Stat 521A
Lecture 21

Outline

- Overview of structure learning
- Constraint based approach (18.2)
- Scoring functions (18.3)

Overview of structure learning

- Goals: density estimation and knowledge discovery
- Can only learn graph up to Markov equivalence
- 2 main approaches:
- Find PDAG which is an I-map of the empirical distribution, using conditional independence test (eg \chi^2) at the 5% level in lieu of oracle
- Find MAP DAG by defining a scoring and search through DAG space
- Can also do Bayes model averaging over DAGs to get posterior of features of interest eg predictive density, edge/path marginals, etc

Assumptions behind constraint based

- Each node has a fan-in of at most d
- We have a CI oracle $X \perp Y \mid Z$ that gives correct results for conditioning sets up to size 2d+2
- P* is faithful to G*
- Def 3.3.4. A distribution P is faithful to G if, whenever $X \perp Y \mid Z$ in I(P), we have dsep_G(X;Y|Z)

  i.e., there are no “non-graphical” independencies buried in the parameters

Deriving graphs from distributions

- Sec 3.4, from Lecture 2
- So far, we have discussed how to derive distributions from graphs.
- But how do we get the DAG?
- Assume we have access to the true distribution P, and can answer questions of the form

$$P \models X \perp Y | Z$$

- For finite data samples, we can approximate this oracle with a CI test – the frequentist approach to graph structure learning (see ch 18)
- What DAG can be used to represent P?

Minimal I-map

- The complete DAG is an I-map for any distribution (since it encodes no CI relations)
- Def 3.4.1. A graph K is a minimal I-map for a set of independencies I if it is an I-map for I, and if the removal of even a single edge from K renders it not an I-map.
- To derive a minimal I-map, we pick an arbitrary node ordering, and then find some minimal subset U to be X_i's parents, where

$$X_i \perp \{X_1, \ldots, X_{i-1}\} \setminus U | U$$

- (K2 algorithm replace this CI test with a Bayesian scoring metric: sec 18.4.2).

Constructing I-map given ordering

Algorithm 3.2 Procedure to build a minimal I-map given an ordering

```
Procedure Build-Minimal-I-Map (
  $X_1, \ldots, X_n$    // an ordering of random variables in $\mathcal{X}$
  $\mathcal{I}$    // Set of independencies
)
  Set $\mathcal{G}$ to an empty graph over $\mathcal{X}$
  for $i = 1, \ldots, n$
    $U \leftarrow \{X_1, \ldots, X_{i-1}\}$    // $U$ is the current candidate for parents of $X_i$
    for $U' \subseteq \{X_1, \ldots, X_{i-1}\}$
      if $U' \subset U$ and $(X_i \perp \{X_1, \ldots, X_{i-1}\} - U' \mid U') \in \mathcal{I}$ then
        $U \leftarrow U'$
      // At this stage $U$ is a minimal set satisfying $(X_i \perp$
         $\{X_1, \ldots, X_{i-1}\} - U \mid U)$
      // Now set $U$ to be the parents of $X_i$
    for $X_j \in U$
      Add $X_j \rightarrow X_i$ to $\mathcal{G}$
  return $\mathcal{G}$
```

Effect of node ordering

- “Bad” node orderings can result in dense, unintuitive graphs.
- Eg L,S,G,I,D. Add L. Add S: must add L as parent, since $P \not\models L \perp S$ Add G: must add L,S as parents.

