Machine Learning

Prof. Dr. Stefan Kramer
Johannes Gutenberg-Universität Mainz

Acknowledgements

- Eibe Frank
- Ian Witten
- Tom Mitchell

Outline

- Bayesian learning and Naive Bayes
- Brief introduction to Bayesian Networks
- Linear regression

Bayesian Theorem

likelihood

posterior

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)}$$

prior(s)

P(h) = prior probability of hypothesis h

P(D) = prior probability of training data D

P(h|D) = conditional probability of h given D

P(D|h) = conditional probability of D given h

Naive Bayes Classifier

- Assume target function f: X → V, where each instance x is described by attributes $\{a_1, \ldots, a_n\}$
- Most probable value of f(x) is:

$$v_{MAP} = \operatorname*{argmax}_{v_j \in V} P(v_j | a_1, a_2 \ldots a_n)$$

$$v_{MAP} = \operatorname*{argmax}_{v_j \in V} \frac{P(a_1, a_2 \ldots a_n | v_j) P(v_j)}{P(a_1, a_2 \ldots a_n)}$$

$$= \operatorname*{argmax}_{v_j \in V} P(a_1, a_2 \ldots a_n | v_j) P(v_j)$$

- The Naive Bayes assumption of conditional independence

$$P(a_1, a_2 \ldots a_n | v_j) = \prod_i P(a_i | v_j)$$

gives the *Naive Bayes classifier*:

$$v_{NB} = \operatorname*{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$$

Naive Bayes Example

- Consider PlayTennis again, and a new instance (Outlook=sunny, Temp=cool, Humid=high, Wind=strong)
- Want to compute:

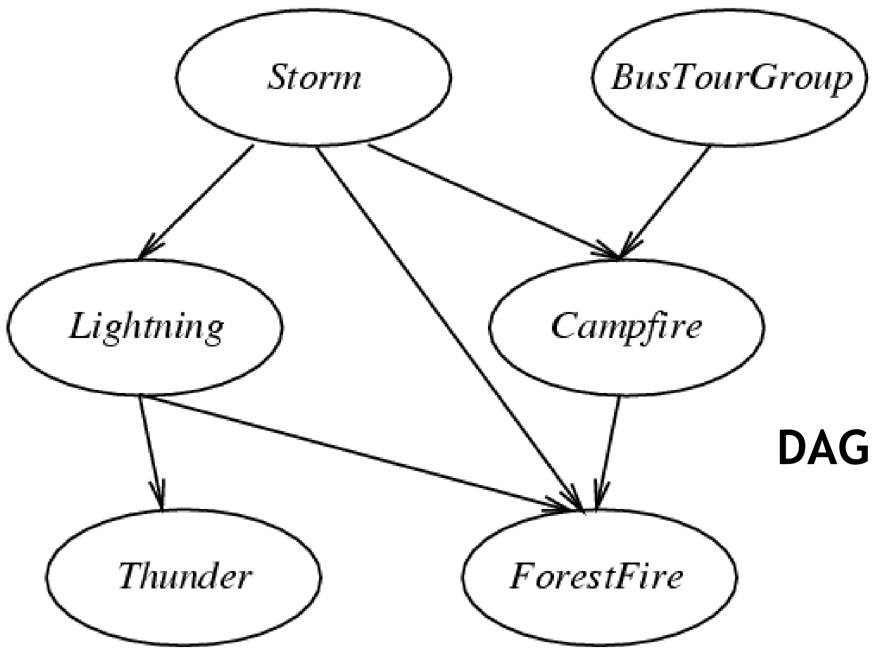
$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}} P(v_j) \prod_i P(a_i|v_j)$$

- P(y) P(sunny|y) P(cool|y) P(high|y) P(strong|y) = 0.005
- P(n) P(sunny|n) P(cool|n) P(high|n) P(strong|n) = 0.021

Bayesian Networks

Bayesian Networks

- Each node is asserted to be *conditionally independent* of its *non-descendants*, *given* its immediate *predecessors*
- Represents *joint probability distribution* over all variables, e.g., P(Storm, BusTourGroup, ..., ForestFire)



