= d_{su} + d_{ut}. \end{aligned}$$

The proof of the general cases follows similar arguments.  $\square$

## 8.2 Empirical Tree Metric Estimator and Its Concentration Property

The definition in (21) involves population quantities. Given observational data, the pseudo-determinant tree distance between two variables  $s$  and  $t$  can be estimated by the following plug-in estimator

$$\hat{d}_{st} = -\sum_{i=1}^k \log [\sigma_i(\hat{\mathcal{C}}_{st})] + \frac{1}{2} \sum_{i=1}^k \log [\sigma_i(\hat{\mathcal{C}}_{ss})] + \frac{1}{2} \sum_{i=1}^k \log [\sigma_i(\hat{\mathcal{C}}_{tt})], \quad (24)$$

where  $\hat{\mathcal{C}}$  is an estimator of the covariance operator. From the definition in (24), we see that the key to evaluate the tree metric is to calculate  $\sigma_i(\hat{\mathcal{C}}_{st})$  and  $\sigma_i(\hat{\mathcal{C}}_{ss})$ . For this, we use the same generalized eigenvalue decomposition method as described in Section 7.1.

In this subsection, we show that the estimated distance  $\widehat{d}_{st}$  converges to its popular quantity  $d_{st}$  with a fast rate of convergence. More specifically, let  $K(x, x')$  be a universal reproducing kernel satisfying

$$\sup_{x, x'} K(x, x') \leq \kappa.$$

We define

$$T_{\mathcal{F}}^{(ss)} = \int_{\Omega} \langle \cdot, K(x_s, \cdot) \rangle_{\mathcal{F}} K(x_s, \cdot) dP(x_s) = \mathcal{C}_{ss}, \quad (25)$$

$$T_n^{(ss)} = \frac{1}{n} \sum_{i=1}^n \langle \cdot, K(x_s^i, \cdot) \rangle_{\mathcal{F}} K(x_s^i, \cdot) = \widehat{\mathcal{C}}_{ss}, \quad (26)$$

and

$$T_{\mathcal{F}}^{(st)} = \int_{\Omega \times \Omega} \langle \cdot, K(x_t, \cdot) \rangle_{\mathcal{F}} K(x_s, \cdot) dP(x_s, x_t) = \mathcal{C}_{st}, \quad (27)$$

$$T_n^{(st)} = \frac{1}{n} \sum_{i=1}^n \langle \cdot, K(x_t^i, \cdot) \rangle_{\mathcal{F}} K(x_s^i, \cdot) = \widehat{\mathcal{C}}_{st}. \quad (28)$$

The following lemmas from Rosasco et al. (2010) show that  $T_{\mathcal{F}}^{(ss)}$  and  $T_{\mathcal{F}}^{(st)}$  have the same set of eigenvalues (up to possible zero entries) as  $T_n^{(ss)}$  and  $T_n^{(st)}$ .

**Lemma 4** (Rosasco et al. (2010)).  *$T_{\mathcal{F}}^{(ss)}$  and  $T_{\mathcal{F}}^{(st)}$  have the same set of eigenvalues (up to possible zero entries) as  $T_n^{(ss)}$  and  $T_n^{(st)}$ .*

The following theorem is proved by Rosasco et al. (2010)

**Theorem 5.** *Let  $\delta \in (0, 1)$  and  $\|\cdot\|_{HS}$  be the Hilbert-Schmidt norm, we have*

$$\mathbb{P} \left( \left\| T_{\mathcal{F}}^{(ss)} - T_n^{(ss)} \right\|_{HS} \leq \frac{2\sqrt{2}\kappa\delta}{\sqrt{n}} \right) \geq 1 - 2\exp(-\delta). \quad (29)$$

$$\mathbb{P} \left( \left\| T_{\mathcal{F}}^{(st)} - T_n^{(st)} \right\|_{HS} \leq \frac{2\sqrt{2}\kappa\delta}{\sqrt{n}} \right) \geq 1 - 2\exp(-\delta). \quad (30)$$

*Proof.* The result in (29) has been shown by Rosasco et al. (2010). We only need to show (30). Let  $\xi_i$  be defined as

$$\xi_i = \langle \cdot, K(x_t^i, \cdot) \rangle_{\mathcal{F}} K(x_s^i, \cdot) - T_{\mathcal{H}}^{(st)}. \quad (31)$$

It is easy to see that  $\mathbb{E}\xi_i = 0$ . Furthermore, we have

$$\sup_{s, t} \left\| \langle \cdot, K(x_t, \cdot) \rangle_{\mathcal{F}} K(x_s, \cdot) \right\|_{HS}^2 \leq \kappa^2, \quad (32)$$

which implies that  $\left\| T_{\mathcal{F}}^{(st)} \right\|_{HS} \leq 2\kappa$ . The result then follows from the Hoeffding's inequality in Hilbert space.  $\square$

The next theorem provides a uniform concentration result of the estimated tree metric  $\hat{d}_{st}$  to its population quantity.

**Theorem 6.** *Under Assumption (A3), we have*

$$\mathbb{P} \left( \sup_{st} \left| \hat{d}_{st} - d_{st} \right| \leq \frac{8\kappa}{\sigma_{\min}} \cdot \sqrt{\frac{2k \log |\mathcal{O}|}{n}} \right) \geq 1 - o(1). \quad (33)$$

*Proof.* Recall from Assumption (A2) that  $\sigma_{\min} = \inf_{s,t} \sigma_k(\mathcal{C}_{st})$ , there exists a finite  $n_0$ , for  $n \geq n_0$ , we have, for  $1 \leq i \leq k$ ,

$$\frac{1}{2} \leq \inf_{s,t} \frac{\sigma_i(\hat{\mathcal{C}}_{st})}{\sigma_i(\mathcal{C}_{st})} \leq \sup_{s,t} \frac{\sigma_i(\hat{\mathcal{C}}_{st})}{\sigma_i(\mathcal{C}_{st})} \leq \frac{3}{2}.$$

We then have

$$\begin{aligned} \left| \sum_{i=1}^k \log[\sigma_i(\hat{\mathcal{C}}_{st})] - \sum_{i=1}^k \log[\sigma_i(\mathcal{C}_{st})] \right| &\leq \sum_{i=1}^k \left| \log \left[ \frac{\sigma_i(\hat{\mathcal{C}}_{st}) - \sigma_i(\mathcal{C}_{st})}{\sigma_i(\mathcal{C}_{st})} + 1 \right] \right| \\ &\leq 2 \sum_{i=1}^k \frac{|\sigma_i(\hat{\mathcal{C}}_{st}) - \sigma_i(\mathcal{C}_{st})|}{\sigma_i(\mathcal{C}_{st})} \\ &\leq \frac{2}{\sigma_{\min}} \sum_{i=1}^k |\sigma_i(\hat{\mathcal{C}}_{st}) - \sigma_i(\mathcal{C}_{st})| \\ &\leq \frac{2\sqrt{k}}{\sigma_{\min}} \sqrt{\sum_{i=1}^k [\sigma_i(\hat{\mathcal{C}}_{st}) - \sigma_i(\mathcal{C}_{st})]^2} \\ &\leq \frac{2\sqrt{k}}{\sigma_{\min}} \|T_{\mathcal{F}}^{(st)} - T_n^{(st)}\|_{HS} \\ &\leq \frac{\sqrt{k}}{\sigma_{\min}} \frac{4\sqrt{2}\kappa\sqrt{\delta}}{\sqrt{n}} \end{aligned}$$

with probability larger than  $1 - 2\exp(-\delta)$ . Here the second to last inequality follows from the Mirsky Theorem. Using the above arguments and the union bound, we get the desired result.  $\square$

### 8.3 Structure Learning Algorithm

Once we define the tree metric as in the previous section, we can the neighbor-joint algorithm from the phylogeny tree literature to recover the latent tree structure (Saitou et al., 1987). The neighbor joint algorithm takes a distance matrix between any pair of observed variables as input and output a tree by iteratively adding hidden nodes. More specifically, let  $s$  and  $t$  be two nodes in the latent tree structure. They could be either observed or latent variables. We call  $d_{st}$  to be the information distance between  $s$  and  $t$ . The neighbor joining algorithm requires the input of the distance matrix between all observed variables. We summarize the algorithm in Algorithm 3.

