Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

F. Schnitzler

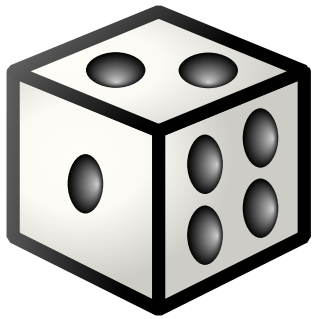
University of Liège

24 September 2012

Density estimation

- Density estimation consists in learning a joint probability density $\mathbb{P}(\boldsymbol{\mathcal{X}})$, based on N realisations of the problem: $\left(\mathbf{x}^i \sim \mathbb{P}(\boldsymbol{\mathcal{X}})\right)_{i=1}^{N}$.

- Example: estimating $\mathbb{P}(\mathcal{X}=$"result of a dice throw"$)$.
  - A realisation belongs to $\{1,2,3,4,5,6\}$.
  - 10 realisations: $D = (2,3,1,3,6,1,4,2,6,2)$.
  - A possible estimate, based on these realisations:
    $\mathbb{P}(\mathcal{X}=1) = 2/10$, $\mathbb{P}(\mathcal{X}=2) = 3/10$, $\mathbb{P}(\mathcal{X}=3) = 3/10$,
    $\mathbb{P}(\mathcal{X}=4) = 1/10$, $\mathbb{P}(\mathcal{X}=5) = 0/10$, $\mathbb{P}(\mathcal{X}=6) = 1/10$.

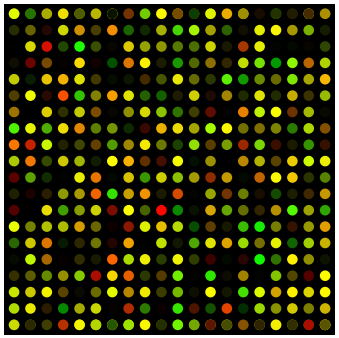


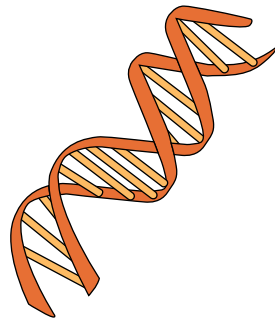
- In this thesis: density estimation for high-dimensional problems:
  - **high number of discrete variables** p (thousands or more),
  - **low number of samples** N (hundreds).

Density estimation for electrical networks

Dimensions	Tens of thousands to millions (depending on the level of detail). Example: in the order of 10 000 transmission nodes at extra-high voltage level, 100 000 wind turbines.
Prediction	Power flow in/out countries, based on local consumption? Production of solar/wind energy, based on the weather? Power in each line, based on production?

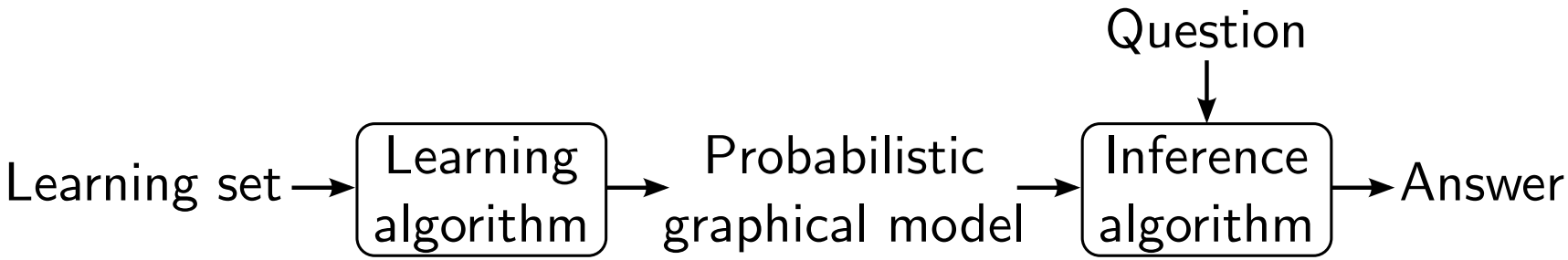
Density estimation in bioinformatics





Dimensions	thousands to tens of thousands of genes, hundreds of thousands of proteins.
Prediction	Effect of a combination of diseases and treatments on gene expression level? Most efficient medicine to tackle a particular disease?

Problem solving with probabilistic graphical models



Definitions

- **Learning** refers to the automatic construction of a model from a set of observations, **the learning set**. It may be done only once.
- A **probabilistic graphical model** encodes a probability density, e.g. a joint probability density over a set of variables: $\mathbb{P}(\boldsymbol{\mathcal{X}})$.
- Probabilistic **inference**, on a given model and for a particular question, consists in computing an answer to the query. The more general the model, the more questions can be answered.

What is so difficult about it?

Algorithmic complexity

- Algorithmic complexity refers to the asymptotic complexity of an algorithm as a function of the size of the input problem.
  Example: a $\mathcal{O}(p)$ algorithm has a run-time that increases linearly with p, as $p \to \infty$.
- For problems with a high number of variables p, the algorithms must have a complexity that is a very low-order polynomial in p.

Small number of samples $\to$ high variance

- Having few samples leads to variability in the models constructed.
- Illustration: dice throw example, 2 sequences of observations:
  - $D_1 = (2, 3, 1, 3, 6, 1, 4, 2, 6, 2) : \mathbb{P}_1(\text{"5"}) = 0/10$
  - $D_2 = (1, 5, 2, 6, 1, 1, 6, 1, 6, 5) : \mathbb{P}_2(\text{"5"}) = 2/10$
- Both models cannot be right: variance is a source of error.

What is so difficult about it?

Algorithmic complexity

- Algorithmic complexity refers to the asymptotic complexity of an algorithm as a function of the size of the input problem.
  Example: a $\mathcal{O}(p)$ algorithm has a run-time that increases linearly with p, as $p \to \infty$.
- For problems with a high number of variables p, the algorithms must have a complexity that is a very low-order polynomial in p.

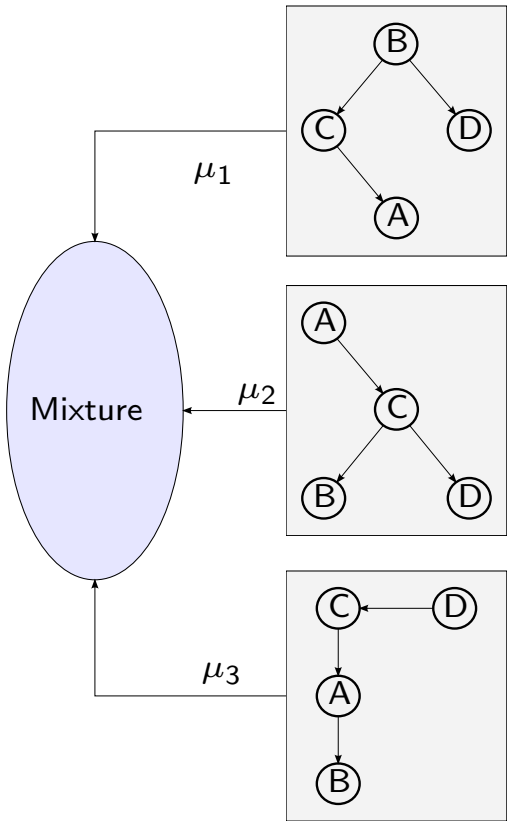
Small number of samples $\to$ high variance

- Having few samples leads to variability in the models constructed.
- Illustration: dice throw example, 2 sequences of observations:
  - $D_1 = (2, 3, 1, 3, 6, 1, 4, 2, 6, 2) : \mathbb{P}_1(\text{"5"}) = 0/10$
  - $D_2 = (1, 5, 2, 6, 1, 1, 6, 1, 6, 5) : \mathbb{P}_2(\text{"5"}) = 2/10$
- Both models cannot be right: variance is a source of error.

Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

Goal: Density estimation for high p (number of variables) low N (number of samples).

- algorithmic complexity
  - $\rightarrow$ simple models ($\equiv$Markov trees)
- high variance
  - $\rightarrow$ simple models
  - $\rightarrow$ mixture of simple models



$$\mathbb{P}_{\hat{\mathcal{T}}}(\boldsymbol{\mathcal{X}}) = \sum_{i=1}^{m} \mu_i \mathbb{P}_{\mathcal{T}_i}(\boldsymbol{\mathcal{X}})$$

Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

1. Background
2. Contributions (x3)
3. Final words

Background: What is it you are doing again?

- ? Probabilistic graphical models
- ? Tree-structured probabilistic graphical models
- ? Mixtures

Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

1. Background
2. Contributions (x3)
3. Final words

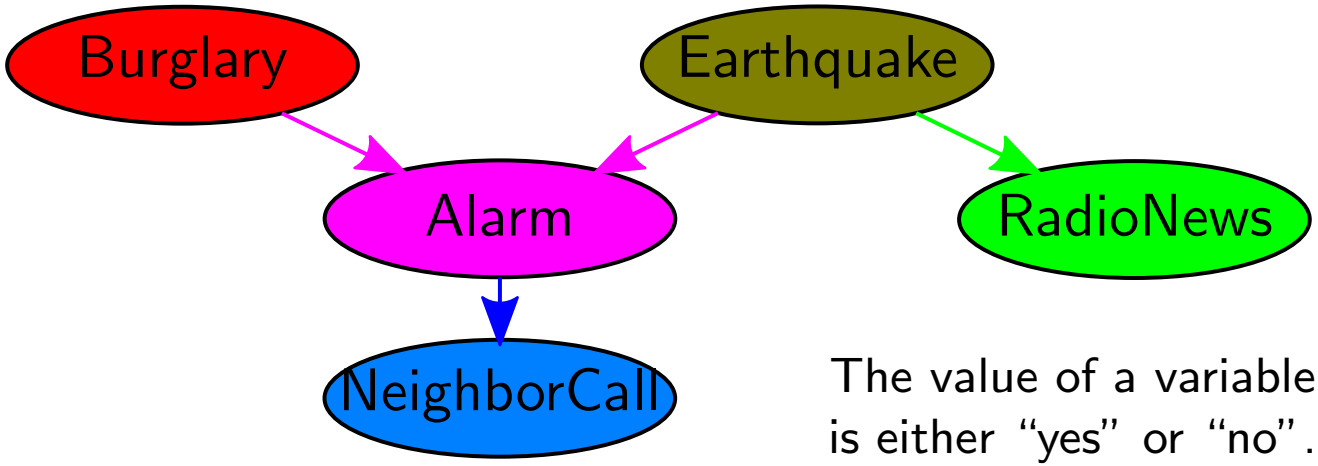
Background: What is it you are doing again?

- ? Probabilistic graphical models
- ? Tree-structured probabilistic graphical models
- ? Mixtures

A Bayesian network is a PGM encoding a joint probability density over a set of variables $\mathcal{X}$.

A Bayesian network is composed of 2 elements.

- The directed acyclic graph structure $\mathcal{G}$ encodes (conditional independence) relationships among variables.
- The set of parameters θ quantifies the probability density.

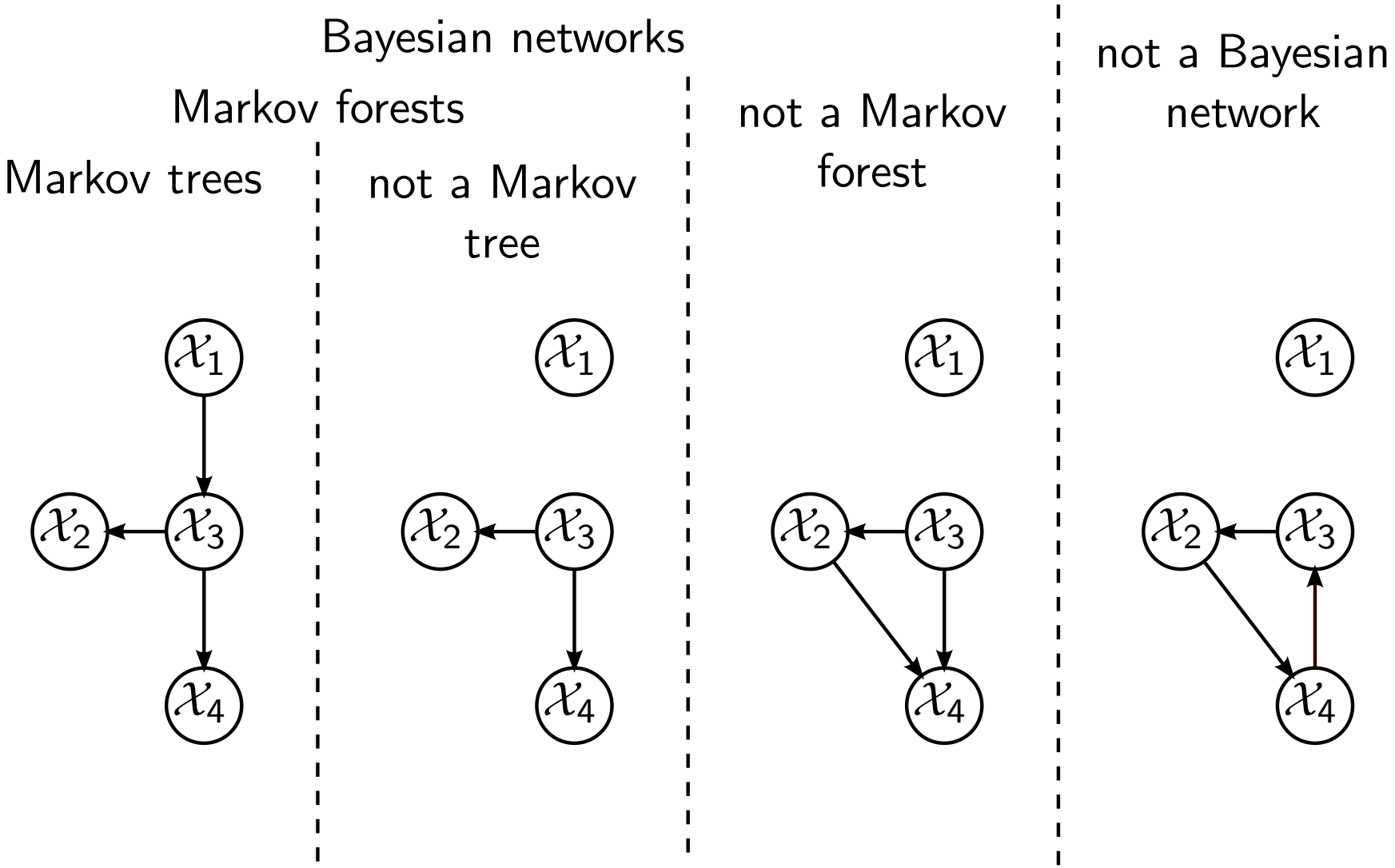


The value of a variable is either "yes" or "no".

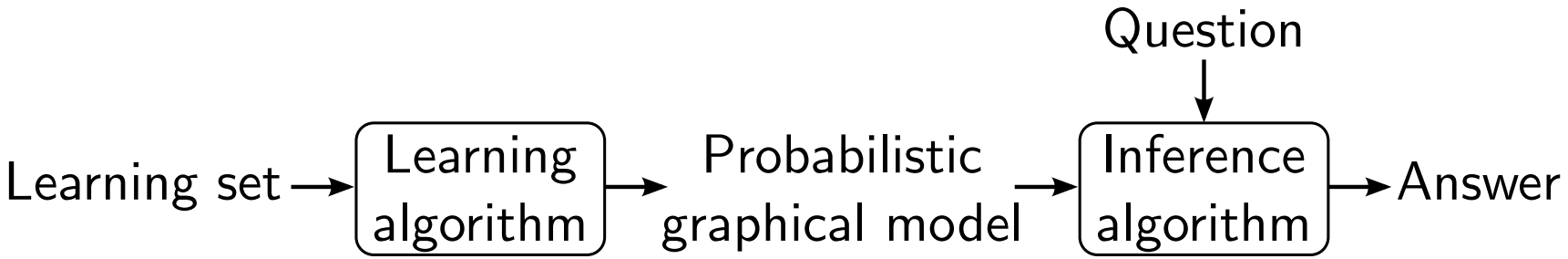
The Bayesian network reduces the number of parameters stored:

$$\begin{array}{lccccc}
\mathbb{P}(\mathcal{B},\mathcal{E},\mathcal{A},\mathcal{R},\mathcal{N}) = & \mathbb{P}(\mathcal{B}) & \mathbb{P}(\mathcal{E}|\mathcal{B}) & \mathbb{P}(\mathcal{A}|\mathcal{B},\mathcal{E}) & \mathbb{P}(\mathcal{R}|\mathcal{B},\mathcal{E},\mathcal{A}) & \mathbb{P}(\mathcal{N}|\mathcal{B},\mathcal{E},\mathcal{A},\mathcal{R}) \\
= & \mathbb{P}(\mathcal{B}) & \mathbb{P}(\mathcal{E}) & \mathbb{P}(\mathcal{A}|\mathcal{B},\mathcal{E}) & \mathbb{P}(\mathcal{R}|\mathcal{E}) & \mathbb{P}(\mathcal{N}|\mathcal{A}) \\
\text{parameters} : 2^5-1=31 \leftrightarrow & 1 & +1 & +4 & +2 & +2=10
\end{array}$$

A Markov tree is Bayesian network where the graphical structure is constrained to be a (connected) tree.



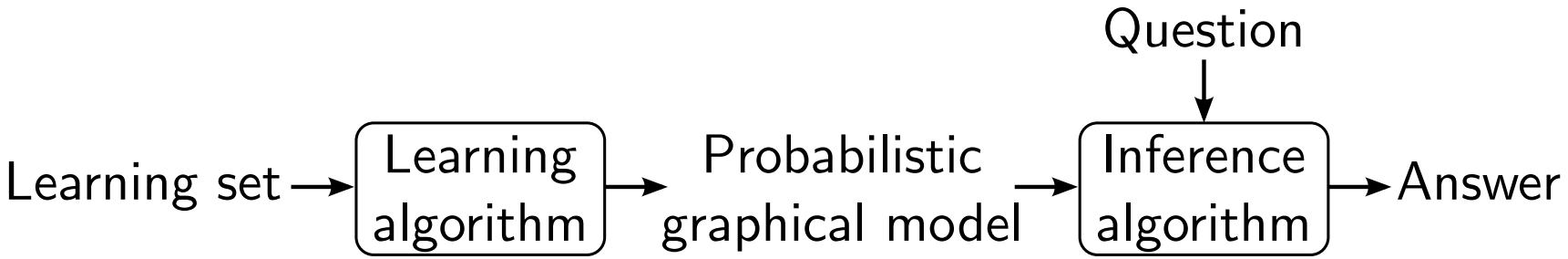
The class of models considered is a tradeoff between capacity of representation and computational complexity.



	Bayesian networks	Markov trees
Complexity	**superexponential** in p for both learning and inference	**tractable** for both learning ($\mathcal{O}(p^2 \log p)$) and inference ($\mathcal{O}(p)$)
Accuracy	**any probability density** can be encoded	only a **restricted set of probability densities** can be encoded

Capacity of representation might be detrimental to accuracy!

The class of models considered is a tradeoff between capacity of representation and computational complexity.

