**Structure learning: why?**

- We often want to learn the structure of the graphical model:
  - Scientific discovery (data mining)
  - Use a good model for prediction, compression, classification etc.
- Often there may be more than one good model
  - Look for features that they all share
  - Average predictions over models
LEARNING GENE REGULATORY PATHWAYS
STRUCTURE LEARNING: HOW?

• Constraint-based approach:
  – Assume some way of testing conditional independencies
    \[ X_1 \perp X_2 \mid X_3 \]
  – Then construct model consistent with these results

• Search-and-score approach:
  – Define a scoring function for measuring model quality (e.g., marginal
    likelihood or penalized likelihood)
  – Use a search algorithm to find a (local) maximum of the score
**Identifiability**

- DAGs are I-equivalent if they encode the same set of conditional independencies, e.g., $X \rightarrow Y \rightarrow Z$ and $X \leftarrow Y \leftarrow Z$ are indistinguishable given just observational data.
- However, $X \rightarrow Y \leftarrow Z$ has a v-structure, which has a unique statistical signature. Hence some arc directions can be inferred from passive observation.
- The set of I-equivalent DAGs can be represented by a PDAG (partially directed acyclic graph).
- Distinguishing between members of an equivalence class requires interventions/ experiments.
The build-PDAG algorithm from K&F chapter 3 can recover the true DAG up to I-equivalence in $O(N^{3.2d})$ time if we make the following assumptions:

- The maximum fan-in (number of parents) of any node is $d$
- The independence test oracle can handle up to $2d + 2$ variables
- The underlying distribution $P^*$ is faithful to $G^*$ i.e., there are no spurious independencies that are not sanctioned by $G^*$ ($G^*$ is a P-map of $P^*$).

This is often called the IC or PC algorithm.
CONSTRANT-BASED APPROACH

• Bad
  – Faithfulness assumption rules out certain CPDs like noisy-OR.
  – Hard to make a reliable independence test (especially given small data sets) which does not make too many errors (either false positives or false negatives).
  – One misleading independence test result can result in multiple errors in the resulting PDAG, so overall the approach is not very robust to noise.

• Good
  – PC algorithm is less dumb than local search
Independence tests

- An independence test $X \perp Y$ seeks to accept or reject the null hypothesis $H_0$ that $P^*(X, Y) = P^*(X)P^*(Y)$.
- We need a decision rule that maps data to accept/reject.
- We define a scalar measure of deviance $d(D)$ from the null hypothesis.
- The p-value of a threshold $t$ is the probability of falsely rejecting the null hypothesis:

$$p(t) = P\left(\{D : d(D) > t\}|H_0, N\right)$$

- Note that we need to know the size of the data set $N$ (stopping rule) ahead of time!
- We usually choose a threshold $t$ so that the probability of a false rejection is below some significance level $\alpha = 0.05$. 
INDEPENDENCE TESTS

• For discrete data, a common deviance is the $\chi^2$ statistic, which measures how far the counts are from what we would expect given independence:

$$d_{\chi^2}(D) = \sum_{x,y} \frac{(O_{x,y} - E_{x,y})^2}{E_{x,y}} = \sum_{x,y} \frac{(N(x,y) - NP(x)P(y))^2}{NP(x)P(y)}$$

• The p-value requires summing over all datasets of size $N$:

$$p(t) = P\{D : d(D) > t\} | H_0, N$$

• Since this is expensive in general, a standard approximation is to consider the expected distribution of $d(D)$ (under the null hypothesis) as $N \rightarrow \infty$, and use this to define thresholds to achieve a given significance.
Example of classical hypothesis testing

• When spun on edge $N = 250$ times, a Belgian one-euro coin came up heads $Y = 140$ times and tails 110.

• We would like to distinguish two models, or hypotheses: $H_0$ means the coin is unbiased (so $p = 0.5$); $H_1$ means the coin is biased (has probability of heads $p \neq 0.5$).

• p-value is “less than 7%”: $p = P(Y \geq 140) + P(Y \leq 110) = 0.066$:

  n=250; p = 0.5; y = 140;
  p = (1-binocdf(y-1,n,p)) + binocdf(n-y,n,p)

• If $Y = 141$, we get $p = 0.0497$, so we can reject the null hypothesis at significance level 0.05.

