# Mixtures of Bagged Markov Tree Ensembles

François Schnitzler, Pierre Geurts, Louis Wehenkel
Université de Liège, Belgium
fschnitzler@ulg.ac.be, P.Geurts@ulg.ac.be, L.Wehenkel@ulg.ac.be

**Abstract**

Markov trees, a probabilistic graphical model for density estimation, can be expanded in the form of a weighted average of Markov Trees. Learning these mixtures or ensembles from observations can be performed to reduce the bias or the variance of the estimated model. We propose a new combination of both, where the upper level seeks to reduce bias while the lower level seeks to reduce variance. This algorithm is evaluated empirically on datasets generated from a mixture of Markov trees and from other synthetic densities.

## 1 Introduction

Estimating a multivariate probability density over a set $\mathcal{X}$ of $p$ variables $X_1, \ldots, X_p$, based on a sample of joint observations of these variables, is an important challenge to deal with the uncertainty inherent to many fields, such as bioinformatics or fault diagnostics. A well-known framework to encode multivariate densities is the class of Bayesian Networks. Such a probabilistic graphical model defines a factorization of the joint probability distribution :

$$\mathbb{P}_{\mathcal{G}}(\mathcal{X}) = \prod_{i=1}^{p} \mathbb{P}_{\mathcal{G}}(X_i | Pa_{\mathcal{G}}(X_i)) \ , \qquad (1)$$

where $\mathcal{G}$ is the graphical structure of the Bayesian network (a directed acyclic graph whose vertices are in bijection with $\mathcal{X}$) and $Pa_{\mathcal{G}}(X_i)$ denotes the subset of variables parent of $X_i$ in $\mathcal{G}$. $\mathcal{G}$ encodes conditional independence relationships implied by this factorization (Pearl, 1988; Koller and Friedman, 2009).

A structure $\mathcal{G}$ of a Bayesian network can be learned from a sample $\mathbf{D}$ of $N$ observations generated from the target density. This is typically done by optimizing a decomposable score (maximum likelihood, BIC, AIC...) over the space of candidate structures, or by matching a set of conditional independences inferred from $\mathbf{D}$. Learning the parameters of the conditional densities $\mathbb{P}_{\mathcal{G}}(X_i|Pa_{\mathcal{G}}(X_i))$ essentially amounts to counting in $\mathbf{D}$ the frequencies of joint configurations of the variables $\{X_i\} \cup Pa_{\mathcal{G}}(X_i)$.

A Bayesian network model can be exploited to answer probabilistic queries, such as estimating the conditional density of a subset of variables given the values of another subset of variables. This process is also called inference.

The complexity of learning (Chickering et al., 1994) and inference (Cooper, 1990; Kwisthout et al., 2010) for the class of Bayesian networks is however NP-hard, and in practice not possible beyond a few thousand variables (Auvray and Wehenkel, 2008; Elidan and Gould, 2008). With the rapid expansion in data acquisition technology, many applications exceed this limit. One strategy to tackle this problem is to constrain the structures considered, e.g. (Bach and Jordan, 2001; Elidan and Gould, 2008).

Markov trees are such a restricted subclass of Bayesian networks, where the graphical structure underlying any model is connected and $|Pa_{\mathcal{G}}(X_i)| \leq 1$ (Pearl, 1988). This subclass is particularly interesting: constructing the structure maximizing the likelihood of the sample $\mathbf{D}$ has a quadratic complexity in the number of variables while inference has linear complexity.

Mixtures of Markov trees are a convex combination of a set $\hat{\mathcal{T}} = \{T_1, \ldots, T_m\}$ of $m$ elementary Markov tree densities. They were introduced in (Meila and Jordan, 2001) to improve on Markov trees while retaining their interesting scaling behavior with respect to the num-

ber of variables. There are two frameworks for building those mixtures. The first one attempts to improve the modeling capacity of Markov trees, which may be insufficient to model a problem correctly due to the structural constraints. Learning the mixture is in that case viewed as a global optimization problem aiming at maximizing the data likelihood (Meila and Jordan, 2001). In the second framework, $\hat{\mathcal{T}}$ can be viewed as a set of different alternatives that cannot be discriminated based on the data set. Considering different possibilities obtained by a randomized algorithm reduces the variance of the Chow-Liu tree (Ammar et al., 2010). Another method to reduce overfitting is regularization, but it requires tuning the regularization strength. It has been applied to Markov trees as well (Tan et al., 2011; Liu et al., 2011).

