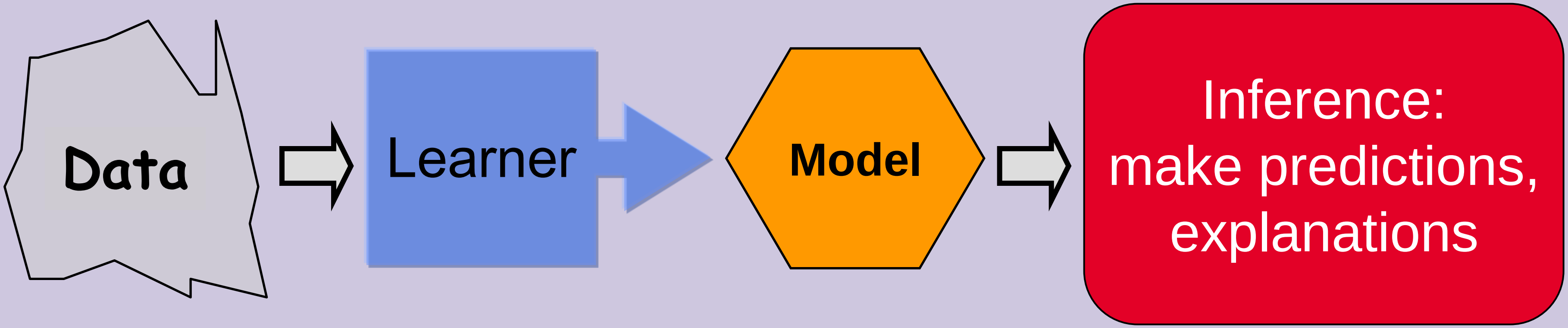
Learning Graphical Models

Gal Elidan

Hebrew University

(few slides are thanks to Nir Friedman and Daphne Koller)

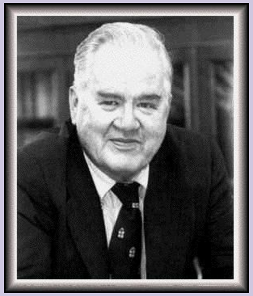
Why Machine Learning



Statistics and ML have many common goals:

models that fit data, prediction, probabilistic explanation, ways to cope with uncertainty, discovering truths about the data…

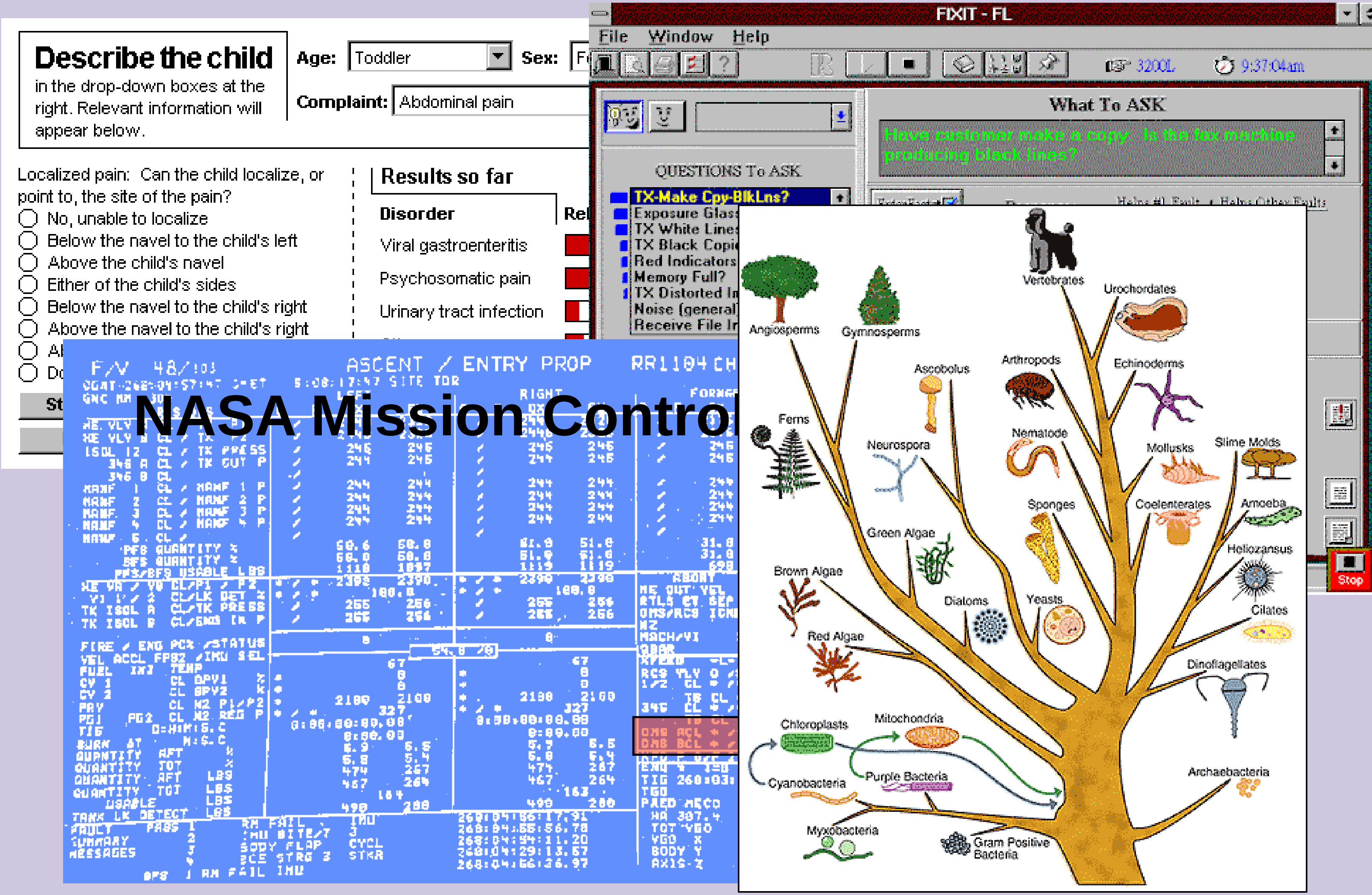
John Tukey
(“bit”,”software”)



“Exploratory data analysis is an attitude, a flexibility…” (1980)

- Learning = hypothesis exploration + estimation
- Algorithms cope with high-dimensional domains

