# Parameter Estimation and Model Selection for Mixtures of Truncated Exponentials

Helge Langseth[a], Thomas D. Nielsen[b], Rafael Rumí[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 Statistics and Applied Mathematics, University of Almería, Almería (Spain)*

**Abstract**

Bayesian networks with mixtures of truncated exponentials (MTEs) support efficient inference algorithms and provide a flexible way of modeling hybrid domains (domains containing both discrete and continuous variables). On the other hand, estimating an MTE from data has turned out to be a difficult task, and most prevalent learning methods treat parameter estimation as a regression problem. The drawback of this approach is that by not directly attempting to find the parameter estimates that maximize the likelihood, there is no principled way of performing subsequent model selection using those parameter estimates. In this paper we describe an estimation method that directly aims at learning the parameters of an MTE potential following a maximum likelihood approach. Empirical results demonstrate that the proposed method yields significantly better likelihood results than existing regression-based methods. We also show how model selection, which in the case of univariate MTEs amounts to partitioning the domain and selecting the number of exponential terms, can be performed using the BIC-score.

## 1. Introduction

Domains involving both discrete and continuous variables represent a challenge to Bayesian networks. The main difficulty is to find a representation of the joint distribution of the continuous and discrete variables that supports an efficient implementation of the usual inference operations over Bayesian networks (like those found in junction tree-based algorithms for exact inference). Computationally, exact inference algorithms require that the joint distribution over the variables of the domain is from a distribution-class that is closed under addition and multiplication. The simplest way of obtaining such a distribution is to perform a *discretization* of the continuous variables (Friedman and Goldszmidt, 1996; Kozlov and Koller, 1997). Mathematically, this amounts to approximating the density function of every continuous variable by a step-function. However, discretization of variables can lead to a dramatic loss in precision, which is one of the reasons why other approaches have received much attention recently. The

*Preprint submitted to Elsevier* *May 28, 2009*

mixtures of truncated exponentials (MTE) framework (Moral et al., 2001) has also received increasing interest over the last few years. 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 addition and multiplication, inference in an MTE network can be performed efficiently using the Shafer-Shenoy architecture (Shafer and Shenoy, 1990).

Cobb et al. (2006) empirically showed that many distributions can be approximated accurately by means of an MTE distribution, and they argue that this makes the MTE framework very attractive for Bayesian network models. Nevertheless, data-driven learning methods for MTE networks have received only little attention. In this context, focus has mainly been directed towards parameter estimation, where the most prevalent methods look for the MTE parameters minimizing the mean squared error w.r.t. a kernel density estimate of the data (Romero et al., 2006).

Although the least squares estimation procedure can yield a good MTE model in terms of generalization properties, there is no guarantee that the estimated parameter values will be close to the maximum likelihood (ML) parameters. This has a significant impact when considering more general problems such as model selection and structural learning, as many standard score functions for model selection, including the Bayesian information criterion (BIC) (Schwarz, 1978), assume ML parameter estimates to be available.

In this paper we propose a new parameter estimation procedure for univariate MTE potentials. The procedure directly aims at estimating the ML parameters for an MTE density, and we show how to utilize the learned ML estimates for model selection using the BIC score. The proposed learning method is empirically compared to the least squares estimation method described by Romero et al. (2006), and it is shown that it offers a significant improvement in terms of likelihood as well as in generalization ability.

The method described in this paper is a first step towards a general maximum likelihood-based approach for learning Bayesian networks with MTE potentials. Thus, our objective is solely to demonstrate that maximum likelihood estimators for MTE distributions can be found, and show how these estimators can be utilised for model selection. We will therefore prefer simple and robust methods over state-of-the-art optimisation techniques that can be harder to understand, implement, and examine. Furthermore, we shall only hint at some of the complexity problems that are involved in learning general MTE potentials. Learning MTE potentials using more efficient techniques is left as a topic for future research.

## 2. Preliminaries

Throughout this paper, random variables will be denoted by capital letters, and their values by lowercase letters. In the multi-dimensional case, boldfaced characters will be used. The domain of the variables $\mathbf{X}$ is denoted by $\Omega_{\mathbf{X}}$. The

MTE model is defined by its corresponding potential and density as follows (Moral et al., 2001):

**Definition 1.** (MTE potential) *Let* $\mathbf{X}$ *be a mixed* $n$*-dimensional random vector. Let* $\mathbf{W} = (W_1, \ldots, W_d)^{\mathrm{T}}$ *and* $\mathbf{Z} = (Z_1, \ldots, Z_c)^{\mathrm{T}}$ *be the discrete and continuous parts of* $\mathbf{X}$*, respectively, with* $c + d = n$*. We say that a function* $f : \Omega_{\mathbf{X}} \mapsto \mathbb{R}_0^+$ *is a* Mixture of Truncated Exponentials (MTE) potential *if for each fixed value* $\mathbf{w} \in \Omega_{\mathbf{W}}$ *of the discrete variables* $\mathbf{W}$*, the potential over the continuous variables* $\mathbf{Z}$ *is defined as:*

$$f(\mathbf{z}) = a_0 + \sum_{i=1}^{m} a_i \exp\left\{\mathbf{b}_i^{\mathrm{T}} \mathbf{z}\right\}, \tag{1}$$

*for all* $\mathbf{z} \in \Omega_{\mathbf{Z}}$*, where* $a_i \in \mathbb{R}$ *and* $\mathbf{b}_i \in \mathbb{R}^c$*,* $i = 1, \ldots, m$*. We also say that* $f$ *is an MTE potential if there is a partition* $D_1, \ldots, D_k$ *of* $\Omega_{\mathbf{Z}}$ *into hypercubes and in each* $D_\ell$*,* $f$ *is defined as in Equation (1). An MTE potential is an* MTE density *if it integrates to 1.*

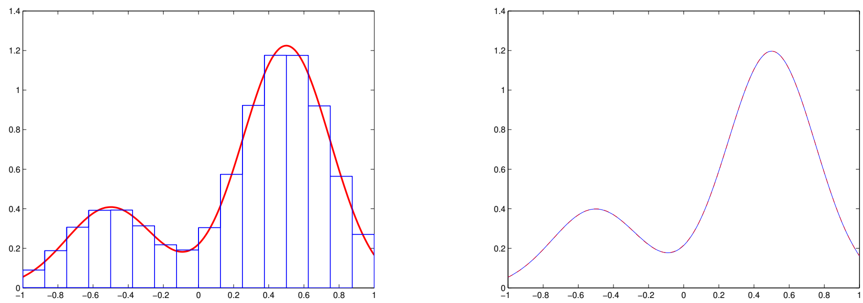
In the remainder of this paper we shall focus on estimating the parameters for a univariate MTE density. Not surprisingly, the proposed methods also immediately generalize to the special case of conditional MTEs having only discrete conditioning variables.

## 3. Expressiveness of the MTE models

In this section we will explore the expressiveness of the MTE framework, with the aim of showing that any univariate distribution function can be approximated arbitrarily well by an MTE potential. We will tie our argument to the example in Figure 1, but the results obtained are general in nature.

Consider first the left panel of Figure 1, where the target distribution of our example is given by the solid line. The target distribution, $f(x)$, is a mixture of two Gaussian distributions, one centred at $-\frac{1}{2}$ and the other at $\frac{1}{2}$. Both Gaussian distributions have standard deviation 1, and the mixture weights are .25 and .75, respectively. The left panel of Figure 1 also shows how a standard discretization scheme can be utilised to approximate $f(x)$. Let $\hat{f}_D(x \mid k)$ be the approximation of $f(x)$ obtained by dividing the range of $X$ into $k$ equally sized intervals. Note that $\hat{f}_D(x \mid k)$ requires $k-1$ parameters to be fully specified, namely the amount of mass allocated to each interval except the last one. It is obvious that if we measure the error of $\hat{f}_D(x \mid k)$ as $\int_{x=a}^{b} \left(\hat{f}_D(x \mid k) - f(x)\right)^2 \mathrm{d}x$, then this error can be made arbitrarily small by increasing $k$. Since the class of MTE distributions contains all distributions obtainable by discretization, we can also approximate any $f(x)$ arbitrarily well using MTEs. This representation may not be optimal, though. In the left panel of Figure 1, 15 parameters were used to obtain the approximation, but it is still rather crude and the representational power of MTEs are not fully utilised.

