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 $dsep_G(X;Y\mid Z)$
i.e., there are no “non-graphical” independencies buried in the parameters
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

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

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

Procedure Build-Minimal-I-Map (X_1, \ldots, X_n \quad \text{// an ordering of random variables in } \mathcal{X}
\mathcal{I} \quad \text{// Set of independencies}
)

1. Set \mathcal{G} to an empty graph over \mathcal{X}
2. for i = 1, \ldots, n
3. \quad U \leftarrow \{X_1, \ldots, X_{i-1}\} \quad \text{// } U \text{ is the current candidate for parents of } X_i
4. \quad \text{for } U' \subseteq \{X_1, \ldots, X_{i-1}\}
5. \quad \quad \text{if } U' \subseteq U \text{ and } (X_i \perp \{X_1, \ldots, X_{i-1}\} - U' \mid U') \in \mathcal{I} \text{ then}
6. \quad \quad \quad U \leftarrow U'
7. \quad \quad \text{// At this stage } U \text{ is a minimal set satisfying } (X_i \perp \{X_1, \ldots, X_{i-1}\} - U \mid U)
8. \quad \quad \text{// Now set } U \text{ to be the parents of } X_i
9. \quad \text{for } X_j \in U
10. \quad \quad \text{Add } X_j \rightarrow X_i \text{ to } \mathcal{G}
11. \quad \text{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 $P \not\perp L \perp S$. Add G: must add L,S as parents.

Figure 8.8: Three minimal I-maps for $P_{visdist}$, 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 \( Z = \text{xor}(X,Y) \) and use some independent prior on \( X, Y \). Minimal I-map is \( X \rightarrow Z \leftarrow Y \). However, \( X \perp Z \) in \( I(P) \), but not in \( I(G) \).
• Eg. \( A \perp C \mid \{B,D\} \) and \( B \perp D \mid \{A,C\} \), A dep \( \mid B,C \), 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

Identifying the undirected skeleton

- Initially connect all node pairs
- Remove an edge if we find a $U$ st $X_i \perp X_j | U$

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

- Hence we can restrict our search for witnesses $U$ to the sets and

\[
U \subseteq \mathcal{X} - \{X_i, X_j\} - \text{Nb}^{\mathcal{H}}_{X_i},
\]

\[
U \subseteq \mathcal{X} - \{X_i, X_j\} - \text{Nb}^{\mathcal{H}}_{X_j}.
\]
Algorithm 3.3 Algorithm for recovering undirected a distribution $P$ for which $G^*$ is a P-map

**Procedure** Build-PMap-Skeleton (  

$\mathcal{X} = \{X_1, \ldots, X_n\}$,  

$P$,  

// Set of random variables  

// Distribution over $\mathcal{X}$

// Bound on witness set

1. Let $\mathcal{H}$ be the complete undirected graph over $\mathcal{X}$
2. for $X_i, X_j$ in $\mathcal{X}$
3. \quad $U_{X_i, X_j} \leftarrow \emptyset$
4.   for $U \in \text{Witnesses}(X_i, X_j, \mathcal{H}, d)$
5.     // Consider $U$ as a witness set for $X_i, X_j$
6.       if $P \models (X_i \perp X_j \mid U)$ then
7.         \quad $U_{X_i, X_j} \leftarrow U$
8.       Remove $X_i \! - \! X_j$ from $\mathcal{H}$
9.       break
10.  return $(\mathcal{H}, \{U_{X_i, X_j} : i, j \in \{1, \ldots, n\}\}$

This algorithm will recover the correct skeleton given that $G^*$ 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 $G^*$ 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, $P \not\models (X \perp Y \mid U)$ for any set $U$ that contains $Z$.