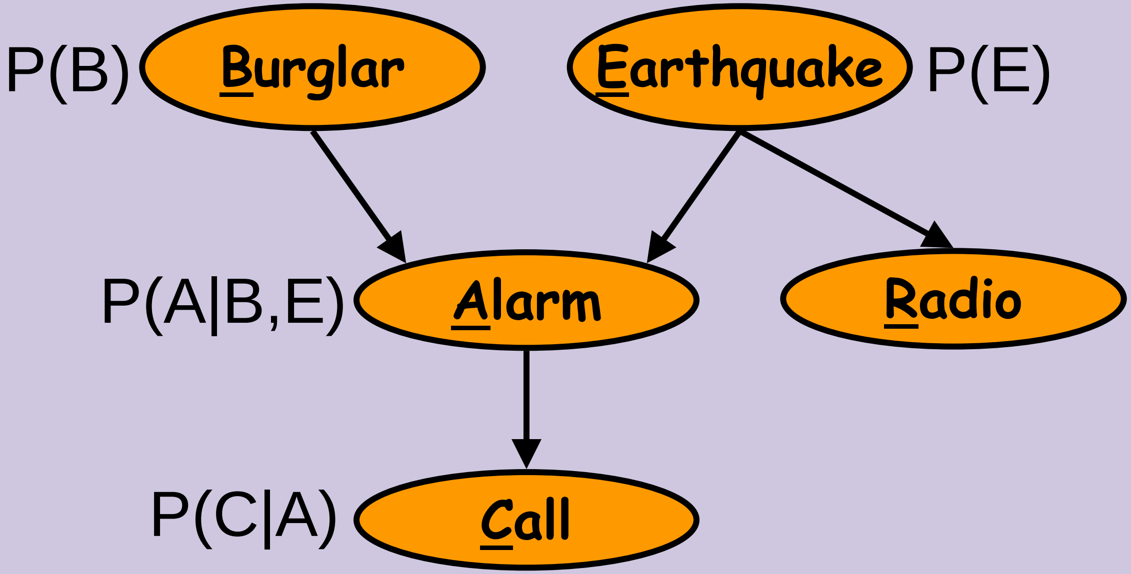
Bayesian networks are everywhere



Overview

- Introduction
- Inference
- Parameter Estimation
- Model Selection

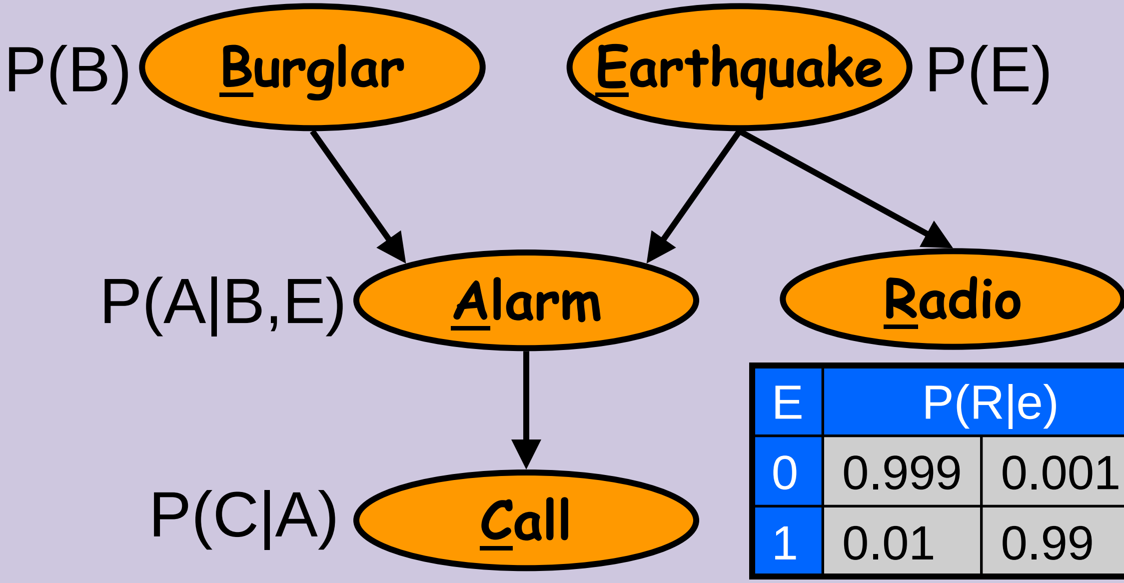
Bayesian Networks



Independence assumptions:

$C \perp E, B, R \mid A$

Bayesian Networks



E	P(R|e)	
0	0.999	0.001
1	0.01	0.99

Independence assumptions:

$$X_i \perp \mathrm{NonDesc}_i \mid \mathrm{Par}_i$$

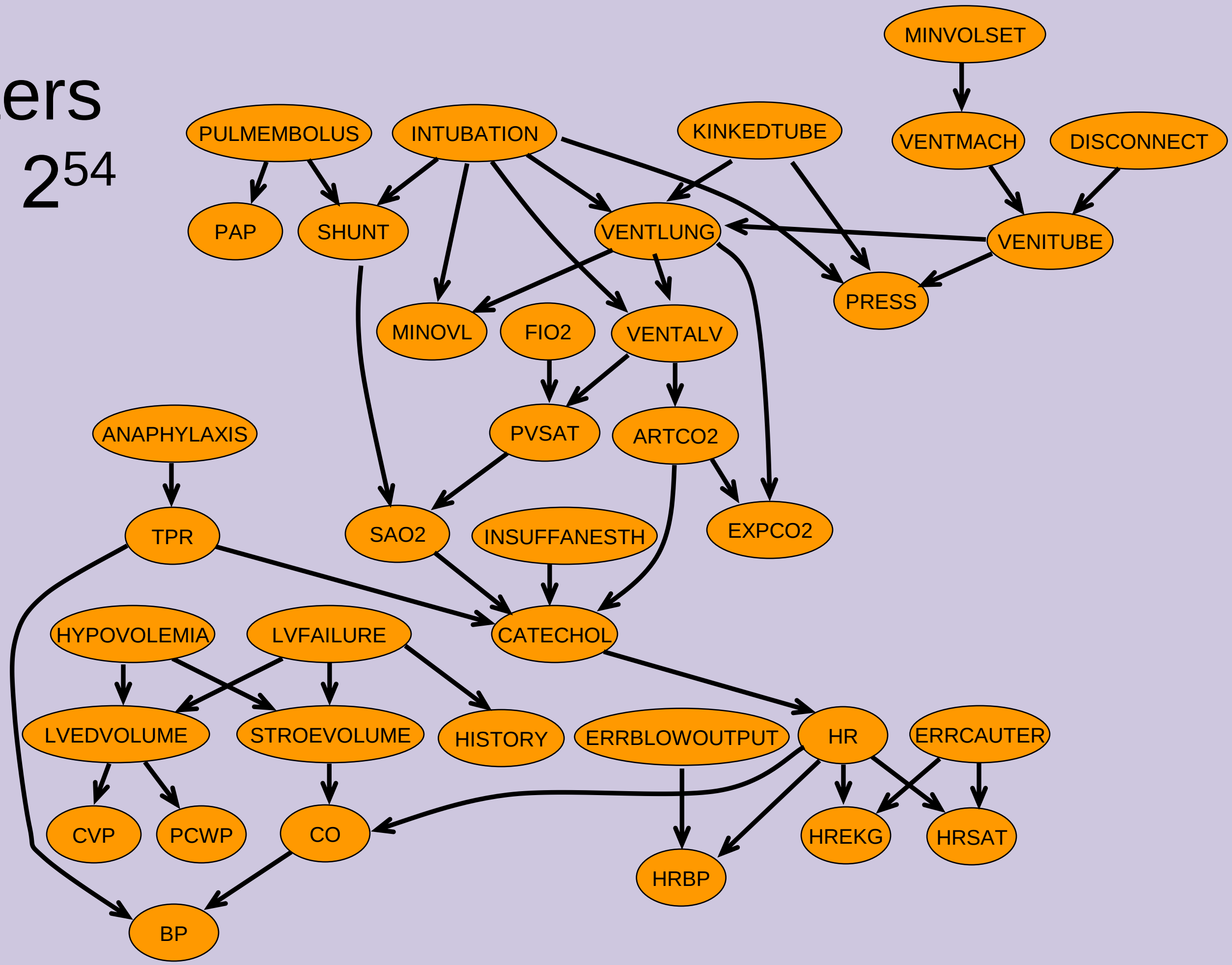
$$\Rightarrow \quad P(\cdot) = \prod_i P(X_i \mid Par_i) = P(B)P(E)P(A \mid B, E)P(R \mid E)P(C \mid A)$$

What are the implications of this?

Example: "ICU Alarm" network

Domain: Monitoring Intensive-Care Patients

- 37 variables
- 509 parameters
  …instead of 2^{54}



Proof

- w.l.o.g. let $X_1,\ldots,X_n$ be an order in which a parent appears before a child (topological ordering)
- assume $X_i \perp \mathrm{NonDesc}_i \mid \mathrm{Par}_i$

$$P(\cdot) = \prod_i P(X_i \mid X_i,\ldots,X_{i-1})$$

chain rule

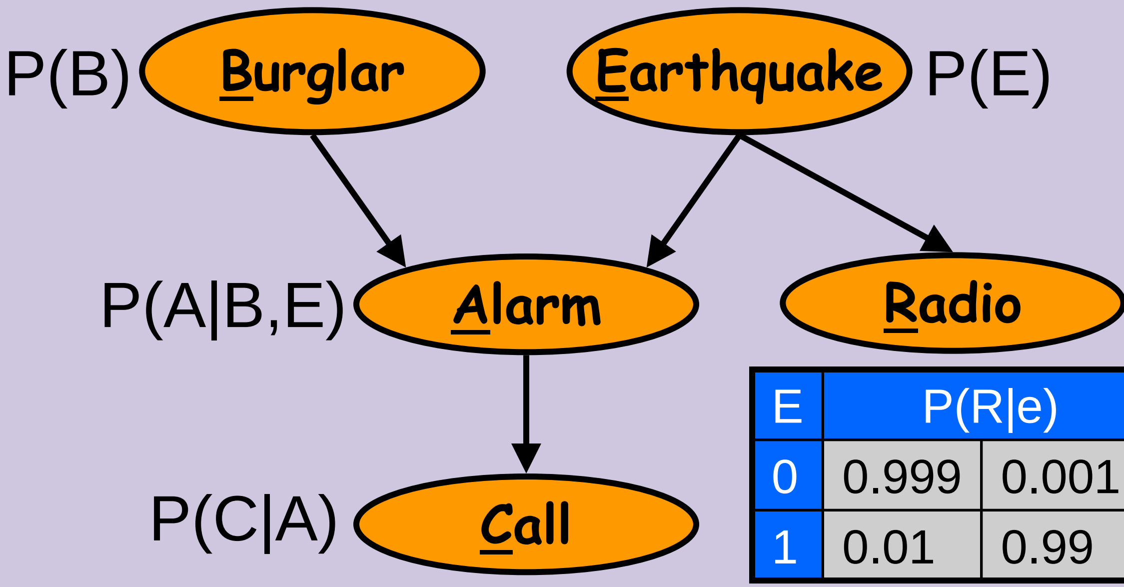
Topological ordering
$S_i \subseteq ND_i$