These two types of mixtures are combined by Kollin and Koivisto (2006), who define a Markov-Chain Monte Carlo over k-clusterings of the observations and construct one term on each cluster by exact Bayesian averaging over the space of Markov trees (Meila and Jaakkola, 2006); and by Kirshner and Smyth (2007), who use a similar approach but allow k to vary and sample one tree from the Bayesian posterior. While both papers report an improvement over the original mixture of Meila and Jordan (2001), these methods are of cubic complexity in the number of variables, thus deteriorating the scalability of Markov trees.

In this work we develop a more scalable (quadratic) alternative to combine these two frameworks, by replacing Markov trees in the first approach (maximum likelihood, ML) by a mixture of trees generated by the second (model averaging). Rather than learning a single Markov tree by the Chow-Liu algorithm for each term of the upper mixture, a randomized procedure is used to replace each term by a lower level mixture. The combination order is thus the opposite of (Kirshner and Smyth, 2007). This opposite order is essentially motivated by computational considerations.

We begin by recalling the Chow-Liu algorithm in Section 2 and presenting mixtures of Markov trees more formally in Section 3. The proposed approach is described in Section 4, its interest and variance reduction capacity are experimentally studied in Section 5.

**Algorithm 1** Chow-Liu algorithm

**Input:** data $\mathbf{D}$
$M = [0]_{p \times p}$
**for** $i_1 = 1$, $i_2 = i_1 + 1$ **to** $p$, $p$ **do**
  $M[i_1, i_2] = I_{\mathbf{D}}(X_{i_1}; X_{i_2})$
**end for**
**return** $T = \mathbf{MWST}(M)$

## 2 Markov Trees

The Chow-Liu algorithm (1968) is used as a building block in most methods for learning a mixture of Markov trees. It outputs the Markov tree structure maximizing the likelihood of the learning sample $\mathbf{D}$, the solution of the following optimization problem :

$$T_{CL}(\mathbf{D}) = \arg\max_{T} \sum_{(X_{i_1}, X_{i_2}) \in \mathcal{E}(T)} I_{\mathbf{D}}(X_{i_1}; X_{i_2}) \ , \quad (2)$$

where $\mathcal{E}(T)$ is the set of edges of the tree $T$.

The Chow-Liu algorithm (algorithm 1) first estimates the mutual information $I_{\mathbf{D}}(X_{i_1}; X_{i_2})$ between any pair of variables $X_{i_1}$ and $X_{i_2}$, hence defining the symmetric adjacency matrix $M$ of a complete graph. In a second step, a maximum-weight spanning tree is computed based on $M$, yielding an undirected graph which may be directed by choosing an arbitrary node as root.

## 3 Mixtures of Markov Trees

Working in the class of Markov trees is algorithmically efficient, but can be either too restrictive, i.e. have a large bias, when samples are plentiful, or lead to overfitting, i.e. have a large variance, when only few samples are available.

Mixtures of Markov trees can reduce either shortcoming of Markov trees while preserving their algorithmic scalability in $p$. Such a mixture $\{\hat{\mathcal{T}}, w\}$ is composed of a set $\hat{\mathcal{T}} = \{T_1, \ldots, T_m\}$ of $m$ elementary Markov tree densities and a set $w = \{w_i\}_{i=1}^m$ of weights, yielding

a density in the form of a convex combination:

$$\mathbb{P}_{\hat{\mathcal{T}}}(\mathcal{X}) = \sum_{k=1}^{m} w_k \mathbb{P}_{T_k}(\mathcal{X}) \ . \qquad (3)$$

This is a proper density, assuming $w_i \in [0,1]$ and $\sum_{i=1}^{m} w_i = 1$. Those two frameworks are briefly presented below, each with an exemplative algorithm, also used as building blocks in the combination proposed in Section 4.

### 3.1 Mixture for Reducing the Bias

Constraining a Bayesian network to a Markov tree when the target distribution is more complex than such a tree and for a sufficiently large number of samples will cause some relationships between variables to be missed and the distribution won't be perfectly estimated, an error called the bias. Using a mixture rather than a single Markov tree results in an improved modeling power (Meila and Jordan, 2001) and a higher achievable likelihood.

