MLE for general BNs

- If we assume the parameters for each CPD are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node:

\[
\ell(\theta; D) = \log p(D | \theta) = \log \prod \prod p(x_{n,j} | x_{n-1}, \theta_i) = \sum \sum \log p(x_{n,j} | x_{n,n-1}, \theta_j)
\]

How to define parameter prior?

Factorization: \( p(X = x) = \prod_{i=1}^{M} p(x_i | x_{\pi_i}) \)

Local Distributions defined by, e.g., multinomial parameters:

\[
p(x_i^k | x_{\pi_i}) = \theta_{x_i^k | x_{\pi_i}}
\]

\[
p(\theta | G) \text{ ?}
\]
Global & Local Parameter Independence

- **Global Parameter Independence**
  
  For every DAG model:
  \[
  p(\theta \mid G) = \prod_{i=1}^{M} p(\theta_i \mid G)
  \]

- **Local Parameter Independence**
  
  For every node:
  \[
  p(\theta_i \mid G) = \prod_{j=1}^{q_i} p(\theta_{x_{i,j}} \mid G)
  \]

- **The Bayesian posterior**
  
  \[
  P(\theta \mid D, G) = \frac{P(D \mid \theta) P(\theta \mid G)}{\sum_{\theta} P(D \mid \theta) P(\theta \mid G)}
  = \prod_{i,j} p(x_i \mid x_{i,j}, \theta_{i,j}) P(\theta_{i,j} \mid G)
  \]

Example: decomposable likelihood of a directed model

- Consider the distribution defined by the directed acyclic GM:
  \[
  p(x \mid \theta) = p(x_1 \mid \theta_1) p(x_2 \mid x_1, \theta_1) p(x_3 \mid x_1, \theta_1) p(x_4 \mid x_2, x_3, \theta_1)
  \]

- \( \theta_1^* = \text{argmax} \ p(x \mid \theta) = \text{argmax} \ p(x \mid \theta_1, G) \)

- This is exactly like learning four separate small BNs, each of which consists of a node and its parents.
MLE for BNs with tabular CPDs

- Assume each CPD is represented as a table (multinomial) where
  \[ \theta_{ik} \overset{\text{def}}{=} p(X_j = j \mid X_{-j} = k) \]
  - Note that in case of multiple parents, \( X_{-j} \) will have a composite state, and the CPD will be a high-dimensional table.
  - The sufficient statistics are counts of family configurations.
  \[ n_{ijk} \overset{\text{def}}{=} \sum x_{ij}^k \]
- The log-likelihood is
  \[ \ell(\theta, D) = \log \prod_{t,j,k} \theta_{ijk}^{n_{ijk}} = \sum_{t,j,k} n_{ijk} \log \theta_{ijk} \]
- Using a Lagrange multiplier to enforce \( \sum_j \theta_{ijk} = 1 \), we get:
  \[ \theta_{ijk}^{\text{ML}} = \frac{n_{ijk}}{\sum_j n_{ijk}} \]

Parameter sharing

- Consider a time-invariant (stationary) 1st-order Markov model
  - Initial state probability vector: \( \pi_1 \overset{\text{def}}{=} p(X_1 = 1) \)
  - State transition probability matrix: \( A_t \overset{\text{def}}{=} p(X_t = 1 \mid X_{t-1} = 1) \)
- The joint:
  \[ p(X_{1:T} \mid \theta) = p(x_1 \mid \pi) \prod_{t=2}^T p(X_t \mid X_{t-1}) \]
- The log-likelihood:
  \[ \ell(\theta, D) = \sum_n \log p(x_n \mid \pi) + \sum_{t=2}^T \sum_n \log p(x_n \mid x_{n-1}, A) \]
- Again, we optimize each parameter separately
  - \( \pi \) is a multinomial frequency vector, and we’ve seen it before
  - What about \( A \)?
Learning a Markov chain transition matrix

- $A$ is a stochastic matrix: $\sum_i A_{ij} = 1$
- Each row of $A$ is a multinomial distribution.
- So MLE of $A_{ij}$ is the fraction of transitions from $i$ to $j$

$$A_{ij}^{ML} = \frac{\#(i \to j)}{\#(i \to \bullet)} = \frac{\sum_{t=2}^{T} x_{i,j,t-1} x_{i,j,t}}{\sum_{t=1}^{T} x_{i,j,t-1}}$$

