Freie Universität Berlin
Dept. of Mathematics and Computer Science
and
Max Planck Institute for Molecular Genomics
Computational Molecular Biology

Structure Learning of Conditional Trees

Bachelor Thesis

by
Christoph Hafemeister
June 2006

Contents

1 Introduction

The analysis of gene expression data is a wide field in bioinformatics and a lot of work was put into finding methods that group genes based on their expression profiles. The goal of these clustering methods is to group genes based on their function in the cell.

This work focuses on a method that can be used to analyze gene expression data during cell proliferation and differentiation. [Costa et al., 2006] use a statistical framework called mixture of trees to query gene expression data that was collected during blood cell development. As all blood cells have a common ancestor, the self-renewing hematopoietic stem cell, the lineage of a certain type of blood cell resembles the path of a tree, when one cell differentiates its path is divided in multiple new paths. When querying the data for lineage specific gene expression profiles, [Costa et al., 2006] assumed the structure of the cell development tree to be given.

In this paper algorithms are proposed, which can estimate the structure from the given data, while clustering the genes. The algorithms and models were first introduced by [Chow and Lui, 1968] and [Meila-Predoviciu, 1999] and had to be extended to continuous variables for the purpose of this work. The statistical models used in this work are based on Bayesian networks (BN) which are formally introduced in section 2.2. A BN is a graphical model that intuitively describes the causal relations of variables. The parameters that define the relations of the variables can be learned from data [Dempster et al., 1977] so that the resulting BN resembles the properties of the given data as closely as possible. The structure of a Bayesian network holds the information about how one variable is influenced by the other, or in other words, whether one gains information about one variable, when knowing the other. If the dependencies of the variables and thus the structure of the BN are unknown, it can be learned from the given data [Friedman, 1999]. For a gene expression profile during blood cell development the different time points correspond to variables and the gene expression data represent the different observations. Recovering the dependencies of the time points for all, or groups of genes can give information about blood cell development and which genes mediate cell differentiation. In this work methods for learning the structure of conditional trees, which can be used to query gene expression profiles, are introduced. In order to validate these methods, they were implemented and rum with simulated data.

This paper is organized in the following way. First Bayesian networks and linear Normal distributions as their conditional propability distributions are introduced. Then, structural learning of Bayesian networks with a focus on finding the best tree structure for a set of normally distributed variables is explained. After introducing mixtures of trees, the EM algorithm is extended to two versions of the structural EM. In section 5 the experiments are described and the results are presented. In

the last section the results are interpreted and their meaning is discussed.

2 Basics and notation

2.1 Notation

Random variables are denoted by an uppercase token (e.g. X, Y_i) and a state or value of that variable by the same token in lower case(e.g. x, y_i). Sets of variables are denoted with bold-face capitalized tokens (e.g. $\boldsymbol{X}, \boldsymbol{Y}$) and corresponding sets of values by bold-face lower case tokens (e.g. $\boldsymbol{x}, \boldsymbol{y}$). At last, calligraphic letters (e.g. $\mathcal{G}, \mathcal{B}$) are used to denote statistical models and graphs.

2.2 Bayesian networks

Before defining Bayesian networks and their properties, some graph theory basics are recapitulated. The following definitions are taken from [Diestel, 2005]. An undirected graph is a pair $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ of sets such that $\boldsymbol{E} \subseteq [\boldsymbol{V}]^2$; thus, the elements of $\boldsymbol{E}$ are 2-element subsets of $\boldsymbol{V}$. the elements of $\boldsymbol{V}$ are the vertices (or nodes, or points) of the graph $\mathcal{G}$, the elements of $\boldsymbol{E}$ are its edges (or lines). The usual way to picture a graph is by drawing a dot (or circle) for each vertex and joining two of these dots by a line if the corresponding two vertices form an edge. A directed graph is a pair $(\boldsymbol{V}, \boldsymbol{E})$ of disjoint sets of vertices and edges, where every edge $e = (x, y)$ is assigned an initial vertex x and a terminal vertex y, e is considered to be directed from x to y. The vertex x is called a parent of vertex y. If for every vertex v in a directed graph, there is no nonempty directed path starting and ending in v, the graph is called a directed acyclic graph (DAG). A tree is defined as a DAG, that has exactly one vertex that does not serve as a terminal vertex (the root) and all other vertices have at most one parent.

A Bayesian network (BN) is a graphical structure that represents probabilistic relationships for a given domain. Generally speaking, a BN $\mathcal{B} = (\mathcal{G}, \boldsymbol{\theta})$ consists of two distinct parts: a graphical structure $\mathcal{G}$ and a set of parameters $\boldsymbol{\theta}$ that define probability distributions.

The BN $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ is a DAG with nodes $\boldsymbol{V}$ and directed edges $\boldsymbol{E}$. Nodes stand in a one-to-one correspondence to the set $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ of random variables and will be used interchangeably. The edges are ordered pairs of elements of $\boldsymbol{V}$ and correspond to conditional dependencies between them. $\boldsymbol{Pa}^{\mathcal{G}}(X_i)$ denotes the parents of a node X_i defined by $\mathcal{G}$. Note that if the structure is clear from the context the superior $\mathcal{G}$ is omitted. The graph $\mathcal{G}$ represents conditional independence assumptions that allow decomposition of the joint distribution, reducing the number of parameters. The graph $\mathcal{G}$ holds the Markov Assumption:

(*) Each variable X_i depends only on the variables that directly precede it.

Thus, by applying the chain rule of probabilities, any joint distribution that satisfies (*) can be decomposed into the product form

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | \boldsymbol{Pa}^{\mathcal{G}}(X_i)), \tag{2.1}$$

see Figure 1 for an example.

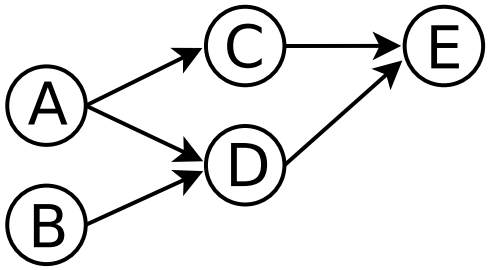

Figure 1: A basic graph structure for a BN. The product form it implies is $P(A, B, C, D, E) = P(A)P(B)P(C|A)P(D|A, B)P(E|C, D)$.

The parameters $\boldsymbol{\theta}$ define conditional probability distributions (CPDs) over $\boldsymbol{X}$ fully specifying the joint distribution. The CPDs $P(X_i|\boldsymbol{Pa}(X_i))$ are specified by a set of parameters $\boldsymbol{\theta}_i$. For a random variable X that is normally distributed (also called Normal or Gaussian) the probability density function is defined by

$$f(x) = \left(\sqrt{2\pi}\sigma\right)^{-1} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right). \tag{2.2}$$

In this case $\boldsymbol{\theta}$ is the mean μ and the variance σ^2. For discrete variables $\boldsymbol{\theta}$ specifies the probabilities of the discrete states resulting in a conditional probability table (CPT). For a more detailed description of Bayesian networks and how to use them, see [Spirtes et al., 1993].