• But is the coin really biased?
Bayesian approach

- We want to compute the posterior ratio of the 2 hypotheses:
  \[
  \frac{P(H_1|D)}{P(H_0|D)} = \frac{P(D|H_1)P(H_1)}{P(D|H_0)P(H_0)}
  \]

- Let us assume a uniform prior \( P(H_0) = P(H_1) = 0.5 \).

- Then we just focus on the ratio of the marginal likelihoods:
  \[
  P(D|H_1) = \int_0^1 d\theta \ P(D|\theta, H_1)P(\theta|H_1)
  \]

- For \( H_0 \), there is no free parameter, so
  \[
  P(D|H_0) = 0.5^N
  \]
  where \( N \) is the number of coin tosses in \( D \).
Parameter prior

• How to compute $P(D|H_1)$?
• Let us assume a beta prior on the coin bias $\theta$

$$P(\theta|\alpha, H_1) = \beta(\theta; \alpha_h, \alpha_t) = \frac{1}{Z(\alpha_h, \alpha_t)} \theta^{\alpha_h-1}(1 - \theta)^{\alpha_t-1}$$

where

$$Z(\alpha_h, \alpha_t) = \int_0^1 d\theta \theta^{\alpha_h-1}(1 - \theta)^{\alpha_t-1} = \frac{\Gamma(\alpha_h)\Gamma(\alpha_t)}{\Gamma(\alpha_h + \alpha_t)}$$

• $\Gamma(n) = (n - 1)!$ for positive integers.
• Mean $E\theta = \frac{\alpha_h}{\alpha_h + \alpha_t}$.
• If we set $\alpha_h = \alpha_t = 1$, we get a uniform prior (and $Z = 1$).
Suppose we see $D_h$ heads and $D_t$ tails. The parameter posterior is

\[
P(\theta|D, \alpha) = \frac{p(\theta|\alpha)P(D|\theta, \alpha)}{P(D|\alpha)}
\]

\[
= \frac{1}{P(D|\alpha)} \frac{1}{Z(\alpha_h, \alpha_t)} \theta^{\alpha_h-1}(1 - \theta)^{\alpha_t-1} \theta^D_h (1 - \theta)^D_t
\]

\[
= \beta(\theta; \alpha_h + D_h, \alpha_t + D_t)
\]
Parameter posterior - small sample, uniform prior

prior, 1.0, 1.0

likelihood, 1 heads, 0 tails

posterior

prior, 1.0, 1.0

likelihood, 1 heads, 1 tails

posterior

prior, 1.0, 1.0

likelihood, 10 heads, 1 tails

posterior

prior, 1.0, 1.0

likelihood, 10 heads, 5 tails

posterior

prior, 1.0, 1.0

likelihood, 10 heads, 10 tails

posterior
PARAMETER POSTERIOR - SMALL SAMPLE, STRONG PRIOR
thetas = 0:0.01:1;
alphaH = 1; alphaT = 1;
prior = betapdf(thetas, alphaH, alphaT);
lik = thetas.^Nh .* (1-thetas).^Nt;
post = betapdf(thetas, alphaH+Nh, alphaT+Nt);
Model evidence

- Suppose we see \( D_h \) heads and \( D_t \) tails. The parameter posterior is

\[
P(\theta|D, \alpha) = \frac{p(\theta|\alpha)P(D|\theta, \alpha)}{P(D|\alpha)}
= \frac{1}{P(D|\alpha)Z(\alpha_h, \alpha_t)}\theta^{\alpha_h - 1}(1 - \theta)^{\alpha_t - 1}\theta^{D_h}(1 - \theta)^{D_t}
= \beta(\theta; \alpha_h + D_h, \alpha_t + D_t)
\]

where the marginal likelihood (evidence) is

\[
P(D|\alpha) = \frac{Z(\alpha_h + N_h, \alpha_t + N_t)}{Z(\alpha_h, \alpha_t)}
= \frac{\Gamma(\alpha)}{\Gamma(\alpha + N)} \cdot \frac{\Gamma(\alpha_h + N_h)}{\Gamma(\alpha + N)} \cdot \frac{\Gamma(\alpha_t + N_t)}{\Gamma(\alpha + N)}
\]
Sequentially Evaluating the Evidence