- Application:
  - if the states $X_t$ represent words, this is called a bigram language model
- Sparse data problem:
  - If $i \to j$ did not occur in data, we will have $A_{ij} = 0$, then any further sequence with word pair $i \to j$ will have zero probability.
  - A standard hack: backoff smoothing or deleted interpolation

$$\tilde{A}_{j \to \bullet} = \lambda \eta_j + (1 - \lambda) A_{j \to \bullet}^{ML}$$

Bayesian language model

- Global and local parameter independence
- The posterior of $A_{i \to}$ and $A_{i' \to}$ is factorized despite v-structure on $X_t$, because $X_{t-1}$ acts like a multiplexer
- Assign a Dirichlet prior $\beta_i$ to each row of the transition matrix:

$$A_{ij}^{Bayes} \overset{def}{=} p(j | i, D, \beta) = \frac{\#(i \to j) + \beta_{ij} \lambda}{\#(i \to \bullet) + |D\beta|} = \lambda \beta_{ij} + (1 - \lambda) A_{ij}^{ML}, \text{ where } \lambda = \frac{|\beta|}{|\beta| + \#(i \to \bullet)}$$

- We could consider more realistic priors, e.g., mixtures of Dirichlets to account for types of words (adjectives, verbs, etc.)
Recall definition of HMM

- Transition probabilities between any two states

\[ p(y'_i = 1 | y'_{i-1} = 1) = a_{i,j}, \]

or

\[ p(y_i | y'_{i-1} = 1) \sim \text{Multinomial}(a_{i,1}, a_{i,2}, \ldots, a_{i,U}) \forall i \in 1. \]

- Start probabilities

\[ p(y_1) \sim \text{Multinomial}(\pi_1, \pi_2, \ldots, \pi_U) \]

- Emission probabilities associated with each state

\[ p(x_i | y'_i = 1) \sim \text{Multinomial}(h_{i,1}, h_{i,2}, \ldots, h_{i,K}) \forall i \in 1. \]

or in general:

\[ p(x_i | y'_i = 1) \sim \text{f}(\cdot | \theta), \forall i \in 1. \]

Example: HMM: two scenarios

- **Supervised learning**: estimation when the “right answer” is known

  - Examples:

    GIVEN: a genomic region \( x = x_1 \ldots x_{1,000,000} \) where we have good (experimental) annotations of the CpG islands

    GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

- **Unsupervised learning**: estimation when the “right answer” is unknown

  - Examples:

    GIVEN: the porcupine genome; we don’t know how frequent are the CpG islands there, neither do we know their composition

    GIVEN: 10,000 rolls of the casino player, but we don’t see when he changes dice

- **QUESTION**: Update the parameters \( \theta \) of the model to maximize \( P(x | \theta) \) - Maximal likelihood (ML) estimation
Supervised ML estimation

- Given \( x = x_1 \ldots x_N \) for which the true state path \( y = y_1 \ldots y_N \) is known,

- Define:
  \[ A_{ij} = \# \text{times state transition } i \rightarrow j \text{ occurs in } y \]
  \[ B_{ik} = \# \text{times state } i \text{ in } y \text{ emits } k \text{ in } x \]

- We can show that the maximum likelihood parameters \( \theta \) are:

  \[ a_{ij}^M = \frac{\#(i \rightarrow j)}{\#(i \rightarrow \bullet)} = \frac{\sum_{t=1}^{T} y_{t}^i y_{t+1}^j}{\sum_{t=1}^{T} y_{t}^i} \quad A_{ij} \]
  \[ b_{ik}^M = \frac{\#(i \rightarrow k)}{\#(i \rightarrow \bullet)} = \frac{\sum_{t=1}^{T} y_{t}^i x_{t}^k}{\sum_{t=1}^{T} y_{t}^i} \quad B_{ik} \]

- What if \( x \) is continuous? We can treat \( \{x_{t_1}, y_{n_1} ; t=1:T, n=1:N\} \) as \( N \times T \) observations of, e.g., a Gaussian, and apply learning rules for Gaussian ...

Learning BN Structure

Probabilistic Graphical Models (10-708)
Lecture 10, Oct 17, 2007

Reading: KF-Chap. 16
ML Structural Learning for completely observed GMs

Where are we now on the map?

