Maximum Likelihood Learning of Conditional MTE Distributions

Helge Langseth[1], Thomas D. Nielsen[2], Rafael Rumí[3], and Antonio Salmerón[3]

[1] Department of Computer and Information Science,
The Norwegian University of Science and Technology, Norway
[2] Department of Computer Science, Aalborg University, Denmark
[3] Department of Statistics and Applied Mathematics University of Almería, Spain

Abstract. We describe a procedure for inducing conditional densities within the mixtures of truncated exponentials (MTE) framework. We analyse possible conditional MTE specifications and propose a model selection scheme, based on the BIC score, for partitioning the domain of the conditioning variables. Finally, experimental results demonstrate the applicability of the learning procedure as well as the expressive power of the conditional MTE distribution.

1 Introduction

The main difficulty when modelling hybrid domains (i.e., domains containing both discrete and continuous variables) using Bayesian networks, is to find a representation of the joint distribution that is compatible with the operations used by existing inference algorithms: Algorithms for exact inference based on local computations, like the Shenoy-Shafer scheme [1], require that the joint distribution over the variables in the network are closed under marginalization and multiplication.

This can be achieved by discretizing the domain of the continuous variables [2, 3], which is a simple (but sometimes inaccurate) solution. A more elaborate approach is based on the use of mixtures of truncated exponentials (MTE) [4]. One of the advantages of this representation is that MTE distributions allow discrete and continuous variables to be treated in a uniform fashion, and since the family of MTEs is closed under marginalization and multiplication, inference in an MTE network can be performed efficiently using the Shenoy-Shafer architecture [1]. Also, the expressive power of MTEs was demonstrated in [5], where the most commonly used distributions were accurately approximated by MTEs.

The task of learning MTEs from data was initially approached using least squares estimation [6, 7]. However, this technique does not combine well with more general model selection problems, as many standard score functions for model selection, including the Bayesian information criterion (BIC) [8], assume Maximum likelihood (ML) parameter estimates to be available.

Two kinds of distributions can be found in a Bayesian network: univariate distributions (for nodes with no parents), and conditional distributions (for nodes with parents). ML learning of univariate distributions was introduced in [9]. However, the problem of learning conditional densities has so far only been described using least squares estimation [10]. In this paper, we study ML-based learning of conditional densities from data.

2 Preliminaries

Consider the problem of estimating a conditional density $f(x|\boldsymbol{y})$ from data. In this paper we will concentrate on the case in which X and $\boldsymbol{Y} = \{Y_1, \ldots Y_r\}$ are continuous, and use $\Omega_{X,\boldsymbol{Y}} \subseteq \mathbb{R}^{r+1}$ to represent the support of the distribution function $f(x, \boldsymbol{y})$. Furthermore, we let $\{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_K\}$ be a partition of $\Omega_{X,\boldsymbol{Y}}$. An *MTE potential* [4] for the random vector $\{X, Y_1, \ldots, Y_r\}$ is a function that, for each $k \in \{1, \ldots, K\}$, can be written as

$$f(x, \boldsymbol{y}) = a_0 + \sum_{j=1}^{m} a_j \exp\left(b_j x + \boldsymbol{c}_j^{\mathrm{T}} \boldsymbol{y}\right), (x, \boldsymbol{y}) \in \boldsymbol{I}_k. \quad (1)$$

The main problems to solve when inducing MTE potentials from data are *i*) determining the partition $\{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_K\}$, *ii*) determining m (the number of exponential terms) for each $\boldsymbol{I}_k$, and *iii*) estimating the parameters. Throughout the paper we will consider a training data set $\mathcal{D}$ with n records, and each record containing observations of all $r+1$ variables without missing values. We will write $\mathcal{D}(\boldsymbol{R})$ to denote the subset of $\mathcal{D}$ where the restriction $\boldsymbol{R}$ is fulfilled. For example $\mathcal{D}(y_1 \leq \alpha)$ selects all records for which the variable $y_1 \leq \alpha$.