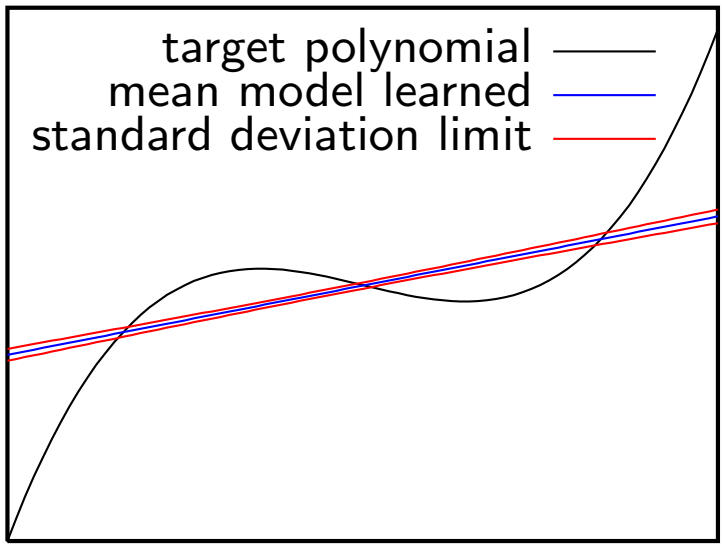


	Bayesian networks	Markov trees
Complexity	**superexponential** in p for both learning and inference	**tractable** for both learning ($\mathcal{O}(p^2 \log p)$) and inference ($\mathcal{O}(p)$)
Accuracy	**any probability density** can be encoded	only a **restricted set of probability densities** can be encoded

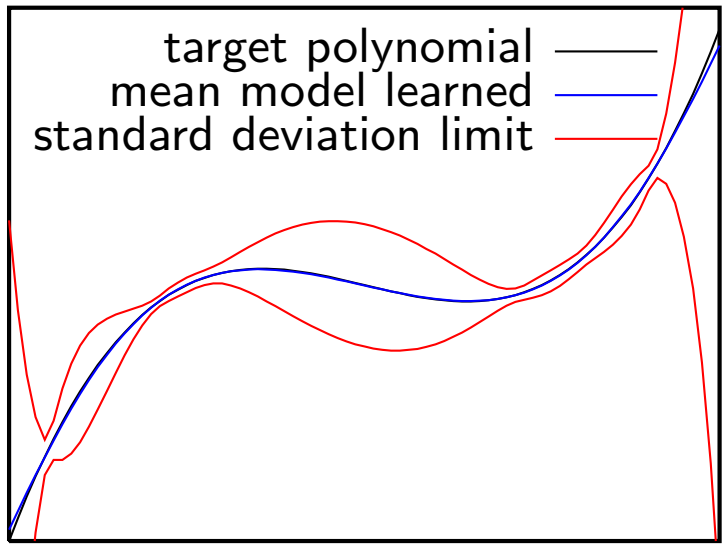
Capacity of representation might be detrimental to accuracy!

Average error of a model learnt = bias + variance.

Overly simple class of models



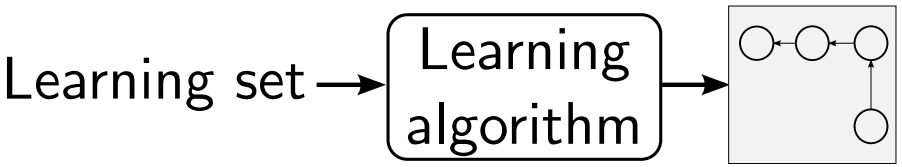
Overly complex class of models



- **Bias** is the difference between the mean model learned (over all possible sets of observations of a given size) and the best model.
- **Variance** is the average variability of a model learnt, with respect to the mean model learned.
- When the complexity of the class of models increases:
  - bias tends to decrease,
  - variance to increase.

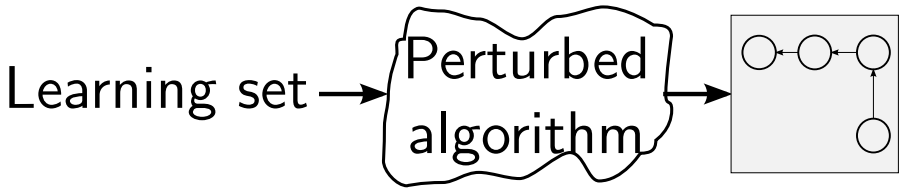
Constructing a mixture can reduce the variance.

Normal algorithm:

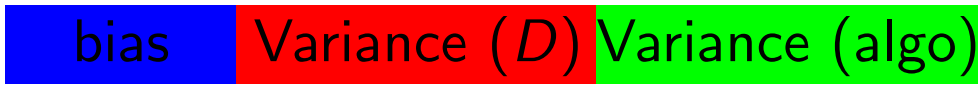


Error:

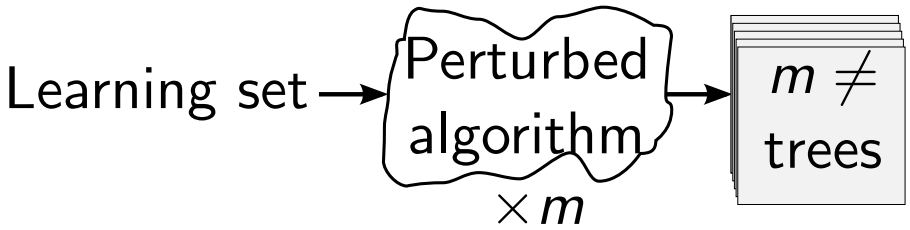
Perturb: randomize the learning algorithm



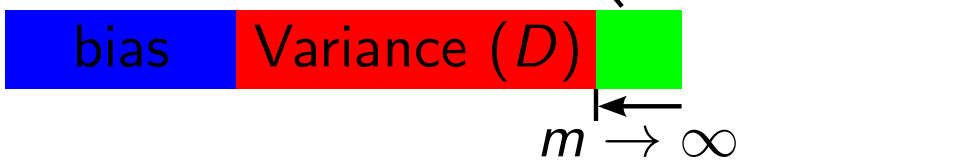
Error:



Perturb&Combine: generate several models and combine them

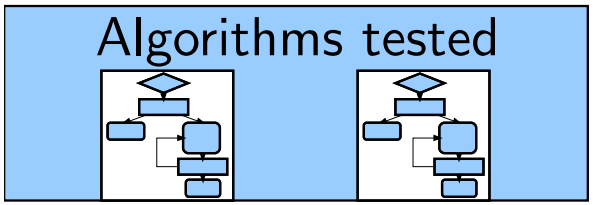


Error:



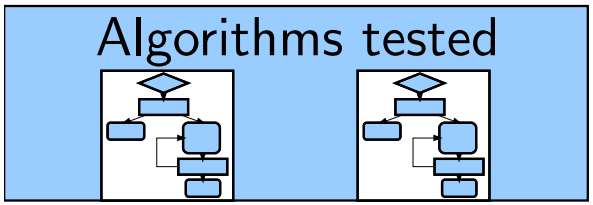
$$\mathbb{P}(\boldsymbol{\mathcal{X}}) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{P}_i(\boldsymbol{\mathcal{X}})$$

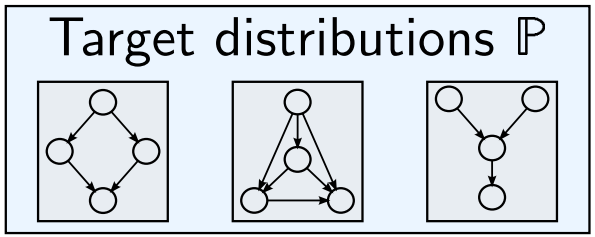
The validation of the algorithms is empirical.



Score of the algorithms:
- accuracy
- efficiency

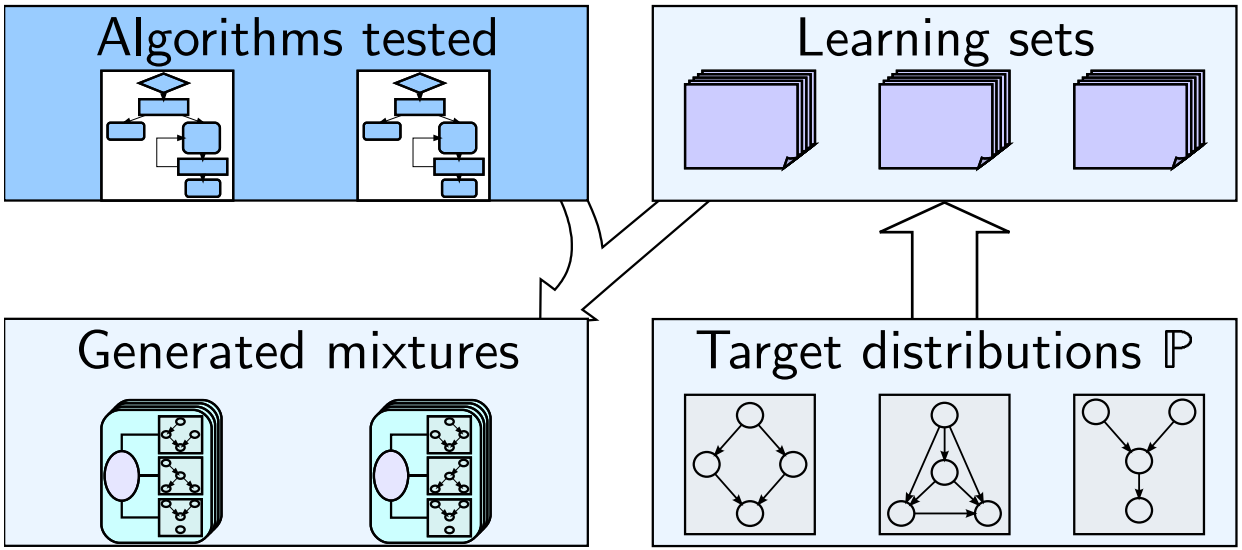
The validation of the algorithms is empirical.





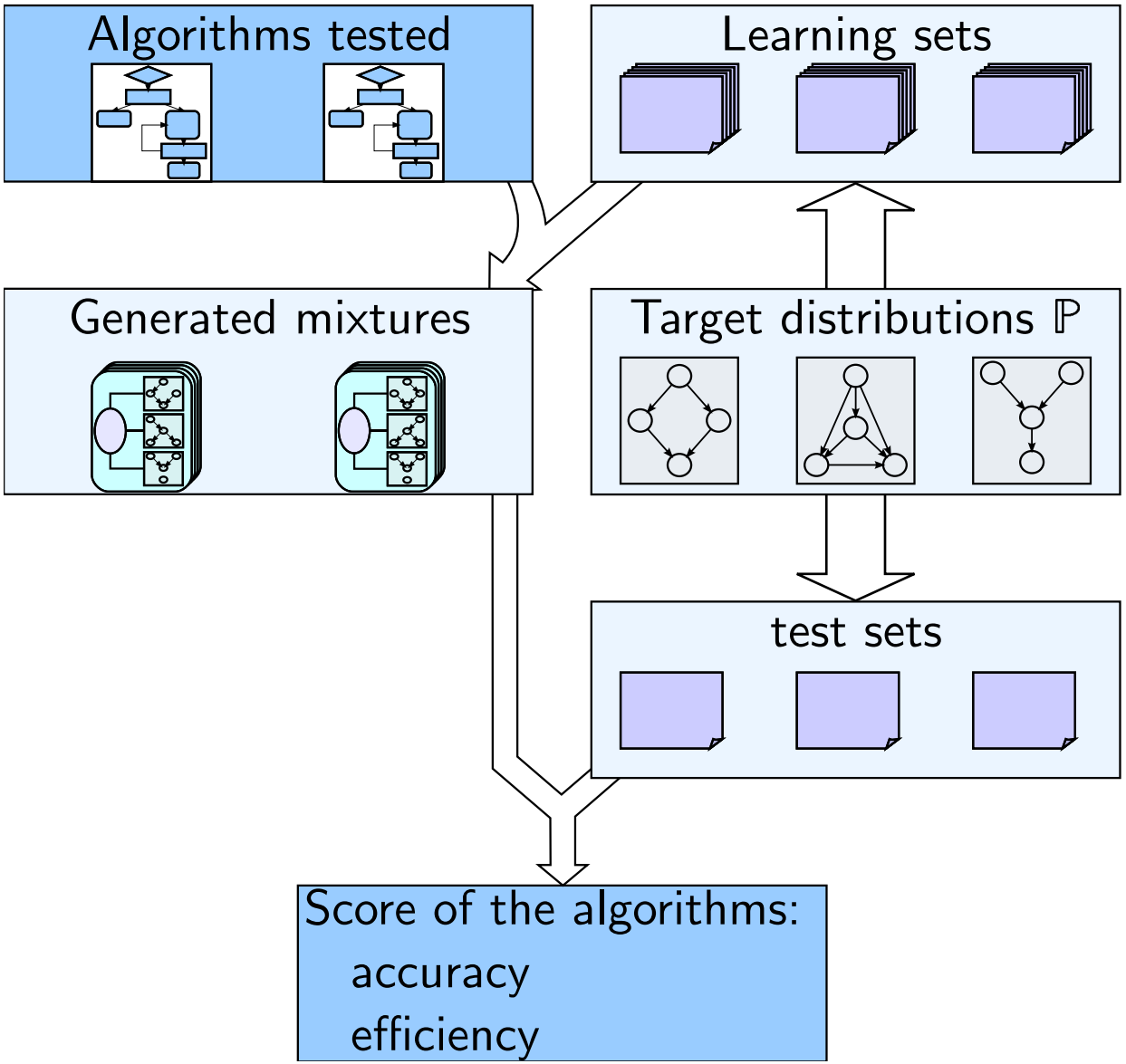
Score of the algorithms:
- accuracy
- efficiency

The validation of the algorithms is empirical.

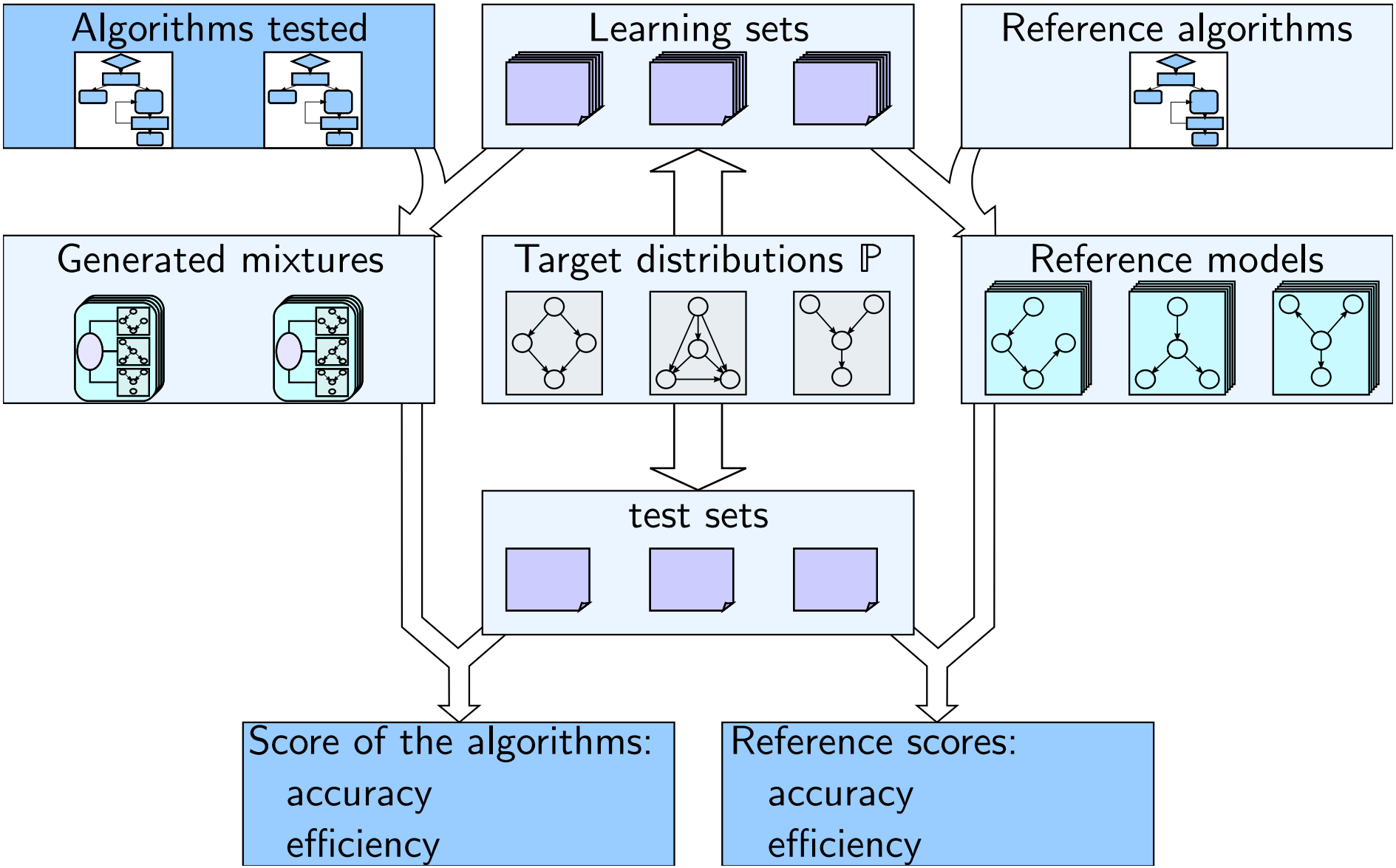


Score of the algorithms:
- accuracy
- efficiency

The validation of the algorithms is empirical.



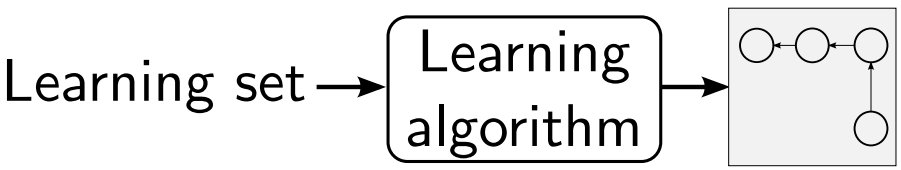
The validation of the algorithms is empirical.



Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

1. Background
2. Contributions
   - Repeatedly learning a perturbed Markov tree
   - Building a sequence of Markov trees
   - Combining mixtures
3. Final words

Learning a Markov tree is a 2-step process.



Error:

1. Learn the best structure $\mathcal{T}$, given the observations.
   - Define a score on the structures, and find the structure maximizing it.
2. Learn the parameters θ for the selected structure $\mathcal{G}$.
   - It amounts to counting observations.