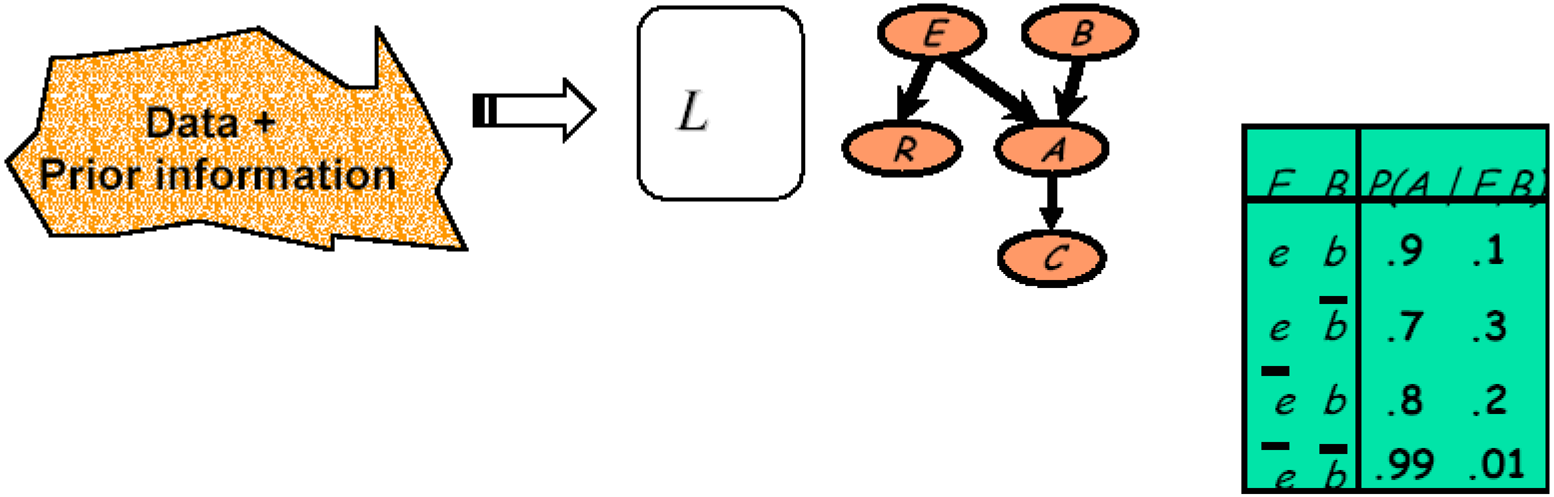
DAG

	S,B	$S,\neg B$	$\neg S,B$	$\neg S,\neg B$
C	0.4	0.1	0.8	0.2
$\neg C$	0.6	0.9	0.2	0.8

Inference in Bayesian Networks

- How can one infer the (probabilities of) values of one or more network variables, given observed values of others?
  - Bayes net contains all information needed for this inference
  - if only one variable with unknown value, easy to infer it
  - in general case, problem is *NP hard*
- In practice, can succeed in many cases
  - exact inference methods work well for some network structures
  - Monte Carlo methods “simulate” the network randomly to calculate approximate solutions

Learning in Bayesian Networks

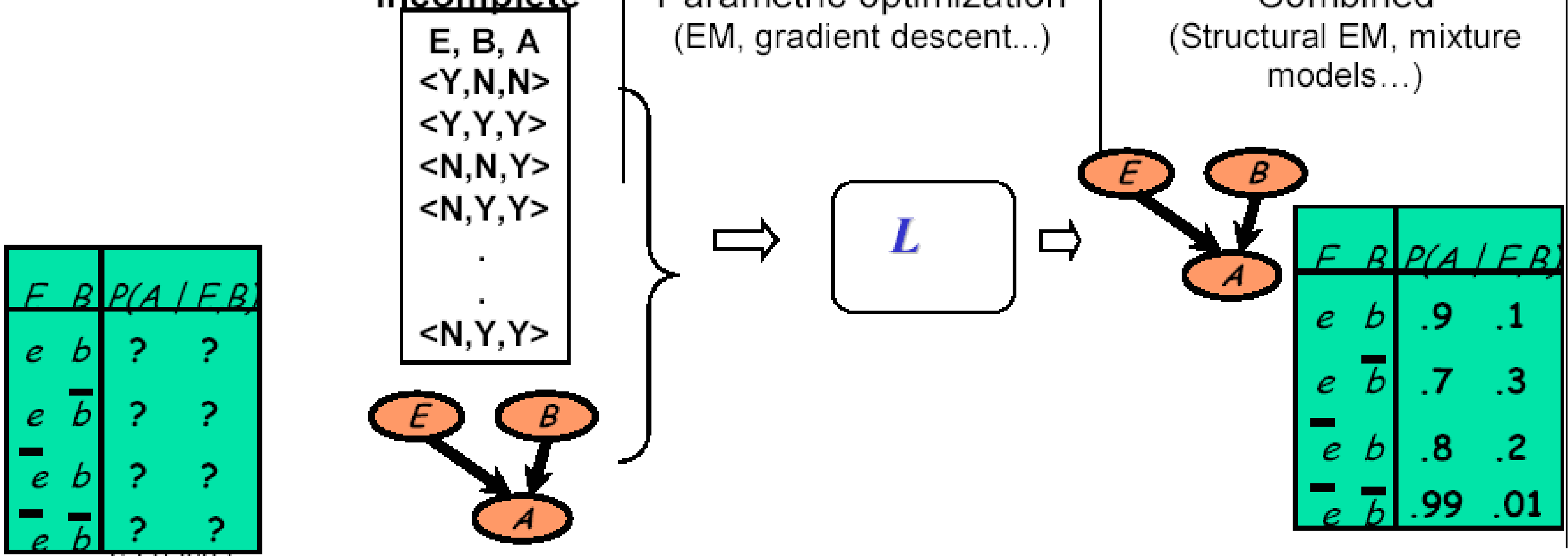


Learning Problems

	Known Structure	Unknown Structure
Complete Data	Statistical parametric estimation (closed-form eq.)	Discrete optimization over structures (discrete search)
Incomplete Data	Parametric optimization (EM, gradient descent...)	Combined (Structural EM, mixture models…)

Learning Problem

	Known Structure	Unknown Structure
Complete	Statistical parametric estimation (closed-form eq.)	Discrete optimization over structures (discrete search)
Incomplete	Parametric optimization (EM, gradient descent...)	Combined (Structural EM, mixture models…)



Learning Parameters

- Estimate of parameters relies on *sufficient statistics*, i.e., functions summarizing the data / relevant information for calculating likelihood
  - e.g., N_H and N_T are sufficient statistics for the binomial distribution
- One option: choose parameters maximizing the likelihood function (*maximum likelihood estimation*)