The right panel of Figure 1 explore a different strategy for approximating $f(x)$, as we here define an approximation by increasing the number of exponential terms, without dividing the support of the distribution into intervals. We will denote approximations generated in this way by $\hat{f}_e(x \mid m)$, and for a given set of parameters $\boldsymbol{\alpha}$ we define $\hat{f}_e(x \mid m, \boldsymbol{\alpha}) = \sum_{s=-m}^{m} \alpha_s \exp(s\, x)$. In the reminder of this section we investigate approximations of the type $\hat{f}_e(x \mid m, \boldsymbol{\alpha})$ and we will show that this strategy for approximating $f(x)$ can also be made arbitrarily accurate (wrt. our error measure) simply by increasing $m$. The right panel of Figure 1 shows this visually: We are again using 15 parameters, but the quality of the approximation is now so good that it is not possible to visually distinguish the true distribution from the approximation.



(a) Approximation improved by increasing no. intervals

(b) Approximation improved by increasing no. exponential terms

Figure 1: Two different strategies for approximating a distribution function. The gold standard model is in this case a mixture of two Gaussians, one centred at $-\frac{1}{2}$, the other at $\frac{1}{2}$.

We will make this argument by first restating a basic result from linear algebra, and then show how this generalises to our setting. Let us start by considering an $n$-dimensional real vector $\mathbf{z} \in \mathbb{R}^n$. Let $\{\mathbf{e}_1, \ldots, \mathbf{e}_k\}$ be a set of orthogonal basis vectors in $\mathbb{R}^n$ $(k < n)$, and consider the task of approximating $\mathbf{z}$ by a vector in the span of the basis vectors. Let $\langle \mathbf{x}, \mathbf{y} \rangle$ denote the inner product between two vectors $\mathbf{x}$ and $\mathbf{y}$; when both $\mathbf{x}$ and $\mathbf{y}$ are in $\mathbb{R}^n$ we use $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^{\mathrm{T}}\mathbf{y}$. It is well-known that the least-squares solution to this approximation problem is to find the projection of $\mathbf{z}$ onto the space spanned by the basis vectors, i.e., to choose

$$\hat{\mathbf{z}} \leftarrow \sum_{j=1}^{k} \frac{\langle \mathbf{e}_j, \mathbf{z} \rangle}{\sqrt{\langle \mathbf{e}_j, \mathbf{e}_j \rangle}} \cdot \mathbf{e}_j. \tag{2}$$

Next, we generalize this result from approximations in $\mathbb{R}^n$ to approximations in a space containing *functions*, with the idea that we can approximate a function $f$ as a linear combination of *basis functions*. In particular, we will consider the space of all functions that are squared integrable on an interval $[a, b]$. This space is often denoted $L_2[a, b]$, so for a real function $f$ we have that $f \in L_2[a, b]$ if and only if $\int_{x=a}^{b} f(x) \cdot f(x)\,\mathrm{d}x < \infty$. To find an analogue to the projections in Equation (2) we must define the inner product between two functions $f$ and $g$, and in $L_2[a, b]$ this is done as $\langle f, g\rangle = \int_a^b f(x) \cdot g(x)\,dx$. Furthermore, we say that two functions $f$ an $g$ are orthogonal if and only if $\langle f, g\rangle = 0$.

Focus on $L_2[0, 2\pi]$ and consider the task of finding the Fourier series approximation of a function $f$. This amounts to approximating $f$ by a sum of trigonometric functions, i.e., it is a solution to our original approximation problem, where the functions $\{1, \sin(x), \cos(x), \sin(2x), \cos(2x), \sin(3x), \ldots\}$ take the role of the orthogonal[1] basis-vectors $\{\mathbf{e}_1 \ldots \mathbf{e}_k\}$ used when making approximations in $\mathbb{R}^n$. Recall that the Fourier series approximation of $f$ can be written as

$$\hat{f}(x) \leftarrow \sum_{j=1}^{k} \frac{\langle e_j, f\rangle}{\sqrt{\langle e_j, e_j\rangle}} \cdot e_j(x). \tag{3}$$

This gives an operational description of how to approximate any $f \in L_2[0, 2\pi]$ by a sum of trigonometric functions.

The last step in our argument is to recall that the approach of Equation (3) is valid also when the trigonometric functions are replaced by other orthogonal basis functions; in this case Equation (3) is called a *Generalized Fourier series*. Since we look for MTE approximations, we are interested in the span of the exponential functions $\{1, \exp(x), \exp(-x), \exp(2x), \exp(-2x), \ldots\}$. The exponential functions are *dense* in $L_2[a, b]$, loosely meaning that any function $f$ can be approximated arbitrarily well by a linear combination of exponential functions. Unfortunately, the specified exponential functions are not orthogonal, so an orthogonalisation process (also known as a Gram-Schmidt process) must be conducted before the generalised Fourier coefficients can be found. The approximation of Figure 1 is made in this way, starting from the 15 functions $\{\exp(-7x), \ldots, \exp(7x)\}$.

When we in the following look at ways to learn MTEs, we are trying to find a balance between the number of split points and the number of exponential terms: we aim for an approximation, where the support of the density may be divided into "a few" intervals, each interval containing "a few" exponential terms. Our goal is therefore to find a parameterization that is close to minimal, and that at the same time offers robust techniques for learning the parameters from data. This will be the topic of the remainder of the paper.

## 4. Maximum Likelihood Parameter Estimation for Univariate MTEs

The problem of estimating a univariate MTE density from data can be divided into three tasks: *i*) Partition the domain of the variable into disjoint

[1] Recall that we have $\int_{x=0}^{2\pi} \sin(nx)\cos(mx)\,\mathrm{d}x \equiv 0$ for integer $n$,$m$, and that if we also assume that $n \neq m$ we get $\int_{x=0}^{2\pi} \sin(nx)\sin(mx)\,\mathrm{d}x = \int_{x=0}^{2\pi} \cos(nx)\cos(mx)\,\mathrm{d}x \equiv 0$.

intervals, *ii*) determine the number of exponential terms for each interval, and *iii*) estimate the parameters for a given interval and a fixed number of exponential terms. At this point we will concentrate on the estimation of the parameters, assuming that the split points are known, and that the number of exponential terms is fixed. We will return to the two remaining tasks in Section 5.

We start this section by introducing some notation. Consider a random variable $X$ with density function $f(x)$ and assume that the support of $f(x)$ is divided into $M$ intervals $\{\Omega_i\}_{i=1}^{M}$. Focus on one particular interval $\Omega_m$. As a target density for $x \in \Omega_m$ we will consider an MTE with 2 exponential terms:

$$f(x|\boldsymbol{\theta}_m) = k_m + a_m e^{b_m x} + c_m e^{d_m x},\ x \in \Omega_m. \tag{4}$$

This function has 5 parameters, namely $\boldsymbol{\theta}_m = (k_m, a_m, b_m, c_m, d_m)^{\mathrm{T}}$. For notational convenience we may sometimes drop the subscript $m$ when clear from the context.

### *4.1. The Likelihood Landscape*

For an MTE of the form given in Equation (4), the shape of the likelihood landscape is not well-known. To investigate this, we sampled two datasets of 50 and 1000 samples from the distribution

$$f(x) = \begin{cases} \frac{5}{2(\exp(5)-1)}\exp(5x) - \frac{5}{2(\exp(-5)-1)}\exp(-5x) & \text{if } x \in [-1,1], \\ 0 & \text{otherwise.} \end{cases}$$

The profile likelihood of the two datasets are shown in Figure 2, where the value at the point $(b_0, d_0)$ is given as $\max_{k,a,c} \prod_i \{k + a\exp(b_0 \cdot x_i) + c\exp(d_0 \cdot x_i)\}$ and the product is over all samples in the data set.

From the figure we see that the profile likelihood is symmetric around the line $b = d$, i.e. that the profile likelihood of a sample at point $(b_0, d_0)$ is identical to the one at $(d_0, b_0)$. The consequence is that the parameters of an MTE are not identifiable in a strict sense. This is not surprising, as MTE models are generalized mixture models ("generalized" because we do not demand the weights to be positive and sum to one). Furthermore, we see that for the relatively small dataset of 50 samples, the profile likelihood is fairly flat, so finding a local maxima using a standard hill-climbing approach may be very slow. Furthermore, the profile likelihood is multi-modal. On the other hand, the profile likelihood is peaked for the large sample. To be successful in learning MTEs, an algorithm must therefore be able to handle "flat" multi-modal likelihood landscapes as well as very peaked likelihood landscapes.

