Learning Mixtures of Truncated Basis Functions from Data

Langseth, Helge; Nielsen, Thomas Dyhre; Pérez-Bernabé, Inmaculada; Salmerón, Antonio

Published in:
International Journal of Approximate Reasoning

DOI (link to publication from Publisher):
10.1016/j.ijar.2013.09.012

Publication date:
2014

Document Version
Early version, also known as pre-print

Learning Mixtures of Truncated Basis Functions from Data

Helge Langseth[a], Thomas D. Nielsen[b], Inmaculada Pérez-Bernabé[c], Antonio Salmerón[c]

[a] *Department of Computer and Information Science, The Norwegian University of Science and Technology, Trondheim (Norway)*
[b] *Department of Computer Science, Aalborg University, Aalborg (Denmark)*
[c] *Department of Mathematics, University of Almería, Almería (Spain)*

Abstract

In this paper we investigate methods for learning hybrid Bayesian networks from data. First we utilize a kernel density estimate of the data in order to translate the data into a *mixture of truncated basis functions* (MoTBF) representation using a convex optimization technique. When utilizing a kernel density representation of the data, the estimation method relies on the specification of a kernel bandwidth. We show that in most cases the method is robust wrt. the choice of bandwidth, but for certain data sets the bandwidth has a strong impact on the result. Based on this observation, we propose an alternative learning method that relies on the cumulative distribution function of the data.

Empirical results demonstrate the usefulness of the approaches: Even though the methods produce estimators that are slightly poorer than the state of the art (in terms of log-likelihood), they are significantly faster, and therefore indicate that the MoTBF framework can be used for inference and learning in reasonably sized domains. Furthermore, we show how a particular subclass of MoTBF potentials (learnable by the proposed methods) can be exploited to significantly reduce complexity during inference.

1. Introduction

In domains involving both discrete and continuous variables, Bayesian networks with mixtures of truncated exponentials (MTEs) (Moral et al., 2001) and mixtures of truncated polynomials (MoPs) (Shenoy and West, 2011; Shenoy, 2012) have received increasing interest over the last few years. A recent addition to the fold is the *mixtures of truncated basis functions* (MoTBFs) framework (Langseth et al., 2012), which offers a unified theory for MTEs and MoPs. The MoTBF framework allows discrete and continuous variables to co-exist in a Bayesian network without any structural constraints, and since the family of MoTBFs is closed under addition, multiplication, and integration, inference in an MoTBF network can be performed efficiently using the Shafer-Shenoy architecture (Shafer and Shenoy, 1990; Cobb and Shenoy, 2006).

The problem of learning MoTBF models from data has been only scarcely considered (Romero et al., 2006; Langseth et al., 2009, 2010; López-Cruz et al., 2012). Romero et al. (2006) used a kernel estimator to represent the data distribution, and thereafter fitted an MTE to the kernel using regression. Langseth et al. (2009, 2010) attempted to find maximum likelihood parameters directly but since the maximum likelihood equations have no analytic solutions in general, they instead proposed an iterative scheme utilizing Newton's method and Lagrange multipliers. This resulted in better estimators (in terms of likelihood on the training-set as well as on a hold-out test-set), but at the cost of a steep increase in computational complexity. Finally, López-Cruz et al. (2012) used B-splines for learning MoPs from data. Their focus was primarily on univariate distributions and on continuous variables with only discrete parents.

The main idea of the learning scheme that we pursue in this paper is to use a translation procedure (Langseth et al., 2012) that can, in principle, be used to approximate any distribution by an MoTBF. Our first approach is to represent the data by a kernel density estimate, and then translate this representation into an MoTBF. Motivated by preliminary empirical findings we,

secondly, consider the related idea of approximating the cumulative density function by a function that has an MoTBF potential as its derivative. The two approaches are compared empirically to those of Langseth et al. (2009, 2010), and we find that although the new methods find parameters that are slightly worse (in terms of likelihood on a test-set), they are more than an order of magnitude faster than previous techniques. Finally, we consider a sub-class of MoTBF potentials that support a computationally efficient inference scheme, and explore the use of these potentials in a learning context.

The rest of the paper is organized as follows: We start with an introduction to the MoTBF framework in Section 2. The simplified problem of learning univariate MoTBFs from data is considered in Section 3, and we discuss learning conditional distributions in Section 4. We report on some experiments in Section 5, and finally we conclude in Section 6.

2. The MoTBF model

2.1. Definition of the framework

The MoTBF framework is based on the abstract notion of real-valued *basis functions* $\psi(\cdot)$, which includes both polynomial and exponential functions as special cases. The first building-block of the framework is the marginal distribution: Let X be a continuous variable[1] with (truncated) domain $\Omega_X \subseteq \mathbb{R}$ and let $\psi_i : \mathbb{R} \to \mathbb{R}$, for $i = 0, \ldots, k$, define a collection of real basis functions. We say that a function $g_k : \Omega_X \mapsto \mathbb{R}_0^+$ is an MoTBF potential of level k wrt. $\Psi = \{\psi_0, \psi_1, \ldots, \psi_k\}$ if g_k can be written as

$$g_k(x) = \sum_{i=0}^{k} a_i \, \psi_i(x), \tag{1}$$

where a_i are real numbers. The potential is a density if $\int_{\Omega_X} g_k(x)\, dx = 1$. Note that as opposed to the MTE and MoP definitions (Moral et al., 2001; Shenoy and West, 2011), a marginal MoTBF potential does not rely on a partitioning of Ω_X to improve its expressive power.

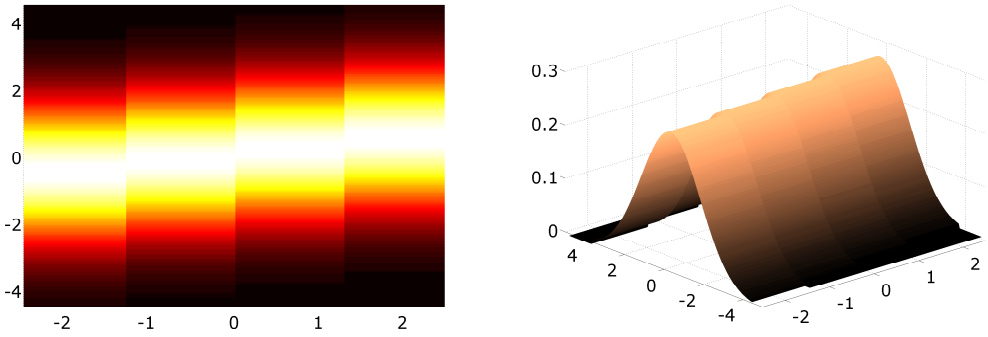
Example 1. *By letting the basis functions correspond to polynomial functions, $\psi_i(x) = x^i$ for $i = 0, 1, \ldots$, the MoTBF model reduces to an MoP model for univariate distributions. Similarly, if we define the basis functions as $\psi_i(x) = \{1, \exp(-x), \exp(x), \exp(-2x), \exp(2x), \ldots\}$, the MoTBF model corresponds to an MTE model with the exception that the parameters in the exponential functions are fixed. Notice, however, that in both situations the model does not rely on a partitioning of the interval over which the distribution is defined (as opposed to the standard definitions of MoPs and MTEs).*

Next, we turn to the MoTBF definition of conditional distributions, which mirrors the corresponding definition for MTEs. Thus, the influence a set of continuous parent variables $\boldsymbol{Z}$ has on their child variable X is encoded only through the partitioning of the domain of $\boldsymbol{Z}$, $\Omega_{\boldsymbol{Z}}$, into hyper-cubes, and not directly in the functional form of $g_k(x|\boldsymbol{z})$ inside the hyper-cubes $\Omega_{\boldsymbol{Z}}^{\ell}$. More precisely, for a partitioning $\mathcal{P} = \{\Omega_{\boldsymbol{Z}}^1, \ldots, \Omega_{\boldsymbol{Z}}^m\}$ of $\Omega_{\boldsymbol{Z}}$, the conditional MoTBF is defined for $\boldsymbol{z} \in \Omega_{\boldsymbol{Z}}^j$, $1 \leq j \leq m$, as

$$g_k^{(j)}(x|\boldsymbol{z} \in \Omega_{\boldsymbol{Z}}^j) = \sum_{i=0}^{k} a_i^{(j)} \, \psi_i^{(j)}(x). \tag{2}$$

Example 2. *Figure 1 shows an MoTBF representation of the conditional linear Gaussian distribution $Y|\{X = x\} \sim N(0.3 \cdot x, 1.5)$. The representation was found using the translation procedure by Langseth et al. (2012), which is further described below.*

[1] In this paper we will often refer to MoTBF potentials defined over continuous variables only. Unless the contrary is specified, all claims about these potentials are extensible to potentials also containing discrete variables in their domains; the claims simply apply to each configuration of the discrete variables.



(a) 2D-view (b) 3D-view

Figure 1: An MoTBF representation of the conditional linear Gaussian distribution $Y|\{X = x\} \sim N(0.3 \cdot x, 1.5)$. Part (a) shows X on the horizontal axis and Y on the vertical axis; the lighter the colors, the higher the value of the density. Part (b) shows the same structure in three dimensions.

Finally, the joint MoTBF distribution over $\boldsymbol{x} = (x_1, \ldots, x_n)$ is found using the usual factorization, $g_{\boldsymbol{k}}(\boldsymbol{x}) = \prod_{i=1}^{n} g_{k_i}(x_i|\text{pa}(x_i))$, where the marginal and conditional distributions are defined using Equations (1) and (2), respectively.