$$= \prod_i P(X_i \mid Par_i \cup S_i)$$

$$= \prod_i P(X_i \mid Par_i)$$

Independence assumptions

Bayesian Networks



E	P(R|e)	
0	0.999	0.001
1	0.01	0.99

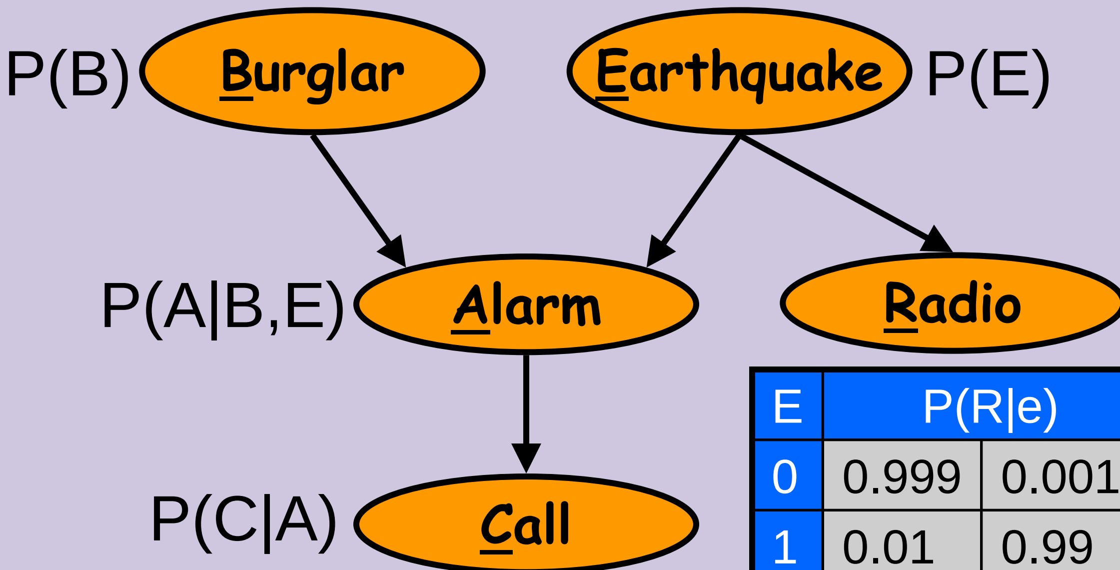
Independence assumptions:

$$X_i \perp \mathrm{NonDesc}_i \mid \mathrm{Par}_i$$

$$\Rightarrow P(\cdot) = \prod_i P(X_i \mid Par_i) = P(B)P(E)P(A \mid B,E)P(R \mid E)P(C \mid A)$$

- ✓ Compact representation of uncertainty
- ✓ Intuitive and interpretable representation
- ✓ Bidirectional inferences (prediction, explanation)
- ✓ Amenable to inference and learning algorithms

Bayesian Networks



E	P(R|e)	
0	0.999	0.001
1	0.01	0.99

Independence assumptions:

$$X_i \perp \mathrm{NonDesc}_i \mid \mathrm{Par}_i$$

$$\Rightarrow P(\cdot) = \prod_i P(X_i \mid Par_i) = P(B)P(E)P(A \mid B, E)P(R \mid E)P(C \mid A)$$

Formalism captures many common models:
mixture/clustering, hierarchical Bayes,
logistic regression, HMMs, factor analysis…

Take Home Problems

1) Assume we are given valid $\left\{ P(X_i \mid Par_i) \right\}$

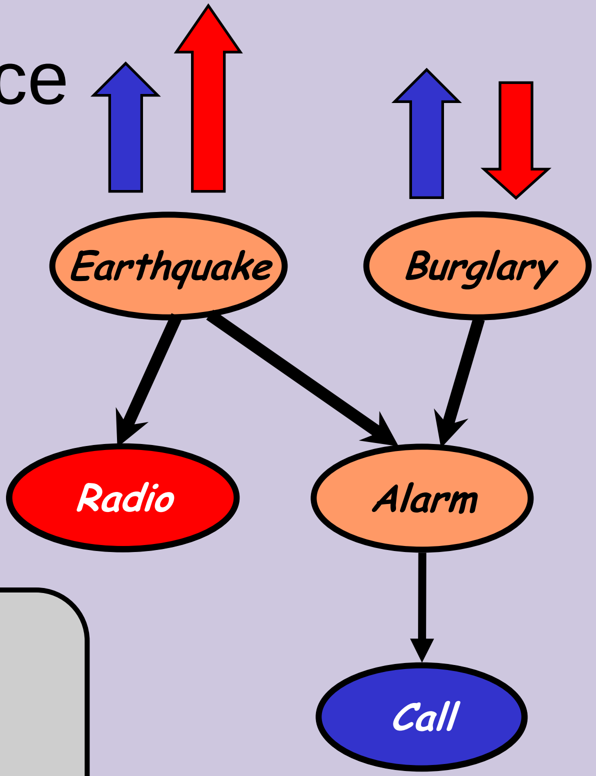
Prove that $P_B(\cdot) = \prod_i P(X_i \mid Par_i)$ is a distribution

Overview

- Introduction
- **Inference**
- Parameter Estimation
- Model Selection

Inference

- **Posterior probabilities**
  - Probability of any event given any evidence
- Most likely explanation
  - Scenario that explains evidence
- Rational decision making
  - Maximize expected utility
  - Value of Information
- Effect of intervention

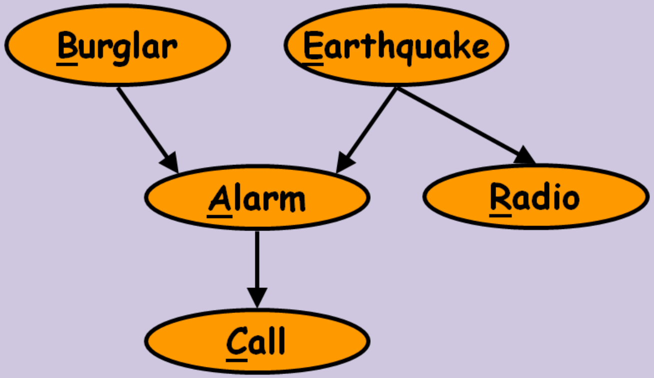


why is this difficult?

$$P(X_i) = \sum_{X_1,\dots,X_{i-1},X_{i+1},X_n} P(X_1,\dots,X_n)$$

Does Decomposition Help?

Let's say we are interested in P(C)



$$P(C) = \sum_{b,e,a,r} P(B)P(E)P(A \mid B,E)P(R \mid E)P(C \mid A)$$

$$= P(C \mid A)\sum_{b} P(B)\sum_{e} P(E)\sum_{a} P(A \mid B,E)\sum_{r} P(R \mid E)$$

Still difficult in general…

Take Home Problems

1) Assume we are given valid $\left\{P(X_i \mid Par_i)\right\}$

Prove that $P_B(\cdot) = \prod_i P(X_i \mid Par_i)$ is a distribution

2) Let $P_B(X_1, \ldots, X_n) = \prod_i P(X_i \mid X_{i-1})$ be a distribution represented by a chain network. How many operations (+,x) are required to compute $P_B(X_n)$ naively? By taking advantage of decomposition?

Overview

- Introduction
- Inference
- **Parameter Estimation**
- Model Selection

Why learn from data?