2.3 Linear Normal distributions as CPDs

The previous described Normal distribution is fully specified by the variables μ and σ^2. The CPDs in a BN also have to depend on the values of the parents. In this section, the linear Normal distribution, first introduced by [Lauritzen and Spiegelhalter, 1990], is described, which is used throughout this work.

Linear Normal distributions differ in their distribution from the previous described Normals, as that they estimate a linear fit on their parent. Let $\mathcal{G}$ be a tree, so that each node has at most one parent, then the distribution of node B with $Pa(B) = A$ is

$$f(b|a) = \left(\sqrt{2\pi}\sigma_{B|A}\right)^{-1} \exp\left(\frac{-(b - w_{B|A}a - \mu_{B|A})^2}{2\sigma^2_{B|A}}\right) \tag{2.3}$$

and the following holds,

$$w_{B|A} = \frac{\mathrm{COV}(B, A)}{\mathrm{VAR}(A)}, \tag{2.4}$$

$$\mu_{B|A} = \mu_B - \mu_A w_{B|A}, \text{and} \tag{2.5}$$

$$\sigma^2_{B|A} = \mathrm{VAR}(B) - w^2_{B|A}\mathrm{VAR}(A). \tag{2.6}$$

The meaning of the above definitions is best described with an example. Let A and B be two random variables that are expected to have a Normal distribution and $Pa(B) = A$. $N = 1000$ observations from these variables are drawn. Figure 2 shows the observed values plotted versus each other and their histograms. The parameters μ and σ^2 can be calculated from the give data and the resulting density functions are also depicted in Figure 2. The covariance $\mathrm{COV}(B, A)$ measures how much the two variables B and A vary together from their means. It takes positive values if the values of B and A tend to vary together in the same direction and negative values if they vary in the opposite directions. Thus, $w_{B|A}$ measures the strength and direction of the linear relationship of B and A. In this example A and B have a positive linear relationship and $w_{B|A} = 0.4$. The resulting probability density function of the linear Normal distribution of B given A can be seen in Figure 3. As a result, the CPD for the node B given its parent A is fully described by $f(b|a)$, the probability of B taking on value b, given the value a of its parent A.

In a tree the root node has no incoming edges and $Pa(root) = \emptyset$. Let $w_{root|\emptyset} = 0$, thus the CPD of the root is normally distributed as described by (2.2).

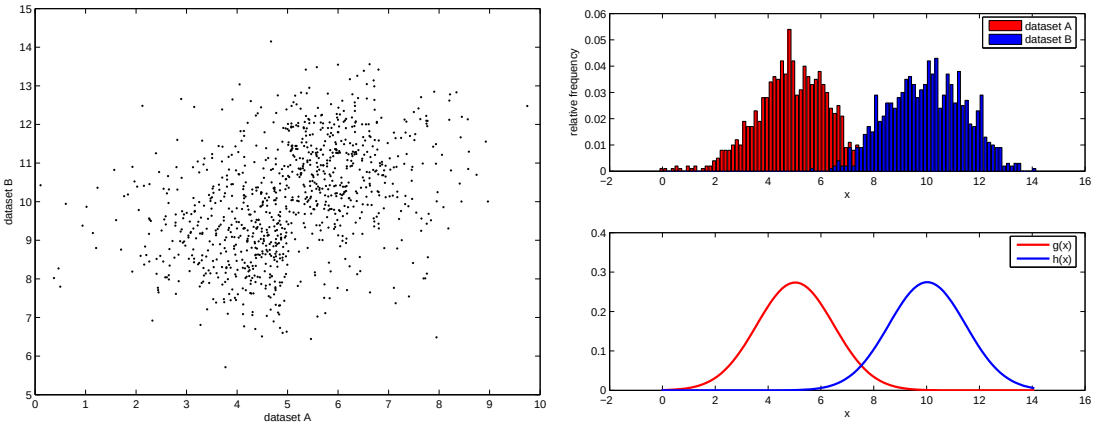

Figure 2: *left:* The values of dataset A plotted versus the values of dataset B. *right:* The histograms for both datasets and the probability density functions $g(x)$ and $h(x)$ of the Normal distributions that fit best to the given data. $g(x)$ has the parameters $\mu = 5$ and $\sigma^2 = 2.12$, $h(x)$ has the parameters $\mu = 10$ and $\sigma^2 = 2.11$.

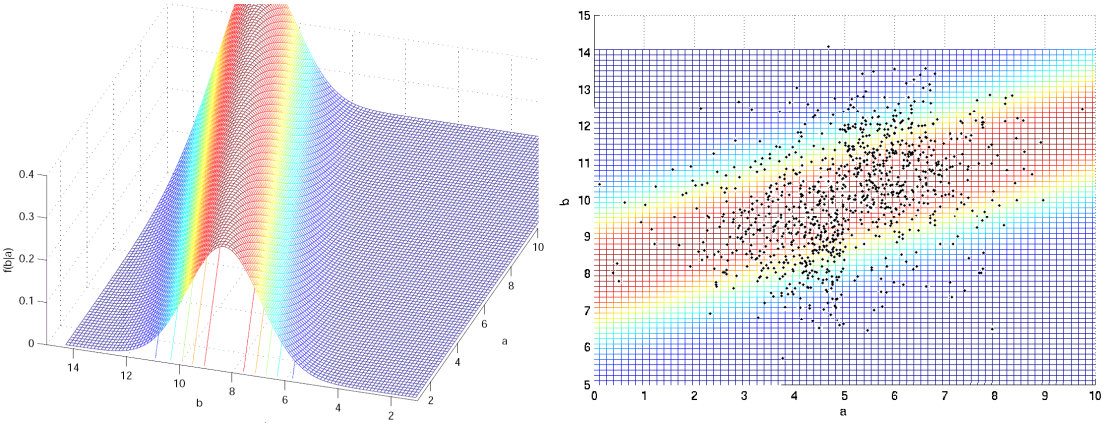

Figure 3: *left:* The probability density function $f(b|a)$ of the linear Normal distribution of B given the parent A. The parameters are $w_{B|A} = 0.4$, $\mu_{B|A} = 7.92$ and $\sigma^2_{B|A} = 1.77$. *right:* The datasets plotted against each other and $f(b|a)$ color coded in the background.

3 Structural learning

3.1 Finding the most likely structure

Learning the structure of a BN is a model selection problem in the sense that each possible structure corresponds to a model for which the parameters have to be estimated, and one model needs to be selected based on the data. As shown by [Robinson, 1976], the number of possible structures for a BN grows super-exponential to the number of vertices. For a Bayesian network having 5 nodes 29281 possible structures exist, for 10 nodes this number grows to $\simeq 4.2 \times 10^{18}$. Therefore, the problem of finding the best structure can require extensive computational power, but many heuristics have been proposed, see [Friedman, 1999] for an example. Additional information, like the order of vertices, can reduce the size of the search space significantly. In this work the search space of possible structures is reduced to tree structures. Finding the best tree structure for a given set of vertices is a much simpler task and can be solved by well known algorithms. Let $\mathcal{G}_d$ be the family of DAGs with maximum number of parents equal to d. Then it is NP-hard to find

$$\mathcal{G}^* = \arg\max_{\mathcal{G}\in\mathcal{G}_d} score(\mathcal{G}, D) \tag{3.1}$$

for any $d \geq 2$ where D is the set of nodes and *score* some scoring function e.g. the Likelihood or the BIC for the resulting Bayesian network [Chickering and Heckerman, 1994]. For trees the maximum number of parents $d \leq 1$ holds and the problem can be solved in $\mathcal{O}(n^2)$ time using maximum weight spanning tree (MWST) algorithms, as proposed by [Chow and Lui, 1968]. Before presenting the algorithms needed for finding the best tree structure for a given set of data in section 3.3, mutual information for normally distributed variables is introduced.