The next theorem shows that the neighbor joining algorithm recovers the true latent tree structure with high probability.

---

**Algorithm 3** Neighbor Joining Algorithm

---

**Input:** Pairwise distance  $d_{st}$  between observed variables in  $\mathcal{O}$

**Output:** Latent tree structure  $\mathcal{T} = (\mathcal{V}, \mathcal{E})$

Initialize the latent tree structure:  $\mathcal{V} = \{1, \dots, |\mathcal{O}|\}, \mathcal{E} = \emptyset$

Define a working set:  $\mathcal{N} = \{1, \dots, |\mathcal{O}|\}$

$u = |\mathcal{O}| + 1$

**while**  $|\mathcal{N}| \geq 3$  **do**

**for**  $s \in \mathcal{N}$  **do**

**for**  $t \in \mathcal{N}, t \neq s$  **do**

$$Q_{st} = Q_{ts} = (|\mathcal{N}| - 2)d_{st} - \sum_{l=1}^{|\mathcal{N}|} d_{sl} - \sum_{l=1}^{|\mathcal{N}|} d_{tl}$$

**end for**

**end for**

$(s^*, t^*) = \operatorname{argmin}_{s, t \in \mathcal{N}, s \neq t} Q_{st}$

  Create a new node with index  $u$  to join node  $s^*$  and  $t^*$

  Update distances to the new node  $u$ :

$$d_{s^*u} = d_{us^*} = \frac{1}{2}d_{s^*t^*} + \frac{1}{2(|\mathcal{N}| - 2)} \left( \sum_{l=1}^{|\mathcal{N}|} d_{s^*l} - \sum_{l=1}^{|\mathcal{N}|} d_{t^*l} \right),$$

$$d_{t^*u} = d_{ut^*} = d_{s^*u} - d_{s^*t^*}$$

$$d_{lu} = d_{ul} = \frac{1}{2}(d_{s^*l} + d_{t^*l} - d_{s^*t^*}), \forall l \in \mathcal{N}, l \neq s^*, l \neq t^*$$

Update the working set:

$$\mathcal{N} = \mathcal{N} \setminus \{s^*, t^*\}, \quad \mathcal{N} = \mathcal{N} \cup \{u\}, \quad u = u + 1$$

Update latent tree structure  $\mathcal{T}$ :

$$\mathcal{V} = \mathcal{V} \cup \{u\}, \quad \mathcal{E} = \mathcal{E} \cup \{(u, s^*), (u, t^*)\}$$

**end while**

Create a new node with index  $u$  to join the 3 remaining nodes  $s, t, l \in \mathcal{N}$ , and update the latent tree structure  $\mathcal{T}$ :

$$\mathcal{V} = \mathcal{V} \cup \{u\}, \quad \mathcal{E} = \mathcal{E} \cup \{(u, s), (u, t), (u, l)\}$$

---

**Theorem 7** (Sparsistency). *Under Assumptions (A1) and (A2), let  $\hat{T}$  be the estimated tree structure using the neighbor joining algorithm and  $T^*$  be the true tree structure. We define*

$$I_{\min} := \min_{s, t} \{d_{st}, (s, t) \in T^*\}, \quad (34)$$

where  $d_{st}$  is the population tree metric. Then, under the condition that

$$\frac{n}{k \log |\mathcal{O}|} \cdot I_{\min} \sigma_{\min}^2 \rightarrow \infty, \quad (35)$$

we have  $\liminf_{n \rightarrow \infty} \mathbb{P}(\hat{T} = T^*) = 1$ .

From the above theorem, we see that even though our latent variable tree model is nonparametric, we get the nearly parametric scaling (we allow the dimension  $|\mathcal{O}|$  to increase almost exponentially fast than the sample size  $n$ ) for structure estimation when  $k$ ,  $I_{\min}$ , and  $\sigma_{\min}$  are constants.

*Proof.* Our analysis is based on the stability result of the neighbor joining algorithm. From Theorem 34 of Mihaescu et al. (2007), we have that, if the estimated tree metric  $\hat{d}_{st}$  satisfies

$$\max_{s,t} |\hat{d}_{st} - d_{st}| \leq \frac{I_{\min}}{4}, \quad (36)$$

the neighbor joining algorithm correctly recovers the latent tree structure.

From Theorem 6, we have

$$\mathbb{P} \left( \sup_{st} |\hat{d}_{st} - d_{st}| \leq \frac{8\kappa}{\sigma_{\min}} \cdot \sqrt{\frac{2k \log |\mathcal{O}|}{n}} \right) \geq 1 - o(1). \quad (37)$$

Therefore, it suffices if

$$\frac{8\kappa}{\sigma_{\min}} \cdot \sqrt{\frac{2k \delta \log |\mathcal{O}|}{n}} \leq \frac{I_{\min}}{4}. \quad (38)$$

This proves the desired result.  $\square$

## 9 Experiments

We evaluate our method on synthetic data as well as a real-world crime/communities dataset (Asuncion and Newman, 2007; Redmond and Baveja, 2002). For all experiments we compare to 2 existing approaches. The first is to assume the data is multivariate Gaussian and use the tree metric defined in (Choi et al., 2010) (which is essentially a function of the correlation coefficient). The second existing approach we compare to is the Nonparanormal (NPN) (Liu et al., 2009) which assumes that there exist marginal transformations  $f_1, \dots, f_p$  such that  $f(X_1), \dots, f(X_p) \sim N(\mu, \Sigma)$ . If the data comes from a Nonparanormal distribution, then the transformed data are assumed to be multivariate Gaussian and the same tree metric as the Gaussian case can be used on the transformed data. Our approach makes much fewer assumptions about the data than either of these two methods which can be more favorably in practice.

To perform inference in our approach, we use the spectral algorithm described earlier in the paper. For inference in the Gaussian (and nonparanormal) cases, we use the technique in (Choi et al., 2010) to learn the model parameters (covariance matrix). Once the covariance matrix has been estimated, marginalization in a Gaussian graphical model reduces to solving a linear equation of one variable if we are only computing the marginal of one variable given a set of evidence (Bickson, 2008).

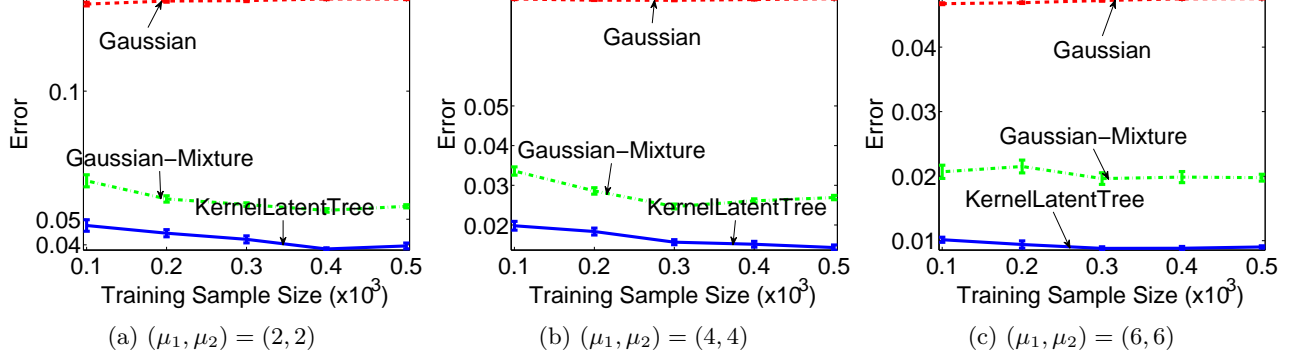


Figure 3: Density estimation of 2-dimensional mixture of laplace distributions.