• By the chain rule of probability,

\[ P(x_1:N) = P(x_1)P(x_2|x_1)P(x_3|x_1:2) \ldots \]

• Also, after \( N \) data cases, \( P(X|D_{1:N}) = \text{Dir}(\vec{\alpha} + \vec{N}) \), so

\[ P(X = k|D_{1:N}, \vec{\alpha}) = \frac{N_k + \alpha_k}{\sum_i N_i + \alpha_i} = \frac{N_k + \alpha_k}{N + \alpha} \]

• Suppose \( D = H, T, T, H, H, H \). Then

\[
P(D) = \frac{\alpha_h}{\alpha} \cdot \frac{\alpha_t}{\alpha + 1} \cdot \frac{\alpha_t + 1}{\alpha + 2} \cdot \frac{\alpha_h + 1}{\alpha + 3} \cdot \frac{\alpha_h + 2}{\alpha + 4}
\]

\[
= \frac{[\alpha_h(\alpha_h + 1)(\alpha_h + 2)] [\alpha_t(\alpha_t + 1)]}{\alpha(\alpha + 1) \cdots (\alpha + 4)}
\]

\[
= \frac{[(\alpha_h) \cdots (\alpha_h + N_h - 1)] [(\alpha_t) \cdots (\alpha_t + N_t - 1)]}{(\alpha) \cdots (\alpha + N)}
\]
Model evidence

• For integers,

\[(\alpha)(\alpha + 1) \cdots (\alpha + M - 1) = \frac{(a + M - 1)!}{(\alpha - 1)!} = \frac{(a + M - 1)(a + M - 2) \cdots (a + M - M)(a + M - M - 1) \cdots 2 \cdot 1}{(a - 1)(a - 2) \cdots 2 \cdot 1} = \frac{(a + M - 1)(a + M - 2) \cdots (a)(a - 1) \cdots 2 \cdot 1}{(a - 1)(a - 2) \cdots 2 \cdot 1}\]

• For reals, we replace \((a - 1)!\) with \(\Gamma(a)\).

• Hence

\[P(D) = \frac{[(\alpha_h) \cdots (\alpha_h + N_{h} - 1)] [(\alpha_t) \cdots (\alpha_t + N_{t} - 1)]}{(\alpha) \cdots (\alpha + N)} = \frac{\Gamma(\alpha)}{\Gamma(\alpha + N)} \cdot \frac{\Gamma(\alpha_h + N_{h})}{\Gamma(\alpha + N)} \cdot \frac{\Gamma(\alpha_t + N_{t})}{\Gamma(\alpha + N)}\]
We compute the ratio of marginal likelihoods (evidence):

\[
\frac{P(H_1|D)}{P(H_0|D)} = \frac{P(D|H_1)}{P(D|H_0)} = \frac{Z(\alpha_h + N_h, \alpha_t + N_t)}{Z(\alpha_h, \alpha_t)} \frac{1}{0.5^N}
\]

\[
= \frac{\Gamma(140 + \alpha)\Gamma(110 + \alpha)}{\Gamma(250 + 2\alpha)} \times \frac{\Gamma(2\alpha)}{\Gamma(\alpha)\Gamma(\alpha)} \times 2^{250}
\]

Must work in log domain!

```matlab
alphas = [0.37 1 2.7 7.4 20 55 148 403 1096];
Nh = 140; Nt = 110; N = Nh+Nt;
numer = gammaln(Nh+alphas) + gammaln(Nt+alphas) + gammaln(alphas);
denom = gammaln(N+2*alphas) + 2*gammaln(alphas);
r = exp(numer ./ denom);
```
So, is the coin biased or not?

- We plot the likelihood ratio vs hyperparameter $\alpha$:

- For a uniform prior, $\frac{P(H_1|D)}{P(H_0|D)} = 0.48$, (weakly) favoring the fair coin hypothesis $H_0$!

- At best, for $\alpha = 50$, we can make the biased hypothesis twice as likely.