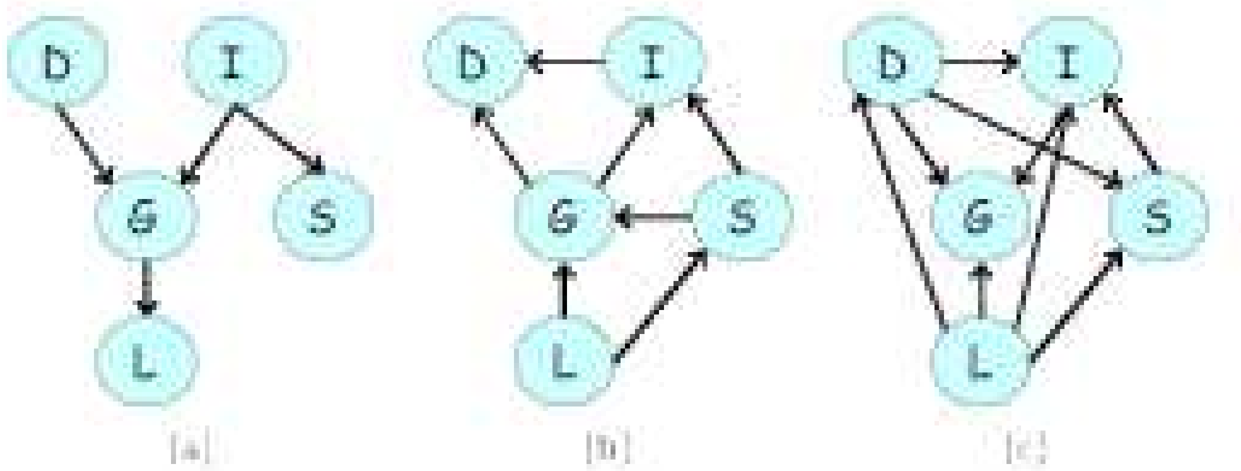


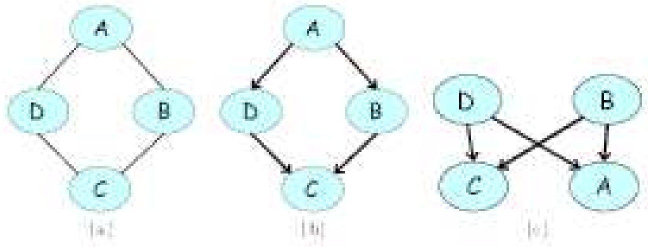
Figure 3.8: Three minimal I-maps for $P_{student}$, induced by different orderings: (a) D, I, S, G, L (b) L, S, G, I, D (c) L, D, S, I, G

Dealing with node ordering

- Search over orders
- Work with PDAGs

Perfect maps

- Minimal I-maps can have superfluous edges.
- Def 3.4.2. Graph K is a perfect map for a set of independencies I if I(K)=I. K is a perfect map for P if I(K)=I(P).
- Not all distributions can be perfectly represented by a DAG.
- Eg let Z = xor(X,Y) and use some independent prior on X, Y. Minimal I-map is X -> Z <- Y. However, X $\perp$ Z in I(P), but not in I(G).
- Eg. A $\perp$ C | {B,D} and B $\perp$ D | {A,C}, A dep | B,C, etc



Finding perfect maps

- If P has a perfect map, we can find it in polynomial time, using an oracle for the CI tests.
- We can only identify the graph up to I-equivalence, so we return the PDAG that represents the corresponding equivalence class.
- The method has 3 steps (see sec 3.4.3)
  - Identify undirected skeleton
  - Identify immoralities
  - Compute eclass (compelled edges)
- This algorithm has been used to claim one can infer causal models from observational data, but this claim is controversial

Algorithm due to Verma & Pearl 1991, Spirtes, Glymour, Scheines 1993, Meek 1995

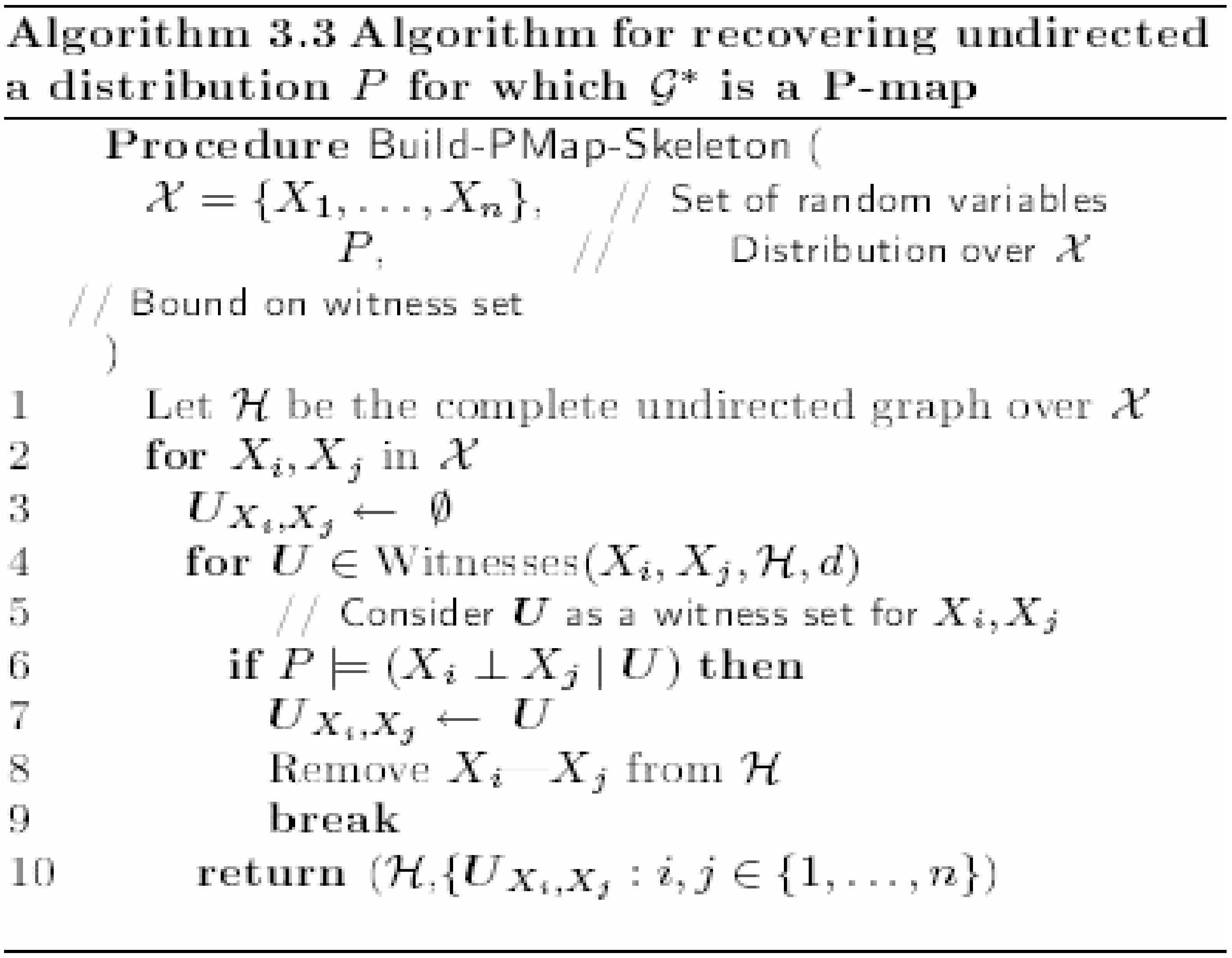
Identifying the undirected skeleton

- Initially connect all node pairs
- Remove an edge if we find a U st Xi ⊥ Xj | U