This optimization can be done e.g. by the EM algorithm (Dempster et al., 1977; McLachlan and Krishnan, 2008). The weights are considered as the marginal probabilities $\mu_k = \mathbb{P}(Z = k)$ of $Z$, a hidden variable selecting one distribution $\mathbb{P}_{T_k}(\mathcal{X}) = \mathbb{P}(\mathcal{X}|Z = k)$:

$$\mathbb{P}_{\hat{\mathcal{T}}}(\mathcal{X}) = \sum_{k=1}^{m} \mathbb{P}(Z = k)\mathbb{P}(\mathcal{X}|Z = k) \ . \qquad (4)$$

This estimate of $\mathbb{P}(\mathcal{X}, Z)$ is optimized by alternating between estimating a distribution of the hidden variable Z for each observation $\mathbf{D}[i]$ and optimizing $\mathbb{P}(Z = k)$ and $\mathbb{P}(\mathcal{X}|Z = k)$. This process is described in algorithm 2, where $\gamma_k(i)$ can be seen as the probability that $Z = k$ for observation $\mathbf{D}[i]$ and according to the current estimate of $\mathbb{P}(\mathcal{X}, Z)$. The vectors $(\gamma_1(i), \ldots, \gamma_m(i))_{i=1}^{N}$ define a soft partition of $\mathbf{D}$ into $m$ weighted learning samples $\mathbf{D}'_k$, where each $\mathbf{D}'_k$ is constructed by associating a weight $\gamma_k(i)$ to each $\mathbf{D}[i]$. By comparison, all observations in $\mathbf{D}$ have a weight of one. In the second step of the iteration, the Chow-Liu algorithm is applied to each $\mathbf{D}'_k$ to obtain $T_k$ and $\mu_k$ is estimated based on the $\gamma_k(i)$.

**Algorithm 2** MT-EM: ML mixture of Markov trees

**Input:** data $\mathbf{D}$, mixture size $m$
Initialize $\{\hat{\mathcal{T}}, \mu\}$
**repeat**
  **for** $k = 1$ **to** $m$ and $i = 1$ **to** $N$ **do**
    $\gamma_k(i) = \dfrac{\mu_k \mathbb{P}_{T_k}(\mathbf{D}[i])}{\sum_{k=1}^{m} \mu_k \mathbb{P}_{T_k}(\mathbf{D}[i])}$
  **end for**
  **for** $k = 1$ **to** $m$ **do**
    $\mathbf{D}'_k = (\mathbf{D}[i], \gamma_k(i))_{i=1}^{N}$
    $T_k = \textbf{Chow-Liu}(\mathbf{D}'_k)$ ; $\mu_k = \dfrac{\sum_{i=1}^{N} \gamma_k(i)}{N}$
  **end for**
**until** convergence
**return** mixture of Markov trees $\{\hat{\mathcal{T}}, \mu\}$

### 3.2 Mixture for Reducing the Variance

When the number of available samples is small, two datasets $\mathbf{D}_1$ and $\mathbf{D}_2$ will probably result in two different models, and the smaller the sample size $N$, the greater the difference. This is called the variance. When this variance is too high, the models are said to be overfitting the data, i.e. they are modeling noise caused by the low number of samples. To reduce this effect, the second framework builds mixtures as an average over different plausible models.

A Bayesian approach where all trees are averaged is possible (Meila and Jaakkola, 2006), but it is of cubic complexity in $p$ and does not permit inference. Although sampling models from this distribution is possible, we want to retain the quadratic complexity of the Chow-Liu algorithm. This is possible using the "perturb and combine" framework, which was really successful in supervised learning and can also be applied to density estimation. It consists in perturbing a learning algorithm to randomize it (making it suboptimal), and in combining the models output by different calls to this randomized algorithm.

Bagging (Breiman, 1996) is a widely used method of this category that consists in applying a given learning algorithm on $m$ bootstrap replicates of an original data set $\mathbf{D}$, resulting in $m$ different models. A bootstrap replicate

**Algorithm 3** MT-Bag: bagging Markov trees

**Input:** data $\mathbf{D}$, mixture size $m$
$\hat{\mathcal{T}} = \emptyset$, $\lambda = \{1/m, \cdots, 1/m\}$
**for** $k = 1$ **to** $m$ **do**
  $\mathbf{D}'$= bootstrap($\mathbf{D}$)
  $\hat{\mathcal{T}} = \hat{\mathcal{T}} \cup \{\textbf{Chow-Liu}(\mathbf{D}')\}$
**end for**
**return** mixture of Markov trees $\{\hat{\mathcal{T}}, \lambda\}$

is constructed by randomly sampling (with replacement) observations from $\mathbf{D}$ and copying them in the replicate. Typically, the replicate has the same size as $\mathbf{D}$.