### *4.2. Parameter Estimation by Maximum Likelihood*

Assume that we have a sample $\mathbf{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ and that $n_m$ of the $n$ observations are in $\Omega_m$. To ensure that the overall parameter-set is a maximum likelihood estimate for $\boldsymbol{\Theta} = \cup_m \boldsymbol{\theta}_m$, it is required that

$$\int_{x\in\Omega_m} f(x|\boldsymbol{\theta}_m)\, dx = n_m/n. \tag{5}$$

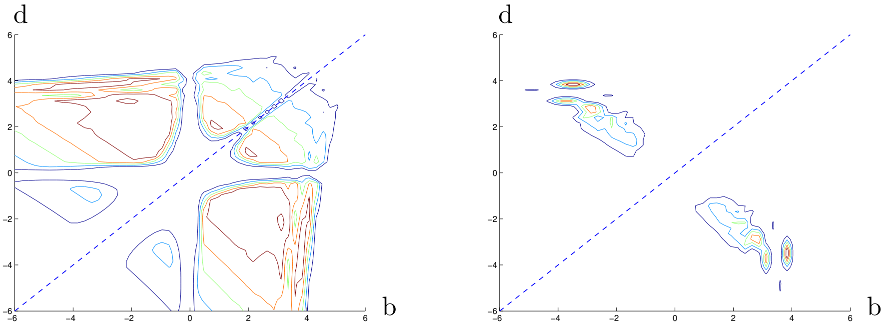



Figure 2: Profile likelihood of example data sampled from a known distribution. The left panel shows the results using 50 samples, the right plot gives the result for 1000 samples.

Given this normalization, we can fit the parameters for each interval $\Omega_m$ separately, i.e., the parameters in $\boldsymbol{\theta}_m$ are optimized independently of those in $\boldsymbol{\theta}_{m'}$. Based on this observation, we shall only describe the learning procedure for *a fixed interval* $\Omega_m$, since the generalization to the whole support of $f(x)$ is immediate.

Assume now that the target density is as given in Equation (4), in which case the likelihood function for a sample $\mathbf{x}$ is

$$L(\boldsymbol{\theta}_m|\mathbf{x}) = \prod_{i:x_i\in\Omega_m} \left\{k_m + a_m e^{b_m x_i} + c_m e^{d_m x_i}\right\}. \tag{6}$$

To find a closed-form solution for the maximum likelihood estimators, we need to differentiate Equation (6) wrt. the different parameters and set the results equal to zero. To exemplify, we perform this exercise for $b_m$, and obtain

$$\frac{\partial L(\boldsymbol{\theta}_m|\mathbf{x})}{\partial b_m} = \sum_{i:x_i\in\Omega_m} \left\{ \frac{\partial L(\boldsymbol{\theta}_m|x_i)}{\partial b_m} \prod_{j:x_j\in\Omega_m, j\neq i} L(\boldsymbol{\theta}_m|x_j) \right\}$$
$$= a_m x_i \sum_{i:x_i\in\Omega_m} e^{b_m x_i} \left\{ \prod_{j:x_j\in\Omega_m, j\neq i} \left(k_m + a_m e^{b_m x_j} + c_m\, e^{d_m\, x_j}\right) \right\}. \tag{7}$$

Unfortunately, Equation (7) is non-linear in the unknown parameters $\boldsymbol{\theta}_m$. Furthermore, both the number of terms in the sum as well as the number of terms inside the product operator grows as $O(n_m)$; thus, the maximization of the likelihood becomes increasingly difficult as the number of observations rise.

Alternatively, one might consider maximizing the logarithm of the likelihood, or more specifically a *lower bound* for the likelihood using Jensen's inequality. By assuming that $k_m > 0$, $a_m > 0$ and $c_m > 0$ we have

$$
\begin{aligned}
&\log\left(L(\boldsymbol{\theta}_m|\mathbf{x})\right) = \sum_{i:x_i\in\Omega_m} \log\left(k_m + a_m\, \exp(b_m\, x_j) + c_m\, \exp(d_m\, x_j)\right) \\
&\geq \sum_{i:x_i\in\Omega_m} \log\left(k_m\right) + \sum_{i:x_i\in\Omega_m} \log\left(a_m\, \exp(b_m\, x_j)\right) + \sum_{i:x_i\in\Omega_m} \log\left(c_m\, \exp(d_m\, x_j)\right) \\
&= n_m \left[\log(k_m) + \log(a_m) + \log(c_m)\right] + (b_m + d_m) \sum_{i:x_i\in\Omega_m} x_i,
\end{aligned}
\tag{8}
$$

and the idea would then be to maximize the lowerbound of Equation (8) to push the likelihood upwards (following the same reasoning underlying the EM algorithm (Dempster et al., 1977) and variational methods (Jordan et al., 1999)). Unfortunately, though, restricting $k_m$, $a_m$ and $c_m$ to be positive enforces too strict a limitation on the expressiveness of the distributions we learn.

Another possibility would be to use a modified version of the EM algorithm able to handle negative components. For example, Farag et al. (2004) consider the estimation of linear combinations of Gaussians. However, in each iteration of their procedure, the parameters are optimized by Lagrange maximization, which in the case of our likelihood function (Equation (6)) does not provide any simplification.

The main problem when trying to apply the EM algorithm to MTE densities is to find a formulation of the problem, where the inclusion of latent variables simplifies the estimation when the values of the latent variables are fixed. However, if this conditional density is also of MTE shape, the maximisation of the conditional expectation with respect to the parameters in each iteration would again be as difficult as the original problem. In this case, even the use of a flexible implementation like Monte Carlo EM (see, for instance Tanner (1996)) does not provide any simplification, as approximating the integral associated with the conditional expectation by simulation produces a sum of MTE functions, which again is as difficult to optimize as the original likelihood.

Instead, we opt for an approximate solution obtained by solving the likelihood equations by numerical means. The proposed method for maximizing the likelihood is based on the observation that maximum likelihood estimation for MTEs can be seen as a constrained optimization problem, where constraints are introduced to ensure that both $f(x|\boldsymbol{\theta}_m) \geq 0$, for all $x \in \Omega_m$, and that Equation (5) is fulfilled. A natural framework for solving this is the Lagrange multipliers, but since solving the Lagrange equations are inevitably at least as difficult as solving the unconstrained problem, this cannot be done analytically. In our implementation we have settled for a numerical solution based on Newton's method; this is described in detail in Section 4.2.2. However, it is well-known that Newton's method is quite sensitive to the initialization-values, meaning that if we initialize a search for a solution to the Lagrange equations from a parameter-set far from the optimal values, it will not necessarily converge to a useful solution. Thus, we need a simple and robust procedure for initializing Newton's method, and this is described next.

*4.2.1. Naïve Maximum Likelihood for MTE Distributions*

The general idea of the naïve approach is to iteratively update the parameter estimates until convergence. More precisely, this is done by iteratively tuning *pairs* of parameters, while the other parameters are kept fixed. We do this in a round-robin manner, making sure that all parameters are eventually tuned. Denote by $\hat{\boldsymbol{\theta}}^t = (k^t, a^t, b^t, c^t, d^t)^{\mathrm{T}}$ the parameter values after iteration $t$ of this iterative scheme. Algorithm 4.1 is a top-level description of this procedure, where steps 4 and 5 correspond to the optimization of the shape-parameters and steps 6 and 7 distribute the mass between the terms in the MTE potential (the different steps are explained below).