Lemma 3.4.8: *Let $\mathcal{G}^*$ be an I-map of a distribution P, and let X and Y be two variables that are not adjacent in $\mathcal{G}^*$. Then either $P \models (X \perp Y \mid \mathrm{Pa}_X^{\mathcal{G}^*})$ or $P \models (X \perp Y \mid \mathrm{Pa}_Y^{\mathcal{G}^*})$.*

- Hence we can restrict our search for witnesses U to the sets $U \subseteq \mathcal{X} - \{X_i, X_j\} - \mathrm{Nb}_{X_i}^{\mathcal{H}}$,
  and $U \subseteq \mathcal{X} - \{X_i, X_j\} - \mathrm{Nb}_{X_j}^{\mathcal{H}}$.

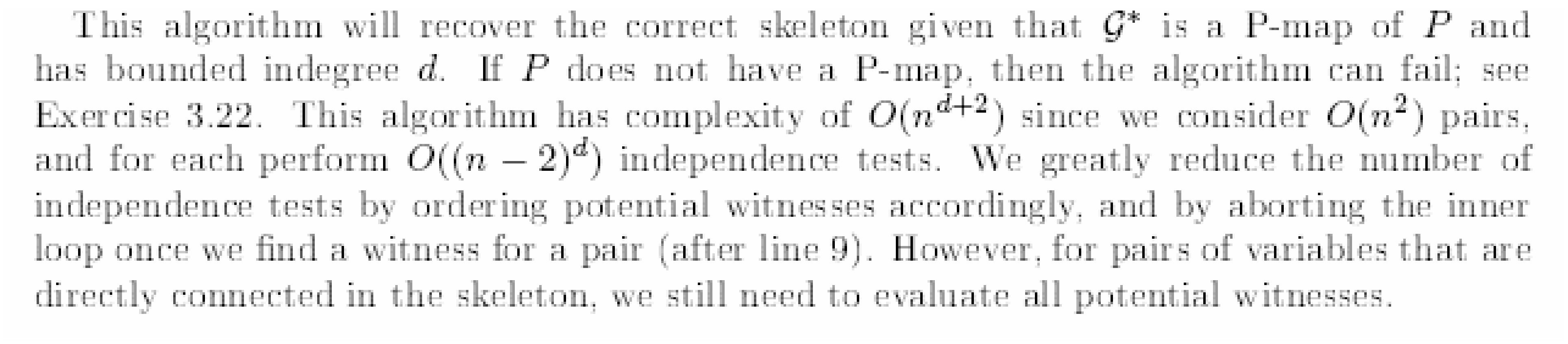
Identifying the undirected skeleton



Algorithm 3.3 Algorithm for recovering undirected a distribution P for which $\mathcal{G}^*$ is a P-map

```
Procedure Build-PMap-Skeleton (
   $\mathcal{X} = \{X_1, \ldots, X_n\}$,   // Set of random variables
   $P$,                                     // Distribution over $\mathcal{X}$
 // Bound on witness set
 )
   Let $\mathcal{H}$ be the complete undirected graph over $\mathcal{X}$
   for $X_i, X_j$ in $\mathcal{X}$
     $U_{X_i,X_j} \leftarrow \emptyset$
     for $U \in$ Witnesses($X_i, X_j, \mathcal{H}, d$)
         // Consider $U$ as a witness set for $X_i, X_j$
       if $P \models (X_i \perp X_j \mid U)$ then
         $U_{X_i,X_j} \leftarrow U$
         Remove $X_i — X_j$ from $\mathcal{H}$
         break
    return ($\mathcal{H}$, $\{U_{X_i,X_j} : i, j \in \{1, \ldots, n\}\}$)
```

Complexity



This algorithm will recover the correct skeleton given that $\mathcal{G}^*$ is a P-map of P and has bounded indegree d. If P does not have a P-map, then the algorithm can fail; see Exercise 3.22. This algorithm has complexity of $O(n^{d+2})$ since we consider $O(n^2)$ pairs, and for each perform $O((n-2)^d)$ independence tests. We greatly reduce the number of independence tests by ordering potential witnesses accordingly, and by aborting the inner loop once we find a witness for a pair (after line 9). However, for pairs of variables that are directly connected in the skeleton, we still need to evaluate all potential witnesses.