NB:
Parameter learning in this thesis:

$$\mathbb{P}(\mathcal{X}_i = x | \boldsymbol{\mathcal{P}a}_{\mathcal{X}_i} = \mathbf{a}) = \frac{1 + N_D(\mathbf{a}, x)}{|Val(\mathcal{X}_i)| + \sum_{x \in Val(\mathcal{X}_i)} N_D(\mathbf{a}, x)} \quad .$$

$N_D(\mathbf{a}, x)$ is the number of samples where $\mathcal{X}_i = x$ and $\boldsymbol{\mathcal{P}a}_{\mathcal{X}_i} = \mathbf{a}$.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$

p variables

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
0	1	0	1
1	1	0	1
0	0	1	1
1	1	1	0

N samples

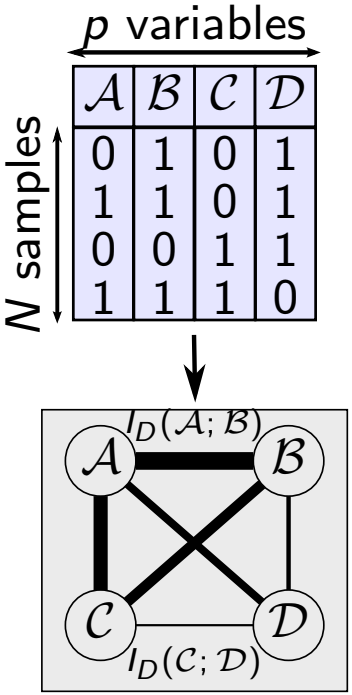
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

$\rightarrow$ Complexity is $\mathcal{O}(p^2 \log p)$.

$\rightarrow$ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$



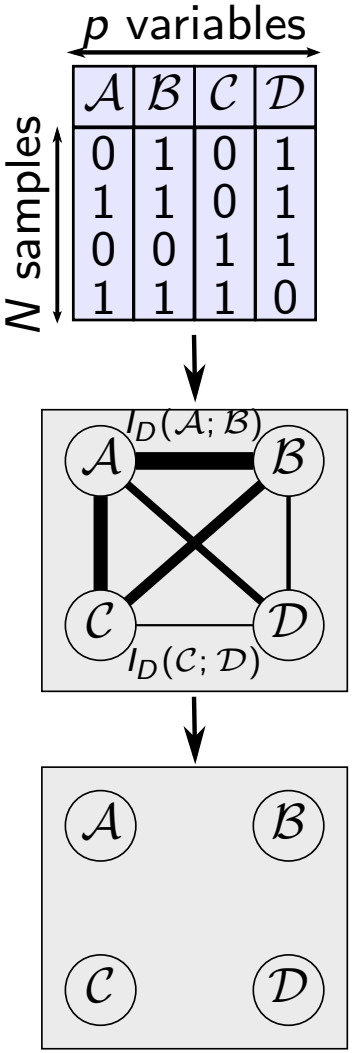
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

$\rightarrow$ Complexity is $\mathcal{O}(p^2 \log p)$.

$\rightarrow$ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$



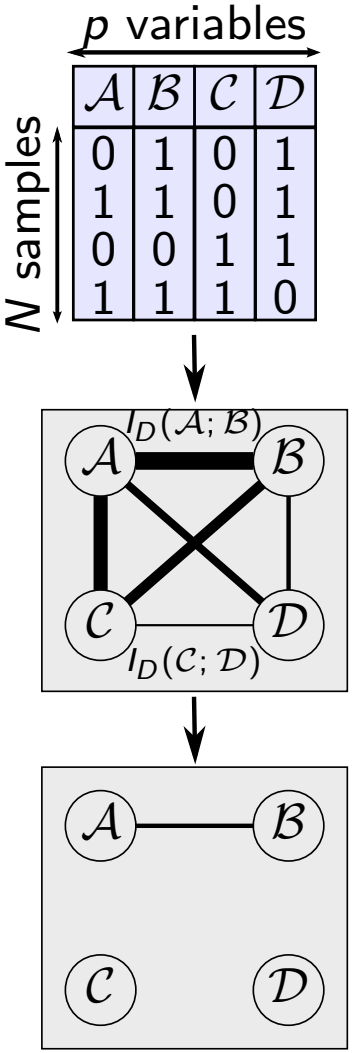
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

→ Complexity is $\mathcal{O}(p^2 \log p)$.

→ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$



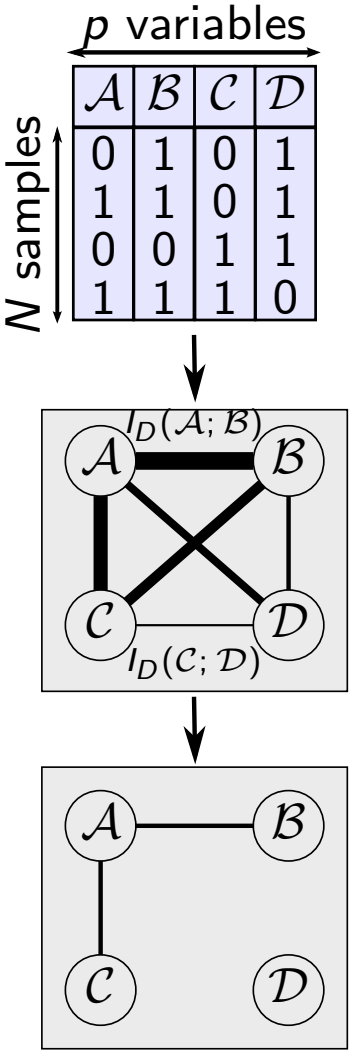
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

→ Complexity is $\mathcal{O}(p^2 \log p)$.

→ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$



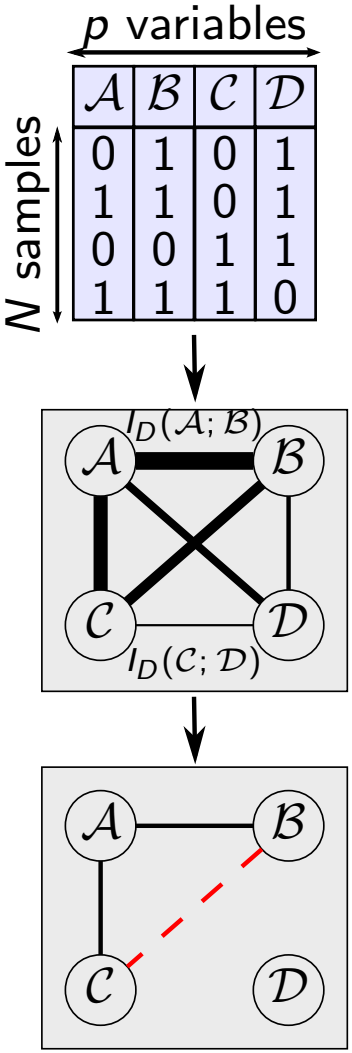
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

→ Complexity is $\mathcal{O}(p^2 \log p)$.

→ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$



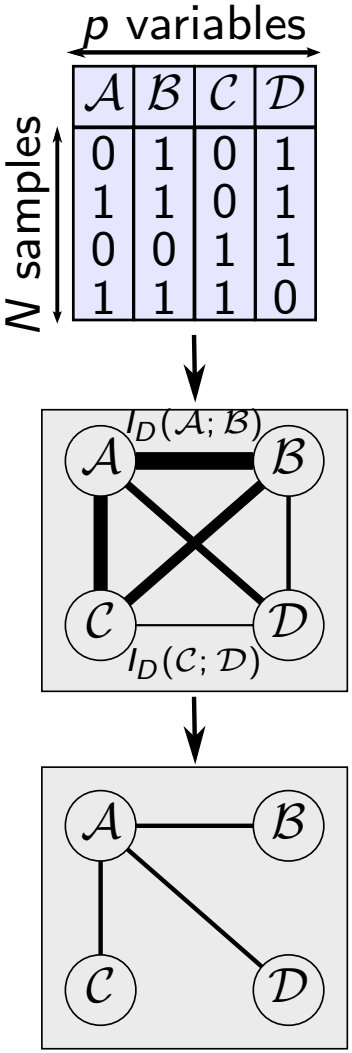
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

→ Complexity is $\mathcal{O}(p^2 \log p)$.

→ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$



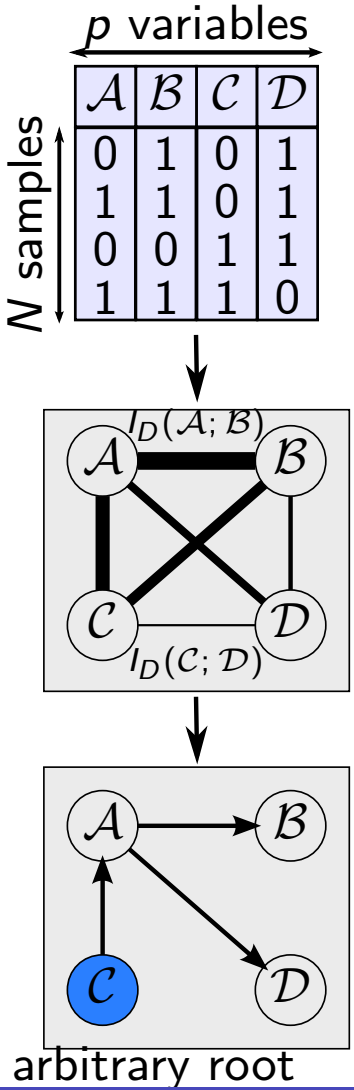
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

→ Complexity is $\mathcal{O}(p^2 \log p)$.

→ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$



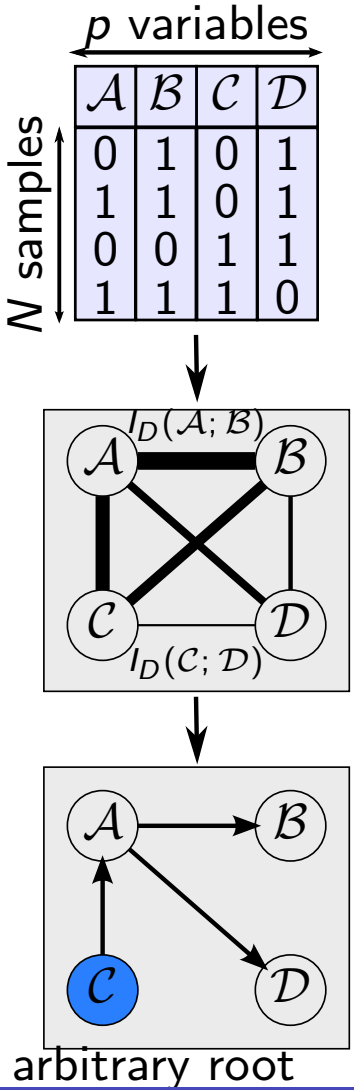
- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

$\rightarrow$ Complexity is $\mathcal{O}(p^2 \log p)$.

$\rightarrow$ Bottleneck: number of candidate edges for each tree.

The Chow-Liu algorithm learns a Markov tree structure able to maximize the likelihood of the learning set.

$$\mathcal{T}_{CL}(D) = \arg\max_{\mathcal{T}} \sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{T})} I_D(\mathcal{X};\mathcal{Y})$$

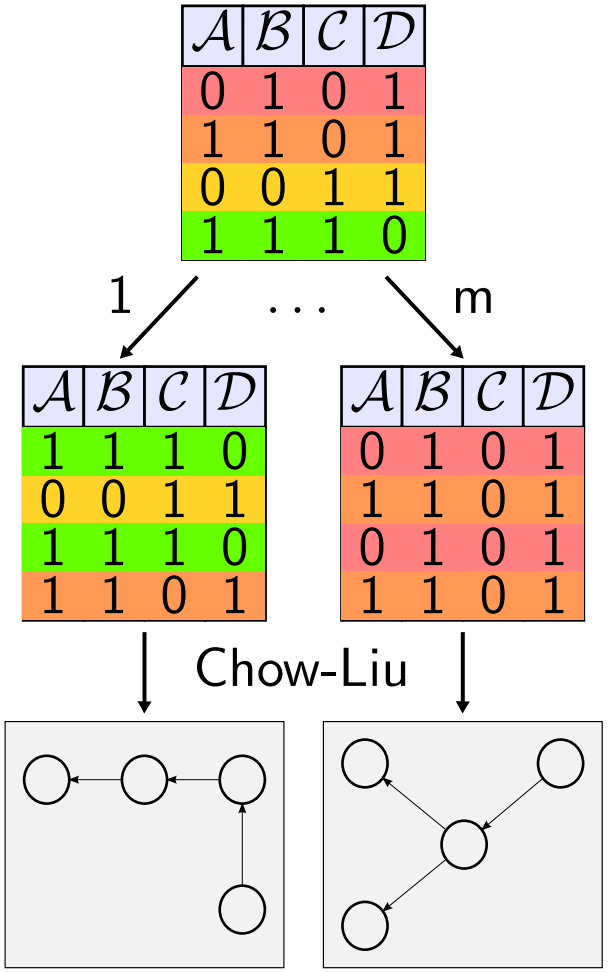


- Best modeling of the learning set
- Structure learning is a 3-step process:
  1. Construction of a complete graph whose edge-weights are empirical mutual informations ($\mathcal{O}(p^2 N)$)
  2. Computation of the maximum weight spanning tree ($\mathcal{O}(p^2 \log p)$)
  3. Orientation ($\mathcal{O}(p)$)

$\rightarrow$ Complexity is $\mathcal{O}(p^2 \log p)$.

$\rightarrow$ Bottleneck: number of candidate edges for each tree.

Example of perturbation: bagging



- Average over m max-likelihood trees learnt from m bootstrap replicates
- A bootstrap replicate $\mathbf{D}'$ of a sample set $\mathbf{D}$ is the same size as $\mathbf{D}$ and is drawn with replacement from $\mathbf{D}$.

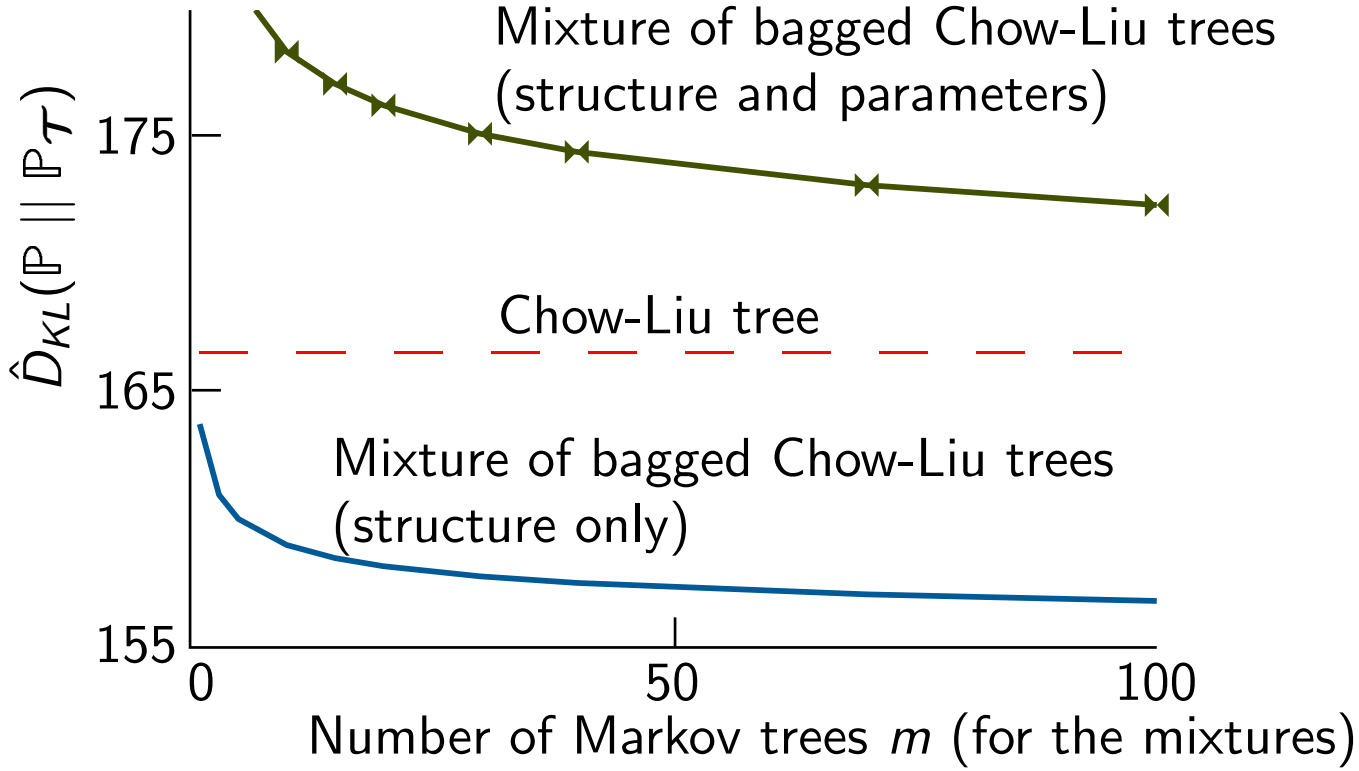
$\rightarrow$ Complexity is $\mathcal{O}(mp^2 \log p)$.

- In my case, θ are learned on $\mathbf{D}$, not $\mathbf{D}'$.

Probability Density Estimation by Perturbing and Combining Tree Structured Markov Networks, S. Ammar, P. Leray, B. Defourny and L. Wehenkel ECSQARU 2009.

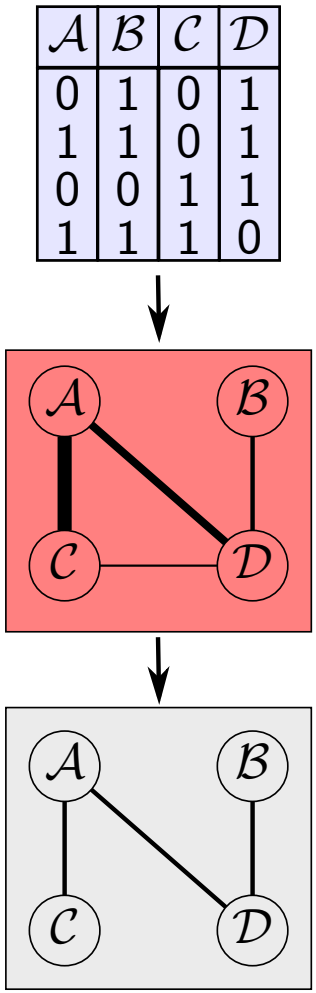
Learning the parameters on the original learning set leads to a better accuracy.

- The more trees in the mixture, the better the accuracy.



Averaged results on 10 randomly generated Bayesian networks (1000 binary variables) $\times$ 10 learning sets, for $N = 300$ samples.

In order to reduce algorithmic complexity, only a subset of K edges must be considered.



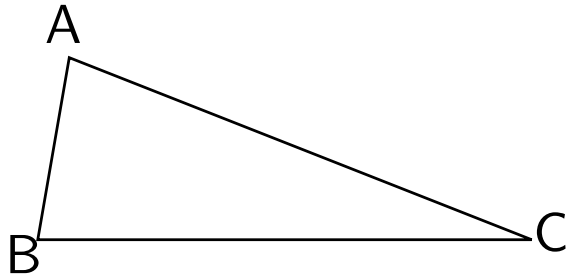
A simple way to reduce complexity is therefore to randomly select a subset of K edges.

- Reduction in complexity (for each term):
  - Construction of an uncomplete graph: $\mathcal{O}(KN)$
  - Computation of the maximum width spanning tree ($\mathcal{O}(K \log K)$)
- The higher K, the closer to the Chow-Liu algorithm.

Intuitively, the structure of the problem can be exploited to target more interesting edges.