- Graphical models
  - Bayesian networks
  - Undirected models
  - Conditional independence statements + factorization law of joint dist.
- Exact inference in GMs
  - Variable elimination <=> Graph elimination
  - Sum-product on tree, factor tree, clique tree
  - Very fast for models with low tree-width
- Learning GMs
  - Given structure, estimate parameters
    - Maximum likelihood estimation (just counts for BNs)
    - Bayesian learning
    - MAP for Bayesian learning
- What about learning structure?
Learning the structure of a BN

- **Data**
  - \((x_1^{(1)}, ..., x_n^{(1)})\)
  - \((x_1^{(2)}, ..., x_n^{(2)})\)
  - ...
  - \((x_1^{(M)}, ..., x_n^{(M)})\)

**Possible structures** → **Learn parameters**

- Maximum likelihood
- Bayesian
- Conditional likelihood
- Margin

**Score struc/param**
- \(10^{-5}\)
- \(10^{-3}\)
- \(10^{-15}\)
- ...

**Constraints**
- \(I(G_j) \in I(P)\)
- \(I(G_k) \in I(P)\)
- ...

---

Learning the structure of a BN

- **Constraint-based approach**
  - BN encodes conditional independencies
  - Test conditional independencies in data
  - Find an I-map

- **Score-based approach**
  - Finding a structure and parameters is a density estimation task
  - Evaluate model as we evaluated parameters
  - Maximum likelihood
  - Bayesian
  - etc.
Recall P-Map

- **Defn (3.4.3):** We say that a graph object G is a perfect map (P-map) for a set of independencies I if we have that I(G) = I. We say that G is a perfect map for P if I(G) = I(P).
  - Not all P has a perfect map as DAG!
  - The P-map of a distribution is unique up to I-equivalence between networks. That is, a distribution P can have many P-maps, but all of them are I-equivalent.
  - The P-DAG algorithm

- **Constraint-based approach:**
  - Key question: Independence test

Constraint-based approach: Independence tests

- Statistically difficult task!
- Intuitive approach:
  - Mutual information
    \[ I(X_i, X_j) = \sum_{x_i, x_j} \log P(x_i, x_j) \frac{P(x_i, x_j)}{P(x_i)P(x_j)} \]
  - Mutual information and independence:
    - \( X_i \) and \( X_j \) are independent if and only if \( I(X_i, X_j) = 0 \)
  - **Conditional mutual information:**
    \[ I(X_i, X_j | Z) \]
Empirical independence tests

- Using the data $D$
  - Empirical distribution:
    $$\hat{P}(x_i, x_j) = \frac{\text{count}(x_i, x_j)}{M}$$
  - Mutual information:
    $$\hat{I}(X_i, X_j) = \sum_{i,j} \log \frac{\hat{P}(x_i, x_j)}{\hat{P}(x_i)\hat{P}(x_j)}$$
  - Similarly for conditional MI

- More generally, use learning PDAG algorithm:
  - When algorithm asks: $(X, Y | U)$?
  - Must check if statistically-significant
  - Choosing $t$
  - See reading...

Score-based approach:

- Desirable properties of a scoring function
  - **Consistency:** i.e., if the data is generated by $G^*$, then $G$ and all I-equivalent models maximize the score.
  - Decomposability:
    $$\text{Score}(G \mid D) = \sum_i \text{FamScore}(D(X_i \mid X_{-i}))$$
    which makes it cheap to compare score of $G$ and $G'$ if they only differ in a small number of families.

- Bayesian score (evidence), likelihood, and penalized likelihood (BIC) are all decomposable and consistent.
Maximizing the score

- Consider the family of DAGs $G_d$ with maximum fan-in (number of parents) equal to $d$.

- **Thm**: It is NP-hard to find
  \[
  G^* = \arg \max_{G \in G_d} \text{Score}(G | D)
  \]
  for any $d \geq 2$.