Identifying immoralities

Proposition 3.4.9: *Let $\mathcal{G}^*$ be a P-map of a distribution P, and let X, Y and Z be variables that form an immorality $X \rightarrow Z \leftarrow Y$. Then, $P \not\models (X \perp Y \mid \boldsymbol{U})$ for any set $\boldsymbol{U}$ that contains Z.*

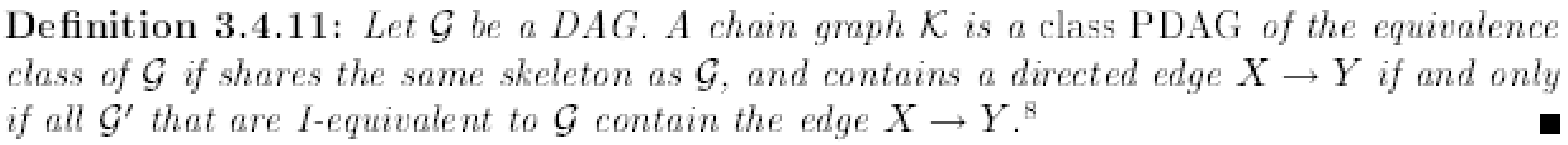
Proposition 3.4.10: *Let $\mathcal{G}^*$ be a P-map of a distribution P, and let the triplet X, Y, Z be a potential immorality in the skeleton of $\mathcal{G}^*$, such that $X \rightarrow Z \leftarrow Y$ is not in $\mathcal{G}^*$. If $\boldsymbol{U}$ is such that $P \models (X \perp Y \mid \boldsymbol{U})$, then $Z \in \boldsymbol{U}$.*

Combining these two results, we see that a potential immorality $X—Z—Y$ is an immorality if and only if Z is not in the witness set(s) for X and Y. That is, if $X—Z—Y$ is an immorality, then Proposition 3.4.9 shows that Z is not in any witness set $\boldsymbol{U}$; conversely, if $X—Z—Y$ is not an immorality, the Z must be in every witness set $\boldsymbol{U}$. Thus, we can use the specific witness set $\boldsymbol{U}_{X,Y}$ that we recorded for X, Y in order to determine whether this triplet is an immorality or not: we simply check whether $Z \in \boldsymbol{U}_{X,Y}$. If $Z \notin \boldsymbol{U}_{X,Y}$, then we declare the triplet an immorality. Otherwise, we declare that it is not an immorality. The Mark-Immoralities procedure shown in Algorithm 3.4 summarizes this process.

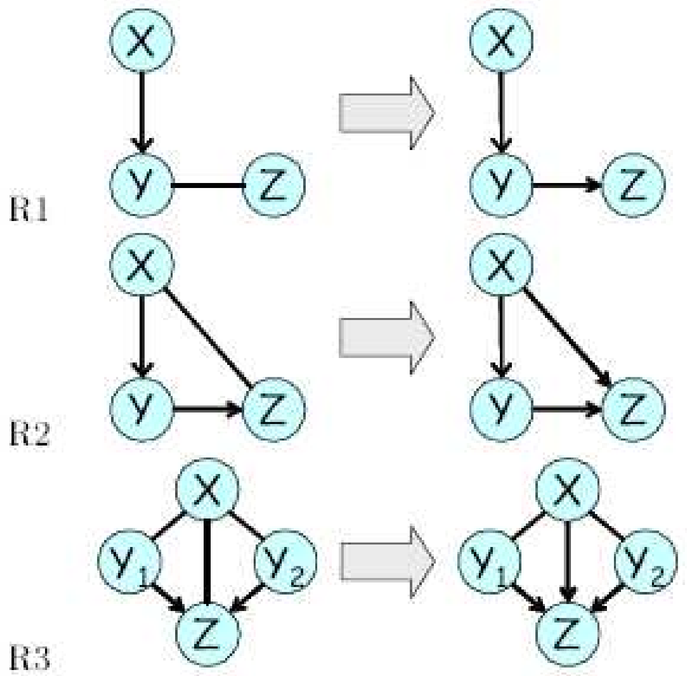
```
    )
$\mathcal{K} \leftarrow S$
for $X_i, X_j, X_k$ such that $X_i—X_j—X_k \in S$ and $X_i—X_k \notin S$
    // $X_i—X_j—X_k$ is a potential immorality
  if $X_j \notin \boldsymbol{U}_{X_i,X_k}$ then
      Add the orientations $X_i \rightarrow X_j$ and $X_j \leftarrow X_k$ to $\mathcal{K}$
return $\mathcal{K}$
```