### 9.1 Synthetic data: density estimation

Before moving on to larger experiments, we first show that our model assumptions can result in more accurate density estimation of non-Gaussian distributions. The underlying data is generated as a two dimensional mixture of exponentials:

$$p(x_1, x_2) \propto \exp(\|x_1 - \mu_1\| + \|x_2 - \mu_2\|) + \exp(\|x_1 + \mu_1\| + \|x_2 + \mu_2\|) \quad (39)$$

Note that the first component has mean  $(\mu_1, \mu_2)$  while the second component has mean  $(-\mu_1, -\mu_2)$ . We experiment with the different values  $(\mu_1, \mu_2) = (2, 2)$ ,  $(\mu_1, \mu_2) = (4, 4)$ , and  $(\mu_1, \mu_2) = (6, 6)$ . For all methods we evaluate the density on a grid  $\mathcal{G}$  of evenly spaced points in  $[-2\mu, 2\mu] \times [-2\mu, 2\mu]$ . The total error is measured as:  $err = \sqrt{\sum_{(x_1, x_2) \in \mathcal{G}} \|p(x_1, x_2) - \hat{p}(x_1, x_2)\|^2}$ .

Figure 3, shows the results where we compare our approach with the Gaussian and Gaussian mixture distributions. As expected, the problem is more difficult when the components are closer together. Our method performs the best for all the cases.

### 9.2 Synthetic data: structure recovery.

The second experiment is to demonstrate how our method compares to the Gaussian and Nonparametric methods in terms of structure recovery for larger trees. We experiment with 3 different tree types (each with 64 leaves or observed variables): a balanced binary tree, a completely binary skewed tree (like an HMM), and randomly generated binary trees. Furthermore we explore with two types of underlying distributions: **(1)** A multivariate Gaussian with mean zero and inverse covariance matrix that respects the tree structure. **(2)** A highly non-Gaussian distribution that uses the following generative process to generate the  $n$ -th sample from a node  $s$  in the tree (denoted  $x_s^{(n)}$ ): If  $s$  is the root, sample from a mixture of 2 Gaussians. Else, with probability  $\frac{1}{2}$  sample from a Gaussian with mean  $-x_{\pi_s}^{(n)}$  and with probability  $\frac{1}{2}$  sample from a Gaussian with mean  $x_{\pi_s}^{(n)}$ .

We vary the training sample size from 200 to 100,000. Once we have computed the empirical tree distance matrix for each algorithm, we use the neighbor joining algorithm (Saitou et al., 1987) to learn the trees. For evaluation we compare the number of hops between each pair of leaves in the true tree to the estimated tree. For a pair of leaves  $i, j$  the error is defined as:  $error(i, j) = \frac{|hops^*(i, j) - \widehat{hops}(i, j)|}{hops^*(i, j)} + \frac{|hops^*(i, j) - \widehat{hops}(i, j)|}{\widehat{hops}(i, j)}$ , where  $hops^*$  is the true number of hops and

$\widehat{hops}$  is the estimated number of hops. The total error is then computed by adding the error for each pair of leaves.

The performance of our method depends on the number of eigenvalues chosen and we experimented with 2, 5 and 8 singular values. Furthermore, we choose the bandwidth  $\sigma$  for the Gaussian RBF kernel needed for the covariance operators using median distance between pairs of training points.

When the underlying distribution is not Gaussian, our method performs better than the Gaussian and Nonparanormal methods for all the tree structures. This is to be expected, since the non-Gaussian data we generated is neither Gaussian or Nonparanormal, yet our method is able to learn the structure correctly. We also note that balanced binary trees are the easiest to learn while the skewed trees are the hardest (Figure 4).

Even when the underlying distribution is Gaussian, our method still performs very well compared to the Gaussian and NPN approaches and outperforms them for the binary and balanced trees. It performs worse for the skewed case likely due to the fact that the eigenvalues (dependence) decay along the length of the tree leading to larger errors in the empirical distance matrix.

Although it would be interesting to compare to the pouch latent tree model of Poon et al. Poon et al. (2010), their model assumes multiple observed variables can exist in the same leaf of the latent tree (unlike our approach) which makes a direct structure comparison difficult.

### 9.3 Synthetic data: model selection.

Next we evaluate the ability of our model to select the correct number of singular values via held-out likelihood. For this experiment we use a balanced binary tree with 16 leaves (total of 31 nodes) and 100000 samples. A different generative process is used so that it is clear what the correct number of singular values should be (When the hidden state space is continuous like in our first synthetic experiment this is unclear). Each internal node is discrete and takes on  $d$  values. The leaf is a mixture of  $d$  gaussians where which Gaussian to sample from is dictated by the discrete value of the parent.

We vary  $d$  from 2 through 5 and then run our method for a range of 2 through 8 singular values. We select the model that has the highest likelihood computed using our spectral algorithm on a hold-out set of 500 examples. We then take the difference between the number of singular values chosen and the true singular values, and plot histograms of this difference (Ideally all the trials should be in the zero bin). The experiment is run for 20 trials. As we can see in Figure 5, when  $d$  is low, the held-out likelihood computed by our method does a fairly good job in recovering the correct number. However, as the true number of eigenvalues rises our method under-estimate the true number (although it is still fairly close).

### 9.4 Crime Dataset.

Finally, we explore the performance of our method on a communities and crime dataset from the UCI repository (Asuncion and Newman, 2007; Redmond and Baveja, 2002). In this dataset several real valued attributes are collected for several communities, such as ethnicity proportions, income, poverty rate, divorce rate etc., and the goal is to predict the number of violent crimes (proportional to size of community) that occur based on these attributes. In general these attributes are highly skewed and therefore not well characterized by a Gaussian model.



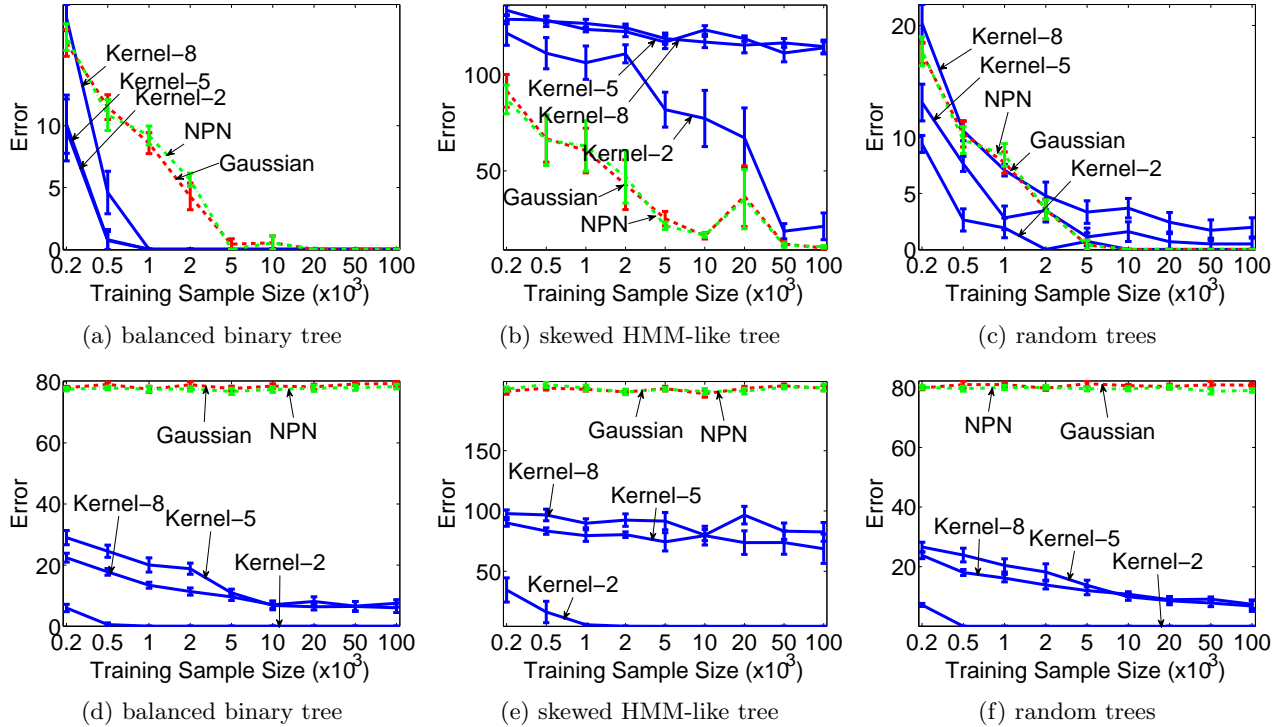


Figure 4: Comparison of our kernel structure learning method to the Gaussian and Nonparanormal methods on different tree structures. Top row: data points are generated from Gaussian distributions with latent variables connected by a tree structure. Bottom row: data points are generated from mixture of Gaussian distributions with latent variables connected by a tree structure. Especially in the latter case, our kernel structure learning method is able to adapt the data distributions and recover the structure in a much more accurate way.