3 Conditional distributions and MTEs

Before we investigate methods for learning conditional distributions from data, we will consider how to define conditional MTE distributions. Unfortunately, since the class of MTE potentials is not closed under division, we do not know the most general form of the conditional distribution function. However, for a function $g(x, \boldsymbol{y})$ to be a conditional MTE distribution, there are three assumptions we will require to be fulfilled:

1. **Generating joint**: $g(x, \boldsymbol{y}) \cdot f(\boldsymbol{y})$ should be an MTE potential over $(x, \boldsymbol{y})$, and the result should be equal to the joint density $f(x, \boldsymbol{y})$, where $f(y)$ is the marginal distribution for Y.
2. **Conditioning**: $g(x, \boldsymbol{y}_0)$ should be an MTE density over X for any fixed $\boldsymbol{y}_0$.
3. **Closed under marginalization**: For any BN structure and specification of conditionals, the product $\prod_{i=1}^{n} g(x_i, \mathrm{pa}(x_i))$ must support marginalization of any variable in closed form, and the result should be an MTE potential.

The first two conditions are local in nature, whilst the third is global. Attending only to the local conditions, the natural way to define a conditional density would be as $f(x|\boldsymbol{y}) = f(x,\boldsymbol{y})/f(\boldsymbol{y})$, where $f(x,\boldsymbol{y})$ and $f(\boldsymbol{y})$ are MTEs. Formally, a conditional MTE density under these assumptions would be of the form

$$f(x|\boldsymbol{y}) = \frac{f(x,\boldsymbol{y})}{f(\boldsymbol{y})} = \frac{a_0}{f(\boldsymbol{y})} + \sum_{i=1}^{m} \frac{a_i \exp(\boldsymbol{c}_i^{\mathrm{T}}\boldsymbol{y})}{f(\boldsymbol{y})} \cdot \exp(b_i x), (x,\boldsymbol{y}) \in \boldsymbol{I}_k, \tag{2}$$

where we have assumed that $f(x,\boldsymbol{y}) = a_0 + \sum_{i=1}^{m} a_i \exp\left(b_i x + \boldsymbol{c}_i^{\mathrm{T}}\boldsymbol{y}\right)$, and $f(\boldsymbol{y}) = \int_x f(x,\boldsymbol{y})dx$. Note that for any $\boldsymbol{y}_0$, $f(x|\boldsymbol{y}_0)$ specifies an MTE potential for x.

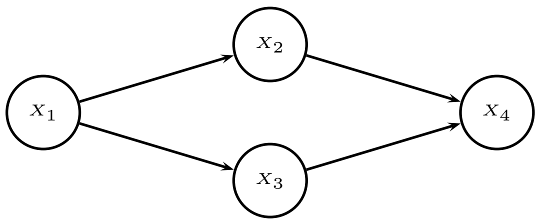


Fig. 1. A example of Bayesian network.

However, the problems come when the defined conditional densities are considered globally in a Bayesian network, in which the marginalization operation is necessary to perform inference. To illustrate the problem, consider the network structure in Fig. 1. Observe that the joint distribution is

$$\begin{aligned} f(x_1,x_2,x_3,x_4) &= f(x_1)f(x_2|x_1)f(x_3|x_1)f(x_4|x_2,x_3) \\ &= f(x_1)\,\frac{f(x_2,x_1)}{f(x_1)}\,\frac{f(x_3,x_1)}{f(x_1)}\,\frac{f(x_4,x_2,x_3)}{f(x_2,x_3)} \\ &= f(x_1,x_2)\frac{f(x_3,x_1)}{f(x_1)}\frac{f(x_4,x_2,x_3)}{f(x_2,x_3)}, \end{aligned} \tag{3}$$

but if the original conditional distributions are as in Equation (2), we find that the joint distribution in the network, shown in Equation (3), is not an MTE. Furthermore, standard inference algorithms, such as variable elimination [11], cannot be directly applied. For instance, if the first variable to eliminate is X_2, the operation to carry out would be

$$\int_{x_2} f(x_1,x_2)\frac{f(x_4,x_2,x_3)}{f(x_2,x_3)}dx_2,$$

which cannot be calculated in closed form if the potentials are as in Equation (2); the variable to integrate out appears in both the numerator and in the denominator.