Compute PDAG

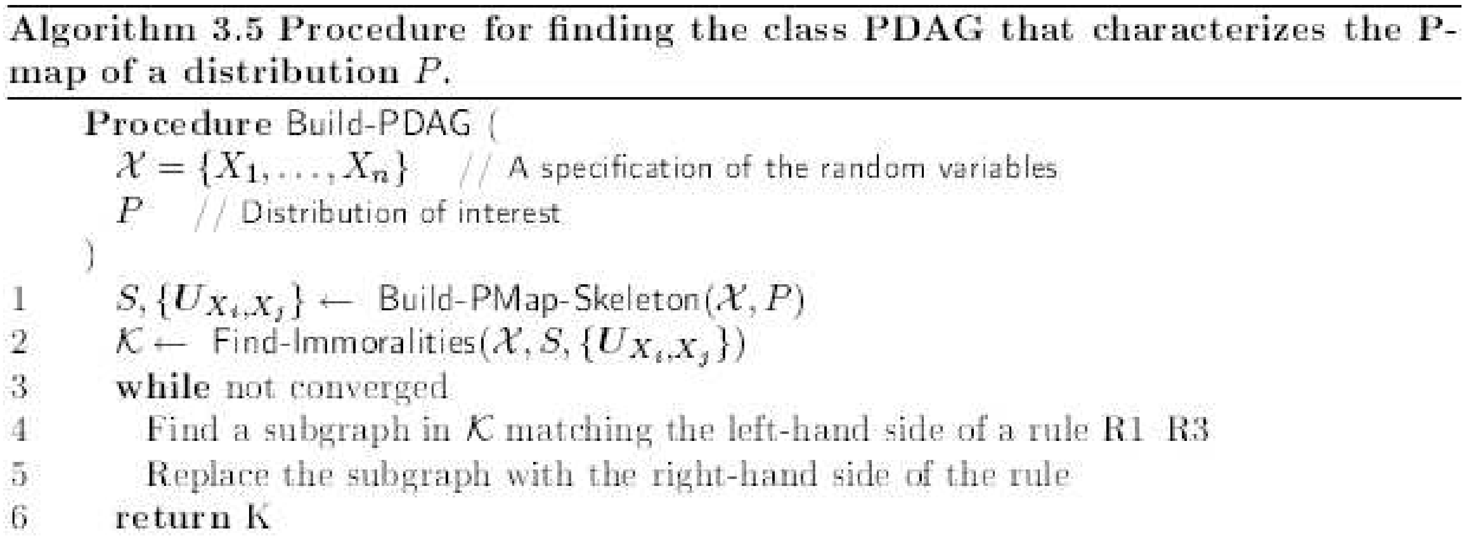
- Skeleton plus immoralities defines equiv class
- But we might want to orient as many edges as possible, not just those in immoralities



Definition 3.4.11: *Let $\mathcal{G}$ be a DAG. A chain graph $\mathcal{K}$ is a* class PDAG *of the equivalence class of $\mathcal{G}$ if shares the same skeleton as $\mathcal{G}$, and contains a directed edge $X \rightarrow Y$ if and only if all $\mathcal{G}'$ that are I-equivalent to $\mathcal{G}$ contain the edge $X \rightarrow Y$.*[8] ■

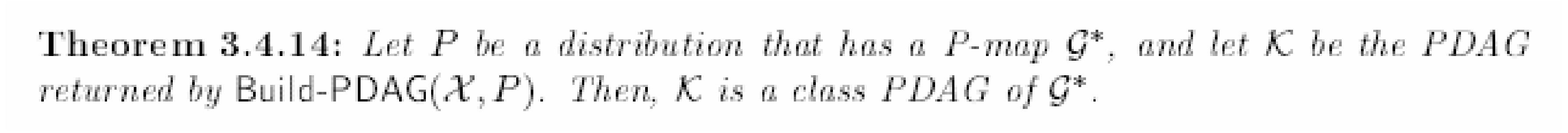


Overall PC algorithm



Algorithm 3.5 **Procedure for finding the class PDAG that characterizes the P-map of a distribution P.**

```
Procedure Build-PDAG (
    $\mathcal{X} = \{X_1, \ldots, X_n\}$   // A specification of the random variables
    $P$   // Distribution of interest
)
1    $S, \{U_{X_i,X_j}\} \leftarrow$ Build-PMap-Skeleton($\mathcal{X}, P$)
2    $\mathcal{K} \leftarrow$ Find-Immoralities($\mathcal{X}, S, \{U_{X_i,X_j}\}$)
3    while not converged
4      Find a subgraph in $\mathcal{K}$ matching the left-hand side of a rule R1 R3
5      Replace the subgraph with the right-hand side of the rule
6    return K
```



Theorem 3.4.14: *Let P be a distribution that has a P-map $\mathcal{G}^*$, and let $\mathcal{K}$ be the PDAG returned by* Build-PDAG($\mathcal{X}, P$). *Then, $\mathcal{K}$ is a class PDAG of $\mathcal{G}^*$.*

n=#nodes, d=fanin, complexity = O(n^{d+2})
One error in a CI test can propagate through whole structure – not robust
Can choose thresholds to control the FDR

Recent developments

Kalisch, M. and Bühlmann, P. (2007). Estimating high-dimensional directed acyclic graphs with the PC-algorithm. Journal of Machine Learning Research 8, 613-636.
[Proves uniform consistency in the Gaussian case]

Kalisch, M. and Bühlmann, P. (2008). Robustification of the PC-algorithm for directed acyclic graphs. Journal of Computational and Graphical Statistics 17, 773-789.
[Uses robust estimate of covariance matrix]

Maathuis, M.H., Kalisch, M. and Bühlmann, P. (2008). Estimating high-dimensional intervention effects from observational data. To appear in the Annals of Statistics.
[Causal DAGs]

Bühlmann, P., Kalisch, M. and Maathuis, M.H. (2009). Variable selection for high-dimensional models: partially faithful distributions and the PC-simple algorithm.
[Lasso-type methods]