Langseth et al. (2012) describe a "translation" procedure for efficiently finding an MoTBF approximation of any density function f. The approximation procedure assumes that the basis functions Ψ are both *legal* and *orthonormal*: Let $\mathcal{Q}$ be the set of all linear combinations of the members of a set of basis functions $\Psi = \{\psi_i(\cdot)\}_{i=0}^{\infty}$. Then Ψ is said to be a *legal* set of basis functions if the following conditions hold:

- ψ_0 is constant in its argument.
- If $\phi_i \in \mathcal{Q}$ and $\phi_j \in \mathcal{Q}$, then $(\phi_i \cdot \phi_j) \in \mathcal{Q}$.
- For any pair of real numbers s and t, there exists a function $\phi \in \mathcal{Q}$ such that $\phi(s) \neq \phi(t)$.

Working with a legal set of basis functions basically ensures that any target density function can be approximated arbitrarily well (see Langseth et al. (2012) for details). It is straight-forward to prove that both the polynomial and exponential basis functions defined in Example 1 are legal sets of basis functions.[2]

When defining *orthonormal* basis functions, we focus on the space $L^2[a, b]$ of quadratically integrable real functions over the finite interval $\Omega = [a, b]$. For two functions ϕ_i and ϕ_j on Ω we define their inner product as

$$\langle \phi_i, \phi_j \rangle = \int_{\Omega} \phi_i(x)\phi_j(x)dx,$$

and say that two functions are orthonormal if and only if $\langle \phi_i, \phi_j \rangle = \delta_{ij}$, where δ_{ij} is the Kronecker delta. A set of non-orthonormal basis functions can easily be orthonormalized using, for instance, the Gram-Schmidt procedure (see, e.g., Fraleigh and Beauregard (1994)).

To set the scene for the translation procedure, we let $f(x)$ be the (target) density, and let $g_k(x) = \sum_{i=0}^{k} a_i \cdot \psi_i(x)$ be an MoTBF of order k. The key idea of Langseth et al. (2012) is to see the MoTBF expansion as a *generalized Fourier series* (Pennell, 1930) in order to find "optimal" values for $a_0, \ldots, a_k$; a generalized Fourier series (GFS) is a functional approximation defined as a sum of basis functions. The GFS approach borrows ideas from approximations in vector-spaces, where the best approximation in a subspace is found by projecting the target vector onto that subspace. Since the basis functions are orthonormal, the GFS solution can be directly applied

[2] In the remainder of the paper we will use $\Psi = \{\psi_i(\cdot)\}_{i=0}^{\infty}$ to denote a legal set of basis functions.

for functional approximations, which corresponds to choosing $a_i = \langle f, \psi_i \rangle$. It can easily be shown that while the GFS approximation up to degree k is guaranteed to minimize the L^2 distance on Ω,

$$\text{error}(f, g_k) = \sqrt{\int_{x \in \Omega} \left(f(x) - g_k(x)\right)^2 dx},$$

g_k may not always be non-negative, and it is thus not a density approximation. A convex optimization scheme (initialized with the GFS coefficients) was therefore employed to obtain parameters guaranteeing that the resulting $g_k(x)$ is a density.

To motivate the optimization procedure, we consider the KL divergence between the target density f and an MoTBF g_k (Kullback and Leibler, 1951):

$$\text{KL}\left(\, f \,\|\, g_k \,\right) = \int_{\Omega} f(x) \log \left(\frac{f(x)}{g_k(x)} \right) dx.$$

It can be shown (Zeevi and Meir, 1997, Lemma 3.3) that the L^2 distance on Ω defines an upper bound for the KL divergence:

$$\text{KL}\left(\, f \,\|\, g_k \,\right) \leq \frac{\left(\text{error}(f, g_k)\right)^2}{\min_{x \in \Omega} g_k}. \tag{3}$$

The optimization procedure of Langseth et al. (2012) minimizes this upper bound of the KL divergence. This results in the following specification, which determines the parameters $(\xi, \boldsymbol{\alpha})$:

$$\begin{aligned} \textbf{Minimize } & \frac{1}{\xi} \cdot \sum_{i=0}^{k} \left(\langle f, \psi_i \rangle - \alpha_i\right)^2 \\ \textbf{Subject to } & \sum_{i=0}^{k} \alpha_i \psi_i(x) \geq \xi, x \in \Omega \\ & \alpha_0 = \langle \psi_0, f \rangle \end{aligned}$$

The first constraint ensures that the function is non-negative on Ω. The second constraint forces the constant term of the MoTBF to be equal to the constant term for the GFS, which, in turn, ensures that $\int_{x \in \Omega} g_k(x)\, dx = \int_{x \in \Omega} f(x)\, dx$ (see Langseth et al. (2012)). Observe that this is a convex optimization problem that can be solved by semi-definite programming (Vandenberghe and Boyd, 1996)[3] and that the approximation can be made arbitrarily tight simply by increasing k. The solution to the program defines the MoTBF approximation $g_k(x|\boldsymbol{\alpha}) = \sum_{i=0}^{k} \alpha_i\, \psi_i(x)$ that minimizes the upper-bound of the KL divergence (Equation (3)).

To conclude this part, Langseth et al. (2012) gives a procedure to "translate" a function f defined on a finite interval $\Omega = [a, b]$ into an MoTBF with basis functions Ψ. For the translation procedure to work, they assume that $f \in L^2[a, b]$, i.e., that $\int_{x \in \Omega} \left(f(x)\right)^2 dx < \infty$. Further, Ψ is constrained to be legal to guarantee that the approximation can be made arbitrarily tight. Finally, to be able to use the GFS formulation, it is assumed, without loss of generality, that Ψ is orthonormal. Our initial idea for learning MoTBF distributions is therefore to find a functional representation for the training data, say h, with $h \in L^2[a, b]$, and then to approximate h with an MoTBF using the above translation procedure. It is expected that if $h \in L^2[a, b]$ closely fits the data, we can obtain an MoTBFs with a correspondingly good fit.

[3] We note that our implementation builds on MATLAB's `Optimization Toolbox`, where the positivity constraint for each $x \in \Omega$ is approximated by a finite number of verification points.

2.2. Inference with MoTBF-potentials

The MoTBF specifications of marginal and conditional distributions not only allow an accurate translation procedure, but they also support an efficient inference scheme. For illustration, consider an MoTBF network with two variables X and Y, where X is the parent of Y. We define the following density functions:

$$\begin{aligned} f(x) &= \sum_{i=0}^{k} a_i \psi_i(x); \\ f(y|x) &= f_\ell(y|x \in \Omega_\ell^X), \quad 1 \leq \ell \leq m; \\ f_\ell(y|x \in \Omega_\ell^X) &= \sum_{i=0}^{k} b_i^{(\ell)} \psi_i^{(\ell)}(y), \end{aligned}$$

where m denotes the number of intervals for X; for notational simplicity we assume the same number k of basis functions for all the potentials. We proceed as follows to calculate the marginal distribution for Y:

$$\begin{aligned} f(y) &= \sum_{\ell=1}^{m} \int_{x \in \Omega_\ell^X} f(x) f_\ell(y|x \in \Omega_\ell^X)\, dx \\ &= \sum_{\ell=1}^{m} f_\ell(y|x \in \Omega_\ell^X) \int_{x \in \Omega_\ell^X} f(x)\, dx \\ &= \sum_{\ell=1}^{m} f_\ell(y|x \in \Omega_\ell^X) \cdot P(X \in \Omega_\ell^X) \\ &= \sum_{\ell=1}^{m} P(X \in \Omega_\ell^X) \sum_{i=0}^{k} b_i^{(\ell)} \psi_i^{(\ell)}(y) \\ &= \sum_{i=0}^{k} \sum_{\ell=1}^{m} P(X \in \Omega_\ell^X)\, b_i^{(\ell)} \psi_i^{(\ell)}(y). \end{aligned}$$

If the basis functions are defined over the same domain for all partitions of X, then we can drop the index ℓ in the basis functions, and we therefore get

$$f(y) = \sum_{i=0}^{k} \left(\sum_{\ell=1}^{m} P(X \in \Omega_\ell^X) \cdot b_i^{(\ell)} \right) \psi_i(y).$$

Intuitively, the resulting coefficients are the weighted averages of the coefficients for the different intervals. From a complexity point of view, the marginalization operation above produces a new potential over a single interval and with no increase in the number of basis functions or exponential terms. This should be considered in contrast to the inference procedure described by, e.g., Moral et al. (2001) and Rumí and Salmerón (2007), where the size of the potentials resulting from marginalizations generally grows exponentially in the number of variables being marginalized out.

A key assumption to achieve this reduction in complexity is that the basis functions used in the specification of Y are defined over the same domain for all values of the parent variable X. The implication of this assumption is that the potentials will generally be specified over larger domains and will typically require more basis functions. We will return to this problem in Section 5.

2.3. MoTBFs vs. mixtures of Gaussians

Mixtures of Gaussians (MoGs) have been previously used as a base model for representing hybrid Bayesian networks (Shenoy, 2006), relying on the fact that mixtures of Gaussians can

approximate any density function arbitrarily well (Titterington et al., 1985). However, MoGs have a series of limitations that motivate the use of MoTBFs instead.

The first limitation is that MoGs are only compatible with networks in which discrete nodes do not have continuous parents (Lauritzen and Jensen, 2001). It is possible, however, to transform any hybrid Bayesian network into a form that meets this restriction through an arc reversal process (Shenoy, 2006). Unfortunately, the price to pay is a significant increase in complexity compared to the initial network as well as a loss of accuracy, as the conditional densities involved in the arc reversals cannot be obtained in closed form and are instead approximated by MoGs.