We divide the data into 1400 samples for training, 300 samples for model selection (held-out likelihood), and 300 samples for testing. We pick the first 50 of these attributes, plus the violent crime variable and construct a latent tree using our tree metric and neighbor joining algorithm (Saitou et al., 1987). We depict the tree in Figure 6 and highlight a few coherent groupings. For example, the “elderly” group attributes are those related to retirement and social security (and thus correlated). The large clustering in the center is where the class variable (violent crimes) is located next to the poverty rate, and the divorce rate among other relevant variables. Other groupings include type of occupation and education level as well as ethnic proportions. Thus, overall our method is able to capture sensible relationships.

For a more quantitative evaluation, we condition on a set of  $\mathcal{E}$  evidence variables where  $|\mathcal{E}| = 30$  and predict the violent crimes class label. We experiment with a varying number of sizes of the training set from 200 to 1400. At test, we evaluate on all the 300 test examples for 10 randomly chosen evidence sets of evidence variables. Since the crime variable is a number between 0 and 1, our error measure is simply  $err(\hat{c}) = |\hat{c} - c^*|$  (where  $\hat{c}$  is the predicted value and  $c^*$  is the true value).

In this experiment, in addition to comparing with the Gaussian and the Nonparanormal, we also

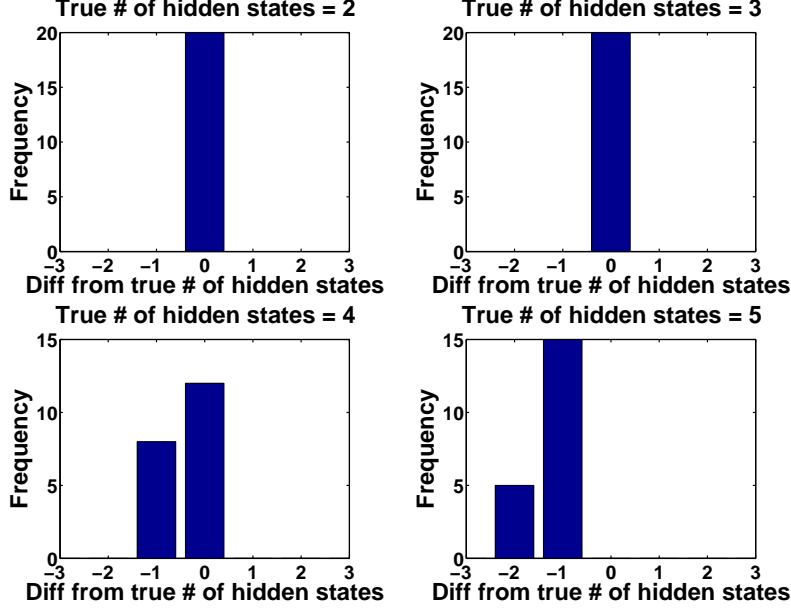


Figure 5: Histogram of the differences between the estimated number of hidden states and the true number of states.

compare with two standard classifiers, (Gaussian) Naive Bayes and Linear Regression. Although Naive Bayes can easily handle missing values, linear regression cannot. To deal with this problem, we simply use the mean value of a variable if it is not in  $\mathcal{E}$  (this performed much better than setting it to zero).

As one can see in Figure 6 our method outperforms all the other approaches. We find that our method performs similarly for different choices of  $\mathcal{E}$ . Moreover, the accuracy of the Gaussian and nonparanormal vary widely for different evidence sets (and thus the more erratic overall performance). Thus, in this case our method is better able to capture the skewed distributions of the variables than the other methods.

## 10 Conclusions

We present a method that uses Hilbert space embeddings of distributions that can recover the latent tree structures, and perform local-minimum-free spectral learning and inference for continuous and non-Gaussian variables. Both simulation and results on real datasets show the advantage of our proposed approach for non-Gaussian data.

## A Kernel Embedding of Distributions

We begin by providing an overview of Hilbert space embedding of distributions, which are *implicit* mappings of distributions into potentially *infinite* dimensional feature spaces. A reproducing kernel Hilbert space (RKHS)  $\mathcal{F}$  on  $\Omega$  with a kernel  $K(x, x')$  is a Hilbert space of functions  $f : \Omega \mapsto \mathbb{R}$  with inner product  $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ . Its element  $K(x, \cdot)$  satisfies the reproducing property:  $\langle f(\cdot), K(x, \cdot) \rangle_{\mathcal{F}} = f(x)$ , and consequently,  $\langle K(x, \cdot), K(x', \cdot) \rangle_{\mathcal{F}} = K(x, x')$ , meaning that we can view the evaluation of a function  $f$  at any point  $x \in \Omega$  as an inner product. Alternatively,  $K(x, \cdot)$  can be viewed as

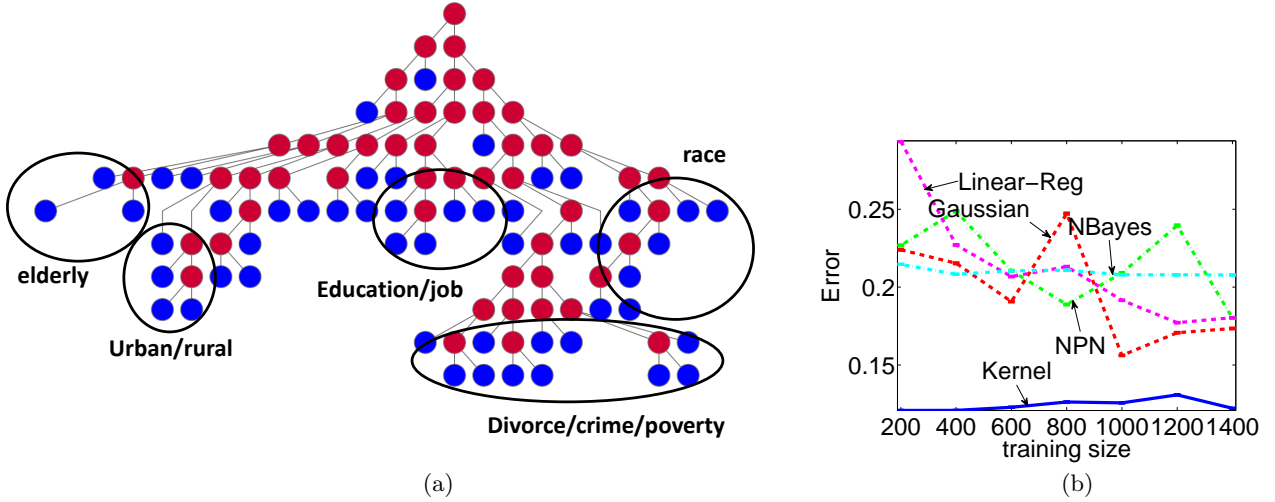


Figure 6: (a) visualization of kernel latent tree learned from crime data (b) Comparison of our method to Gaussian and NPN in predictive task.

an implicit feature map  $\phi(x)$  where  $K(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$ .<sup>1</sup> Popular kernel functions on  $\mathbb{R}^d$  include the polynomial kernel  $K(x, x') = (\langle x, x' \rangle + c)^r$  and the Gaussian RBF kernel  $K(x, x') = \exp(-\sigma \|x - x'\|^2)$ , where  $\|\cdot\|$  is the Euclidean norm on the corresponding domain. Kernel functions have also been defined on graphs, time series, dynamical systems, images and other structured objects (Schölkopf et al., 2004). Thus the methodology presented below can readily be generalized to a diverse range of data types as long as kernel functions are defined for them.