- Not as dramatic as saying “we reject the null hypothesis (fair coin) with significance 6.6%”. 
FROM COINS TO DICE

• Likelihood: binomial → multinomial
  \[ P(D|\theta) = \prod_i \theta_i^{N_i} \]

• Prior: beta → Dirichlet
  \[ P(\theta|\alpha) = \frac{1}{Z(\alpha')} \prod_i \theta_i^{\alpha_i-1} \]
  where
  \[ Z(\alpha') = \frac{\prod_i \Gamma(\alpha_i)}{\Gamma(\sum_i \alpha_i)} \]

• Posterior: beta → Dirichlet
  \[ P(\theta|D) = Dir(\alpha + \tilde{N}) \]

• Evidence (marginal likelihood)
  \[ P(D|\alpha) = \frac{Z(\alpha + \tilde{N})}{Z(\alpha')} = \frac{\prod_i \Gamma(\alpha_i + N_i)}{\prod_i \Gamma(\alpha_i)} \frac{\Gamma(\sum_i \alpha_i)}{\Gamma(\sum_i \alpha_i + N_i)} \]
From dice to tabular Bayes nets

• If we assume global parameter independence, the evidence decomposes into one term per node:

\[ P(D|G) = \prod_i P(D(X_i, X_{\pi_i})|\tilde{\alpha}_i) \]

• If we also assume local parameter independence, each node term decomposes into a product over rows (conditioning cases):

\[
P(D|G) = \prod_i \prod_{k \in Val(\pi_i)} P(D(X_i, X_{\pi_i} = k)|\tilde{\alpha}_i,.,k) \\
= \prod_i \prod_{k \in Val(\pi_i)} \frac{Z(\tilde{\alpha}_i,.,k + N_i,.,k)}{Z(\tilde{\alpha}_i,.,k)} \\
= \prod_i \prod_{k \in Val(\pi_i)} \left[ \prod_j \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})} \right] \left[ \frac{\Gamma(\sum_j \alpha_{ijk})}{\Gamma(\sum_j \alpha_{ijk} + N_{ijk})} \right]
\]
Example of model selection

• Suppose we generate data from $X \rightarrow Y$, where $P(X = 0) = P(X = 1) = 0.5$ and $P(Y = 1|X = 0) = 0.5 - \epsilon$, $P(Y = 1|X = 1) = 0.5 + \epsilon$.

• As we increase $\epsilon$, we increase the dependence of $Y$ on $X$.

• Let us consider 3 hypotheses: $H_0 = X \rightarrow Y$, $H_1 = X \rightarrow Y$, $H_2 = Y \leftarrow X$, and use uniform priors.

• We will plot model posteriors vs $N$ for different $\epsilon$ and different random trials:

$$P(H_i|D_{1:N}) = \frac{P(D_{1:N}|H_i)P(H_i)}{\sum_j P(D_{1:N}|H_j)P(H_j)}$$
Example of model selection

red = $H_0$ (independence), blue/green = $H_1 / H_2$ (dependence).
See BNT/examples/static/StructLearn/model-select1.m.
Score equivalence

- \(X \rightarrow Y\) and \(X \leftarrow Y\) are l-equivalent (have the same likelihood).
- Suppose we use a uniform Dirichlet prior for each node in each graph, with equivalent sample size \(\alpha\) (K2-prior):
  \[
P(\theta_X \mid H_1) = \text{Dir}(\alpha, \alpha), \quad P(\theta_X \mid Y = i \mid H_2) = \text{Dir}(\alpha, \alpha)
  \]
- In \(H_1\), the equivalent sample size for \(X\) is \(2\alpha\), but in \(H_2\) it is \(4\alpha\) (since two conditioning contexts). Hence the posterior probabilities are different.
- The BDe (Bayesian Dirichlet likelihood equivalent) prior is to use weights \(\alpha_{X_i \mid X_{\pi_i}} = \alpha P'(X_i, X_{\pi_i})\) where \(P'\) could be represented by e.g., a Bayes net.
- The BDDeu (uniform) prior is \(P'(X_i, X_{\pi_i}) = \frac{1}{|X_i \mid \mid X_{\pi_i} |} \).
- Using the BDDeu prior, the curves for \(X \rightarrow Y\) and \(X \leftarrow Y\) are indistinguishable. Using the K2 prior, they are not.
Bayesian Occam’s razor

• Why is \( P(H_0|D) \) higher when the dependence on \( X \) and \( Y \) is weak (small \( \epsilon \))?

• It is not because the prior \( P(H_i) \) explicitly favors simpler models (although this is possible).