Secondly, even though MoGs can approximate any density, reaching a satisfactory accuracy level may require a high number of components in the mixture. An extreme case is the approximation of a uniform density. It is naturally captured by a very compact MoTBF model (using only the constant term), whilst an MoG approximation will require a high number of terms to accurately represent it (Shenoy, 2006).[4]

3. Learning univariate distributions

While Langseth et al. (2012) defined their translation procedure as a means to create MoTBF approximations of *known* distributions, this paper will utilize the translation for *learning* hybrid BNs from data. The proposed learning algorithm displays shortcomings when learning from certain scarce data sets (see also Section 5), and this observation has motivated a related but alternative approach for learning MoTBFs. The alternative approach is discussed in Section 3.2.

3.1. Learning utilizing kernel density-based representations

The top-level algorithm is to *i*) approximate the data using a kernel-density, and *ii*) approximate the kernel density with an MoTBF parameterization. We discuss each step below, and start by looking at univariate (marginal) distributions. We will move on to conditional distributions in Section 4.

Assume that $f(x)$ is the (unknown) density, which generated the univariate sample $\mathcal{D} = \{x_1, \ldots, x_N\}$. Next, let $h_{\mathcal{D}}(\cdot|t_w)$ be a kernel density estimator based on the samples $\mathcal{D}$ using kernel function t_w with bandwidth w. We define the kernel density estimator s.t. w approaches zero as $N \to \infty$. The soundness of the approach now rests upon the following proposition:

Proposition 1. *Let* $\tilde{\boldsymbol{\theta}}_N$ *be chosen to minimize* $\mathrm{KL}\left(h_{\mathcal{D}}(x|t_w) \parallel g_k(x|\tilde{\boldsymbol{\theta}}_N) \right)$. *Then* $\tilde{\boldsymbol{\theta}}_N$ *converges to the maximum likelihood estimator of* $\boldsymbol{\theta}$ *as* $N \to \infty$.

Proof. First we note that since

$$\mathrm{KL}\left(h_{\mathcal{D}}(x|t_w) \parallel g_k(x|\boldsymbol{\theta}) \right) = \int_x h_{\mathcal{D}}(x|t_w) \log\left(\frac{h_{\mathcal{D}}(x|t_w)}{g_k(x|\boldsymbol{\theta})} \right) dx,$$

minimizing $\mathrm{KL}\left(h_{\mathcal{D}}(x|t_w) \parallel g_k(x|\boldsymbol{\theta}) \right)$ wrt. $\boldsymbol{\theta}$ is equivalent to maximizing $\mathbb{E}_{h_{\mathcal{D}}}[\log g_k(X|\boldsymbol{\theta})]$ wrt. $\boldsymbol{\theta}$; the expectation is taken wrt. X, which is assumed to have density function $h_{\mathcal{D}}(\cdot|t_w)$. Next, since the bandwidth of $h_{\mathcal{D}}(\cdot|t_w)$ decreases to 0 as $N \to \infty$, we have that

$$h_{\mathcal{D}}(x|t_w) \to \frac{1}{N} \sum_{\ell=1}^{N} \delta(x - x_\ell)$$

as $N \to \infty$, where $\delta(\cdot)$ is Dirac's delta. Therefore,

$$\mathbb{E}_{h_{\mathcal{D}}}[\log g_k(X|\boldsymbol{\theta})] \to \int_x \sum_{\ell=1}^{N} \frac{1}{N} \delta(x - x_\ell) \cdot \log g_k(x|\boldsymbol{\theta}) dx = \frac{1}{N} \sum_{\ell=1}^{N} \log g_k(x_\ell|\boldsymbol{\theta}).$$

[4] Furthermore, the reverse argument does not favor MoGs over MoTBFs: A Gaussian distribution can accurately be represented by an MoTBF using only a few basis functions (Langseth et al., 2012).

It follows that the parameters that minimize the KL divergence are asymptotically also those that maximize the likelihood. □

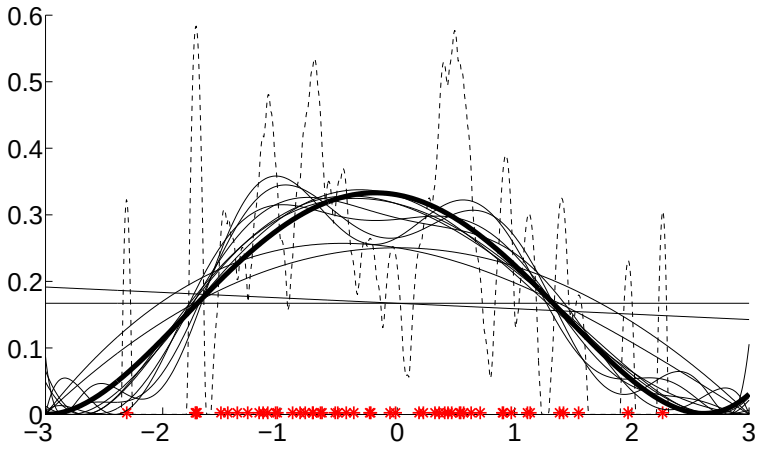


Figure 2: BIC-based learning: 50 samples from a standard Gaussian (stars on the x-axis) are evaluated. The kernel density estimator (thin dashed line) is the target of the MoTBF translations. g_0 up to g_{10} are shown; g_5 is the best in terms of BIC-score, and is drawn with double line-width.

The MoTBF density $g_k(x|\boldsymbol{\theta})$ uses $k+2$ parameters: We learn the interval of support (2 values), then estimate the k "free" θ_i-values (θ_0 is fixed to make sure the function integrates to one). Thus, we can choose between different models using an (approximate) BIC-score (Schwarz, 1978):

$$\mathrm{BIC}(g_k;\mathcal{D}) = \sum_{\ell=1}^{N} \log g_k(x_\ell|\hat{\theta}_N) - \frac{k+2}{2}\log N. \tag{4}$$

In the experiments reported in Section 5 we have used a greedy approach starting from g_0 and stopping as soon as an approximation g_k is better (in terms of BIC) than both g_{k+1} and g_{k+2}.[5] Our greedy procedure is exemplified in Figure 2, where an MoTBF with polynomial basis functions is learned from 50 samples from a standard Gaussian distribution (shown as stars on the x-axis). The kernel density approximation is drawn with a dashed line, and the MoTBF approximations from g_0 up to g_{10} are shown; g_5 is the best in terms of BIC-score, and is drawn with double line-width.

To fully specify the learning approach above, we need to further analyze the use of kernel density approximations as an intermediate representation between the data and the learned MoTBF. The use of kernel estimators for learning MTEs was first proposed by Romero et al. (2006), and further analyzed by Langseth et al. (2010). Previous attempts used Silverman's rule of thumb when selecting the bandwidth, $t_w^S \approx 1.06 \cdot \hat{\sigma} \cdot N^{-1/5}$, where $\hat{\sigma}$ is the empirical standard deviation of the data set; based on preliminary experiments we chose the Gaussian kernel for the results reported in this paper. By following this procedure, we get the results shown in Figure 3 (left-hand part of the figure): a kernel density estimator is fitted to 50 samples from a χ^2 distribution with 8 degrees of freedom, and the kernel density (drawn with the thin dashed line) is then used as a starting point for the BIC-based MoTBF learning. The learned MoTBF representation is defined using 2 basis functions. Visually, the MoTBF approximation is quite poor (it does not resemble the density function that generated the data). We argue that the reason for the poor result is that

[5] Computationally more demanding procedures can also be devised, e.g., to compare all subsets of basis functions from a fixed set $\{\psi_0, \psi_1, \ldots, \psi_\ell\}$.

using t_w^S is an unfortunate bandwidth choice, as it in principle leads us to smooth the data *twice*: once when employing the kernel density, and once when the MoTBF is fitted to the kernel density. Rather, we want the kernel density to be a faithful representation of the data. To illustrate the effect, the right-hand part of Figure 3 shows the result of using the scaled bandwidth $t_w^S/25$. For this bandwidth, the BIC-score is optimized using 6 basis functions. The results are visually more appealing, and this is underlined when calculating the log likelihood of a hold-out test-set, giving -2915.903 and -2828.781 for the two bandwidths, respectively.

We stress again that the role of the kernel density is to serve as a representation of the *data* and not of the *generating density*. The idea is to use the kernel density solely as a representation of the data so that it can become the target for the "translation procedure" (i.e., a function defined on L^2), and it is therefore not necessarily beneficial to find a bandwidth t_w that gives a good density estimator; see also Section 2.1. The idea can be further examined in Figure 2, where the "noisy-looking" kernel estimator is in fact smoothed by the translation procedure. The MoTBFs used in Figure 2 are polynomials, thus g_k is simply a polynomial of order k. The kernel density is the target for the MoTBF approximation, but using low-order polynomials enforces a smoothing effect. Finally, note also that since the BIC-score rates the MoTBF's fit with respect to the data, the kernel density is not used in the evaluation of the model fit.

We have further investigated the effect the kernel's bandwidth has on the learning results by examining a range of different data sets, both small and large, as defined by Langseth et al. (2010). For each data set, we have learned an MoTBF representation using the BIC score for model selection and with a set of bandwidths defined by $t_w \leftarrow t_w^S/\alpha$, where $\alpha \in \{1, 2, 5, 10, 25, 50\}$ is the bandwidth scale. Table 1 lists the results of the experiment in terms of the number of basis functions that are selected as well as the obtained log likelihood on a hold-out test-set. Generally, the results appear to be reasonably robust across the data sets as long as the bandwidth is "sufficiently small" (say, using $\alpha = 25$), but for some data sets the results are less stable (see, e.g., Beta(.5,.5)). This latter observation has motivated an alternative learning approach that attempts to avoid the double-smoothing effect (and the need for specifying a band-width parameter) by instead relying on the empirical cumulative distribution function (CDF) for the data rather than a kernel density estimate. This procedure is described next.