Score functions

- We can treat model selection as an optimization problem: arg max score(G,D)
- ML score: $\mathrm{score}_L(\mathcal{G} : \mathcal{D}) = \ell(\langle \mathcal{G}, \hat{\theta}_{\mathcal{G}} \rangle : \mathcal{D})$
- Obviously this will prefer the fully connected graph
- But if we limit the fan-in (eg restrict attention to simple trees), this can be useful

ML score and Mutual information

- Consider G0: X, Y and G1: X->Y

$$\text{score}_L(\mathcal{G}_0 : \mathcal{D}) = \sum_m \log \hat{\theta}_{x[m]} + \log \hat{\theta}_{y[m]}$$

$$\text{score}_L(\mathcal{G}_1 : \mathcal{D}) = \sum_m \log \hat{\theta}_{x[m]} + \log \hat{\theta}_{y[m]|x[m]}$$

$$\text{score}_L(\mathcal{G}_1 : \mathcal{D}) - \text{score}_L(\mathcal{G}_0 : \mathcal{D}) = \sum_m \log \hat{\theta}_{y[m]|x[m]} - \log \hat{\theta}_{y[m]}$$

$$\text{score}_L(\mathcal{G}_1 : \mathcal{D}) - \text{score}_L(\mathcal{G}_0 : \mathcal{D}) = \sum_{x,y} M[x,y] \log \hat{\theta}_{y|x} - \sum_y M[y] \log \hat{\theta}_y$$

$$\text{score}_L(\mathcal{G}_1 : \mathcal{D}) - \text{score}_L(\mathcal{G}_0 : \mathcal{D}) = M \sum_{x,y} \hat{P}(x,y) \log \frac{\hat{P}(y \mid x)}{\hat{P}(y)} = M \cdot \boldsymbol{I}_{\hat{P}}(X;Y)$$

Proposition 18.3.1: *The likelihood score decomposes as follows:*

$$\text{score}_L(\mathcal{G} : \mathcal{D}) = M \sum_{i=1}^{n} \boldsymbol{I}_{\hat{P}}(X_i; \text{Pa}^{\mathcal{G}}_{X_i}) - M \sum_{i=1}^{n} \boldsymbol{H}_{\hat{P}}(X_i)$$

Bayesian score

Defined as log marginal likelihood plus log prior
Log p(G) is constant whereas log p(D|G) grows linearly with nsamples
Log p(D|G) offers automatic complexity control – Bayesian Occam's razor

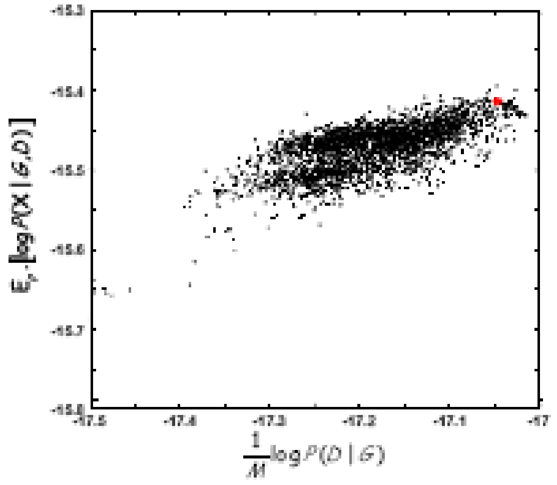
$$\mathrm{score}_B(\mathcal{G} \;:\; \mathcal{D}) = \log P(\mathcal{D} \mid \mathcal{G}) + \log P(\mathcal{G})$$

$$P(\mathcal{D} \mid \mathcal{G}) = \int_{\Theta_{\mathcal{G}}} P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) P(\theta_{\mathcal{G}} \mid \mathcal{G}) d\theta_{\mathcal{G}}$$

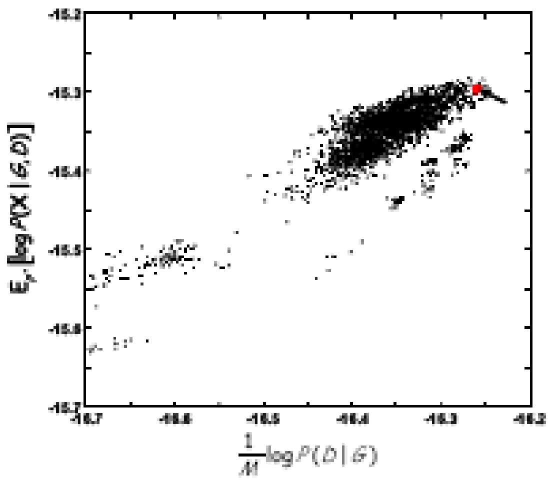
$$P(\mathcal{D} \mid \mathcal{G}) = \prod_{m=1}^{M} P(\xi[m] \mid \xi[1], \ldots, \xi[m-1], \mathcal{G})$$

$$\frac{1}{M} \log P(\mathcal{D} \mid \mathcal{G}) \approx E_{P^*}[\log P(\mathcal{X} \mid \mathcal{G}, \mathcal{D})]$$