### A.1 Population Definition

The Hilbert space embedding approach represents a probability distribution  $P(X)$  by an element in the RKHS associated with a kernel function (Fukumizu et al., 2004; Berlinet and Thomas-Agnan, 2004; Smola et al., 2007b; Sriperumbudur et al., 2008, 2010),

$$\mu_X := \mathbb{E}_X [\phi(X)] = \int_{\Omega} \phi(x) dP(x), \quad (40)$$

where the distribution  $P(X)$  is mapped to its expected feature map, *i.e.*, to a point in a potentially infinite-dimensional and implicit feature space. The mean embedding  $\mu_X$  has the property that the expectation of any RKHS function  $f$  can be evaluated as an inner product in  $\mathcal{F}$ , *i.e.*,

$$\langle \mu_X, f \rangle_{\mathcal{F}} := \mathbb{E}_X [f(X)] \quad \text{for all } f \in \mathcal{F}.$$

Kernel embeddings can be readily generalized to the joint distributions of two or more variables using tensor product feature spaces. For instance, we can embed a joint distribution of two variables  $X$  and  $Y$  into a tensor product feature space  $\mathcal{F} \otimes \mathcal{F}$  by

$$\mathcal{C}_{XY} := \mathbb{E}_{XY} [\phi(X) \otimes \phi(Y)] = \int_{\Omega \times \Omega} \phi(x) \otimes \phi(y) dP(x, y) \quad (41)$$

<sup>1</sup>For notational simplicity, we use the same kernel for  $y$ , *i.e.*,  $K(y, y') = \langle \phi(y), \phi(y') \rangle_{\mathcal{F}}$

where we assume for simplicity that the two variables share the same domain  $\Omega$  and kernel  $k$ , and the tensor product features satisfy  $\langle \phi(x) \otimes \phi(y), \phi(x') \otimes \phi(y') \rangle_{\mathcal{F} \otimes \mathcal{F}} = K(x, x')K(y, y')$ .

The joint embeddings can also be viewed as an uncentered cross-covariance operator  $\mathcal{C}_{XY} : \mathcal{F} \mapsto \mathcal{F}$  by the standard equivalence between a tensor and a linear map. That is, given two functions  $f, g \in \mathcal{F}$ , their covariance can be computed by  $\mathbb{E}_{XY}[f(X)g(Y)] = \langle f, \mathcal{C}_{XY}g \rangle_{\mathcal{F}}$ , or equivalently  $\langle f \otimes g, \mathcal{C}_{XY} \rangle_{\mathcal{F} \otimes \mathcal{F}}$ , where in the former we view  $\mathcal{C}_{XY}$  as an operator while in the latter we view it as an element in tensor product space. By analogy,  $\mathcal{C}_{XX} := \mathbb{E}_X[\phi(X) \otimes \phi(X)]$  and  $\mathcal{C}_{(XX)Y} := \mathbb{E}_X[\phi(X) \otimes \phi(X) \otimes \phi(Y)]$  can also be defined,  $\mathcal{C}_{(XX)Y}$  can be regarded as a linear operator from  $\mathcal{F}$  to  $\mathcal{F} \otimes \mathcal{F}$ . It will be clear from the context whether we use  $\mathcal{C}_{XY}$  as an operator between two spaces or as an element from a tensor product feature space.

Kernel embeddings can be readily generalized to joint distribution of  $O$  variables,  $X_1, \dots, X_O$ , using the  $O$ -th order tensor product feature space  $\mathcal{F}^O$  (Here  $\mathcal{O} := \{X_1, \dots, X_O\}$ ). In this feature space, the feature map is defined as  $\otimes_{i=1}^O \phi(x_i) := \phi(x_1) \otimes \phi(x_2) \otimes \dots \otimes \phi(x_O)$ , and the inner product in this space satisfies  $\langle \otimes_{i=1}^O \phi(x_i), \otimes_{i=1}^O \phi(x'_i) \rangle_{\mathcal{F}^O} = \prod_{i=1}^O \langle \phi(x_i), \phi(x'_i) \rangle_{\mathcal{F}} = \prod_{i=1}^O K(x_i, x'_i)$ . Then we can embed a joint density  $p(x_1, \dots, x_O)$  into a tensor product feature space  $\mathcal{F}^O$  by

$$\mathcal{C}_{\mathcal{O}} := \mathbb{E}_{\mathcal{O}} [\otimes_{i=1}^O \phi(X_i)] = \int_{\Omega^O} (\otimes_{i=1}^O \phi(x_i)) dP(x_1, \dots, x_O). \quad (42)$$

Although the definition of embedding in (40) is simple, it turns out to have both rich representational power and a well-behaved empirical estimate. First, the mapping is injective for characteristic kernels (Sriperumbudur et al., 2008, 2010). That is, if two distributions,  $P(X)$  and  $Q(Y)$ , are different, they will be mapped to two distinct points in the feature space. Many commonly used kernels are characteristic, such as the Gaussian RBF kernel  $\exp(-\sigma\|x - x'\|^2)$  and Laplace kernel  $\exp(-\sigma\|x - x'\|)$ , which implies that if we embed distributions using these kernels, the distance of the mappings in feature space will give us an indication on whether these two distributions are identical. This intuition has been exploited to design state-of-the-art two-sample tests (Gretton et al., 2012) and independence tests (Gretton et al., 2008). For the former case, the test statistic is the squared distance between the embeddings of  $P(Y)$  and  $Q(Y)$ , i.e.,  $\text{mmd}(X, Y) := \|\mu_X - \mu_Y\|_{\mathcal{F}}^2$ . For the latter case, the test statistic is the squared distance between the embeddings of a joint distribution  $P(X, Y)$  and the product of its marginals  $P(X)P(Y)$ , i.e.,  $\text{hsic}(X, Y) := \|\mathcal{C}_{XY} - \mu_X \otimes \mu_Y\|_{\mathcal{F} \otimes \mathcal{F}}^2$ . Similarly, this statistic also has advantages over the Kernel density estimation based statistic. We will further discuss these tests in the next section, following our introduction of finite sample estimates of the distribution embeddings and test statistics.

## A.2 Finite Sample Kernel Estimator

While we rarely have access to the true underlying distribution,  $P(X)$ , we can readily estimate its embedding using a finite sample average. Given a sample  $\mathcal{D}_X = \{x_1, \dots, x_n\}$  of size  $n$  drawn i.i.d. from  $P(X)$ , the empirical Hilbert space embedding is

$$\hat{\mu}_X = \frac{1}{n} \sum_{i=1}^n \phi(x_i). \quad (43)$$

This empirical estimate converges to its population counterpart in RKHS norm,  $\|\hat{\mu}_X - \mu_X\|_{\mathcal{F}}$ , with a rate of  $O_p(n^{-\frac{1}{2}})$  (Berlinet and Thomas-Agnan, 2004; Smola et al., 2007b). We note that this rate is independent of the dimension of  $X$ , meaning that statistics based on Hilbert space embeddings circumvent the curse of dimensionality.

Kernel embeddings of joint distributions inherit the previous two properties of general embeddings: injectivity and easy empirical estimation. Given  $m$  pairs of training examples  $\mathcal{D}_{XY} = \{(x_1, y_1), \dots, (x_n, y_n)\}$  drawn i.i.d. from  $P(X, Y)$ , the covariance operator  $\mathcal{C}_{XY}$  can then be estimated as

$$\hat{\mathcal{C}}_{XY} = \frac{1}{n} \sum_{i=1}^n \phi(x_i) \otimes \phi(y_i). \quad (44)$$

By virtue of the kernel trick, most of the computation required for statistical inference using Hilbert space embeddings can be reduced to the Gram matrix manipulation. The entries in the Gram matrix  $K$  correspond to the kernel value between data points  $x_i$  and  $x_j$ , *i.e.*,  $K_{ij} = K(x_i, x_j)$ , and therefore its size is determined by the number of data points in the sample (similarly Gram matrix  $G$  has entries  $G_{ij} = K(y_i, y_j)$ ). The size of the Gram matrices is in general much smaller than the dimension of the feature spaces (which can be infinite). This enables efficient nonparametric methods using the Hilbert space embedding representation. For instance, the empirical `mm`d can be computed using kernel evaluations,

$$\widehat{\text{mm}}d(P, Q) = \left\| \frac{1}{n} \sum_{i=1}^n \phi(x_i) - \frac{1}{n} \sum_{i=1}^n \phi(y_i) \right\|_{\mathcal{F}}^2 = \frac{1}{n^2} \sum_{i,j=1}^n (K(x_i, x_j) + K(y_i, y_j) - 2K(x_i, y_j)).$$

If the sample size is large, the computation in Hilbert space embedding methods may be expensive. In this case, a popular solution is to use a low-rank approximation of the Gram matrix, such as incomplete Cholesky factorization Fine and Scheinberg (2001), which is known to work very effectively in reducing computational cost of kernel methods, while maintaining the approximation accuracy.