3.2 Mutual information of normally distributed variables

Using the Kullback-Leibler divergence [Chow and Lui, 1968] show that the overall likelihood for a directed or undirected tree can be maximized by maximizing the sum of weights of all edges, where the weight W of an edge is defined by the mutual information. Let X and Y be two continuous random vectors, then the mutual information is defined as the distance from independence between X and Y measured by the Kullback-Leibler divergence [Cover and Thomas, 1991, Kullback and Leibler, 1951]. Given two densities $p(x)$ and $q(x)$ the Kullback-Leibler divergence KL is defined by

$$KL(p(x), q(x)) := \int p(x) \log \frac{p(x)}{q(x)} \mathrm{d}x \tag{3.2}$$

with the convention that $0 \log(0/0) := 0$. Let $p(x, y)$ be the joint density of the two random vectors and $p(x)$ and $p(y)$ their marginal densities. The mutual information I between X and Y is defined by

$$I(X, Y) := KL(p(x, y), p(x) \otimes p(y)), \tag{3.3}$$

where $p(x) \otimes p(y)$ is the tensor product of the two marginal densities. The above definition shows, that the mutual information is symmetric and, by applying the Jensen inequality to the Kullback-Leibler divergence, one can see, that the mutual information is always non negative and zero if and only if X and Y are stochastically independent. The Kullback-Leibler divergence is a natural distance measure from a "true" probability distribution p to an arbitrary probability distribution q.

Mutual information can also be interpreted as the information obtained from the differential entropy H, which extends the idea of the Shannon entropy to continuous probability distributions.

$$H(f(x)) = -\int f(x) \log f(x) \mathrm{d}x, \tag{3.4}$$

where f is a probability density function and with the convention that $0 \log 0 := 0$. With respect to the differential entropy, the mutual information between X and Y can be rewritten as

$$I(X, Y) = H(p(x)) - E_{p(y)}(H(p(x|y))) = H(p(y)) - E_{p(x)}(H(p(y|x))) \tag{3.5}$$

[DeGroot, 1962, Morales et al., 1996]. Thus, the mutual information can be interpreted as the reduction in the uncertainty of X (resp. Y) due to the knowledge of Y (resp. X) [Ullah, 1996]. The expectation in (3.5) can be rewritten as $E_{p(y)}(H(p(x|y))) = H(p(x, y)) - H(p(x))$, and therefore

$$I(X, Y) = H(p(x)) + H(p(y)) - H(p(x, y)). \tag{3.6}$$

For a normally distributed random variable X the differential entropy is

$$H(X) = \frac{1}{2} \log(2\pi e \sigma_X^2) \tag{3.7}$$

and for the two-dimensional case with another Normal Y

$$H(X, Y) = \frac{1}{2} \log(2\pi e)^2 \left|K\right|, \tag{3.8}$$

where $|K|$ is the determinant of the covariance matrix K.

$$K = \begin{bmatrix} \sigma_X^2 & \sigma_{XY} \\ \sigma_{YX} & \sigma_Y^2 \end{bmatrix}$$

Substitution of equations (3.7) and (3.8) into (3.6) results in

$$I(X,Y) = -\frac{1}{2}\log(2\pi e)^2 |K| + \frac{1}{2}\log 2\pi e\sigma_X^2 + \frac{1}{2}\log 2\pi e\sigma_Y^2 \tag{3.9a}$$

$$= -\frac{1}{2}\log\frac{|K|}{\sigma_X^2\sigma_Y^2} \tag{3.9b}$$

for two normally distributed variables X and Y. Thus, given the covariance of two variables with Normal distribution, their mutual information can be computed with little computational cost.

3.3 Finding the best tree structure

In the previous subsection it was shown how to compute the mutual information of two normally distributed variables. Now this information will be used to find the best tree topology for a given set of variables V. Given the mutual information of each pair of the variables V as a weight matrix $W_{ij} = I(V_i, V_j)$, the tree structure with the highest likelihood will be the one maximizing the sum of all edge weights[1]. There are two commonly used MWST algorithms, Prim's algorithm and Kruskal's algorithm [Cormen et al., 1990], both of which are greedy algorithms. Prim's algorithm constructs an unrooted MWST given a set of edges E, nodes V and a weight matrix W. Prim's algorithm is explained in Algorithm 1.

Algorithm 1: Prim's MWST algorithm

input : nodes V, edges E, edge weights W
output: maximum weight spanning tree T

1 $T = (\emptyset, \emptyset)$
2 pick random node $r \in V$ and add it to T
repeat
3 select edge E_{uv} where $u \in T$ and $v \notin T$ and W_{uv} is maximum
4 add E_{uv} and v to T
until *all nodes V are in T*

The algorithm picks a random node $r \in V$ to start with and adds it to the empty spanning tree T. As long as not all nodes V are in T, all edges connecting a node in T with a node not in T are examined. The edge with the highest weight and the node that it connects are then added to T. The output of the algorithm is an undirected graph that connects all nodes and has no cycles, a complete tree. The algorithm runs in time which is $\mathcal{O}(n^2)$, using a binary heap this time can

[1]Since the goal is to maximize the likelihood a *maximum* weight spanning tree algorithm is used, instead of multiplying the edge weights by -1 and calling it *minimum* spanning tree.

be reduced to $\mathcal{O}(E \log V)$ where E is the number of edges and V the number of vertices, see [Aho et al., 1983] for a more detailed description.

4 Structural EM with mixture of trees

In the previous section it was explained how to find the tree structure which maximizes the likelihood for a Bayesian network with normally distributed variables as nodes. The mutual information of all pairs of variables had to be calculated and it was also shown how to do this, given the covariance matrices.

In this section a combination of multiple Bayesian networks with tree structures, called mixtures of trees, is introduced and the algorithms for learning the structure and parameters of such a model are explained.

4.1 Mixtures of trees

The combination of K trees in a mixture by adding a discrete hidden variable Z was introduced by [Meila-Predoviciu, 1999]. The mixture of trees $Q(x)$ is defined as

$$Q(x) = \sum_{k=1}^{K} \lambda_k T_k(x) \tag{4.1}$$

with $\lambda_k \geq 0$ for $k = 1, \ldots, K$ and $\sum_{k=1}^{K} \lambda_k = 1$. The mixture coefficients λ_k represent the probability that for a mixture component T_k the hidden choice variable z takes the value $k \in \{1, \ldots, K\}$. In respect to the value of z the visible nodes form a tree. The K different trees may have different structures and different parameters. As the structures of these trees depends on the value of z, they can be called conditional trees. The mixture components T_k are Bayesian networks with tree structures as described in section 2.2. In this work, linear Normal distributions are used as CPDs. Thus, for every node U with parent V, $w_{U|V,k}$, $\mu_{U|V,k}$ and $\sigma^2_{U|V,k}$ must be specified.