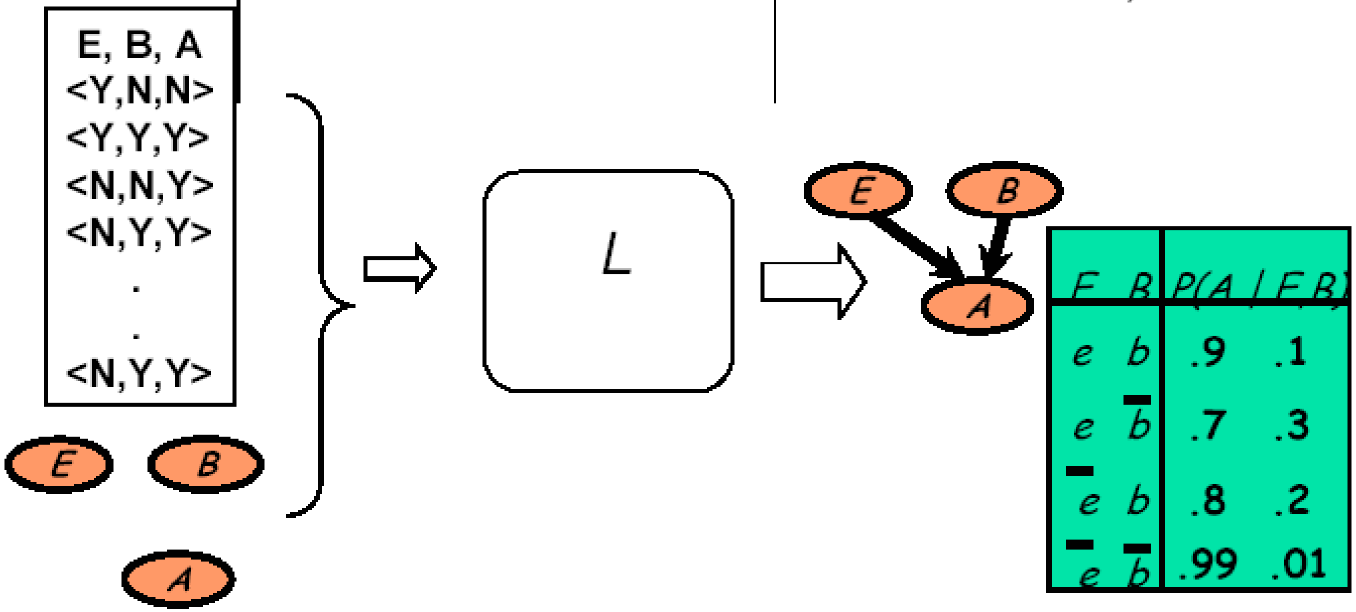
Bayesian Approach

- „Imaginary counts“ / choice of priors?
- *Conjugate families*: posterior distribution follows the same parametric form as prior distribution
- *Dirichlet prior* is the conjugate family for the multinomial likelihood
- Parameter estimation: MLE and Bayesian

$$\hat{\theta}_{x_i|pa_i} = \frac{N(x_i, pa_i)}{N(pa_i)} \qquad \tilde{\theta}_{x_i|pa_i} = \frac{\alpha(x_i, pa_i) + N(x_i, pa_i)}{\alpha(pa_i) + N(pa_i)}$$

Learning Problem

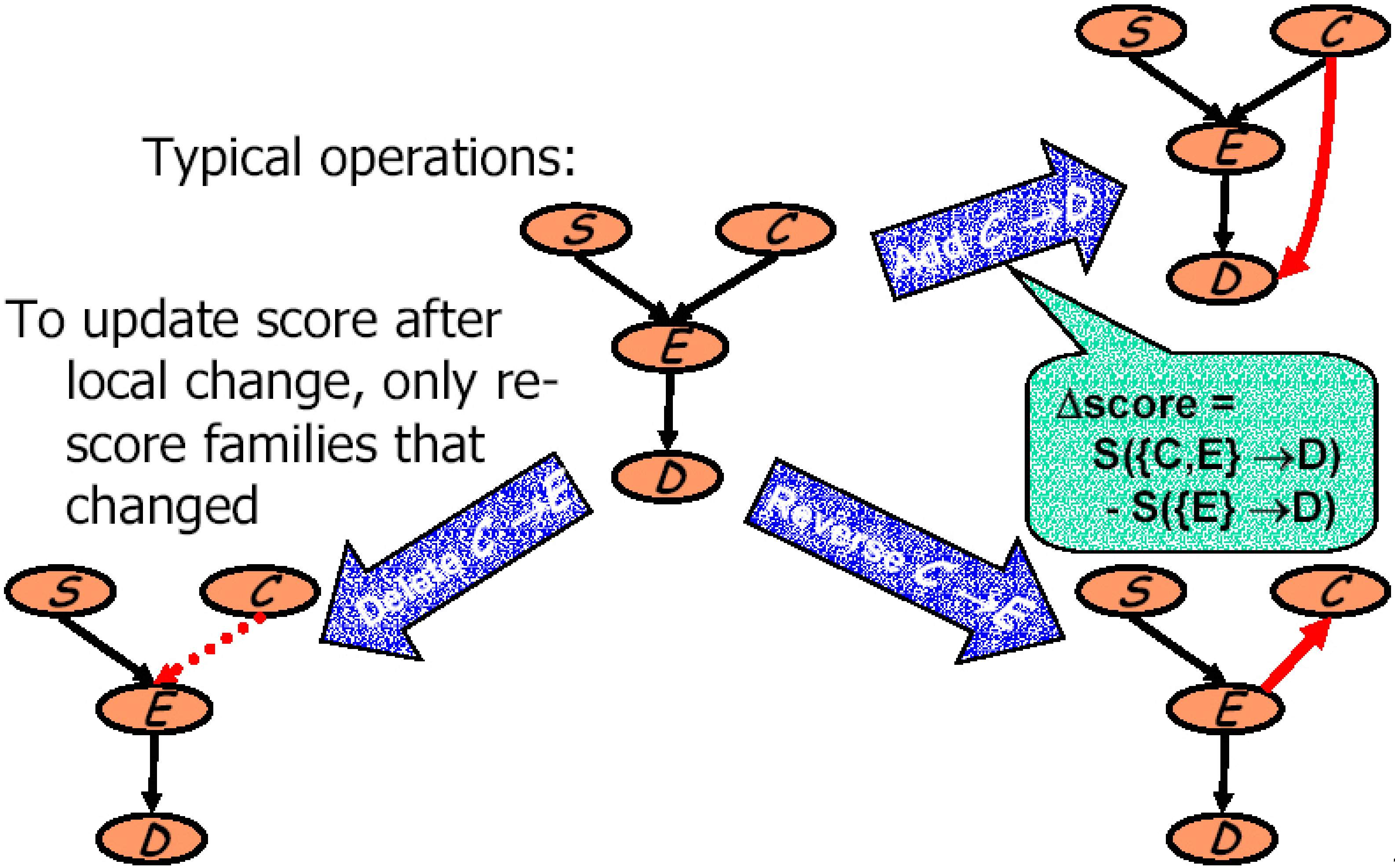
	Known Structure	Unknown Structure
Complete	Statistical parametric estimation (closed-form eq.)	Discrete optimization over structures (discrete search)
Incomplete	Parametric optimization (EM, gradient descent...)	Combined (Structural EM, mixture models…)