**Algorithm 4.1** The algorithm learns a "rough" estimate of the parameters of an MTE with two exponential terms.

1: **function** NAÏVE_MTE($\mathbf{x}$)
2: Initialize $\hat{\boldsymbol{\theta}}^0$; $t \leftarrow 0$.
3: **repeat**
4: $(a', b') \leftarrow \arg\max_{a,b} L(k^t, a, b, c^t, d^t \mid \mathbf{x})$
5: $(c', d') \leftarrow \arg\max_{c,d} L(k^t, a', b', c, d \mid \mathbf{x})$
6: $(k', a') \leftarrow \arg\max_{k,a} L(k, a, b', c', d' \mid \mathbf{x})$
7: $(k', c') \leftarrow \arg\max_{k,c} L(k, a', b', c, d' \mid \mathbf{x})$
8: $\boldsymbol{\theta}^{t+1} \leftarrow (k', a', b', c', d')^{\mathrm{T}}$
9: $t \leftarrow t + 1$
10: **until** convergence
11: **return** $\hat{\boldsymbol{\theta}}^t$

For notational convenience we shall define the auxiliary function $p(s,t) = \int_{x\in\Omega_m} s\exp(tx)\,dx$; $p(s,t)$ is the integral of the exponential function over the interval $\Omega_m$. Note, in particular, that $p(s,t) = s \cdot p(1,t)$, and that $p(1,0) = \int_{x\in\Omega_m} dx$ is the length of the interval $\Omega_m$. The first step above is initialization. In our experiments we have chosen $b^0$ and $d^0$ as $+1$ and $-1$, respectively. The parameters $k^0$, $a^0$, and $c^0$ are set to ensure that each of the three terms in the integral of Equation (5) contribute with equal probability mass, i.e.,

$$\begin{aligned} k^0 &\leftarrow \frac{n_m}{3n \cdot p(1,0)}, \\ a^0 &\leftarrow \frac{n_m}{3n \cdot p(1,b^0)}, \\ c^0 &\leftarrow \frac{n_m}{3n \cdot p(1,d^0)}. \end{aligned}$$

Iteratively improving the likelihood under the constraints is actually quite simple as long as the parameters are considered in pairs. Consider Step 4 above, where we optimize $a$ and $b$ under the constraint of Equation (5) while keeping the other parameters ($k^t$, $c^t$, and $d^t$) fixed. Observe that if Equation (5) is to be satisfied after this step we need to make sure that $p(a', b') = p(a^t, b^t)$.

Equivalently, there is a functional constraint between the parameters that we enforce by setting $a' \leftarrow p(a^t, b^t)/p(1, b')$. Optimizing the value for the pair $(a, b)$ is now simply done by line-search, where only the value for $b$ is considered:

$$b' = \arg\max_b L\left(k, \frac{p(a^t, b^t)}{p(1, b)}, b, c^t, d^t \,\middle|\, \mathbf{x}\right).$$

Note that at the same time we choose $a' \leftarrow p(a^t, b^t)/p(1, b')$. A similar procedure is used in Step 5 to find $c'$ and $d'$.

Steps 6 and 7 utilize the same idea, but with a different normalization equation. We only consider Step 6 here, since the generalization is immediate. For this step we need to make sure that $\int_{x\in\Omega_m} k + a\exp(b'\,x)\,dx = \int_{x\in\Omega_m} k^t + a^t\exp(b'\,x)\,dx$, for any pair of parameter candidates $(k, a)$. By rephrasing, we find that this is obtained if we insist that $k' \leftarrow k^t - p(a' - a^t, b')/p(1, 0)$. Again, the constrained optimization of the pair of parameters can be performed using line-search in one dimension (and let the other parameter be adjusted to keep the total probability mass constant).

Note that Steps 4 and 5 do not move "probability mass" between the three terms in Equation (4), these two steps only fit the shape of the two exponential functions. On the other hand, Steps 6 and 7 assume the shape of the exponentials fixed, and proceed by moving "probability mass" between the three terms in the sum of Equation (4).

*4.2.2. Refining the Initial Estimate*

The parameter estimates returned by the line-search method can be further refined by using these estimates to initialize a nonlinear programming problem formulation of the original optimization problem. In this formulation, the function to be maximized is again the log-likelihood of the data, subject to the constraints that the MTE potential should be nonnegative, and that

$$g_0(\mathbf{x}, \boldsymbol{\theta}) \equiv \int_{x\in\Omega_m} f(x \,|\, \boldsymbol{\theta})\,dx - \frac{n_m}{n} = 0.$$

Ideally the nonnegative constraints should be specified for all $x \in \Omega_m$, but since this is not feasible we only encode that the function should be nonnegative in the endpoints $\omega_s$ and $\omega_e$ of the interval (we shall return to this issue later). Thus, we arrive at the following formulation:

$$\begin{aligned}
\text{Maximize } & \log L(\boldsymbol{\theta} \,|\, \mathbf{x}) = \sum_{i:x_i\in\Omega_m} \log L(\boldsymbol{\theta} \,|\, x_i) \\
\text{Subject to } & g_0(\mathbf{x}, \boldsymbol{\theta}) = 0, \\
& f(\omega_s \,|\, \boldsymbol{\theta}) \geq 0, \\
& f(\omega_e \,|\, \boldsymbol{\theta}) \geq 0,
\end{aligned}$$

To convert the two inequalities into equalities we introduce slack variables:

$$f(x \,|\, \boldsymbol{\theta}) \geq 0 \Leftrightarrow f(x \,|\, \boldsymbol{\theta}) - s^2 = 0, \text{ for some } s \in \mathbb{R};$$

we shall refer to these new equalities using $g_1(e_1, \boldsymbol{\theta}, s_1)$ and $g_2(e_2, \boldsymbol{\theta}, s_2)$, respectively. We now have the following equality constrained optimization problem:

$$\text{Maximize } \log L(\boldsymbol{\theta} \,|\, \mathbf{x}) = \sum_{i: x_i \in \Omega_m} \log L(\boldsymbol{\theta} \,|\, x_i)$$

$$\text{Subject to } \mathbf{g}(\mathbf{x}, \boldsymbol{\theta}) = \begin{bmatrix} g_0(\mathbf{x}, \boldsymbol{\theta}) \\ g_1(\omega_s, \boldsymbol{\theta}, s_1) \\ g_2(\omega_e, \boldsymbol{\theta}, s_2) \end{bmatrix} = \mathbf{0}.$$

This optimization problem can be solved using the method of Lagrange multipliers. That is, with the Lagrangian function $l(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{s}) = \log L(\boldsymbol{\theta} \,|\, \mathbf{x}) + \lambda_0 g_0(x, \boldsymbol{\theta}) + \lambda_1 g_1(\omega_s, \boldsymbol{\theta}, s_1) + \lambda_2 g_2(\omega_e, \boldsymbol{\theta}, s_2)$ we look for a solution to the equalities defined by

$$A(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{s}) = \nabla l(\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\lambda}, \mathbf{s}) = 0.$$

Such a solution can be found numerically by applying Newton's method. Specifically, by letting $\boldsymbol{\theta}' = (\boldsymbol{\theta}^{\mathrm{T}}, s_1, s_2)^{\mathrm{T}}$, the Newton updating step is given by

$$\begin{bmatrix} \boldsymbol{\theta}'_{t+1} \\ \boldsymbol{\lambda}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\theta}'_t \\ \boldsymbol{\lambda}_t \end{bmatrix} - \nabla A(\mathbf{x}, \boldsymbol{\theta}'_t, \boldsymbol{\lambda}_t)^{-1} A(\mathbf{x}, \boldsymbol{\theta}'_t, \boldsymbol{\lambda}_t),$$

where $\boldsymbol{\theta}'_t$ and $\boldsymbol{\lambda}_t$ are the current estimates and

$$A(\mathbf{x}, \boldsymbol{\theta}'_t, \boldsymbol{\lambda}_t) = \begin{bmatrix} \nabla_{\boldsymbol{\theta}'} l(\mathbf{x}, \boldsymbol{\theta}', \boldsymbol{\lambda}) \\ \mathbf{g}(\mathbf{x}, \boldsymbol{\theta}') \end{bmatrix};$$

$$\nabla A(\mathbf{x}, \boldsymbol{\theta}'_t, \boldsymbol{\lambda}_t) = \begin{bmatrix} \nabla^2_{\boldsymbol{\theta}'\boldsymbol{\theta}'} l(\mathbf{x}, \boldsymbol{\theta}', \boldsymbol{\lambda}) & \nabla \mathbf{g}(\mathbf{x}, \boldsymbol{\theta}') \\ \nabla \mathbf{g}(\mathbf{x}, \boldsymbol{\theta}')^{\mathrm{T}} & 0 \end{bmatrix}.$$

As initialization values, $\boldsymbol{\theta}_0$, we use the maximum likelihood estimates returned by the line-search method described in Section 4.2, and in order to control the step size during updating, we employ the Armijo rule (Bertsekas, 1996). For the test results reported in Section 7, the Lagrange multipliers were initialized (somewhat arbitrarily) to 1 and the slack variables were set to $\sqrt{f(\omega_s \,|\, \boldsymbol{\theta}_0)}$ and $\sqrt{f(\omega_e \,|\, \boldsymbol{\theta}_0)}$, respectively.