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

Combining these two results, we see that a potential immorality $X \leftarrow Z \rightarrow Y$ is an immorality if and only if $Z$ is not in the witness set(s) for $X$ and $Y$. That is, if $X \leftarrow Z \rightarrow Y$ is an immorality, then Proposition 3.4.9 shows that $Z$ is not in any witness set $U$; conversely, if $X \leftarrow Z \rightarrow Y$ is not an immorality, the $Z$ must be in every witness set $U$. Thus, we can use the specific witness set $U_{X,Y}$ that we recorded for $X, Y$ in order to determine whether this triplet is an immorality or not: we simply check whether $Z \in U_{X,Y}$. If $Z \not\in U_{X,Y}$, 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.

```
1 \textbf{K} \leftarrow S
2 \textbf{for } X_i, X_j, X_k \textbf{ such that } X_i \leftarrow X_j \leftarrow X_k \in S \textbf{ and } X_i \leftarrow X_k \not\in S
3 \quad \text{// } X_i \leftarrow X_j \leftarrow X_k \text{ is a potential immorality}
4 \quad \textbf{if } X_j \not\in U_{X_i, X_k} \textbf{ then}
5 \quad \quad \text{Add the orientations } X_i \rightarrow X_j \text{ and } X_j \leftarrow X_k \text{ to } \textbf{K}
6 \quad \textbf{return } \textbf{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 $G$ be a DAG. A chain graph $K$ is a class PDAG of the equivalence class of $G$ if shares the same skeleton as $G$, and contains a directed edge $X \rightarrow Y$ if and only if all $G'$ that are I-equivalent to $G$ contain the edge $X \rightarrow Y$.\textsuperscript{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  
)

\[ S, \{U_{X_i,X_j}\} \leftarrow \text{Build-PMap-Skeleton}(\mathcal{X}, P) \]
\[ \mathcal{K} \leftarrow \text{Find-Immoralities}(\mathcal{X}, S, \{U_{X_i,X_j}\}) \]

while not converged

Find a subgraph in $\mathcal{K}$ matching the left-hand side of a rule $R1 \ R3$

Replace the subgraph with the right-hand side of the rule

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


Score functions

- We can treat model selection as an optimization problem: \( \arg \max \text{score}(G, D) \)
- ML score: 
  \[
  \text{score}_L(G : D) = \ell(\langle G, \hat{\theta}_G \rangle : D)
  \]
- 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

\[
\text{score}_L(G₀ : D) = \sum_m \log \hat{\theta}_x[m] + \log \hat{\theta}_y[m]
\]

\[
\text{score}_L(G₁ : D) = \sum_m \log \hat{\theta}_x[m] + \log \hat{\theta}_y[m] | x[m]
\]

\[
\text{score}_L(G₁ : D) - \text{score}_L(G₀ : D) = \sum_m \log \hat{\theta}_y[m] | x[m] - \log \hat{\theta}_y[m]
\]

\[
\text{score}_L(G₁ : D) - \text{score}_L(G₀ : D) = \sum_{x,y} M[x,y] \log \hat{\theta}_y|x - \sum_y M[y] \log \hat{\theta}_y
\]

\[
\text{score}_L(G₁ : D) - \text{score}_L(G₀ : D) = M \sum_{x,y} \hat{P}(x,y) \log \frac{\hat{P}(y | x)}{\hat{P}(y)} = M \cdot I_{\hat{P}}(X; Y)
\]

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

\[
\text{score}_L(G : D) = M \sum_{i=1}^n I_{\hat{P}}(X_i ; \text{Pa}_{\hat{G}}^Q(X_i)) - M \sum_{i=1}^n H_{\hat{P}}(X_i)
\]
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

\[
\text{score}_B(G : D) = \log P(D \mid G) + \log P(G)
\]

\[
P(D \mid G) = \int_{\Theta_G} P(D \mid \theta_G, G)P(\theta_G \mid G)d\theta_G
\]

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

\[
\frac{1}{M} \log P(D \mid G) \approx E_{P*}[\log P(X \mid G, D)]