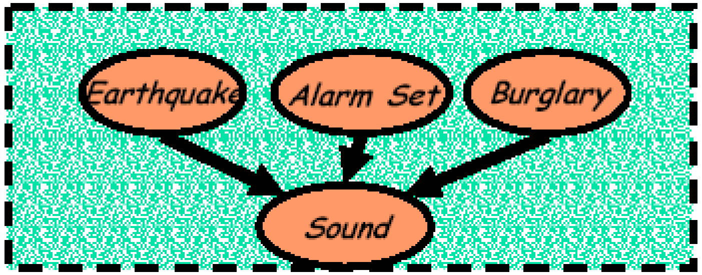
Heuristic Search

Typical operations:

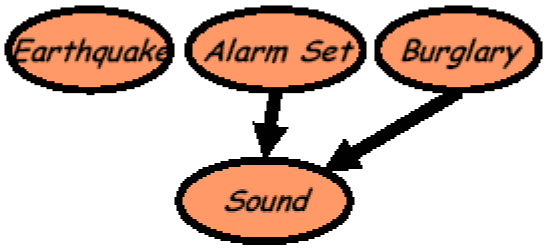
To update score after local change, only re-score families that changed



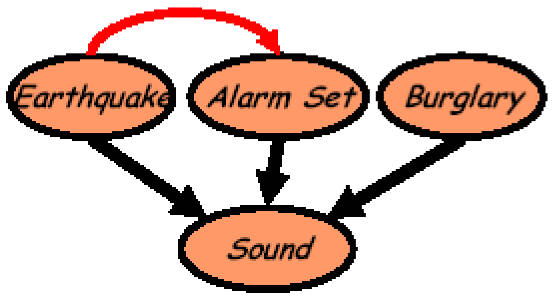
Why Do We Need Accurate Structure?



Missing an arc



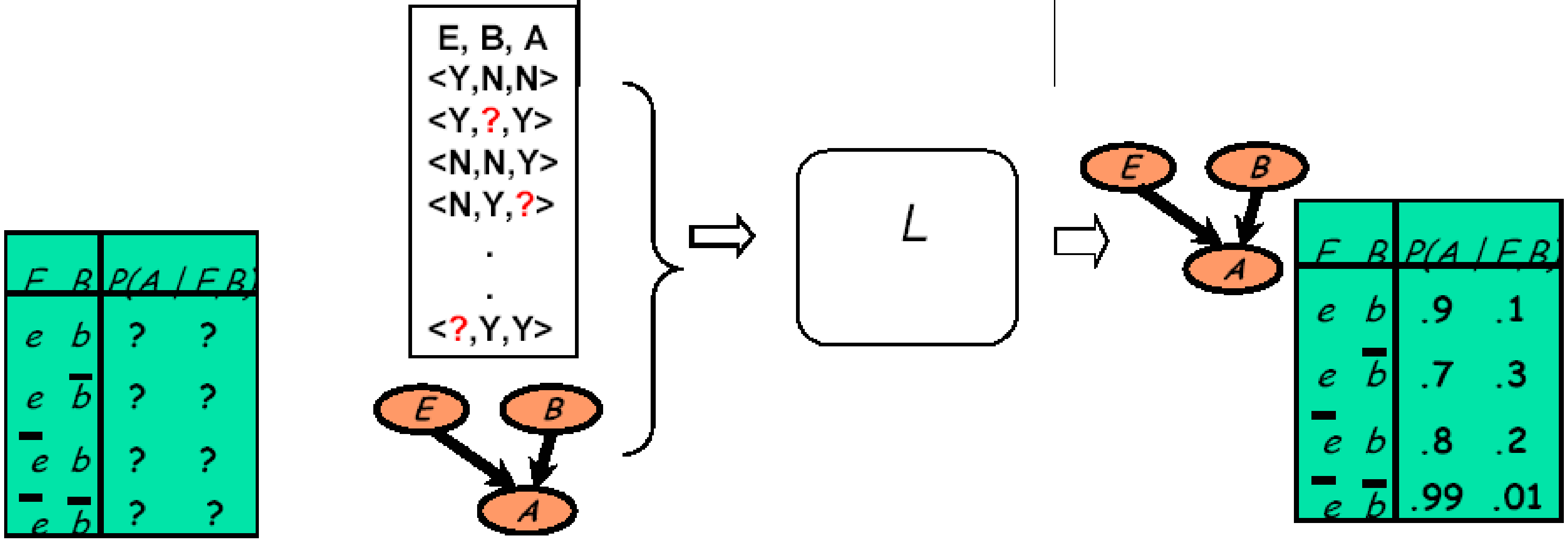
Extraneous arc



- *Missing arc* cannot be compensated for
- *Extraneous arc* increases number of parameters to be estimated
- *Wrong assumptions about domain structure*

