Graphical Models For Complex Health Data (P8124)

Daniel Malinsky

Columbia University
dsm2128@cumc.columbia.edu

Structure Learning (part 1)

Where does a model come from?

Experimental/problem design

- not always possible, feasible, or obvious

Where does a model come from?

Experimental/problem design

- not always possible, feasible, or obvious

Theory

- may be wrong
- may be underspecified

Where does a model come from?

Experimental/problem design

- not always possible, feasible, or obvious

Theory

- may be wrong
- may be underspecified

Experts

- may be wrong
- experts may disagree
- problem may be intractable (high-dimensional settings)

Where does a model come from?

Experimental/problem design

- not always possible, feasible, or obvious

Theory

- may be wrong
- may be underspecified

Experts

- may be wrong
- experts may disagree
- problem may be intractable (high-dimensional settings)

Data

- question is: how?

Approaches to graphical structure learning

Constraint-based: hypothesis tests of conditional independence (e.g., PC)

Greedy score-based: optimization of a model fit score (e.g., GES)

Semi-parametric methods: exploiting semi-parametric assumptions (e.g., ICA-LiNGaM)

Optimization-based: continuous optimization of a penalized likelihood or score (e.g., CCDr)

Hybrid methods: mix of above

Approaches to graphical structure learning

Constraint-based: hypothesis tests of conditional independence (e.g., PC)

Score-based: greedy optimization of a model fit score (e.g., GES)

Constraint-based versus score-based

Consider a DAG $\mathcal{G}$. By the global Markov property, we know that if $X_i \perp_d X_j | X_S$ in $\mathcal{G}$ then $X_i \perp\!\!\!\perp X_j | X_S$ $(X_i, X_j \in V, S \subseteq V \setminus \{X_i, X_j\})$.

So we can say that the graphical structure places an independence *constraint* on the data distribution. Constraint-based methods directly exploit those implied constraints on the data to try and infer backwards from data to graphical structure.

Constraint-based versus score-based

Consider a DAG $\mathcal{G}$. By the global Markov property, we know that if $X_i \perp_d X_j | X_S$ in $\mathcal{G}$ then $X_i \perp\!\!\!\perp X_j | X_S$ $(X_i, X_j \in V, S \subseteq V \setminus \{X_i, X_j\})$.

So we can say that the graphical structure places an independence *constraint* on the data distribution. Constraint-based methods directly exploit those implied constraints on the data to try and infer backwards from data to graphical structure.

Score-based methods instead proceed by assigning to every a graph $\mathcal{G}$ a score based on the data, some measure of how well the graph "fits" the data. Higher scores mean better fit, and a score-based approach can be viewed as an optimization task: search for the graph which has the highest score.

Constraint-based versus score-based

Consider a DAG $\mathcal{G}$. By the global Markov property, we know that if $X_i \perp_d X_j | X_S$ in $\mathcal{G}$ then $X_i \perp\!\!\!\perp X_j | X_S$ $(X_i, X_j \in V, S \subseteq V \setminus \{X_i, X_j\})$.

So we can say that the graphical structure places an independence *constraint* on the data distribution. Constraint-based methods directly exploit those implied constraints on the data to try and infer backwards from data to graphical structure.

Score-based methods instead proceed by assigning to every a graph $\mathcal{G}$ a score based on the data, some measure of how well the graph "fits" the data. Higher scores mean better fit, and a score-based approach can be viewed as an optimization task: search for the graph which has the highest score.

$\Rightarrow$ neither of these approaches is universally "better" than the other, but they have their own virtues and drawbacks depending on the setting.

What might you do if computational limitations weren't an issue?

Brute force enumerate all possible DAGs, derive independence implications for each, and test them.