- In general, we need to use heuristic local search

  - For $d \leq 1$ (i.e., trees), we can solve the problem in $O(n^2)$ time using max spanning tree (forthcoming)
  - If we know the ordering of the nodes, we can solve the problem in $O(d^\frac{n}{d})$ time

---

Information Theoretic Interpretation of ML

\[
\ell(\theta_G, G; D) = \log p(D | \theta_G, G)
\]
\[
= \log \prod_x \left( \prod_x p(x_{i|x(G)}, \theta_{G(i)}) \right)
\]
\[
= \sum_x \left( \sum_{x_{i|x(G)}} \log p(x_{i|x(G)}, \theta_{G(i)}) \right)
\]
\[
= M \sum_i \left( \sum_{x_{i|x(G)}} \frac{\text{count}(x_i, x_{i|x(G)})}{M} \log p(x_i | x_{i|x(G)}, \theta_{G(i)}) \right)
\]
\[
= M \sum_i \left( \sum_{x_{i|x(G)}} \hat{p}(x_i, x_{i|x(G)}) \log p(x_i | x_{i|x(G)}, \theta_{G(i)}) \right)
\]

From sum over data points to sum over count of variable states
Information Theoretic Interpretation of ML (con'd)

\[ \ell(\theta_{\phi}, G; D) = \log \hat{p}(D | \theta_{\phi}, G) \]

\[ = M \sum \left( \sum \hat{p}(x_i, x_{\pi(x_i)}) \log \hat{p}(x_i | x_{\pi(x_i)}, \theta_{\pi(x_i)}) \right) \]

Decomposable score and a function of the graph structure

Decomposable Score

- Log data likelihood
  \[ \ell(\theta_{\phi}, G; D) = \log \hat{p}(D | \theta_{\phi}, G) \]
  \[ = M \sum \hat{I}(x_i, x_{\pi(x_i)}) - M \sum \hat{H}(x_i) \]

- Decomposable score:
  - Decomposes over families in BN (node and its parents)
  - Will lead to significant computational efficiency!!!
  - The score function:
    \[ \text{Score}(G | D) = \sum \text{FamScore}(D(X_i | X_{\pi_i})) \]

- Search space:
Structural Search

- How many graphs over \( n \) nodes? \( O(2^n) \)
- How many trees over \( n \) nodes? \( \mathcal{O}(2^{n\log n}) \)
- But it turns out that we can find exact solution of an optimal tree (under MLE)!
  - Trick: in a tree each node has only one parent!
  - Chow-liu algorithm

Scoring a tree 1: equivalent trees

\[
\ell(\theta; G; D) = M \sum_i \hat{I}(x_i; x_{x_i(G)}) - M \sum_i \hat{H}(x_i)
\]
Scoring a tree 2: similar trees

\[ \ell(\theta_i, G; D) = M \sum_i \hat{I}(x_i, x_{x_i(0)}) - M \sum_i \hat{H}(x_i) \]

Chow-Liu tree learning algorithm

- **Objection function:**
  \[ \ell(\theta_i, G; D) = \log \hat{p}(D | \theta_i, G) = M \sum_i \hat{I}(x_i, x_{x_i(0)}) - M \sum_i \hat{H}(x_i) \]
  \[ \Rightarrow C(G) = M \sum_i \hat{I}(x_i, x_{x_i(0)}) \]

- **Chow-Liu:**
  - For each pair of variable \( x_i \) and \( x_j \)
    - Compute empirical distribution:
      \[ \hat{p}(X_i, X_j) = \frac{\text{count}(x_i, x_j)}{M} \]
    - Compute mutual information:
      \[ \hat{I}(X_i, X_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i) \hat{p}(x_j)} \]
  - Define a graph with node \( x_1, \ldots, x_n \)
  - Edge (i,j) gets weight \( \hat{I}(X_i, X_j) \)
Chow-Liu algorithm (con'd)

- Objection function:
  \[
  \ell(\theta_0, G; D) = \log \hat{p}(D \mid \theta_0, G) = M \sum_i \hat{I}(x_i, x_{x_0(G_i)}) - M \sum_i \hat{H}(x_i) \Rightarrow C(G) = M \sum_i \hat{I}(x_i, x_{x_0(G_i)})
  \]

- Chow-Liu:
  
  Optimal tree BN
  - Compute maximum weight spanning tree
  - Direction in BN: pick any node as root, do breadth-first-search to define directions
  - I-equivalence:

\[
C(G) = I(A, B) + I(A, C) + I(C, D) + I(C, E)
\]

Extensions of Chow-Liu

- Tree augmented naïve Bayes(TAN) [Friedman et al. ’97]
  - Naïve Bayes model overcounts, because correlation between features not considered
  - Tree-augmented feature list