## B Kernel Embeddings of Conditional Distributions

In this section, we define the Hilbert space embeddings of conditional distributions, which take into account complex conditional independence relations in graphical models.

### B.1 Population Definition

The Hilbert space embedding of a conditional distribution  $P(Y|X)$  is defined as (Song et al., 2009)

$$\mu_{Y|x} := \mathbb{E}_{Y|x}[\phi(Y)] = \int_{\Omega} \phi(y) dP(y|x). \quad (45)$$

Given this embedding, the conditional expectation of a function  $g \in \mathcal{F}$  can be computed as

$$\mathbb{E}_{Y|x}[g(Y)] = \langle g, \mu_{Y|x} \rangle_{\mathcal{F}}.$$

This may be compared with the property of the mean embedding in the previous section, where the *unconditional* expectation of a function may be written as an inner product with the embedding. Unlike the embeddings discussed in the previous section, an embedding of conditional distribution is not a single element in the RKHS, but will instead sweep out a family of points in the RKHS, each indexed by a fixed value  $x$  of the conditioning variable  $X$ . It is only by fixing  $X$  to a particular

value  $x$ , that we will be able to obtain a single RKHS element,  $\mu_{Y|x} \in \mathcal{F}$ . In other words, we need to define an operator, denoted as  $\mathcal{C}_{Y|X}$ , which can take as input an  $x$  and output an embedding. More specifically, we require it to satisfy

$$\mu_{Y|x} = \mathcal{C}_{Y|X}\phi(x). \quad (46)$$

Based on the relation between conditional expectation and covariance operators, Song et al. (2009) show that that, under the assumption  $\mathbb{E}_{Y| \cdot} [g(Y)] \in \mathcal{F}$ ,

$$\mathcal{C}_{Y|X} := \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}, \quad \text{and hence } \mu_{Y|x} = \mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}\phi(x) \quad (47)$$

satisfy the requirement in (46). We remark that the assumption  $\mathbb{E}_{Y| \cdot} [g(Y)] \in \mathcal{F}$  always holds for finite domains with characteristic kernels, but it is not necessarily true for continuous domains (Fukumizu et al., 2004). In the cases where the assumption does not hold, we will use the expression  $\mathcal{C}_{YX}\mathcal{C}_{XX}^{-1}\phi(x)$  as an approximation of the conditional mean  $\mu_{Y|x}$ . In practice, the inversion of the operator can be replaced by the regularized inverse  $(\mathcal{C}_{XX} + \mathcal{I})^{-1}$ , where  $\mathcal{I}$  is the identity operator.

## B.2 Finite Sample Kernel Estimator

Given a dataset  $\mathcal{D}_{XY} = \{(x_1, y_1), \dots, (x_n, y_n)\}$  of size  $n$  drawn i.i.d. from  $P(X, Y)$ , we will estimate the conditional embedding operator as

$$\hat{\mathcal{C}}_{Y|X} = \Phi(\mathbf{K} + \lambda\mathbf{I})^{-1}\Upsilon^\top \quad (48)$$

where  $\Phi := (\phi(y_1), \dots, \phi(y_n))$  and  $\Upsilon := (\phi(x_1), \dots, \phi(x_n))$  are implicitly formed feature matrix,  $\mathbf{K} = \Upsilon^\top \Upsilon$  is the Gram matrix for samples from variable  $X$ , and  $\mathbf{I}$  is an identity matrix. Furthermore, we need an additional regularization parameter  $\lambda$  to avoid overfitting. Then  $\hat{\mu}_{Y|x} = \hat{\mathcal{C}}_{Y|X}\phi(x)$  becomes a weighted sum of the feature mapped data points  $\phi(y_1), \dots, \phi(y_n)$ ,

$$\begin{aligned} \hat{\mu}_{Y|x} &= \sum_{i=1}^m \beta_i(x) \phi(y_i) = \Phi \beta(x) \quad \text{where} \\ \beta(x) &= (\beta_1(x), \dots, \beta_n(x))^\top = (\mathbf{K} + \lambda\mathbf{I})^{-1} \mathbf{K}_{:x}, \end{aligned} \quad (49)$$

and  $\mathbf{K}_{:x} = (K(x, X_1), \dots, K(x, X_n))^\top$ . The empirical estimator of the conditional embedding is similar to the estimator of the ordinary embedding from equation (43). The difference is that, instead of applying uniform weights  $\frac{1}{n}$ , the former applies *non-uniform* weights,  $\beta_i(x)$ , on observations which are, in turn, determined by the value  $x$  of the conditioning variable. These non-uniform weights reflect the effects of conditioning on the embeddings. It is also shown that this empirical estimate converges to its population counterpart in RKHS norm,  $\|\hat{\mu}_{Y|x} - \mu_{Y|x}\|_{\mathcal{F}}$ , with rate of  $O_p(n^{-\frac{1}{4}})$  if one decreases the regularization  $\lambda$  with rate  $O(n^{-\frac{1}{2}})$ . With appropriate assumptions on the joint distribution of  $X$  and  $Y$ , better rates can be obtained (Grunewald et al., 2012).

## C Hilbert Space Embeddings as Infinite Dimensional Higher Order Tensors

The above Hilbert space embedding  $\mathcal{C}_{\mathcal{O}}$  can also be viewed as a multi-linear operator (tensor) of order  $O$  mapping from  $\mathcal{F}^O$  to  $\mathbb{R}$ . (More detailed and generic introduction to tensor and tensor

notation can be found in Kolda and Bader (2009)). The operator is linear in each argument (mode) when fixing other arguments. Furthermore, the application of the operator to a set of elements  $\{f_i \in \mathcal{F}\}_{i=1}^O$  can be defined using the inner product from the tensor product feature space, *i.e.*,

$$\mathcal{C}_\theta \bullet_1 f_1 \bullet_2 \dots \bullet_O f_O := \langle \mathcal{C}_\theta, \otimes_{i=1}^O f_O \rangle_{\mathcal{FO}} = \mathbb{E}_\theta \left[ \prod_{i=1}^O \langle \phi(X_i), f_i \rangle_{\mathcal{F}} \right], \quad (50)$$

where  $\bullet_i$  means applying  $f_i$  to the  $i$ -th argument of  $\mathcal{C}_\theta$ . Furthermore, we can define Hilbert-Schmidt norm  $\|\cdot\|_{HS}$  of  $\mathcal{C}_\theta$  as  $\|\mathcal{C}_\theta\|_{HS}^2 = \sum_{i_1=1}^\infty \dots \sum_{i_O=1}^\infty (\mathcal{C}_\theta \bullet_1 e_{i_1} \bullet_2 \dots \bullet_O e_{i_O})^2$  using an orthonormal basis  $\{e_i\}_{i=1}^\infty \subset \mathcal{F}$ <sup>2</sup>. The Hilbert-Schmidt norm can be viewed as a generalization of the Frobenius norm from matrices to general operators. For the space of operators with finite Hilbert-Schmidt norms, we can also define the Hilbert-Schmidt inner product of such operators as

$$\langle \mathcal{C}_\theta, \tilde{\mathcal{C}}_\theta \rangle_{HS} = \sum_{i_1=1}^\infty \dots \sum_{i_O=1}^\infty (\mathcal{C}_\theta \bullet_1 e_{i_1} \bullet_2 \dots \bullet_O e_{i_O}) (\tilde{\mathcal{C}}_\theta \bullet_1 e_{i_1} \bullet_2 \dots \bullet_O e_{i_O}). \quad (51)$$

When  $\mathcal{C}_\theta$  has the form of  $\mathbb{E}_\theta [\otimes_{i=1}^O \phi(X_i)]$ , the above inner product reduces to  $\mathbb{E}_\theta [\tilde{\mathcal{C}}_\theta \bullet_1 \phi(X_1) \bullet_2 \dots \bullet_O \phi(X_O)]$ .