4.1.1 Covariance of conditional normally distributed variables

As this paper focuses on conditional tree models, it is necessary to compute the covariance of two conditional normally distributed variables in order to calculate their mutual information. Given a normally distributed variable Y with normally distributed parent X, and a discrete variable Z with Z being the parent of both of them, [Murphy, 1998] shows how to estimate the covariance matrix. See Figure 6 for the graphical representation of this BN. Assume e_t for $t = 1, \ldots, N$ training cases to be given.Let the share $s_{t,i}$ be the posterior probabilities $\mathrm{P}(Z = i|e_t)$, then

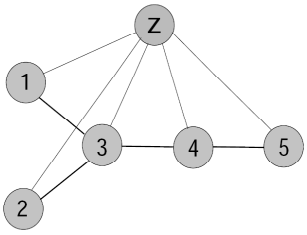

Figure 4: A mixture of trees sharing the same structure, taken from [Meila-Predoviciu, 1999].

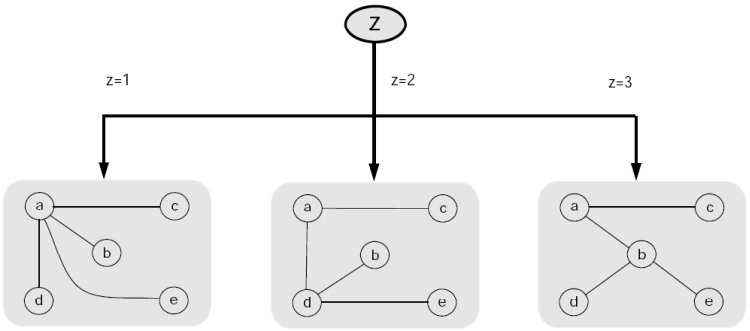

Figure 5: A mixture of trees with different structures with $K = 3$ and 5 nodes, taken from [Meila-Predoviciu, 1999].

is

$$\mathrm{COV}(X,Y)_i = \frac{\sum_t s_{t,i}\, x_t\, y_t}{\sum_t s_{t,i}} - \left(\frac{\sum_t s_{t,i}\, x_t}{\sum_t s_{t,i}} \frac{\sum_t s_{t,i}\, y_t}{\sum_t s_{t,i}}\right) \tag{4.2}$$

the covariance of X and Y and

$$\mathrm{VAR}(X)_i = \frac{\sum_t s_{t,i}\, x_t^2}{\sum_t s_{t,i}} - \left(\frac{\sum_t s_{t,i}\, x_t}{\sum_t s_{t,i}}\right)^2 \tag{4.3}$$

the variance of X and

$$\mathrm{VAR}(Y)_i = \frac{\sum_t s_{t,i}\, y_t^2}{\sum_t s_{t,i}} - \left(\frac{\sum_t s_{t,i}\, y_t}{\sum_t s_{t,i}}\right)^2 \tag{4.4}$$

the variance of Y in respective to $Z = i$. With these equations, a covariance matrix for every possible value of Z can be computed. How much an observation affects the variance and covariance depends on its value for $s_{t,i}$. If the probability of Z having value i given the observation e_t is small, then this observation has little influence on the estimation of the covariance matrix for $Z = i$.

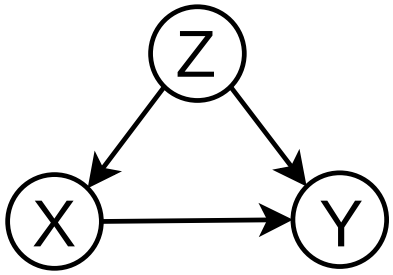

Figure 6: Conditional normally distributed variables X and Y with discrete parent Z.

4.1.2 Tree learning with conditional normally distributed variables

As shown in the previous subsection, K multiple covariance matrices exist for conditional variables that all have the same discrete parent Z and $Z = 1, \ldots, K$. To find the tree structure with the highest likelihood for a set of conditional normally distributed variables given the value of its discrete parent, first compute the covariance matrix as described in 4.1.1. Given the covariances, compute all pairwise mutual information and use a MWST algorithm with the mutual information as weights [Meila-Predoviciu, 1999]. Algorithm 2 summarizes the required steps.

Given the dataset D and the posterior probabilities w, see 4.1.1, the algorithm first computes the mutual information I of all node pairs. Then, the MWST algorithm with I as input finds the best tree structure T.

Algorithm 2: TreeLearn

input : posterior probabilities w, dataset D with variables V
output: tree T maximizing likelihood

1 Compute mutual information $I(u, v)$ for $u, v \in V$ using w and D in equations from 4.1.1
2 T = MWST(V, all possible edges, I)

4.2 EM algorithm

The hidden choice variable Z of a mixture of trees model is previously unknown but can be marginalized after all other parameters have been learned from data. One algorithm that maximizes the complete likelihood when incomplete data is given is the Expectation-maximization (EM) algorithm from [Dempster et al., 1977].

The algorithm starts from some initial parameter set θ^0, e.g. randomly chosen. In the expectation step (E step) the sufficient statistics are estimated from the observed data by using the current estimate θ^t. The maximization step (M step) then takes the estimated parameters and computes the MLE. The result is the new estimate θ^{t+1} that is then used as the estimate for the next iteration.

Given the estimated parameters, it is then possible to compute the posterior probability of $z = k$ for $k = 1, \ldots, K$ for every observation of the given dataset. For gene expression data where g_i for $i = 1, \ldots, N$ is the ith gene of N, each gene could be assigned a cluster by selecting a z so that $z_i = \arg\max_{1 \leq k \leq K} P(z_i = k | g_i, \theta_k)$. In the following sections the terms 'number of mixture components' and 'number of clusters' are used interchangeably.

4.2.1 Structural EM with mixed structures

To also learn the structure of a conditional tree while learning its parameters, the TreeLearn algorithm (see Algorithm 2) has to be embedded in the EM algorithm.

The MixTree algorithm takes as input the initial mixture of trees model Q with K mixture components T and a dataset g with N observations. The number of variables in the dataset has to match the number of nodes of the mixture components. The output of the algorithm is the new model Q^* with newly learned structures for the mixture components T^* and estimated model parameters θ^*.