Learning Problem

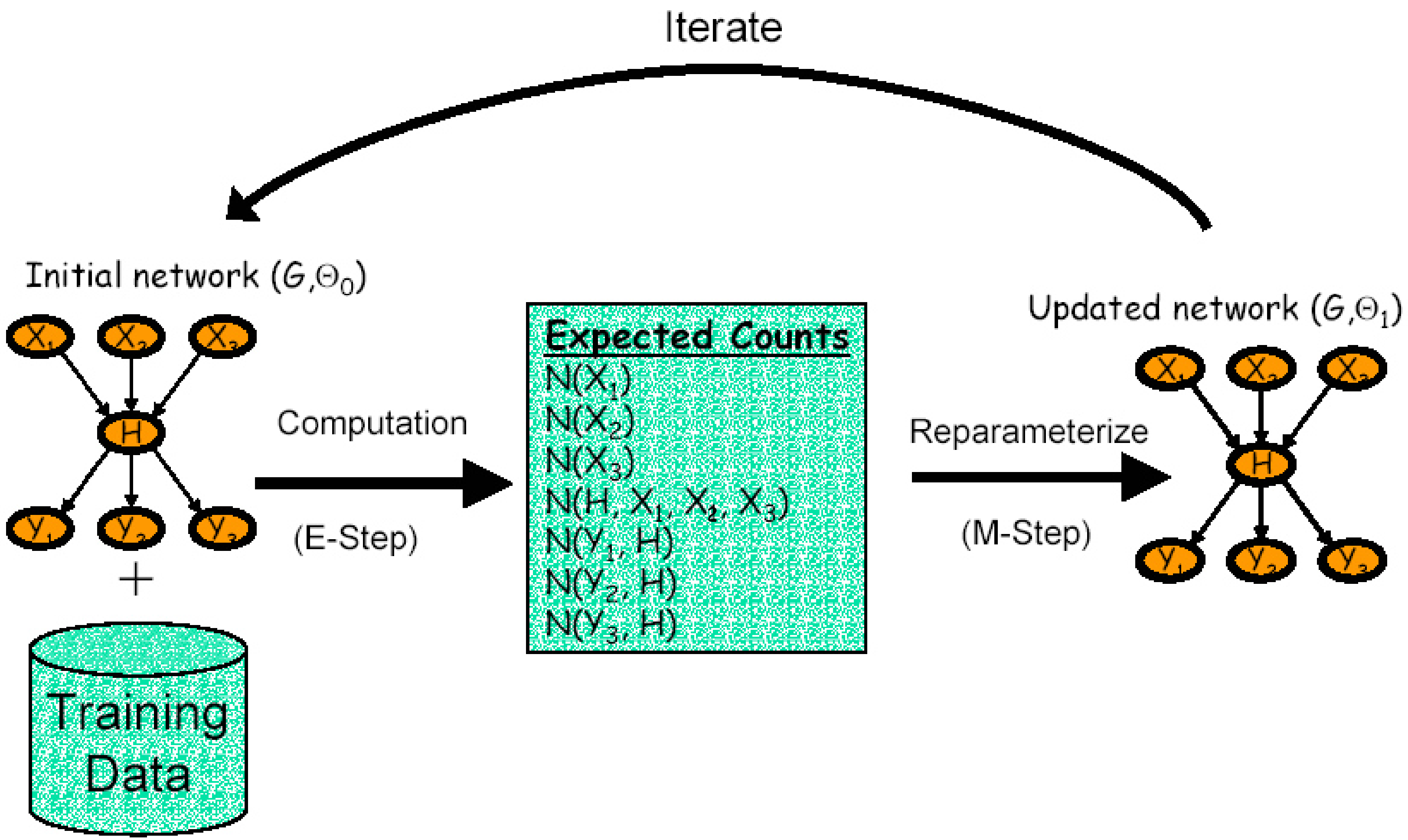
	Known Structure	Unknown Structure
Complete	Statistical parametric estimation (closed-form eq.)	Discrete optimization over structures (discrete search)
Incomplete	Parametric optimization (EM, gradient descent...)	Combined (Structural EM, mixture models…)



Incomplete Data

- Data are often incomplete
  - some variables of interest are not assigned values
- This phenomenon occurs when we have *missing values*
- Some *variables unobserved* in some instances
  - hidden variables
  - some variables may *never be observed*
  - *we might not even* know *they exist*

Expectation Maximization (EM)