• It because the evidence \( P(D) = \int dwP(D|w)P(w) \), automatically penalizes complex models.

• Occam’s razor says “If two models are equally predictive, prefer the simpler one”.

• This is an automatic consequence of using Bayesian model selection.

• Maximum likelihood would always pick the most complex model, since it has more parameters, and hence can fit the training data better.

• Good test for a learning algorithm: feed it random noise, see if it “discovers” structure!
Laplace approximation to the evidence

- Consider a large sample approximation, where the parameter posterior becomes peaked.
- Take a second order Taylor expansion around \( \hat{\theta}_{MP} \):
  \[
  \log P(\theta|D) \approx \log P(\hat{\theta}_{MP}|D) - \frac{1}{2}(\theta - \hat{\theta})^T H (\theta - \hat{\theta})
  \]
  where
  \[
  H \overset{\text{def}}{=} -\frac{\partial^2 \log P(\theta|D)}{\partial \theta \partial \theta^T} \bigg|_{\hat{\theta}_{MP}}
  \]
  is the Hessian.
- By properties of Gaussian integrals,
  \[
  P(D) \approx \int d\theta \ P(D|\hat{\theta})P(\hat{\theta})e^{-\frac{1}{2}(\theta - \hat{\theta})^T H (\theta - \hat{\theta})}
  \]
  \[
  = P(D|\hat{\theta})P(\hat{\theta})(2\pi)^{d/2}|H|^{-\frac{1}{2}}
  \]
**Occam Factor**

- $H$ is like the precision (inverse covariance) of a Gaussian.
- In the 1d case, $|H|^{-\frac{1}{2}} = \sigma_\theta|_D$, the width of the posterior.
- Consider a uniform prior with width $\sigma_\theta$.
  
  Then $P(D) \approx P(D|\hat{\theta})P(\hat{\theta})|H|^{-\frac{1}{2}} \approx P(D|\hat{\theta})\frac{1}{\sigma_\theta}\sigma_\theta|_D$

- The ratio of posterior accessible volume of the parameter space to the prior, $\sigma_\theta|_D/\sigma_\theta$, is called the Occam factor, i.e., the factor by which $H_i$'s hypothesis space collapses when the data arrive.
Bayesian Occam’s razor

- $P(D|H_1)$ is smallest, since it is too simple a model.
- $P(D|H_3)$ is second smallest, since it is too complex, so it spreads its probability mass more thinly over the $(D, \theta)$ space (fewer dots on the horizontal line).
- We trust an expert who predicts a few specific (and correct!) things more than an expert who predicts many things.
Bayesian image interpretation

• How many boxes behind the tree?
• The interpretation that the tree is in front of one box is much more probable than there being 2 boxes which happen to have the same height and color (suspicious coincidence).
• This can be formalized by assuming (uniform) priors on the box parameters, and computing the Occam factors.
Leave one out cross validation (LOOCV)

- The evidence can be evaluated sequentially
  \[ P(x_{1:N}) = P(x_1)P(x_2|x_1)P(x_3|x_1:2) \ldots \]

- LOOCV approximates \( P(X_t|X_{1:t-1}, \hat{\theta}_{1:t-1}) \) under different permutations of the data.

- Advantages of LOOCV
  - Simple (no need to integrate out parameters)
  - Robust (works well even if “truth not in model class”)

- Advantages of LOOCV
  - Slow (in general, must rerun training many times)
  - Does not use all the data
Another way of thinking about Bayesian Occam’s razor is in terms of information theory.

To losslessly send a message about an event $x$ with probability $P(x)$ takes $L(x) = -\log_2 P(x)$ bits.

Suppose instead of sending the raw data, you send a model and then the residual errors (the parts of the data not predicted by the model).