When MTEs were first introduced [4], Moral *et al.* avoided these problems by defining the conditional MTE distribution as follows:

Definition 1. *Let $\boldsymbol{X}_1 = (\boldsymbol{Y}_1, \boldsymbol{Z}_1)$ and $\boldsymbol{X}_2 = (\boldsymbol{Y}_2, \boldsymbol{Z}_2)$ be two mixed random variables. We say that an MTE potential ϕ defined over $\Omega_{\boldsymbol{X}_1 \cup \boldsymbol{X}_2}$ is a* conditional MTE density *if for each $\boldsymbol{x}_2 \in \Omega_{\boldsymbol{X}_2}$, it holds that the restriction of ϕ to $\boldsymbol{x}_2$, $\phi^{R(\boldsymbol{X}_2 = \boldsymbol{x}_2)}$, is an MTE density for $\boldsymbol{X}_1$.*

In this paper we focus on conditional distributions of continuous variables with continuous parents. In our notation, Definition 1 is therefore equivalent to requiring that the conditional distribution must have the functional form

$$f(x|\boldsymbol{y}) = \alpha_0 + \sum_{j=1}^{m} \alpha_j \exp\left(\beta_j x + \boldsymbol{\gamma}_j^{\mathrm{T}} \boldsymbol{y}\right), (x, \boldsymbol{y}) \in \boldsymbol{I}_k, \tag{4}$$

where we will assume that $m < \infty$ in the following.

Moral *et al.* [4] noted that if one adopts the structural form of Equation (4), then specific requirements are in play to ensure that $f(x|\boldsymbol{y})$ is a conditional distribution. We will investigate one of these requirements in the following, namely that $\sum_k \int_{x:(x,\boldsymbol{y}) \in \boldsymbol{I}_k} f(x|\boldsymbol{y})\, dx = 1$, for all $\boldsymbol{y}$. As an example, consider Fig. 2, where the support for $f(x|y)$ is divided into 4 hypercubes $\boldsymbol{I}_1, \ldots, \boldsymbol{I}_4$, such that a specific MTE potential MTE$_k$ is defined for each $\boldsymbol{I}_k$. In this example, the requirement above implies that, e.g., $f(x|y = 0)$ ties the two MTE potentials MTE$_1$ and MTE$_2$ together, with the consequence that we cannot learn the MTE potentials separately.

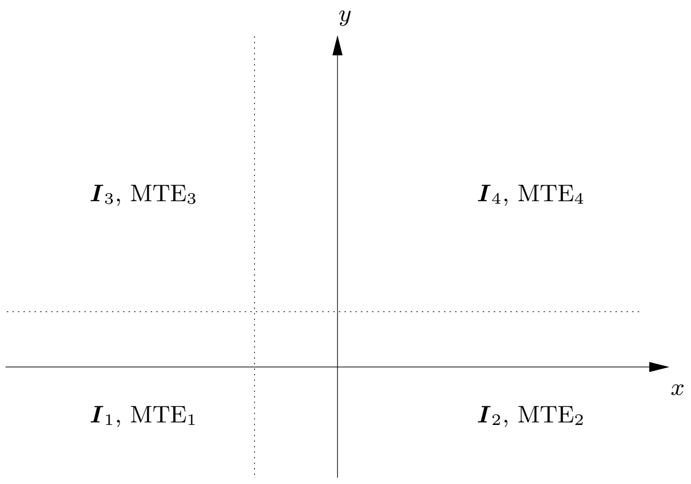

y
x
$\boldsymbol{I}_3$, MTE$_3$
$\boldsymbol{I}_4$, MTE$_4$
$\boldsymbol{I}_1$, MTE$_1$
$\boldsymbol{I}_2$, MTE$_2$

Fig. 2. The support for $f(x|y)$ is depicted, and divided into 4 hypercubes $\boldsymbol{I}_1$, $\boldsymbol{I}_2$, $\boldsymbol{I}_3$, and $\boldsymbol{I}_4$. An MTE potential MTE$_k$ is connected to each $\boldsymbol{I}_k$.

The effect of tying the parameters of different MTE potentials is a dramatic increase in the computational burden of learning conditional MTE distributions.