$X_1 \rightarrow X_2 \rightarrow X_3$ $\Downarrow$ $X_1 \perp\!\!\!\perp X_3 | X_2$

$X_1 \leftarrow X_2 \leftarrow X_3$ $\Downarrow$ $X_1 \perp\!\!\!\perp X_3 | X_2$

$X_1 \leftarrow X_2 \rightarrow X_3$ $\Downarrow$ $X_1 \perp\!\!\!\perp X_3 | X_2$

$X_1 \rightarrow X_2 \leftarrow X_3$ $\Downarrow$ $X_1 \perp\!\!\!\perp X_3$, $X_1 \not\perp\!\!\!\perp X_3 | X_2$

Markov equivalence

We typically identify causal structure only up to Markov equivalence.

$$X_i \rightarrow X_j \rightarrow X_k$$

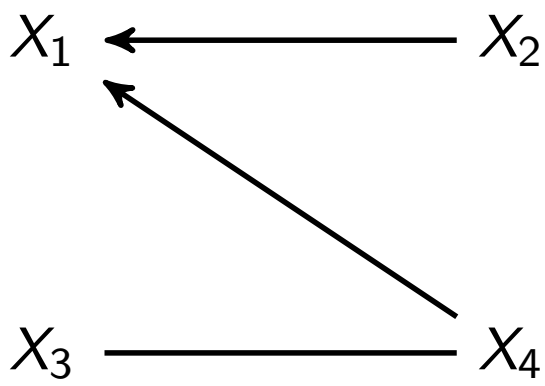
$$X_i \leftarrow X_j \leftarrow X_k$$

$$X_i \leftarrow X_j \rightarrow X_k \qquad X_i - X_j - X_k \qquad X_i \rightarrow X_j \leftarrow X_k$$

a) b) c)

Figure: a) Three Markov equivalent DAG models. b) The CPDAG representation of (a). c) A DAG that is not Markov equivalent to the graphs in (a).

CPDAG



Let $\mathcal{G}$ be a DAG. The CPDAG $\mathcal{C}$ implied by $\mathcal{G}$ is a mixed graph (directed and undirected edges) that has the same adjacencies as $\mathcal{G}$ and a directed edge $X_i \to X_j$ if and only if the edge $X_i \to X_j$ is common to all DAGs Markov equivalent to $\mathcal{G}$; all other edges in $\mathcal{C}$ are undirected.

The *skeleton* of a (partially) directed graph is the undirected graph obtained by replacing all edges by undirected edges. Recall that Markov equivalent DAGs share the same skeleton and v-structures (unshielded colliders).

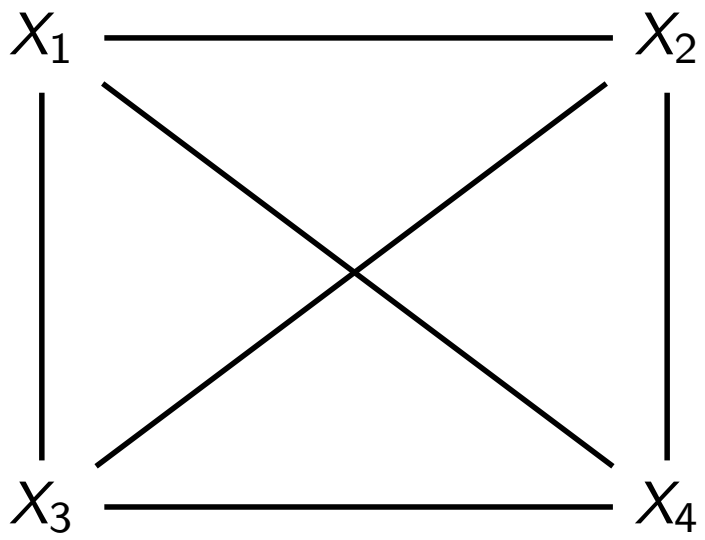
Testing all subsets?

We want to search for separating sets which make each pair X_i, X_j conditionally independent. A naive approach would be to consider all possible subsets $S \subseteq V \setminus \{X_i, X_j\}$ to evaluate whether there exists a separating set.