- Same as Chow-Liu, but score edges w

\[
\hat{p}(X_i, X_j \mid C) = \frac{\text{count}(x_i, x_j \mid C)}{M} \quad \hat{I}(X_i, X_j) = \sum_{i, j} \hat{p}(x_i, x_j \mid C) \log \frac{\hat{p}(x_i, x_j \mid C)}{\hat{p}(x_i \mid C) \hat{p}(x_j \mid C)}
\]
Structure Learning for general graphs

- **Theorem:**
  - The problem of learning a BN structure with at most $d$ parents is NP-hard for any (fixed) $d \geq 2$

- **Most structure learning approaches use heuristics**
  - Exploit score decomposition
  - Two heuristics that exploit decomposition in different ways
    - Greedy search through space of node-orders
    - Local search of graph structures
**Known order (K2 algorithm)**

- Suppose we have a total ordering of the nodes $X_1 < X_2 < \cdots < X_n$ and want to find a DAG consistent with this with maximum score.
  - The choice of parents for $X_i$ from $\text{Pa}(X_i, \ldots, X_{i-1})$, is independent of the choice for $X_j$ since we obey the ordering, we cannot create a cycle.
  - Hence we can pick the best set of parents for each node independently.
  - For $X_i$, we need to search all $\binom{d-1}{d}$ subsets of size up to $d$ for the set which maximizes FamScore.
  - We can use greedy techniques for this, c.f., learning a decision tree.

- What if order isn’t known
  - Search in the space of orderings, then conditioned on , pick best graph using K2
  - Search in the space of DAGs.

**Learn BN structure using local search**

Starting from Chow-Liu tree  \rightarrow Local search  \rightarrow Select using favorite score

**Possible moves:**
- Add edge
- Delete edge
- Invert edge

Only if acyclic!!!

$10^5$  
$10^3$  
$10^{-15}$  
$\cdots$
Exploit score decomposition in local search

- Add edge and delete edge
  - Only rescore one family

- Reverse edge
  - Rescore only two families

- Simplest search algorithm: greedy hill climbing.

\[ I(J, S) \Rightarrow I(J, S, I) \]

Local maxima

- Greedy hill climbing will stop when it reaches a local maximum or a plateau (a set of neighboring networks that have the same score).

- Unfortunately, plateaus are common, since equivalence classes form contiguous regions of search space (thm 14.4.4), and such classes can be exponentially large.

- Solutions:
  - Random restarts
  - TABU search (prevent the algorithm from undoing an operator applied in the last L steps, thereby forcing it to explore new terrain).
  - Data perturbation (dynamic local search): reweight the data and take step.
  - Simulated annealing: if \( x(o) > 0 \), take move, else accept with probability \( e^{x(o)/t} \), where \( t \) is the temperature. Slow!
Order search versus graph search

- **Order search advantages**
  - For fixed order, optimal BN – more “global” optimization
  - Space of orders much smaller than space of graphs

- **Graph search advantages**
  - Not restricted to k parents
  - Especially if exploiting CPD structure, such as CSI
  - Cheaper per iteration
  - Finer moves within a graph

Scoring a tree 1: equivalent trees

\[
\ell(\theta_c, G; D) = M \sum_i \hat{I}(x_i, x_{\pi(i)}) - M \sum_i \hat{H}(x_i)
\]

- $X \rightarrow Z \rightarrow Y$
- $I(X, Z) + I(Z, Y)$

- $X \leftarrow Z \rightarrow Y$
- $I(Z, X) + I(Z, Y)$

- $X \leftarrow Z \leftarrow Y$
- $\ldots$
Scoring a tree 2: similar trees

\[ L(\theta_G, G; D) = M \sum_i \hat{I}(x_i, x_{(G)}) - M \sum_i \hat{I}(x_i) \]

Identifiability

- DAGs are I-equivalent if they encode the same set of conditional independencies
  - e.g., \( X \rightarrow Y \rightarrow Z \) and \( X \leftarrow Y \leftarrow Z \) are indistinguishable given just observational data.

- However, \( X \rightarrow Y \leftarrow Z \) has a v-structure, which has a unique statistical signature. Hence some arc directions can be inferred from passive observation.