In a Euclidean space, similar problems can be approximated by sub-quadratic algorithms. When 2 points B and C are close to A, they are likely to be close as well.

$$d(B, C) \leqslant d(A, B) + d(A, C)$$



Mutual information is *not* a Euclidean distance. However the same reasoning can be applied. If the pairs $\mathcal{A};\mathcal{B}$ and $\mathcal{A};\mathcal{C}$ have high mutual information values, $I(\mathcal{B};\mathcal{C})$ may be high as well.

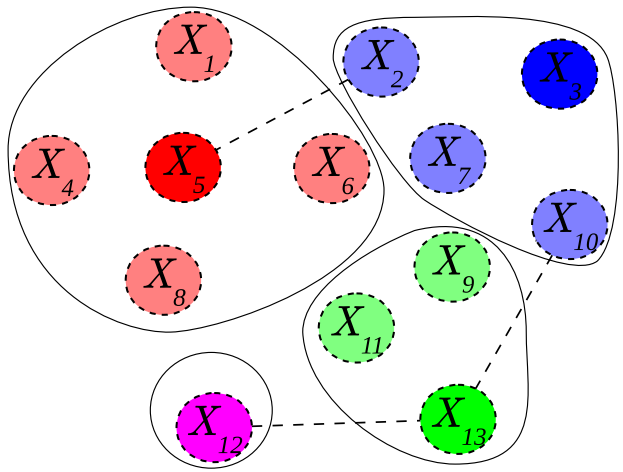
$$I(\mathcal{B};\mathcal{C}) \geqslant I(\mathcal{A};\mathcal{B}) + I(\mathcal{A};\mathcal{C}) - H(\mathcal{A})$$

I want to group related variables together and to use this structure to target interesting edges.

The algorithm

- builds a clustering of the variables and relationships between clusters,
- exploits this structure to target interesting edges, i.e. edges with strong associated mutual information,
- uses these edges to build a Markov tree.

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
0	1	0	1
1	1	0	1
0	0	1	1
1	1	1	0



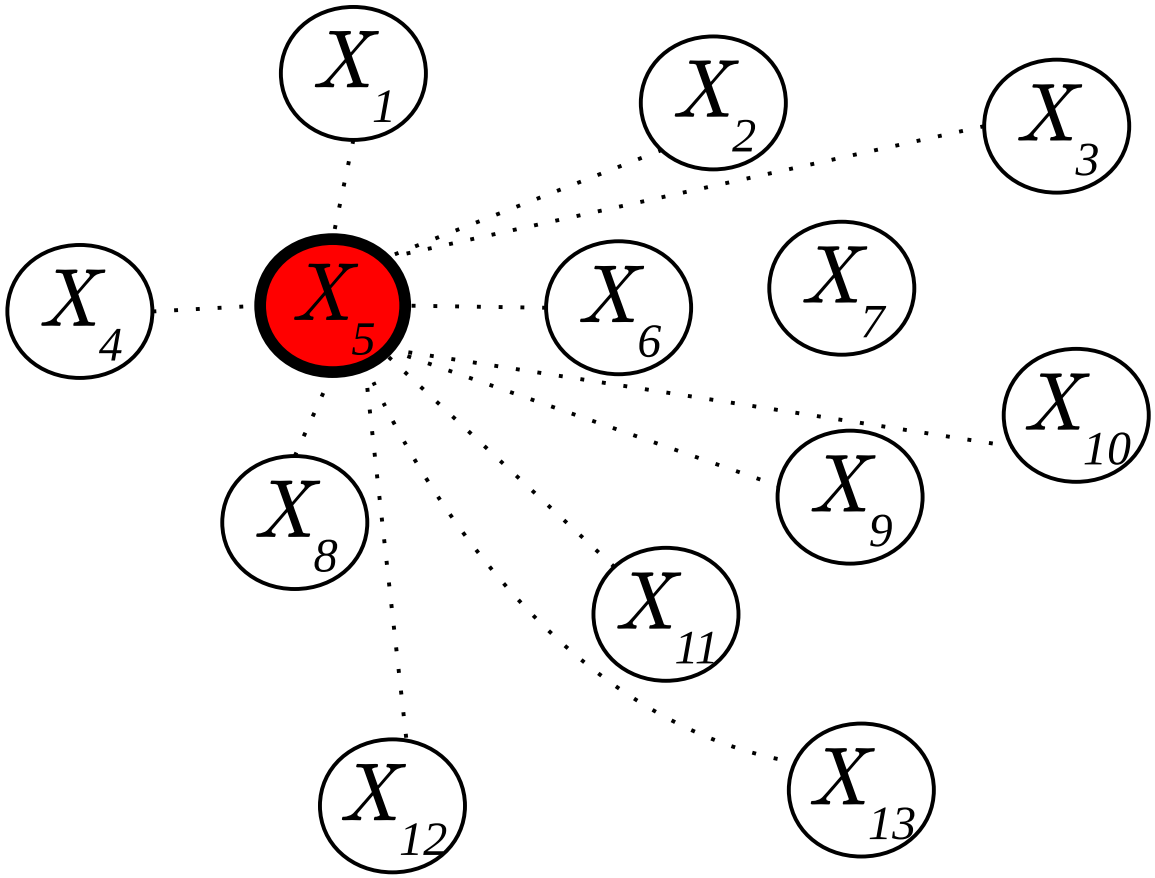
interesting edges and associated mutual information values → MWST → Markov tree structure

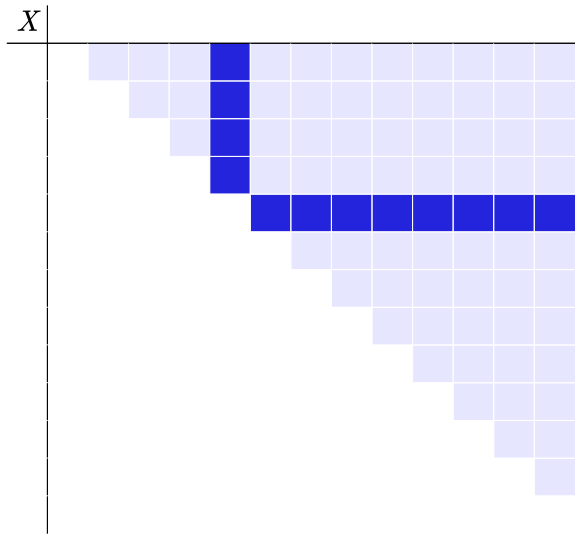
Classical approach to build clusters: compute distance for each object pair

→ not suitable

The clusters are built iteratively, one at a time:

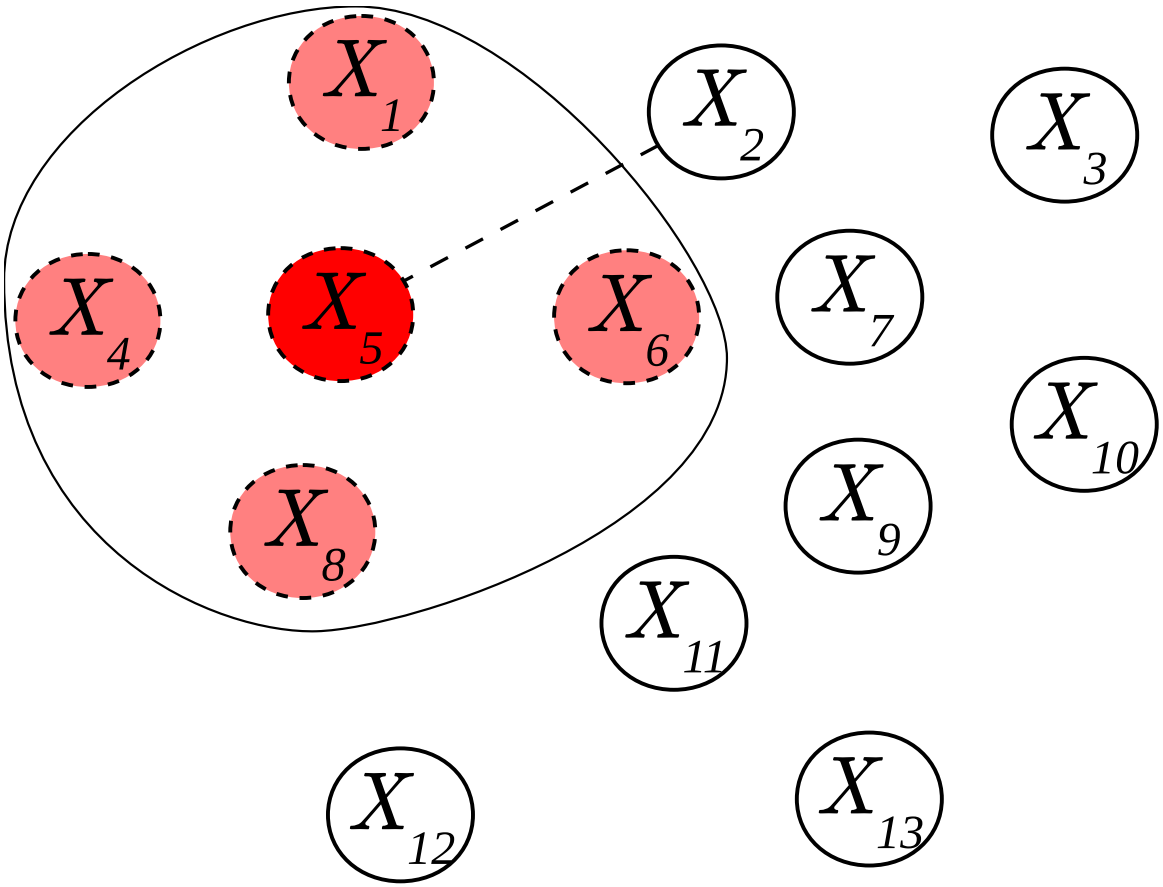
A center (X_5) is randomly chosen and compared to the 12 other variables.

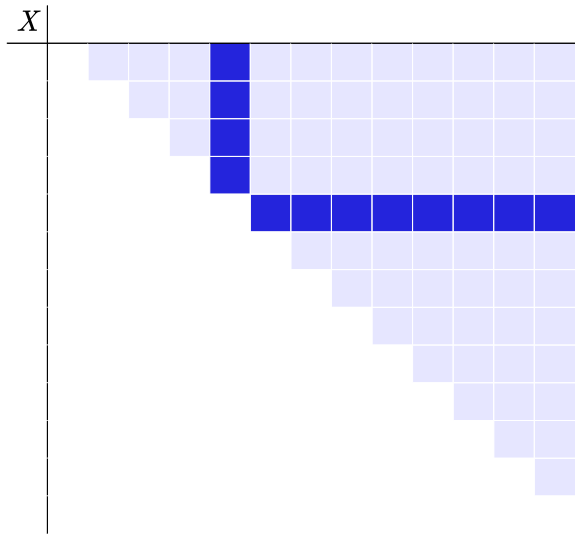




The clusters are built iteratively, one at a time:

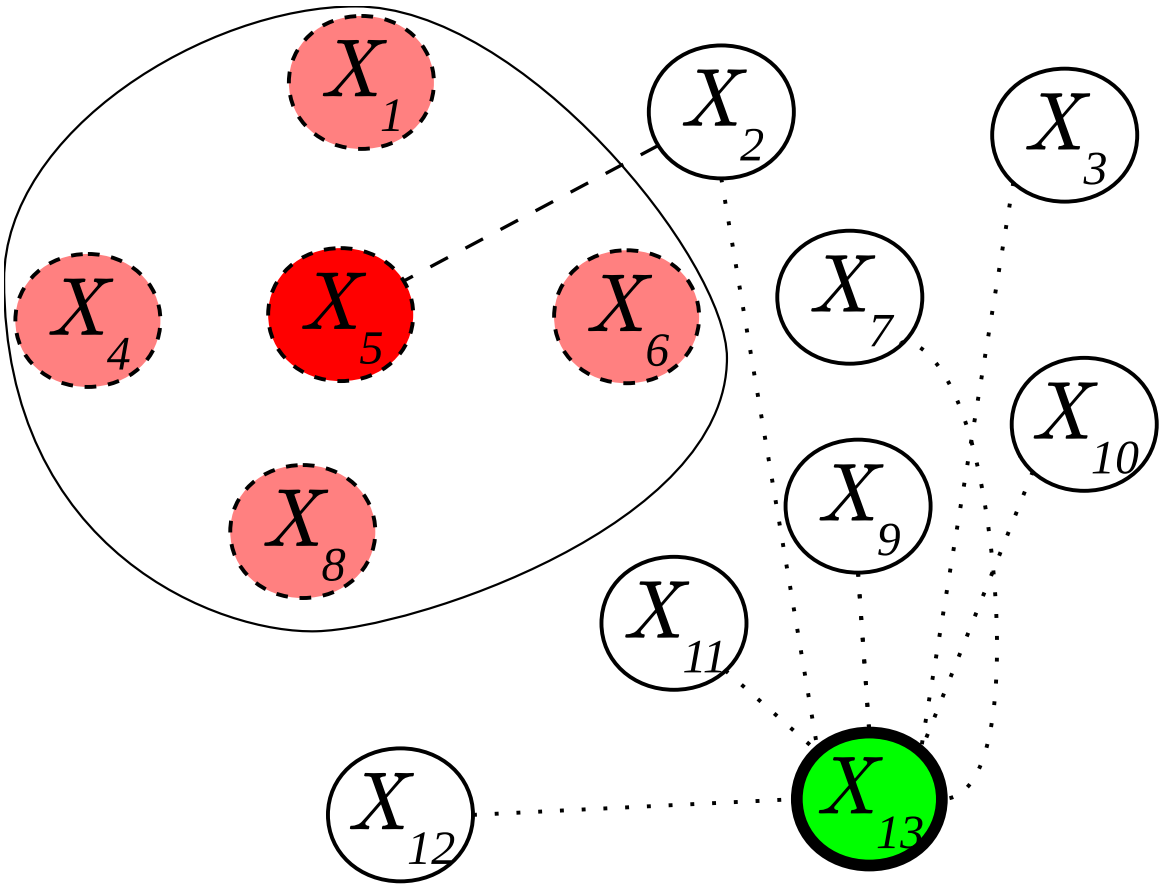
First cluster is created: it is composed of 5 members and 1 neighbour.
Variables are assigned to a cluster based on two thresholds and their empirical mutual information with the center of the cluster.

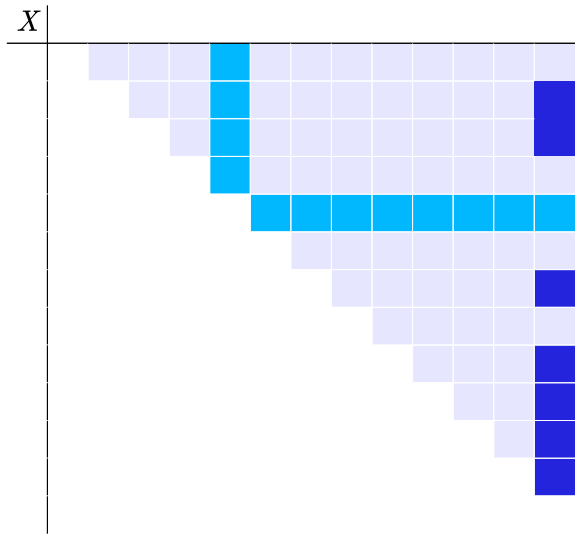




The clusters are built iteratively, one at a time:

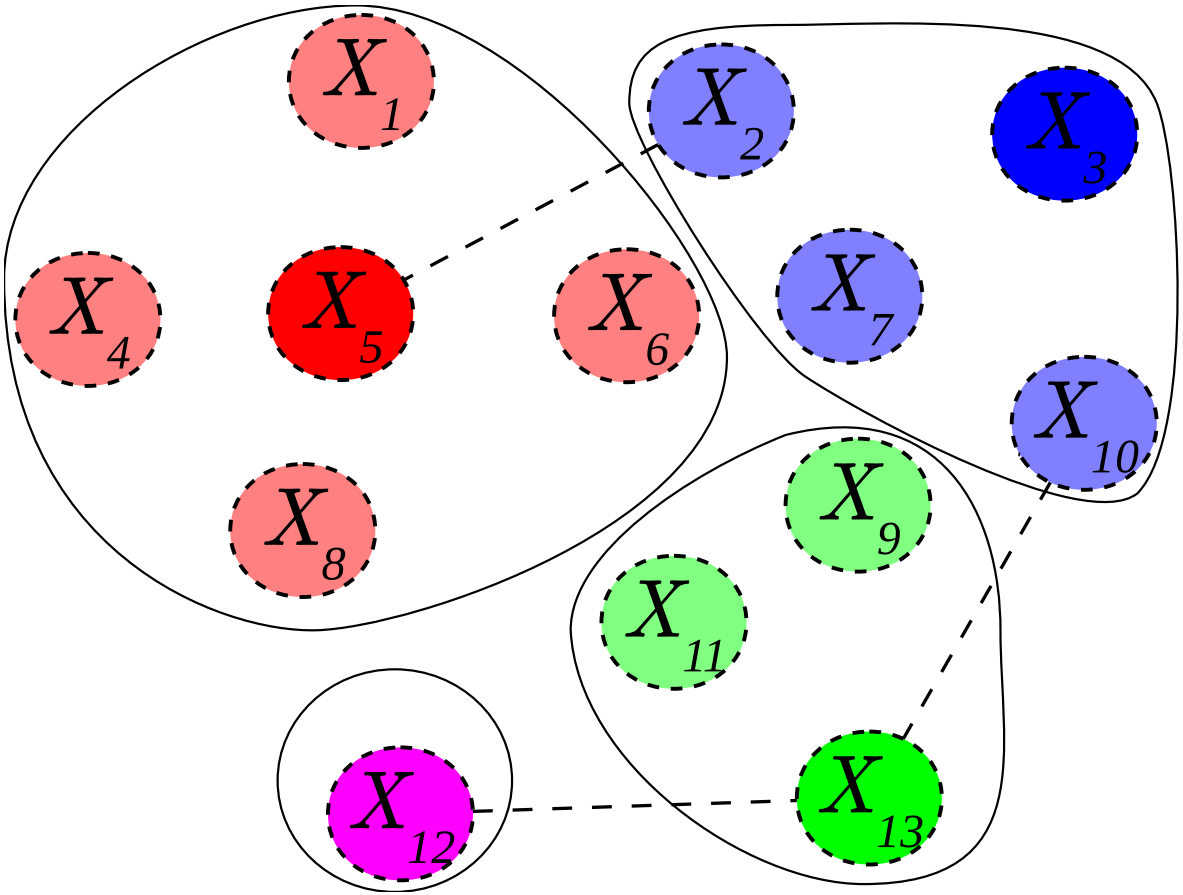
The second cluster is built around X_{13}. It is only compared to the 7 remaining variables.