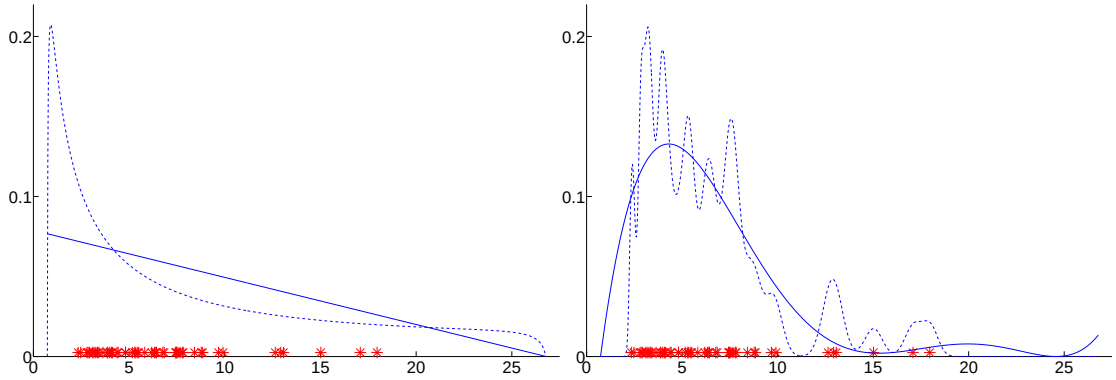


Figure 3: 50 samples from a χ_8^2 distribution are learned using a kernel density. The kernel density is shown with a thin dashed line, where the bandwidth is set using Silverman's rule of thumb (left figure) and one twenty-fifth of Silverman's rule of thumb (right). The learned MoTBF representations are defined using 2 and 6 basis functions in the left and right hand plots, respectively, and drawn in a thick line.

3.2. Learning utilizing a cumulative distribution function representation

The estimation procedure described above uses a kernel density estimate as a proxy for the data (relying on a user specified scaling parameter), which can lead to a "double-smoothing" effect of the data. One way to sidestep this problem is to use the empirical distribution function, defined

Distribution	Scaler α	$N = 25$		$N = 50$		$N = 1000$	
		#BF	Logllik	#BF	Logllik	#BF	Loglik
MTE	1	2	-2557.420	3	-2470.686	5	-2349.280
	2	3	-2481.669	5	-2352.511	9	-2315.446
	5	3	-2481.669	5	-2352.511	9	-2315.446
	10	3	-2481.669	5	-2352.511	12	-2308.048
	25	3	-2481.669	5	-2352.511	12	-2308.048
	50	3	-2481.669	5	-2352.511	12	-2308.048
Beta(.5,.5)	1	3	-1.878	5	169.211	9	253.910
	2	3	-1.878	5	169.211	9	253.910
	5	3	-1.878	5	169.211	9	253.910
	10	3	-1.878	3	78.774	9	253.910
	25	3	-1.878	3	78.774	9	253.910
	50	1	0.000	1	0.000	9	253.910
χ^2_8	1	2	-2915.903	2	-2915.903	5	-2724.892
	2	3	-2858.830	3	-2858.821	5	-2724.892
	5	3	-2858.830	6	-2828.781	6	-2724.121
	10	3	-2858.830	6	-2828.781	10	-2714.062
	25	3	-2858.830	6	-2828.781	10	-2714.062
	50	3	-2858.830	6	-2828.781	10	-2714.062
Gauss(0, 1)	1	5	-1431.732	5	-1441.553	7	-1424.159
	2	5	-1431.732	5	-1441.553	7	-1424.159
	5	5	-1431.732	5	-1441.553	7	-1424.159
	10	5	-1431.732	5	-1441.553	7	-1424.159
	25	5	-1431.732	5	-1441.553	9	-1419.724
	50	5	-1431.732	5	-1441.553	9	-1419.724
LN(0,1)	1	5	-1558.841	6	-1514.952	8	-1410.472
	2	5	-1558.841	7	-1445.409	8	-1410.472
	5	5	-1558.841	7	-1445.409	7	-1423.303
	10	5	-1558.841	7	-1445.409	8	-1410.472
	25	5	-1558.841	7	-1445.409	8	-1410.472
	50	6	-1476.036	7	-1445.409	8	-1410.472

Table 1: The effect of the chosen bandwidth wrt. the log likelihood of a test-set. In general, we see that results for "large" bandwidths ($\alpha = 1$) can be poor due to "double smoothing".

for a sample $\mathcal{D} = \{x_1, \ldots, x_N\}$ as

$$G_N(x) = \frac{1}{N} \sum_{\ell=1}^{N} \mathbf{1}\{x_\ell \leq x\}, \quad x \in \mathbb{R}, \tag{5}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function.

The density and distribution functions of an MoTBF are tightly connected. Therefore, if we can find an accurate approximation of the CDF, we can also get an accurate estimation of the density. The next results establish the relation between the parameters of the density and the CDF. We define $\eta_0 = 1$ and let η_i denote the anti-derivative of the basis function ψ_{i-1}. Thus, $\{\eta_0, \eta_1, \ldots\}$ is the set of functions that can be obtained from Ψ.

Lemma 1. *Consider a random variable X with CDF*

$$F(x) = \sum_{i=0}^{k} c_i \eta_i(x). \tag{6}$$

Then X has an MoTBF density.

Proof. The density of X is just the derivative of F,

$$f(x) = F'(x) = \sum_{i=0}^{k} c_i \, \eta_i'(x) = \sum_{i=1}^{k} c_i \, \psi_{i-1}(x) = \sum_{i=0}^{k-1} \tilde{c}_i \, \psi_i(x),$$

which is an MoTBF. □

Taking Lemma 1 into account, we can formulate an optimization procedure for estimating CDFs. The estimation of the parameters of a CDF, corresponding to an MoTBF on an interval $\Omega = [a, b]$, consists of solving the program

$$\begin{aligned} \textbf{minimize} \quad & \sum_{\ell=1}^{N} \left(G_N(x_\ell) - \sum_{i=0}^{k} c_i \eta_i(x_\ell) \right)^2 \\ \textbf{subject to} \quad & \sum_{i=0}^{k-1} c_i \, \psi_i(x) \geq 0 \quad \forall x \in \Omega, \\ & \sum_{i=0}^{k} c_i \eta_i(a) = 0, \text{ and } \sum_{i=0}^{k} c_i \eta_i(b) = 1, \end{aligned} \tag{7}$$

where the first constraint ensures that the CDF is non-decreasing, whereas the second and third constraints together ensures that the approximation will integrate to one. For this optimization we will still require that the set of basis functions Ψ is legal, but as the program is neither initialized with the GFS parameters nor are tied to them, we do not require the basis functions to be orthonormal.

As an example, let us consider the polynomial basis functions. Since this procedure does not require the basis functions to be orthonormal, we can simply use $\Psi = \{1, x, x^2, x^3, \ldots\}$, which entails that $\eta_i = x^i$ for $i = 0, 1, 2, \ldots$ Now the optimization problem is defined as follows:

$$\begin{aligned} \textbf{minimize} \quad & \sum_{\ell=1}^{N} \left(G_N(x_\ell) - \sum_{i=0}^{k} c_i \, x_\ell^i \right)^2 \\ \textbf{subject to} \quad & \sum_{i=1}^{k} i \, c_i \, x^{i-1} \geq 0 \quad \forall x \in \Omega, \quad \sum_{i=0}^{k} c_i \, a^i = 0 \text{ and } \sum_{i=0}^{k} c_i \, b^i = 1. \end{aligned}$$

The convergence of the densities estimated by the procedure described above is stated next.

Proposition 2. *Let f be an MoTBF density and let F be the corresponding CDF. Let X be a random variable with CDF F and support $\Omega = [a, b]$. Consider a sample of size N of variables with the same distribution as X. Let G_N be the empirical CDF, as defined in Equation (5). Then $\hat{F}_N(x)$, obtained by solving Program (7) is a consistent estimator of $F(x)$ in terms of the mean squared error for all $x \in \Omega$.*

Proof. Using Glivenko-Cantelli's theorem, we have that $|G_N(x) - F(x)| \to 0$ as $N \to \infty$. Now, taking into account that $\hat{F}_N$ is an MoTBF as well, it follows that $\hat{F}_N$ is an M-estimator (Huber, 1964) of F, which leads to a linear regression problem. The consistency of M-estimators in regression problems was analyzed, for instance, by Bai and Wu (1997). Thus,

$$\lim_{N\to\infty} \mathbb{E}\left[(\hat{F}_N(x) - F(x))^2\right] = 0 \text{ for all } x \in [a, b].$$

□

Proposition 2 states the consistency of the least squares estimators under the assumption that the target density is an MoTBF. In practice, this assumption is not too restrictive, as it is known that MoTBFs are able to accurately represent any univariate probability distribution (Langseth et al., 2010, 2012).

4. Learning conditional distributions

Recall that for a conditional MoTBF $f(x|\boldsymbol{z})$, the variables $\boldsymbol{Z}$ influence X only through the partitioning of $\Omega_{\boldsymbol{Z}}$, see Equation (2). Learning a conditional MoTBF therefore amounts to finding

- a partitioning $\mathcal{P}$ of $\Omega_{\boldsymbol{Z}}$, and
- a (univariate) MoTBF for each hyper-cube in the partitioning.