In this paper we will therefore assume *parameter independence* for simplicity. One consequence of this assumption is that $\int_x f(x|\boldsymbol{y})\,dx$ must be a constant w.r.t. $\boldsymbol{y}$, which corresponds to:

$$\begin{aligned}\frac{\partial}{\partial y_\ell}\int_x f(x|\boldsymbol{y})\,dx &= \frac{\partial}{\partial y_\ell}\int_x \sum_{j=1}^{m}\alpha_j \exp\left(\beta_j x + \boldsymbol{\gamma}_j^{\mathrm{T}}\boldsymbol{y}\right)\,dx\\ &= \sum_{j=1}^{m}\alpha_j\gamma_{j\ell}\exp\left(\boldsymbol{\gamma}_j^{\mathrm{T}}\boldsymbol{y}\right)\int_x \exp\left(\beta_j x\right)\,dx\\ &= 0.\end{aligned}$$

Thus, for all $(x_0, \boldsymbol{y}_0) \in \boldsymbol{I}_k$ where $f(\boldsymbol{y}_0) > 0$, we should have

$$\sum_{j=1}^{m}\alpha_j\gamma_{j\ell}\exp\left(\boldsymbol{\gamma}_j^{\mathrm{T}}\boldsymbol{y}_0\right)\int_x \exp\left(\beta_j x\right)\,dx = 0. \tag{5}$$

Now, fixate an ϵ-ball around $(x_0, \boldsymbol{y}_0)$ s.t. the ball is in $\boldsymbol{I}_k$ and in the support of $f(x, \boldsymbol{y})$. We are interested in varying $\boldsymbol{y}$ inside this ball. Then, Equation (5) gives rise to uncountably many constraints (one for each $\boldsymbol{y}$ in the ball), but where we only have $\mathcal{O}(m)$ parameters that can be used to adhere to the constraints. This over-specified system of equations can only be solved if $\boldsymbol{\gamma}_j = \boldsymbol{0}$ for all j (remember that $\alpha_j = 0$ is not a viable solution, since we need the density to have some mass allocated). Thus, if the conditional distribution functions are to follow Definition 1, and at the same time adhere to parameter independence, we must constrain the functional form of the conditional distribution to

$$f(x|\boldsymbol{y}) = \sum_{j=1}^{m}\alpha_j \exp\left(\beta_j x\right), (x, \boldsymbol{y}) \in \boldsymbol{I}_k. \tag{6}$$

Thus, the conditional MTE potential $f(x|\boldsymbol{y})$ is constant in $\boldsymbol{y}$ inside each hypercube $\boldsymbol{I}_1, \ldots, \boldsymbol{I}_K$, and the only effect of the conditioning variables $\boldsymbol{y}$ on x is through the definition of the hypercubes. This may at first glance seem like a serious limitation on the expressiveness of conditional MTE distributions, however, as we show in Section 5 this restricted form can still capture complex conditional distributions.

In summary, there are some conditions that apply to the specification of conditional MTE potentials in order to use MTEs with standard inference algorithms. Moral *et al.* [4] therefore defined that the conditional MTE distributions must be of the functional form given in Equation (4). However, this general form implies parameter dependence, making automatic learning intractable. One approach to solve this problem is to assume parameter independence, which restricts conditional MTE distributions to the form given in Equation (6). In this case, learning conditional distributions reduces to the following two tasks:

1. Finding the *split points*/hybercubes for the conditioning variables.
2. Learning the parameters of Equation (6) for each hybercube.

The latter item can be solved by algorithms for learning univariate MTE potentials [9]. We will turn to this issue shortly, and thereafter look at a method for learning the definition of the hypercubes from data.

4 Learning maximum likelihood distributions

4.1 The univariate MTE potentials

As already mentioned, MTE learning can be seen as a model selection problem where the number of exponential terms and the split points must be determined. In the following we briefly describe a learning procedure for the univariate case, the interested reader is referred to [9] for details.

When determining the number of exponential terms for a fixed interval $\boldsymbol{I}_k$, we iteratively add exponential terms (starting with the MTE potential having only a constant term) as long as the BIC score improves or until some other termination criterion is met. The learning algorithm regards the parameter learning (with fixed structure) as a constrained optimisation problem, and uses Lagrange multipliers to find the maximum likelihood parameters.