Finally, it should be emphasized that the above search procedure may lead to $f(x \,|\, \boldsymbol{\theta})$ being negative for some $x$. In the current implementation we have addressed this problem rather crudely: simply terminate the search when negative values are encountered. Moreover, due to numerical instability, the search is also terminated if the determinant for the system is close to zero ($< 10^{-9}$) or if the condition number is large ($> 10^9$). Note that by terminating the search before convergence, we have no guarantees about the solution. In particular, the solution may be worse than the initial estimate. In order to overcome this problem, we always store the best parameter estimates found so far (including those found by line search) and return these estimates upon termination.

## 5. Model Selection for Univariate MTE

So far we have mainly considered MTEs with two exponential terms and with pre-specified spilt points. However, when learning an MTE potential these model parameters (i.e., the model structure) should ideally also be deduced from data.

In this section we pose MTE structure learning as a model selection problem. For the score function specification, one might take a Bayesian approach and define a candidate score function based on a conjugate prior for the MTE distribution. As a possible prior distribution, we could again look towards the MTE distribution; recall that the class of MTE distributions is closed under addition and multiplication. Unfortunately, the parameters defining an MTE distribution are not independent (as we shall see below), and it is not apparent how to specify a joint prior MTE distribution so that only admissible parameter configurations (i.e., those specifying an MTE density) contribute with non-zero probability mass. As an alternative, we resort to penalized log-likelihood when scoring the model structures. Specifically, we use the Bayesian information criterion (BIC) (Schwarz, 1978):

$$BIC(f) = \sum_{i=1}^{n} \log f(x_i \,|\, \hat{\boldsymbol{\theta}}) - \frac{\dim(f)}{2}\log(n),$$

where $\dim(f)$ is the number of free parameters in the model. To determine $\dim(f)$, consider an MTE potential $f(x \,|\, \boldsymbol{\theta})$ defined by the functions

$$f_j(x|\boldsymbol{\theta}_j) = a_0^{(j)} + \sum_{i=1}^{m_j} a_i^{(j)} \exp\left(b_i^{(j)} \cdot x\right), \qquad 1 \le j \le M,$$

for all $x \in \Omega_j = [\omega_s^{(j)}; \omega_e^{(j)}]$. In order for $f(x)$ to be a density, each sub-function $f_j(x \,|\, \boldsymbol{\theta}_j)$ should be both non-negative and account for some probability mass $c_j$ (i.e., $\int_{x\in\Omega_j} f_j(x \,|\, \boldsymbol{\theta}_j)dx = c_j$) s.t. $\sum_{j=1}^{M} c_j = 1$. First, we ensure that $f_j(x \,|\, \boldsymbol{\theta}_j)$ is non-negative by adding a positive value $a_0'$, thus tying the parameter $a_0$ in $\boldsymbol{\theta}_j$. Next, to ensure that $\int_{x\in\Omega_j} f_j(x \,|\, \boldsymbol{\theta}_j)dx = c_j$ we note that

$$\int_{x\in\Omega_j} f(x \,|\, \boldsymbol{\theta}_j)dx = a_0\left(\omega_e^{(j)} - \omega_s^{(j)}\right) + \sum_{i=1}^{m_j} \frac{a_i^{(j)}}{b_i^{(j)}}\left(\exp\left(b_i^{(j)}\omega_e^{(j)}\right) - \exp(b_i^{(j)}\omega_s^{(j)})\right),$$

and by tying, say $a_1^{(j)}$, such that

$$a_1^{(j)} = b_1^{(j)}\left[\frac{c_j - \sum_{i=2}^{m_j} \frac{a_i^{(j)}}{b_i^{(j)}}\left(\exp\left(b_i^{(j)}\omega_e^{(j)}\right) - \exp\left(b_i^{(j)}\omega_s^{(j)}\right)\right) - a_0\left(\omega_e^{(j)} - \omega_s^{(j)}\right)}{\exp(b_1^{(j)}\omega_e^{(j)}) - \exp(b_1^{(j)}\omega_s^{(j)})}\right]$$

we have that $\int_{x\in\Omega_j} f(x)dx = c_j$. For the function $f_j(x \,|\, \boldsymbol{\theta}_j)$, the number of free parameters is therefore $2 \cdot m_j - 1$, and hence the number of free parameters in

$f(x \mid \boldsymbol{\theta})$ is given by $\dim(f) = \sum_{j=1}^{M}(2 \cdot m_j - 1) + (M - 1) = \sum_{j=1}^{M} 2 \cdot m_j - 1$, where the last $M - 1$ parameters encode the probability masses assigned to the $M$ intervals or, equivalently, the choice of split points.

Based on the score function above, we can now define a search method for learning the structure of an MTE. The method is recursive and relies on two other methods for learning the number of exponential terms and the location of the split points, respectively.

When learning the number of exponential terms, for a fixed interval $\Omega_j$, we follow a greedy approach and 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 method is summarized by the pseudo-code in Algorithm 5.1, where the function $\textsc{Estimate\_MTE\_Parameters}(\mathbf{x}, \omega_s, \omega_e, i)$ implements the parameter estimation procedure described in Section 4.2 for an MTE density defined over the interval $[\omega_s, \omega_e]$ and with $i$ exponential terms. It should be emphasized that the main aim of Algorithm 5.1 (as well as Algorithms 5.2 and 5.3 below) is only to convey the general structure of a possible learning algorithm. Time complexity is therefore given less attention at this point, but will be further addressed at the end of the section.

**Algorithm 5.1** The algorithms learns an MTE density (including the number of exponential terms) for the interval $[\omega_s, \omega_e]$.

```
function CANDIDATE_MTE(x, ω_s, ω_e)
    i ← 0                                        ▷ i specifies the number of exponential terms
    (tmpθ, tmpBIC) ← ESTIMATE_MTE_PARAMETERS(x, ω_s, ω_e, i)
    repeat
        i ← i + 1
        (bestθ, bestBIC) ← (tmpθ, tmpBIC)
        (tmpθ, tmpBIC) ← ESTIMATE_MTE_PARAMETERS(x, ω_s, ω_e, i)
    until termination              ▷ E.g. tmpBIC ≤ bestBIC or max-iter. reached
    return (bestθ, bestBIC)
```

For a given interval $\Omega_j$ there is in principle an uncountable number of possible split points; however, the BIC function will assign the same score to any two split points that define the same partitioning of the training data. We therefore define the candidate split points based on the number of ways in which we can split the training data. Given these candidate split points we take a myopic approach and select the split point with the highest BIC score, assuming that the score cannot be improved by further refinement of the two sub-intervals defined by the chosen split point (see Algorithm 5.2).

Based on the two methods above, we can now outline a simple procedure for learning the structure (and the parameters) for an MTE density: recursively select the best split point (if any) for the current (sub)-interval. The overall algorithm is summarized in Algorithm 5.3, which takes as input an MTE model $current\boldsymbol{\theta}$ defined over the interval $[\omega_s, \omega_e]$, i.e., the algorithm should be invoked

**Algorithm 5.2** The algorithm finds a candidate split point for the interval $[\omega_s, \omega_e]$. We use the notation $\mathbf{x}(x > x_i)$ to denote the data points $x$ in $\mathbf{x}$ for which $x > x_i$; analogously for $\mathbf{x}(x \leq x_i)$.

**function** CANDIDATE_SPLIT_MTE($\mathbf{x}, \omega_s, \omega_e$)
    $bestBIC \leftarrow -\infty$
    **for** $i \leftarrow 1 : n-1$ **do** ▷ $n$ is the number of data points
        $\boldsymbol{\theta}_1 \leftarrow$ CANDIDATE_MTE($\mathbf{x}(x \leq x_i), \omega_s, x_i$)
        $\boldsymbol{\theta}_2 \leftarrow$ CANDIDATE_MTE($\mathbf{x}(x > x_i), x_i, \omega_e$)
        **if** $\mathrm{BIC}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \mathbf{x}) > bestBIC$ **then**
            $bestBIC \leftarrow \mathrm{BIC}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \mathbf{x})$
            $bestSplit \leftarrow (x_{i+1} - x_i)/2$
            $\boldsymbol{\theta}_1^* \leftarrow \boldsymbol{\theta}_1$
            $\boldsymbol{\theta}_2^* \leftarrow \boldsymbol{\theta}_2$
    **return** $(\boldsymbol{\theta}_1^*, \boldsymbol{\theta}_2^*, bestBIC, bestSplit)$

with $\mathbf{x}$, $\omega_s$, $\omega_e$, and CANDIDATE_MTE($\mathbf{x}, \omega_s, \omega_e$).