The algorithms for learning conditional MoTBFs proceed by iterating over the two steps above, however, for a given partitioning of the parent variables, the two proposed learning algorithms differ in how the conditional distribution is learned. With a fixed partitioning of the parent variables, the CDF-based learning approach considers each partition in isolation and simply learns an MoTBF model based on the data points consistent with that partition. For the kernel-based approach, the estimation procedure is slightly more elaborate as detailed below.

4.1. Kernel-based learning with a fixed partitioning

For a given hyper-cube $\Omega^l_{\boldsymbol{Z}} \in \mathcal{P}$ we start by approximating the conditional empirical distribution with a conditional kernel density estimate. For ease of exposition, consider a variable Y with parent X for which we have a data sample $\mathcal{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_N\}$, where $\boldsymbol{d}_i = (x_i, y_i)$. We now define the conditional kernel density estimate for Y given X as

$$h_{\mathcal{D}}(y|x, t_{w_y}, t_{w_x}) = \frac{\sum_{i=1}^N h_{y_i}(y|t_{w_y}) h_{x_i}(x|t_{w_x})}{\sum_{i=1}^N h_{x_i}(x|t_{w_x})},$$

where $h_{x_i}(x|t_{w_x})$ is a kernel density estimator based on x_i only and with bandwidth t_{w_x}; $h_{y_i}(y|t_{w_y})$ is defined similarly. Given a conditional kernel density estimator $h_{\mathcal{D}}(y|x, t_{w_y}, t_{w_x})$ and a partitioning $\mathcal{P}$ of Ω_X, we approximate $h_{\mathcal{D}}(y|x, t_{w_y}, t_{w_x})$ with an MoTBF potential $f(y|x)$ (see Equation (2)) by following the procedure of Langseth et al. (2012). Thus, for all $\Omega^l_X \in \mathcal{P}$, we seek

$$f(y|x \in \Omega^l_X) \sim h_{\mathcal{D}}(y|x \in \Omega^l_X, t_{w_y}, t_{w_x}) = \int_x h_{\mathcal{D}}(y|x, t_{w_y}, t_{w_x}) h_{\mathcal{D}}(x|x \in \Omega^l_X, t_{w_x}) dx,$$

where the integral can be approximated by $\sum_{i=1}^\tau h_{\mathcal{D}}(y|x_i, t_{w_y}, t_{w_x}) h_{\mathcal{D}}(x_i|x_i \in \Omega^l_X, t_{w_x})$ using data samples $x_1, \ldots, x_\tau$ from $\mathcal{D}$ belonging to Ω^l_X. That is, for a fixed partitioning of Ω_X learning a conditional MoTBF potential reduces to estimating a univariate MoTBF potential (as described in Section 3) for each partition $\Omega^l_X \in \mathcal{P}$.

4.2. Finding a partitioning of the conditioning variables

In order to find a partitioning of $\Omega_{\boldsymbol{Z}}$ we employ a myopic strategy, where we in each step consider a bisection of an existing partition along each $Z \in \boldsymbol{Z}$. That is, for each partition $\Omega'_{\boldsymbol{Z}} \in \mathcal{P}$ the algorithm evaluates the potential gain of splitting the partition along $Z \in \boldsymbol{Z}$. After scoring the candidate partitions the algorithm selects the highest scoring partition $\Omega^{\text{BS}}_{\boldsymbol{Z}}$ and splitting variable Z_{BS}, and learns MoTBF representations of the two induced sub-partitions $\Omega^{\text{BS},1}_{\boldsymbol{Z}}$ and $\Omega^{\text{BS},2}_{\boldsymbol{Z}}$. To guide the selection of a candidate partition $\Omega'_{\boldsymbol{Z}}$ we consider the potential improvement in BIC score resulting from splitting that partition:

$$\text{BIC-Gain}(\Omega'_{\boldsymbol{Z}}, Z) = BIC(f', \mathcal{D}) - BIC(f, \mathcal{D}),$$

where f' is the conditional MoTBF potential defined over the partitioning $\{\mathcal{P} \setminus \Omega'_{\boldsymbol{Z}}\} \cup \{\Omega^{Z,1}_{\boldsymbol{Z}}, \Omega^{Z,2}_{\boldsymbol{Z}}\}$. In principle, when scoring the model f' one would need to find the basis functions (and the corresponding parameters) maximizing this score. This will, however, be computationally difficult, and instead we lower-bound the improvement in BIC score by using the same set of basis functions as was used for the parent partition $\Omega'_{\boldsymbol{Z}}$. It should be noted that for the calculation of the improvement in BIC score, we only need to consider the parts of the score relating to the partition $\Omega'_{\boldsymbol{Z}}$, since the contributions from the partitions for which f and f' agree cancel out; this property also supports an efficient caching scheme for BIC-Gain. The overall procedure for learning conditional MoTBFs is summarized in Algorithm 1. Note that when learning MoTBF potentials for different intervals (Line 5 in Algorithm 1), the learning algorithm may chose different basis functions for the different intervals.

Algorithm 1 Learning conditional MoTBFs.

1: $\mathcal{P} \leftarrow \{\Omega_{\boldsymbol{Z}}\}$
2: **repeat**
3: $(\Omega^{\text{BS}}_{\boldsymbol{Z}}, Z_{\text{BS}}) \leftarrow \arg\max_{\Omega'_{\boldsymbol{Z}} \in \mathcal{P}, Z \in \boldsymbol{Z}} \text{BIC-Gain}(\Omega'_{\boldsymbol{Z}}, Z)$
4: **if** $\text{BIC-Gain}(\Omega^{\text{BS}}_{\boldsymbol{Z}}, Z_{\text{BS}}) > 0$ **then**
5: Learn MoTBF potentials for $\Omega^{\text{BS},1}_{\boldsymbol{Z}}$ and $\Omega^{\text{BS},2}_{\boldsymbol{Z}}$.
6: $\mathcal{P} \leftarrow \left\{\mathcal{P} \setminus \Omega^{\text{BS}}_{\boldsymbol{Z}}\right\} \cup \left\{\Omega^{\text{BS},1}_{\boldsymbol{Z}}, \Omega^{\text{BS},2}_{\boldsymbol{Z}}\right\}$.
7: **else**
8: terminate.
9: **end if**
10: **until** false

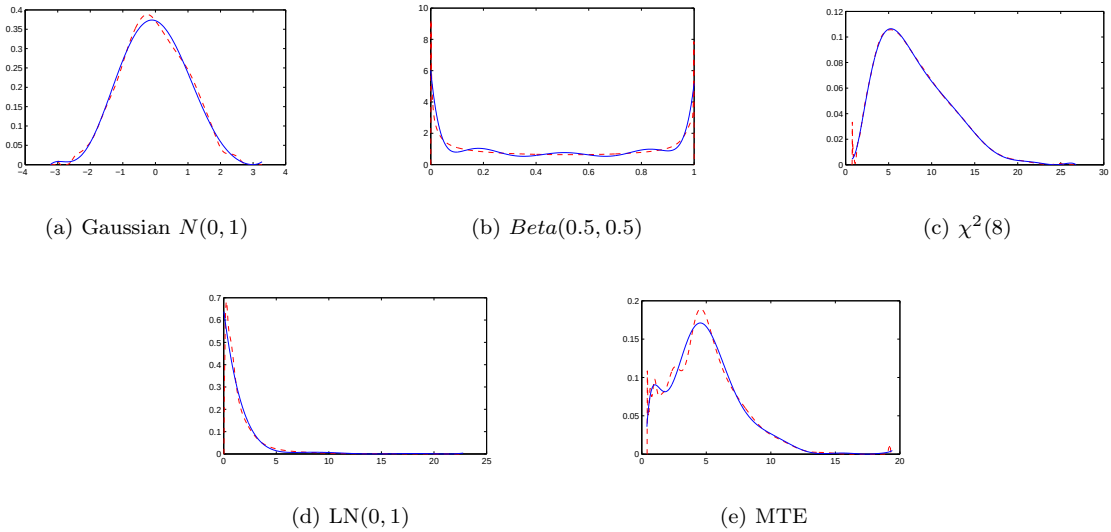
5. Experiments

In this section we report on a series of experimental studies undertaken to analyze the proposed methods and compare them to their most immediate competitors.

5.1. Learning univariate probability distributions

In order to analyze the behavior of the algorithm when learning univariate distributions we have sampled data sets from five different distributions: an MTE distribution (*MTE*), a beta distribution (*Beta*$(0.5, 0.5)$), a χ^2 distribution with eight degrees of freedom (χ^2_8), a standard normal distribution (*Gaussian*), and a log-normal distribution with $\mu = 0$ and $\sigma = 1$ (*Log-Normal*). Specifically, for each distribution we have generated one data set of 1000 samples and 10 data sets of 25 samples and 50 samples, respectively, giving a total of 21 data sets for each of the distributions. In order to test the predictive ability of the methods, a separate test-set of 1000 samples was also generated for each of the distributions.

When learning the models, we used the BIC-score as a model selection criteria. For illustration, consider Figure 4 showing the MoTBF models learned using the kernel approach and based on



(a) Gaussian $N(0,1)$ (b) $Beta(0.5, 0.5)$ (c) $\chi^2(8)$

(d) LN$(0,1)$ (e) MTE

Figure 4: The figure shows the MoTBF models learned based on 1000 samples from five different distributions using the kernel-based approach. The dashed lines show the kernel density estimates and the solid lines show the learned MoTBFs.

the data sets consisting of 1000 samples. The displayed MoTBFs correspond to the best scoring models spanned by the first twelve (polynomial) basis functions.