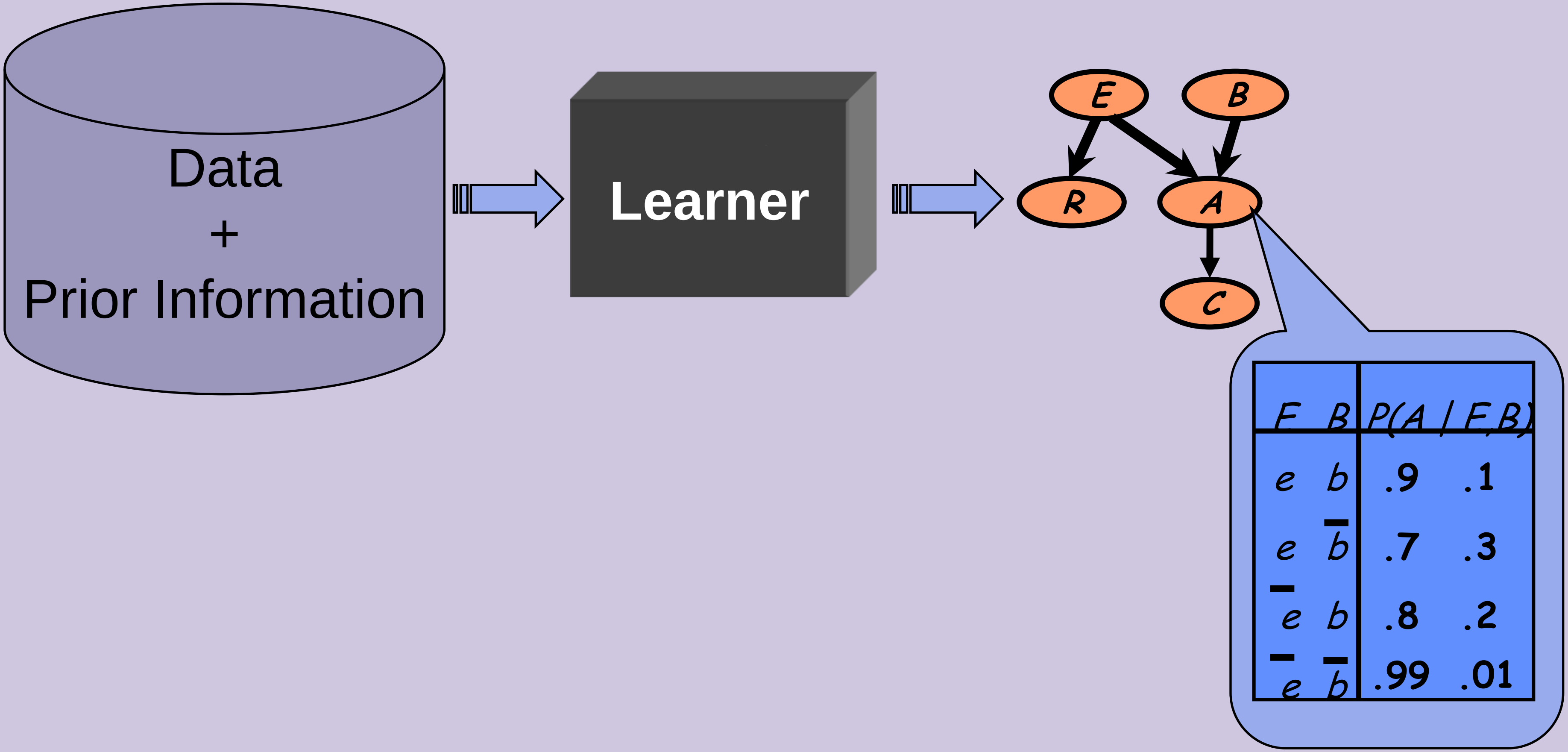
Knowledge acquisition bottleneck

- Knowledge acquisition is an expensive process
- Often we don’t have an expert
- Robust encoding is often quite challenging (hard for humans to estimate global effects)

Data is cheap

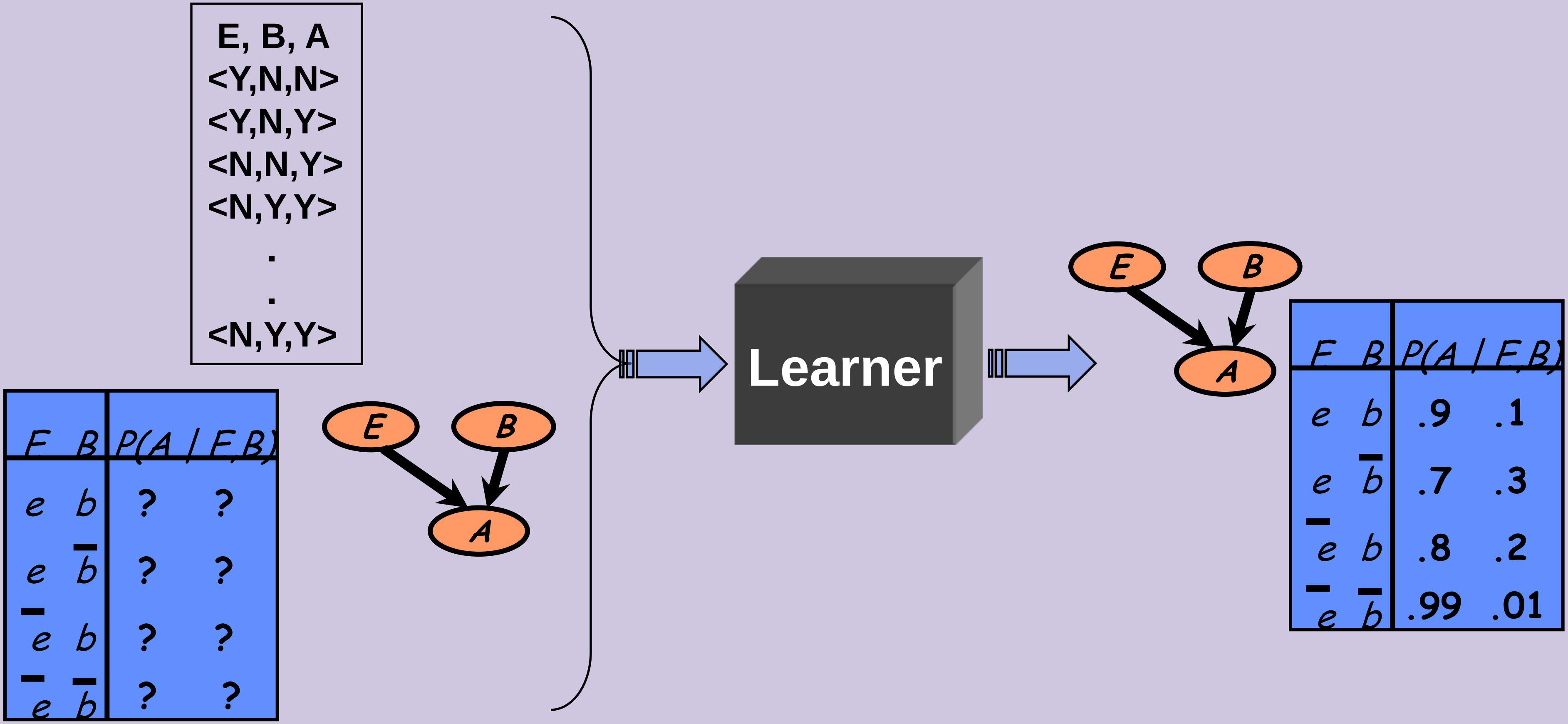
- Amount of available information growing rapidly
- Learning allows us to construct models from raw data