**Algorithm 5.3** The algorithm learns the parameters and the structure of an MTE potential for the interval $[\omega_s, \omega_e]$. The algorithm is invoked with $current\boldsymbol{\theta}$, which is an MTE for $[\omega_s, \omega_e]$ without split points (found using Algorithm 4.1). Note that $\mathbf{x}(x \leq candSplit)$ denotes the data points $x$ in $\mathbf{x}$ for which $x \leq candSplit$; analogously for $\mathbf{x}(x > candSplit)$

**function** LEARN_MTE($\mathbf{x}, \omega_s, \omega_e, current\boldsymbol{\theta}$)
    $currentBIC \leftarrow \mathrm{BIC}(\mathbf{x}, \omega_s, \omega_e, current\boldsymbol{\theta})$
    $[\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, candSplit, candBIC] \leftarrow$ CANDIDATE_SPLIT_MTE($\mathbf{x}, \omega_s, \omega_e$)
    **if** $candBIC > currentBIC$ **then**
        $(\mathbf{splits}_l, \boldsymbol{\theta}_l) \leftarrow$ LEARN_MTE($\mathbf{x}(x \leq candSplit)$, $\omega_s$, $candSplit, \boldsymbol{\theta}_1$)
        $(\mathbf{splits}_r, \boldsymbol{\theta}_r) \leftarrow$ LEARN_MTE($\mathbf{x}(x > candSplit)$, $candSplit, \omega_e, \boldsymbol{\theta}_2$)
        $\mathbf{splits} \leftarrow [\mathbf{splits}_l, candSplit, \mathbf{splits}_r]$
        $\boldsymbol{\Theta} \leftarrow (\boldsymbol{\theta}_l; \boldsymbol{\theta}_r)$
    **else**
        $\mathbf{splits} = []$
        $\boldsymbol{\Theta} = []$
    **return** $(\mathbf{splits}, \boldsymbol{\theta})$

The worst case computational complexity of the algorithm above is $O(n^3)$, where $n$ is the number of data points (we assume that the number of exponential terms is significantly smaller than $n$). This is clearly prohibitive for all but the smallest data sets. In order to overcome this problem, we can instead pre-specify a collection of candidate split points found by making an equal-width or equal frequency partitioning of the data. If there are $r$ such split points, then the worst case time complexity becomes $O(n \cdot r^2)$. The results presented in Section 7 are based on $r = 5$ candidate split points found by equal frequency partitioning of the data.

## 6. Parameter Estimation by Least Squares

We have now presented a maximum likelihood framework for learning univariate MTE potentials from data. In order to compare the merits of this new learning algorithm with a baseline method, we will proceed by describing the hitherto most used method for learning MTEs from data (Rumí et al., 2006; Romero et al., 2006). This technique is commonly denoted *least squares* (LS) estimation because it looks for parameter values that minimize the mean squared error between the fitted model and the empirical density of the sample. In early work on MTE parameter estimation (Rumí et al., 2006), the empirical density was estimated using a histogram. In order to avoid the lack of smoothness, especially when data is scarce, Romero et al. (2006) proposed to use kernels to approximate the empirical density instead of histograms, and this is also the approach we will follow here.

As the LS method does not directly seek to maximize the likelihood of the model, the resulting LS parameters are not guaranteed to be close to the ML parameters. This difference was confirmed by our preliminary experiments, and has resulted in a few modifications to the LS method presented by Romero et al. (2006): *i*) Instead of using Gaussian kernels, we used Epanechnikov kernels, which tended to provide better ML estimates in our preliminary experiments. *ii*) Since the smooth kernel density estimate assigns positive probability mass, $p^*$, outside the truncated region (called the boundary bias by Simonoff (1996)), we truncate and reweight the kernel density with $1/(1 - p^*)$. *iii*) In order to reduce the effect of low probability areas during the least squares calculations, the summands in the mean squared error are weighted according to the empirical density at the corresponding points.

Assume that there are $n_m$ points in the original sample, $\mathbf{x}$, that fall inside $\Omega_m$. Without loss of generality, in order to simplify the notation, we will assume within this section that all the elements of sample $\mathbf{x}$ belong to $\Omega_m$. In what follows we denote by $\mathbf{y} = (y_1, \ldots, y_{n_m})^{\mathrm{T}}$ the values of the empirical kernel for sample $\mathbf{x} = (x_1, \ldots, x_{n_m})^{\mathrm{T}}$, and with reference to the target density in Equation (4), we assume initial estimates for $a_0, b_0$ and $k_0$ (we will later discuss how to get these initial estimates). With this outset, $c$ and $d$ can be estimated by minimizing the *weighted mean squared error* between the function $c \exp\{d\mathbf{x}\}$ and the points $(\mathbf{x}, \mathbf{w})$, where $\mathbf{w} = \mathbf{y} - a_0 \exp\{b_0\mathbf{x}\} - k_0$. Specifically, by taking logarithms, the problem reduces to linear regression:

$$\ln\{\mathbf{w}\} = \ln\{c \exp\{d\mathbf{x}\}\} = \ln\{c\} + d\mathbf{x},$$

which can be written as $\mathbf{w}^* = c^* + d\mathbf{x}$; here $c^* = \ln\{c\}$ and $\mathbf{w}^* = \ln\{\mathbf{w}\}$. Note that we here assume that $c > 0$. In fact the data $(\mathbf{x}, \mathbf{w})$ is transformed, if necessary, to fit this constraint, i.e., to be convex and positive. This is achieved by changing the sign of the values $\mathbf{w}$ and then adding a constant to make them positive. We then fit the parameters taking into account that afterwards the sign of $c$ should be changed and the constant used to make the values positive should be subtracted.

A solution to the regression problem is then defined by

$$(c^*, d) = \arg\min_{c^*,d} (\mathbf{w}^* - c^* - d\mathbf{x})^{\mathrm{T}} \mathrm{diag}\,(\mathbf{y})\,(\mathbf{w}^* - c^* - d\mathbf{x}),$$

where diag$(\cdot)$ takes a vector as input and returns a diagonal matrix with that vector on its diagonal.

The solution can be described analytically:

$$c^* = \frac{\mathbf{w}^{\mathrm{T}}\mathrm{diag}\,(\mathbf{y})\,\mathbf{x} - d \cdot \left(\mathbf{x}^{\mathrm{T}}\mathbf{y}\right)^2}{\mathbf{x}^{\mathrm{T}}\mathbf{y}}$$

$$d = \frac{(\mathbf{w}^{\mathrm{T}}\mathbf{y})(\mathbf{x}^{\mathrm{T}}\mathbf{y}) - \left(\sum_i y_i\right)\left(\mathbf{w}^{\mathrm{T}}\mathrm{diag}\,(\mathbf{y})\,\mathbf{x}\right)}{\left(\mathbf{x}^{\mathrm{T}}\mathbf{y}\right)^2 - \left(\sum_i y_i\right) \cdot \mathbf{x}^{\mathrm{T}}\mathrm{diag}\,(\mathbf{y})\,\mathbf{x}}.$$

Once $a, b, c$ and $d$ are known, we can estimate $k$ in $f^*(x) = k + ae^{bx} + ce^{dx}$. If we let $\mathbf{s} = \mathbf{y} - ae^{b\mathbf{x}} - ce^{d\mathbf{x}} - k$, we have that $k \in \mathbb{R}$ should be the value minimizing the error

$$\mathrm{Error}(k) = \frac{1}{n_m}\mathbf{s}^{\mathrm{T}}\mathrm{diag}\,(\mathbf{y})\,\mathbf{s}.$$

This is achieved for

$$\hat{k} = \frac{(\mathbf{y} - ae^{b\mathbf{x}} - ce^{d\mathbf{x}})^{\mathrm{T}}\mathbf{y}}{\sum_i y_i}.$$

Here we are assuming a fixed number of exponential terms. However, as the parameters are not optimized globally, there is no guarantee that the fitted model minimizes the *weighted mean squared error*. This fact can be somewhat corrected by determining the contribution of each term to the reduction of the error as described by Rumí et al. (2006).

The initial values $a_0$, $b_0$ and $k_0$ can be arbitrary, but "good" values can speed up convergence. We consider two alternatives: *i*) Initialize the values by fitting a curve $ae^{bx}$ to the modified sample by exponential regression, and compute $k$ as before. *ii*) Force the empiric density and the initial model to have the same derivative. In the current implementation, we try both initializations and choose the one that minimizes the squared error.

