

Graphical Models for Inference and Decision Making

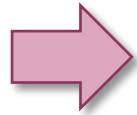
Unit 5: Learning Graphical Models from Observations

Instructor: Kathryn Blackmond Laskey
Spring 2019

Learning Objectives

- Describe the elements of a graphical model learning algorithm, and the main methods for each
- Given a Beta or Dirichlet prior distribution and a sample of cases, compute the posterior distribution for a local distribution for a Bayesian network
- Compute the relative posterior probability for two structures for a Bayesian network when the prior distribution is a Beta or Dirichlet mixture distribution
- Describe methods for learning graphical models with missing observations and latent/hidden variables

Unit 5 Outline



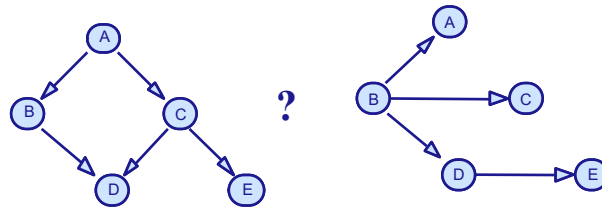
- Overview of Learning in Graphical Models
- Parameter Learning in Directed Graphical Models
- Structure Learning in Directed Graphical Models
- Learning in Undirected Graphical Models
- Statistical Relational Learning

Learning Graphical Models from Data

- A graphical model is defined by:
 - Structure
 - Parameters
- Structure
 - Nodes and states
 - Arcs
 - Functional form of local distributions (noisy-OR; context-specific independence; parameterized distributions; etc.)
- Parameters
 - Probabilities in a conditional probability table
 - Parameters of a parameterized local distribution
- The learning task can be decomposed into two parts:
 - Learning unknown parameters conditional on structure
 - Learning unknown structure
 - » This is usually taken to mean learning conditional dependence relationships with random variables taken as given

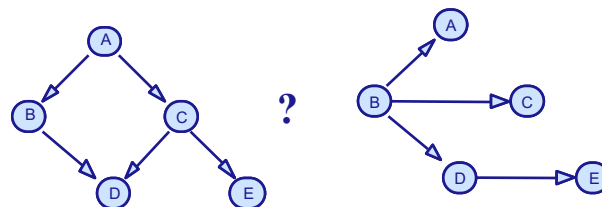
Raw Material: The Observations

- Easiest case:
 - Random sample (iid observations) of cases from the network to be learned
 - Each case contains an observed value for all variables in the network



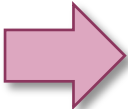
A	B	C	D	E
a0	b1	c1	d2	e0
a1	b1	c0	d1	e1
a1	b1	c1	d1	e1
a0	b0	c0	d0	e1
a0	b1	c0	d2	e0
a0	b1	c0	d1	e1
a1	b0	c0	d2	e1
a1	b0	c1	d2	e1
a1	b0	c1	d1	e1

- Complexities:
 - Missing observations: Some variables are not observed in some cases
 - Hidden or latent variables: Some variables are not observed in any cases
 - Non-random sampling: Sampled cases are not representative of the population for which the graphical model is being learned
 - Relational learning: the model to be learned has relational structure
 - Need to combine expert knowledge & data



A	B	C	D	E
a1	?	c1	d2	e1
a0	?	c1	d2	?
a1	?	c1	d1	e0
a0	?	c0	d0	e1
a0	?	c0	d1	e1
a1	?	c1	?	e1
a1	?	c1	d1	e1
a1	?	c1	d2	e0
a0	?	c0	d2	e0
a0	?	c0	d0	e1

Unit 5 Outline

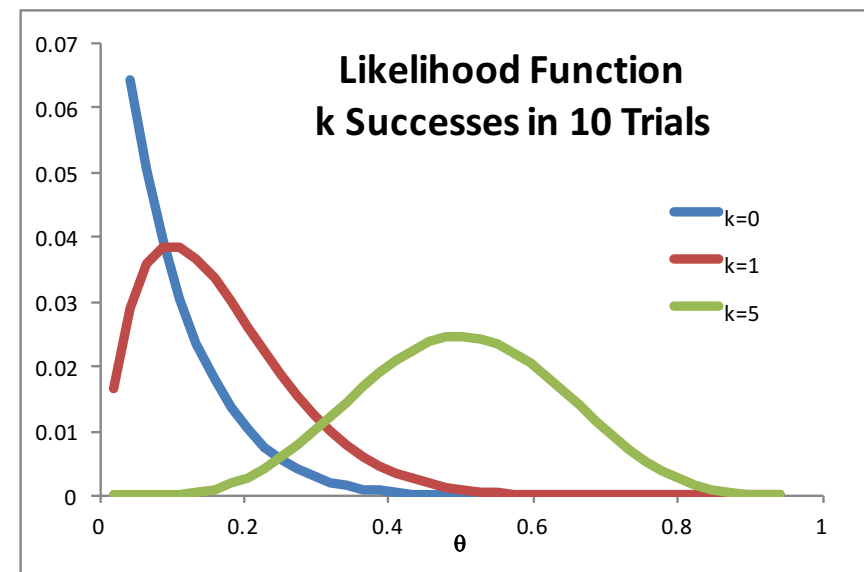
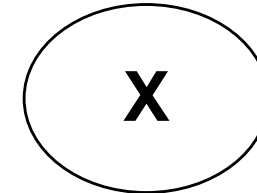
- Overview of Learning in Graphical Models
-  • Parameter Learning in Directed Graphical Models
- Structure Learning in Directed Graphical Models
- Learning in Undirected Graphical Models
- Statistical Relational Learning

Parameter Learning

- The distribution of a node X is a function of the parents of X and a set of parameters
 - For a CPT, the parameters are the entries in the CPT
 - We may be able to construct the CPT from fewer parameters, e.g.:
 - » Noisy-or: a probability for each trigger and a leak probability
 - » Partitions: a probability for each partition element
 - In general, a local distribution can be any parameterized function that maps parent configurations to probability distributions
- *A local distribution is a regression model*
 - A regression model expresses the probability distribution for a dependent variable as a function of independent variables and parameters
 - The local distribution for X is a regression model in which X is the dependent variable and $\text{pa}(X)$ are the independent variables
 - Sometimes we write $\Pr(X|\text{pa}(X),\theta)$ to emphasize that the local distribution depends on both the parents of X and the parameter θ
 - Parameter learning uses data to estimate θ
- Parameter learning can be recast as inference in a larger graphical probability model (data and parameters)

Parameter Learning: Starting Simple

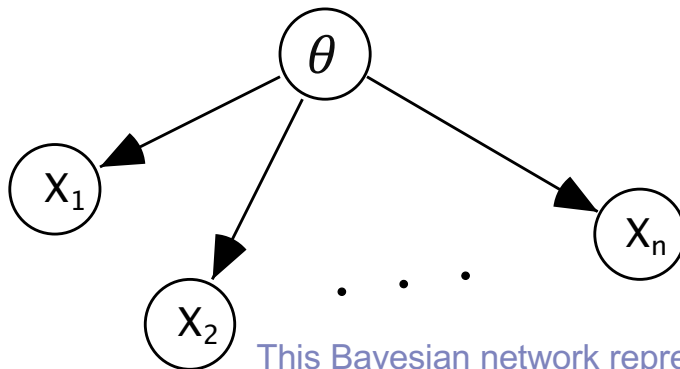
- Simplest case of learning:
 - Bayesian network with one random variable
 - Two states, $X=1$ or $X=0$
 - Unknown probability $\theta = P(X=1)$
 - Independent and identically distributed observations X_1, \dots, X_n
- A natural idea is to estimate θ by the frequency of 1's in our sample
 - If there are k 1's out of n observations, estimate is $\hat{\theta} = k / n$
 - This estimator has some nice properties
 - » Maximizes the *likelihood function* (i.e., we choose θ to make the data as probable as possible)
 - » Consistent (i.e., k/n converges to θ with probability 1 as $n \rightarrow \infty$)
 - » Other nice theoretical properties
- But what if there are no 1's in the sample? Do we want $\hat{\theta} = 0$?



The *likelihood function* is $P(\text{data} \mid \text{parameter})$
viewed as a function of the parameter

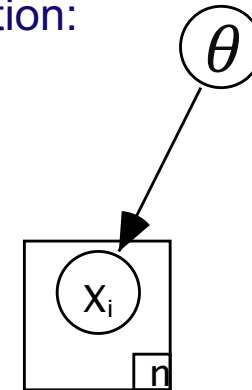
Learning a Parameter θ : Graphical Model Formulation

- A Bayesian network for $(\theta, X_1, \dots, X_n)$:



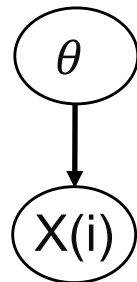
This Bayesian network represents conditional independence but does not represent that local distributions are same for all x_i

Plate notation:



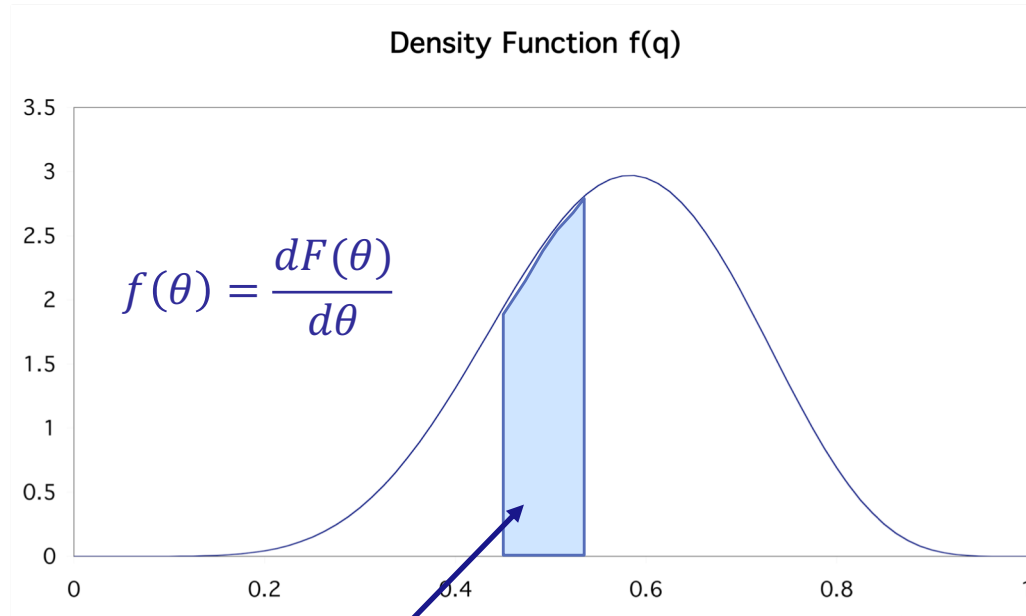
- "Plate" stands for n copies of X
- Plate notation is widely used in Bayesian machine learning

MFrag notation:



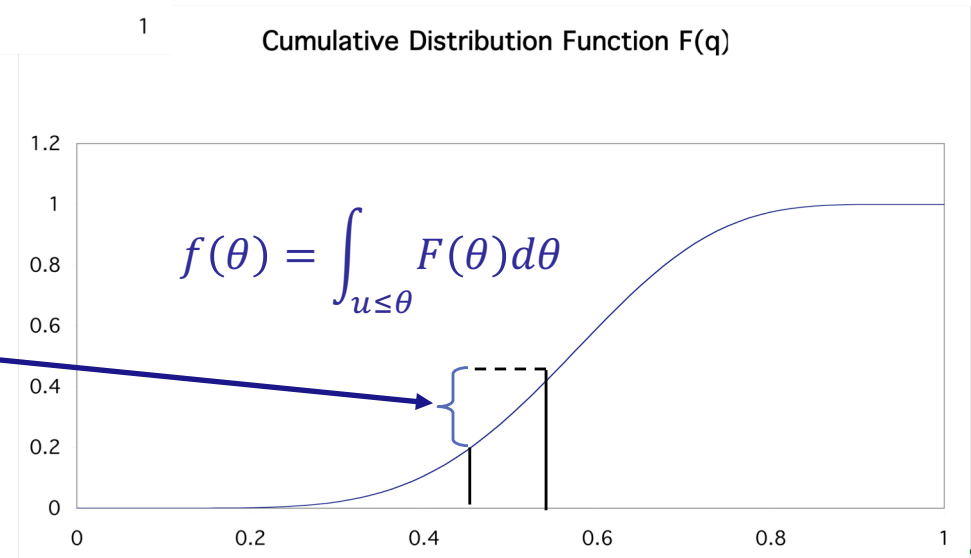
- Given θ , the X 's are independent and have the same distribution
- The model specifies a prior distribution for θ and one distribution $P(X_i | \theta)$ (same for all X_i)
- Note: θ is a continuous random variable taking on values in the unit interval
- Observations on the X 's can be used to estimate θ
 - Common approaches: maximum likelihood, maximum a posteriori, full Bayes

Density and Cumulative Distribution Functions: Review

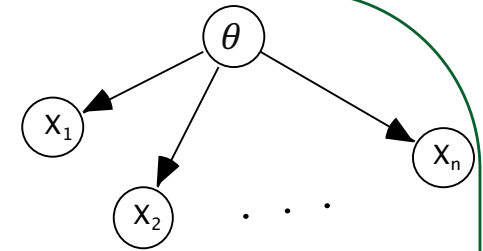


Each individual value of a continuous random variable has probability zero

Area under $f(\theta)$ between a and b is equal to difference $F(b) - F(a)$ and is probability that value of random variable lies between a and b



Beta Family of Distributions



- Convenient family of prior distributions for an unknown probability θ
 - Possible values are real numbers between 0 and 1
 - Closed under Binomial sampling: posterior distribution for θ is also Beta
 - Uniform distribution (all probabilities equally likely) is a member of the Beta family
- The density function for the Beta distribution with integer parameters α and β is:

$$f(\theta|\alpha, \beta) = \frac{(\alpha+\beta-1)!}{(\alpha-1)!(\beta-1)!} \theta^{\alpha-1} (1-\theta)^{\beta-1} \quad \text{for } 0 < \theta < 1 \quad (1a)$$