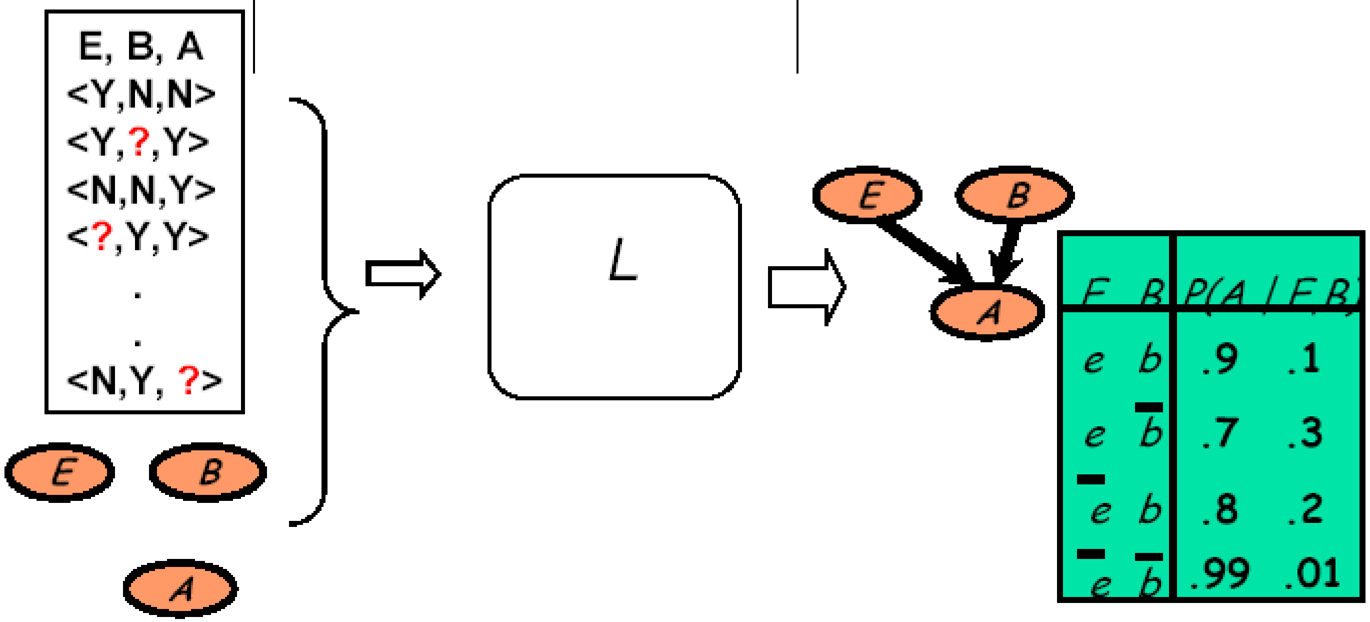


Expectation Maximization (EM)

- Computational bottleneck: computation of expected counts in E-Step
  - need to compute posterior *for each unobserved variable in each instance of training set*
  - *all posteriors for an instance can be derived from one pass of standard BN inference*

Learning Problem

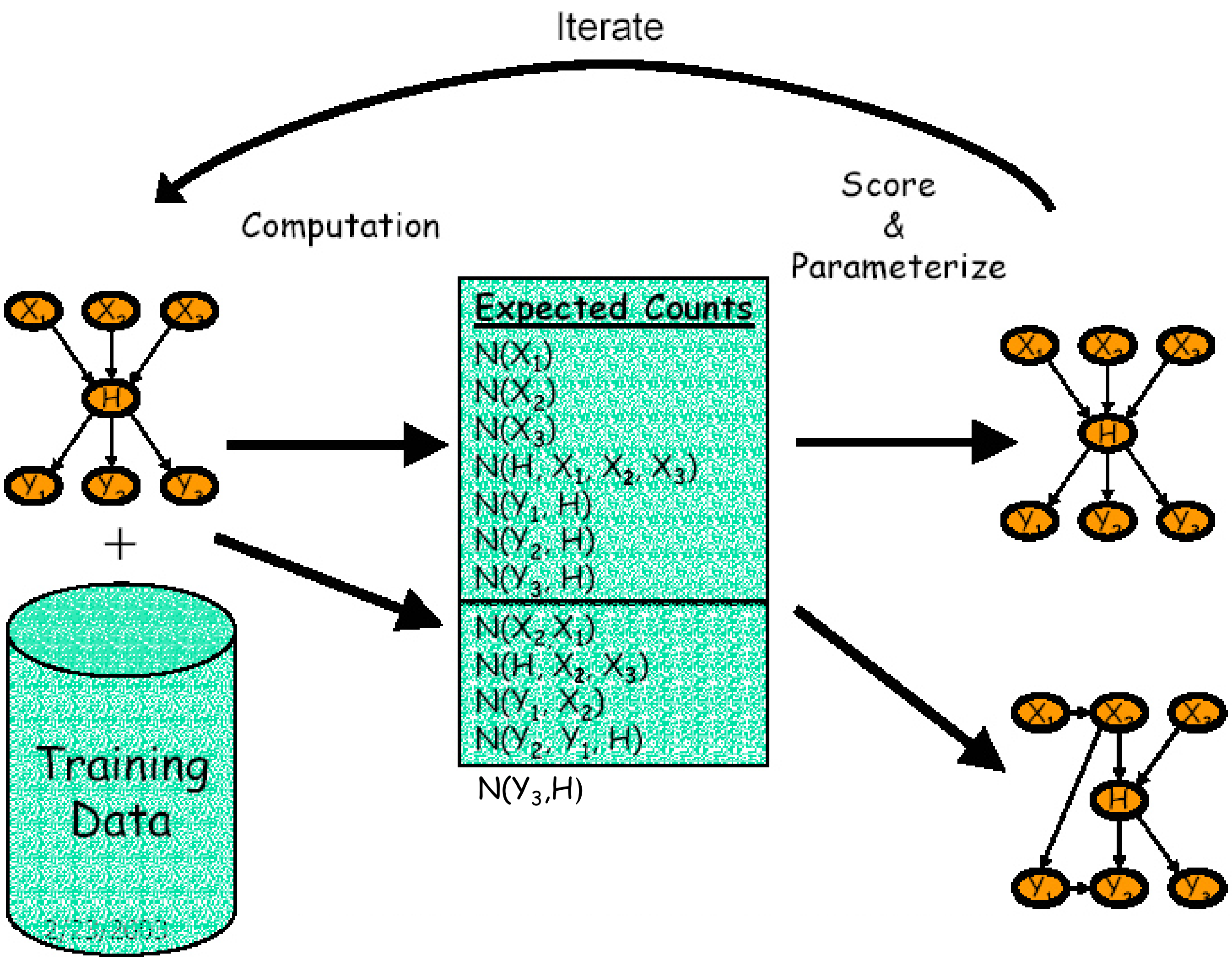
	Known Structure	**Unknown Structure**
Complete	Statistical parametric estimation (closed-form eq.)	Discrete optimization over structures (discrete search)
Incomplete	Parametric optimization (EM, gradient descent...)	Combined (Structural EM, mixture models…)