To determine the split points of the domain of the variable, a set of candidate split points is chosen. Since the BIC score will be the same for any split points defining the same partitioning of the data, it is not required to look at a set of possible splits that is larger than the set of midpoints between every two consecutive observations of Y. However, to reduce the computational complexity of the learning algorithm we consider a smaller set of potential split points in the current implementation: Each lth consecutive midpoint is selected, where l is chosen so that we get a total of 10 candidate split points. We use a myopic approach to select among the candidate split points, so that the one offering the highest gain in BIC score is selected at each iteration. This is repeated until no candidate split point increases the BIC score.

4.2 Learning conditional distributions

After having defined how to learn the parameters of the marginal distribution of a variable X from data, we will now consider learning the hybercubes (i.e., the split points) that define the conditioning part of the distribution (cf. Section 3). We will again use the BIC-score for model selection, and for simplicity we will start the discussion assuming that X has only one continuous parent Y. Recall that learning the conditional distribution $f(x|y)$ consists of two parts: *i*) Find the split points for Y, and *ii*) learn the parameters of the marginal distribution for X inside each interval.

The previous subsection reviewed how we can learn the marginal distribution for X, and we will now turn to finding split points for Y. As was the case when we learned the split points of the marginal distributions, we will also now learn the split points of the conditioning variable using a myopic strategy: When evaluating a candidate split point for a given interval $\boldsymbol{I}_k$, we compare the

BIC score when partitioning $\boldsymbol{I}_k$ into two (convex) sub-intervals with the score obtained with no partitioning, i.e., we employ a one step look-a-head that does not consider possible further refinements of the two sub-intervals.

Recall that we use $\mathcal{D} = [\boldsymbol{y}, \boldsymbol{x}]$ to describe the data, and the notation $\mathcal{D}(R)$ to denote the subset of data for which a restriction R is true. The skeleton of the learning algorithm can then be described as in Algorithm 1.

```
1  Function LearnConditionalsplit points
   Data: $\mathcal{D}$, $\omega_s$, $\omega_e$
   Result: splits
2  currentScore = Score($\mathcal{D}$, $\omega_s$, $\omega_e$);
3  newScore = $-\infty$;
4  for each potential split point $s_i$ do
5      tmpScore = Score($\mathcal{D}(y \leq s_i), \omega_s, s_i$) + Score($\mathcal{D}(y > s_i), s_i, \omega_e$) ;
6      if tmpScore > newScore then
7          newScore = tmpScore;
8          bestSplit = $s_i$;
9      end
10 end
11 if newScore > currentScore then
12     splits = [
13       LearnConditionalsplit points($\mathcal{D}(y \leq$ bestSplit$)$, $\omega_s$,
         bestSplit),
14       bestSplit,
15       LearnConditionalsplit points($\mathcal{D}(y >$ bestSplit$)$, bestSplit,
         $\omega_e$)];
16 else
17     splits =$\emptyset$;
18 end
19 return splits;
```

Algorithm 1: Skeleton for the algorithm that learns the split points for the conditioning variable y.

Algorithm 1 calls the external function `Score` to evaluate the different configurations of split points, both the current setting (in Line 2), and the one after adding a potential split point (in Line 5); note that the score function takes three or four parameters depending on whether we split the interval. One way of defining this score could be to fit a marginal MTE potential for each interval (looking only at data defined for the corresponding intervals), and afterwards calculate the BIC score for each of the intervals.

In Line 4, all potential split points for the conditioning variable are considered. Obviously, it suffices to only consider the observed values of the conditioning variable as potential split points. Note, however, that if t different split points are considered in Line 4, we will have to calculate the score $2t$ times in Line 5, and if we let t be equal to the number of observations in our database, the computational complexity of the algorithm will be intractable. We solve this in the same way as we did when learning marginal distributions, and select a fixed number of candidate split points. In the current implementation, we select the candidate splits by performing equal frequency-binning (with 10 bins) for each of the conditioning variables, and using the boundaries as candidate split points during learning.