- The density function for the Beta distribution with arbitrary parameters $\alpha > 0$ and $\beta > 0$ is:

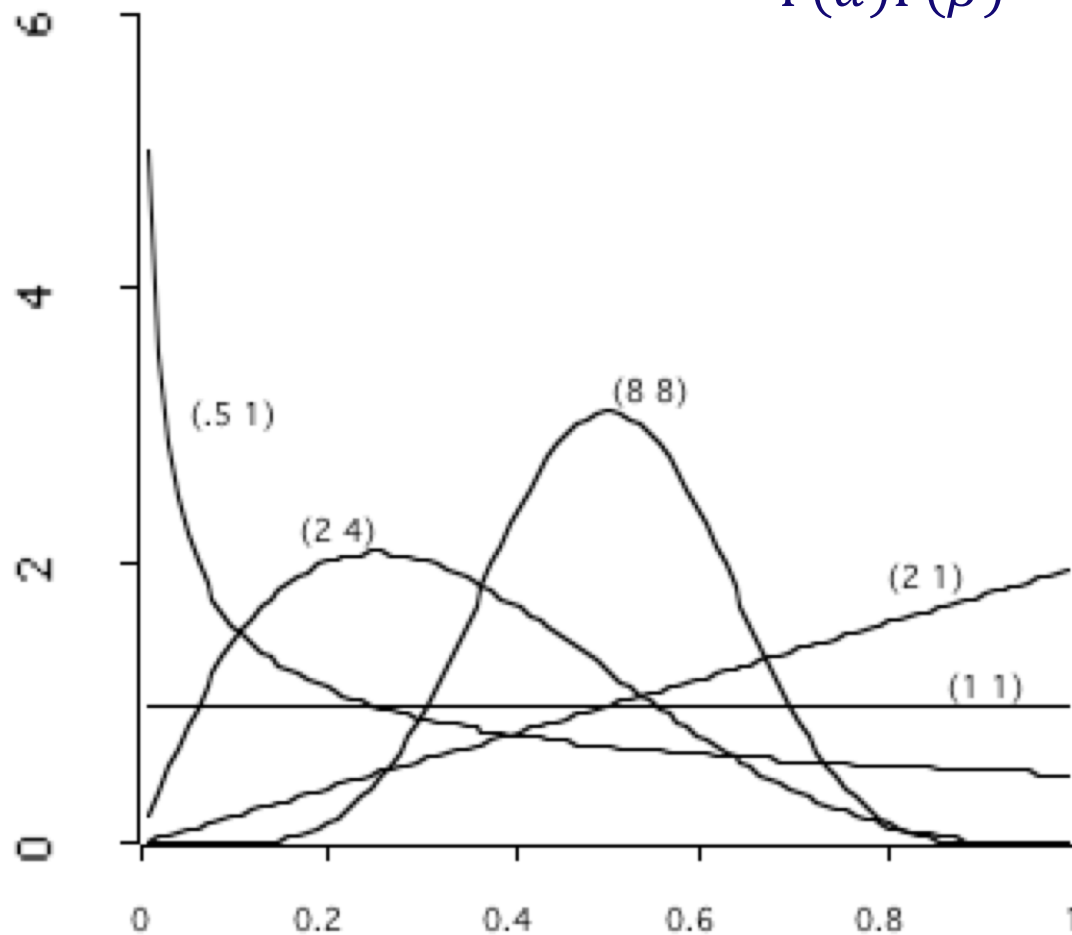
$$f(\theta|\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} \quad \text{for } 0 < \theta < 1 \quad (1b)$$

$$\Gamma(\alpha) = \int_0^{\infty} x^{\alpha-1} e^{-x} dx$$

$$\Gamma(\alpha) = (\alpha-1)! \quad \text{for integer } \alpha$$

Density Functions for Various Beta Distributions

$$f(\theta|\alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1 - \theta)^{\beta-1} \quad \text{for } 0 < \theta < 1$$



$\Gamma(\alpha)$ is the Gamma function

$$\Gamma(\alpha) = \int_0^{\infty} x^{\alpha-1} e^{-x} dx$$

$$\Gamma(\alpha) = (\alpha - 1)! \quad \text{for integer } \alpha$$

$$E[\theta|\alpha, \beta] = \frac{\alpha}{\alpha + \beta}$$

$$V[\theta|\alpha, \beta] = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}$$

Prior to Posterior Distribution

- The prior density function (for integer α, β):

$$f(\theta|\alpha, \beta) = \frac{(\alpha+\beta-1)!}{(\alpha-1)!(\beta-1)!} \theta^{\alpha-1} (1-\theta)^{\beta-1} \quad (1a)$$

- Data likelihood for k 1's and $n-k$ 0's:

$$P((X_1, \dots, X_n) = (x_1, \dots, x_n) | \theta) = \theta^k (1-\theta)^{n-k} \quad (\text{Bernoulli distribution})$$

- Prior times likelihood:

$$P((X_1, \dots, X_n) = (x_1, \dots, x_n) | \theta) f(\theta|\alpha, \beta) = \frac{(\alpha+\beta-1)!}{(\alpha-1)!(\beta-1)!} \theta^{\alpha+k-1} (1-\theta)^{\beta+n-k-1} \quad (2)$$

Note: (2) is proportional to the density function of the Beta distribution with parameters $\alpha+k$ and $\beta+n-k$.

- Applying Bayes Rule:

$$\begin{aligned} P(\theta | (X_1, \dots, X_n) = (x_1, \dots, x_n)) &= \frac{P((X_1, \dots, X_n) = (x_1, \dots, x_n) | \theta) f(\theta|\alpha, \beta)}{\int_{\theta=0}^1 P((X_1, \dots, X_n) = (x_1, \dots, x_n) | \theta) f(\theta|\alpha, \beta) d\theta} \\ &= \frac{\theta^{\alpha+k-1} (1-\theta)^{\beta+n-k-1}}{\int_{\theta=0}^1 \theta^{\alpha+k-1} (1-\theta)^{\beta+n-k-1} d\theta} = \frac{(\alpha+\beta+n-1)!}{(\alpha+k-1)! (\beta+n-k-1)!} \theta^{\alpha+k-1} (1-\theta)^{\beta+n-k-1} \quad (3) \end{aligned}$$

Note: (3) is the density function of the Beta distribution with parameters $\alpha+k$ and $\beta+n-k$.

Beta – Bernoulli Conjugate Pair

- IF prior density function is Beta(α, β):

$$f(\theta|\alpha, \beta) = \frac{(\alpha+\beta-1)!}{(\alpha-1)!(\beta-1)!} \theta^{\alpha-1} (1-\theta)^{\beta-1} \quad (1a)$$

- AND likelihood function is Bernoulli(θ):

$$P((X_1, \dots, X_n) = (x_1, \dots, x_n) | \theta) = \theta^k (1-\theta)^{n-k}$$

- THEN posterior density function is Beta($\alpha + k, \beta + n - k$):

$$f(\theta|\alpha + k, \beta + n - k) = \frac{(\alpha+\beta+n-1)!}{(\alpha+k-1)!(\beta+n-k-1)!} \theta^{\alpha+k-1} (1-\theta)^{\beta+n-k-1} \quad (3)$$

The Beta family of prior distributions and the Bernoulli family of likelihoods are a conjugate pair

Conjugate Pairs of Distributions

- A conjugate pair is a pair $g(\theta|\alpha)$ / $f(x|\theta)$ of prior/likelihood families that is closed under sampling:
 - IF Observations X_1, \dots, X_n are a random sample from $f(x|\theta)$ and prior distribution for θ is $g(\theta|\alpha)$
 - THEN Posterior distribution for θ is $g(\theta|\alpha^*)$, another member of the conjugate family
- Example: The Beta and Binomial families of distributions are a conjugate pair:
 - IF Observations X_1, \dots, X_n are a random sample from $\text{Binomial}(\theta)$ and prior distribution for θ is $\text{Beta}(\alpha, \beta)$
 - THEN Posterior distribution for θ is $\text{Beta}(\alpha+k, \beta+n-k)$, another member of the conjugate family
- Conjugate pairs simplify Bayesian inference
 - Posterior distribution can be found exactly
 - There is a simple updating rule to find the parameters of the posterior distribution from parameters of the prior distribution and data summaries

Beta Conjugate Updating

- Inference from prior to posterior
 - Prior distribution is Beta (α, β) with expected value $E[\theta] = \alpha/(\alpha+\beta)$
 - Data: n observations with k 1's
 - Posterior distribution is Beta ($\alpha+k, \beta+n-k$) with expected value

$$E[\theta | X] = (\alpha+k)/(\alpha+\beta+n)$$
- Interpretation of Beta prior distribution:
 - Prior information is “like” a previous sample of $\alpha+\beta$ observations with α 1's
 - After n observations with k 1's, posterior information is “like” a previous sample of $\alpha+\beta+n$ observations with $\alpha+k$ 1's.
 - The “hyperparameters” α and β are called “virtual counts” or “pseudo counts”
- Precision of estimate increases as sample size gets larger

- Variance of the prior distribution

$$V[\theta] = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} = \frac{E[\theta](1-E[\theta])}{(\alpha+\beta+1)}$$

- Variance of the posterior distribution

$$V[\theta | X] = \frac{(\alpha+k)(\beta+n-k)}{(\alpha+\beta+n)^2(\alpha+\beta+n+1)} = \frac{E[\theta | X](1-E[\theta | X])}{(\alpha+\beta+n+1)}$$

Uniform Prior Distribution

- Uniform distribution is $\text{Beta}(1,1)$
- If prior distribution for θ is uniform then posterior distribution for θ is $\text{Beta}(k+1, n-k+1)$
- Posterior expected value of θ is

$$E[\theta | X] = (k+1)/(n+2)$$

(This formula is called *Laplace's rule of succession*)

- Posterior variance of θ is

$$V[\theta | X] = \frac{(k+1)(n-k+1)}{(n+2)^2(n+3)}$$

- A posterior credible interval can be obtained from quantiles of the Beta distribution
 - Lower bound of 90% interval is 0.05 quantile of the Beta distribution, $\text{qbeta}(0.05, k+1, n-k+1)$ in R
 - Upper bound of 90% interval is 0.95 quantile of the Beta distribution, $\text{qbeta}(0.95, k+1, n-k+1)$ in R

Learning Unknown Probabilities in a Binary Bayesian Network

- Parameter learning can be represented as Bayesian network
- Assumptions:
 - Local independence: $\theta_{B|a1}$ is independent of $\theta_{B|a0}$
 - Global independence: $\theta_{B|a1}$ and $\theta_{B|a0}$ are independent of θ_A
 - All parameters $\theta_{B|a1}$, $\theta_{B|a0}$, and θ_A , have Beta distributions
 - We will assume $\alpha=1$ and $\beta=1$ (uniform prior distribution) for all θ 's
- Data: the number of observations in each category:

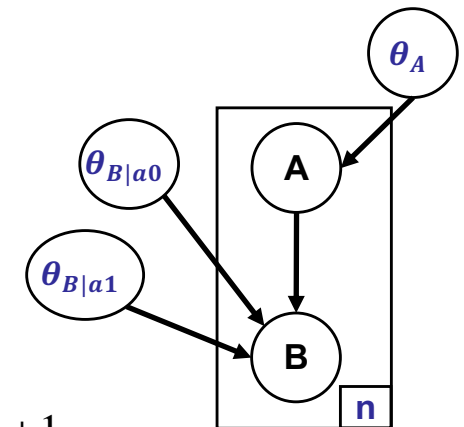
a1,b1	n_{11}
a1,b0	n_{10}
a0,b1	n_{01}
a0,b0	n_{00}

- Posterior distributions for Beta (1,1) (uniform) prior:

$$\theta_A \sim \text{Beta}(n_{11} + n_{10} + 1, n_{01} + n_{00} + 1) \quad E[\theta_A | \text{data}] = \frac{n_{11} + n_{10} + 1}{n_{11} + n_{10} + n_{01} + n_{00} + 2}$$

$$\theta_{B|a1} \sim \text{Beta}(n_{11} + 1, n_{10} + 1) \quad E[\theta_{B|a1} | \text{data}] = \frac{n_{11} + 1}{n_{11} + n_{10} + 2}$$

$$\theta_{B|a0} \sim \text{Beta}(n_{01} + 1, n_{00} + 1) \quad E[\theta_{B|a0} | \text{data}] = \frac{n_{01} + 1}{n_{01} + n_{00} + 2}$$



Numerical Example

- Beta(1,1) (uniform) prior distribution for θ_A , $\theta_{B|a1}$ and $\theta_{B|a0}$
- The data:

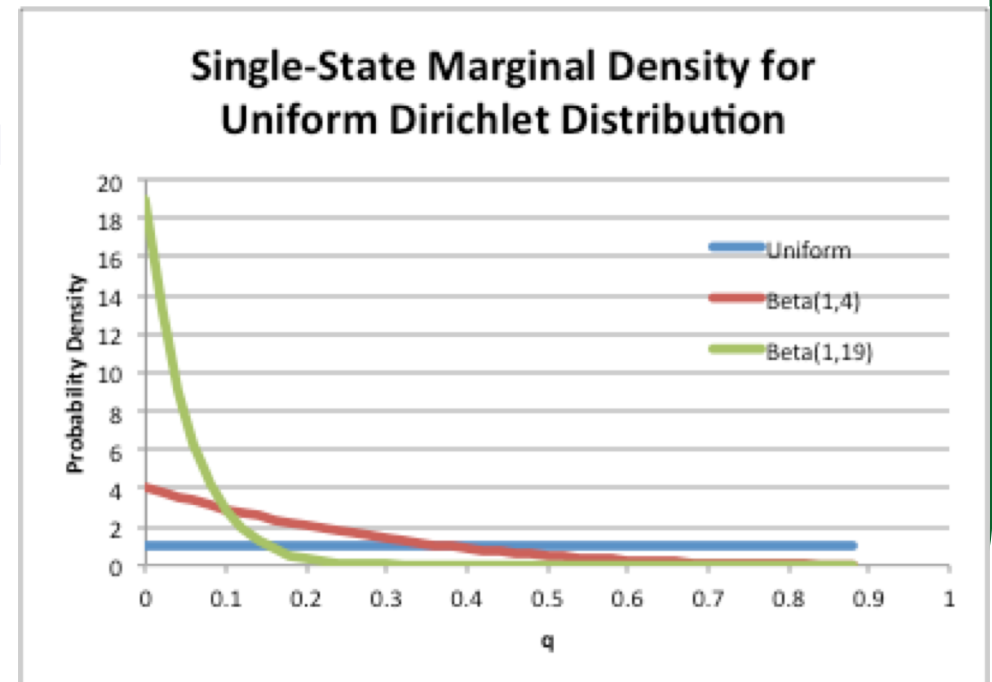
a1,b1	10	n_{11}
a1,b0	6	n_{10}
a0,b1	2	n_{01}
a0,b0	7	n_{00}
- Posterior distribution for θ_A
 - Beta distribution with parameters $17 = 1+10+6$ and $10 = 1+2+7$
 - Mean: $17/(17+10) = 0.63$
 - Standard deviation: $((0.63)(0.37)/(28))^{1/2} = 0.09$
- Posterior distribution for $\theta_{B|a1}$
 - Beta distribution with parameters $11 = 1+10$ and $7 = 1+6$
 - Mean: $11/(11+7) = 0.61$
 - Standard deviation: $((0.61)(0.39)/(19))^{1/2} = 0.11$
- Posterior distribution for $\theta_{B|a0}$
 - Beta distribution with parameters $3 = 1+2$ and $8 = 1+7$
 - Mean: $3/(3+8) = 0.27$
 - Standard deviation: $((0.27)(0.73)/(12))^{1/2} = 0.13$

Nodes with More than 2 States

- For nodes with more than 2 states we can use a Dirichlet prior distribution for the local distributions
- Dirichlet distribution is a multivariate generalization of the Beta distribution
 - If a p -dimensional random variable $(\theta_1, \dots, \theta_p)$ has a Dirichlet($\alpha_1, \dots, \alpha_p$) distribution then:
 - » Only values $0 < \theta_i < 1$ and $\sum_i \theta_i = 1$ have positive probability density
 - »
$$E[\theta_i] = \frac{\alpha_i}{\alpha_1 + \dots + \alpha_p} \quad V[\theta_i] = \frac{E[\theta_i](1 - E[\theta_i])}{\alpha_1 + \dots + \alpha_p + 1}$$
 - Dirichlet(1,...,1) puts equal density on all probability distributions (uniform distribution)
 - If θ has a Beta(α, β) distribution then $(\theta, 1 - \theta)$ has a Dirichlet(α, β) distribution
 - If $(\theta_1, \dots, \theta_p)$ has a Dirichlet($\alpha_1, \dots, \alpha_p$) distribution then θ_1 has a Beta($\alpha_1, \alpha_2 + \dots + \alpha_p$) distribution
- If the prior distribution is Dirichlet($\alpha_1, \dots, \alpha_p$) and n_i observations are observed in each state, then:
 - Posterior distribution is Dirichlet($n_1 + \alpha_1, \dots, n_p + \alpha_p$)
 - $$E[\theta_i | \text{data}] = \frac{n_i + \alpha_i}{n_1 + \dots + n_p + \alpha_1 + \dots + \alpha_p}$$

Uniform Prior Distribution with Many States

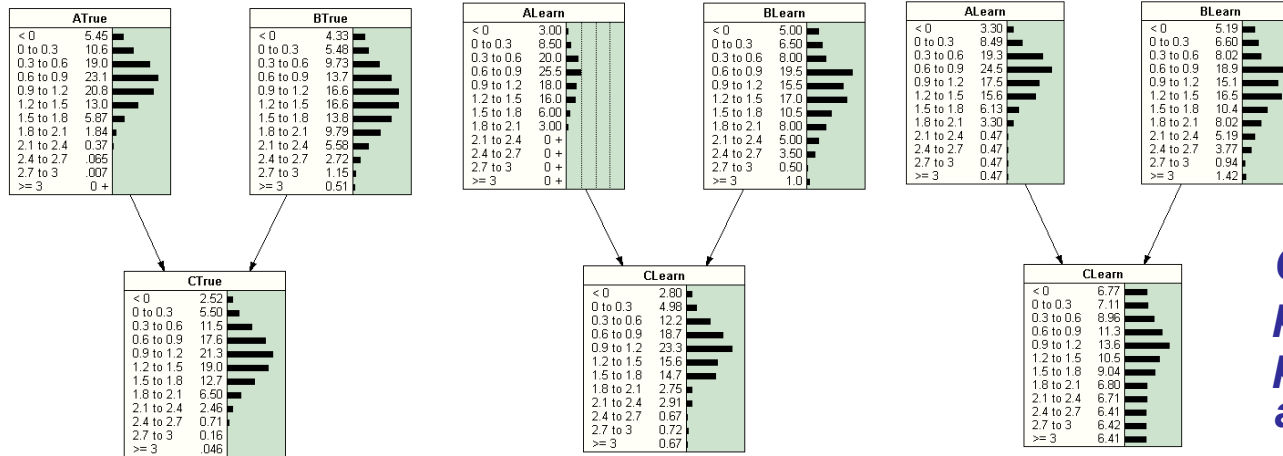
- Uniform prior distribution Dirichlet(1, ..., 1) corresponds to assigning one “virtual observation” per state
 - A uniform distribution on k probabilities is not a uniform distribution on each probability if $k > 2$!*
 - If there are k states each state has a Beta(1, $k-1$) distribution with expected value $1/k$
- Accurate estimation from data requires the sample size to be large relative to the number of “virtual observations”
- When there are many states and few observations the uniform prior distribution results in a posterior distribution that is very “flat.”
- This can happen even with large sample sizes when a node has many parents



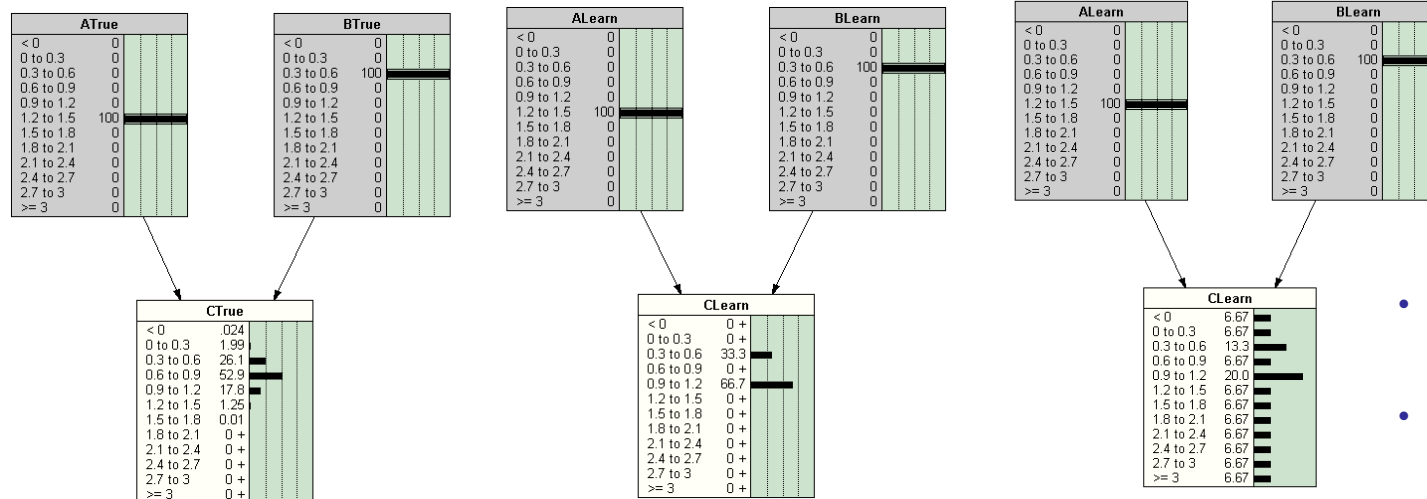
Effect of Many States: Example

- Consider a distribution for a node A which has p states
 - Observations: 40 cases in state a_1 ; 10 cases in state a_2
 - Uniform prior distribution
- Case 1: Node A has 2 states: a_1 and a_2
 - Posterior distribution for θ_1 is Beta(41, 11)
 - Posterior Mean for θ_1 is 0.79
 - Posterior 90% interval for θ_1 is (0.69, 0.87)
- Case 2: Node A has 20 states: a_1, a_2, \dots, a_{20}
 - Posterior distribution for $(\theta_1, \theta_2, \dots, \theta_{20})$ is Dirichlet (41, 11, 1, ..., 1)
 - Posterior distribution for θ_1 is Beta(41, 29)
 - Posterior Mean for θ_1 is 0.59
 - Posterior 90% interval is (0.49, 0.68)
- Uniform prior for node with p states assigns p virtual observations uniformly (one to each state)
 - This can give problematic results when the number of virtual observations is large relative to the number of actual observations

Curse of Dimensionality



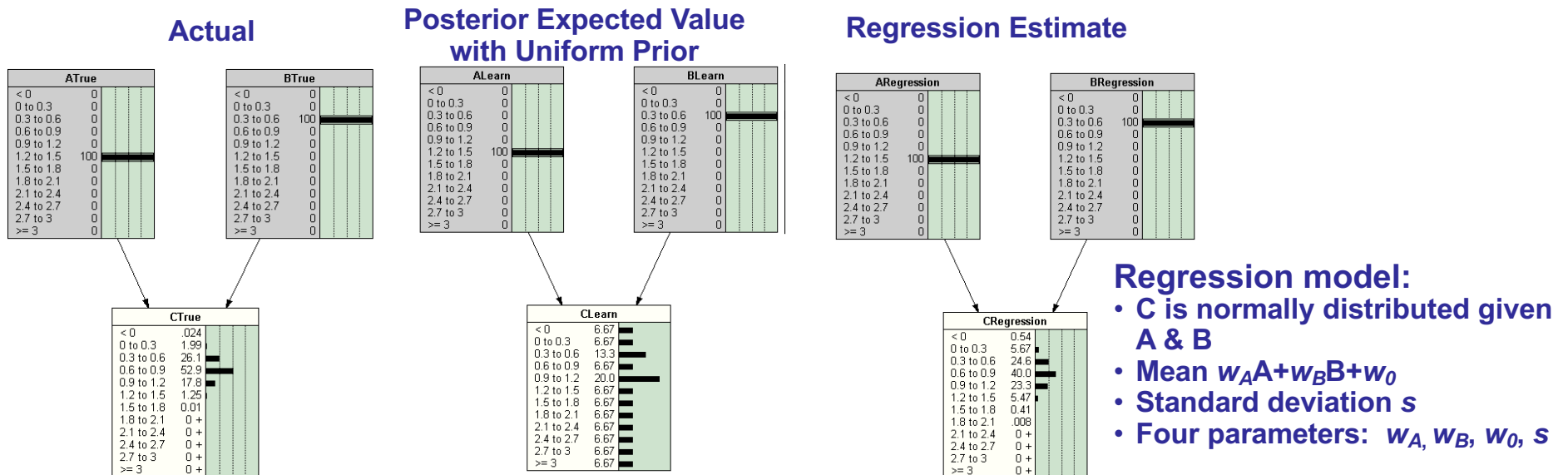
Comparison of actual probabilities with probabilities learned from a sample of size 200



- There are no observations for many combinations of A and B
- Many distributions are based on very small sample sizes

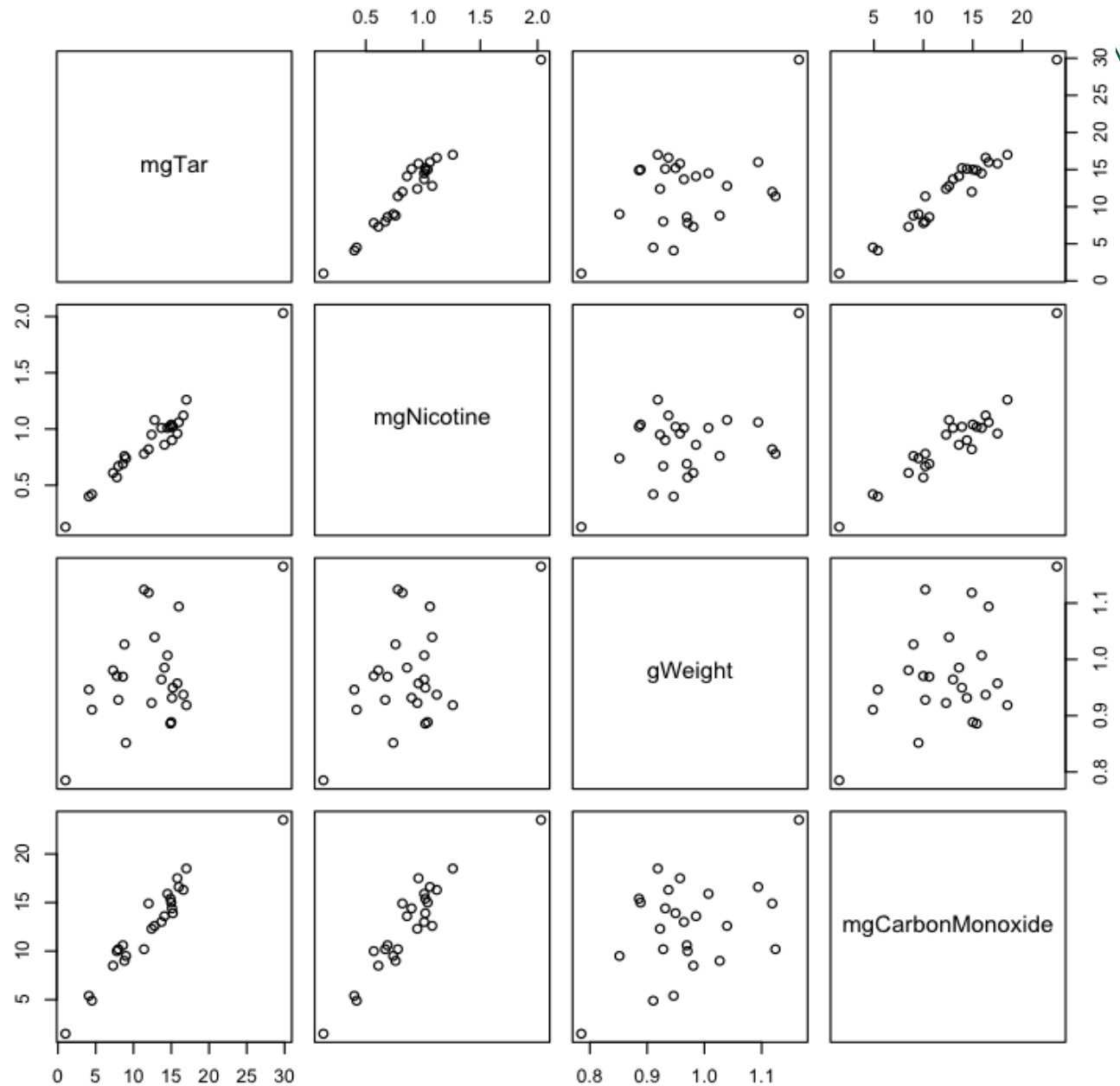
Modeling Local Distributions

- Parameterized models can increase efficiency of learning
 - Context-specific independence
 - » Same child distribution for different combinations of parents
 - » This is a special type of parameterized distribution
 - General parameterized distribution
 - » e.g., child node has normal distribution with parameters depending on parent