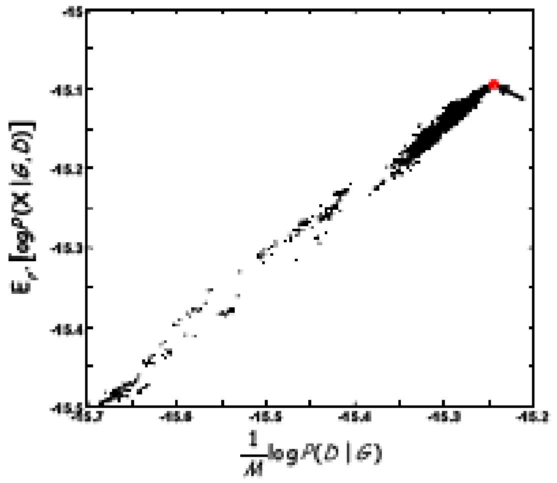
Expected log pred lik vs avg log marg lik



500 instances



1000 instances



10000 instances

Computation of marginal likelihood

- For a Dirichlet-multinomial we have

$$P(x[1], \ldots, x[M]) = \frac{\Gamma(\alpha)}{\Gamma(\alpha + M)} \cdot \prod_{i=1}^{k} \frac{\Gamma(\alpha_i + M[x^i])}{\Gamma(\alpha_i)}.$$

- For a DAG X->Y we have

$$\begin{aligned} P(\mathcal{D} \mid \mathcal{G}_{X \to Y}) \quad = \quad & \left(\int_{\Theta_X} P(\theta_X \mid \mathcal{G}_{X \to Y}) \prod_m P(x[m] \mid \theta_X, \mathcal{G}_{X \to Y}) d\theta_X \right) \\ & \left(\int_{\Theta_{Y \mid x^0}} P(\theta_{Y \mid x^0} \mid \mathcal{G}_{X \to Y}) \prod_{m : x[m] = x^0} P(y[m] \mid \theta_{Y \mid x^0}, \mathcal{G}_{X \to Y}) d\theta_{Y \mid x^0} \right) \\ & \left(\int_{\Theta_{Y \mid x^1}} P(\theta_{Y \mid x^1} \mid \mathcal{G}_{X \to Y}) \prod_{m : x[m] = x^1} P(y[m] \mid \theta_{Y \mid x^1}, \mathcal{G}_{X \to Y}) d\theta_{Y \mid x^1} \right) \end{aligned}$$

- For CPTs with dirichlet priors:BDe score

$$P(\mathcal{D} \mid \mathcal{G}) = \prod_i \prod_{\boldsymbol{u}_i \in Val(\mathrm{Pa}^{\mathcal{G}}_{X_i})} \frac{\Gamma(\alpha^{\mathcal{G}}_{X_i \mid \boldsymbol{u}_i})}{\Gamma(\alpha^{\mathcal{G}}_{X_i \mid \boldsymbol{u}_i} + M[\boldsymbol{u}_i])} \prod_{x_i^j \in Val(X_i)} \left[\frac{\Gamma(\alpha^{\mathcal{G}}_{x_i^j \mid \boldsymbol{u}_i} + M[x_i^j, \boldsymbol{u}_i])}{\Gamma(\alpha^{\mathcal{G}}_{x_i^j \mid \boldsymbol{u}_i})} \right]$$

Asymptotic approximations to Bayesian score

- We have

Theorem 18.3.4: *If we use a Dirichlet parameter prior for all parameters in our network, then, as $M \to \infty$, we have that:*

$$\log P(\mathcal{D} \mid \mathcal{G}) = \ell(\hat{\theta}_{\mathcal{G}} : \mathcal{D}) - \frac{\log M}{2}\mathrm{Dim}[\mathcal{G}] + O(1)$$

where $\mathrm{Dim}[\mathcal{G}]$ *is the number of independent parameters in* $\mathcal{G}$.

$$\mathrm{score}_{BIC}(\mathcal{G} \ : \ \mathcal{D}) = \ell(\hat{\theta}_{\mathcal{G}} : \mathcal{D}) - \frac{\log M}{2}\mathrm{Dim}[\mathcal{G}]$$

$$\mathrm{score}_{BIC}(\mathcal{G} \ : \ \mathcal{D}) = M\sum_{i=1}^{n} \boldsymbol{I}_{\hat{P}}(X_i; \mathrm{Pa}_{X_i}) - M\sum_{i=1}^{n} \boldsymbol{H}_{\hat{P}}(X_i) - \frac{\log M}{2}\mathrm{Dim}[\mathcal{G}]$$

MDL = BIC

Thm 18.3.6. BIC, MDL and Bayesian score are consistent (so score(G)=score(G*) iff G is I-equivlent to G*)

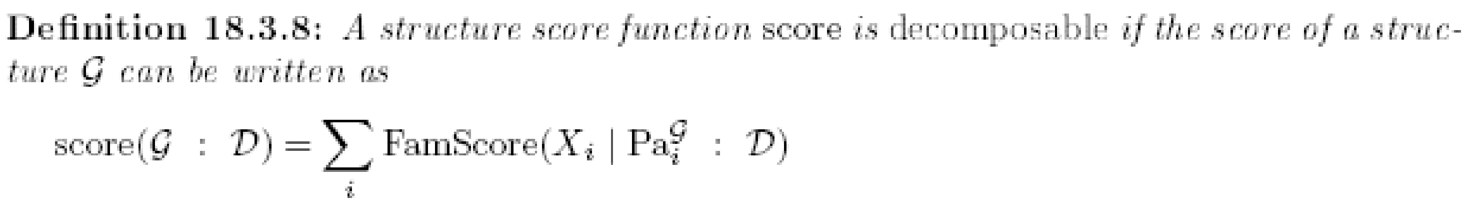
Structure priors

- P(G) only matters in small sample setting

- Penalized number of edges $P(\mathcal{G}) \propto c^{|\mathcal{G}|}$
- Penalize deviation from fixed prior structure