In order to estimate the predictive performance of the algorithms we learned an MoTBF model for each of the generated data sets and evaluated them in terms of their log-likelihood on the hold-out test-set. The results for the data sets with 1000 samples are reported in Table 2. The column named *Baseline* gives the test-set log-likelihoods obtained using the method by Langseth et al. (2010) and, for ease of presentation, the remaining columns give the difference between the baseline and the associated models. Here, a positive value corresponds to an improvement over the baseline in terms of test-set log-likelihood, and a negative value indicates a decrease in quality. For instance, Kernel/MoP obtained a likelihood score of -2315.45 on the MTE-data, which is 30.20 poorer than the baseline model.

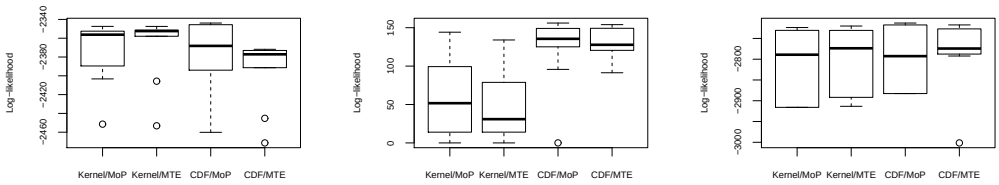
Data set	Baseline	Kernel/MoP	Kernel/MTE	CDF/MoP	CDF/MTE
MTE	-2285.25	-30.20	-69.31	-50.90	-31.42
Beta(0.5, 0.5)	249.45	4.46	-38.24	2.16	-40.82
χ^2_8	-2702.67	-21.45	-18.83	-6.62	-4.06
Gaussian	-1430.88	11.16	-156.85	-3.34	-1.56
Log-Normal	-1358.38	-52.09	-50.86	-60.73	-37.46

Table 2: Difference in test-set log likelihood between the baseline (the method by Langseth et al. (2010)) and the proposed estimators. The BIC score was used for model selection.

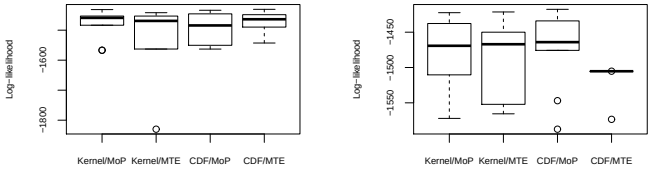
To determine whether the results reported in Table 2 discriminate among the different algorithms, we have run Friedman's test with significance level 0.05. The results indicate that there are no significant differences among the learning methods used.

An analysis similar to the one above was performed for the data sets with 25 and 50 samples each. The results are summarized in Figure 5 and Figure 6, showing box-plots generated from the data sets with 50 and 25 samples, respectively.

For some of the data sets Friedman's test indicated that there were significant differences between the methods. In those cases we deployed Wilcoxon-Nemenyi-McDonald-Thompson's post-hoc test (Hollander and Wolfe, 1999) to make a pair-wise comparison of the methods. The results



(a) MTE (b) Beta (c) χ^2



(d) Gaussian (e) Log-normal

Figure 5: Boxplots for the 50 samples data sets.

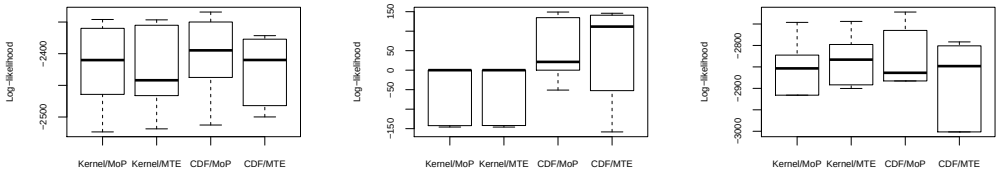
of the these experiments are summarized in Table 3; see the appendix for more details about the results.

	MTE	*Beta*$(0.5, 0.5)$	χ^2	*Gaussian*	*Log-norm*
25	CDF/MoP≻CDF/MTE CDF/MoP≻Kernel/MTE	No diff.	CDF/MoP≻CDF/MTE	No diff.	No diff.
50	Kernel/MTE≻CDF/MTE	CDF/MoP≻Kernel/MoP CDF/MTE≻Kernel/MoP CDF/MoP≻Kernel/MTE	No diff.	No diff.	No diff.

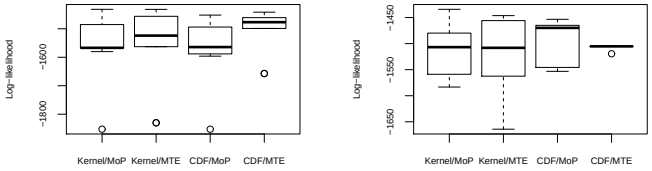
Table 3: Summary of the experimental results for the data sets with 25 and 50 samples. The table lists the significant differences found using Wilcoxon-Nemenyi-McDonald-Thompson's post-hoc test with significance level 0.05.

From the results we see that the CDF-based approach appears to have an advantage over the kernel-based approach for smaller data sets. We hypothesize that this is due to the target-distribution having relatively large areas of low density in these situations, which in turn means that it requires a high number of basis functions to be faithfully captured. Due to our greedy model evaluation strategy (based on the BIC-score, which penalizes model complexity), these volatile model candidates are not considered, and the search strategy fails to find suitable representations.

As a last comment we would like to note that the speedup from the direct maximum likelihood approach (Langseth et al., 2010) to our approaches is above a factor 10. The main contribution to the speed increase is that while the previous technique was based on an iterative scheme, where each potential solution (living in a very complicated likelihood-landscape) needed to be evaluated using a computationally expensive procedure, the current approaches cast the learning problem as convex optimization problems with much cheaper evaluations.



(a) MTE (b) Beta (c) χ^2



(d) Gaussian (e) Log-normal

Figure 6: Boxplots for the 25 samples data sets.

5.2. Learning conditional probability distributions

We exemplify the learning of conditional distributions by generating data from a model with two variables X and Y, where X follows a standard Gaussian distribution and $Y|\{X = x\} \sim \mathcal{N}(x/2, 1)$. Data sets containing 50, 500, 2500 and 5000 cases were generated as input to Algorithm 1, which was instantiated using the two different methods for learning MoTBF distributions.

The conditional distributions produced by the kernel-based approach are shown in Figure 7, with the parent variable displayed on the x-axis and the child variable on the y-axis. For the smallest data set consisting of 50 cases, the BIC-score only gave support for a single split-point that was inserted at the midpoint of the support interval for X. As the size of the training-sets increases, the BIC score selects more and more refined models, allowing itself to use more parameters to represent the correlation between X and Y as the correlation is more and more clearly manifested in the training data. Notice that the algorithm uses more effort on refining the part of the model, where the bulk of the data is found (i.e., around $x \approx 0$). The results of learning the conditional distribution using the CDF approach show the same behavior as in Figure 7.

Returning to the issue of inference as described in Section 2.2, we have analyzed the effect of forcing all the basis functions for a conditional distribution to be defined over the same domain. As mentioned in Section 2.2, the larger the domain of the density being approximated, the more basis functions will usually be required in order to find good representations of the low density regions. This is also reflected in Figure 8, which shows the results of learning a conditional MoTBF based on the data described above, but this time using the CDF learning approach with the restriction that the basis functions are defined over the same domain. Compared to Figure 7 we see that fewer split points are inserted for the parent variable. This is due to the increase in the number of basis functions that are needed to capture the behavior of the density in the extended regions, thereby increasing the complexity of the overall model.

We have also experimented with learning these extended conditional distributions using the

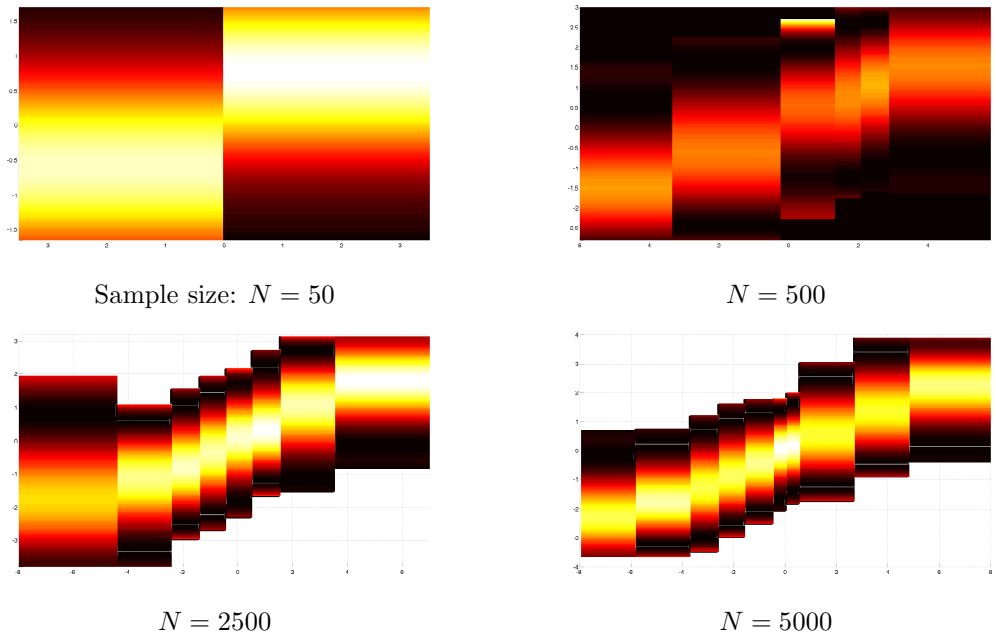


Figure 7: Learning a conditional linear Gaussian distribution using the kernel-based method and polynomial basis functions. Models were selected based on their BIC score. Note how finer model granularity is selected as the size of the training-set grows, and how the discretization effort is kept to the area with the bulk of the data.

kernel-based method. As indicated by the results for the univariate distributions, the kernel-based method appears to be less robust when learning from few data samples and, by hypothesis, when learning distributions with larger low-density regions. This observation was also reflected in the initial learning experiments with extended conditional densities, where the complexity of the MoTBF models (i.e., the number of basis functions) dominated the overall model complexity so that model selection using the BIC-score failed to divide the interval for the parent variable and, in effect, did not capture the correlation between the variables.