If we can model the relationship well with a parameterized model, regression dramatically out-performs unconstrained Dirichlet learning

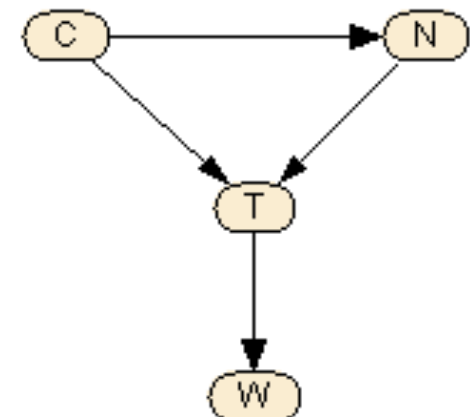
Example: Cigarettes



Cigarette data set <http://www.mste.uiuc.edu/regression/cig.html> consists of measurements of weight, tar, nicotine, and CO content of 25 cigarette brands

Graphical Model for Cigarette Data

- Assume a graphical model in which all RVs have normal distribution with mean a linear function of parents:
 - $C \sim N(\beta_C, \sigma_C)$
 - $N \sim N(\beta_{0N} + \beta_{1N}C, \sigma_N)$
 - $T \sim N(\beta_{0T} + \beta_{1T}C + \beta_{2T}N, \sigma_T)$
 - $W \sim N(\beta_{0W} + \beta_{1W}T, \sigma_W)$
 (statistical tests indicate W conditionally independent of C & N given T)
- Parameters of this model can be estimated by estimating a linear regression model using statistical software
 - Maximum likelihood
 - Maximum a posteriori
 - Fully Bayesian



Representing Regression Model in a BN Package

- To represent this model in Netica:

- Enter equations using scripting language

$$P(C |) = \text{NormalDist}(C, 12.5, 4.74)$$

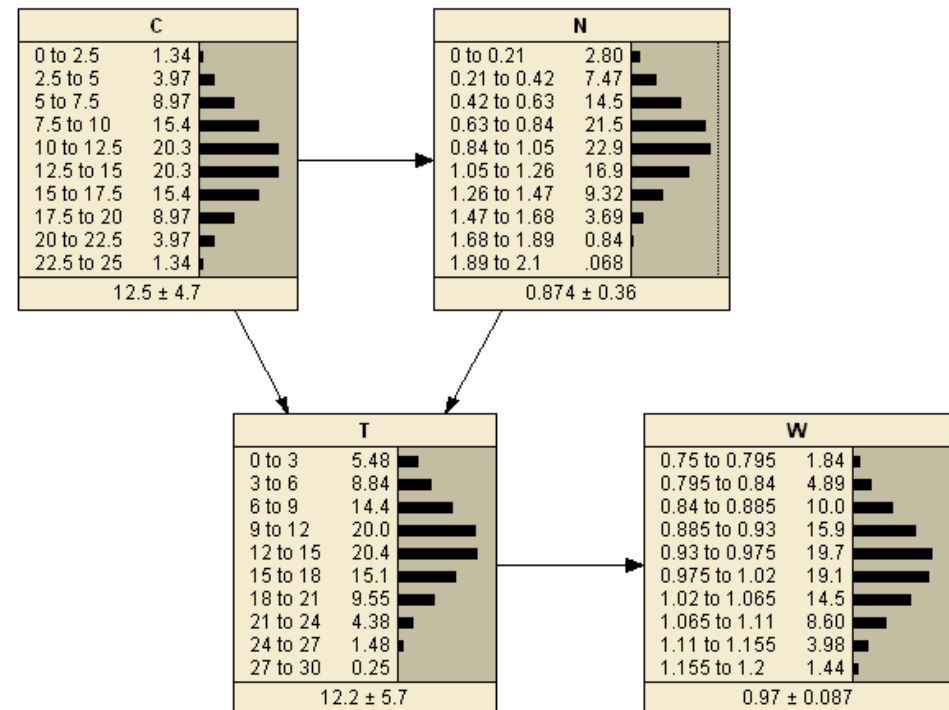
$$P(N | C) = \text{NormalDist}(N, 0.01 + 0.069 * C, 0.137)$$

$$P(T | N, C) = \text{NormalDist}(T, -2.22 + 10.1 * N + 0.446 * C, 0.96)$$

$$P(W | T) = \text{NormalDist}(W, 0.877 + 0.0076 * T, 0.078)$$

- Discretize all nodes

- Convert all equations to tables



Parameter Estimation in High Dimensions

- Parameter learning in graphical models is a problem of statistical estimation in high-dimensional parameter spaces
- Statistical estimates perform poorly when the number of parameters being estimated is large relative to the number of observations
- When expert knowledge is available it can be used to
 - Specify prior distributions for parameters
 - Find models that capture essential aspects of the problem and have fewer parameters
- Both these uses of expert knowledge involve application of expert judgment
- Good modeling practice includes many practical tools for coping with the “large k small n ” problem
- Bayesian analysis is a formal, theory-based approach to combining expert judgment with empirical observations
- Most methods that work well in practice can be given a Bayesian justification

Parameterized Models for Local Distributions

- Context-specific independence:
 - Probabilities are identical for a subset of configurations of the parent variables
 - Example: Sensor A and B have the same detection probability in the daytime in clear weather
 - We can group the cases for Sensor A and Sensor B under these conditions and estimate a single probability
 - This is called “parameter tying” (parameters for grouped cases are “tied” to each other)
- Independence of causal influence
 - ICI models have fewer parameters than general local distribution
 - There is no closed-form solution for parameter learning with common ICI models such as noisy-OR
 - Can be handled with methods for hidden variables (auxiliary variables are treated as hidden)
- Numeric and continuous random variables
 - A RV’s distribution given its parents can be specified using any parameterized statistical model
 - Parameters can be estimated by regression methods
 - Conjugate priors are often used for continuous distributions to simplify computation

Conjugate Families and Virtual Sufficient Statistics

- Conjugate families of distributions are very useful in Bayesian modeling
- Many commonly applied distributions have conjugate families
 - We have already discussed the Beta/Bernoulli (or Beta/Binomial) conjugate pair and its generalization to the Dirichlet/Multinomial conjugate pair
 - The Gamma distribution is a conjugate prior for the rate parameter for Poisson observations (or exponential observations parameterized by rate)
 - The Normal distribution is a conjugate prior for mean of a normal distribution with known covariance
 - The Normal / inverse Gamma distribution is conjugate prior for the mean and precision (inverse covariance) of a normal distribution with unknown mean and covariance
- Any conjugate prior distribution has an interpretation as a summary of “virtual prior samples”
- Parameter learning with conjugate priors can be viewed as updating “virtual sufficient statistics”
 - A function of the observations is a *sufficient statistic* if it captures all the information needed to obtain the posterior distribution

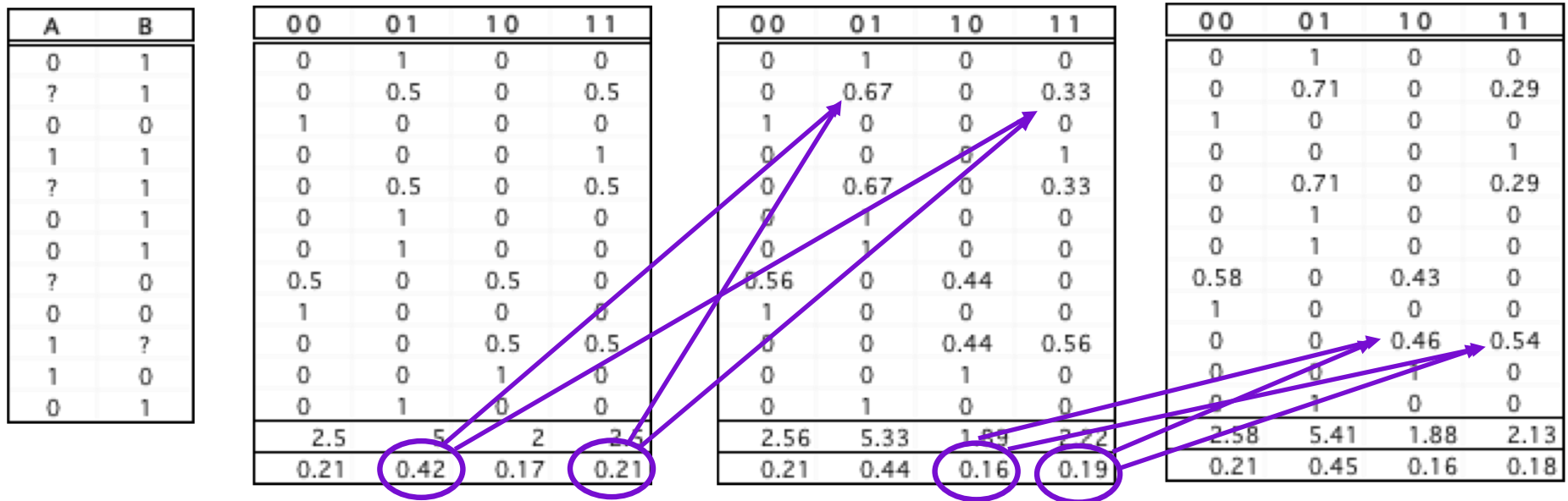
Missing Observations

- Many standard learning algorithms discard cases with missing data
- This can seriously degrade learning algorithms
- There are statistically justified methods for learning with missing data
- The most common are imputation and EM algorithm
 - Imputation inserts educated guess for missing observation
 - » Random or deterministic
 - » Heuristic or theory-based
 - » Single pass or iterative
 - EM algorithm is an iterative method for maximum likelihood or maximum a posteriori estimation in presence of missing data

EM Algorithm

- General method for statistical parameter estimation in the presence of missing data
- The method consists of two steps:
 - E-step: Given the current estimate of the parameter, compute the expected value of the “missing data sufficient statistics”
 - » In the case of learning a belief table this means “filling in” missing values with probabilities
 - M-step: Given the current estimate of the “missing data sufficient statistics” compute the maximum a posteriori (or maximum likelihood) estimate of the parameter
 - » In the case of learning a belief table this means estimating local probabilities from counts
- Under regularity conditions (data are “missing at random” and the model is exponential family) the EM algorithm converges to a local maximum of the posterior distribution
 - Missing at random (MAR): no systematic relationship between the propensity to be missing and the value of the missing data
 - Example: missing responses to “Did you cheat on your taxes?” are probably not MAR
 - Note: if uniform prior, then MAP is same as MLE

Example of EM Algorithm



A	B
0	1
?	1
0	0
1	1
?	1
0	1
0	1
?	0
0	0
1	?
1	0
0	1

	00	01	10	11
0	1	0	0	0
0	0.5	0	0.5	0
1	0	0	0	0
0	0	0	0	1
0	0.5	0	0.5	0
0	1	0	0	0
0	1	0	0	0
0.5	0	0.5	0	0
1	0	0	0	0
0	0	0.5	0.5	0
0	0	1	0	0
0	1	0	0	0
2.5	5	2	2.5	
0.21	0.42	0.17	0.21	

	00	01	10	11
0	1	0	0	0
0	0.67	0	0.33	0
1	0	0	0	0
0	0	0	0	1
0	0.67	0	0.33	0
0	1	0	0	0
0	1	0	0	0
0.56	0	0.44	0	0
1	0	0	0	0
0	0	0.44	0.56	0
0	0	1	0	0
0	1	0	0	0
2.56	5.33	1.89	2.22	
0.21	0.44	0.16	0.19	

	00	01	10	11
0	1	0	0	0
0	0.71	0	0.29	0
1	0	0	0	0
0	0	0	0	1
0	0.71	0	0.29	0
0	1	0	0	0
0	1	0	0	0
0.58	0	0.43	0	0
1	0	0	0	0
0	0	0.46	0.54	0
0	0	1	0	0
0	1	0	0	0
2.58	5.41	1.88	2.13	
0.21	0.45	0.16	0.18	

The Data

Iteration 1

Iteration 2

Iteration 3

- Detailed steps:

- Estimate all missing observations

- » Iteration 1: use uniform probabilities, possible values are all 0.5

- » Iteration 2, Rows 2& 5: $.42/(.42+.21)$ and $.21/(.42+.21)$

- » Iteration 3, Row 10: $.16/(.16+.19)$ and $.19/(.16+.19)$

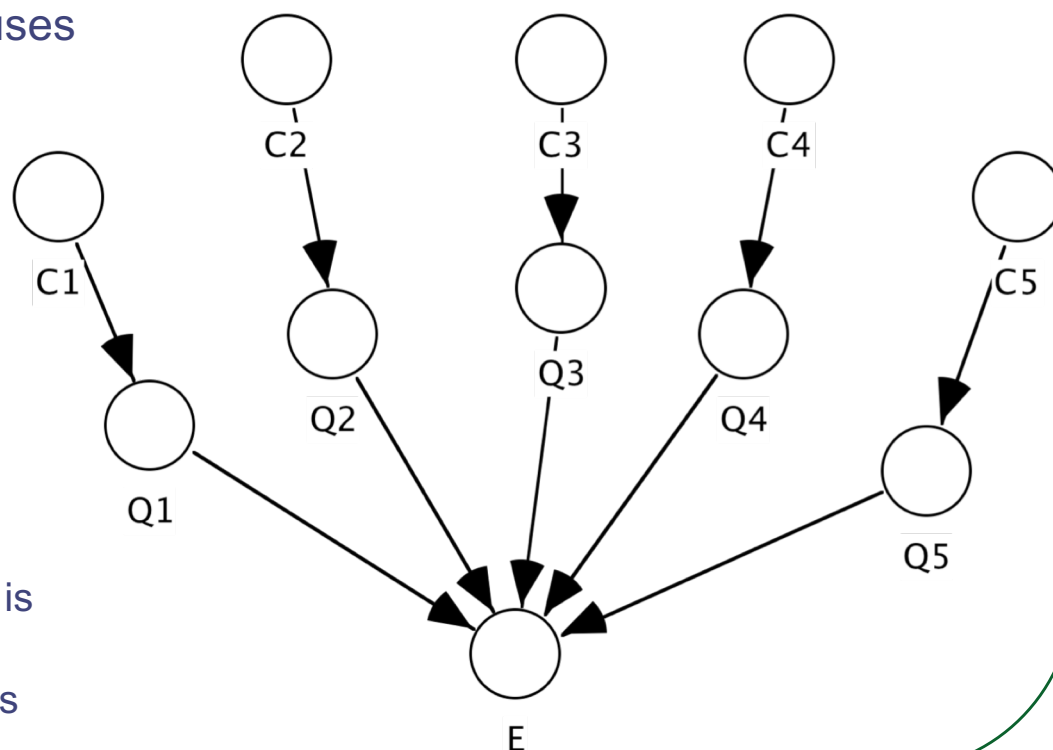
- Compute column sums

- Normalize to obtain probabilities for 00, 01, 10, 11

Learning with Independence of Causal Influence

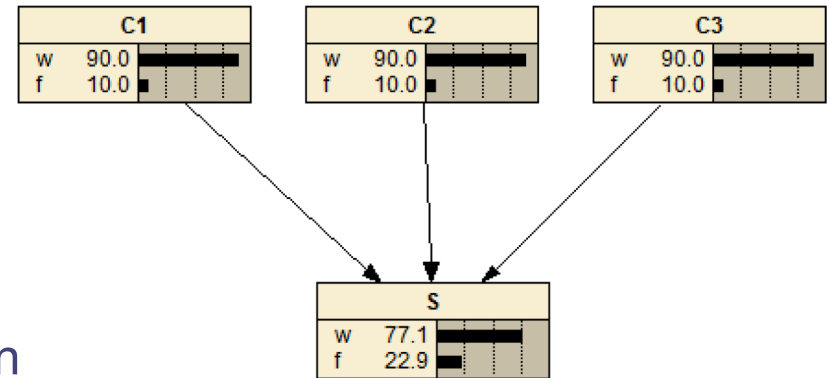
- Example: noisy-OR
 - Represent as "trigger variable" model
 - Estimate $P(\theta_i \mid C_i = \text{TRUE})$
 - All other probabilities are given
 - » $P(\theta_i \mid C_i = \text{FALSE}) = 0$
 - » $P(E \mid C_1, \dots, C_n) = 0$ if all causes are false
 - » $P(E \mid C_1, \dots, C_n) = 1$ if any cause is true
- C's and E are observed but Q's are not
 - No closed form solution
 - EM algorithm can be used to estimate parameters
 - » Local distribution for E given Q's is specified by the model
 - » Local distribution for Q_i given C_i is estimated via EM

- Other ICI models can be handled in a similar way
 - Represent as "trigger variable" model
 - Apply EM to learn parameters



Example: Learning Noisy-Or with EM

- Data: 2000 samples from Noisy-Or model
- Comparison:
 - Model 1: standard parameter learning, unrestricted model
 - Model 2: trigger variable representation of noisy-or model, learned by EM, then trigger variables absorbed



Note: 2000 observations contain only 3 cases with $C1 = C2 = C3 = f$

C1	C2	C3	w	f
w	w	w	99	1
w	w	f	19.8	80.2
w	f	w	29.7	70.3
w	f	f	5.94	94.06
f	w	w	9.9	90.1
f	w	f	1.98	98.02
f	f	w	2.97	97.03
f	f	f	0.594	99.406

True noisy or distribution

C1	C2	C3	w	f
w	w	w	99.028	0.972
w	w	f	20.112	79.888
w	f	w	27.907	72.093
w	f	f	10.526	89.474
f	w	w	11.25	88.75
f	w	f	5.263	94.737
f	f	w	9.091	90.909
f	f	f	20	80

Learned via unrestricted parameter learning

C1	C2	C3	w	f
w	w	w	99.096	0.904
w	w	f	19.628	80.372
w	f	w	27.841	72.159
w	f	f	5.514	94.486
f	w	w	10.77	89.23
f	w	f	2.133	97.867
f	f	w	3.026	96.974
f	f	f	0.599	99.401

Learned by EM

How EM Works for Bayesian Dirichlet Parameter Learning in Bayesian Networks

1. Initialize table of expected data for each clique in the junction tree
 - Rows are cases; columns are configurations of clique variables
 - For each case, assign probability 0 to configurations inconsistent with observations; uniform probabilities for other configurations
2. Sum over cases and normalize to find expected clique marginal distributions
3. Estimate local distributions $P(X|pa(X))$ from clique marginal distributions
4. Compute new clique tables in JT using new estimates of $P(X|pa(X))$
5. For each case with missing data, update expected data tables for each clique in JT:
 - Insert observed data and use JT algorithm to propagate evidence
 - Use clique marginal distribution given observed data to assign probabilities to configurations consistent with the data (this will assign 0 probability to configurations inconsistent with data)
6. If change since last iteration is greater than tolerance, go to Step 2, else stop

Parameter Learning – It Can Get Complicated!

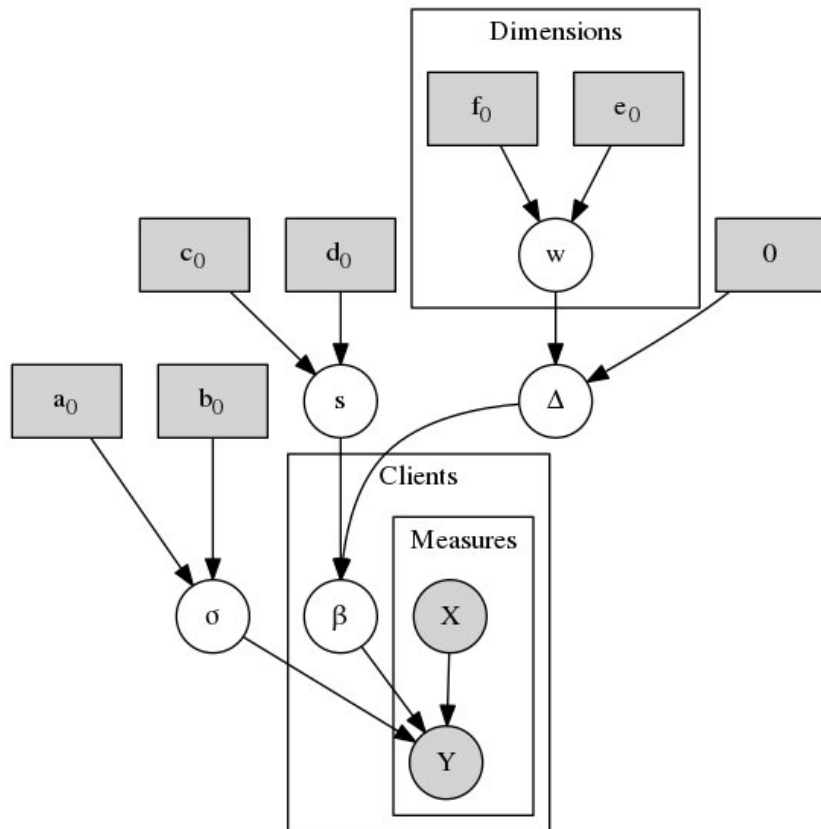
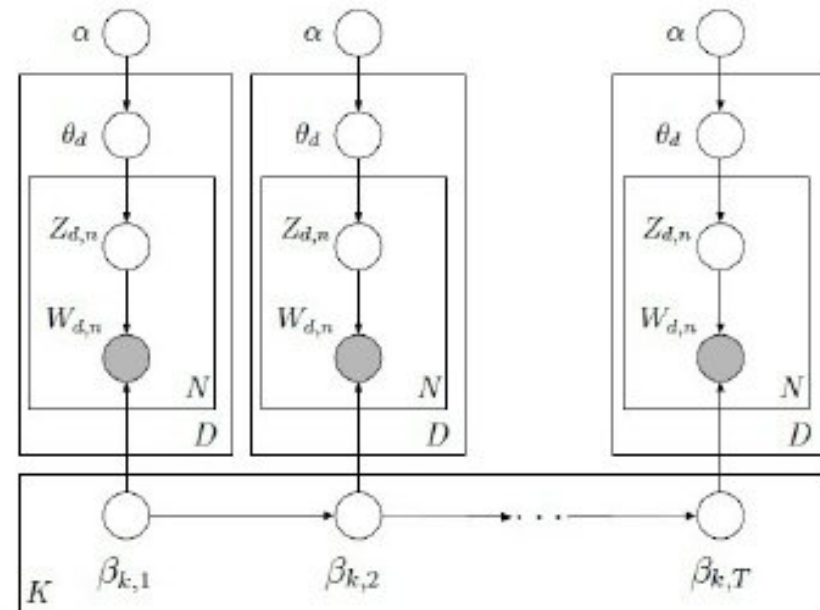


Plate notation for a hierarchical linear regression model*

Plate notation for a dynamical topic model**



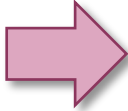
* https://www.researchgate.net/publication/328878614_Variational_Bayesian_hierarchical_regression_for_data_analysis/figures?lo=1

** https://www.researchgate.net/publication/261424851_Topic_models_and_advanced_algorithms_for_profiling_of_knowledge_in_scientific_papers/figures?lo=1

Summary: Learning Local Distributions in BNs

- Beta or Dirichlet conjugate updating is appropriate when:
 - A random variable has finitely many states (two states for Beta distribution; k states for Dirichlet distribution)
 - There are no constraints on the probabilities
 - Prior information is “like” having observed a previous sample with α_i observations in category i , for $i = 1, \dots, p$
- When there is context-specific independence:
 - Combine all configurations of parent states for which the child node has the same distribution (this is an example of “parameter tying”)
- In general, the local distribution for node X can be any function of the states of $\text{pa}(X)$
 - A node’s local distribution is a regression model with that node as the dependent variable and its parents as the independent variables
- Missing data can be handled with EM
- ICI can be represented as hidden variables and handled with EM
- General Bayesian parameter learning is inference on a set of Bayesian regression problems
 - Difficulty depends on distributions of random variables & pattern of missing data and latent variables

Unit 5 Outline

- Overview of Learning in Graphical Models
- Parameter Learning in Directed Graphical Models
-  • Structure Learning in Directed Graphical Models
- Learning in Undirected Graphical Models
- Statistical Relational Learning

Learning the Structure of a BN

- Structure of a BN represents qualitative information
 - Is there an arc between two random variables?
 - » If so, what is the direction of the arc?
 - What are the functional forms of the local distributions?
 - » Are two rows of the belief table identical (context-specific independence)?
 - » Is there a parametric equation to represent the relationship between parents and children?
 - Is there a hidden (unobserved, latent) random variable?
- Methods for learning structure
 - Score based methods
 - » Method for searching over structures
 - » Method for scoring how good a structure is
 - Bayesian score; description length; BIC or AIC score; others
 - Constraint-based methods
 - » Use conditional independence tests to identify the Markov blanket of all variables
 - » Find BN structure (or structures) matching the Markov blanket
 - Arc orientation may be undetermined for some arcs