In this paper, the ordering of the tensor modes is not essential. Therefore we simply label them using the corresponding random variables. We can reshape a higher order tensor into a lower order one by partitioning its modes into several disjoint groups. For instance, let  $\mathcal{J}_1 = \{X_1, \dots, X_s\}$  be the set of modes corresponding to the first  $s$  variables and  $\mathcal{J}_2 = \{X_{s+1}, \dots, X_O\}$ . Similarly to the Matlab notation, we can obtain a 2nd order tensor by

$$\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2} = \text{reshape}(\mathcal{C}_\theta, \mathcal{J}_1, \mathcal{J}_2) : \mathcal{F}^s \times \mathcal{F}^{O-s} \mapsto \mathbb{R}. \quad (52)$$

To under the reshape operator, if we have a set of functions  $\{f_i\}_{i=1}^O$ , then

$$\mathcal{C}_\theta \bullet_1 f_1 \bullet_2 f_2 \bullet_3 \dots \bullet_O f_O = \mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2} \bullet_1 (f_1 \otimes f_2 \otimes \dots \otimes f_s) \bullet_2 (f_{s+1} \otimes f_{s+2} \otimes \dots \otimes f_O).$$

Note that given an orthonormal basis  $\{e_i\}_{i=1}^\infty \in \mathcal{F}$ , we can readily obtain an orthonormal basis for  $\mathcal{F}^s$  as  $\{e_{i_1} \otimes \dots \otimes e_{i_s}\}_{i_1, \dots, i_s=1}^\infty$ , and for  $\mathcal{F}^{O-s}$  as  $\{e_{i_1} \otimes \dots \otimes e_{i_{O-s}}\}_{i_1, \dots, i_{O-s}=1}^\infty$ . Hence we can define the Hilbert-Schmidt norm for  $\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2}$ . Using (50), it is easy to see that  $\|\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2}\|_{HS} = \|\mathcal{C}_\theta\|_{HS}$ .

In the reverse direction, we can also reshape a lower order tensor into a higher order one by further partitioning certain mode of the tensor. For instance, we can partition  $\mathcal{J}_1$  into  $\mathcal{J}'_1 = \{X_1, \dots, X_t\}$  and  $\mathcal{J}''_1 = \{X_{t+1}, \dots, X_s\}$ , and turn  $\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2}$  into a 3rd order tensor by

$$\mathcal{C}_{\mathcal{J}'_1; \mathcal{J}''_1; \mathcal{J}_2} = \text{reshape}(\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2}, \mathcal{J}'_1, \mathcal{J}''_1, \mathcal{J}_2) : \mathcal{F}^t \times \mathcal{F}^{s-t} \times \mathcal{F}^{O-s} \mapsto \mathbb{R}, \quad (53)$$

meaning that if we have a set of functions  $\{f_i\}_{i=1}^O$ , then

$$\begin{aligned} & \mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2} \bullet_1 (f_1 \otimes f_2 \otimes \dots \otimes f_s) \bullet_2 (f_{s+1} \otimes f_{s+2} \otimes \dots \otimes f_O) \\ &= \mathcal{C}_{\mathcal{J}'_1; \mathcal{J}''_1; \mathcal{J}_2} \bullet_1 (f_1 \otimes f_2 \otimes \dots \otimes f_t) \bullet_2 (f_{t+1} \otimes f_2 \otimes \dots \otimes f_s) \bullet_3 (f_{s+1} \otimes f_{s+2} \otimes \dots \otimes f_O), \end{aligned}$$

and furthermore, we have  $\|\mathcal{C}_{\mathcal{J}'_1; \mathcal{J}''_1; \mathcal{J}_2}\|_{HS} = \|\mathcal{C}_\theta\|_{HS}$ .

---

<sup>2</sup>The definition of Hilbert-Schmidt norm is invariant to arbitrary orthonormal basis.

The 2nd order tensor  $\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2}$  are the same as the cross-covariance operator between two sets of variables in  $\mathcal{J}_1$  and  $\mathcal{J}_2$ . In this case, we adopt the same notation and operations as for matrices. For instance, we can perform singular value decomposition of  $\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2} = \sum_{i=1}^{\infty} s_i (u_i \otimes v_i)$  where  $s_i \in \mathbb{R}$  are ordered in non-increasing manner,  $\{u_i\}_{i=1}^{\infty} \subset \mathcal{F}^s$  and  $\{v_i\}_{i=1}^{\infty} \subset \mathcal{F}^{d-s}$  are left and right singular vectors. The rank of  $\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2}$  is the smallest  $r$  such that  $s_i = 0$  for  $i \geq r$ . In this case, we will also define  $\mathcal{U}_r = (u_1, u_2, \dots, u_r)$ ,  $\mathcal{V}_r = (v_1, v_2, \dots, v_r)$  and  $\mathcal{S}_r = \text{diag}(s_1, s_2, \dots, s_r)$ , and denote the low rank representation as  $\mathcal{C}_{\mathcal{J}_1; \mathcal{J}_2} = \mathcal{U}_r \mathcal{S}_r \mathcal{V}_r^\top$ . We denote this to be "thin" SVD. Finally, reshape( $\mathcal{C}_\emptyset, \{\emptyset\}, \emptyset$ ), a 1st order tensor, is simply a vector where we will use vector notation.

The Hilbert space embedding framework represents the building blocks from probabilistic graphical models, such as marginal distributions over single variables, joint distributions over variable pairs, triplets and more, as infinite-dimensional vectors, matrices, tensors and high-order tensors respectively; furthermore, the operations fundamental to probabilistic reasoning and graphical models, *i.e.*, conditioning, Sum Rule, Product Rule and Bayes' Rule, become linear transformations and relations between the embeddings (see Figure 7 for the analogy between discrete probability tables and kernel embeddings of distributions). We may combine these building blocks so as to reason about interactions between a large collection of variables, even in the absence of parametric models.

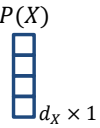
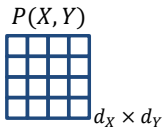
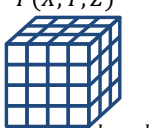
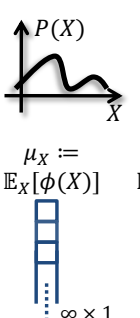
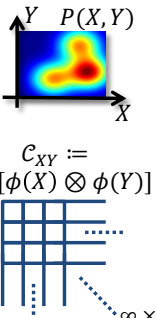
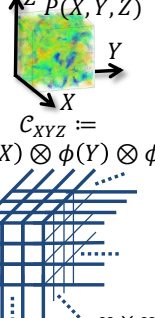
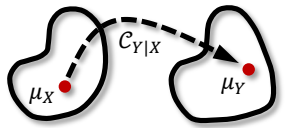
	Distributions			Probabilistic Operations
Discrete	 $d_X \times 1$	 $d_X \times d_Y$	 $d_X \times d_Y \times d_Z$	<p>Sum Rule: <math>Q(X) = \sum_Y P(X Y)\pi(Y)</math></p> <p>Product Rule: <math>Q(X,Y) = P(X Y)\pi(Y)</math></p> <p>Bayes Rule: <math>Q(Y x) = \frac{P(x Y)\pi(Y)}{Q(X)}</math></p>
Kernel Embedding	 $\mu_X := \mathbb{E}_X[\phi(X)]$ $\infty \times 1$	 $C_{XY} := \mathbb{E}_{XY}[\phi(X) \otimes \phi(Y)]$ $\infty \times \infty$	 $C_{XYZ} := \mathbb{E}_{XYZ}[\phi(X) \otimes \phi(Y) \otimes \phi(Z)]$ $\infty \times \infty \times \infty$	 <p>Sum Rule: <math>\mu_X^\pi = C_{X Y} \mu_Y^\pi</math></p> <p>Product Rule: <math>C_{XY}^\pi = C_{X Y} C_{YY}^\pi</math></p> <p>Bayes Rule: <math>\mu_{Y x}^\pi = C_{Y x}^\pi \phi(x)</math></p>

Figure 7: Analogy between discrete and kernel embedding representations of marginal distributions and joint distributions of variable pairs and triplets. Probabilistic operations, such as conditioning, Sum Rule, Product Rule and Bayes' Rule become linear operations on the embedding representations. The discrete case is a specific instance of our embedding framework, given an appropriate choice of kernel.