Algorithm 3 describes bagging applied to Markov trees. The resulting set $\hat{\mathcal{T}}$ associated to uniform weights can be viewed as a mixture.

## 4 Two-level Mixtures

In this section we describe our new algorithm to combine the methods presented in the previous section. Combining a bias and a variance reducing framework has also been considered in supervised learning, see e.g. (Breiman, 1999).

The EM algorithm builds a soft partition of size $m$ of the learning set and one maximum-likelihood tree on each downweighted data set $\mathbf{D}'_k$. Therefore each tree is built using a smaller effective number of samples than in the original data set. On the other hand, algorithms of the second category build mixtures that are increasingly better than a single tree when the number of samples shrinks (Schnitzler et al., 2010).

We therefore propose to replace the Chow-Liu step in algorithm 2 by an algorithm of the variance reduction framework to learn each term of the ML mixture. Each algorithm can operate in the configuration it excels in. MT-EM operates on the whole, preferably large, learning set at the upper level, to decrease the bias. MT-Bag works on each smaller subset of observations at the lower level, to reduce the variance. The resulting model is thus a two-level mixture of Markov trees of the following form:

$$\mathbb{P}_{\hat{\mathcal{T}}}(\mathcal{X}) = \sum_{k=1}^{m_1} \mu_k \mathbb{P}_{\hat{\mathcal{T}}_k}(\mathcal{X}) \tag{5}$$

**Algorithm 4** MT-EM-Bag: two-level mixture

**Input:** data $\mathbf{D}$, mixture sizes $m_1$, $m_2$
$\{\hat{\mathcal{T}}, \mu\} = \textbf{MT-EM}(\mathbf{D}, m_1)$
**for** $i = 1$, $k = 1$ **to** $N$, $m_1$ **do**
  $\gamma_k(i) = \dfrac{\mu_k \mathbb{P}_{T_k}(\mathbf{D}[i])}{\sum_{k=1}^{m} \mu_k \mathbb{P}_{T_k}(\mathbf{D}[i])}$ {EM weights}
**end for**
**for** $k = 1$ **to** $m_1$ **do**
  $\mathbf{D}'_k = (\mathbf{D}[i], \gamma_k(i))_{i=1}^{N}$
  $\{\hat{\mathcal{T}}_k, \lambda_k\} = \textbf{MT-Bag}(\mathbf{D}'_k, m_2)$
**end for**
**return** mixture of Markov trees $\{\hat{\mathcal{T}}, \lambda\}$

$$\mathbb{P}_{\hat{\mathcal{T}}_k}(\mathcal{X}) = \sum_{j=1}^{m_2} \lambda_{k,j} \mathbb{P}_{T_{k,j}}(\mathcal{X}) \quad \forall k \in [1, m_1] \ , \tag{6}$$

where $m_1$ and $m_2$ are respectively the number of terms in the upper level EM optimization of the mixture and in the lower level mixtures $\hat{\mathcal{T}}_k$ learned on each $\mathbf{D}'_k$; $T_{k,j}$ is the $j$th Markov tree in $\hat{\mathcal{T}}_k$ and $\lambda_{k,j}$ denotes its weight.

Using algorithms for reducing the variance rather than the Chow-Liu algorithm during the iterative process of the EM algorithm substantially slows down each step and alters the convergence, resulting in a lower accuracy than MT-EM. Hence, we substitute MT-Bag to Chow-Liu only after a first run until convergence of the basic EM algorithm using single trees and do not modify the $\gamma_k(i)$. The resulting method is specified by algorithm 4.

Note that bagging is now performed on a weighted data set. The samples composing the replicates in MT-Bag are thus no longer drawn uniformly, but based on a distribution where the probability of selecting sample $i$ equals $\gamma_k(i) / \sum_{i=1}^{N} \gamma_k(i)$. $\sum_{i=1}^{N} \gamma_k(i)$ (truncated to an integer) observations are drawn (and associated to a weight of 1) to form each replicate.