Bayesian Structure Score

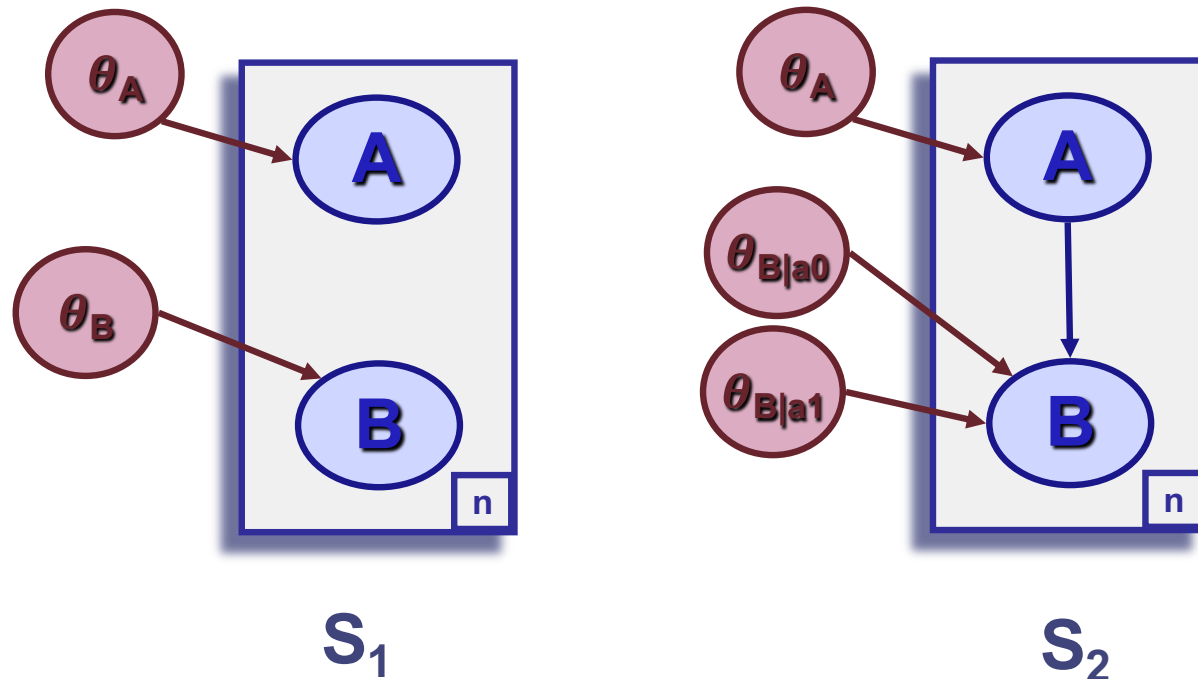
- We can represent uncertainty about the structure of a Bayesian network as a probability distribution over structures
 - Assign prior probability to each structure and to local distributions conditional on structure
 - Apply Bayes Rule to obtain posterior distribution for structures and parameters given data D
- Bayes rule for learning structure:

$$P(S_i | D) = \frac{P(D | S_i)P(S_i)}{P(D)} \quad \frac{P(S_i | D)}{P(S_j | D)} = \frac{P(D | S_i)P(S_i)}{P(D | S_j)P(S_j)}$$

- $P(D|S)$ is the marginal distribution (integrating out parameters) of $P(D|S, \theta_S)$, where θ_S is the structure-specific vector of parameters
 - Note that θ_S has different dimension for different structures
- The ratio $P(D|S_i)/P(D|S_j)$ is called the *Bayes factor* for comparing structure i against structure j
- The value $\ln P(D|S)$ is called the *Bayesian score* for S

Example of Learning Structure

- Bayesian network with 2 nodes, A and B, each with 2 possible values
- We are considering two structures, S1 and S2
- Assign prior probabilities $P(S1)$ and $P(S2)$ to the two structures
- Assign independent uniform prior distributions for parameters given structures



Computing $P(\text{Data}|\text{Structure})$

- $P(D|S, \theta_S)$ can be written as a product of factors

- For structure S1:

$$P(D|S_1, \theta_A, \theta_B) = (\theta_A)^{n_{11}+n_{10}}(1 - \theta_A)^{n_{01}+n_{00}}(\theta_B)^{n_{11}+n_{01}}(1 - \theta_B)^{n_{10}+n_{00}}$$

- For structure S2:

$$P(D|S_2, \theta_A, \theta_{B|a1}, \theta_{B|a0}) =$$

$$(\theta_A)^{n_{11}+n_{10}}(1 - \theta_A)^{n_{01}+n_{00}}(\theta_{B|a1})^{n_{11}}(1 - \theta_{B|a1})^{n_{10}}(\theta_{B|a0})^{n_{01}}(1 - \theta_{B|a0})^{n_{00}}$$

- The local and global independence assumptions allow us to integrate the θ 's out of each factor separately and multiply the results together
 - The factor for parameter θ has the form of an integral:

$$\int_{\theta=0}^1 \theta^{n_1}(1 - \theta)^{n_0} f(\theta|\alpha, \beta) d\theta \quad f(\theta|\alpha, \beta) \text{ is the Beta density function}$$

where n_{v1} observations have $V=v1$ and n_{v0} observations have $V=v0$

- This integral is called the *marginal likelihood* of D (parameter θ is integrated out)

Marginal Likelihood for the Beta Distribution

- The marginal likelihood is the likelihood of the data D integrated over the prior distribution for θ

$$\begin{aligned}
 & \int_{\theta=0}^1 \theta^{n_{v1}} (1 - \theta)^{n_{v0}} f(\theta | \alpha, \beta) d\theta \\
 &= \int_{\theta=0}^1 \theta^{n_{v1}} (1 - \theta)^{n_{v0}} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha) \Gamma(\beta)} \theta^{\alpha-1} (1 - \theta)^{\beta-1} d\theta \\
 &= \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha) \Gamma(\beta)} \int_{\theta=0}^1 \theta^{n_{v1} + \alpha - 1} (1 - \theta)^{n_{v0} + \beta - 1} d\theta \\
 &= \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha) \Gamma(\beta)} \frac{\Gamma(\alpha + n_{v1}) \Gamma(\beta + n_{v0})}{\Gamma(\alpha + \beta + n_{v1} + n_{v0})}
 \end{aligned}$$

Prior normalizing constant
divided by
Posterior normalizing constant

- If α and β are integers:

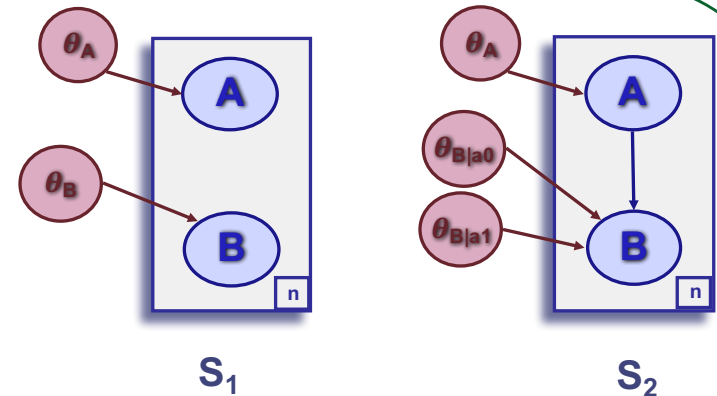
$$\int_{\theta=0}^1 \theta^{n_{v1}} (1 - \theta)^{n_{v0}} f(\theta | \alpha, \beta) d\theta = \frac{(\alpha + \beta - 1)! (\alpha + n_{v1} - 1)! (\beta + n_{v0} - 1)!}{(\alpha - 1)! (\beta - 1)! (\alpha + \beta + n_{v1} + n_{v0} - 1)!}$$

- If $\alpha = \beta = 1$ (uniform distribution):

$$\int_{\theta=0}^1 \theta^{n_{v1}} (1 - \theta)^{n_{v0}} f(\theta | \alpha, \beta) d\theta = \frac{(n_{v1})! (n_{v0})!}{(n_{v1} + n_{v0} + 1)!}$$

Example

- The learning problem:
 - Two binary nodes, A and B
 - Two possible structures, S1 and S2
- The prior distribution:
 - Structures have prior probabilities $P(S1) = 0.7$ and $P(S2)=0.3$
 - Conditional on structure, all local distributions have independent uniform distributions



- The data: Counts in each category

$n_{a1,b1}$	5	$n_{a0,b1}$	2
$n_{a1,b0}$	3	$n_{a0,b0}$	3

- The marginal likelihoods:
- Structure posterior probabilities:

$$P(D|S_1) = \frac{n_{a1}!n_{a0}!n_{b1}!n_{b0}!}{(n+1)!(n+1)!} = \frac{8!5!7!6!}{14!14!}$$

$$P(S_1|D) \propto P(S_1)P(D|S_1) = 0.7 \frac{8!5!}{14!} \frac{7!6!}{14!} = 1.62 \times 10^{-9}$$

$$P(D|S_2) = \frac{n_{a1}!n_{a0}!n_{a1b1}!n_{a1b0}!n_{a0b1}!n_{a0b0}!}{(n+1)!(n_{a1}+1)!(n_{a0}+1)!} = \frac{8!5!5!3!2!3!}{14!9!6!}$$

$$P(S_2|D) \propto P(S_2)P(D|S_2) = 0.3 \frac{8!5!}{14!} \frac{5!3!}{9!} \frac{2!3!}{6!} = 5.51 \times 10^{-10}$$

Example (cont)

- Usually we compare structures by computing relative posterior probabilities $P(S_i|D)/P(S_j|D)$

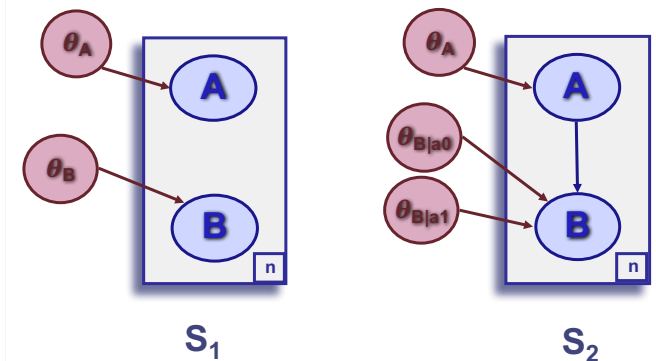
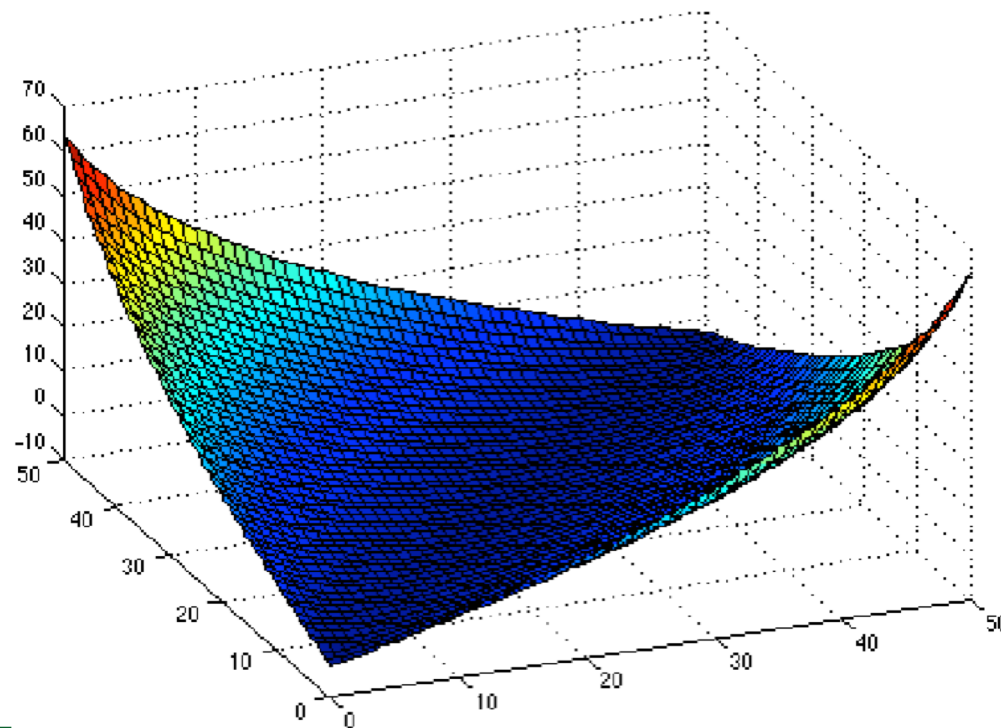
$$- P(S_1|D)/P(S_2|D) = \frac{0.7}{0.3} \frac{\frac{8!5!7!6!}{14!14!}}{\frac{8!5!5!3!2!3!}{14!9!6!}} = 2.94$$

$$- P(S_1|D) = 2.94/(1+2.94) = 0.746$$

- If all structures have equal prior probabilities we need only $P(D|S)$
 - Ratio $P(D|S_1) / P(D|S_2)$ depends only on marginal likelihoods for nodes with different parents in S_1 than in S_2
 - Marginal likelihoods for nodes with same parents cancel out
- For large networks and large numbers of observations:
 - The number of structures is astronomical and each has tiny probability
 - We usually work with $\ln P(D|S)$ to prevent numeric underflow
 - $\ln P(D|S)$ is called the *Bayesian Dirichlet (BD) score*
 - The goal is usually to find one or a few good structures, not to estimate the posterior probability of any one structure

How the Posterior Probability Varies as Counts Vary

- The data: 100 observations divided into categories as follows:
 - a1,b1 m1
 - a1,b0 50-m1
 - a0,b1 m2
 - a0,b0 50-m2
- Plot of $\ln [P(S_2|D)/P(S_1|D)]$ against m_1 and m_2 (uniform prior)



Structure S_2 with arc from A to B becomes more probable as k_1 differs more from k_2

Marginal Likelihood for the Dirichlet Distribution

- When a node has more than 2 states we use the Dirichlet distribution
- The marginal likelihood for the Dirichlet distribution is similar to the marginal likelihood for the Beta distribution
- The marginal likelihood for n_1, n_2, \dots, n_r observations in states 1, 2, ..., r is:

$$\frac{\Gamma(\alpha_1 + \dots + \alpha_r)}{\Gamma(\alpha_1) \dots \Gamma(\alpha_r)} \frac{\Gamma(\alpha_1 + n_1) \dots \Gamma(\alpha_r + n_r)}{\Gamma(\alpha_1 + \dots + \alpha_r + n_1 + \dots + n_r)}$$

- If the α_i are all integers then the marginal likelihood is:

$$\frac{(\alpha_1 + \dots + \alpha_r - 1)!}{(\alpha_1 - 1)! \dots (\alpha_r - 1)!} \frac{(\alpha_1 + n_1 - 1)! \dots (\alpha_r + n_r - 1)!}{(\alpha_1 + \dots + \alpha_r + n_1 + \dots + n_r - 1)!}$$