First, the algorithm picks some structures and parameters for the mixture components to start with. Then, the parameters of the unobserved data are estimated given the current estimate (E step). For every mixture component the probability of every data observation that it was generated by this component, given the current model and estimate, is computed. These posterior probabilities and the given data is then used to learn the structure of that component with the TreeLearn

Algorithm 3: MixTree

input : initial model Q, dataset g with N observations
output: Q^* with MLE θ^* and learned structures T^*

pick initial structures T_k and parameters θ_k for Q, $k = (1, \ldots, K)$
repeat
    E step: estimate sufficient statistics from g and current θ, given Q
    for *k=1* **to** K **do**
        compute marginals $m_i = \mathrm{P}(z = k|g_i, \theta_k)$ for every observation g_i, $i = (1, \ldots, N)$
        T_k^* =TreeLearn(m,g)
    endfor
    M step: compute MLE θ^*
until *convergence*

algorithm, see Algorithm 2. The M step follows and a new estimate is computed by maximizing the likelihood for the parameters given the data and the sufficient statistics from the E step. These steps are repeated until the complete likelihood and the parameters of the model converge.

4.2.2 Structural EM with shared structure

If all K mixture components are to share the same structure the MixTree algorithm has to be adapted to this requirement. Algorithm 4 shows the steps for the EM algorithm for finding the shared structure.

This algorithm differs from the one with different structures, as that the mutual information of every variable pair are summed up for all K mixture components in a distance matrix S. This matrix is then used in the MWST algorithm to compute one tree that best compromises the structure for all components. This structure can be seen as an average structure of all components.

Algorithm 4: MixTreeS

input : initial model Q, dataset g with N observations and D variables V
output: Q^* with MLE θ^* and learned structure T^*

pick initial structure T and parameters θ_k for Q, $k = (1, \ldots, K)$
repeat
  E step: estimate sufficient statistics from g and current θ, given Q
  distance matrix $S_{u,v} = 0$ for $u, v \in V$
  for *k=1* **to** K **do**
    compute marginals $m_i = \mathrm{P}(z = k|g_i, \theta_k)$ for every observation g_i, $i = (1, \ldots, N)$
    compute mutual information $I_{u,v}$ for $u, v \in V$ given m as described in 3.2
    add I to S, $S^* = S + I$
  endfor
  $T^* = \mathrm{MWST}(V, \text{all possible edges}, S)$
  M step: compute MLE θ^*
until *convergence*

5 Experiments

In order to validate the proposed algorithms, they were implemented and tested in various experiments. Mixture of trees models were generated and sample data drawn from them. Using the structural EM algorithms it was tried to recover the parameters and structure of the original models. These estimated models were then compared to the original models to see how well the algorithms performed.

5.1 Method

The mixture of trees and the structure learning algorithms were implemented in Matlab 7 using the Bayesian Network Toolbox (`http://bnt.sourceforge.net/`).

For given datasets the mixture of trees models with the highest likelihood were estimated using the structural EM algorithms. Linear Normal distributions as described in section 2.3 were used as CPDs for the continuous variables. Random parameters and structures were picked for the initial models of the EM algorithm. After learning the parameters and structures of the mixture components, the data samples were clustered. The posterior probability of $z_i = k$ for $k = 1, \ldots, K$ for every sample d_i was computed and d_i was assigned cluster $c_i = \arg\max_{1 \leq k \leq K} P(z_i = k|g_i, \theta_k)$. For clustering the data with K-means the implementation of the Matlab Statistics Toolbox was used.

5.1.1 Data

For the experiments simulated data was used. Mixture of trees models were generated and datasets sampled from them. The used tree topology was generated randomly, for the mixed structure datasets this was done separately for each mixture component. If not stated otherwise, the following parameters were randomly chosen: $\mu_{U|V,k}$ from the range $[-1.5, 1.5]$, $\sigma^2_{U|V,k}$ from $[0, 1]$ and $w_{U|V,k}$ from the ranges $[-0.7, -0.2]$,$[0.2, 0.7]$. 10 to 25 mixtures were generated for each K and a total of $K\lfloor 1000/K \rfloor$ samples were drawn from each model.

5.1.2 K-means

In order to compare sensitivity and specifity of the discussed methods, K-means was also used to cluster the datasets. For each dataset the clustering was repeated 10 times and the best solution was kept. Squared Euclidean distance was used and the maximum number of iterations was 100.

5.2 Results

5.2.1 Mixture of trees with same structure

For $K = (3, 5, 10, 11, 12, 13, 14, 15)$ 25 mixtures were generated and from each of these 800 models a dataset was sampled. Each dataset was processed 10 times and the result with the highest likelihood was kept. In all cases the original structure was found, recovering every edge of the models used to generate the data. In addition, 10 mixtures with $K = (3, 5, 10, 11, 12, 13, 14)$ and higher absolute values for $w_{U|V,k}$ were generated. The w values were picked randomly from the ranges $[-1, -0.5]$,$[0.5, 1]$.

Figure 7 shows the sensitivity and specifity of the clustering using the mixtures of trees method compared to K-means. For a low number of clusters, $K \leq 10$, the mixture of trees clustering is more accurate than K-means and sensitivity and specifity are ≥ 0.9. For higher numbers of clusters, the number of samples from each cluster becomes too low and the clustering performance drops. Sensitivity and specifity of both methods drop to values between 0.8 and 0.9, with K-means being a little better than the mixture of trees method.

Figure 8 shows the results for datasets with stronger linear dependencies between their variables. In this scenario, the clustering performance of the mixture of trees method remains accurate for higher numbers of clusters. Sensitivity and specifity are about 0.1 higher for the mixture of trees method than for K-means.

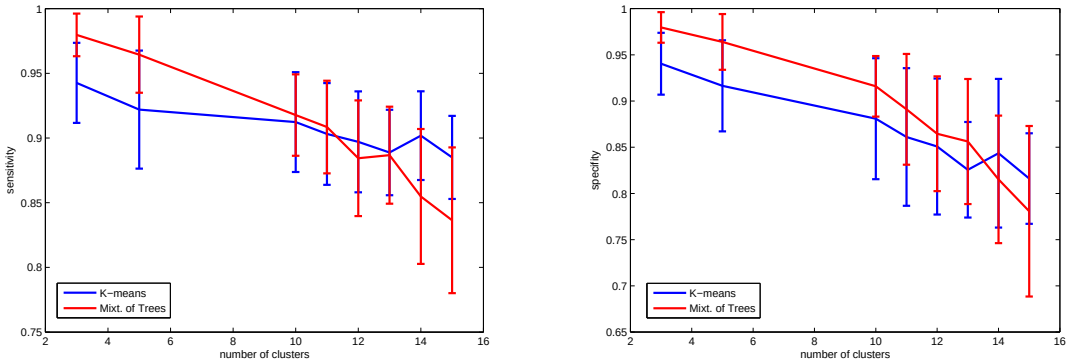

Figure 7: Sensitivity and specifity of the estimated mixture of trees model sharing one structure for $K = (3, 5, 10, 11, 12, 13, 14, 15)$, compared to K-means. The original structure was recovered in all cases.