## 7. Experimental Comparison

In order to evaluate the proposed learning algorithm we have sampled datasets from six distributions: An MTE density defined by two regions, a beta distribution $Beta(0.5, 0.5)$, a standard normal distribution, a $\chi^2$ distribution with eight degrees of freedom, and a log-normal distribution $LN(0, 1)$. From each distribution we sampled two different training sets having sizes 1000 and 50, respectively. This last dataset is devoted to show the performance of the different methods when data is scarce. In order to check the predictive ability of the estimated models, a test set of size 1000 was also sampled.

| | MTE | Beta | $\chi^2$ | Normal 1 split | Normal 3 splits | Log-normal |
|---|---|---|---|---|---|---|
| ML | $-2263.37$ | 160.14 | $-2695.02$ | $-1411.79$ | $-1380.45$ | $-1415.06$ |
| LS | $-2307.21$ | 68.26 | $-2739.24$ | $-1508.62$ | $-1403.46$ | $-1469.21$ |
| Original LS | $-2338.46$ | 39.68 | $-2718.99$ | $-1570.62$ | $-1406.23$ | $-1467.24$ |

| | MTE | Beta | $\chi^2$ | Normal 1 split | Normal 3 splits | Log-normal |
|---|---|---|---|---|---|---|
| ML | $-2263.13$ | 160.69 | $-2685.76$ | $-1420.34$ | $-1392.28$ | $-1398.30$ |
| LS | $-2321.18$ | 60.29 | $-2742.80$ | $-1509.11$ | $-1468.11$ | $-2290.17$ |
| Original LS | $-2556.68$ | 39.42 | $-2766.86$ | $-1565.28$ | $-1438.67$ | $-1636.99$ |

Table 1: Comparison of ML vs. LS in terms of likelihood. In the upper table the split points were found using the method described in (Rumí et al., 2006), and in the lower table they were defined by the extreme points and the inflexion points of the exact density.

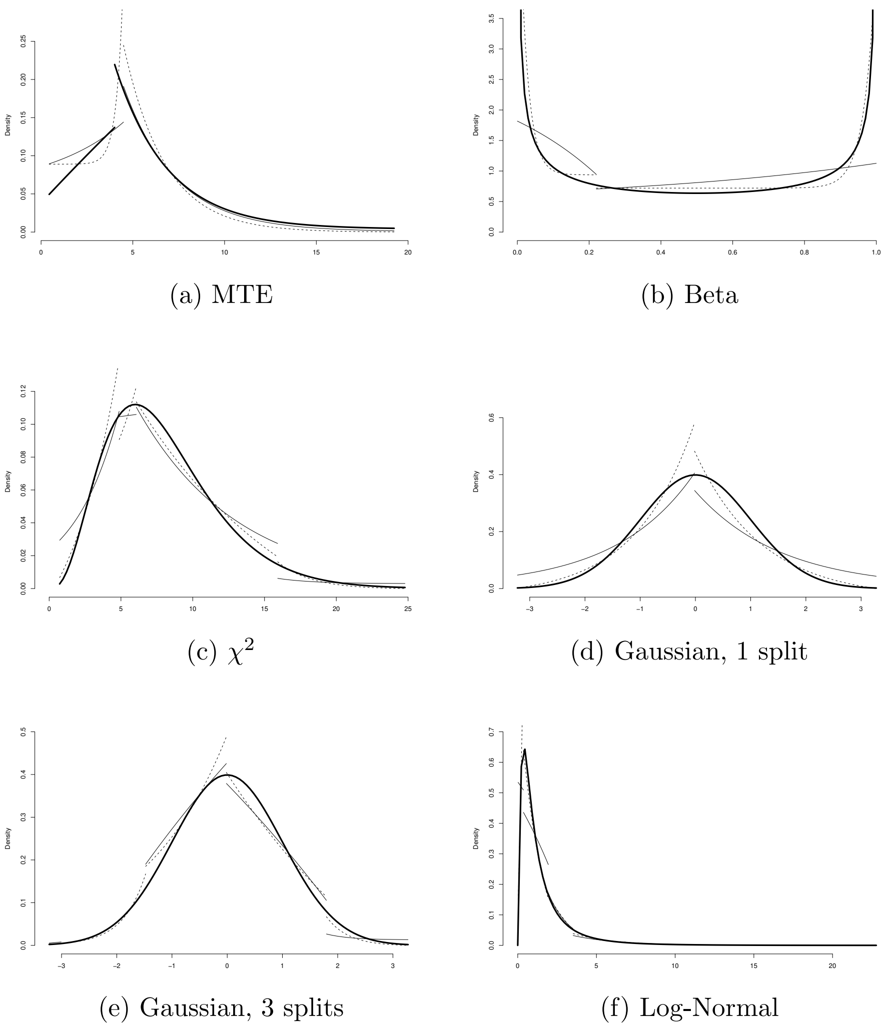
| | MTE | Beta | $\chi^2$ | Normal 1 split | Normal 3 splits | Log-normal |
|---|---|---|---|---|---|---|
| ML | $-2284.61$ | 279.24 | $-2719.88$ | $-1434.46$ | $-1417.35$ | $-1375.81$ |
| LS | $-2312.69$ | 88.04 | $-2719.47$ | $-1513.32$ | $-1424.08$ | $-1411.67$ |
| Original LS | $-2335.80$ | 50.62 | $-2713.43$ | $-1585.86$ | $-1417.88$ | $-1394.78$ |

| | MTE | Beta | $\chi^2$ | Normal 1 split | Normal 3 splits | Log-normal |
|---|---|---|---|---|---|---|
| ML | $-2283.73$ | 253.78 | $-2705.98$ | $-1433.32$ | $-1415.26$ | $-1362.56$ |
| LS | $-2327.12$ | 78.65 | $-2713.23$ | $-1514.73$ | $-1474.88$ | $-2256.10$ |
| Original LS | $-2550.00$ | 50.44 | $-2744.77$ | $-1579.82$ | $-1445.67$ | $-1594.44$ |

Table 2: Comparison of ML vs. LS in terms of the test set likelihood. In the upper table the split points were found using the method described in (Rumí et al., 2006), and in the lower table they were defined by the extreme points and the inflexion points of the exact density.

Our first group of tests consider learning of MTEs assuming that the domain has already been divided into intervals, and that the number of exponential terms has been fixed to 2. When testing with data from the MTE, beta and normal distributions, we have used one split point, whereas for the log-normal and the $\chi^2$ distributions, the number of split points was set to three. We have also run the experiment with three split points for the standard normal distribution. We have used two methods for finding split points: *i*) Define the split points to be the extreme points and inflection points of the true generating density function, and *ii*) use the procedure described by Rumí et al. (2006). The plots of the fitted models using the training set of size 1000 together with the original density are displayed in Figure 3. The split points used for these plots were selected using the second approach above; results using the former approach for detecting split points are qualitatively similar.

Turning to the quantitative results, Table 1 shows the likelihood of the different samples for the models fitted with the 1000 size training set using the direct ML approach, the modified LS method, and the original LS method described in Rumí et al. (2006). The two sub-tables correspond to the split points found using the method described in Rumí et al. (2006) and split points found by identifying the extreme points and the inflexion points of the true density, respectively. Table 2 shows the likelihood of the test set for the same models,



(a) MTE (b) Beta

(c) $\chi^2$ (d) Gaussian, 1 split

(e) Gaussian, 3 splits (f) Log-Normal

Figure 3: The plots show the results of samples from different distributions. The gold-standard distribution is drawn with a thick line, the MTE with Lagrange-parameters are given with the dashed line, and the results of the LS approach are given with the thin, solid line.

| | MTE | Beta | $\chi^2$ | Normal 1 split | Normal 3 splits | Log-normal |
|---|---|---|---|---|---|---|
| ML | −2319.24 | 256.27 | −2265.76 | −1486.58 | −1530.58 | −1605.60 |
| LS | −2612.27 | 48.82 | −2858.66 | −1506.39 | −1491.58 | −1652.98 |
| Original LS | −2337.54 | −9.57 | −2823.48 | −1505.06 | −1455.52 | −1462.99 |

| | MTE | Beta | $\chi^2$ | Normal 1 split | Normal 3 splits | Log-normal |
|---|---|---|---|---|---|---|
| ML | −2318.29 | 228.45 | −2588.22 | −1470.2 | −1501.53 | −1491.56 |
| LS | −2649.21 | 68.37 | −2885.46 | −1585.78 | −1513.75 | −2277.66 |
| Original LS | −2529.87 | −28.05 | −2837.81 | −1527.32 | −1484.77 | −2165.59 |

Table 3: Comparison of ML vs. LS estimated with a sample of size 50 in terms of the test set likelihood. In the upper table the split points were found using the method described in (Rumí et al., 2006), and in the lower table they were defined by the extreme points and the inflexion points of the exact density.

and Table 3 shows the likelihood of the test set for the models fitted with the 50 size training set. From the results we clearly see that the ML-based method outperforms the LS method in terms of likelihood. This is hardly a surprise, as the ML method is actively using likelihood maximization as its target, whereas the LS methods do not. On the other hand, the LS and Original LS seem to be working at comparable levels. Most commonly (in 15 out of 24 runs), LS is an improvement over its original version with large training sets, but with small training sets it behaves much worse. The explanation is that the weights used in the new version of LS are not so accurate, and so the estimations are unstable. We can also see from Table 3 that the ML approach appears to overfit the data, and therefore achieves a lower testset likelihood than the original LS for the "Normal 3 splits" and "Log-normal" datasets. This is not surprising, as the number of parameters used by the ML approach to fit the distributions are far above what turns out to be "BIC-optimal" (see below).