## 5 Experimental Validation

We consider two strategies for building the second level mixture. In addition to bagging Markov trees (algorithm 4, MT-EM-Bag), we consider locking the first tree in each second-level mixture to the Chow-Liu tree for that term, i.e. a mix of the first two methods (MT-

EM-BagCl). In our implementation, the parameters of a tree are always learned from the full $\mathbf{D}'_k$ and not from a bootstrap replicate, since it improves accuracy (Schnitzler et al., 2010).

We evaluate the interest of these combinations against the baseline algorithm 2 described in Section 3. The EM algorithm may converge to local maxima, so in order to remove the influence of its random initialization, every comparison between the methods is performed based on mixtures built on similar convergence points of the EM algorithm. In other words, for every run of the EM algorithm, we use its final soft partition of the learning set $(\gamma_k(i))$ to build one two-level mixture based on each algorithm considered. We also use the weights $\mu_k$ returned by this algorithm to formulate the resulting model.

The evaluation is performed using two different target distribution settings:

1. 1 mixture model of three randomly generated Markov trees defined on 100 variables;
2. 5 synthetic relatively sparse Bayesian networks of 200 variables, generated by randomly sampling for each variable $X_i$, $k_i$ parents in $\{X_1, \cdots, X_{i-1}\}$, where $k_i$ is randomly sampled in $[0, \min(i-1, 5)]$;

The accuracy of any model $\mathbb{P}_{\hat{\mathcal{T}}}$ constructed is quantified using a Monte Carlo estimate of the Kullback-Leibler divergence, with respect to the target distribution $\mathbb{P}$:

$$\hat{D}_{KL}(\mathbb{P} \,||\, \mathbb{P}_{\hat{\mathcal{T}}}) = \frac{1}{N'} \sum_{x \sim \mathbb{P}}^{N'} \log_e \left( \frac{\mathbb{P}(x)}{\mathbb{P}_{\hat{\mathcal{T}}}(x)} \right) \quad , \quad (7)$$

where $N'$ the number of observations $x$ for the estimate equals 10000 in the first case and 50000 in the second. The same samples are used for every evaluation to a given target distribution.

In the first setting, the similarity of any learned mixture to the target mixture was also assessed, using the weight of the maximum weighted bipartite matching between the terms in the estimated mixture and those of the target model. For MT-EM-Bag and MT-EM-BagCl, an original Markov tree $T$ is compared to a mixture through the average number of common edges between $T$ and each tree of the mixture.

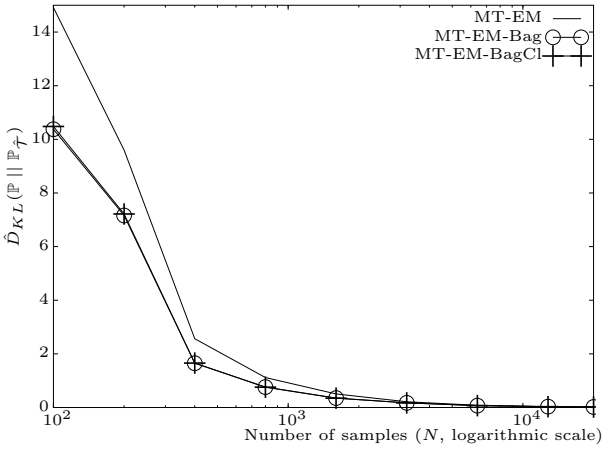



Figure 1: Setting 1: The Average KL divergence with respect to the target mixture of 3 trees converges to zero for all methods as N increases.

Below $m_2$ is always 10 for the lower level mixtures. Increasing $m_2$ further reduces variance without effect on bias, and so is liable to further increase the accuracy of the models.

Due to space restrictions, data sets, supplementary results and graphics (including on more realistic problems) are provided online[1].

### 5.1 The Target is a Mixture of 3 Trees

A first set of experiments is performed on a target distribution belonging to the class of models considered by the EM learning algorithm, here the set of mixtures of 3 Markov trees.

The number of upper-level terms $m_1$ is always fixed to 3 (since in this context the size of the target mixture is known a priori); the number $m_2$ of lower level terms is fixed to 10 to limit execution time. Increasing $m_2$ may further reduce the variance and improve the accuracy. Learning samples sizes from 100 to 20000 were considered, and several initializations (905 to 50 runs, see Figure 3) were performed for each sample size.