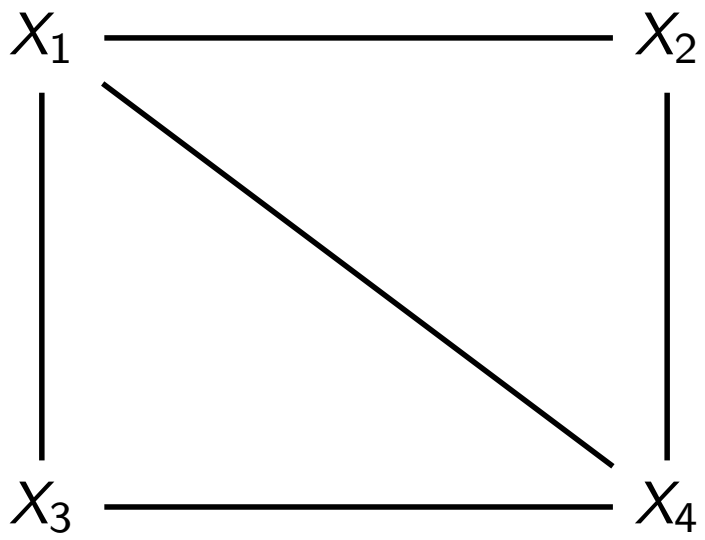
However, by the Markov properties we know that $X_i \not\perp\!\!\!\perp X_j$ if and only if $X_i \perp\!\!\!\perp X_j | \operatorname{Pa}(X_i, \mathcal{G})$ or $X_i \perp\!\!\!\perp X_j | \operatorname{Pa}(X_j, \mathcal{G})$.

We don't know the parent set ahead of time (we don't know the graph!) so we look at $S \subseteq \operatorname{Adj}(X_i, \mathcal{G}')$ and $S \subseteq \operatorname{Adj}(X_j, \mathcal{G}')$ for some $\mathcal{G}'$ which is a supergraph of the true unknown skeleton.

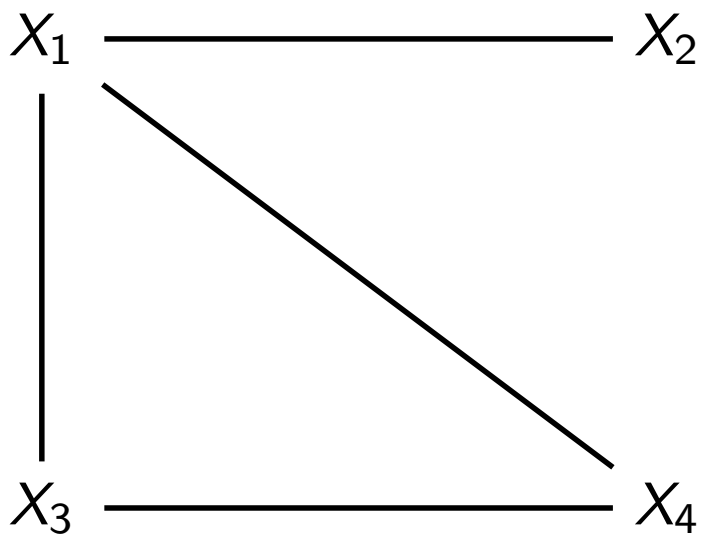
Constraint-based learning: the PC algorithm



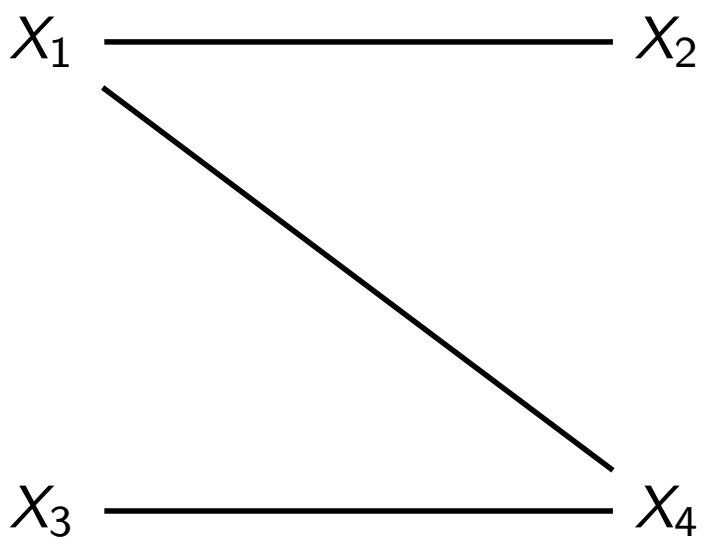
Constraint-based learning: the PC algorithm