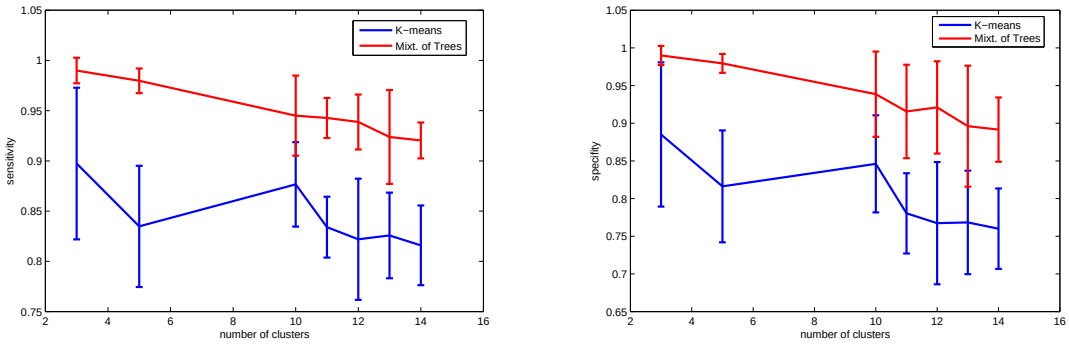

Figure 8: Sensitivity and specifity of the estimated mixture of trees model sharing one structure, compared to K-means. For $K = (3, 5, 10, 11, 12, 13, 14)$ 10 datasets with higher absolute values for $w_{U|V,k}$ were generated. They were picked randomly from the ranges $[-1, -0.5]$,$[0.5, 1]$. The original structure was recovered in all cases.

5.2.2 Mixture of trees with different structure

For $K = (3, 5, 10)$ 10 mixtures were generated. Each dataset was processed 10 times and the result with the highest likelihood was kept.

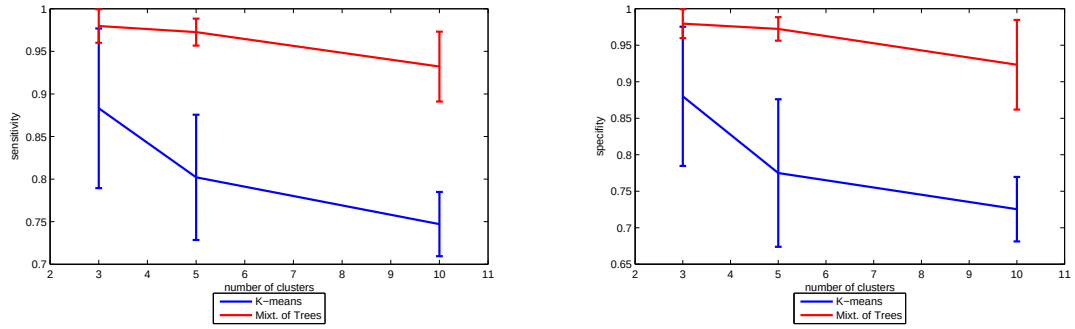

Figure 9: Sensitivity and specifity of the estimated model with different structures for the $K = (3, 5, 10)$ mixture components, compared to K-means.

For this model with a different randomly generated structure for each cluster the right structure was not always recovered. To measure how well an estimated structure compares to the original structure that was used when generating the data, wrong edges and their difference in mutual information were counted. A wrong edge is an edge connecting two nodes which in the original structure do not have an edge between them. As the original structures are complete trees, there must be one edge missing for every wrong edge in an estimated structure. The difference in mutual information MI_{dif} of the original model o and the estimated model e is defined as

$$MI_{dif}(o, e) = \left| \sum_{wrong\, edges(e)_o} I_{wrong\, edge(e)} - \sum_{missed\, edges(e)_o} I_{missed\, edge(e)} \right| \tag{5.1}$$

the absolute value of the difference of the mutual information of all wrong edges and all missed edges of the estimated model given the original model. This value will indicate how much different (in terms of mutual information) the falsely picked edges are than the true ones.

Sensitivity and specifity for the clustering with the mixture of trees model is above 0.95 for 3 and 5 clusters and above 0.9 for 10 clusters. K-means is significantly less accurate, especially for a higher number of clusters. For 10 clusters, sensitivity and specifity of K-means is around 0.75. See Figure 9 for more details.

The ratio of wrong edges in an estimated mixture of trees model increased with the number of clusters. For a model with 3 clusters, less than 5% of the recovered edges were wrong. This value increases to 18% for a model with 10 clusters, which

has a sample size of 100 per cluster. This means, that on average, 12 to 13 of the 70 edges of the mixture components are wrong for an estimated mixture of trees model with $K = 10$. Figure 11 summarizes these results. It also shows that the difference in mutual information of the estimated models compared to the original models grown with the ratio of wrong edges.

For an example of the structure and the mutual information of an estimated component see Figure 10.

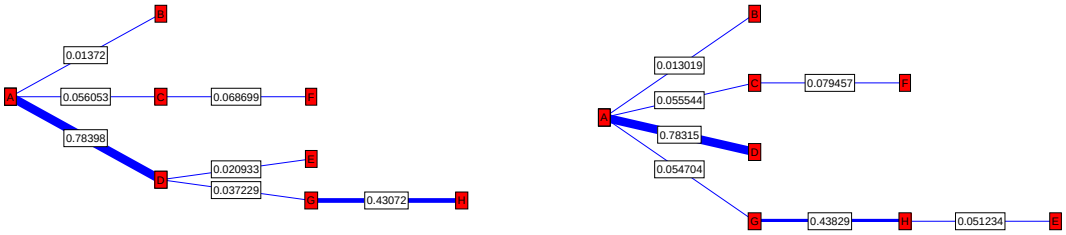

Figure 10: Structure and mutual information of one cluster from a model with mixed structures and $K = 10$. *left:* The original structure and mutual information *right:* Estimated structure and mutual information. The found structure has 2 wrong edges compared to the structure used when generating the data. The difference in mutual information is 0.0035 .

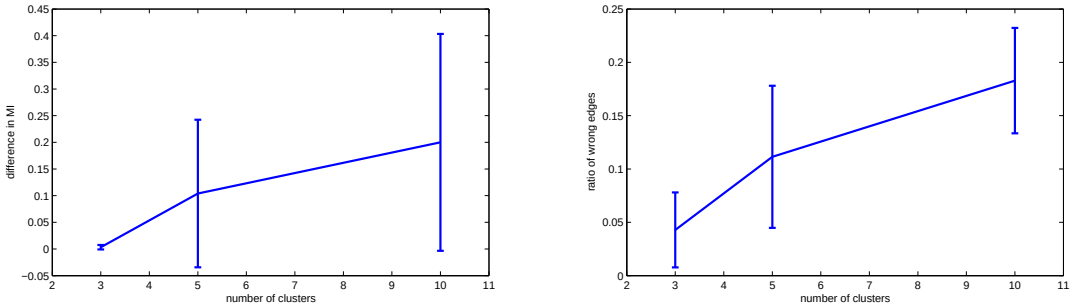

Figure 11: *left:* Average Difference in mutual information per component of the estimated mixture, compared to the original structure. *right:* Ratio of wrong edges in found structure compared to original structure. All values are averages of 10 datasets for each number of clusters.

5.2.3 Evaluating an estimated structure

Once the two presented algorithms have found a structure for a cluster or a group of clusters, it is interesting to know how confident one can be that this structure is supported by the data. The MWST algorithm will always return a complete tree

but with real life data, the found structure could be a result of chance. In order to separate spurious from significant results, one needs to validate the estimated model.