Decomposable score

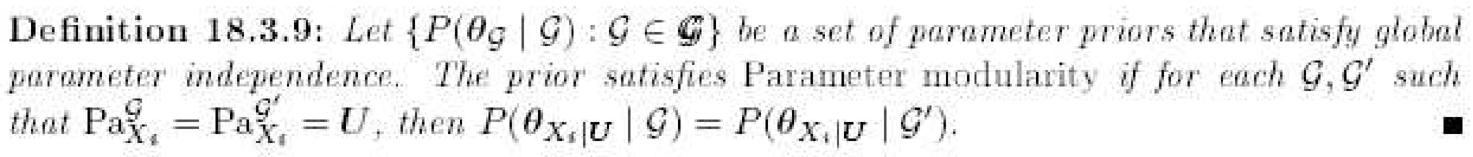
- When we make local changes to a graph, we want to evaluate the score change in constant time



Definition 18.3.8: *A structure score function* score *is* decomposable *if the score of a structure $\mathcal{G}$ can be written as*

$$\text{score}(\mathcal{G} : \mathcal{D}) = \sum_i \text{FamScore}(X_i \mid \text{Pa}_i^{\mathcal{G}} : \mathcal{D})$$

- BIC score is decomposable



Definition 18.3.9: *Let $\{P(\theta_{\mathcal{G}} \mid \mathcal{G}) : \mathcal{G} \in \mathbb{G}\}$ be a set of parameter priors that satisfy global parameter independence. The prior satisfies* Parameter modularity *if for each $\mathcal{G}, \mathcal{G}'$ such that $\text{Pa}_{X_i}^{\mathcal{G}} = \text{Pa}_{X_i}^{\mathcal{G}'} = \boldsymbol{U}$, then $P(\theta_{X_i \mid \boldsymbol{U}} \mid \mathcal{G}) = P(\theta_{X_i \mid \boldsymbol{U}} \mid \mathcal{G}')$.* ■

- Thm 18.3.10. parameter modularity => BDe score is decomposable
- Defn: Structural modularity if p(G) decomposes
- Thm 18.3.10. param & struct modularity => Bayesian score decomposable

Score equivalence

- Def 18.3.11. Score() is score equiv if score(G)=score(G') if G, G' are I-equiv
- Thm 18.3.12. Likelihood and BIC scores are score equiv.
- BDe score is only score equivalent if we set the Dirichlet hyper-parameters as follows

$$\alpha_{x_i|\mathrm{pa}_i} = \alpha \cdot P'(x_i, \mathrm{pa}_i).$$

[illegible]

- Eg if P' is a uniform prior network, then

$$\begin{aligned} \theta_{ijk} &\overset{\text{def}}{=} p(X_i = k | X_{\pi_i} = j) \\ \theta_{ijk} &\sim \text{Dir}(\alpha_{ijk}) \\ \alpha_{ijk} &= \alpha \frac{1}{q_i r_i} \end{aligned}$$

$\alpha_{\iota\varphi\kappa}$=1 (K2 prior) is not score equiv

thetaY ~ Dir(1,1) ESS 2

$\left.\begin{array}{l}\text{thetaY|X=1 ~ Dir(1,1)} \\ \text{thetaY|X=0 ~ Dir(1,1)}\end{array}\right)$ ESS 4

Decomposable score

- When we make local changes to a graph, we want to evaluate the score change in constant time

Definition 18.3.8: *A structure score function* score *is* decomposable *if the score of a structure $\mathcal{G}$ can be written as*

$$\mathrm{score}(\mathcal{G} : \mathcal{D}) = \sum_i \mathrm{FamScore}(X_i \mid \mathrm{Pa}_i^{\mathcal{G}} : \mathcal{D})$$

- BIC score is decomposable
- We say a prior satisfies structural modularity if

$$P(\mathcal{G}) \propto \prod_i P(\mathrm{Pa}_{X_i} = \mathrm{Pa}_{X_i}^{\mathcal{G}})$$

Definition 18.3.9: *Let $\{P(\theta_{\mathcal{G}} \mid \mathcal{G}) : \mathcal{G} \in \mathbb{G}\}$ be a set of parameter priors that satisfy global parameter independence. The prior satisfies* Parameter modularity *if for each $\mathcal{G}, \mathcal{G}'$ such that $\mathrm{Pa}_{X_i}^{\mathcal{G}} = \mathrm{Pa}_{X_i}^{\mathcal{G}'} = \boldsymbol{U}$, then $P(\theta_{X_i \mid \boldsymbol{U}} \mid \mathcal{G}) = P(\theta_{X_i \mid \boldsymbol{U}} \mid \mathcal{G}')$.* ■

- Thm 18.3.10. Structural & parameter modularity => Bayesian score is decomposable