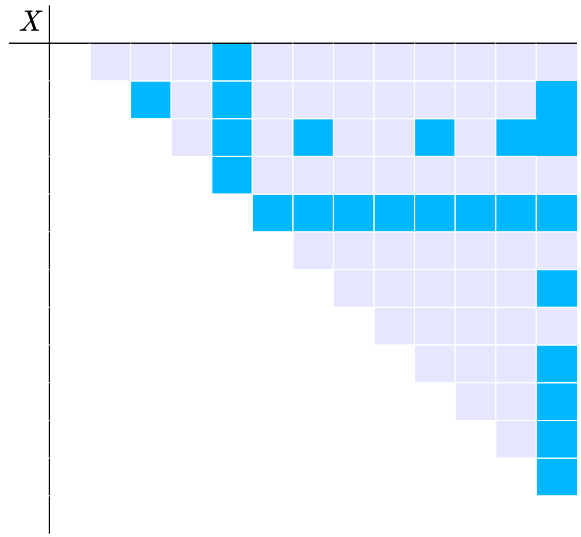




The clusters are built iteratively, one at a time:

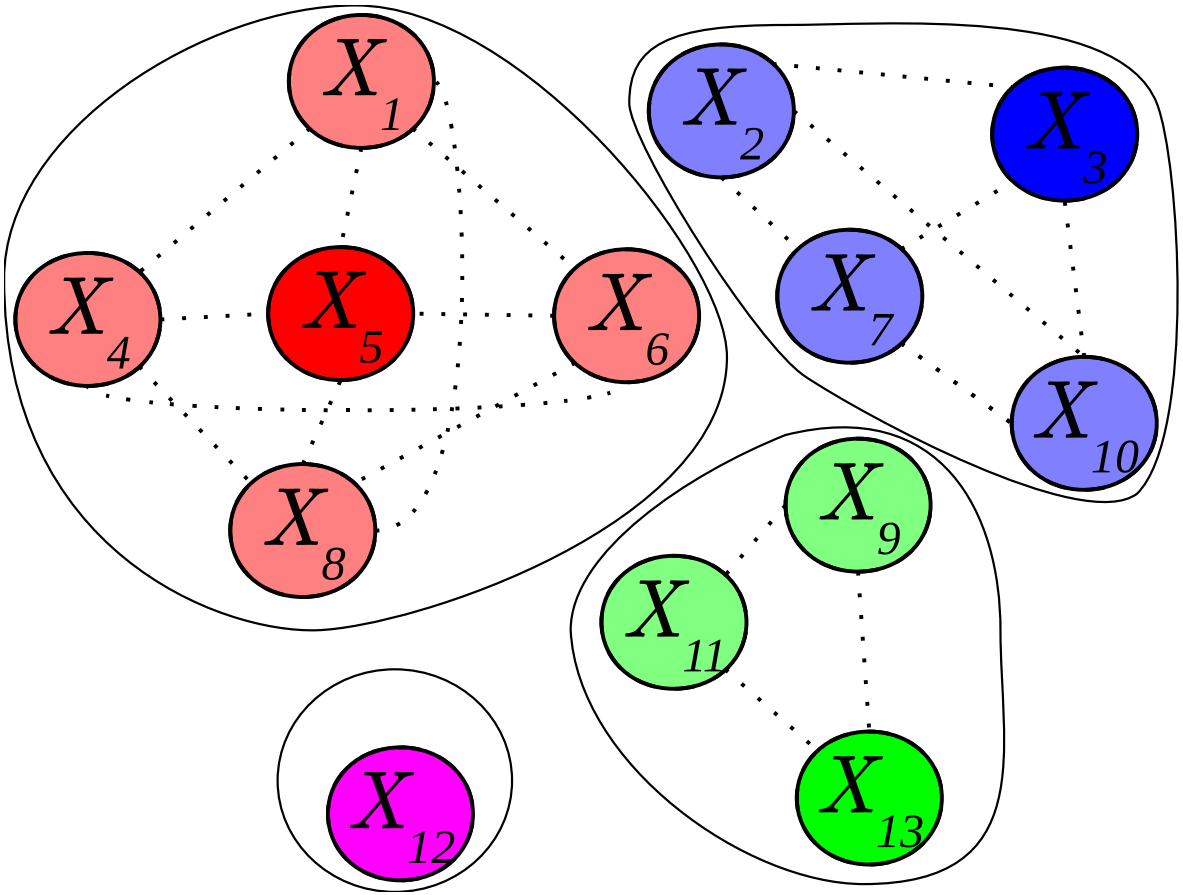
After 4 iterations, all variables belong to a cluster. The algorithm stops.

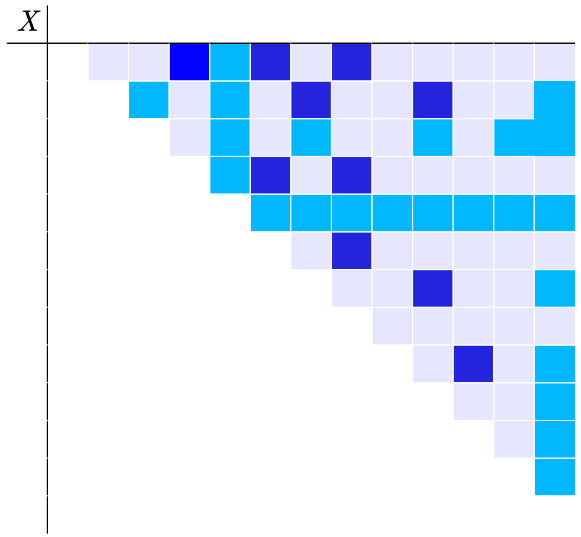




The clusters are exploited to target interesting edges:

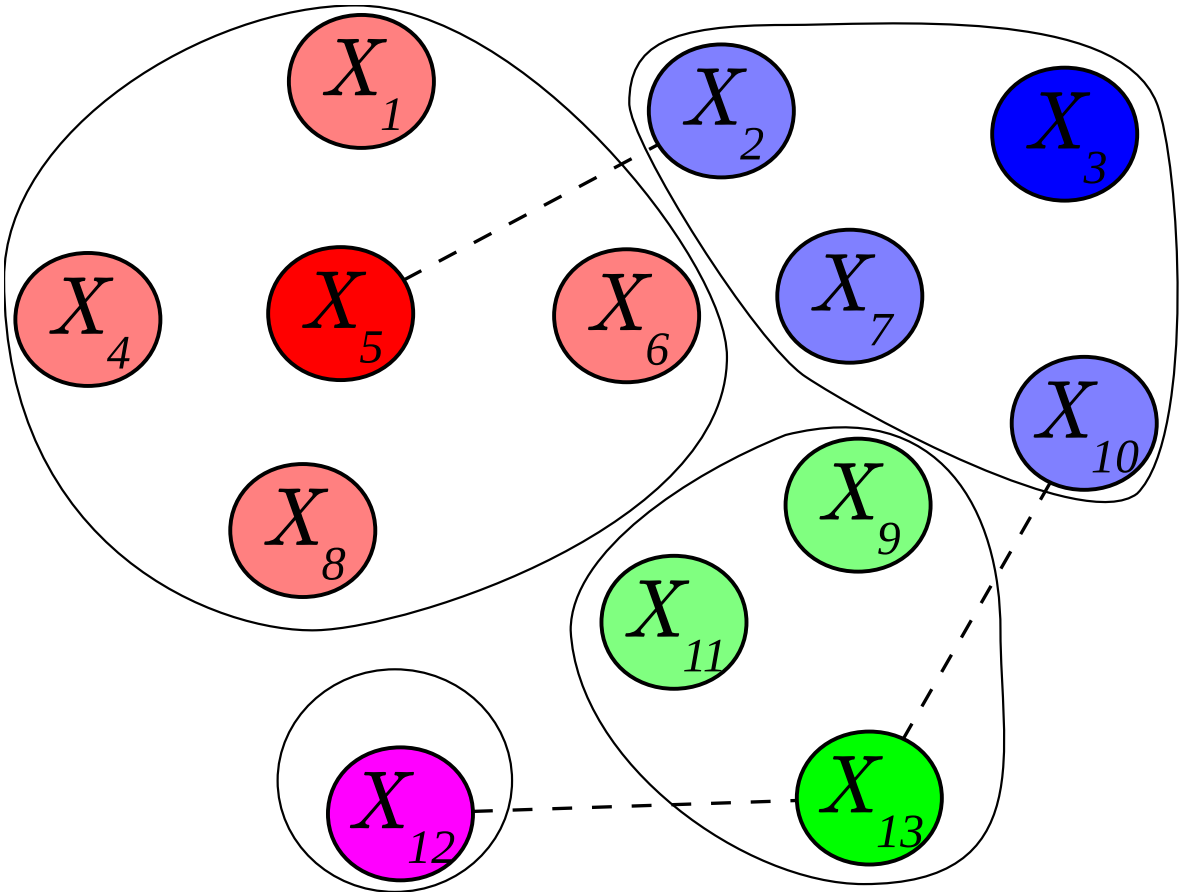
Computation of mutual information among variables belonging to the same cluster.

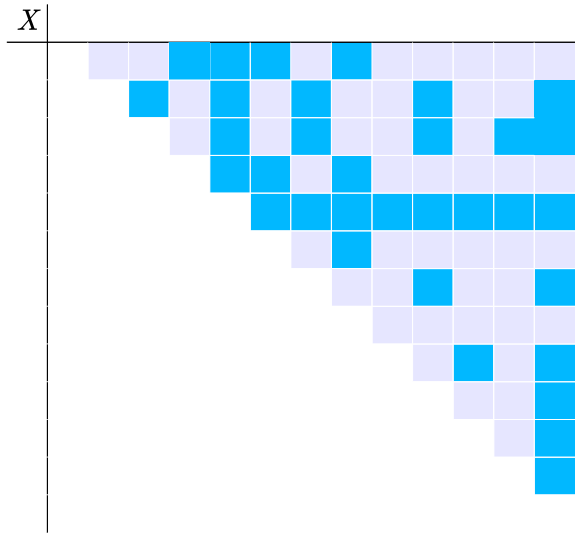




The clusters are exploited to target interesting edges:

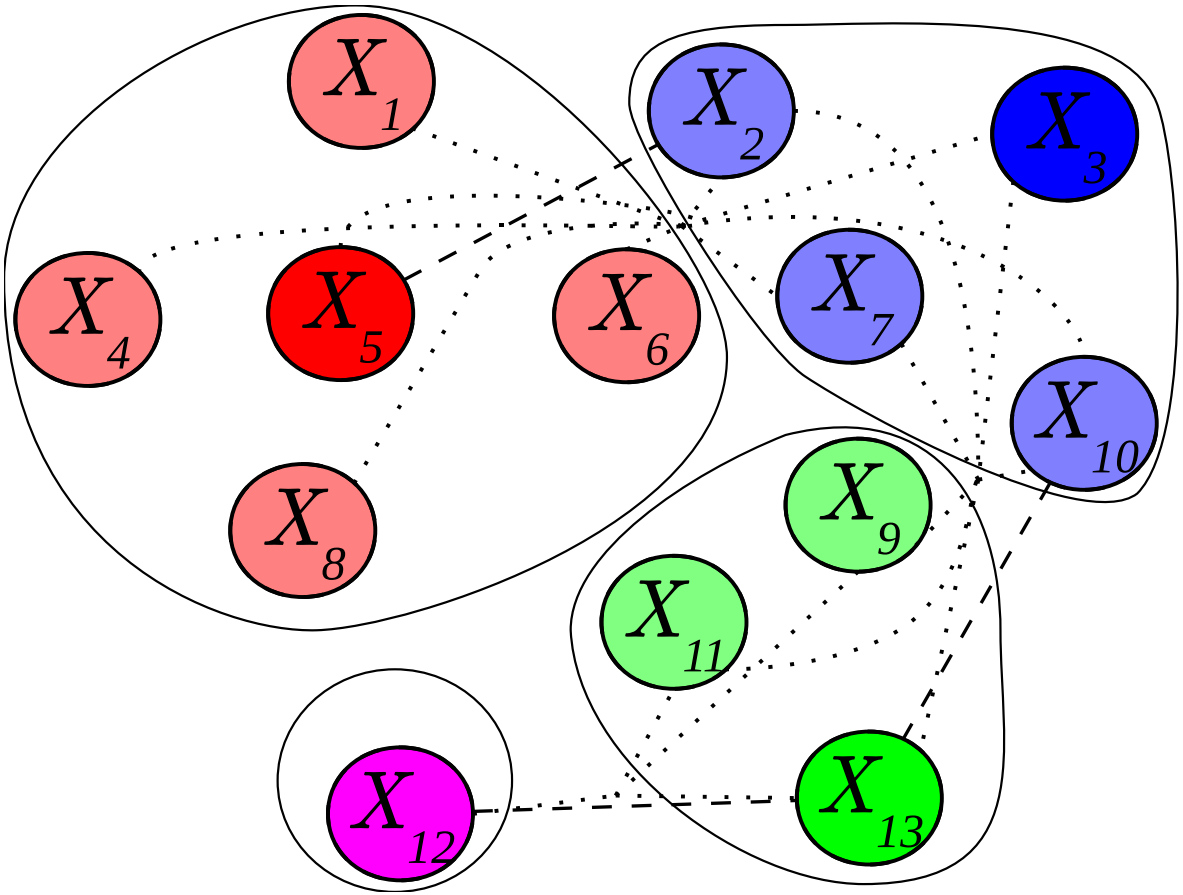
Two clusters containing neighbor variables are neighbors.

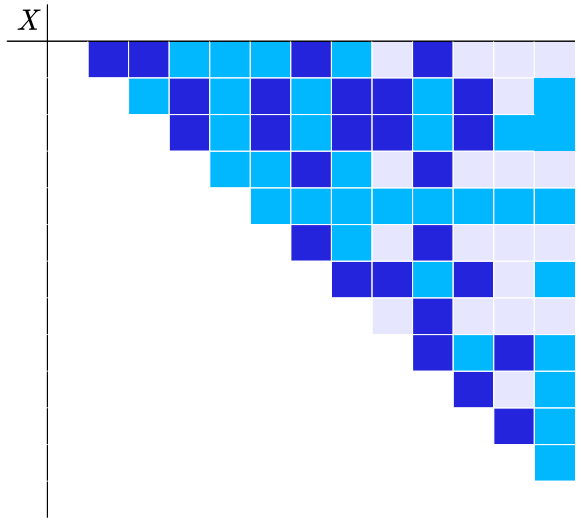




The clusters are exploited to target interesting edges:

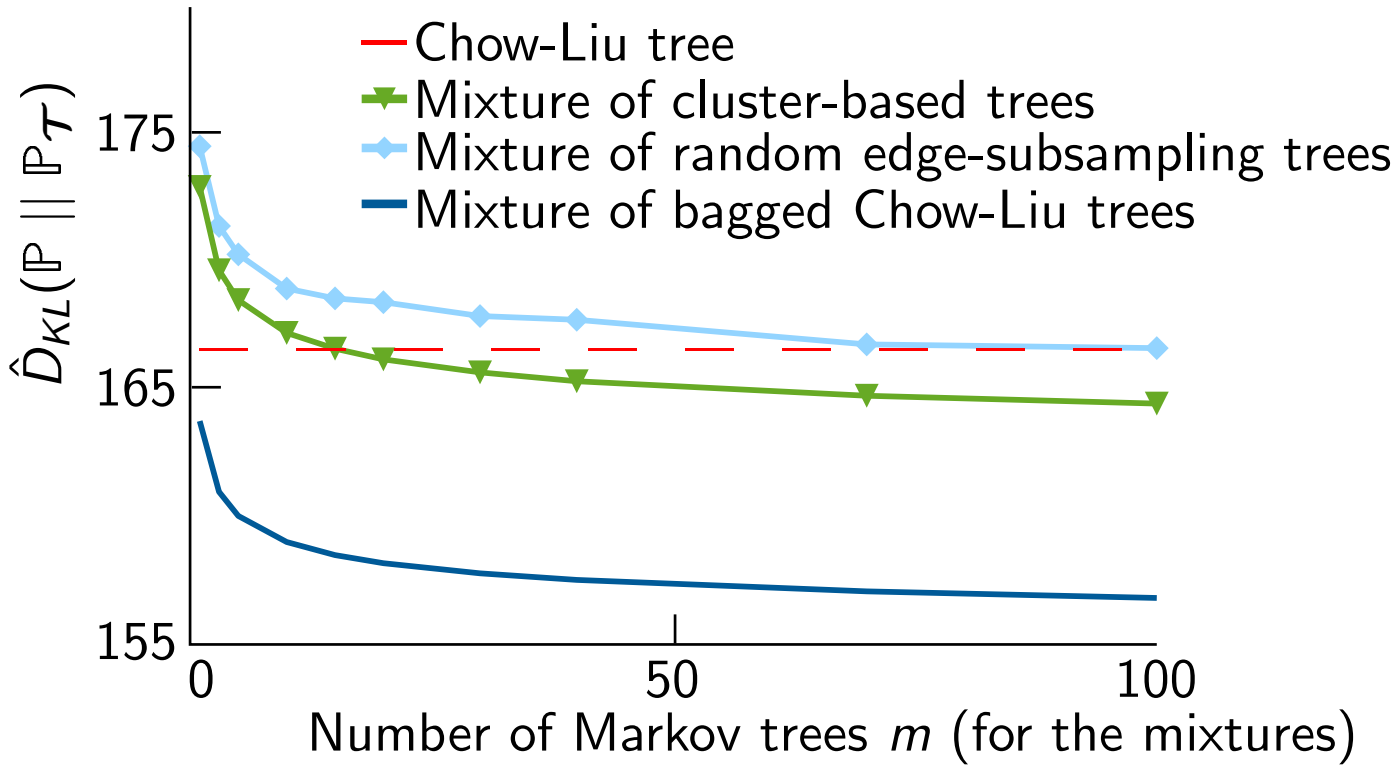
Computation of mutual information between variables belonging to neighboring clusters.





The mixtures developed can achieve a better accuracy than a single Chow-Liu tree

- Targeting interesting edges is better than random subsampling.



Averaged results on 10 randomly generated Bayesian networks (1000 binary variables) $\times$ 10 learning sets, for $N = 300$ samples.

Targeting interesting rather than random edges also leads to an improvement in computational complexity.

Table: Relative CPU times for training the Chow-Liu tree and 10×10 mixtures of size $m = 100$, cumulated on 100 data sets of 1000 samples and 200 variables. ($1 \approx 16.5$ hours)

	Random edge subsampling	1.08
	Cluster-based trees	1
	Bagged Chow-Liu trees	2.8
(	Single Chow-Liu tree	0.03)

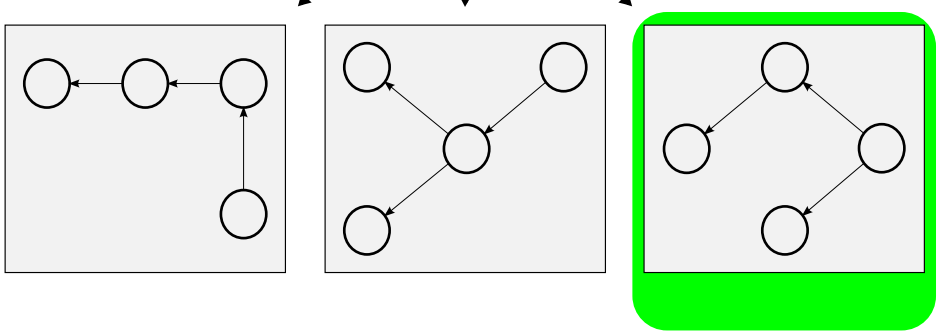
- This improvement might be a consequence of the determinism of the cluster-based learning procedure, once a root is chosen.
- However, the number of edges, and therefore the run-time, of this procedure is not directly controlable.

Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

1. Background
2. Contributions
   - Repeatedly learning a perturbed Markov tree
   - Building a sequence of Markov trees
   - Combining mixtures
3. Final words

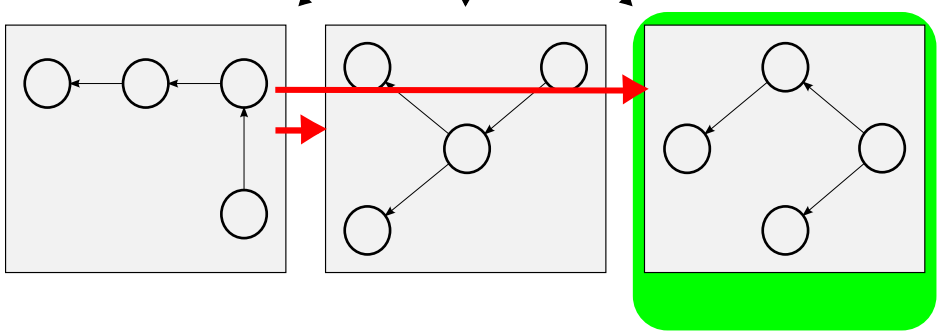
Previous chapter:

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
0	1	0	1
1	1	0	1
0	0	1	1
1	1	1	0



This chapter:

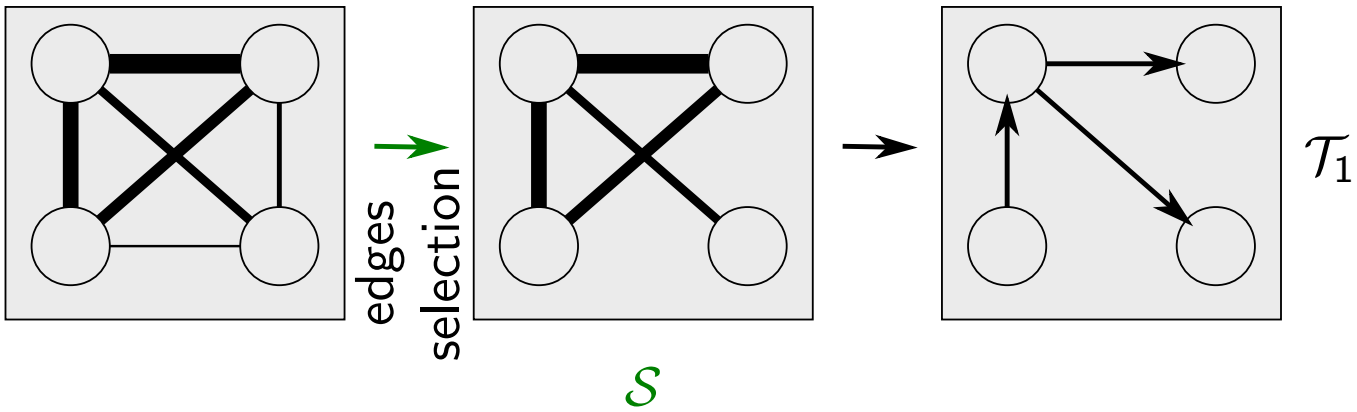
$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
0	1	0	1
1	1	0	1
0	0	1	1
1	1	1	0



Structural knowledge obtained from the first tree learning operation can guide the other tree learning procedures.

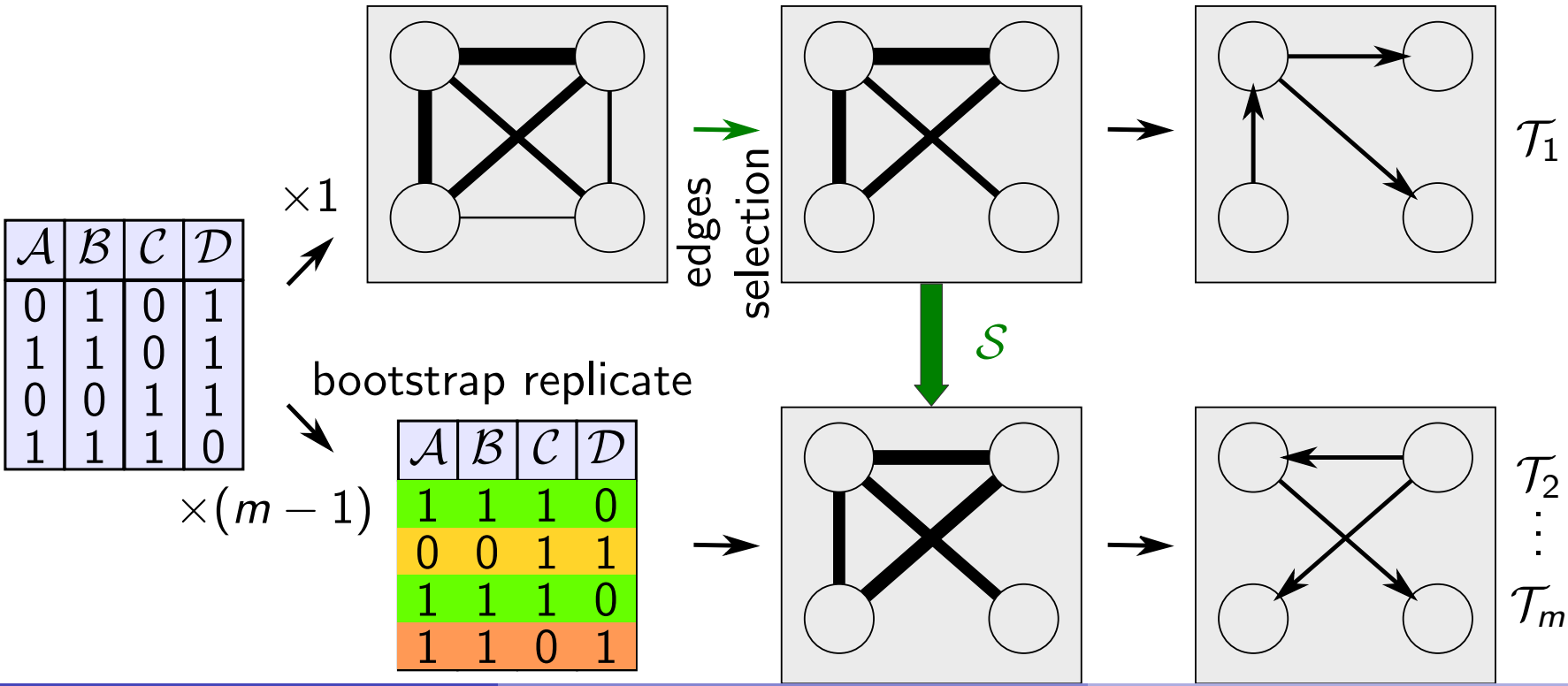
- Starting each tree learning operation from scratch throws away valuable information about the problem.
- Exploit the first tree to select a good subset $\mathcal{S}$ of candidate edges.
- The search is constrained to the tree (or forest) structures spanning $\mathcal{S}$.
- Complexity: $\mathcal{O}(mp^2 \log p) \rightarrow \mathcal{O}(p^2 \log p + m|\mathcal{S}| \log |\mathcal{S}|)$.

$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$
0	1	0	1
1	1	0	1
0	0	1	1
1	1	1	0



Structural knowledge obtained from the first tree learning operation can guide the other tree learning procedures.

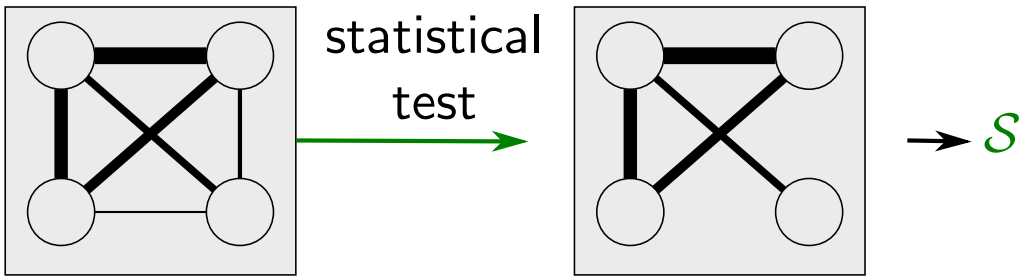
- Starting each tree learning operation from scratch throws away valuable information about the problem.
- Exploit the first tree to select a good subset $\mathcal{S}$ of candidate edges.
- The search is constrained to the tree (or forest) structures spanning $\mathcal{S}$.
- Complexity: $\mathcal{O}(mp^2 \log p) \rightarrow \mathcal{O}(p^2 \log p + m|\mathcal{S}| \log |\mathcal{S}|)$.



The skeleton $\mathcal{S}$ should contain edges with a strong associated mutual information.

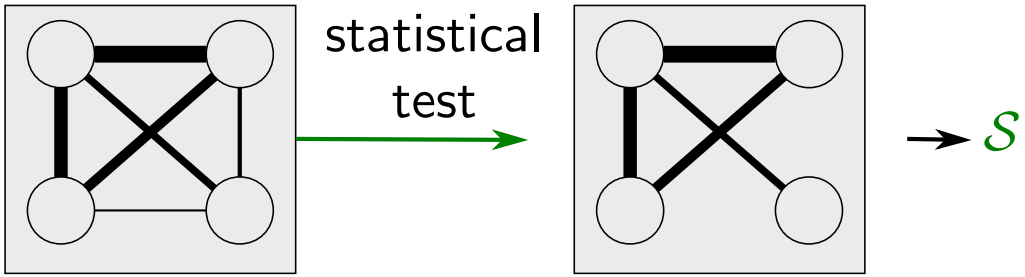
- Edges with weak weights are
  - not likelily to be part of a tree (even if weights are perturbed),
  - probably not meaningful (noise or not direct relation).

  $\rightarrow$ We can ignore them in the search.



Edges are included in $\mathcal{S}$ based on an independence test.

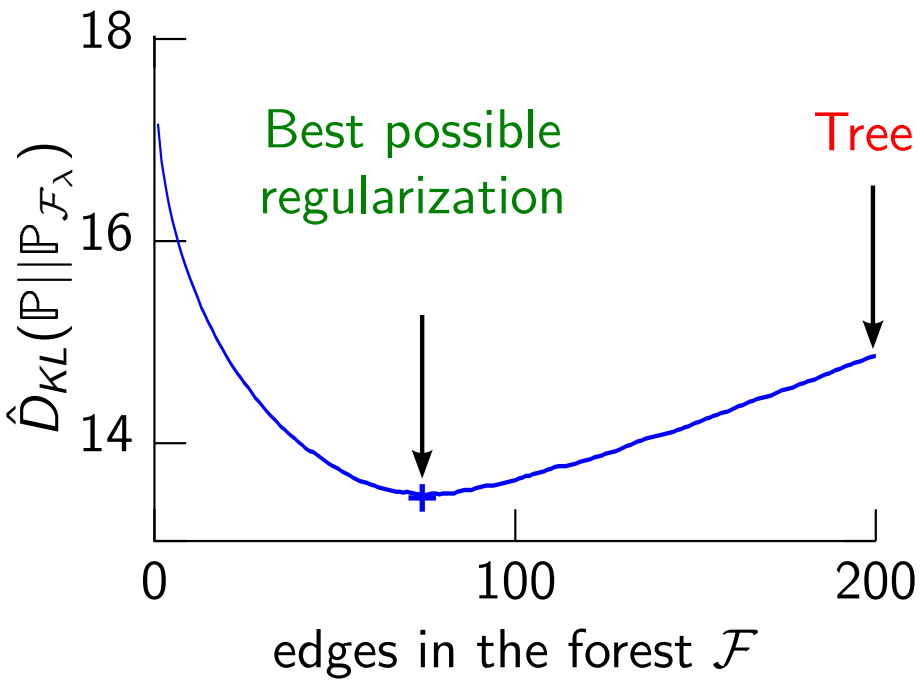
- Comparing $I_D(\mathcal{X};\mathcal{Y})$ (χ-square distributed under independence) to a threshold depending on a postulated p-value, say $\alpha = 0.05$ or smaller.
- $\mathcal{S}$ contains the pairs of variables whose mutual information (on the original data set) is above the threshold.
- Mutual information values are a by-product of the computation of the first tree.



Regularization is another way to reduce variance.

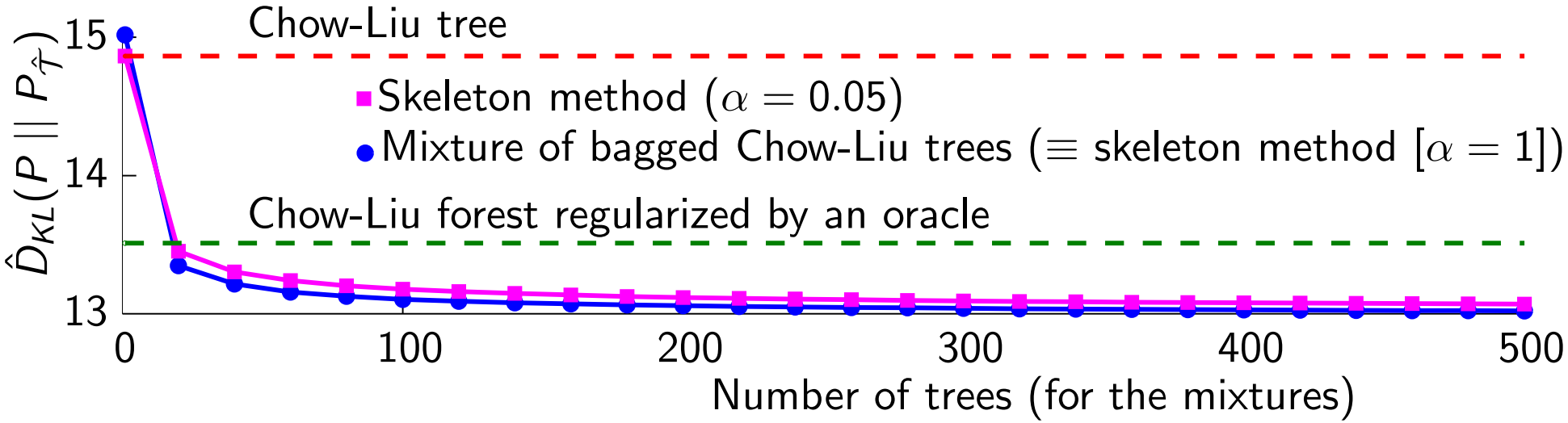
$$\mathcal{F}_{CL}(D) = \arg\max_{\mathcal{F}} \left[\sum_{(\mathcal{X},\mathcal{Y})\in E(\mathcal{F})} I_D(\mathcal{X};\mathcal{Y}) - \lambda |E(\mathcal{F})| \right]$$

New reference algorithm based on regularization:
λ is optimized to maximize the evaluation score (i.e. on the test set)
$\rightarrow$ Best possible regularization (optimistic score)



The skeleton method can achieve an accuracy close to a mixture of bagged Chow-Liu trees, but is faster.

The mixtures are better than the regularized Chow-Liu algorithm.

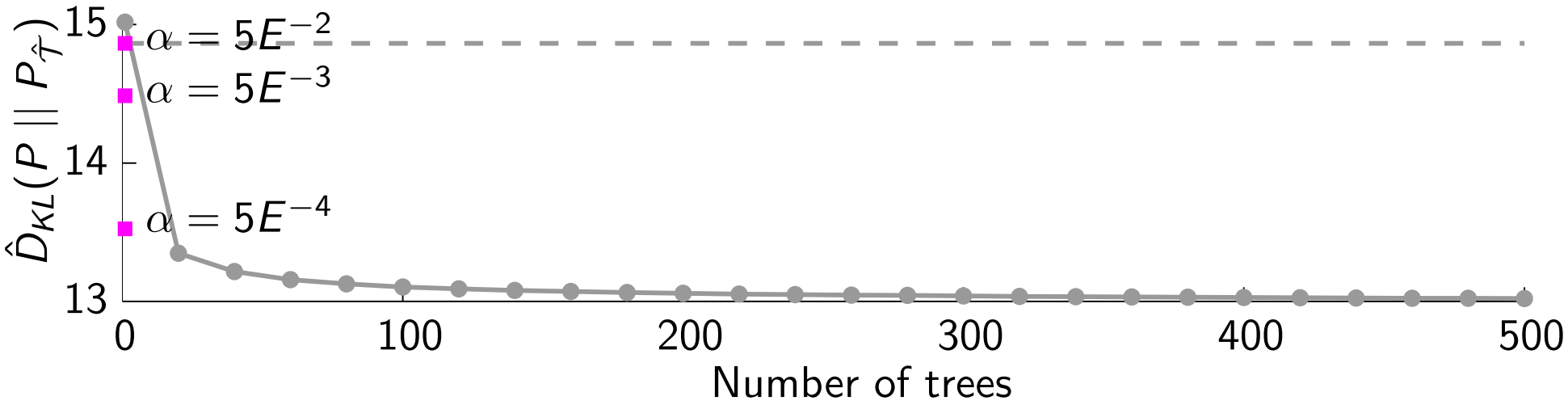


Relative run-time for mixtures of 500 trees (one max-likelihood tree: 1):

- Mixture of bagged Chow-Liu trees: 532
- Skeleton-based approximations: 21

Averaged results on 5 randomly generated Bayesian networks (200 binary variables) $\times$ 6 learning sets of 200 samples.

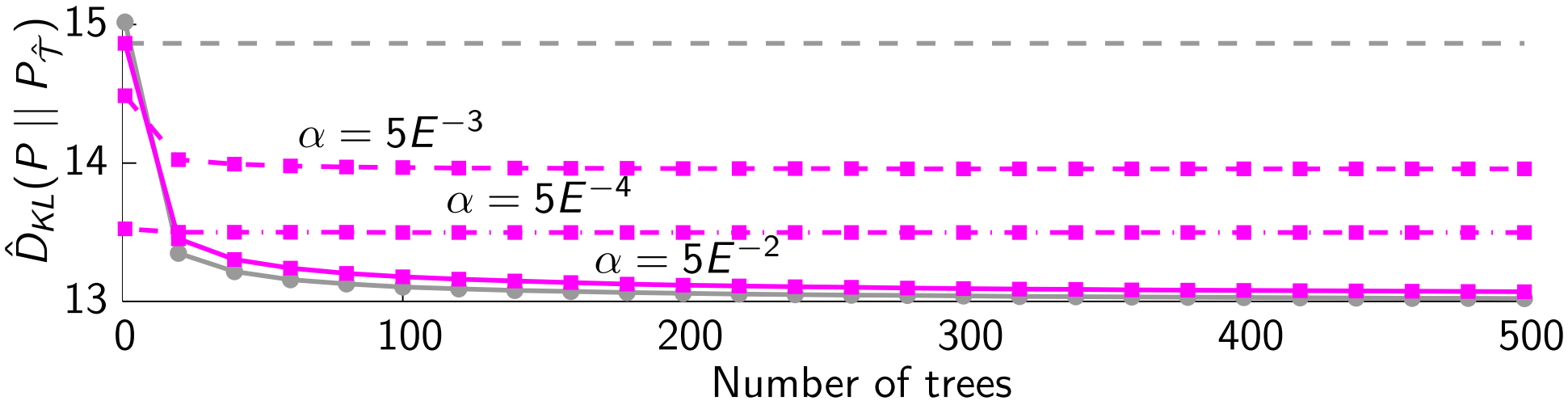
A closer look at the influence of α



- The smaller α, the smaller the variance of the first tree.
  - $\rightarrow$ Increase in accuracy (here).
    - Increase of the bias.
- The larger α, the slower the convergence, but the better the accuracy.
  - $\rightarrow$ Larger diversity in the Markov trees generated
  - $\rightarrow$ Better reduction of the variance by the mixture.
    - The bias of these Markov trees is also smaller.