- For a uniform prior distribution the marginal likelihood is:

$$(r-1)! \frac{n_1! \dots n_r!}{(n_1 + \dots + n_r + r - 1)!}$$

- Usually take logs to avoid numeric underflows

The K2 BN Learning Algorithm

- We have covered the machinery necessary to understand a basic Bayesian learning algorithm
- Assumptions behind the K2 learning algorithm
 - Assume ordering of nodes is given
 - All graphs consistent with node ordering are equally likely *a priori*
 - All local distributions of nodes given parents are uniform
 - » Beta(1,1) for binary nodes
 - » Dirichlet(1,1,...,1) distribution for nodes with more than 2 states
- K2's basic approach to structure learning:
 - K2 evaluates structures by $P(D|S)$
 - Under the equal-prior assumption, this is proportional to $P(S|D)$
 - The algorithm uses a greedy search over structures and returns the structure with the highest $P(D|S)$
 - This is a local optimum of the posterior distribution of structures given the observations
 - In $P(D|S)$ is called the K2 score; the K2 algorithm finds a local optimum of the K2 score

Steps in the K2 BN Learning Algorithm

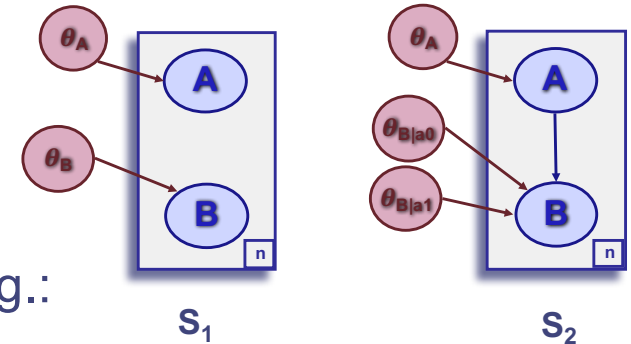
(Assume a given order of the nodes)

1. Begin with a network with no arcs
2. Look for the best arc to add
 - a) Compute score $\ln P(D|S)$ of network with each new arc
 - b) This can be computed efficiently because only one term in the sum changes when an arc is added (the local distribution of the node at the tail of the arc)
 - c) Pick highest scoring arc
3. If best arc from Step 2 does not increase posterior probability of resulting network, go to Step 4. Else add best-scoring arc to the network and go to Step 2.
4. Output network with highest posterior probability

(Cooper and Herskovits, 1992)

Some Common Score Functions

- K2 score
- BD score
- BDe (e = likelihood equivalence)
 - Uses a single global virtual sample size, e.g.:
 - » For S_1 - $\theta_A \sim \text{Beta}(1,1), \theta_B \sim \text{Beta}(1,1)$
 - » For S_2 - $\theta_A \sim \text{Beta}(1,1), \theta_{B|a1} \sim \text{Beta}\left(\frac{1}{2}, \frac{1}{2}\right), \theta_{B|a0} \sim \text{Beta}\left(\frac{1}{2}, \frac{1}{2}\right)$
 - » Prior for *all* structures is “like” 2 total prior observations distributed evenly across states $a_1 b_1, a_1 b_0, a_0 b_1, a_0 b_0$
 - Ensures that any two networks with same likelihood have the same score
- Bayesian Information Criterion (BIC) / Minimum Description Length (MDL) / Schwartz Information Criterion
- Akaike Information Criterion (AIC)
- Log-likelihood Score / Entropy



The above score functions are available in the bnlearn R package
 For more information see http://www.lx.it.pt/~asmc/pub/talks/09-TA/ta_pres.pdf

Structure Learning Using MB Detection: Basic Method

- Use conditional independence tests to detect the strongly relevant variables for each node
 - Strongly relevant variables carry information that cannot be obtained from any other variable
 - In a causal graph these are the parents, children and co-parents
- Find V-structures (aka “colliders”) and remove co-parent links
 - V-structures are independent causes that become dependent when conditioned on common effect
 - Arcs in a V-structure can be oriented
- Propagate orientation constraints

Example Markov Blanket Detection algorithms:

- Grow-shrink algorithm (the original and the simplest)
- Incremental Association Markov Blanket (IAMB) and variants
- Total Conditioning (TC) and variants

Pellet and Elisseff (2008)

Grow-Shrink Algorithm

1. For each node X , initialize MB_X (Markov Blanket of X) to empty set
2. Grow phase: Add variables to MB_X if dependent on X given current contents of MB_X
 - We keep adding new variables to MB_X until X is conditionally independent of rest of network given MB_X
3. Shrink phase: Identify and remove any variables from MB_X that are conditionally independent of X given the rest of MB_X
 - The process of adding nodes to MB_X may have rendered other variables in MB_X independent of X given the rest of MB_X
 - The shrink phase removes these extra variables from MB_X

<https://www.cs.cmu.edu/~dmarg/Papers/PhD-Thesis-Margaritis.pdf>

Special Structures for Bayesian Network Classifiers

- Naïve Bayes
 - Class node is root
 - All features independent given class node
 - Simple, robust, commonly used classifier
- Tree-augmented naïve Bayes (TAN)
 - Class node is root
 - All features are children of class node
 - When class node is removed, remaining network is a tree
 - Often improves on naïve Bayes classifier
- Bayes net augmented naïve Bayes (BAN)
 - Class node is root
 - All features are children of class node
 - When class node is removed, remaining nodes form arbitrary Bayesian network
 - Sometimes improves on TAN, but requires more search

Examples of Free Software for Learning Bayesian Network Structure

- bnlearn is an R package for learning Bayesian networks from data
 - Discrete or continuous variables
 - » Continuous variables must have Gaussian distribution and may have no discrete children – this is called a conditional Gaussian network
 - Score-based or conditional independence test-based learning algorithms
 - Supports blacklist (arcs not allowed) and whitelist (arcs required)
 - Inference in learned network
 - » Approximate inference via likelihood weighting algorithm
 - » Translate to gRain network for exact inference
 - <http://www.jstatsoft.org/v35/i03/paper>
- bnstruct is an R package for learning Bayesian networks with missing data
 - <https://academic.oup.com/bioinformatics/article/33/8/1250/2730229>
- SMILearn (from Bayes Fusion, formerly GeNIe/SMILE) is free for academic use
 - <https://www.bayesfusion.com/academic-users>
- Weka has several Bayesian network classifiers:
 - <http://www.cs.waikato.ac.nz/~remco/weka.bn.pdf>
- Other packages (last updated June 2014) can be found at
 - <http://www.cs.ubc.ca/~murphyk/Software/bnsoft.html>

Efficient Summing over Many Structures (Friedman and Koller, 2002)

- Most learning algorithms attempt to find a single high-probability structure
- There typically are many structures consistent with the data
 - The number of structures is super-exponential in the number of nodes
- We would like to be able to compute the posterior probability of a feature (e.g., is A a parent of B?)
 - This requires summing over all structures:
$$P(F \mid D) = \sum_F P(F \mid S)P(S \mid D)$$
- Under certain assumptions on the prior distribution, the structure score decomposes into a product of factors
 - Structure modularity: Given a node ordering, the choice of parents for any node is independent of the choice of parents for other nodes
 - Global parameter independence: parameter for one node is independent of parameters for other nodes
 - Parameter modularity: If X has the same parents in two different structures then the parameter prior is the same for both structures
- If the node ordering is fixed and the number of parents per node is bounded, the above sum can be computed efficiently by factoring products outside the sum

Computing the Posterior Probability of a Feature

- Suppose the ordering of the nodes is fixed and known and the number of parents of each node is smaller than a bound k
- We can write:

$$P(D | O) = \sum_{S \in S_O} \prod_i score(X_i, Pa_S(X_i) | D) = \prod_i \sum_{U \in \mathcal{U}_{i,O}} score(X_i, U | D)$$

- O is a fixed node ordering
- S_O is the set of structures consistent with node ordering O
- $\mathcal{U}_{i,O}$ is the set of possible parents of node X_i consistent with node ordering O and the bound on the number of parents
- If there are n nodes and no more than k parents per node there are fewer than $nk+1$ terms to compute in this product of sums
- For certain types of feature (e.g., $A \rightarrow B$?) this same trick can be applied to computing feature probabilities. Then we obtain:

$$P(F | O, D) = \frac{P(F, D | O)}{P(D | O)}$$

- For unknown node ordering use Monte-Carlo sampling to average over node orderings

Learning with Local Structure: Partitions

- Partitions of parent variable
 - When there is context-specific independence, local distributions are the same for some combinations of parent variable
 - In a partition model, we need to learn one probability distribution for each element of a partition of the state space of a parent variable
 - We apply exactly the same method for computing $P(D|S)$ except that we pool all the cases in a given partition element
- Modifications to basic learning algorithm:
 - Define a procedure for searching over partitions
 - Scoring rule pools cases in same partition element
- Friedman and Goldszmidt (1997 UAI)
 - 2 different ways to define partitions: classification tree or default+exception
 - Learning was more efficient
 - » Richer graph structures with less data
 - » Better predictive performance with less data
- Edera et al (2014) – Grow-Shrink algorithm with context-specific independence

Recap

- We considered methods for parameter learning
 - Beta and Dirichlet conjugate prior distributions treated in detail
 - EM algorithm can be used when observations are missing
 - General regression methods were mentioned briefly
- We considered methods for structure learning
 - Simple K2 Bayesian search and score method
 - » Fixed node order is given
 - » Greedy search over arc presence
 - » Scoring method based on uniform prior distributions
 - Different score functions
 - Markov blanket detection algorithms
 - Using within-node structure (e.g., ICI, context-specific independence) to increase efficiency of learning
 - We have not discussed:
 - » Searching over node orders (stochastic search is commonly used)
 - » Learning structure with missing observations
- We have seen that in fully Bayesian learning parameter and structure learning are intertwined

Hard Problems in Learning

- Learning with more complex kinds of information
 - Missing data / hidden variables
 - Non independent and identically distributed data
 - Combining observations with other kinds of information (not just Beta or Dirichlet priors)
 - Constraints on the model (partitions are easy; other kinds of constraints can be very hard)
- Combinatorics and search over structures
 - There are 2 parts to a learning algorithm: searching for plausible structures and evaluating structures
 - Search for structures is especially important for more complex learning problems
 - » Missing data and hidden variables
 - » Local search algorithms get "stuck" at local optima
 - We have discussed a Bayesian evaluation method: compare structures by their relative posterior probabilities
 - There is a LARGE number of possible structures when there are many variables
 - Open research issue: search methods that quickly focus in on a high probability set of structures

Learning with Incomplete Information

- Learning is hard with missing data and hidden variables
- Structural EM (Friedman, 1978) combines greedy search over structures with the EM algorithm for estimating parameters
 - Need to approximate marginal likelihood
 - Can get stuck in local minima
 - Use random restarts to avoid being stuck in a bad local optimum
 - bnstruct R package implements SEM
- Monte Carlo simulation (e.g., Laskey and Myers, 2003) can be used to learn structure and parameters in presence of missing data and hidden variables
 - Initialize structure and missing/hidden observations
 - Make random changes in structure (add/delete arcs) and missing/hidden observations
 - Accept or reject changes according to a probabilistic rule
 - Long-run distribution: arcs and missing/hidden observations are sampled from posterior distribution given the actual observations
 - Algorithm generates a sample of structures and missing/hidden observations -- can make inferences about probability of arc

Summary: Bayesian Learning of Parameters Given Structure

- Given the graph structure, we need to learn a set of probability distributions for each node, one distribution for each configuration of its parent variables
 - If there is local structure in the form of partitions, then one distribution needs to be learned for each partition element
 - Probability of child given parent is a regression model
- Conjugate families are mathematically convenient for representing prior information about the probabilities
 - Simple updating formula
 - Interpretation as “virtual sample summary”
 - Larger virtual prior sample size → more highly concentrated prior distribution
 - Ratio of actual sample size to virtual prior sample size determines relative effects of prior and likelihood
- The Beta (for binary nodes) or Dirichlet (for many-valued nodes) are conjugate families for unconstrained discrete probabilities
- Parameterized models increase efficiency of learning
- EM algorithm can be used when there are missing data

Summary: Bayesian Learning of Structures

- The posterior probability of a structure $P(S|D)$ is proportional to the marginal likelihood of the observations $P(D|S)$ times the prior probability of a structure $P(S)$
- There is an explicit formula for the marginal likelihood when the prior distributions for the local distributions comes from a conjugate family
 - We examined the case of Beta / Binomial and Dirichlet / Multinomial conjugate families
 - There are other common conjugate pairs
- Generally there are too many structures to enumerate them all
- Approaches:
 - Heuristic search
 - Stochastic search
 - Summing over large number of structures

Combining Data with Expert Judgment in Learning Bayesian Networks

- Heckerman et al (1995) suggest the following method for Bayesian learning to combine expert knowledge with data:
 - Expert specifies a Bayesian network (structure and probabilities)
 - This expert-specified Bayesian network defines a joint distribution on all variables in the network
 - This joint distribution plus a global virtual sample size defines a BDe prior distribution on parameters given any structure
 - We can search over structures using the BDe score to evaluate structures
 - Steck (2008) provided a method to find an optimal virtual sample size from data
- Expert knowledge about arc existence and orientation can be included
 - Many BN learning algorithms allow “blacklists” (arcs not allowed in learned network) and “whitelists” (arcs that must be in the learned network)
 - » Expert can say “A cannot cause B,” or “A must be included as a cause of B”
 - We can define an informative prior on structures by allowing expert to assign probabilities for arc existence and orientation

Unit 5 Outline

- Overview of Learning in Graphical Models
- Parameter Learning in Directed Graphical Models
- Structure Learning in Directed Graphical Models
-  • Learning in Undirected Graphical Models
- Statistical Relational Learning

Learning Undirected Graphical Models

- The problem: learn structure and parameters for undirected graphical model

$$p(x) = \frac{1}{Z} \prod_c \psi_c(x_c | \theta)$$

- Structure: Arcs (implies cliques) and functional form of potential functions
 - Parameters: Parameters of parameterized potential functions
- Complicating factor: partition function Z is global normalization and prevents decomposing into separate learning problem for each clique
- With complete data, parameter learning problem by MAP or MLE is a convex optimization problem
 - Solve by iterative gradient ascent or by Monte Carlo
 - EM for missing data (no longer convex)
- Structure learning
 - Algorithms based on conditional independence tests
 - Algorithms based on scores
 - » Use approximations to marginal likelihood