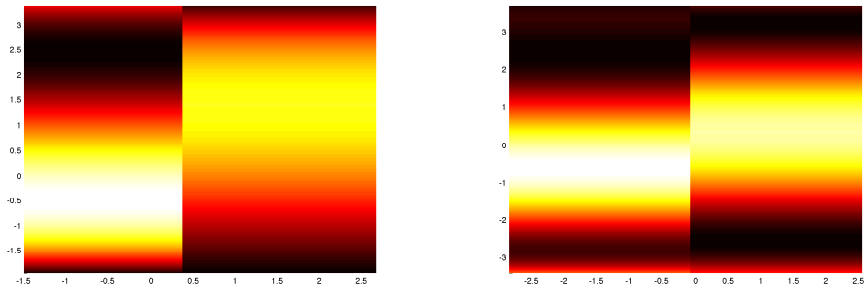
5.3. Learning from real-world datasets

In order to test the performance of the proposed estimation procedure in real-world scenarios, we conducted an experiment over a set of databases taken from the UCI machine learning repository (Bache and Lichman, 2013). Details of the datasets are given in Table 4. In the experiments we repeated the following steps for each database:

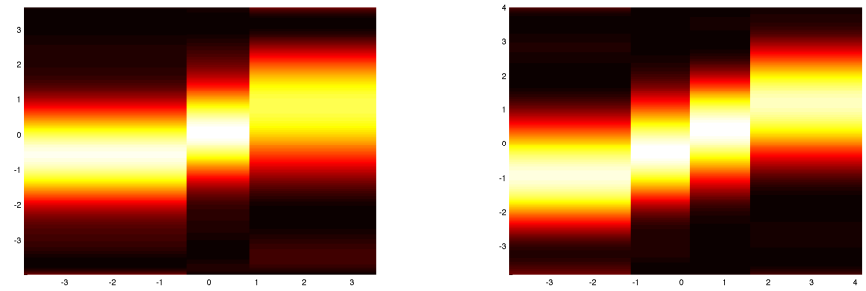
1. Discretize the variables in the database using equal width binning with three resulting values.
2. Use the PC algorithm (Spirtes et al., 1993) to obtain a Bayesian network structure from the discretized database.
3. Estimate a conditional MoTBF density for each variable given its parents in the learned network structure. The estimation procedure follows the scheme given in Section 4 combined with the CDF approach for learning univariate densities.

The algorithms used in this experiment are denoted as follows.

- MTEi, with $i = 2, 3, 4$. The algorithm described in Section 4 particularized for MTEs by using exponential functions as basis functions, and allowing the domain of the parent variables in conditionals to be split at most i times.
- MOPi, with $i = 2, 3, 4$. Similar specification as above, but using polynomials as basis functions instead of exponential functions.



Sample size: $N = 50$ $N = 500$



$N = 2500$ $N = 5000$

Figure 8: Learning a conditional linear Gaussian distribution using the CDF-based approach and polynomial basis functions. The models are learned with the constraint that the domain of all conditional distributions should be the same (see Section 2.2). Note that the selected models typically are less refined than their counter-parts learned without this restriction.

dataset	#instances	#variables
`diabetes`	768	8
`disclosure`	662	4
`ecoli`	336	5
`glass`	163	7
`iris`	150	4
`seeds`	209	7
`segmentation`	210	16
`slump`	103	10
`vertebral`	310	6
`waveform`	5000	21

Table 4: Description of the databases used in the experiments

- Discrete$i$, with $i = 3, 6, 8, 10$. The algorithm described in Section 4 dividing the domain of each variable into i pieces by equal frequency binning and fitting a constant density to each interval.

For each of the networks, we ran the above mentioned algorithms and computed the log-likelihood and the BIC score for each database using a 5-fold cross validation scheme. Notice that, as the likelihood is not comparable between discrete and continuous distributions due to the fact that continuous densities are not bounded above, we have represented the distribution of each discretized variable as a continuous density with a constant value in each of the regions corresponding to the discrete values. For instance, assume a variable X with support $[0, 3]$ is discretized by equal width binning into three values 0, 1 and 2 corresponding to intervals $[0, 1)$, $[1, 2)$ and $[2, 3]$. Assume also that $P(X = 0) = 0.2$, $P(X = 1) = 0.5$ and $P(X = 2) = 0.3$. The distribution of X is then represented with a continuous density

$$f(x) = \begin{cases} 0.2 & \text{if } 0 \leq x < 1, \\ 0.5 & \text{if } 1 \leq x < 2, \\ 0.3 & \text{if } 2 \leq x \leq 3, \\ 0 & \text{otherwise.} \end{cases}$$

The results are displayed in Tables 5 and 6. For each table, the column named *Baseline* give the actual results of the Discrete3 algorithm, whereas the other columns give the difference between that approach and Discrete3 (positive numbers indicate improvements). To determine whether the results discriminate among the different algorithms, we have run Friedman's test with significance level 0.05. For the data sets where Friedman's test indicated that there were significant differences between the methods we carried out a pair-wise comparison using Wilcoxon-Nemenyi-McDonald-Thompson's post-hoc test (Hollander and Wolfe, 1999). The results of the comparison are displayed in Table 7. The statistical analysis suggests that MoTBFs are superior to discretization in terms of BIC score while no significant differences are found in terms of likelihood, except for the simplest discrete model (Discrete3), that is outperformed by MOP3 and Discrete10.

We have conducted the same statistical tests but considering only three groups of results, namely Discrete, MTE and MOP, where for each dataset, Discrete represents the best score among those obtained by Discrete3, 6, 8 and 10, and equivalently for MTE and MOP. The result of Friedman's test indicates that there are no significant differences in terms of likelihood between the three groups, with significance level 0.05. In what concerns BIC score, the only significant difference found points at MOP algorithms outperforming the Discrete ones.

6. Conclusions

In this paper we have examined two techniques for learning the parameters of a hybrid Bayesian network from data. In the first approach we find a kernel density estimate of the data and utilize an

Database	Baseline	Discrete6	Discrete8	Discrete10	MTE2	MTE3	MTE4	MOP2	MOP3	MOP4
diabetes	-2055.67	139.55	200.70	338.70	**788.82**	785.30	775.92	773.25	788.32	779.41
disclosure	-960.68	6.66	9.77	14.37	**210.61**	**210.61**	**210.61**	210.47	210.47	210.47
ecoli	-474.79	54.86	68.82	**83.10**	31.68	37.95	32.44	30.54	35.08	31.53
glass	-316.58	55.68	**70.80**	68.98	40.14	43.04	39.28	58.09	61.11	61.85
iris	-127.13	9.81	**11.29**	10.23	-17.88	-13.10	-16.85	-5.92	-3.86	-5.08
seeds	-309.44	63.90	78.15	**78.29**	-20.02	-31.79	-33.37	-16.69	-25.06	-31.46
segmentation	-775.69	143.50	192.22	228.89	248.76	247.27	236.30	**316.45**	308.20	313.18
slump	-265.45	7.05	8.66	12.33	43.20	43.20	**43.20**	36.84	36.84	36.84
vertebral	-542.91	62.92	97.22	**110.47**	63.47	62.33	56.37	68.83	64.88	61.49
waveform	-35620.98	2291.83	3493.09	4436.38	7206.36	7005.30	6884.96	**8081.11**	7805.20	7559.40

Table 5: Average log-likelihood computed using 5-fold cross validation for the networks learned by the tested algorithms. The actual numbers are reported for the baseline model, which is standard discretization using three states (Discrete3). For the other models we report their improvement wrt. the baseline. Boldfaced numbers indicate the best algorithm for each data set.

Algorithm	Baseline	Discrete6	Discrete8	Discrete10	MTE2	MTE3	MTE4	MOP2	MOP3	MOP4
diabetes	-2169.45	-522.00	-1239.69	-1876.56	**741.51**	711.79	682.29	714.35	691.66	678.21
disclosure	-986.09	-53.44	-105.07	-166.94	155.89	155.89	155.89	**173.34**	**173.34**	**173.34**
ecoli	-527.82	-193.05	-420.27	-606.22	1.82	-8.74	-16.78	**18.77**	-9.51	-10.53
glass	-384.17	-179.93	-317.82	-437.90	15.43	14.07	6.17	**30.28**	25.21	13.01
iris	**-150.25**	-58.90	-120.34	-177.86	-30.82	-34.19	-32.50	-28.04	-22.92	-25.50
seeds	**-380.35**	-285.08	-590.34	-915.64	-78.67	-80.74	-87.53	-52.18	-76.57	-68.06
segmentation	-883.34	-197.37	-495.50	-901.37	107.48	93.29	83.81	**191.99**	166.92	161.43
slump	-306.58	-93.91	-180.90	-278.49	2.06	2.06	2.06	**6.89**	**6.89**	**6.89**
vertebral	-618.80	-370.55	-753.61	-849.30	13.98	12.02	4.41	**32.96**	26.94	25.20
waveform	-36751.08	-9671.72	-21957.17	-34613.86	6929.35	6422.97	6154.81	**7557.50**	7243.59	7057.20

Table 6: Average BIC score computed computed using 5-fold cross validation for the networks learned by the tested algorithms. The actual numbers are reported for Discrete3 (denoted "Baseline"); for the others we report the improvement wrt. the baseline. Boldfaced numbers indicate the best algorithm for each data set.