Averaged results on 5 randomly generated Bayesian networks (200 binary variables) $\times$ 6 learning sets of 200 samples.

A closer look at the influence of α



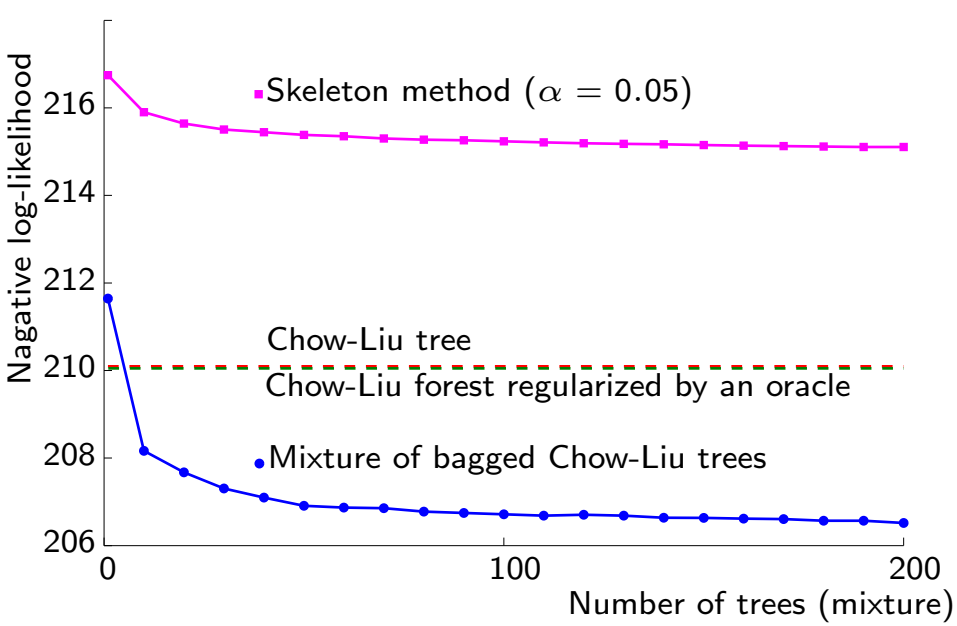
- The smaller α, the smaller the variance of the first tree.
  - $\rightarrow$ Increase in accuracy (here).
    - Increase of the bias.
- The larger α, the slower the convergence, but the better the accuracy.
  - $\rightarrow$ Larger diversity in the Markov trees generated
  - $\rightarrow$ Better reduction of the variance by the mixture.
    - The bias of these Markov trees is also smaller.

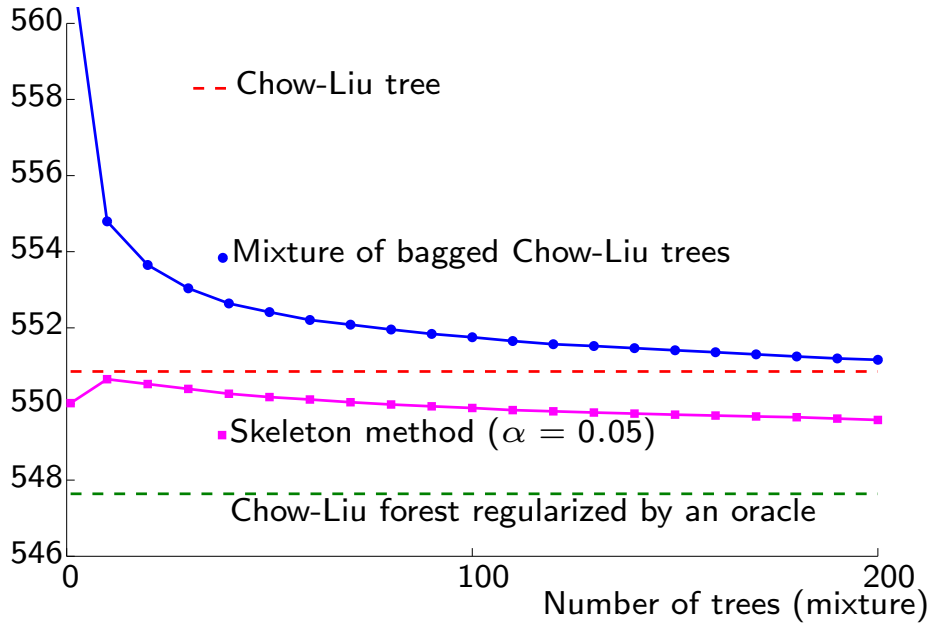
Averaged results on 5 randomly generated Bayesian networks (200 binary variables) $\times$ 6 learning sets of 200 samples.

Results on realistic learning set

On 8 more realistic models $\times$ 2 learning set sizes (200 and 500), the skeleton based method ($\alpha = 0.05$) achieves:

- a worse accuracy than bagging in 3/16 settings,
- an accuracy similar to bagging in 9/16 settings,
- a better accuracy than bagging in 4/16 settings.





Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

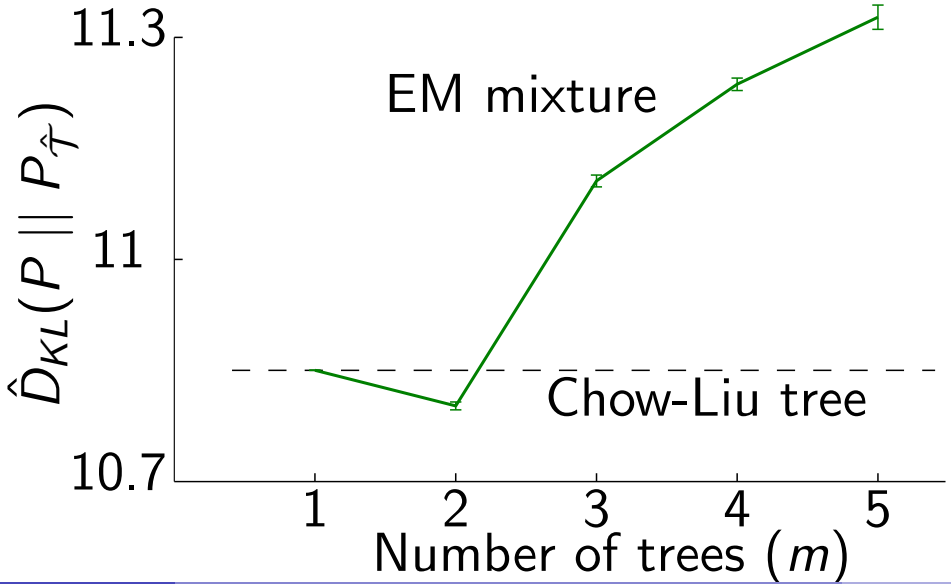
1. Background
2. Contributions
   - Repeatedly learning a perturbed Markov tree
   - Building a sequence of Markov trees
   - Combining mixtures
3. Final words

Mixtures can also be built to reduce the **bias** with respect to a single Chow-Liu tree.

Expectation-Maximization mixture

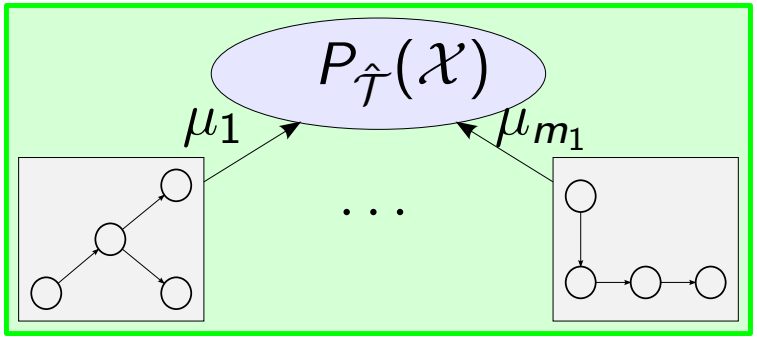
- Learning the mixture is viewed as a global optimization problem aiming at maximizing the data likelihood.
- There is a bias-variance trade-off associated to the number of terms.
- Soft partition of the learning set: each tree models a subset of observations.

Example: 200 variables and 2000 samples

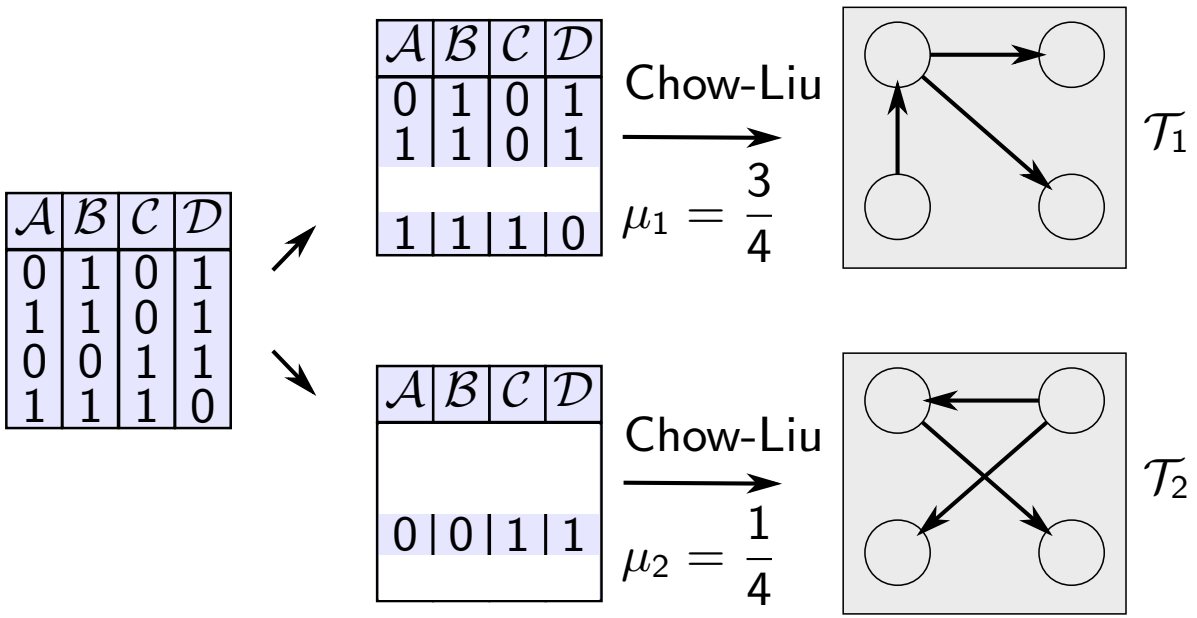


I combine the two types of mixtures.

1. Build an EM mixture and associated soft partition $\{\mathbf{D}_k\}_{k=1}^{m_1}$.
2. Replace each tree T_k by a variance reducing mixture learnt on $\mathbf{D}_k$.

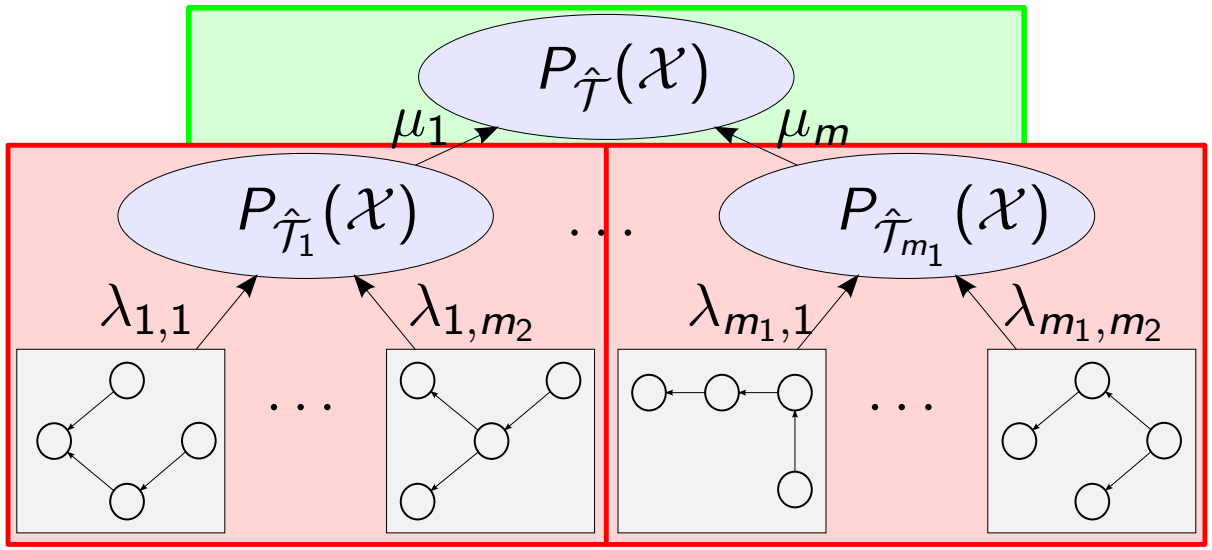


The EM algorithm creates a partition of the data set. E.g., for $m_1 = 2$:



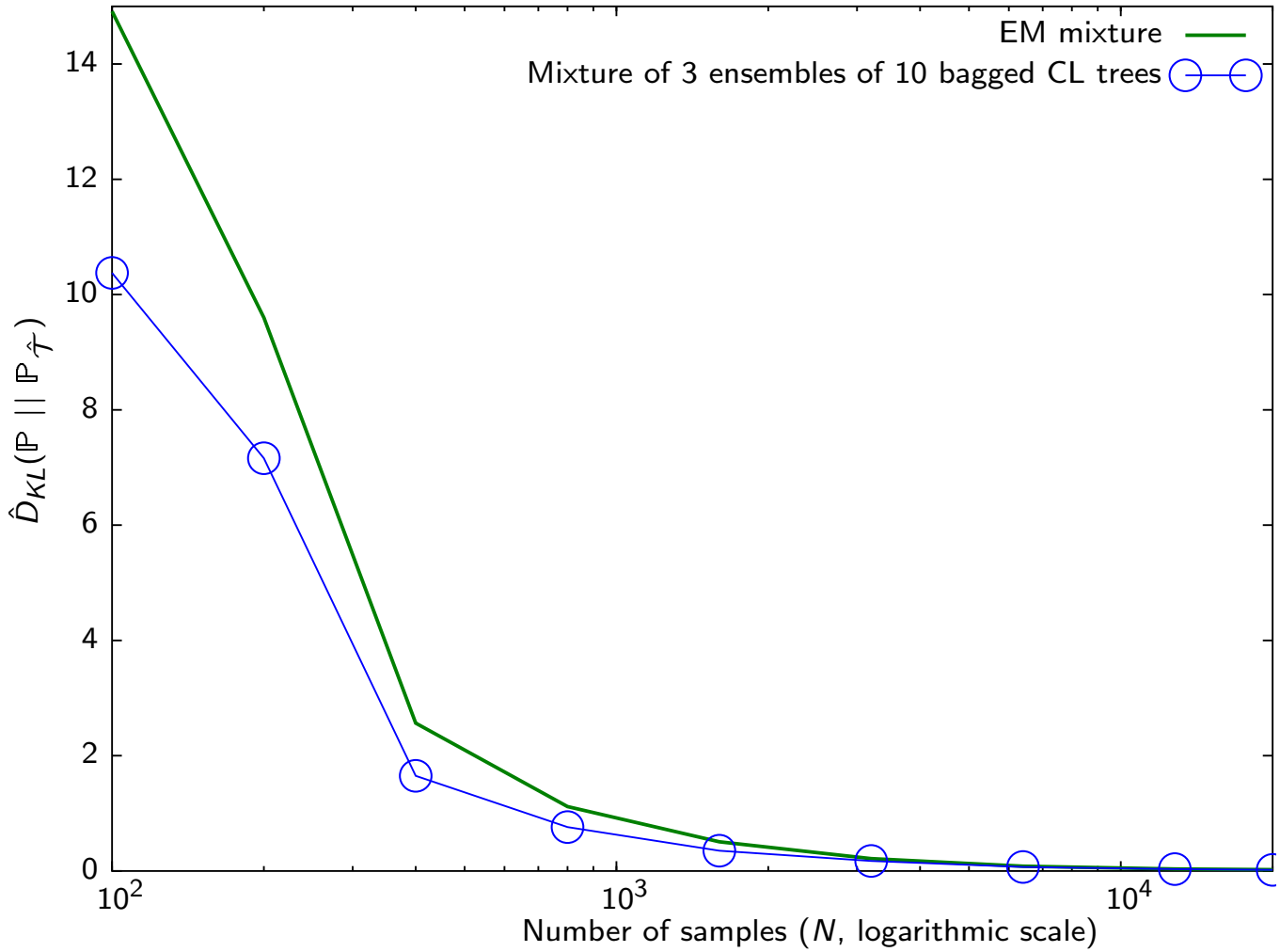
I combine the two types of mixtures.

1. Build an EM mixture and associated soft partition $\{\mathbf{D}_k\}_{k=1}^{m_1}$.
2. Replace each tree T_k by a variance reducing mixture learnt on $\mathbf{D}_k$.



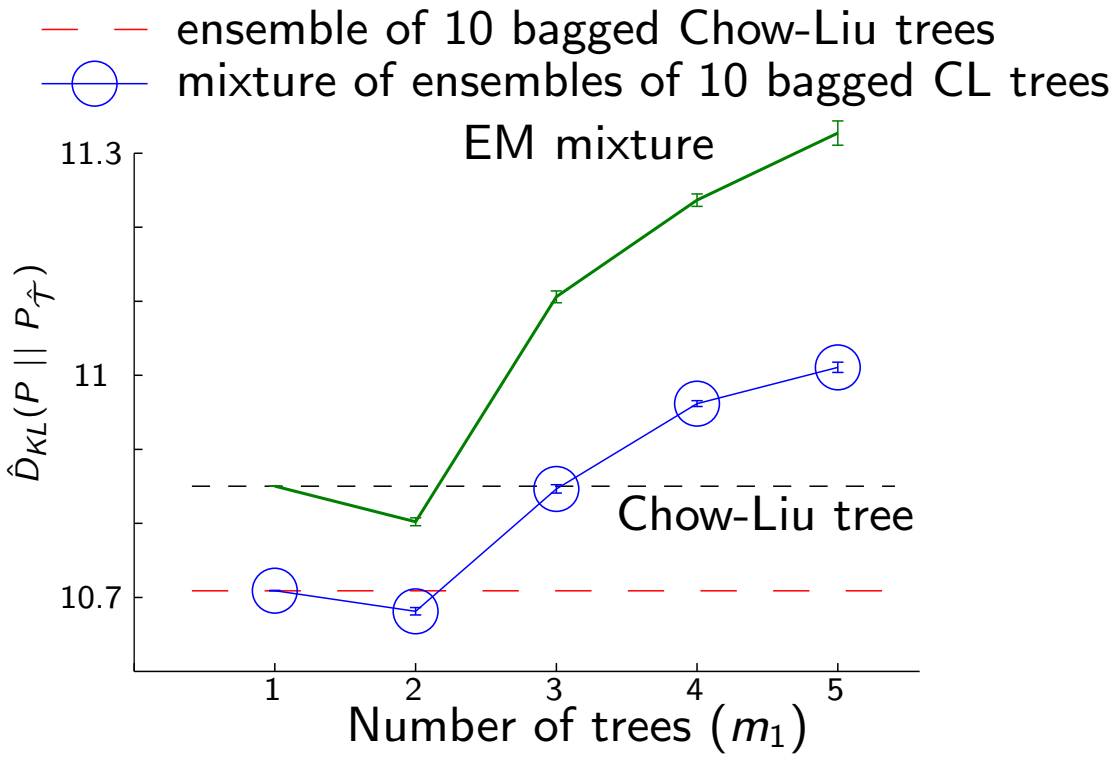
Each tree of the EM mixture is replaced by m_2 bagged Chow-Liu trees.