Figure 1 displays the decrease of the KL divergence to zero as $N$ increases, as expected since the target distribution is a mixture of 3 Markov trees. We observe that the two-level mixtures are better that the baseline, showing the interest of their variance reduction. Actu-

[1] www.montefiore.ulg.ac.be/∼schnitzl/PGM2012/

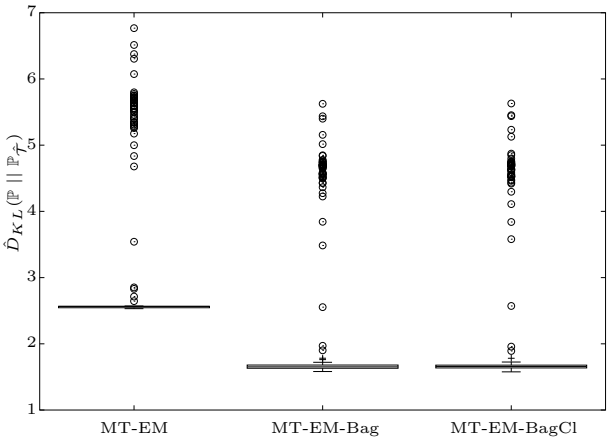



Figure 2: Setting 1: For each method, a box-plot showing the distribution of the KL divergences of the models it produces over 400 runs on a dataset of $N = 400$ samples.

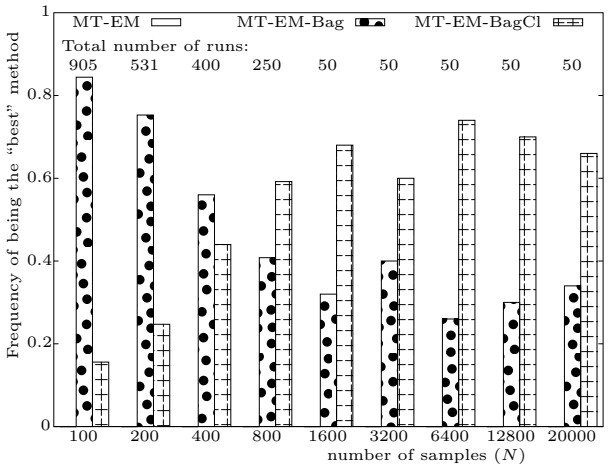



Figure 3: Setting 1: Relative number of runs where each method is the best, displayed by number of samples. MT-EM is always at 0.

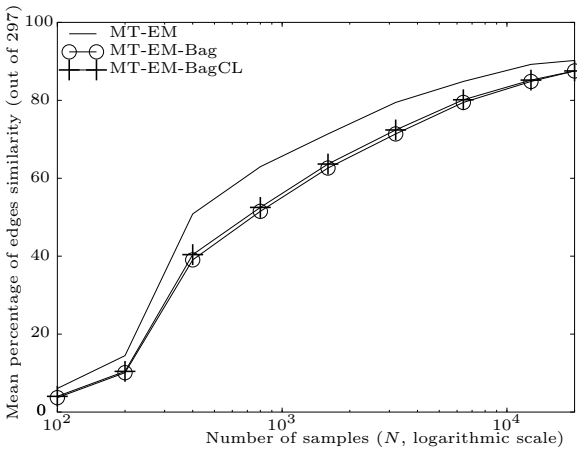



Figure 4: Setting 1: The average edge similarity increases with the learning sample size. MT-EM is better for structure recovery.

ally, MT-EM-Bag and MT-EM-BagCl achieve with $N$ samples roughly the same accuracy as the baseline MT-EM method with $2N$ samples.