Log-likelihood	BIC
Discrete10 $\succ$ Discrete3	Discrete3 $\succ$ Discrete10
MOP3 $\succ$ Discrete3	MOP2 $\succ$ Discrete10
	MOP3 $\succ$ Discrete10
	MOP4 $\succ$ Discrete10
	MTE2 $\succ$ Discrete10
	MTE3 $\succ$ Discrete10
	MOP2 $\succ$ Discrete6
	MOP3 $\succ$ Discrete6
	MOP4 $\succ$ Discrete6
	MOP2 $\succ$ Discrete8
	MOP3 $\succ$ Discrete8
	MOP4 $\succ$ Discrete8
	MTE2 $\succ$ Discrete8

Table 7: Summary of the experimental results for the real-world data sets. The table lists the significant differences found using Wilcoxon-Nemenyi-McDonald-Thompson's post-hoc test with significance level 0.05.

effective translation procedure designed for approximating any marginal or conditional distribution function (in this case a kernel density) by an MoTBF distribution. The resulting parameters were asymptotically shown to be the maximum likelihood estimators for the MoTBF model, and we found promising results when learning from relatively large data sets. On the other hand, we noted a lack of robustness when learning from small data sets, and this observation motivated a second approach. Here we used the empirical cumulative distribution function as target for the MoTBF approximation, and the empirical results point towards this strategy being more robust when data are scarce.

Although both methods were found to be slightly worse than state-of-the-art techniques in terms of the log-likelihood score, the speed-up over previous methods is substantial, and we are currently investigating how the method scales to larger domains. Additionally, the types of conditional distributions that can be learned seems to support an efficient inference scheme and will be analyzed and further developed as part of future work.

Acknowledgments

This work has been supported by a Senior Grant in the frame of the CALL UCM-EEA-ABEL-02-2009 of the Abel Extraordinary Chair (NILS Project), and by the Spanish Ministry of Economy and Competitiveness, through projects TIN2010-20900-C04-02,03 (entitled Data mining with PGMs: New algorithms and applications), by Junta de Andalucía through project P11-TIC-7821 and by ERDF (FEDER) funds.

References

K. Bache and M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Z.D. Bai and Y Wu. General M-estimation. *Journal of Multivariate Analysis*, 63:119–135, 1997.

Barry R. Cobb and Prakash P. Shenoy. Inference in hybrid Bayesian networks with mixtures of truncated exponentials. *International Journal of Approximate Reasoning*, 41(3):257–286, April 2006. ISSN 0888613X. doi: 10.1016/j.ijar.2005.06.002. URL `http://dx.doi.org/10.1016/j.ijar.2005.06.002`.

John B. Fraleigh and Raymond A. Beauregard. *Linear Algebra*. Pearson, third edition, 1994.

Myles Hollander and Douglas A. Wolfe. *Nonparametric Statistical Methods*. Wiley, 2nd edition, 1999.

P.J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35: 73–101, 1964.

S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:76–86, 1951.

H. Langseth, T.D. Nielsen, R. Rumí, and A. Salmerón. Maximum likelihood learning of conditional MTE distributions. *ECSQARU 2009. Lecture Notes in Computer Science*, 5590:240–251, 2009.

H. Langseth, T.D. Nielsen, R. Rumí, and A. Salmerón. Parameter estimation and model selection for mixtures of truncated exponentials. *International Journal of Approximate Reasoning*, 51: 485–498, 2010.

H. Langseth, T.D. Nielsen, R. Rumí, and A. Salmerón. Mixtures of truncated basis functions. *International Journal of Approximate Reasoning*, 53:212–227, 2012.

S.L. Lauritzen and F. Jensen. Stable local computation with conditional Gaussian distributions. *Statistics and Computing*, 11:191–203, 2001.

Pedro Luis López-Cruz, Concha Bielza, and Pedro Larrañaga. Learning mixtures of polynomials from data using B-spline interpolation. In Andrés Cano, Manuel Gómez-Olmedo, and Thomas D. Nielsen, editors, *Proceedings of the 6th European Workshop on Probabilistic GraphicalModels (PGM'12)*, pages 211–218, 2012.

S. Moral, R. Rumí, and A. Salmerón. Mixtures of truncated exponentials in hybrid Bayesian networks. In *ECSQARU'01. Lecture Notes in Artificial Intelligence*, volume 2143, pages 135–143, 2001.

W. O. Pennell. A generalized fourier series representation of a function. *The American Mathematical Monthly*, 37(9):462–472, 1930.

V. Romero, R. Rumí, and A. Salmerón. Learning hybrid Bayesian networks using mixtures of truncated exponentials. *International Journal of Approximate Reasoning*, 42:54–68, 2006.

R. Rumí and A. Salmerón. Approximate probability propagation with mixtures of truncated exponentials. *International Journal of Approximate Reasoning*, 45:191–210, 2007.

Gideon Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

Glenn R. Shafer and Prakash P. Shenoy. Probability Propagation. *Annals of Mathematics and Artificial Intelligence*, 2:327–352, 1990.

P.P. Shenoy. Inference in hybrid Bayesian networks with mixtures of Gaussians. In R. Dechter and T. Richardson, editors, *Uncertainty in Artificial Intelligence: Procs. of the Twenty Second Conf.*, pages 428–436. Morgan Kaufmann, San Francisco, CA, 2006.

P.P. Shenoy and J.C. West. Inference in hybrid Bayesian networks using mixtures of polynomials. *International Journal of Approximate Reasoning*, 52:641–657, 2011.

Prakash P. Shenoy. Two issues in using mixtures of polynomials for inference in hybrid Bayesian networks. *International Journal of Approximate Reasoning*, 53(5):847–866, July 2012. ISSN 0888613X. doi: 10.1016/j.ijar.2012.01.008. URL `http://dx.doi.org/10.1016/j.ijar.2012.01.008`.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, prediction and search*, volume 81 of *Lecture Notes in Statistics*. Springer Verlag, 1993.

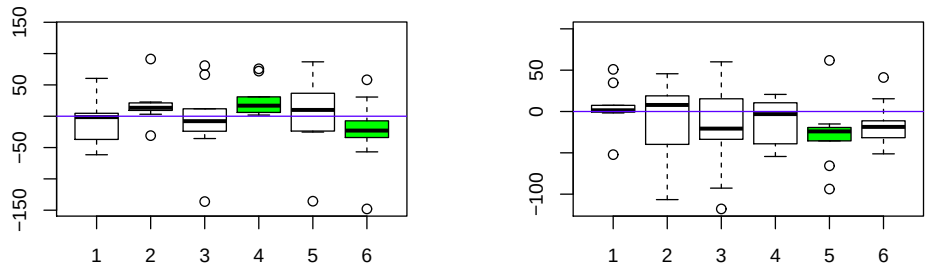
D.M. Titterington, A.F.M. Smith, and U.E. Makov. *Statistical Analysis of Finite Mixture Distributions*. John Wiley and Sons, New York, 1985.

Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM Review*, 38:49–95, 1996.

Assaf J. Zeevi and Ronny Meir. Density estimation through convex combinations of densities: Approximation and estimation bounds. *Neural Networks*, 10:99–109, 1997.

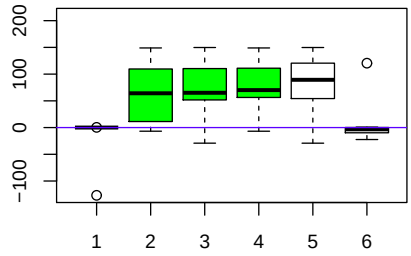
Appendix A. Detailed experimental results

Figure A.9 shows box-plots of the difference in the log-likelihood results for those data sets, where Friedman's test indicated a significant difference when comparing all the learning algorithms.

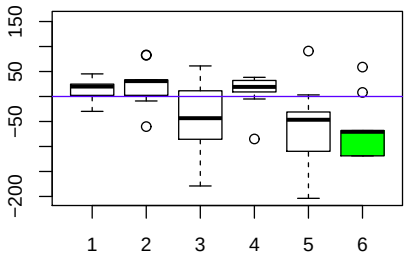


(a) MTE, 25 samples

(b) MTE, 50 samples



(c) Beta(0.5,0.5), 50 samples



(d) χ^2_8, 25 samples

Figure A.9: Box-plots of the log-likelihood differences for all pairwise comparisons of the learning algorithms. The labels in the plots correspond to 1 - Kernel/MTE vs. Kernel/MoP 2 - CDF/MoP vs. Kernel/MoP , 3 - CDF/MTE vs. Kernel/MoP , 4 - CDF/MoP vs. Kernel/MTE , 5 - CDF/MTE vs. Kernel/MTE , 6 - CDF/MTE vs. CDF/MoP. The colored boxes indicate situations, where there is a significant difference the results according to Nemenyi's test with significance level 0.05.