One way to analyze the found structure is to look at the mutual information of the found edges and compare it to the mutual information of nodes that are not connected.

For data with a high structural background the mutual information of dependent variable pairs should be higher than that of variable pairs that are not connected in the structure. As a result the mutual information of variables sharing an edge in the estimated structure is considerably higher than that of variables not connected, see Figure 12. If there is no structure underlying the data, all variables are independent and identically distributed (i.i.d.) random variables and all mutual information is very close to 0. As can be seen in Figure 13 all mutual information values are near 0 and there is no significant difference between mutual information of connected nodes and not connected nodes.

Figure 14 and Figure 15 are examples of mutual information histograms of an estimated structure with some wrong edges and an estimated structure that contains no wrong edges. These examples demonstrate how the inspection of the mutual information can help to verify the learned structure. By closely looking at the mutual information of found edges and comparing it to the pairwise mutual information of nodes that are not connected by an edge, it is possible to estimate the possibility of wrong edges.

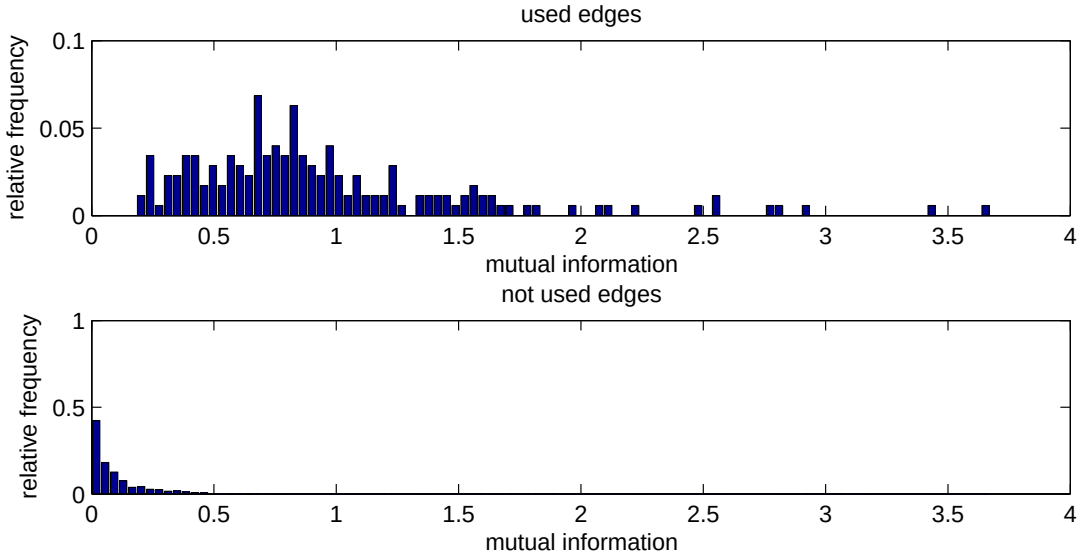

Figure 12: Histogram of the mutual information of used edges and not used edges of the generating models. For all 25 datasets with $K = 5$ and same structure (for parameters see 5.1.1) the nodes and their mutual information were observed. *top:* 175 true edges. *bottom:* 525 pairwise mutual information that does not correspond to an edge.

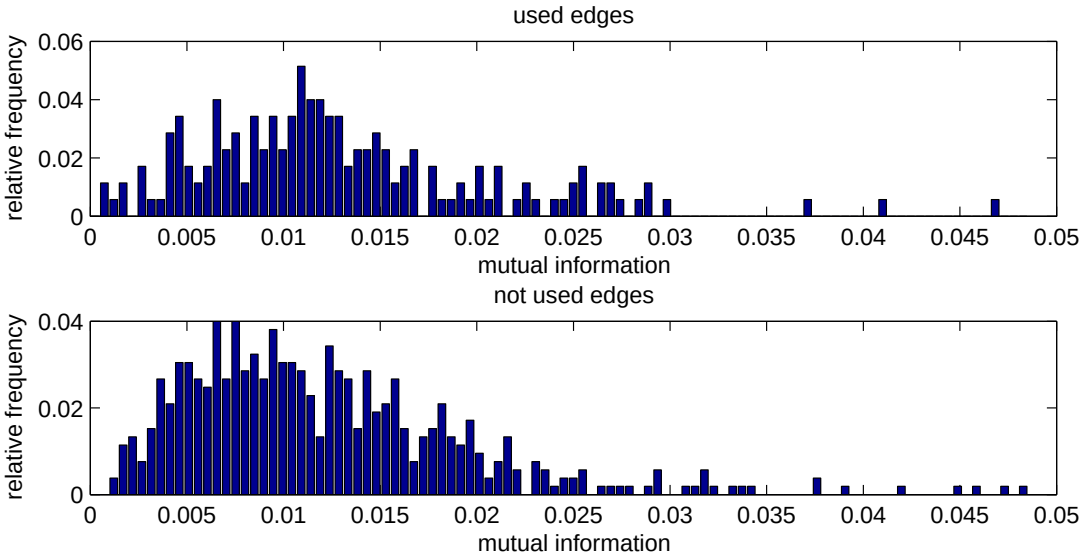

Figure 13: Mutual information frequency of used edges and not used edges of generated models. For $K = 5$, 25 datasets with $w_{U|V,k} = 0$ and shared structure were generated and the nodes and their mutual information were observed. *top:* 175 true edges. *bottom:* 525 pairwise mutual information that does not correspond to an edge.

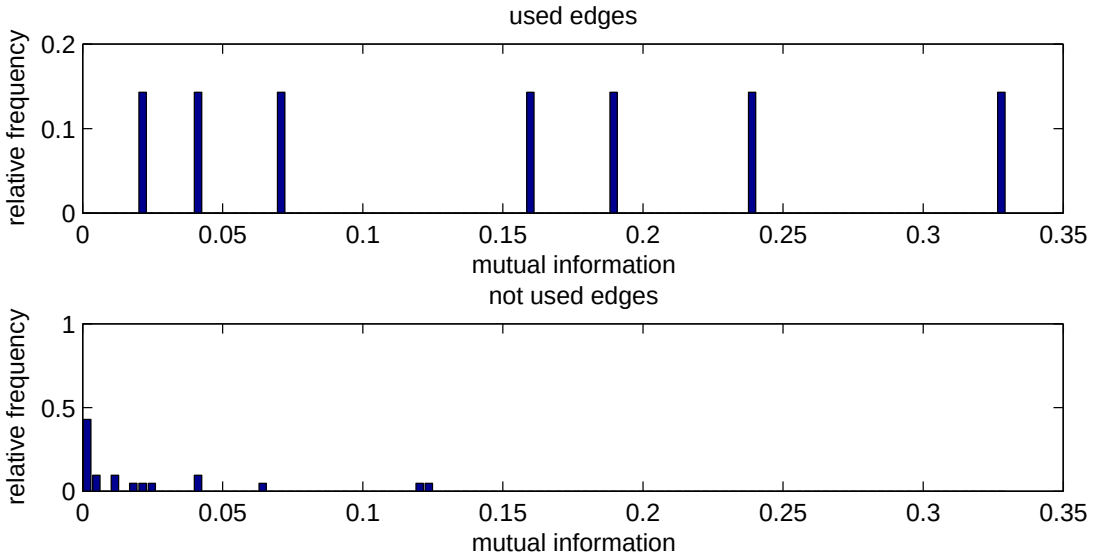

Figure 14: Mutual information frequency of used edges and not used edges of a learned model. One cluster of the model with $K = 10$ and mixed structure was analyzed. Note, that with only 28 pairwise mutual information there is not much data do examine the structure with, but the overlap in mutual information of used and not used edges can be an indicator for wrong edges. In this example cluster the estimated structure had 3 wrong edges.