The computational cost of calling the score-function in Line 5 may still be substantial though, and we have therefore considered alternatives ways of evaluating a split point. First of all, recall that the intuition behind defining a split point s for the conditioning variable y is that $f(x|y \leq s)$ and $f(x|y > s)$ are different (otherwise, we would reduce the BIC-score by introducing additional parameters for the new hypercube). By following this line of argument, we drop the calculations of the BIC-score in Line 5, and rather try to find good split points based on the Kolmogorov-Smirnov test [12] for determining whether two sets of data come from the same distribution. This modification is immediately accommodated in Algorithm 1 by simply replacing lines 2 and 5 with `CurrentScore=` $-\infty$ and `tmpScore=` $1 -$ `kstest`$(\mathcal{D}_j(y \leq s_i), \mathcal{D}_j(y > s_i))$, respectively; `kstest(`$\mathcal{D}_1$`,` $\mathcal{D}_2$`)` returns the p-value of the test that $\mathcal{D}_1$ and $\mathcal{D}_2$ come from the same distribution.

When working with more than one conditioning variable (i.e., $r > 1$) we need to select both a split variable and a split point. As before we do the selection greedily: iterate over all the conditioning variables, and for each variable find the best split point. After that select the conditioning variable having the best scoring split point. The recursive nature of the algorithm defines a binary tree in which each internal node is a conditioning variable and the arcs emanating from a node defines a partitioning of the associated interval (a so-called *probability tree*, [4]). Each leaf is associated with a univariate MTE distribution over x conditioned on the hybercube defined by the path from the root to the leaf. The final algorithm is similar to Algorithm 1 except that for each conditioning variable we also need to iterate over lines 4–10 and pick the best scoring variable to split on; the full specification has been left out due to space restrictions.

5 Examples

Our first example shows how our Algorithms learn a conditional Gaussian distribution, and in particular we examine the effect of the size of the dataset we learn from. We generated datasets of size 30, 50, 100, 250, and 1000 from the distribution

$$\begin{bmatrix} x \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 5 & 2 \\ 2 & 1 \end{bmatrix}\right), \tag{7}$$

and focused our attention on learning the conditional distribution $f(x|y)$. The potential split points for the conditioning variable y were defined by using the split points in an 11-bin equal-frequency histogram, meaning that 10 candidate split points were considered in each learning situation. As previously described, we used the results of the Kolmogorov-Smirnov tests to prioritise these candidate split points, and the BIC-score to determine whether or not a given candidate split point is to be accepted. The distribution over x for each y-interval was selected in order to maximise the BIC score.

The obtained results for all datasets are given in Fig. 3 part (a) to (e) respectively, and should be compared to the exact conditional density given in Fig. 3 (f); for further comparison, a scatter plot of the dataset with 250 cases in shown in Fig. 4. The results are promising: We see a strong resemblance between the target distribution and the gold standard distribution for all data sizes. We can also see the effect of using the BIC-score to determine whether or not a candidate split point for y should be accepted: When only a few splits are employed for the smaller datasets, all candidate split points for y were used when learning from the largest dataset. Finally, it is also worth noticing that the support of x is never divided into subintervals in these runs. This is to be expected, considering that these MTE potentials are typically learned from only about 25 observations each.

For illustration, we also ran the algorithm using a data set containing 250 configurations sampled from the distribution

$$\begin{bmatrix} x \\ y_1 \\ y_2 \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix}, \begin{bmatrix} 5 & 2 & 2 \\ 2 & 1 & 1 \\ 2 & 1 & 2 \end{bmatrix} \right). \tag{8}$$

The result of the learning can be seen in Fig. 5, where we have skipped the specification of the marginal MTE distributions in the leaves. Observe that the tree is more fine-grained around the mean of Y_1 compared to the parts of the interval with smaller support in the distribution (i.e., more data is available to capture the refinement). In particular, note also that the algorithm conditions on Y_2 is this interval.

6 Conclusions

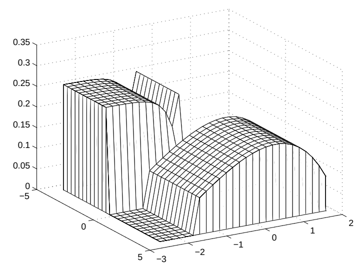
In this paper we have investigated two alternatives for the definition of conditional MTE densities. We have shown that only the most restrictive one is compatible with standard efficient algorithms for inference in Bayesian networks.