## References

ANANDKUMAR, A., CHAUDHURI, K., HSU, D., KAKADE, S. M., SONG, L. and ZHANG, T. (2011). Spectral methods for learning multivariate latent tree structure. *arXiv preprint arXiv:1107.1283*



- ANANDKUMAR, A. and VALLUVAN, R. (2013). Learning loopy graphical models with latent variables: Efficient methods and guarantees. *The Annals of Statistics* **41** 401–435.
- ASUNCION, A. and NEWMAN, D. (2007). Uci machine learning repository.
- BERLINET, A. and THOMAS-AGNAN, C. (2004). *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer.
- BICKSON, D. (2008). Gaussian belief propagation: Theory and application. *Arxiv preprint arXiv:0811.2518* .
- BLEI, D., NG, A. and JORDAN, M. (2002). Latent dirichlet allocation. In *Advances in Neural Information Processing Systems 14* (T. G. Dietterich, S. Becker and Z. Ghahramani, eds.). MIT Press, Cambridge, MA.
- CHOI, M., TAN, V., ANANDKUMAR, A. and WILLSKY, A. (2010). Learning latent tree graphical models. *Arxiv preprint arXiv:1009.2722* .
- CHOW, C. and LIU, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory* **14** 462–467.
- CLARK, A. (1990). Inference of haplotypes from PCR-amplified samples of diploid populations. *Molecular Biology and Evolution* **7** 111–122.
- DASKALAKIS, C., MOSSEL, E. and ROCH, S. (2006). Optimal phylogenetic reconstruction. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. ACM.
- DASKALAKIS, C., MOSSEL, E. and ROCH, S. (2011). Phylogenies without branch bounds: Contracting the short, pruning the deep. *SIAM Journal on Discrete Mathematics* **25** 872–893.
- DEMPSTER, A. P., LAIRD, N. M. and RUBIN, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B* **39** 1–22.
- ERDÖS, P. L., SZÉKELY, L. A., STEEL, M. A. and WARNOW., T. J. (1999a). A few logs suffice to build (almost) all trees (I). *Random Structures and Algorithms* **14** 153–184.
- ERDÖS, P. L., SZÉKELY, L. A., STEEL, M. A. and WARNOW., T. J. (1999b). A few logs suffice to build (almost) all trees: Part II. *Theoretical Computer Science* **221** 77–118.
- FINE, S. and SCHEINBERG, K. (2001). Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research* **2** 243–264.
- FUKUMIZU, K., BACH, F. R. and JORDAN, M. I. (2004). Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research* **5** 73–99.
- GRETTON, A., BORGWARDT, K., RASCH, M., SCHOELKOPF, B. and SMOLA, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research* **13** 723–773.
- GRETTON, A., FUKUMIZU, K., TEO, C.-H., SONG, L., SCHÖLKOPF, B. and SMOLA, A. J. (2008). A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20*. MIT Press, Cambridge, MA.

- GRONAU, I., MORAN, S. and SNIR, S. (2008). Fast and reliable reconstruction of phylogenetic trees with very short edges. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics.
- GRUNEWALDER, S., LEVER, G., BALDASSARRE, L., PATTERSON, S., GRETTON, A. and PONTIL, M. (2012). Conditional mean embeddings as regressors. In *Proceedings of the International Conference on Machine Learning*.
- HARMELING, S. and WILLIAMS, C. (2010). Greedy learning of binary latent trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence* .
- HELLER, K. and GHAHRAMANI, Z. (2005). Bayesian hierarchical clustering. In *Proceedings of the 22nd international conference on Machine learning*. ACM.
- HOFF, P. D., RAFTERY, A. E. and HANDCOCK, M. S. (2002). Latent space approaches to social network analysis. *Journal of the American Statistical Association* **97** 1090–1098.
- IHLER, A. and MCALLESTER, D. (2009). Particle belief propagation. In *Proc. Intl. Conference on Artificial Intelligence and Statistics*.
- KOLDA, T. G. and BADER, B. W. (2009). Tensor decompositions and applications. *SIAM Review* **51** 455–500.
- LIU, H., LAFFERTY, J. and WASSERMAN, L. (2009). The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *The Journal of Machine Learning Research* **10** 2295–2328.
- MINKA, T. (2001). Expectation Propagation for Approximate Bayesian Inference. In *UAI*.
- MOSSEL, E. (2007). Distorted metrics on trees and phylogenetic forests. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* **4** 108–116.
- MOSSEL, E. and ROCH, S. (2006). Learning nonsingular phylogenies and hidden markov models. *Annals of Applied Probability* **16** 583–614.
- MOSSEL, E., ROCH, S. and SLY, A. (2011a). On the inference of large phylogenies with long branches: How long is too long? *Bulletin of mathematical biology* **73** 1627–1644.
- MOSSEL, E., ROCH, S. and SLY, A. (2011b). Robust estimation of latent tree graphical models: Inferring hidden states with inexact parameters. *IEEE Transactions on Information Theory* .
- PARIKH, A., SONG, L. and XING, E. P. (2011). A spectral algorithm for latent tree graphical models. In *Proceedings of the International Conference on Machine Learning*.
- PARZEN, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics* **33** 1065–1076.
- PEARL, J. (2001). *Causality: Models, Reasoning and Inference*. Cambridge University Press.
- POON, L., ZHANG, N. L., CHEN, T. and WANG, Y. (2010). Variable selection in model-based clustering: To do or to facilitate. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*.

- RABINER, L. R. and JUANG, B. H. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine* **3** 4–16.
- REDMOND, M. and BAVEJA, A. (2002). A data-driven software tool for enabling cooperative information sharing among police departments. *European Journal of Operational Research* **141** 660–678.
- ROCH, S. (2010). Toward extracting all phylogenetic information from matrices of evolutionary distances. *Science* **327** 1376–1379.
- ROSENBLATT, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics* 832–837.
- SAITOU, N., NEI, M. ET AL. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Mol Biol Evol* **4** 406–425.
- SCHÖLKOPF, B., TSUDA, K. and VERT, J.-P. (2004). *Kernel Methods in Computational Biology*. MIT Press, Cambridge, MA.
- SEMPLE, C. and STEEL, M. (2003). *Phylogenetics*, vol. 24. Oxford University Press, USA.
- SHAWE-TAYLOR, J. and CRISTIANINI, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, UK.
- SILVERMAN, B. W. (1986). *Density Estimation for Statistical and Data Analysis*. Monographs on statistics and applied probability, Chapman and Hall, London.
- SMOLA, A., GRETTON, A., SONG, L. and SCHÖLKOPF, B. (2007a). A hilbert space embedding for distributions. In *Algorithmic Learning Theory* (E. Takimoto, ed.). Lecture Notes on Computer Science, Springer.
- SMOLA, A. J., GRETTON, A., SONG, L. and SCHÖLKOPF, B. (2007b). A Hilbert space embedding for distributions. In *Proceedings of the International Conference on Algorithmic Learning Theory*, vol. 4754. Springer.
- SONG, L., GRETTON, A. and GUESTIN, C. (2010). Nonparametric tree graphical models. In *Proc. Intl. Conference on Artificial Intelligence and Statistics*.
- SONG, L., HUANG, J., SMOLA, A. J. and FUKUMIZU, K. (2009). Hilbert space embedding of conditional distributions. In *Proceedings of the International Conference on Machine Learning*.
- SRIPERUMBUDUR, B., GRETTON, A., FUKUMIZU, K., LANCKRIET, G. and SCHÖLKOPF, B. (2008). Injective Hilbert space embeddings of probability measures. In *Proc. Annual Conf. Computational Learning Theory*.
- SRIPERUMBUDUR, B., GRETTON, A., FUKUMIZU, K., LANCKRIET, G. and SCHÖLKOPF, B. (2010). Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research* **11** 1517–1561.
- SUDDERTH, E., IHLER, A., FREEMAN, W. and WILLSKY, A. (2003). Nonparametric belief propagation. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*.

- WASSERMAN, L. (2006). *All of Nonparametric Statistics*. Springer.
- WEISS, Y. and FREEMAN, W. T. (2001). Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation* **13** 2173–2200.
- ZHANG, N. (2004). Hierarchical latent class models for cluster analysis. *The Journal of Machine Learning Research* **5** 697–723.