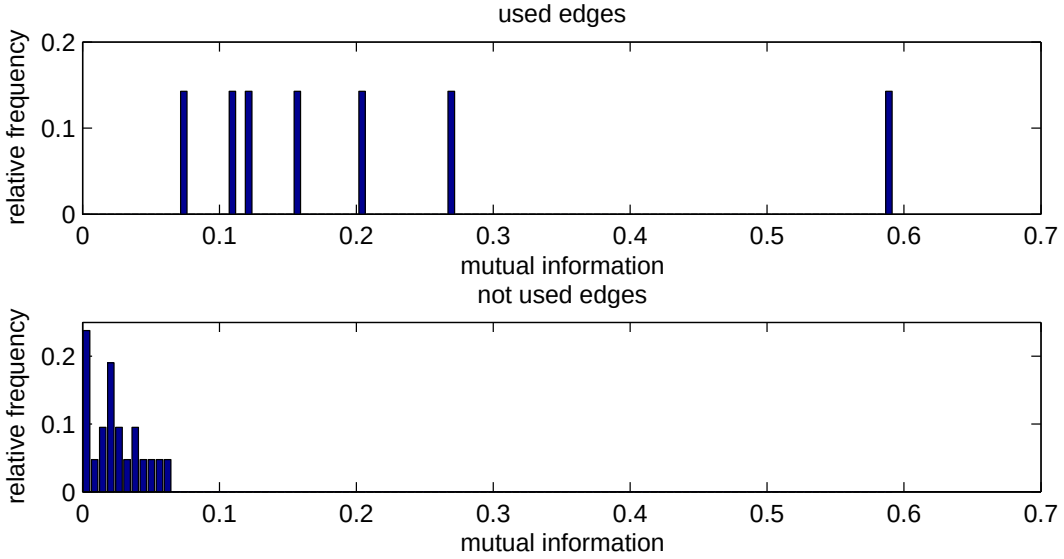

Figure 15: Mutual information frequency of used edges and not used edges of a learned model. One cluster of the model with $K = 10$ and mixed structure was analyzed. The structure of this cluster matched the one of the original one and did not contain wrong edges.

6 Discussion

The results have shown that finding the right tree topology for a mixture of conditional trees that share the same structure is a solvable task. This must be due to the fact that the structural information which is encoded in every observation is fully used. The whole dataset holds the same information about the structure, and during the structure learning this information is used, independently from the learning of the hidden variable. In this case the learning of the structure is independent of the final cluster assignment and the number of clusters, but depends mainly on the number of observations and on the strength of the regression (the w). As the number of clusters increases, the number of observations per cluster decreases, making it harder to reliably estimate the model parameters. The effect of inexact parameters is an inaccurate probability for the hidden choice variable. This explains the stronger decrease of sensitivity and specifity for the mixture of trees model compared to K-means.

For higher absolute values of w the Euclidean distance of the variables decreases and the clustering of K-means becomes less accurate. The structural learning is easier and faster in this case and helps finding the right clusters in the Bayesian approach.

For mixture of trees with different structures, finding the right structures is a more difficult task. When the number of samples for each cluster is too low (100 in the described experiments), wrong are likely to be introduced in the estimated structures. The number of observations per cluster is critical for finding the structure of the component that was used to generate the data. With only little evidence given, the computated mutual information might not reflect the true value. With these limitations in mind, a ratio of 18% of wrong edges for the model with 10 clusters and 100 samples for each cluster, is a good value. Clustering is easier with a mixture of structures, because every cluster has a different set of w values and this additional information is used when calculating the probability of the hidden choice variable. The observations can be distinguished by more parameters than in the shared structure case. For K-means this additional parameter is of no use, instead it acts as noise to the data. The structure learning for the model with mixed structures results in a significantly better clustering of the data than K-means.

As the results show evaluating a found structure can be challenging. By looking at the mutual information it is possible to distinguish between data that has no structural background (i.i.d.) and data that does imply structural dependencies, but it is considerably harder to be confident that a found structure is free of errors. Rather than trying to find the "one true" structure, methods for finding a possible group of structures should be developed in the future. This approach can then be extended to give a probability distribution over a group of structures for every

component.

Using this method for querying gene expression data of blood cells during their development, could lead to interesting insights about gene regulation. Regarding the estimated structures, the above mentioned limitations have to be kept in mind. The datasets could be too small and resulting tree topologies spurious. Despite that, the clustering using the EM algorithm with structure learning yields better results than the mixture of trees method for a fixed topology, and can so lead to a more accurate grouping of the genes. Results of not published experiments have shown that the mutual information matrices for an estimated component can be used to gain additional information about the relation of groups of genes at different time points. When looking for genes which do not differ in their expression at two different time points, comparing mutual information can help identifying those kinds of genes.

In summary, it is shown that structure learning of conditional trees is a well solvable task and its application to gene expression data should be subject to further investigation.

References

A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *Data Structures and Algorithms.* 1983.

D. M. Chickering and D. Heckerman. Learning bayesian networks is np-hard. Technical report, Microsoft Research, Microsoft Corporation, 1994.

C. K. Chow and C. N. Lui. Approximating discrete probability distributions with dependence trees. 14(3):462–467, 1968.

T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms.* MIT Press, 1990.

Ivan G. Costa, Stefan Roepcke, and Alexander Schliep. Gene expression trees in blood cell development. 2006.

T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, 1991.

M. H. DeGroot. Uncertainty, information, and sequential experiments. *The Annals of Mathematical Statistics*, 33(2):404–419, 1962.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Royal Statistical Society Series B*, 39:1–38, 1977.

R. Diestel. *Graph Theory.* Springer-Verlag, 2005.

N. Friedman. The bayesian structural em algorithm. 1999.

S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. pages 415–448, 1990.

M. Meila-Predoviciu. *Learning with mixtures of trees.* PhD thesis, 1999.

D. Morales, L. Pardo, and I. Vajda. Uncertainty of discrete stochastic systems: general theory and statistical inference. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, 6:681–697, 1996.

K. Murphy. Fitting a conditional gaussian distribution, 1998.

R. W. Robinson. Counting unlabeled acyclic digraphs. *Combinatorial Mathematics V: Proceedings of the Fifth Australian Conference, held at the Royal Melbourne Institute of Technology.*, 1976.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search.* 1993.

Ullah. Entropy, divergence and distance measures with economic applications. *Journal Of Statistical Planning And Interence*, 49:137–162, 1996.