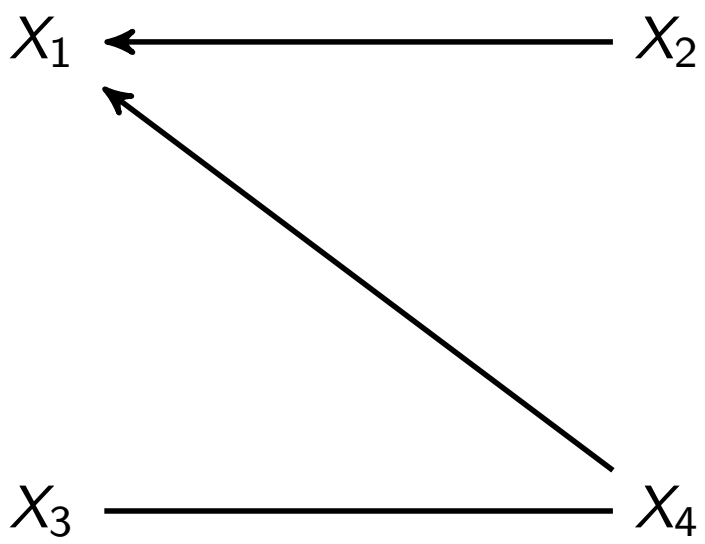
Constraint-based learning: the PC algorithm



Constraint-based learning: the PC algorithm



Constraint-based learning: the PC algorithm



Algorithm 0.1: $\mathrm{PC}(\textsc{Test}, \alpha)$

Input: Samples of the vector $X = (X_1, ..., X_p)$
Output: CPDAG $\mathcal{G}$

1. Form the complete graph $\mathcal{G}$ on vertex set $V = \{1, ..., p\}$ w/ undirected edges.
2. Let $s = 0$
3. **repeat**
4. **for all** pairs of adjacent vertices (X_i, X_j) s.t. $|\operatorname{Adj}(X_i, \mathcal{G}) \setminus \{X_j\}| \geq s$ and subsets $S \subseteq \operatorname{Adj}(X_i, \mathcal{G}) \setminus \{X_j\}$ s.t. $|S| = s$
5. **if** $X_i \perp\!\!\!\perp X_j | X_S$ according to $(\textsc{Test}, \alpha)$
then { Delete edge $X_i - X_j$ from $\mathcal{G}$. ; Let $\operatorname{sepset}(X_i, X_j) = \operatorname{sepset}(X_j, X_i) = S$. }
6. **end**
7. Let $s = s + 1$
8. **until** for each pair of adjacent vertices (X_i, X_j), $|\operatorname{Adj}(X_i, \mathcal{G}) \setminus \{X_j\}| < s$.
9. **for all** triples (i, k, j) s.t. $X_i \in \operatorname{Adj}(X_k, \mathcal{G})$ and $X_j \in \operatorname{Adj}(X_k, \mathcal{G})$ but $X_i \notin \operatorname{Adj}(X_j, \mathcal{G})$, orient $X_i \to X_k \leftarrow X_j$ in $\mathcal{G}$ iff $X_k \notin \operatorname{sepset}(X_i, X_j)$.
10. Exhaustively apply orientation rules (R1-R4) to orient remaining undirected edges.
11. **return** $\mathcal{G}$.

Orientation rules

R1: Orient $X_j - X_k$ into $X_j \to X_k$ whenever there is an arrow $X_i \to X_j$ such that X_i and X_k are nonadjacent.

R2: Orient $X_i - X_j$ into $X_i \to X_j$ whenever there is a path $X_i \to X_k \to X_j$.

R3: Orient $X_i - X_j$ into $X_i \to X_j$ whenever there are two paths $X_i - X_k \to X_j$ and $X_i - X_l \to X_j$ such that X_k and X_l are nonadjacent.