Learning Bayesian networks



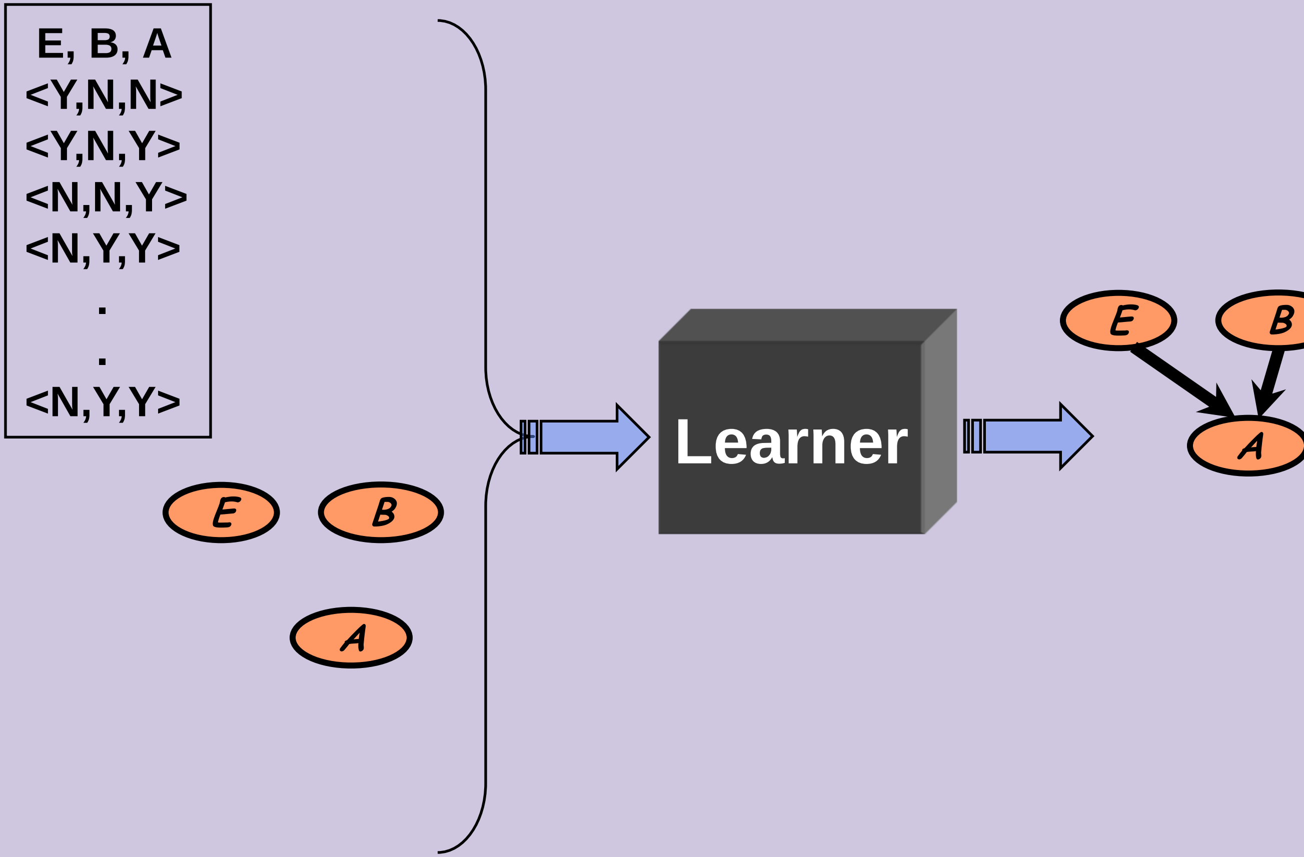
E	B	$P(A \mid E,B)$	
e	b	.9	.1
e	$\bar{b}$	.7	.3
$\bar{e}$	b	.8	.2
$\bar{e}$	$\bar{b}$	.99	.01

Known Structure, Complete Data



- Network structure is specified
  - Inducer needs to estimate parameters
- Data does not contain missing values

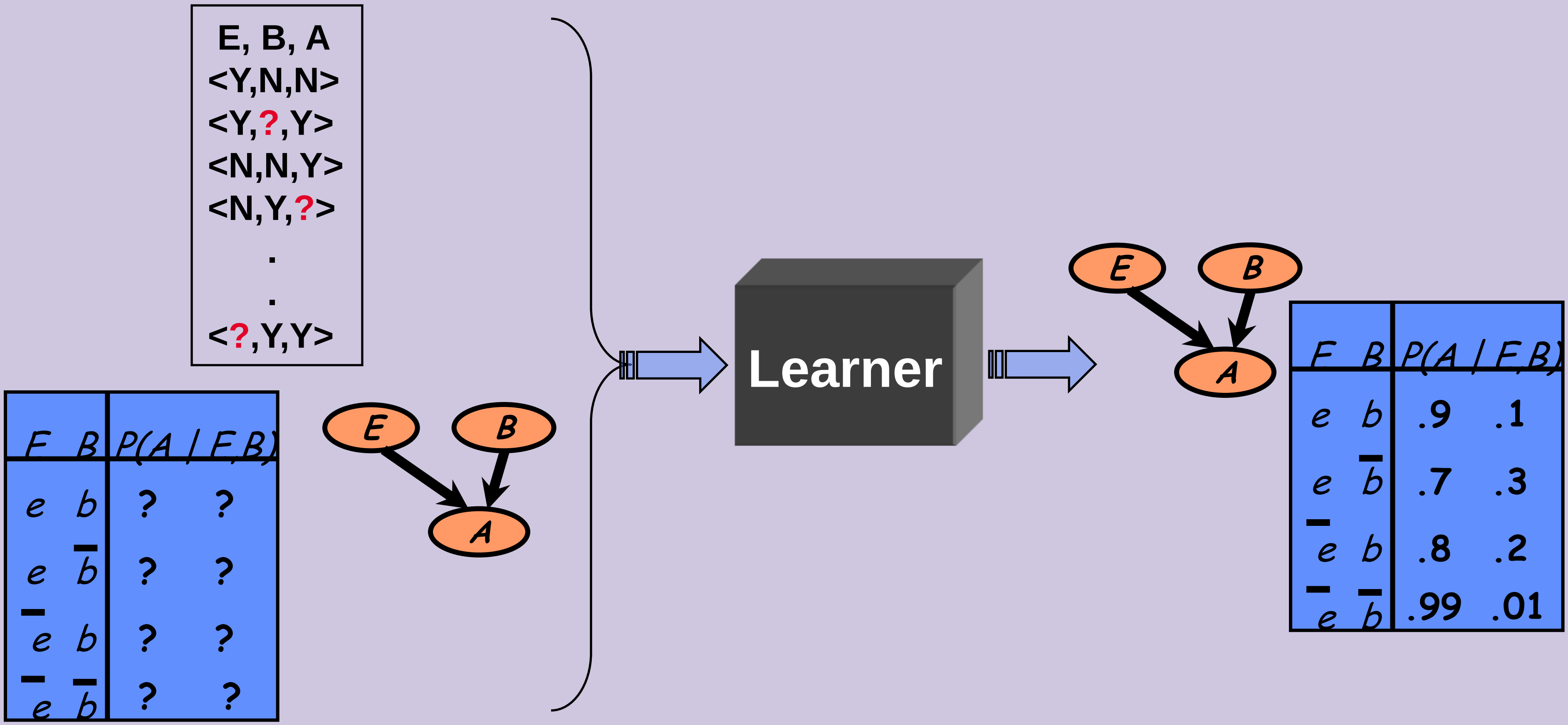
Unknown Structure, Complete Data



E	B	P(A | E,B)	
e	b	.9	.1
e	$\bar{b}$	.7	.3
$\bar{e}$	b	.8	.2
$\bar{e}$	$\bar{b}$	.99	.01

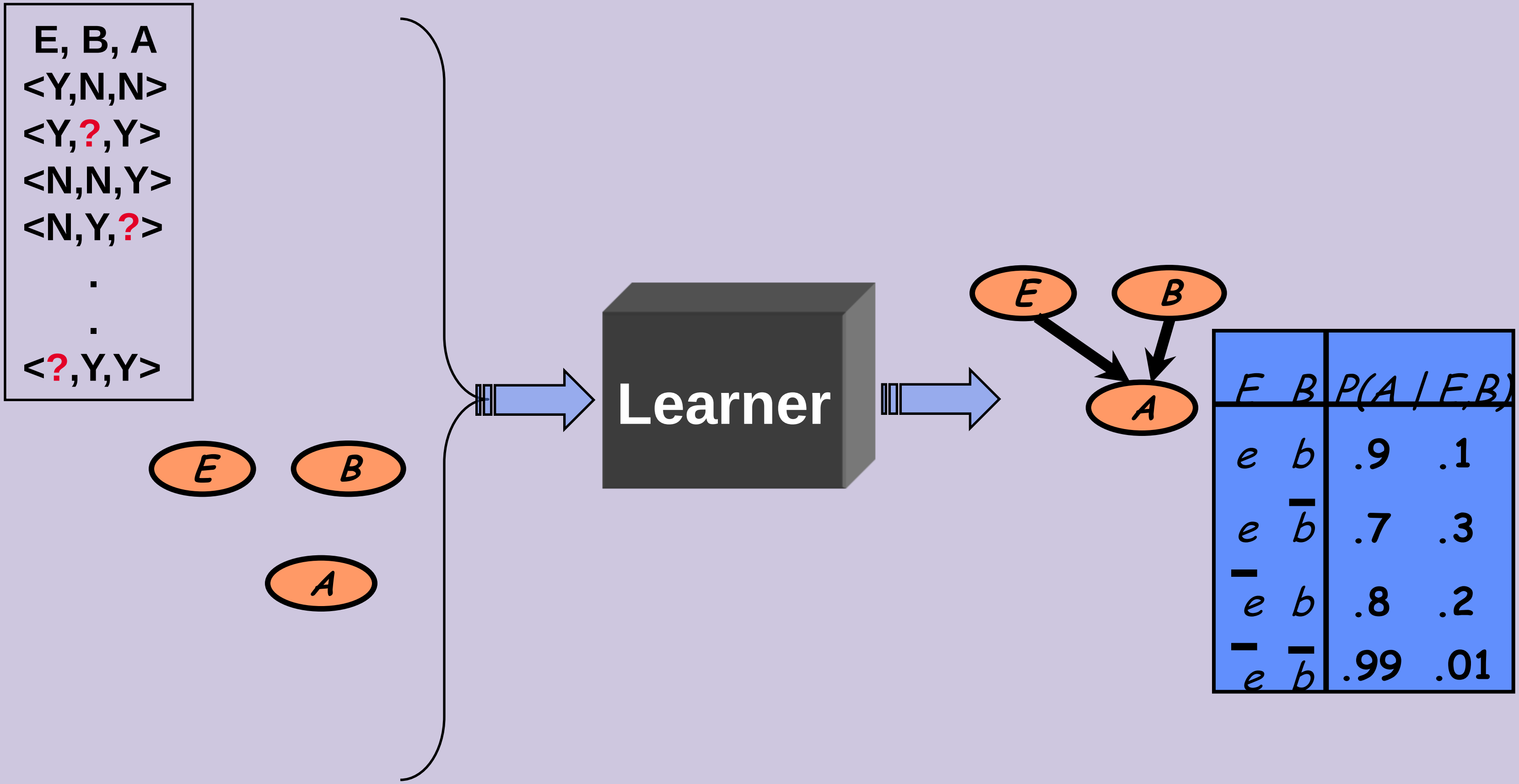
- Network structure is not specified
  - Inducer needs to select arcs & estimate parameters
- Data does not contain missing values