Structural EM

- Idea: use current model to help evaluate new structures
- Outline: perform search in (Structure, Parameters) space
- At each iteration, use current model for finding either
  - better scoring parameters: “parametric” EM step, or
  - better scoring structure: “structural” EM step

Structural EM



Conclusion

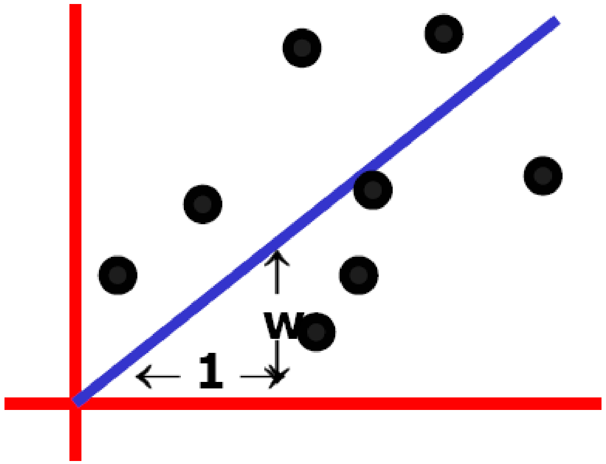
- Learning in Bayesian networks
- Four cases
- From simple counting (easy) to learning both structure and parameters (extremely hard)

Linear Regression

Linear Models

- Work most naturally with numeric attributes
- Standard technique for numeric prediction: *linear regression*
- Outcome is linear combination of attributes:
  $y = w_0 + w_1 x_1 + \ldots + w_m x_m$
- Weights are calculated from the training data
- *Predicted value* for first training instance $\mathbf{x}^{(1)}$:
  $w_0 + w_1 x_1^{(1)} + \ldots + w_m x_m^{(1)}$

Linear Regression



DATASET

inputs	outputs
$x_1 = 1$	$y_1 = 1$
$x_2 = 3$	$y_2 = 2.2$
$x_3 = 2$	$y_3 = 2$
$x_4 = 1.5$	$y_4 = 1.9$
$x_5 = 4$	$y_5 = 3.1$

- Linear regression assumes that the expected value of the output given an input, E[y|x], is linear.
- Simplest case: Out(x) = wx for some unknown w.
- Given the data, we can estimate w.

1-Parameter Linear Regression

Assume that the data is formed by

$$y_i = wx_i + \text{noise}_i$$

where

- the noise signals are independent
- the noise has normal distribution with mean 0 and unknown variance σ^2

$p(y|w,x)$ has normal distribution with

- mean wx
- variance σ^2

Bayesian Linear Regression

$p(y|w,x)$ = Normal(mean wx, var σ^2)

We have a set of data points $(x_1,y_1), (x_2, y_2), \ldots, (x_n, y_n)$, which are evidence about w.

We want to infer w from the data.

$$p(w|x_1, \ldots, x_n, y_1, \ldots, y_n)$$

- You can use Bayes rule to work out a posterior distribution for w given the data
- Or you could do Maximum Likelihood Estimation

MLE of w

Ask the question: „For which value of w is the data most likely to have happened?“

$\Leftrightarrow$

For what w is

$$p(y_1, \ldots, y_n | x_1, \ldots, x_n, w) \text{ maximized?}$$

$\Leftrightarrow$

For what w is

$$\prod_{i=1}^{n} p(y_i | w, x_i) \text{ maximized?}$$

Derivation (cf. Lecture on Bayesian Learning)

For what w is

$$\prod_{i=1}^{n} p(y_i | w, x_i) \text{ maximized?}$$

For what w is

$$\prod_{i=1}^{n} \exp\left(-\frac{1}{2}\left(\frac{y_i - wx_i}{\sigma}\right)^2\right) \text{ maximized?}$$

For what w is

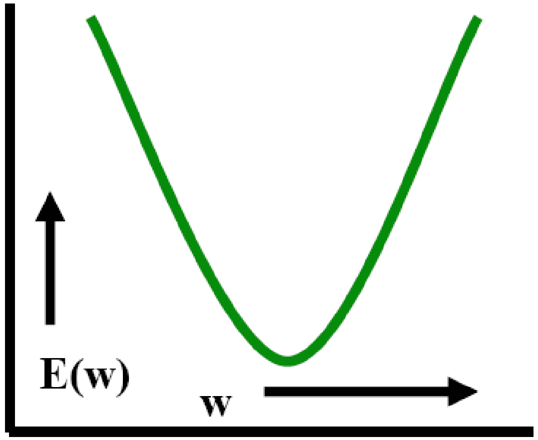
$$\sum_{i=1}^{n} -\frac{1}{2}\left(\frac{y_i - wx_i}{\sigma}\right)^2 \text{ maximized?}$$

For what w is

$$\sum_{i=1}^{n} \left(y_i - wx_i\right)^2 \text{ minimized?}$$

Linear Regression

The maximum likelihood w is the one that minimizes the sum-of-square residuals.



$$E = \sum_i (y_i - wx_i)^2$$

$$= \sum_i y_i^2 - \left(2\sum x_i y_i\right)w + \left(\sum x_i^2\right)w^2$$

We want to minimize a quadratic function of w.

Linear Regression

- Easy to show the sum of squares is minimized when

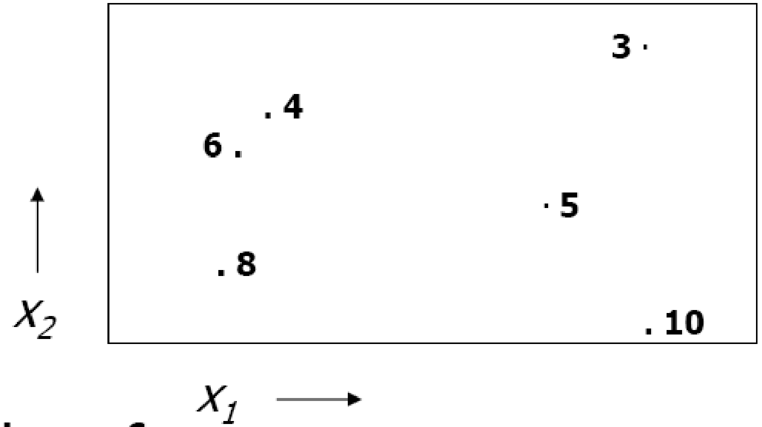
$$w = \frac{\sum x_i y_i}{\sum x_i^{\ 2}}$$

- The maximum likelihood model is Out(x) = wx

- We can use that for prediction.