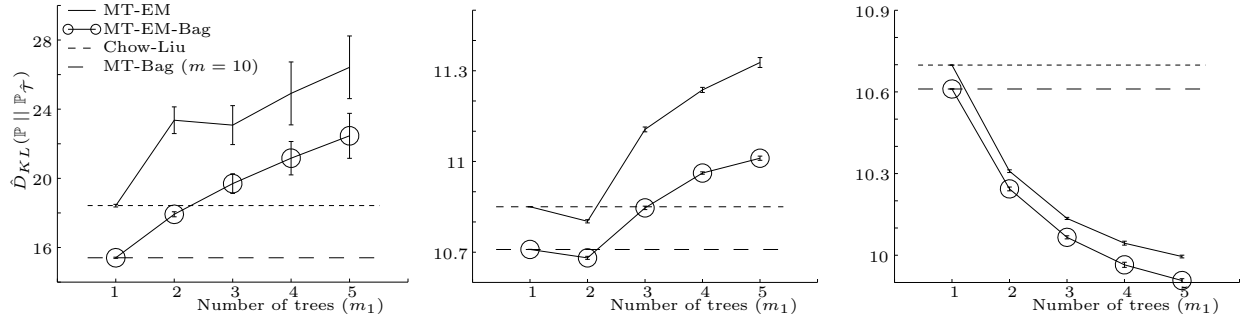
The advantage of the two-level mixtures over MT-EM is significant, as can be seen from the difference between the KL divergence of each method and the KL divergence of the corresponding EM method. This is illustrated in Figure 2, where each box-plot shows the distribution of KL values obtained over 400 runs of MT-EM on a learning set ($N = 400$); similar results are obtained for other sample sizes.

In order to determine which one of the two-level methods is better, Figure 3 displays for each learning sample size the relative number of runs where each method achieves a lower KL divergence than the two others. No definite winner among the two proposed methods can be selected based on those results, but we observe that MT-EM is never better than both (actually, any) of them. However, the interest of using the original learning set to build the first tree of the mixture (MT-EM-BagCl) rather than a bootstrap replicate (MT-EM-Bag) increases with the number of samples.

The edge similarities displayed in Figure 4 reveal an expected increase with the number of samples, but also that the EM algorithm is here the best method. To infer the structure of a mixture of trees, the two-level mixtures are thus not as effective. However, this also indicates that their better accuracy is indeed due to the diversity introduced in each term.

### 5.2 The Target is a Bayesian Network

The proposed algorithms were also tested on distributions encoded by a Bayesian network. The optimal number of terms $m_1$ in the EM method is problem dependent and is e.g. determined by cross-validation. In particular, this value tends to increase with $N$, with the Chow-Liu algorithm ($m_1 = 1$) being better than EM ($m_1 > 1$) for small $N$. In this section we investigate how varying $N$ and $m_1$ impact




(a) $N = 120$ samples (b) $N = 2000$ samples (c) $N = 8000$ samples

Figure 5: Setting 2: On 1 run times 5 distributions times 6 sets, increasing $m_1$ ($m_2 = 10$) reduces the bias and increases the variance. While a mixture is advantageous only for $N$ large enough, MT-EM-Bag is always better than MT-EM.

the difference between the methods and the bias/variance trade-off.

Figure 5 displays for growing values of $N$ the mean accuracy of MT-EM and MT-EM-Bag as a function of $m_1$ and with $m_2 = 10$. Note that when $m_1 = 1$, MT-EM and MT-EM-Bag are respectively equivalent to the Chow-Liu algorithm and MT-Bag (with $m = m_2 = 10$). We outline the results of those methods by two constant lines on the figures. Those results were obtained on 5 networks times 6 data sets for each $N$.

At 120 samples the Chow-Liu algorithm is better than MT-EM: because of the low number of samples, the negative impact of an increased variance is not compensated by the gain in bias when increasing $m_1$. Using MT-EM-Bag rather than MT-EM (with $m_1 = 2$) leads to a better accuracy than Chow-Liu, but is worse than MT-Bag alone. The two-level mixtures can not completely compensate for the lack of samples.

As the number of samples increases, the variance of all models decreases, and the smaller bias of larger mixtures progressively gives them the advantage over Chow-Liu. Constructing an optimal mixture of size 2 is more interesting than a Chow-Liu tree for $N = 2000$ samples. At this point however, reducing the variance of Chow-Liu is still more interesting since MT-Bag is better than MT-EM for all $m_1$.

As more and more samples are available, the variance becomes even lower (MT-Bag is closer to Chow-Liu), and reducing the bias becomes preferable to lowering the variance. At 8000 samples, MT-EM is definitely better than MT-Bag and the KL divergence decreases when $m_1$ increases. Nevertheless, MT-EM-Bag is still better than MT-EM, showing that reducing the variance of each term is still interesting.

Moreover, the gap between MT-EM and MT-EM-Bag is widening as $m_1$ increases. A larger $m_1$ means building each second-level mixture on fewer samples, increasing the variance, a situation favoring it over a single Chow-Liu tree.