Known Structure, Incomplete Data



- Network structure is specified
- Data contains missing values
  - Need to consider assignments to missing values

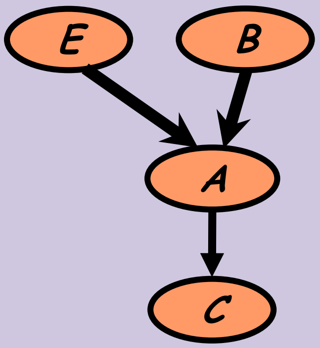
Unknown Structure, Incomplete Data



- Network structure is not specified
- Data contains missing values
  - Need to consider assignments to missing values

Learning Parameters

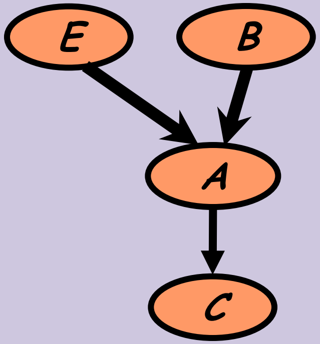
Training data has the form:



$$D = \begin{bmatrix} E[1] & B[1] & A[1] & C[1] \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ E[M] & B[M] & A[M] & C[M] \end{bmatrix}$$

Likelihood Function

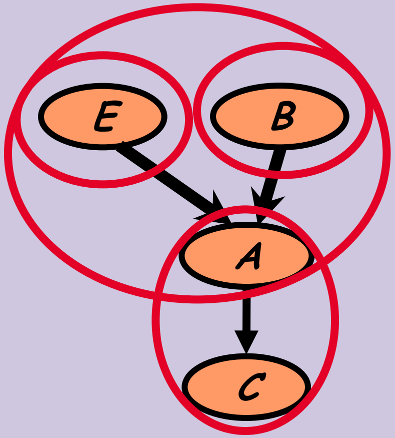
- Assume i.i.d. samples
- Likelihood function is



$$L(\Theta : D) = \prod_m P(E[m], B[m], A[m], C[m] : \Theta)$$

Likelihood Function

By definition of network, we get



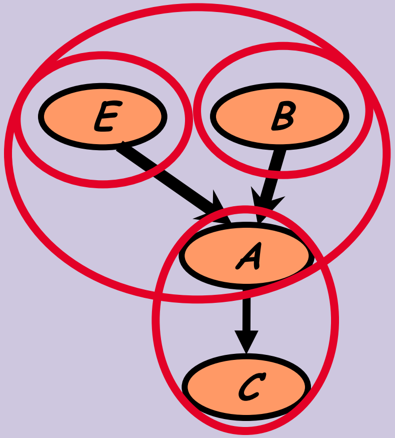
$$L(\Theta : D) = \prod_m P(E[m], B[m], A[m], C[m] : \Theta)$$

$$= \prod_m \begin{pmatrix} P(E[m] : \Theta) \\ P(B[m] : \Theta) \\ P(A[m] \mid B[m], E[m] : \Theta) \\ P(C[m] \mid A[m] : \Theta) \end{pmatrix}$$

$$\begin{bmatrix} E[1] & B[1] & A[1] & C[1] \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ E[M] & B[M] & A[M] & C[M] \end{bmatrix}$$

Likelihood Function

Rewriting terms, we get



$$L(\Theta : D) = \prod_m P(E[m], B[m], A[m], C[m] : \Theta)$$

$$= \begin{array}{l} \prod_m P(E[m] : \Theta) \\ \prod_m P(B[m] : \Theta) \\ \prod_m P(A[m] \mid B[m], E[m] : \Theta) \\ \prod_m P(C[m] \mid A[m] : \Theta) \end{array}$$

$$\begin{bmatrix} E[1] & B[1] & A[1] & C[1] \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ E[M] & B[M] & A[M] & C[M] \end{bmatrix}$$

General Bayesian Networks

Generalizing for any Bayesian network:

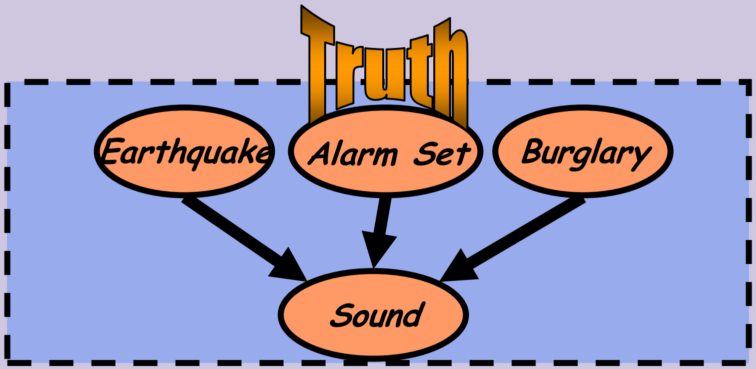
$$
\begin{aligned}
L(\Theta : D) &= \prod_m P(x_1[m], \ldots, x_n[m] : \Theta) \\
&= \prod_i \prod_m P(x_i[m] \mid Pa_i[m] : \Theta_i) \\
&= \prod_i L_i(\Theta_i : D)
\end{aligned}
$$

Now you can use all that you have learned…

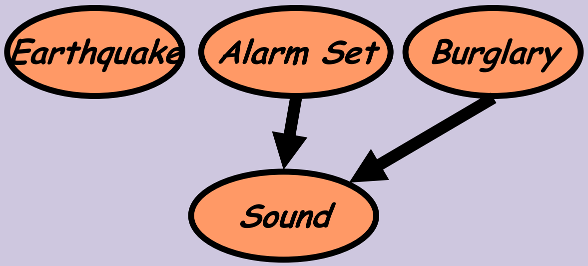
Overview

- Introduction
- Inference
- Parameter Learning
- **Model Selection**
  - Scoring function
  - Structure search

Why Struggle for Accurate Structure?