R4: Orient $X_i - X_j$ into $X_i \to X_j$ whenever there are two paths $X_i - X_k \to X_j$ and $X_i - X_l \to X_k$ such that X_j and X_l are nonadjacent.

Constraint-based learning: the PC algorithm

The "logic" of PC:

- conditional independencies correspond to missing edges
- the "collider" rule enables orientation of triples $X_i \rightarrow X_j \leftarrow X_k$
- other orientations follow from the acyclicity assumption
- remaining unoriented edges reflect the Markov equivalence of models consistent with the data

Theory

Computational complexity:

We note that for graphs with bounded degree, i.e., a bound on $d = \max_{X_i \in V} |\text{Adj}(X_i, \mathcal{G})|$, the PC algorithm has a running time that is polynomial in the number of variables. The running time depends exponentially on the degree. Specifically the number of tests is bounded by $2\binom{p}{2} \sum_{i=0}^{d} \binom{p-1}{i}$ where $p = |V|$ (in the worst case).

Theory

Correctness (soundness and completeness):

Theorem: Assume the distribution $p(x)$ is Markov and faithful to some DAG $\mathcal{G}$. Let $\mathcal{C}$ be the CPDAG implied by $\mathcal{G}$. The "oracle" PC algorithm returns $\mathcal{C}$.

Consistency:

Theorem (informal): Assume the distribution $p(x)$ is Markov and faithful to some DAG $\mathcal{G}$. Let $\mathcal{C}$ be the CPDAG implied by $\mathcal{G}$. Let $\widehat{\mathcal{C}}$ be the output of PC given some consistent conditional independence test and level α_n. Then there is a sequence of $\alpha_n \to 0$ $(n \to \infty)$ such that $\lim_{n\to\infty} \mathbb{P}(\widehat{\mathcal{C}} = \mathcal{C}) = 1$.

Conditional independence tests

With finite data, the PC algorithm needs a procedure for deciding whether $X_i \perp\!\!\!\perp X_j | X_S$.

In practice, test the null hypothesis

$$H_0 : X_i \perp\!\!\!\perp X_j | X_S$$

and reject the null if some test statistic $T(x) < \alpha$, where α is a user-specified threshold (which may depend on the sample size). That is, if we reject the null hypothesis, we keep the edge, and if we fail to reject, we remove the edge.

Conditional independence tests

Note: α in PC has no straightfoward interpretation in terms of the Type I error, since in fact we are testing multiple hypotheses and an a priori unknown number of them.

So, α should be better thought of as a *tuning parameter* which controls the **sparsity** of the estimated graph. (Smaller $\alpha \implies$ more sparse)

Though it is not straightforward, one may try to be clever and control the FDR. In practice: make α small as the number of variables and sample size gets big. (More on choosing α later.)

Conditional independence tests

The choice of statistical test depends on the distribution $p(x)$. If you're willing to assume that $p(X)$ is in some nice parametric family (Gaussian, multinomial) then life is easier. However, nonparametric tests of conditional independence are also available. First, consider the case where X is multivariate Gaussian.

Test for Gaussian data

Theorem. If X is multivariate Gaussian then $X_i \perp\!\!\!\perp X_j | S$ if and only if $\rho_{ij.S} = 0$.

$\rho_{ij.S}$ is called the partial correlation (of X_i and X_j given S) and can be defined as follows: for any $X_k \in S$,

$$\rho_{ij.S} = \frac{\rho_{ij.S\setminus X_k} - \rho_{ik.S\setminus X_k}\rho_{jk.S\setminus X_k}}{\sqrt{(1-\rho^2_{ik.S\setminus X_k})(1-\rho^2_{kj.S\setminus X_k})}}$$

This is a recursive definition that terminates at $|S| = 1$ (regular pairwise correlations). Moreover define Fisher's Z-transform:

$$Z(\rho_{ij.S}, n) = \frac{1}{2}\sqrt{n - |S| - 3}\log\left(\frac{1+\rho_{ij.S}}{1-\rho_{ij.S}}\right)$$

under the null hypothesis $\rho_{ij.S} = 0$: $Z(\rho_{ij.S}, n) - Z(\widehat{\rho}_{i,j.S}, N) \sim N(0,1)$ as $n \to \infty$. So for a level α test we reject if $Z(\widehat{\rho}_{ij.S}, n) > \Phi^{-1}(1-\alpha/2)$

Other parametric tests

There are similar tests for multinomial discrete data based on the χ^2 and G^2 test statistics.