We have also shown how the induction of this kind of conditional densities can be approached from the point of view of maximum likelihood estimation, including model selection for determining the partitioning of the domain of the conditioning variables based on the BIC score.

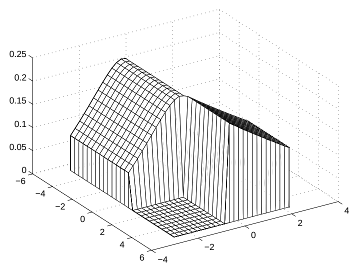
Our future work on this subject will include the definition of a structural learning algorithm based on the tools proposed on this paper and in [9].

Acknowledgments

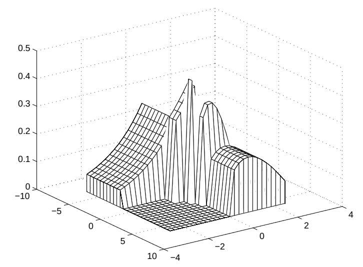
This work has been partially supported by the Spanish Ministry of Science and Innovation under project TIN2007-67418-C03-02 and by FEDER funds.



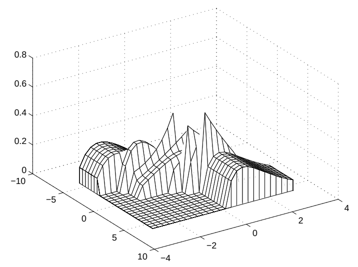
(a) 30 cases



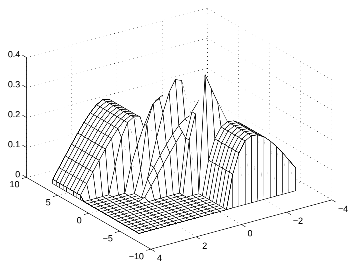
(b) 50 cases



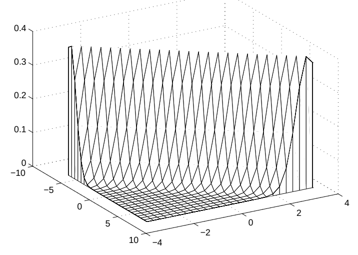
(c) 100 cases



(d) 250 cases



(e) 1000 cases



(f) Gold standard distribution

Fig. 3. The plots show the results of learning from 30, 50, 100 and 250 cases respectively. The gold-standard distribution, shown in part (f), is the conditional Gaussian distribution $f(x|y)$ derived from the joint distribution in Equation (7). The Kolmogorov-Smirnov tests were used to find the split points of the conditioning variable (axis on the left-hand side), whereas the marginal distributions were obtained by maximisation of the BIC score.

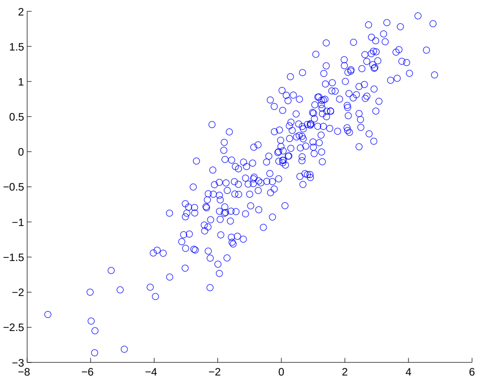


Fig. 4. A scatter of the data set with 250 cases sampled from the joint distribution in Equation (7). Note, in particular, the few cases sampled from the tails of the distribution.