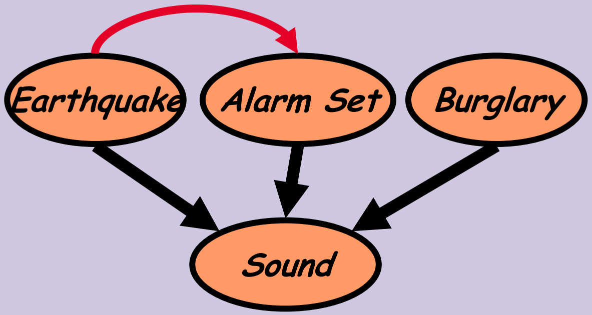


Missing an arc



- Cannot be compensated for by fitting parameters
- Wrong assumptions about domain structure

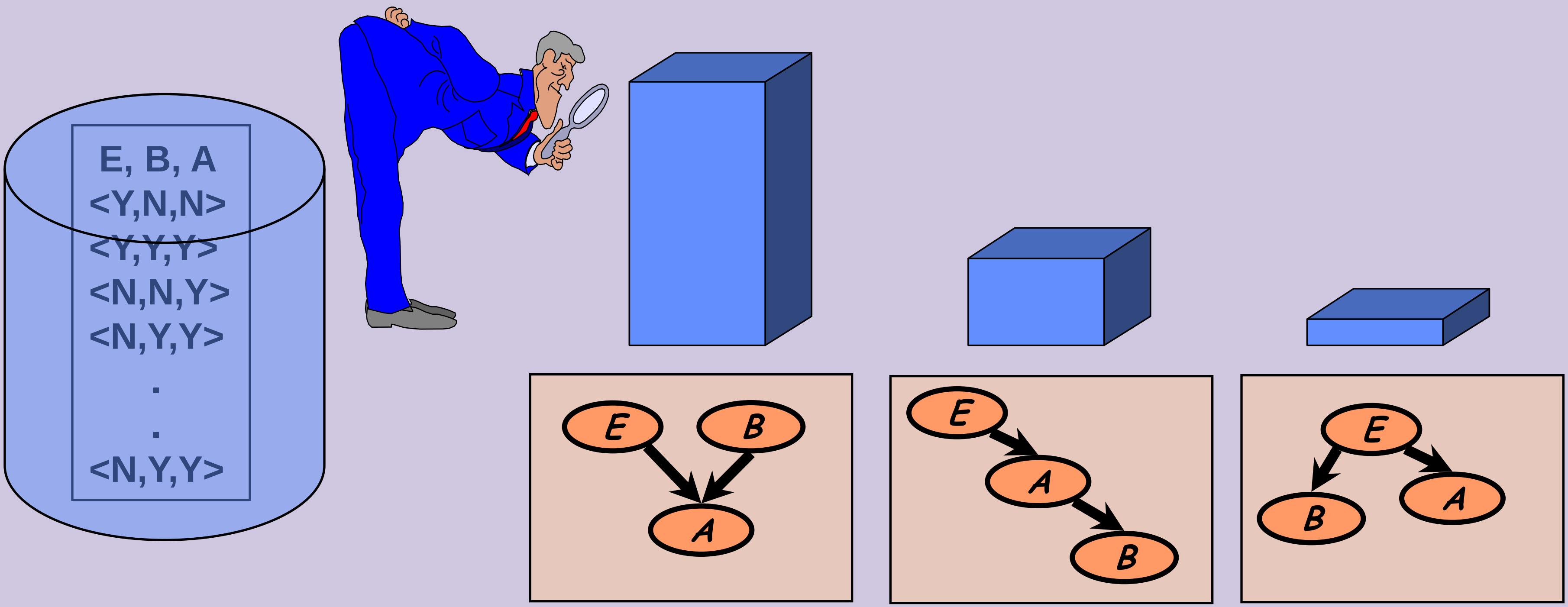
Adding an arc



- Increases the number of parameters to be estimated
- Wrong assumptions about domain structure

Score-based Learning

Define scoring function that evaluates how well a structure matches the data



Search for a structure that maximizes the score

Likelihood Score for Structure

$$\ell(G : D) = \log L(G : D) = M \sum_i \left(I(X_i ; Pa_i^G) - H(X_i) \right)$$

Mutual information between X_i and its parents

- Larger dependence of X_i on $Pa_i \Rightarrow$ higher score

- Adding arcs always helps
  - $I(X; Y) \leq I(X; \{Y,Z\})$
  - Max score attained by fully connected network
  - Overfitting: A bad idea…

Bayesian Score

Likelihood score: $L(G : D) = P(D \mid G, \hat{\theta}_G)$

Max likelihood params

Bayesian approach:

Deal with uncertainty by assigning probability to all possibilities

$$P(D \mid G) = \int P(D \mid G, \theta) P(\theta \mid G) d\theta$$

Marginal Likelihood

Likelihood

Prior over parameters

Bayesian Score

Likelihood score: $L(G:D) = P(D \mid G, \hat{\theta}_G)$

Max likelihood params

Bayesian approach:

Deal with uncertainty by assigning probability to all possibilities

$$P(D \mid G) = \int P(D \mid G, \theta) P(\theta \mid G) d\theta$$

Fortunately, in many cases integral has closed form.

Asymptotically we get:

$$\log P(D \mid G) = \ell(G:D) - \frac{\log M}{2} dim(G) + O(1)$$

Fit empirical distribution

Complexity penalty

Structure Search as Optimization

Input:

- Training data
- Scoring function
- Set of possible structures

Output:

- A network that maximizes the score

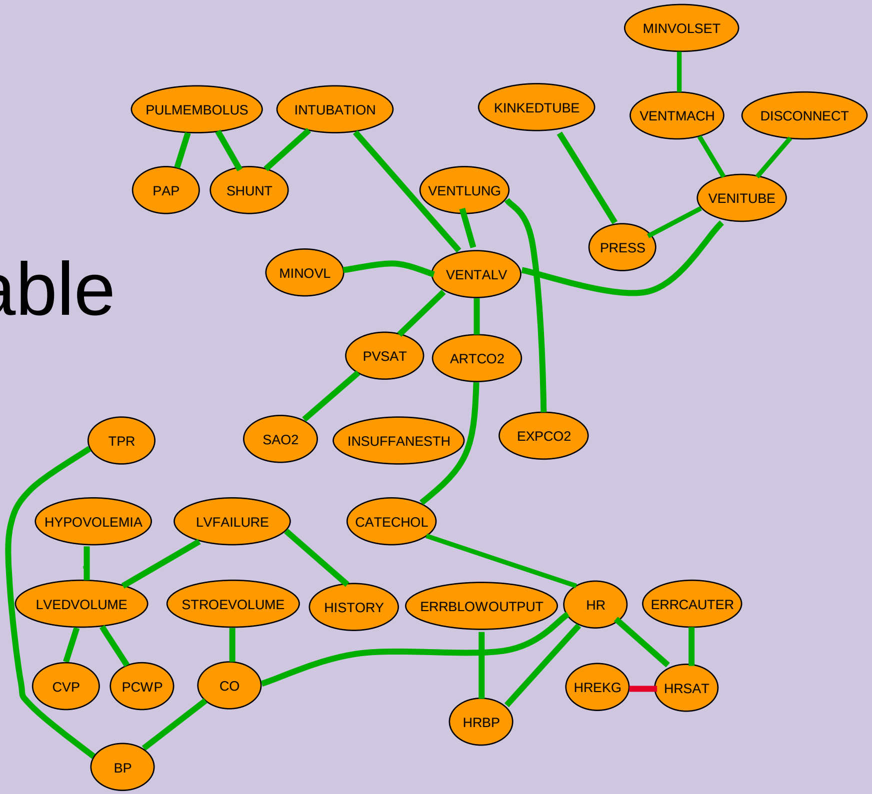
Key Computational Property: Decomposability:

$$\text{score}(G) = \sum \text{score (family of X in G)}$$

Tree-Structured Networks

Trees:

At most one parent per variable



Why trees?

- Elegant math

  ⇒ we can solve the optimization problem

- Sparse parameterization

  ⇒ avoid over-fitting

Learning Trees

- Let *p(i)* denote parent of X_i
- We can write the Bayesian score as

$$
\begin{aligned}
Score(G:D) &= \sum_i \log P(X_i : Pa_i) - Pen_i \\
&= \sum_i Score(X_i : Pa_i) \\
&= \sum_i \left(Score(X_i : X_{p(i)}) - Score(X_i) \right) + \sum_i Score(X_i)
\end{aligned}
$$

Improvement over "empty" network

Score of "empty" network

Score = sum of edge scores + constant

⇨ Can find the optimal tree using max-spanning tree algorithm

Beyond Trees

Essentially everything else is computationally difficult:

- Learning the optimal chain is NP-hard (exponential in the number of variables)
- Learning the optimal poly-tree is NP-hard
- Learning the optimal Bayesian network with at most k parents per node is NP-hard for k>1

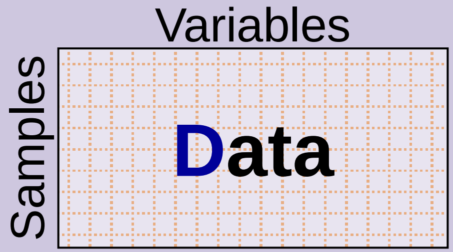
→ This is where computer science comes in…

Heuristic Search

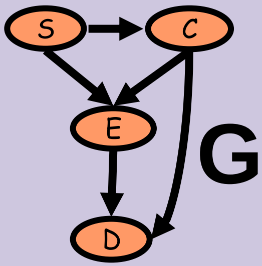
- Define a search space:
  - search states are possible structures
  - operators make small changes to structure
- Traverse space looking for high-scoring structures
- Search techniques:
  - Greedy hill-climbing
  - Best first search
  - Annealing
  - ...