Multivariate Regression

What if the inputs are vectors?



2-d input example

Dataset has form

$\mathbf{x}_1$	y_1
$\mathbf{x}_2$	y_2
$\mathbf{x}_3$	y_3
$\vdots$	$\vdots$
$\mathbf{x}_R$	y_R

Multivariate Regression

$$\mathbf{x} = \begin{bmatrix} \cdots\cdots\mathbf{x}_1\cdots\cdots \\ \cdots\cdots\mathbf{x}_2\cdots\cdots \\ \vdots \\ \cdots\cdots\mathbf{x}_R\cdots\cdots \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ & & \vdots & \\ x_{R1} & x_{R2} & \cdots & x_{Rm} \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_R \end{bmatrix}$$

Matrix **X** and **y**: R data points, inputs consisting of m components

The linear regression model assumes a vector ***w*** such that

$$\mathrm{Out}(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} = w_1 x[1] + w_2 x[2] + \ldots . w_m x[D]$$

The max. likelihood ***w*** is $\boldsymbol{w} = (X^T X)^{-1}(X^T Y)$

Multivariate Regression

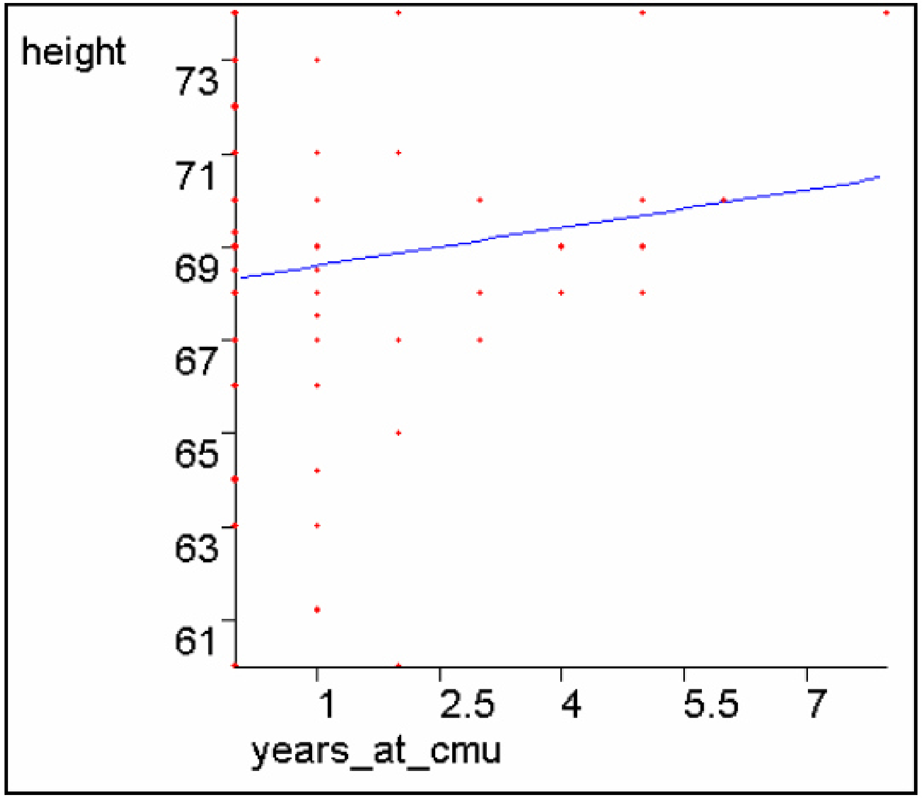
The max. likelihood $\boldsymbol{w}$ is $\boldsymbol{w} = (X^T X)^{-1}(X^T Y)$

$X^T X$ is an m x m matrix: i,j'th elt is $\sum_{k=1}^{R} x_{ki} x_{kj}$

$X^T Y$ is an m-element vector: i'th elt $\sum_{k=1}^{R} x_{ki} y_k$

Constant Term?

- We may expect linear data that does not go through the origin.
- Statisticians and neural net folks all agree on a simple obvious hack.
- *Can you guess?*



Use of Constant Term

The trick is to create a fake input X_0 *that always takes the value 1!*

X_1	X_2	Y
2	4	16
3	4	17
5	5	20

Before:

$Y = w_1 X_1 + w_2 X_2$

…has to be a poor model

X_0	X_1	X_2	Y
1	2	4	16
1	3	4	17
1	5	5	20

After:

$Y = w_0 X_0 + w_1 X_1 + w_2 X_2$

$= w_0 + w_1 X_1 + w_2 X_2$

…has a fine constant term

In this example, You should be able to see the MLE w_0, w_1 and w_2 by inspection

Ridge Regression

- Problem: if independent variables (features) are strongly correlated (near collinearity), then the estimated regression coefficients are unstable (high variance)
- Possible solution: e.g., *ridge regression*
  - Regression coefficients stabilized by mathematical trick ($X^T X$ is artificially modified by adding λI)
  - Not least squares (LS) solution anymore (no unbiased estimates)
  - Ridge parameter λ (e.g., set by cross-validation) determines how much ridge regression deviates from LS regression (if too small, it cannot fight collinearity, if too large, bias too strong)