Our last set of tests focused on the BIC-based model selection algorithm for finding split points and determining the number of parameters inside each interval; for these tests we allowed at most two exponential terms and five candidate split points (found using equal-frequency binning). The results, given in Table 4, clearly show the desired effect: The BIC-based method is less prone to overfitting the data, and although a smaller likelihood is obtained on the training data, the predictive ability of the data is better than when learning with fixed split points. Plots of the learned MTEs are shown in Figure 4, where it is interesting to note how the BIC-based learning algorithm chooses different model structures for the different data-sizes. Look, for instance, at the Beta distribution in Part (b) of Figure 4. When only 50 training examples are used, we fit a function with two exponential terms to the whole support of the density (no split points are selected); when 1000 cases are available, the learning prefers to use two intervals. Furthermore, for the first interval of the MTE distribution (Part (a)), the learning based on 1000 cases finds support for using one exponential term to approximate the density. When learning from 50 cases, the algorithm did not get the same support, and therefore opted for

| | MTE | Beta | $\chi^2$ | Normal | Log-normal |
|---|---|---|---|---|---|
| Training-set Log likelihood | −94.90 | 14.80 | −122.30 | −62.84 | −65.18 |
| Test-set Log likelihood | −2052.00 | 130.60 | −2500.93 | −1210.18 | −1225.36 |

| | MTE | Beta | $\chi^2$ | Normal | Log-normal |
|---|---|---|---|---|---|
| Training-set Log likelihood | −2270.76 | 161.21 | −2727.76 | −1422.58 | −1432.83 |
| Test-set Log likelihood | −2285.25 | 249.45 | −2702.67 | −1430.88 | −1358.38 |

Table 4: The results of the BIC-based learning approach. In the upper table the results are based on learning from 50 data-points, the lower table reports the results based on 1000 training examples.

a constant in that part of the domain. Finally, it is interesting to see that the log-normal (Part (e)) is approximated using 0 split points (when learning from 50 cases) or 1 split point (when learning from 1000 cases). This should be compared to the 3 split points used to generate the results in Figure 3 (f).

## 8. Conclusions and Future Work

In this paper we have introduced *maximum likelihood* learning of MTEs. Finding maximum likelihood parameter estimates is interesting not only in its own right, but also as a tool for doing more advanced learning, like model selection. We have proposed algorithms that use the BIC criteria (Schwarz, 1978) to choose the number of exponential terms required to approximate the density function properly, as well as for determining the location of the split-points for partitioning the domain of the variables. The experiments carried out show that the estimations obtained by ML improve the ones provided by the least squares method both in terms of likelihood of the training data and of the predictive ability (measured by likelihood of a separate test-set).

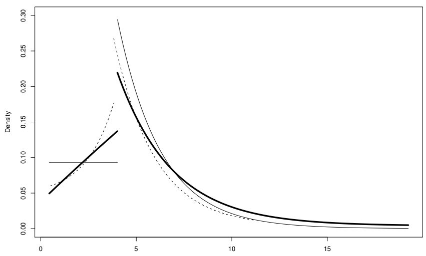
We are currently working on ML-based learning of *conditional distributions*, starting from the ideas published in (Moral et al., 2003). However, accurately locating the split-points for a conditional MTE is even more difficult than when learning marginal distributions; locating the split-points for a variable will not only influence the approximation of its distribution, but also the distributions for all its children.

## Acknowledgments

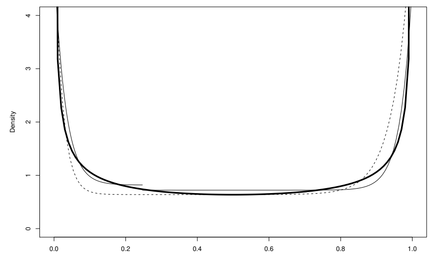
This work has been partly supported by the Spanish Ministry of Science and Innovation through project TIN2007-67418-C03-02 and by EFRD (FEDER) funds.

## References

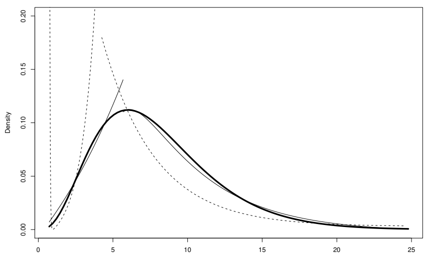
Bertsekas, D., 1996. Constrained optimization and Lagrange multiplier methods. Academic Press Inc.



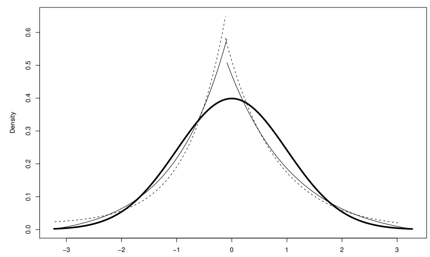

(a) MTE



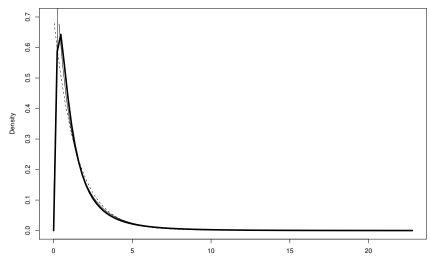

(b) Beta




(c) $\chi^2$




(d) Gaussian




(e) Log-Normal

Figure 4: The plots show the results of the BIC-based learning. The gold-standard distribution is drawn with a thick line, the MTE with BIC-based learning from 50 examples are are given with the dashed line, and the results of the BIC-based learning from 1000 cases are given with the thin, solid line.

Cobb, B., Shenoy, P., Rumí, R., 2006. Approximating probability density functions with mixtures of truncated exponentials. Statistics and Computing 16, 293–308.

Dempster, A., Laird, N., Rubin, D., 1977. Maximun likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society B 39, 1 – 38.

Farag, A., El-Baz, A., Gimel'farb, G., 2004. Density estimation using modified expectation-maximization algorithm for a linear combination of Gaussians. In: Proceedings of the International Conference on Image Processing (ICIP). pp. 1871–1874.

Friedman, N., Goldszmidt, M., 1996. Discretizing continuous attributes while learning Bayesian networks. In: Proceedings of the Thirteenth International Conference on Machine Learning. pp. 157–165.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., Saul, L. K., 1999. An introduction to variational methods for graphical models. Machine Learning 37, 183–233.

Kozlov, D., Koller, D., 1997. Nonuniform dynamic discretization in hybrid networks. In: Geiger, D., Shenoy, P. (Eds.), Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence. Morgan & Kaufmann, pp. 302–313.

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

Moral, S., Rumí, R., Salmerón, A., 2003. Approximating conditional MTE distributions by means of mixed trees. In: ECSQARU'03. Lecture Notes in Artificial Intelligence. Vol. 2711. pp. 173–183.

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

Rumí, R., Salmerón, A., Moral, S., 2006. Estimating mixtures of truncated exponentials in hybrid Bayesian network. Test 15, 397–421.

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

Shafer, G. R., Shenoy, P. P., 1990. Probability Propagation. Annals of Mathematics and Artificial Intelligence 2, 327–352.

Simonoff, J., 1996. Smoothing methods in Statistics. Springer.

Tanner, M., 1996. Tools for statistical inference. Springer.