Hypothesis Exploration (Local Search)

Input:



Output:

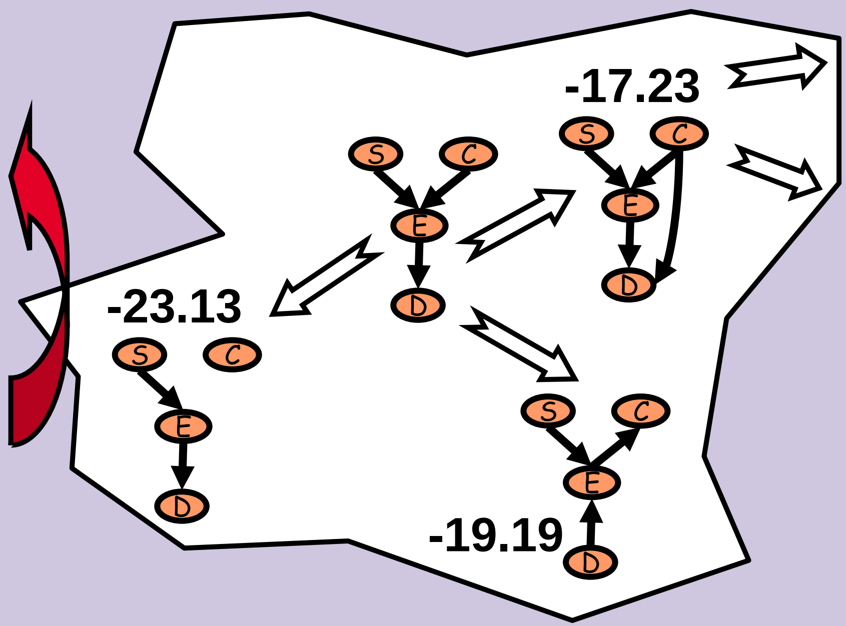


Init: Start with initial structure

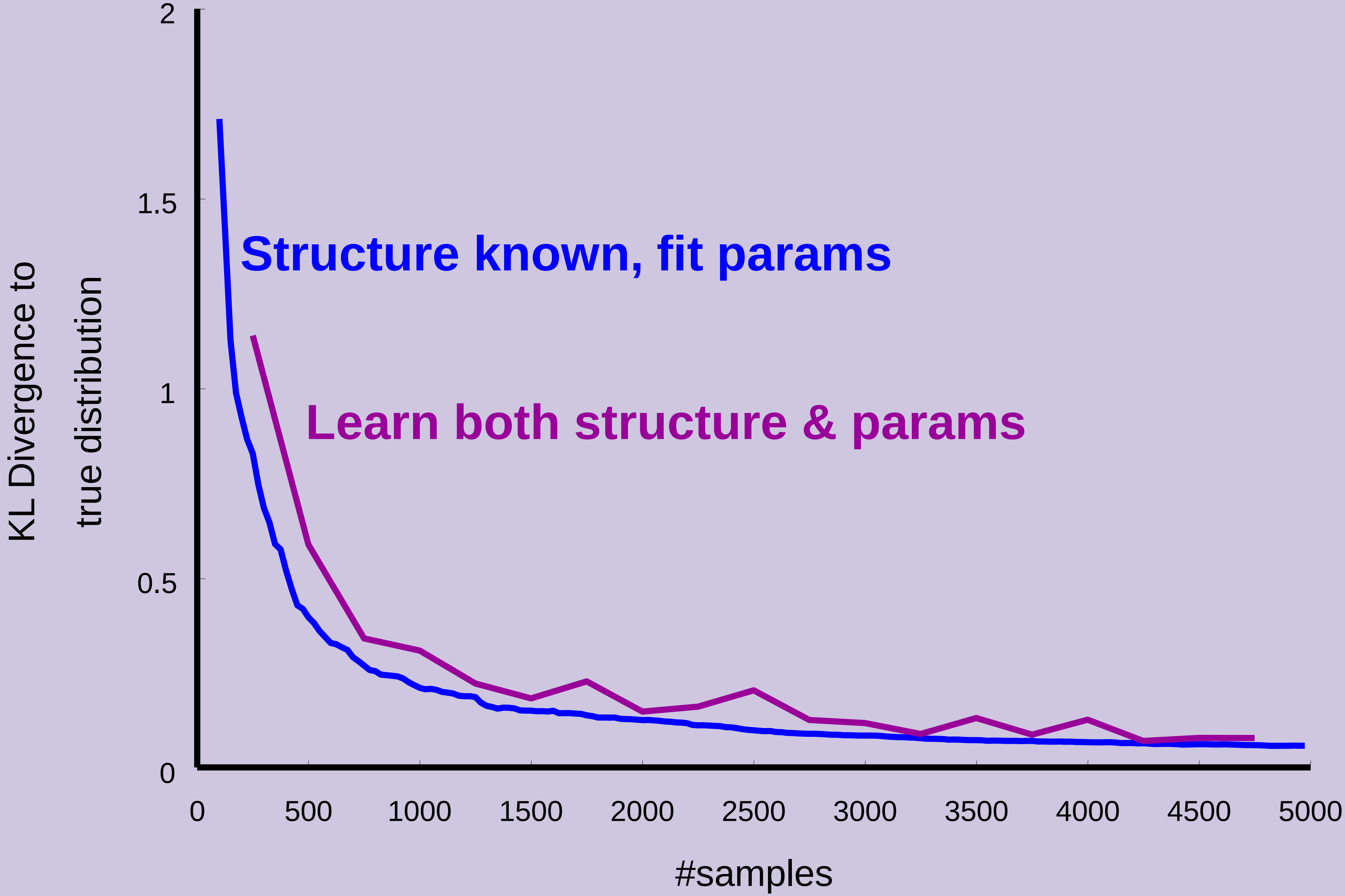
1 Consider local changes

2 Score each candidate

3 Apply best modification



Learning in Practice: Alarm domain



Local Search: Possible Pitfalls

- Local search can get stuck in:
  - **Local Maxima:**
    - All one-edge changes reduce the score
  - **Plateau:**
    - Some one-edge changes leave the score unchanged

- Standard heuristics can escape both
  - Random restarts
  - TABU search
  - Simulated annealing

Structure Search: Summary

- Discrete optimization problem
- In some cases, optimization problem is easy
  - Example: learning trees
- In general, NP-Hard
  - Need to resort to heuristic search
  - In practice, search is relatively fast (~100 vars in ~2-5 min):
    - Decomposability
    - Sufficient statistics
  - Adding randomness to search is critical

Take Home Problems

1) Assume we are given valid $\{P(X_i \mid Par_i)\}$

Prove that $P_B(\cdot) = \prod_i P(X_i \mid Par_i)$ is a distribution

2) Let $P_B(X_1,\ldots,X_n) = \prod_i P(X_i \mid X_{i-1})$ be a distribution represented by a chain network. How many operations (+,x) are required to compute $P_B(X_n)$ naively? By taking advantage of decomposition?

3) When adding/deleting an edge in the search we need to compute the score of the resulting graph. Explain precisely how does decomposability helps in this computation? What if we reverse an edge?

Graphical Models

Goal: make a joint distribution more amenable

Goodman (1970), Haberman (1974)
decomposable models of a
log-llinear joint distribution

Sewall Wright (1921)
directed graphs for
causal models, SEM

Lauritzen (1978)
combine probability
and graph theory

Pearl (1988)

Markov Networks:
structured
log-linear models

Bayesian Networks:
decomposition into
conditional probabilities

Conclusion

- Many distributions have a dependency structure
- Utilizing this structure is good
- Discovering this structure has implications:
  - To density estimation
  - To knowledge discovery
  - To marginal computations
- Many applications
  - Medicine
  - Biology
  - Web
  - …

Conclusion

- Many distributions have a dependency structure
- Utilizing this structure is good
- Discovering this structure has implications:
  - To density estimation
  - To knowledge discovery

Statistics and Learning

on the continuum of data analysis techniques

- **Common goals and similar challenges**
- **Both rely on probability theory**
- **Different methodologies (a plus!)**