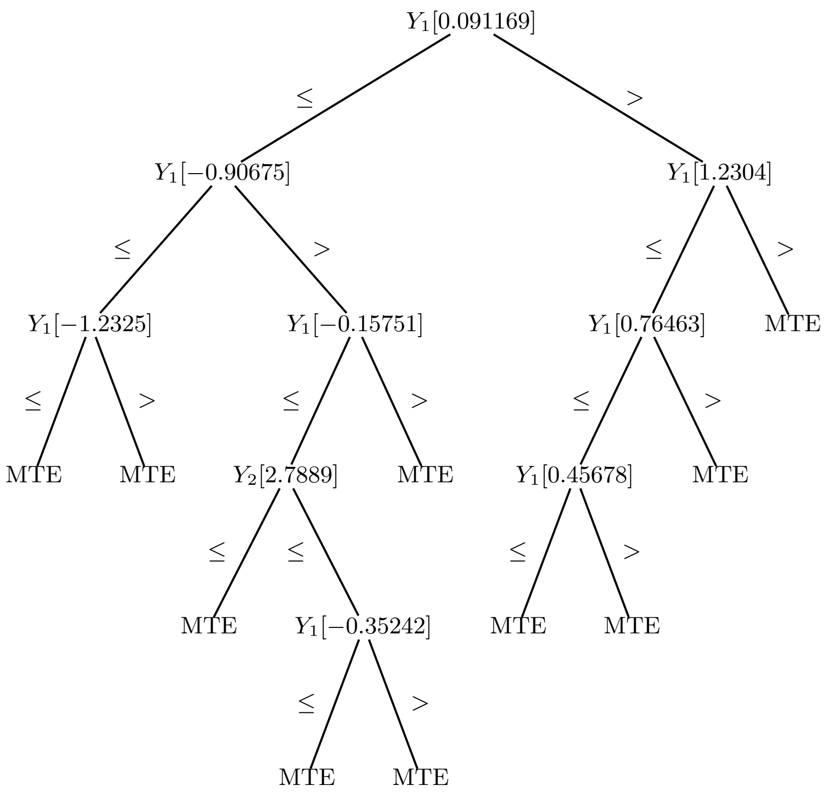


Fig. 5. A binary tree structure representing the conditional distribution learned from 250 cases sampled from the distribution in Equation (8).

References

1. Shenoy, P., Shafer, G.: Axioms for probability and belief function propagation. In Shachter, R., Levitt, T., Lemmer, J., Kanal, L., eds.: Uncertainty in Artificial Intelligence 4, North Holland, Amsterdam (1990) 169–198
2. Friedman, N., Goldszmidt, M.: Discretizing continuous attributes while learning bayesian networks. In: Proceedings of the Thirteenth International Conference on Machine Learning. (1996) 157–165
3. Kozlov, D., Koller, D.: Nonuniform dynamic discretization in hybrid networks. In Geiger, D., Shenoy, P., eds.: Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence, Morgan & Kaufmann (1997) 302–313
4. Moral, S., Rumí, R., Salmerón, A.: Mixtures of truncated exponentials in hybrid Bayesian networks. In: ECSQARU'01. Lecture Notes in Artificial Intelligence. Volume 2143. (2001) 135–143
5. Cobb, B., Shenoy, P., Rumí, R.: Approximating probability density functions with mixtures of truncated exponentials. Statistics and Computing **16** (2006) 293–308
6. Rumí, R., Salmerón, A., Moral, S.: Estimating mixtures of truncated exponentials in hybrid Bayesian network. Test **15** (2006) 397–421
7. Romero, V., Rumí, R., Salmerón, A.: Learning hybrid Bayesian networks using mixtures of truncated exponentials. International Journal of Approximate Reasoning **42** (2006) 54–68
8. Schwarz, G.: Estimating the dimension of a model. Annals of Statistics **6** (1978) 461–464
9. Langseth, H., Nielsen, T., Rumí, R., Salmerón, A.: Parameter estimation in mixtures of truncated exponentials. In Jaeger, M., Nielsen, T., eds.: Proceedings of the 4th European Workshop on Probabilistic Graphical Models (PGM'08). (2008) 169–176
10. Moral, S., Rumí, R., Salmerón, A.: Approximating conditional MTE distributions by means of mixed trees. In: ECSQARU'03. Lecture Notes in Artificial Intelligence. Volume 2711. (2003) 173–183
11. Zhang, N., Poole, D.: Exploiting causal independence in Bayesian network inference. Journal of Artificial Intelligence Research **5** (1996) 301–328
12. DeGroot, M.H.: Optimal Statistical Decisions. McGraw-Hill (1970)