This takes $L(D, H)$ bits:

$$L(D, H) = -\log P(H) - \log(P(D|H)) = -\log P(H|D) + \text{const}$$

The best model is the one with the overall shortest message.
Minimum description length (MDL)

\[ L(H_1) \quad L(w^*_1 | H_1) \quad L(D | w^*_1, H_1) \]

\[ L(H_2) \quad L(w^*_2 | H_2) \quad L(D | w^*_2, H_2) \]

\[ L(H_3) \quad L(w^*_3 | H_3) \quad L(D | w^*_3, H_3) \]
BIC approximation to the evidence

- Laplace approximation

\[ P(D) \approx P(D|\hat{\theta})P(\hat{\theta})(2\pi)^{d/2}|H|^{-\frac{1}{2}} \]

- Taking logs

\[ \log P(D) = \log P(D|\hat{\theta}) + \log P(\hat{\theta}) + \frac{d}{2} \log(2\pi) - \frac{1}{2} \log |H| \]

- BIC (Bayesian Information Criterion): drop terms that are independent of \( N \), and approximate \( \log |H| \approx d \log N \). So

\[ \log P(D) \approx \log P(D|\hat{\theta}_{ML}) - \frac{d}{2} \log N \]

where \( d \) is the number of free parameters.

- AIC (Akaike Information Criterion): derived by minimizing KL divergence independent of \( N \), and approximate \( \log |H| \approx d \log N \). So

\[ \log P(D) \approx \log P(D|\hat{\theta}_{ML}) - \frac{d}{2} \log N \]
Log-likelihood in information theoretic terms

\[
\frac{1}{N} \ell = \frac{1}{N} \sum_i \sum_j \sum_k N_{ijk} \log \theta_{ijk}
\]

\[
= \sum_i \sum_j \sum_k \hat{P}(X_i = j, X_{\pi_i} = k) \log P(X_i = j | X_{\pi_i} = k)
\]

\[
= \sum_{ijk} \hat{P}(X_i = j, X_{\pi_i} = k) \log \frac{P(X_i = j, X_{\pi_i} = k)P(X_i = j)}{P(X_{\pi_i} = k)P(X_i = j)}
\]

\[
= \sum_i \sum_j \sum_k \hat{P}(X_i = j, X_{\pi_i} = k) \log \frac{P(X_i = j, X_{\pi_i} = k)}{P(X_{\pi_i} = k)P(X_i = j)}
\]

\[
+ \sum_{ij} (\sum_k \hat{P}(X_i = j, X_{\pi_i} = k)) \log P(X_i = j)
\]

\[
= \sum_i I(X_i, X_{\pi_i}) - H(X_i)
\]
**BIC in information theoretic terms**

\[
\text{score}_{BIC}(G|D) = \ell(\hat{\theta}) - \frac{d(G)}{2} \log N(D)
\]

\[
= N \sum_i I(X_i, X_{\pi_i}) - N \sum_i H(X_i) - \frac{d}{2} \log N
\]

- The mutual information term grows linearly in \( N \), the complexity penalty is logarithmic in \( N \).
- So for large datasets, we pay more attention to fitting the data better.
- Also, the structural prior is independent of \( N \), so does not matter very much.
Desirable properties of a scoring function

• Consistency: i.e., if the data is generated by $G^*$, then $G^*$ and all $I$-equivalent models maximize the score.

• Decomposability:

$$\text{score}(G|D) = \sum_i \text{FamScore}(D(X_i, X_{\pi_i}))$$

which makes it cheap to compare score of $G$ and $G'$ if they only differ in a small number of families.

• Bayesian score (evidence), likelihood and penalized likelihood (BIC) are all decomposable and consistent.
Maximizing the score

• Consider the family of DAGs $G_d$ with maximum fan-in (number of parents) equal to $d$.

• Theorem 14.4.3: It is NP-hard to find

$$G^* = \arg \max_{G \in G_d} \text{score}(G, D)$$

for any $d \geq 2$.

• In general, we need to use heuristic local search.
Maximizing the score: tractable cases

• For $d \leq 1$ (i.e., trees), we can solve the problem in $O(n^2)$ time using max spanning tree (next lecture).

• If we know the ordering of the nodes, we can solve the problem in $O(d \binom{n}{d})$ time (see below).
**Known order (K2 algorithm)**

- Suppose we a total ordering of the nodes $X_1 \prec X_2 \ldots \prec X_n$ and want to find a DAG consistent with this with maximum score.
- The choice of parents for $X_i$, from $Pa_i \subseteq \{X_1, \ldots, X_{i-1}\}$, is independent of the choice for $X_j$: since we obey the ordering, we cannot create a cycle.
- Hence we can pick the best set of parents for each node independently.
- For $X_i$, we need to search all $\binom{i-1}{d}$ subsets of size up to $d$ for the set which maximizes FamScore.
- We can use greedy techniques for this, c.f., learning a decision tree.
What if order isn’t known?