- The set of I-equivalent DAGs can be represented by a PDAG (partially directed acyclic graph).

- Distinguishing between members of an equivalence class requires interventions/ experiments.
**ML score overfits!**

\[ \ell(\theta, G; D) = \log p(D | \theta, G) = M \sum_i I(x_i, x_{i(G)}) - M \sum_i \hat{H}(x_i) \]

- Information never hurts

\[ I(X, X_F) = H(X) - H(X | X_F) \]

\[ H(X|A) \geq H(X|A|Y) \]

- Adding a parent always increases your score!

**Occam’s Razor**

\[ \rho(\theta_m|m) \]

True distribution

Simple model

Just right

Complicated model

All possible datasets

Eric Xing
Model selection

- Three hypotheses

- As we increase $\epsilon$, we increase the dependence of $Y$ on $X$
- $X \leftarrow Y$ and $X \rightarrow Y$ are I-equivalent (have the same likelihood)

- Suppose we use a uniform Dirichlet prior for each node in each graph, with equivalent pseudo-counts (K2-prior): $P(\theta_i | H_i) = \text{Dir}(\alpha_i, \alpha_i)$
- In $H_1$, the equivalent sample size for $X$ is 2, but in $H_2$ it is 4 (since two conditioning contexts). Hence the posterior probabilities are different.

- Under which $H$ the $P(H|D)$ is higher?
Bayesian model selection

- Why is $P(H_0|D)$ higher when then dependence on $X$ and $Y$ is weak (small)?
  - It is not because the prior $P(H)$ explicitly favors simpler models (although this is possible).
  - It because the evidence $P(D)=\int d\theta P(D|\theta)P(\theta)$ automatically penalizes complex models.

- "Occam’s razor" says “If two models are equally predictive, prefer the simpler one”.
  - This is an automatic consequence of using Bayesian model selection.
  - Maximum likelihood would always pick the most complex model, since it has more parameters, and hence can fit the training data better.

- Good test for a learning algorithm: feed it random noise, see if it “discovers” structure!

Global & Local Parameter Independence

- Global Parameter Independence
  For every DAG model:
  \[
p(\theta | G) = \prod_{i=1}^{M} p(\theta_i | G)
  \]

- Local Parameter Independence
  For every node:
  \[
p(\theta_i | G) = \prod_{j=1}^{q_i} p(\theta_{x_i^j|x_i^j} | G)
  \]

- The Bayesian score
  \[
  \log P(G|D) = \log P(G) + \log \int \log P(D|\theta)P(\theta | G)d\theta + C
  = \log P(G) + \sum_{i,j} \int p(x_i^j|x_i^j, \theta_i)P(\theta_i | G)d\theta_i + C
  = \log P(G) + C + \sum \text{score}(x_i, x_i^j)
  \]
Selection criteria

- BIC (Bayesian Information Criterion):
  \[
  \log P(D) \approx \log P(D | \hat{\theta}_{ML}) - \frac{N}{2} \log n
  \]
  - Quiz: How many boxes behind the tree?

- Other criteria:
  - AIC (Akaike Information Criterion):
  - Minimum description length

Consistency of BIC and Bayesian scores

- A scoring function is **consistent** if, for true model \( G^* \), as \( m \to \infty \), with probability 1
  - \( G^* \) maximizes the score
  - All structures **not I-equivalent** to \( G^* \) have strictly lower score

- **Theorem**: BIC score is consistent
- **Corollary**: the Bayesian score is consistent

- What about maximum likelihood score?
Choice of Priors

- For finite datasets, prior is important!
  - Prior over structure satisfying prior modularity
    
    $d = \mathbb{P}(\text{structure}) \propto \prod_{i=1}^{L} \mathcal{C}(\alpha)$

- What about prior over parameters, how do we represent it?
  - K2 prior: fix an $\alpha$, $P(\theta|\text{Pa}_x) = \text{Dirichlet}(\alpha, \ldots, \alpha)$
  - K2 is “inconsistent”

BDe prior

- Dirichlet parameters analogous to “fictitious samples”
- Pick a fictitious sample size $m'$
  - For each possible family, define a prior distribution $P(x_i, \text{Pa}_x)$
  - Represent with a BN
  - Usually independent (product of marginals)

- BDe prior (Bayesian Dirichlet likelihood equivalent):
  - Has “consistency property”