\]
Expected log pred lik vs avg log marg lik
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

\[
P(D | \mathcal{G}_{X-Y}) = \left( \int_{\theta_X} P(\theta_X | \mathcal{G}_{X-Y}) \prod_{m} P(x[m] | \theta_X, \mathcal{G}_{X-Y}) d\theta_X \right)
\]

\[
\left( \int_{\theta_Y|x^0} P(\theta_Y|x^0 | \mathcal{G}_{X-Y}) \prod_{m: x[m]=x^0} P(y[m] | \theta_Y|x^0, \mathcal{G}_{X-Y}) d\theta_Y|x^0 \right)
\]

\[
\left( \int_{\theta_Y|x^1} P(\theta_Y|x^1 | \mathcal{G}_{X-Y}) \prod_{m: x[m]=x^1} P(y[m] | \theta_Y|x^1, \mathcal{G}_{X-Y}) d\theta_Y|x^1 \right)
\]

• For CPTs with dirichlet priors: BDe score

\[
P(D | \mathcal{G}) = \prod_{i} \prod_{u_i \in Var(Pa_{\mathcal{G}})} \frac{\Gamma(\alpha_{X_i|u_i}^{\mathcal{G}})}{\Gamma(\alpha_{X_i|u_i}^{\mathcal{G}} + M[u_i])} \prod_{x_i^j \in val(X_i)} \left[ \frac{\Gamma(\alpha_{x_i^j|u_i}^{\mathcal{G}} + M[x_i^j, u_i])}{\Gamma(\alpha_{x_i^j|u_i}^{\mathcal{G}})} \right]
\]
Asymptotic approximations to Bayesian score

• We have

\[
\text{Thm 18.3.4: If we use a Dirichlet parameter prior for all parameters in our network, then, as } M \to \infty, \text{ we have that:}
\]

\[
\log P(\mathcal{D} | \mathcal{G}) = \ell(\hat{\theta}_g : \mathcal{D}) - \frac{\log M}{2}\text{Dim}[\mathcal{G}] + O(1)
\]

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

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

\[
\text{score}_{BIC}(\mathcal{G} : \mathcal{D}) = \sum_{i=1}^{n} I_p(X_i; \text{Pa}_{X_i}) - \sum_{i=1}^{n} H_p(X_i) - \frac{\log M}{2}\text{Dim}[\mathcal{G}]
\]

\[
\text{MDL} = \text{BIC}
\]

\[
\text{Thm 18.3.6. BIC, MDL and Bayesian score are consistent (so score(G)=score(G*) iff G is I-equivlent to G*)}
\]
Structure priors

• $P(G)$ only matters in small sample setting

• Penalized number of edges: $P(G) \propto e^{|G|}$

• 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 $G$ can be written as

  $$\text{score}(G : D) = \sum_i \text{FamScore}(X_i | Pa_i^G : D)$$

- **BIC score is decomposable**

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

- Thm 18.3.10. parameter modularity $\Rightarrow$ BDe score is decomposable

- Defn: Structural modularity if $p(G)$ decomposes

- Thm 18.3.10. param & struct modularity $\Rightarrow$ Bayesian score decomposable
Score equivalence

• Def 18.3.11. Score() is score equiv if 
score(G)=score(G’) if G, G’ are l-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

\[ \alpha_{x_i|\text{pa}_i} = \alpha \cdot P'(x_i, \text{pa}_i). \]

• Eg if P’ is a uniform prior network, then

\[ \theta_{ijk} \overset{\text{def}}{=} p(X_i = k|X_{\pi_i} = j) \]
\[ \theta_{ijk} \sim \text{Dir}(\alpha_{ijk}) \]
\[ \alpha_{ijk} = \frac{1}{q_i r_i} \]

\( \alpha_{i\phi\kappa} = 1 \) (K2 prior) is not score equiv

thetaY \sim \text{Dir}(1,1)
thetaY|X=1 \sim \text{Dir}(1,1)
thetaY|X=0 \sim \text{Dir}(1,1)
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 is decomposable if the score of a structure $G$ can be written as

$$\text{score}(G : \mathcal{D}) = \sum_i \text{FamScore}(X_i | Pa_i^G : \mathcal{D})$$

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

$$P(G) \propto \prod_i P(Pa_{X_i} = Pa_{X_i}^G)$$

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

- Thm 18.3.10. Structural & parameter modularity $\Rightarrow$ Bayesian score is decomposable.