Unit 5 Outline

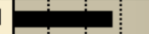

- Overview of Learning in Graphical Models
- Parameter Learning in Directed Graphical Models
- Structure Learning in Directed Graphical Models
- Learning in Undirected Graphical Models
- ➔ • Statistical Relational Learning

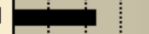

Relational Learning

- In Unit 3 we learned about expressive representations for knowledge
- Relational probabilistic languages (PRM, OOBN, MEBN, etc.) represents probabilistic knowledge about entities, attributes, relationships
 - Fragments of BN can be repeatedly instantiated for different entities
- Learning relational models requires methods for appropriately treating the repeated structure
 - Pool data for learning identical CPTs (*parameter tying*)
 - Learn parameters of combining functions for CPTs with many instantiations of a parent
 - Avoid “double counting” errors of naïve learning approaches
- Relational learning is an active research area



Some Complexities for Relational Learning

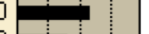

- Cases have different numbers of random variables
 - Case 1 has two random variables
 - Case 2 has three random variables
- Object relationships may have different cardinalities
 - obj1 is related to rep1 (one relationship)
 - obj2 is related to rep2 and rep3 (two relationships)
- Learning algorithms will give incorrect results if repeated structure is not handled properly
 - Naïve “database join” approach is to make a single table with columns for TargetType and ObjectType and a row for each object/report pair
 - If reports are not evenly distributed across object types this can result in poor estimate of TargetType distribution
 - For example: If most friendly objects have only one report and most enemy objects have several reports then this approach will overestimate the frequency of enemy objects
- Relational models may have parameterized combining rules that must be learned



TargetType(obj1)		
Friendly	70.0	
Enemy	30.0	

ReportedType(rep1)		
Friendly	59.0	
Enemy	41.0	

Case 1:
Single Report for Object

TargetType(obj2)		
Friendly	70.0	
Enemy	30.0	

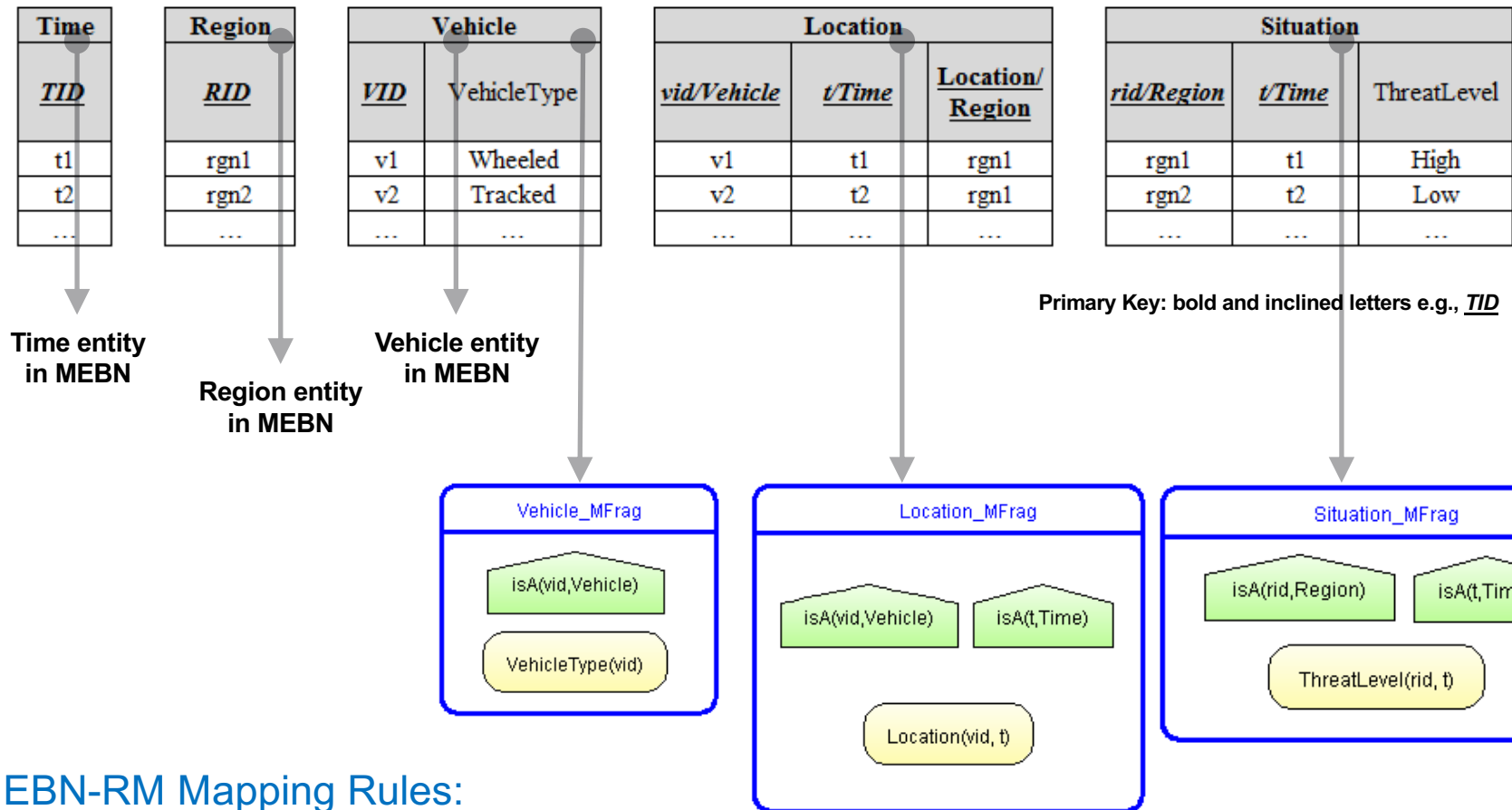
ReportedType(rep2)		
Friendly	59.0	
Enemy	41.0	

ReportedType(rep3)		
Friendly	59.0	
Enemy	41.0	

Case 2:
Two Reports for Object

TargetType	ReportedType
Enemy	Enemy
Friendly	Friendly
Enemy	Enemy
Enemy	Friendly
Friendly	Friendly
Enemy	Enemy
...	...

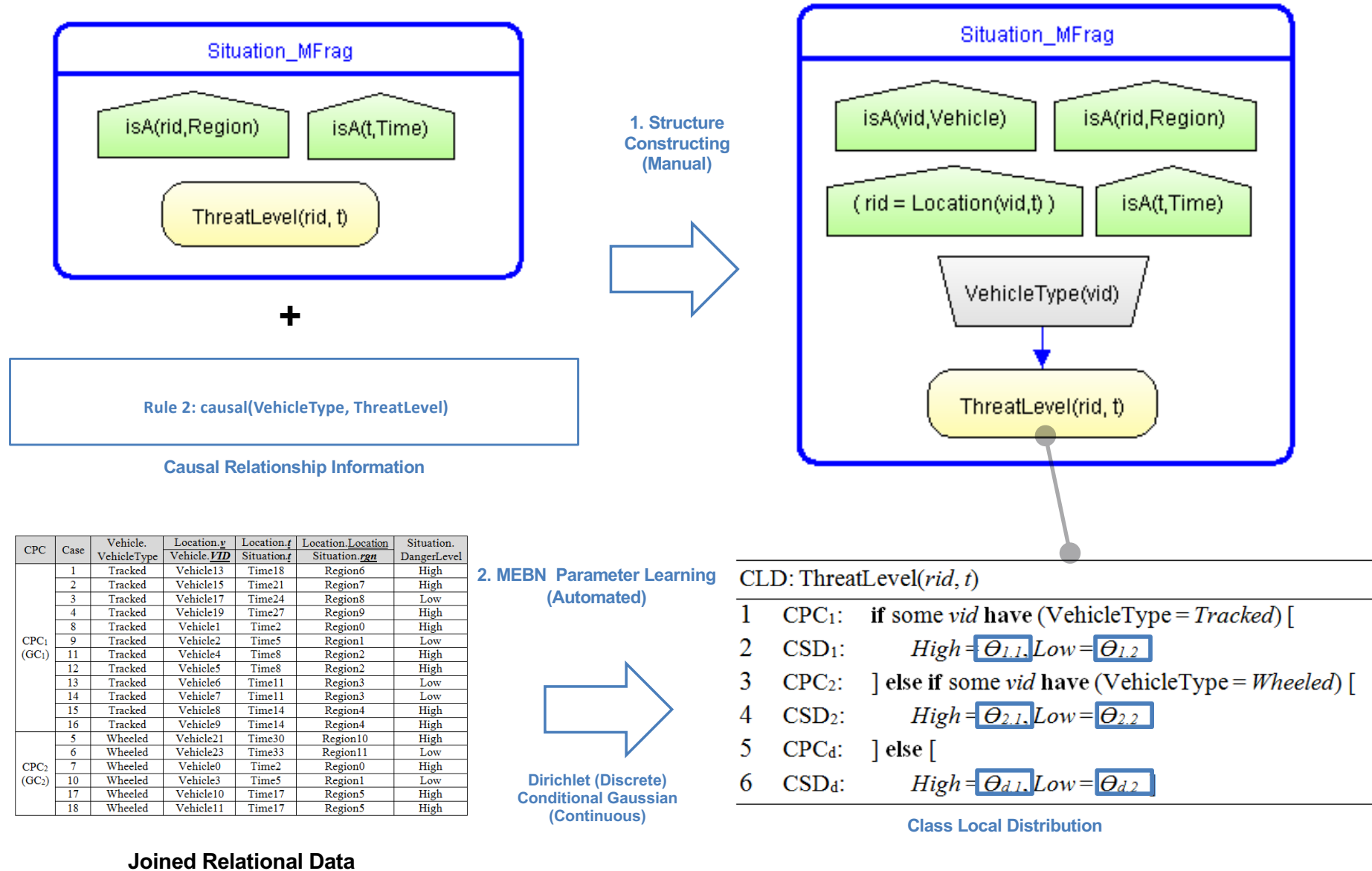
MEBN-RM: Map Relational Database to MEBN Elements for MEBN Learning



MEBN-RM Mapping Rules:

1. Entity Mapping
2. Predicate resident node Mapping
3. Function resident node Mapping
4. Relation Schema and MFrag Mapping
5. Relational Database Schema and MTheory Mapping

MEBN Parameter Learning from Relational Database



Tutorials on Statistical Relational Learning

- <http://people.cs.kuleuven.be/~luc.deraedt/iclp2009.pdf>
- <http://www.cs.umd.edu/~getoor/cmsc828g/Slides/SRL-Tutorial-05-08.pdf>

Summary and Synthesis

- Elements of a typical learning algorithm
 - Inference about parameters given structure
 - Inferring structure
 - » Score based and independence test based approaches
- Learning can be treated as Bayesian inference
 - Directed graphical model relates BN structure & parameters to observations
 - Simplest case: observations are independent realizations from the BN to be learned
- K2 Algorithm (simplest case)
 - Complete data
 - Parameter prior is uniform
 - Evaluation function is marginal likelihood of data
 - Search is greedy; node ordering is assumed given
 - Select single best structure
- Extensions we treated
 - Informative Beta or Dirichlet parameter prior
 - Missing data and hidden variables
 - Stochastic search
 - Multiple structures versus single good structure
- Undirected models introduce complexity due to partition function
- Relational learning introduces complexities due to repeated structure

References for Unit 5

The original reference on learning Bayesian networks from data

- Cooper, G. and Herskovits, E. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning* 9: 309-347, 1992.

Additional general references

- Daly, R., Shen, Q. and Aitken, S. Learning Bayesian Networks: Approaches and Issues. *The Knowledge Engineering Review*, 26(02):99–157, 2011.
- Jordan, M. (ed.) *Learning in Graphical Models* (Adaptive Computation and Machine Learning), MIT Press, 1998.
- D. Heckerman. A tutorial on learning Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, March, 1995.

Combining Expert Knowledge and Data

- Heckerman, D., Geiger, D. and Chickering, D. (1995). Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning* 20: 197-243
- Steck H (2008). Learning the Bayesian Network Structure: Dirichlet Prior versus Data. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI2008)*, pp. 511-518.

EM and Structural EM

- Dempster, A. P., N. M. Laird, and Rubin, D. (1977). Maximum Likelihood Estimation from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society* 39: 1-38
- Friedman, N. (1998a). The Bayesian Structural EM Algorithm. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, San Mateo, CA: Morgan Kaufmann Publishers

Learning with Local Structure

- Friedman, N. and Goldszmidt, M. (1996) Learning Bayesian Networks with Local Structure. *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, San Mateo, CA: Morgan Kaufmann Publishers.
- Edera, A., Strappa, Y. and Bromberg, F. (2014) The Grow-Shrink strategy for learning Markov network structures constrained by context-specific independences, *IBERAMIA*, 2014. <https://arxiv.org/pdf/1407.8088.pdf>

Bayesian Learning of Structures

- Friedman, N. and Koller, D. (2002) Being Bayesian About Network Structure: A Bayesian Approach to Structure Discovery in Bayesian Networks, *Machine Learning*. Available at <http://robotics.stanford.edu/~koller/papers.html>
- Laskey, K.B. and Myers, J. (2001) Population Markov Chain Monte Carlo, *Machine Learning*, 2001.
- Scutari, M. (2010) Learning Bayesian Networks with the bnlearn R Package, *Journal of Statistical Software* 35(3).

Markov blanket detection algorithms

- Pellett, J. and Elisseeff, A. Using Markov Blankets for Causal Structure Learning. (2008) *Journal of Machine Learning Research*. <http://www.jmlr.org/papers/volume9/pellet08a/pellet08a.pdf>

Learning Relational Models

- Getoor, K.; and Jensen, D. 2003. *Proceedings of the IJCAI 2003 Workshop on Learning Statistical Models from Relational Data*. AAAI Press, 2003

For further references browse the reference sources on the course web site.