One may also use tests based on (logistic) regression or odds ratios.

Nonparametric tests

Tests which make minimal distributional assumptions are an active area of research. A well-known test based on Kernel matrices is the KCI (Kernel-based Conditional Independence) test.[1]

KCI assumes the the variables are related by arbitrary square-integrable functions, plus some smoothness/simplicity conditions to test $X_i \perp\!\!\!\perp X_j | X_S$ nonparametrically. The test statistic is:

$$\frac{1}{n} tr(\widetilde{K}_{\ddot{X}_i|X_S} \widetilde{K}_{X_j|X_S})$$

where $\widetilde{K}_{A|B}$ is a conditional centralized kernel matrix constructed with some kernel function, $\ddot{X}_i = (X_i, X_S)$.

[1] Zhang et al. (2011) "Kernel-based conditional independence test and application in causal discovery" in UAI

Nonparametric tests

The KCI test is implemented in R (e.g., `library(CondIndTests)`)

```
n <- 100
Z <- rnorm(n)
X <- 4 + 2 * Z + rnorm(n)
Y <- 3 * X^2 + Z + rnorm(n)
test1 <- CondIndTest(X,Y,Z, method = "KCI")
cat("These data come from a distribution, for which X and Y
are NOT cond. ind. given Z.")
cat(paste("The p-value of the test is: ", test1$pvalue))
```

The problem is that the test is very computationally intensive with big n and big $|S|$.

There are other nonparametric tests in other packages, each have pros and cons

Virtues and drawbacks of constraint-based search

Virtues:

- nonparametric (in principle)
- relatively scaleable
- lots of work on improvements, heuristics, generalizations

Drawbacks:

- statistical test errors propogate, can make a big difference to the output (can be mitigated by some stability techniques)
- does not handle conflicting statistical information in an intelligent way
- not that scaleable (cannot really parallelize)
- tackles problem only "locally," makes each decision only once, produces one estimated CPDAG rather than a range of good options (no "confidence intervals" or "posterior distribution")

Constraint-based learning of undirected models

In an undirected model $\mathcal{G}$, we have the local Markov property: $X_i \not\perp\!\!\!\perp X_j$ implies $X_i \perp\!\!\!\perp X_j | V \setminus \{X_i, X_j\}$. In principle, we could set up a much simpler constraint-based search: just test every pairwise conditional independence relationship. Don't have to worry about colliders, acylicity, or Markov equivalence.

In practice, our tests would have quite low power, because we're conditioning on a big set of variables each time. Another idea is to perform *neighborhood search*: for each X_i, estimate the set $\text{Ne}(X_i, \widehat{\mathcal{G}})$ which makes all other variables independent. Then we could "stitch" all the neighborhoods together, somehow resolving conflicts/inconsistencies (e.g., $X_j \in \text{Ne}(X_i, \widehat{\mathcal{G}})$ but $X_i \notin \text{Ne}(X_j, \widehat{\mathcal{G}})$).

Gaussian undirected models

Consider $X \sim N(0, \Sigma)$. $K \equiv \Sigma^{-1}$ is the precision matrix.
Recall: $X_i \perp\!\!\!\perp X_j | V \setminus \{X_i, X_j\}$ if and only if $K_{ij} = 0$.

So, if $X_i \not\perp X_j$ in $\mathcal{G}$ then $K_{ij} = 0$. So, we can learn undirected graphical structure by estimating "structural" zeroes in the precision matrix, or estimating the entire precision matrix and treating small entries as zero. This has motivated a lot of methods in recent years, since estimating precision matrices is something we can do pretty well.

Graphical Lasso (glasso)

$$\widehat{K}^{gl} = \arg\min_{K}\{-\log\det(K) + tr(SK) + \lambda||K||_1\}$$

where S is the sample covariance matrix, λ is a tuning parameter (sparsity penalty), and $||\cdot||_1$ is the ℓ_1-norm, i.e., $||K||_1 = \sum_{i,j\in V}|K_{ij}|$.