## 6 Conclusion

In this work we propose to incorporate the perturb and combine variance reduction approach within the maximum likelihood approach to learn mixtures of Markov trees. The resulting algorithms build a two-level mixture of Markov trees, by replacing each term (a Chow-Liu tree) of a mixture produced by the EM algorithm by a mixture of bagged Markov trees.

The resulting models are shown to improve those constructed by the base EM algorithm. The performance advantage of the two-level mixture of Markov trees over the single level EM mixture increases with the number $m_1$ of terms used in the upper level of the mixture, even if at very low sample size increasing $m_1$ may be detrimental in general (and hence a simple bagged ensemble of Chow-Liu trees is then better). Although we do not study in detail

the effect of the parameter $m_2$, the lower level mixture size, increasing this number can only further improve the advantage of the two-level methods with respect to the single level one.

Several questions remain to be studied. We replace each term by a mixture only after the convergence of the EM algorithm by fear of disrupting its convergence properties and for complexity reasons. Could doing so earlier further limit overfitting?

The number $m_2$ of terms in the randomized sub-mixture is always fixed to 10. Its effect could be further studied with respect to a possible accuracy/inference time trade-off. In particular, since the number of samples influences the gap between the Chow-Liu tree and the mixture of bagged Markov trees, should this parameter differ for each term of the EM mixture, based on the number of samples associated to that term?

## Acknowledgments

François Schnitzler is supported by a F.R.I.A. scholarship. This work was also funded by the Biomagnet IUAP network of the BELSPO and the Pascal2 network of excellence of the EC. The scientific responsibility is the authors'.

## References

S. Ammar, P. Leray, F. Schnitzler, and L. Wehenkel. 2010. Sub-quadratic Markov tree mixture learning based on randomizations of the Chow-Liu algorithm. In *5th European Workshop on Probabilistic Graphical Models*, pages 17–24.

V. Auvray and L. Wehenkel. 2008. Learning inclusion-optimal chordal graphs. In *24th UAI*, pages 18–25.

F. R. Bach and M. I. Jordan. 2001. Thin junction trees. In *Advances in Neural Information Processing Systems 14*, pages 569–576. MIT Press.

L. Breiman. 1996. Arcing classifiers. Technical report, Dept. of Statistics, University of California.

L. Breiman. 1999. Using adaptive bagging to debias regressions. Technical report, Dept. of Statistics, University of California.

D.M. Chickering, D. Geiger, and D. Heckerman. 1994. Learning Bayesian networks is NP-hard. Technical report, Microsoft Research.

C.I. Chow and C.N. Liu. 1968. Approximating discrete probability distributions with dependence trees. *IEEE Trans. Inf. Theory*, 14:462–467.

G.F. Cooper. 1990. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2-3):393–405.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39:1–38.

G. Elidan and S. Gould. 2008. Learning bounded treewidth Bayesian networks. *JMLR*, 9:2699–2731.

S. Kirshner and P. Smyth. 2007. Infinite mixtures of trees. In *24th International Conference on Machine Learning*, pages 417–423.

D. Koller and N. Friedman. 2009. *Probabilistic Graphical Models*. MIT Press.

J. Kollin and M. Koivisto. 2006. Bayesian learning with mixtures of trees. *ECML*, pages 294–305.

J. Kwisthout, H. Bodlaender, and L. van der Gaag. 2010. The necessity of bounded treewidth for efficient inference in Bayesian networks. In *19th European Conference on Artificial Intelligence*, pages 623–626.

H. Liu, M. Xu, A. Gupta H. Gu, J. Lafferty, and L. Wasserman. 2011. Forest density estimation. *JMLR*, 12:907–951.

G.J. McLachlan and T. Krishnan. 2008. *The EM Algorithm and Extensions*, volume 382. John Wiley and Sons.

M. Meila and T. Jaakkola. 2006. Tractable Bayesian learning of tree belief networks. *Statistics and Computing*, 16:77–92.

M. Meila and M.I. Jordan. 2001. Learning with mixtures of trees. *JMLR*, 1:1–48.

J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

F. Schnitzler, Ph. Leray, and L. Wehenkel. 2010. Towards sub-quadratic learning of probability density models in the form of mixtures of trees. In *18th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 219–224.

V. Y.F. Tan, A. Anandkumar, and A. S. Willsky. 2011. Learning high-dimensional Markov forest distributions: Analysis of error rates. *JMLR*, 12:1617–1653.