- Search in the space of DAGs.
- Search in the space of orderings, then conditioned on $\prec$, pick best graph using K2 (Rao-Blackwellised sampling).
- Can also search in space of undirected graphs.
- Can also search in space of graphs of variable size, to allow creation of hidden nodes (next lecture).
Searching in DAG space

- Typical search operators:
  - Add an edge
  - Delete an edge
  - Reverse an edge
- We can get from any graph to any other graph in at most $O(n^2)$ moves (the diameter of the search space).
- Moves are reversible.
- Simplest search algorithm: greedy hill climbing.
- We can only apply a search operator $o$ to the current graph $G$ if the resulting graph $o(G)$ satisfies the constraints, e.g., acyclicity, indegree bound, induced treewidth bound ("thin junction trees"), hard prior knowledge.
Cost of evaluating moves

• There are $O(n^2)$ operators we could apply at each step.
• For each operator, we need to check if $o(G)$ is acyclic.
• We can check acyclicity in $O(e)$ time, where $e = O(nd)$ is the number of edges.
• For local moves, we can check acyclicity in amortized $O(1)$ time using the ancestor matrix.
• If $o(G)$ is acyclic, we need to evaluate its quality. This requires computing sufficient statistics for every family, which takes $O(Mn)$ time, for $M$ training cases.
• Suppose there are $K$ steps to convergence. (We expect $K \ll n^2$, since the diameter is $n^2$.)
• Hence total time is $O(K \cdot n^2 \cdot Mn)$. 
**Exploiting decomposable score**

- If the operator is valid, we need to evaluate its quality. Define
  \[ \delta_G(o) = \text{score}(o(G)|D) - \text{score}(G|D) \]
- If the score is decomposable, and we want to modify an edge involving \(X\) and \(Y\), we only need to look at the sufficient statistics for \(X\) and \(Y\)’s families.
- e.g., if \(o = \text{add } X \rightarrow Y\):
  \[ \delta_G(o) = \text{FamScore}(Y, \text{Pa}(Y, G) \cup X|D) - \text{FamScore}(Y, \text{Pa}(Y, G)|D) \]
- So we can evaluate quality in \(O(M)\) time by extracting sufficient statistics for the columns related to \(X, Y\) and their parents.
- This reduces the time from \(O(Kn^3M)\) to \(O(Kn^2M)\).
Exploiting decomposable score

- After eg adding $X \rightarrow Y$, we only need to update $\delta(o)$ for the $O(n)$ operators that involve $X$ or $Y$.
- Also, we can update a heap in $O(n \log n)$ time and thereby find the best $o$ in $O(1)$ time at each step.
- So total cost goes from $O(Kn^2M)$ to $O(K(nM + n \log n))$.
- For large $M$, we can use fancy data structures (e.g., kd-trees) to compute sufficient statistics in sub-linear time.
Local maxima

• Greedy hill climbing will stop when it reaches a local maximum or a plateau (a set of neighboring networks that have the same score).

• Unfortunately, plateaux are common, since equivalence classes form contiguous regions of search space (thm 14.4.4), and such classes can be exponentially large.

• Solutions:
  – Random restarts
  – TABU search (prevent the algorithm from undoing an operator applied in the last $L$ steps, thereby forcing it to explore new terrain).
  – Data perturbation (dynamic local search): reweight the data and take step.
  – Simulated annealing: if $\delta(o) > 0$, take move, else accept with probability $e^{\frac{\delta(o)}{t}}$, where $t$ is the temperature. Slow!
Searching in space of equivalence classes

• The space of class PDAGs is smaller.
• We avoid many of the plateaux of I-equivalent DAGs.
• Operators are more complicated to implement and evaluate, but can still be done locally (see paper by Max Chickering).
• Cannot exploit causal/ interventional data (which can distinguish members of an equivalence class).
• Currently less common than searching in DAG space.
Learning the ICU-Alarm network with TABU search

- Learned structures often simpler than “true” model (fewer edges), but predict just as well.
- Can only recover structure up to Markov equivalence.
- 10 minutes to learn structure for 100 variables and 5000 cases.