Then form the graph $\widehat{\mathcal{G}}^{gl}$ with edge $X_i \sim X_j$ iff $\widehat{K}^{gl}_{ij} \neq 0$.

Graphical Lasso (glasso)

$$\widehat{K}^{gl} = \arg\min_{K}\{-\log\det(K) + tr(SK) + \lambda||K||_1\}$$

where S is the sample covariance matrix, λ is a tuning parameter (sparsity penalty), and $||\cdot||_1$ is the ℓ_1-norm, i.e., $||K||_1 = \sum_{i,j\in V}|K_{ij}|$.

Then form the graph $\widehat{\mathcal{G}}^{gl}$ with edge $X_i \sim X_j$ iff $\widehat{K}^{gl}_{ij} \neq 0$.

$\Rightarrow$ how to choose λ? Usually some variety of cross-validation

Score-based model selection for DAGs

Score-based procedures view structure learning as an optimization problem: assign a score to every structure, and find the structure with the highest score.

A natural score to consider is the posterior probability of a particular structure given the observed data. More a posteriori probable structures "fit" the data "better" and we can imagine searching for the structure which has the highest posterior. How would we define this?

Posteriors over BN models

Recall that a Bayesian network model is a pair $(\mathcal{G}, \mathcal{P})$ where $\mathcal{P}$ is a set of distributions that factorize wrt $\mathcal{G}$. Consider a *parameterized* Bayesian network model which we instead write $(\mathcal{G}, \theta_{\mathcal{G}})$, where $\theta_{\mathcal{G}}$ is the set of parameters which index the distributions in $\mathcal{P}$.

We define the postior probability of a structure $\mathcal{G}$ given data D as

$$p(\mathcal{G}|D) = \frac{p(D|\mathcal{G})p(\mathcal{G})}{p(D)}$$

where $p(\mathcal{G})$ is a structure prior, $p(D|\mathcal{G}) = \int_{\Theta_{\mathcal{G}}} p(D|\theta_{\mathcal{G}}, \mathcal{G})p(\theta_{\mathcal{G}}|\mathcal{G})d\theta_{\mathcal{G}}$, and $p(\theta_{\mathcal{G}}|\mathcal{G})$ is a parameter prior. $p(D|\mathcal{G})$ is called the *marginal likelihood*. Note that the denominator $p(D)$ is the same for every structure so does not play a role in structure selection; we can thus ignore it.

Bayesian score

We could follow this reasoning to define a model score $S(\mathcal{G}, D)$ which takes a candidate structure $\mathcal{G}$ and a dataset D and returns a number proportional to the posterior (ignoring the denominator): the Bayesian score $S(\mathcal{G}, D) \equiv \log p(D|\mathcal{G}) + \log p(\mathcal{G})$. We could even assume that every DAG in the space of DAGs has equal prior probability, to make that part easy.

That still leaves us with trying to evaluate $p(D|\mathcal{G}) = \int_{\Theta_\mathcal{G}} p(D|\theta_\mathcal{G}, \mathcal{G}) p(\theta_\mathcal{G}|\mathcal{G}) d\theta_\mathcal{G}$ which depends on a prior for every $\theta_\mathcal{G}$.

$\Rightarrow$ this is very difficult to calculate outside of special cases.

Bayesian score

There was a lot of work in the early 90's on how to evaluate the marginal likelihood for special cases like binary (multinomial) or Gaussian random variables. Turns out if you don't pick the parameter prior right, you probably can't evaluate that integral.

For example, a lot of work was done (in the binary case) with the Dirchelet prior, $\theta_{\mathcal{G}} \sim Dir(\alpha)$ which has a density proportional to $\prod_{i=1}^{n} \theta_i^{\alpha_i - 1}$. This isn't intuitive, but the Dirchelet prior is conjugate for the multinomial model, thus making $p(D|\mathcal{G})$ relatively easy to compute. There are few other examples of convenient priors like this.