The combined method achieves a higher accuracy than an EM mixture for learning a mixture of 3 Markov trees.



Averaged results on 1 uniformly weighted mixture of 3 randomly generated Markov trees (200 binary variables) $\times$ 1 learning set $\times$ many initializations of the EM algorithm.

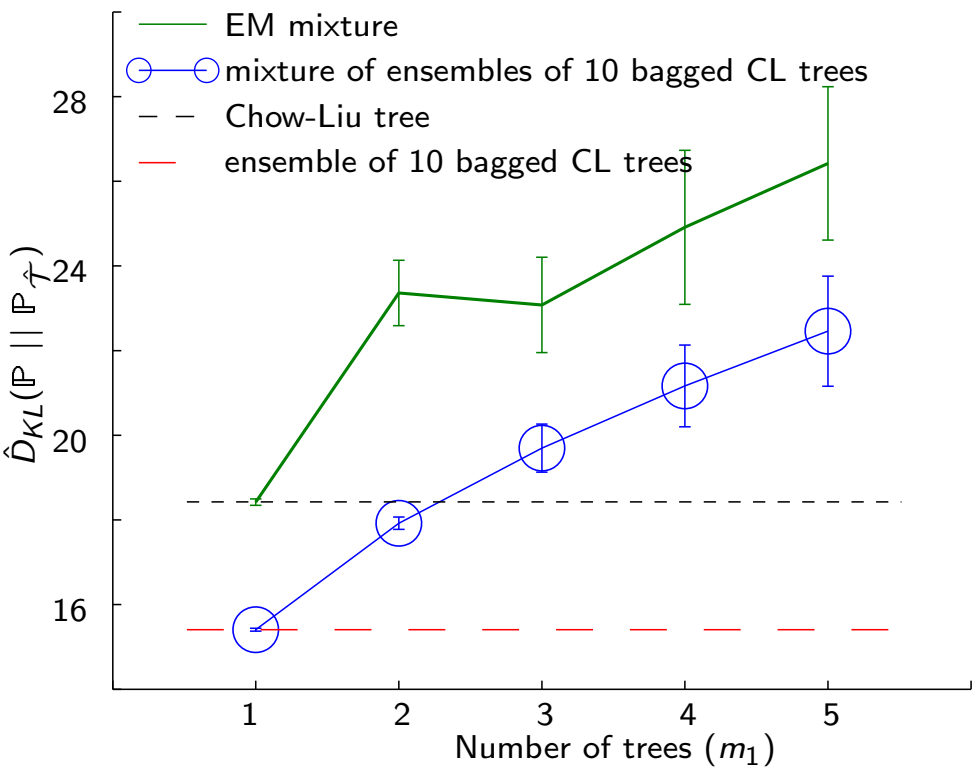
The mixture of ensembles of CL trees can lead to a better accuracy than both the EM mixture or an ensemble of bagged Chow-Liu trees.



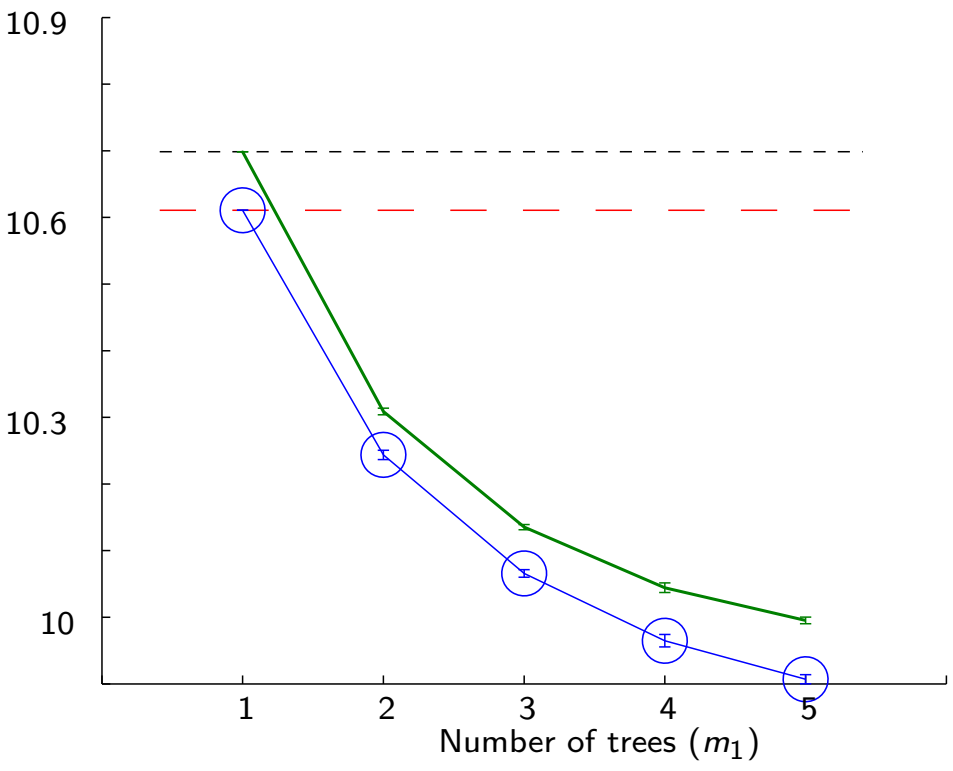
Averaged results on 5 randomly generated Bayesian networks (200 binary variables) $\times$ 6 learning sets of 2000 samples.

The mixture of ensembles of CL trees is better than the EM mixture, no matter what the optimal m_1 is.

120 samples



8000 samples



The 2-level mixture does not compensate the increase in variance when increasing m_1.

The gap between the EM and 2-level mixtures increases with m_1.

Mixtures of Tree-Structured Probabilistic Graphical Models for Density Estimation in High Dimensional Spaces

1. Background
2. Contributions
   - Repeatedly learning a perturbed Markov tree
   - Building a sequence of Markov trees
   - Combining mixtures
3. Final words

Main contributions

- Randomized Chow-Liu algorithm and corresponding mixtures
- Skeleton-based approximation
- Combining bias- and variance- reducing mixtures
- Extensive evaluation on synthetic and realistic probability densities
  - In particular, comparison to regularization

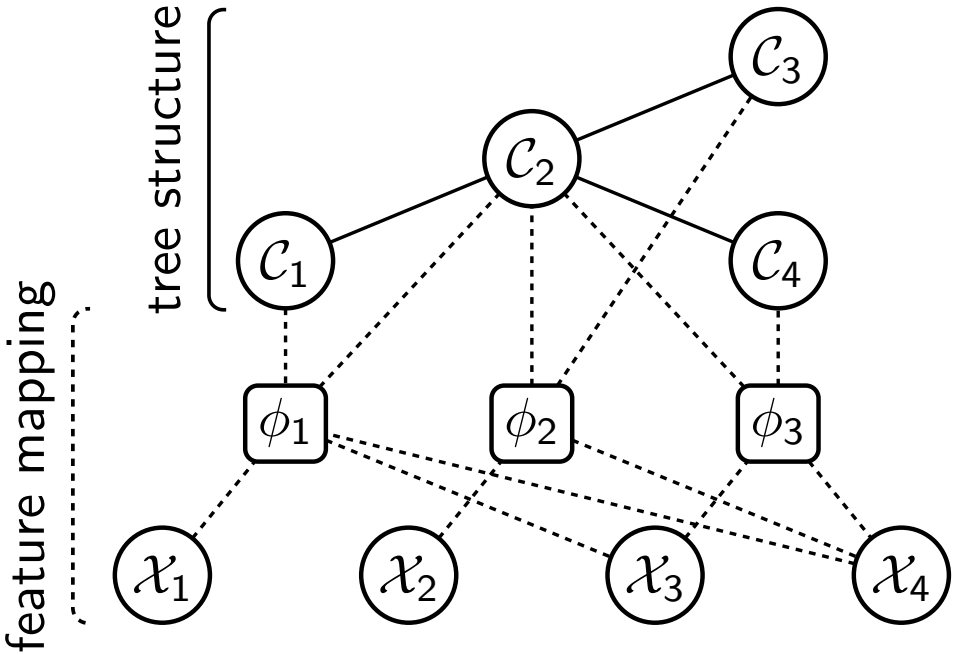
Short term perspectives

- Compare P&C mixtures to other methods.
  - true Bayesian approaches
  - other two-level mixtures
- Understand why and when mixtures are good and when the skeleton method is as good as bagging.
  - application related: automatically set a value for the parameters (e.g. α)
  - tool: better measure the bias and the 2 types of variance of the randomized methods

Long term perspectives

- New methods:
  - Regularization on each term of the mixture
  - New models:
    - ★ bounded tree-width models
    - ★ conditional random fields
- New experimental conditions:
  - Other class of target densities
  - Other types of variables
- Applications: electrical networks, bioinformatics

How can perturb and combine be applied to tree-structured CRFs?



Conditional random fields encode a conditional probability density:

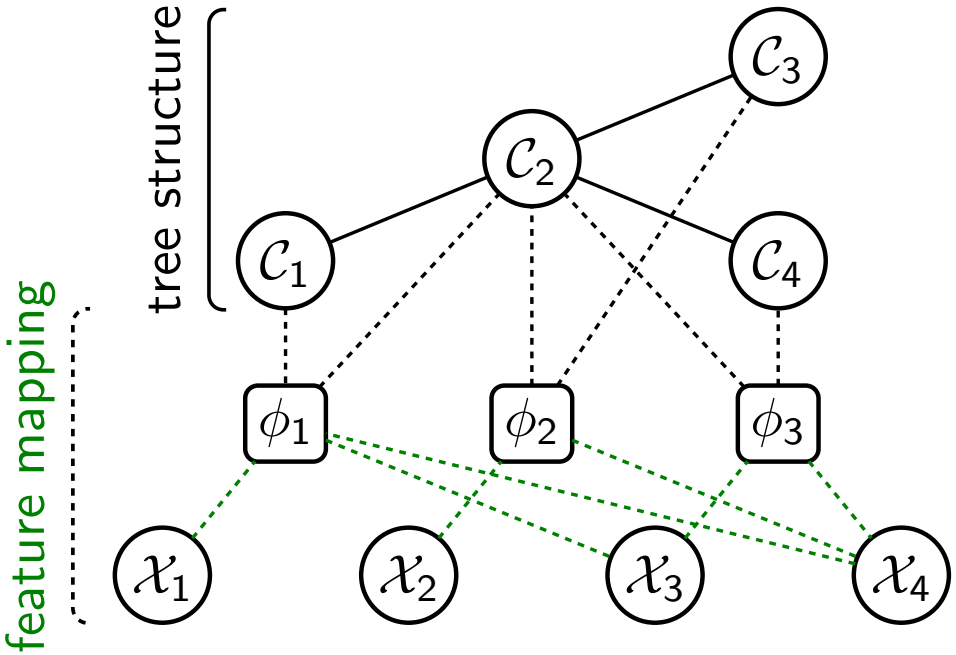
$$\mathbb{P}(\boldsymbol{\mathcal{C}}|\boldsymbol{\mathcal{X}}) = \frac{1}{Z(\boldsymbol{\mathcal{X}})}\phi_1(\mathcal{C}_1,\mathcal{C}_2|\mathcal{X}_1,\mathcal{X}_3,\mathcal{X}_4)$$
$$\phi_2(\mathcal{C}_2,\mathcal{C}_3|\mathcal{X}_2,\mathcal{X}_4)$$
$$\phi_3(\mathcal{C}_2,\mathcal{C}_4|\mathcal{X}_3,\mathcal{X}_4)$$

No modelling of $\mathbb{P}(\boldsymbol{\mathcal{X}})$!

New target for randomization with respect to mixtures of Markov trees networks:

- the feature mapping: what are the $\mathcal{X}$ associated to each edge $(\mathcal{C}_i,\mathcal{C}_j)$, $i \neq j$?
- parameter learning: less trivial (done by gradient descent).

How can perturb and combine be applied to tree-structured CRFs?



Conditional random fields encode a conditional probability density:

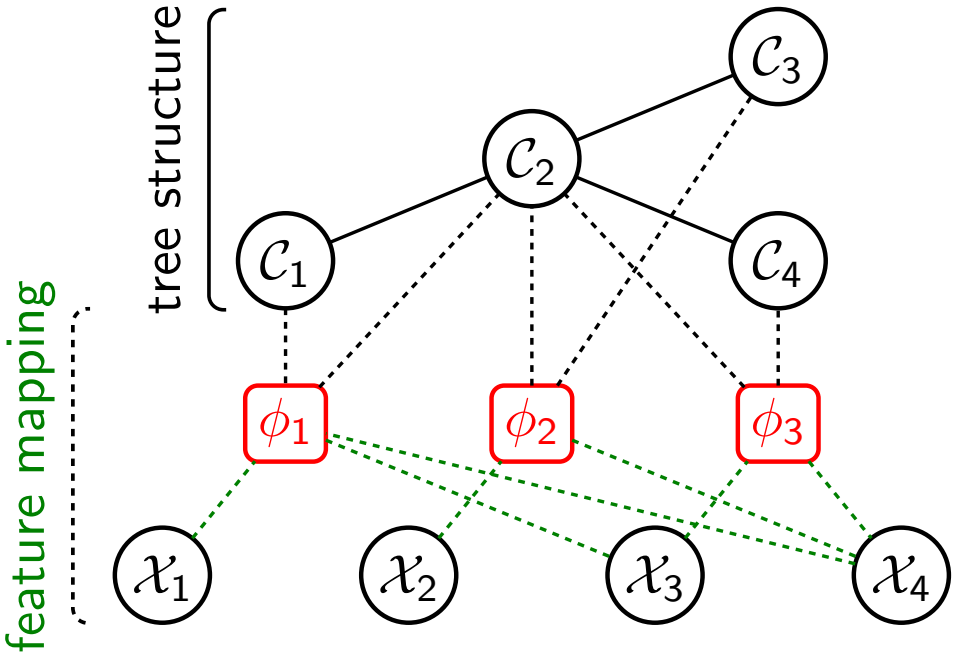
$$\mathbb{P}(\boldsymbol{\mathcal{C}}|\boldsymbol{\mathcal{X}}) = \frac{1}{Z(\boldsymbol{\mathcal{X}})}\phi_1(\mathcal{C}_1,\mathcal{C}_2|\mathcal{X}_1,\mathcal{X}_3,\mathcal{X}_4)$$
$$\phi_2(\mathcal{C}_2,\mathcal{C}_3|\mathcal{X}_2,\mathcal{X}_4)$$
$$\phi_3(\mathcal{C}_2,\mathcal{C}_4|\mathcal{X}_3,\mathcal{X}_4)$$

No modelling of $\mathbb{P}(\boldsymbol{\mathcal{X}})$!

New target for randomization with respect to mixtures of Markov trees networks:

- the feature mapping: what are the $\boldsymbol{\mathcal{X}}$ associated to each edge $(\mathcal{C}_i, \mathcal{C}_j)$, $i \neq j$?
- parameter learning: less trivial (done by gradient descent).

How can perturb and combine be applied to tree-structured CRFs?



Conditional random fields encode a conditional probability density:

$$\mathbb{P}(\boldsymbol{\mathcal{C}}|\boldsymbol{\mathcal{X}}) = \frac{1}{Z(\boldsymbol{\mathcal{X}})}\phi_1(\mathcal{C}_1,\mathcal{C}_2|\mathcal{X}_1,\mathcal{X}_3,\mathcal{X}_4)$$
$$\phi_2(\mathcal{C}_2,\mathcal{C}_3|\mathcal{X}_2,\mathcal{X}_4)$$
$$\phi_3(\mathcal{C}_2,\mathcal{C}_4|\mathcal{X}_3,\mathcal{X}_4)$$

No modelling of $\mathbb{P}(\boldsymbol{\mathcal{X}})$!

New target for randomization with respect to mixtures of Markov trees networks:

- the feature mapping: what are the $\boldsymbol{\mathcal{X}}$ associated to each edge $(\mathcal{C}_i, \mathcal{C}_j)$, $i \neq j$?
- parameter learning: less trivial (done by gradient descent).

Long term perspectives

- New methods:
  - Regularization on each term of the mixture
  - New models:
    - ★ bounded tree-width models
    - ★ conditional random fields
- New experimental conditions:
  - Other class of target densities
  - Other types of variables
- Applications: electrical networks, bioinformatics

More realistic data sets (by C. Aliferis, A. Statnikov, I. Tsamardinos & al).

Name	p	$\lvert\mathcal{X}_i\rvert$	$\lvert E(\mathcal{G})\rvert$	$\lvert\theta\rvert$
Alarm10	370	2-4	570	5468
Child10	200	2-6	257	2323
Gene	801	3-5	977	8348
Hailfinder10	560	2-11	1017	97448
Insurance10	270	2-5	556	14613
Link	724	2-4	1125	14211
Lung Cancer	800	2-3	1476	8452
Munin	189	1-21	282	15622
Pigs	441	3-3	592	3675

Table: Distributions from the literature and their characteristics. p corresponds to the number of variables, $|\mathcal{X}_i|$ to the range of cardinalities of single variables, $|E(\mathcal{G})|$ to the number of edges and $|\theta|$ to the number of independent parameters in the original model.

More terms might be necessary in the 2nd level ensemble when estimating more complex probability densities.

Data set	$N = 200$		$N = 500$		$N = 2500$	
	MT-EM	+bag, bagCl	MT-EM	+bag, bagCl	MT-EM	+bag, bagCl
Child10	-	25	-	25	-	10
Pigs	21	4	-	25	-	10
Alarm10	3	22	-	25	-	10
Gene	25	-	-	25	-	10
Lung Cancer	25	-	8	17	-	10
Link	3	22	-	25	-	10
Insurance10	2	23	1	24	-	10
Munin	1	24	6	19	-	10
Hailfinder10	25	-	25	-	-	10
ALL	105	120	40	185	0	90

Table: Best methods on realistic data sets (by increasing complexity) for 5 learning sets $\times$ several initializations of the EM mixture, with $m_1 = 2$ and $m_2 = 10$. N is the number of learning samples.