BIC score

These days, it much more likely that one would use an approximation to the Bayesian score called the BIC criterion. More specifically, for distributions in the exponential family the BIC is an approximation to the marginal likelihood $p(D|\mathcal{G})$ under some weak assumptions about the prior).

$$\log p(D|\mathcal{G}) \approx \ell(D; \widehat{\theta}_{\mathcal{G}}) - \frac{d}{2} \log n + O(1)$$

where $\ell(\cdot)$ is the log-likelihood, $\widehat{\theta}_{\mathcal{G}}$ is the maximum likelihood estimate of the parameters, n is the sample size, and d is the model dimension. Assume a uniform prior over graphical structures we can just use this approximation as our model score:

$$S(\mathcal{G}, D) \equiv \ell(D; \widehat{\theta}_{\mathcal{G}}) - \frac{d}{2} \log n$$

BIC score interpretation

What's the interpretation here? If we assume a uniform prior over structures and a "smooth" prior over parameters, maximizing the BIC score is approximately equivalent to maximizing the posterior probability of the graph structure, given the data (that is distributed in the exponential family). That's the Bayesian interpretation.

Alternatively, we can view the BIC score as measure of model "fit" which is a penalized MLE – the second term penalizes complex models, preferring models of smaller dimension. Why? If we just used the MLE as measure of model fit, we would always choose the most complex graph, overfitting the data. The penalty prevents overfitting.

BIC score interpretation

You may wonder what would be the non-Bayesian justification of this *particular* sparsity penalty: $\frac{d}{2} \log n$. The answer is that this penalty makes the score consistent, i.e., maximizing the BIC score will select the true model in the limit. Some other penalties you may come up with will not be consistent (example: the Aikake Information Criterion or AIC score which has a penalty $\propto d$, independent of n, is not consistent!)

The BIC score is also easy to evaluate, because we know how to do MLE. For example: When $X \sim N(0, \Sigma)$

$$S(\mathcal{G}, D) = \frac{n}{2} \log |\hat{\Sigma}| - \frac{d}{2} \log n$$

What might you do if computational limitations weren't an issue?

Brute force enumerate all possible DAGs, score each, and select the highest scoring one.

$$X_1 \rightarrow X_2 \rightarrow X_3 \qquad X_1 \leftarrow X_2 \leftarrow X_3 \qquad X_1 \leftarrow X_2 \rightarrow X_3 \qquad X_1 \rightarrow X_2 \leftarrow X_3$$

$$\Downarrow \qquad\qquad \Downarrow \qquad\qquad \Downarrow \qquad\qquad \Downarrow$$

$$S(\mathcal{G}_1, D) \qquad S(\mathcal{G}_2, D) \qquad S(\mathcal{G}_3, D) \qquad S(\mathcal{G}_4, D)$$

What might you do if computational limitations weren't an issue?

Brute force enumerate all possible DAGs, score each, and select the highest scoring one.

$$X_1 \rightarrow X_2 \rightarrow X_3 \qquad X_1 \leftarrow X_2 \leftarrow X_3 \qquad X_1 \leftarrow X_2 \rightarrow X_3 \qquad X_1 \rightarrow X_2 \leftarrow X_3$$

$$\Downarrow \qquad\qquad \Downarrow \qquad\qquad \Downarrow \qquad\qquad \Downarrow$$

$$S(\mathcal{G}_1, D) \qquad S(\mathcal{G}_2, D) \qquad S(\mathcal{G}_3, D) \qquad S(\mathcal{G}_4, D)$$

$$S(\mathcal{G}_1, D) = S(\mathcal{G}_2, D) = S(\mathcal{G}_3, D)$$

Greedy score-based model selection

Greedy score-based algorithms use **stepwise optimization** of the BIC score in such a way that we don't have to traverse the whole space of graphical structures, but which is still globally optimal.

Greedy score-based model selection

Greedy score-based algorithms use **stepwise optimization** of the BIC score in such a way that we don't have to traverse the whole space of graphical structures, but which is still globally optimal.

The Greedy Equivalence Search (GES) algorithm operates in the space of CPDAGs, to respect Markov equivalence.

Score-based learning: the GES algorithm

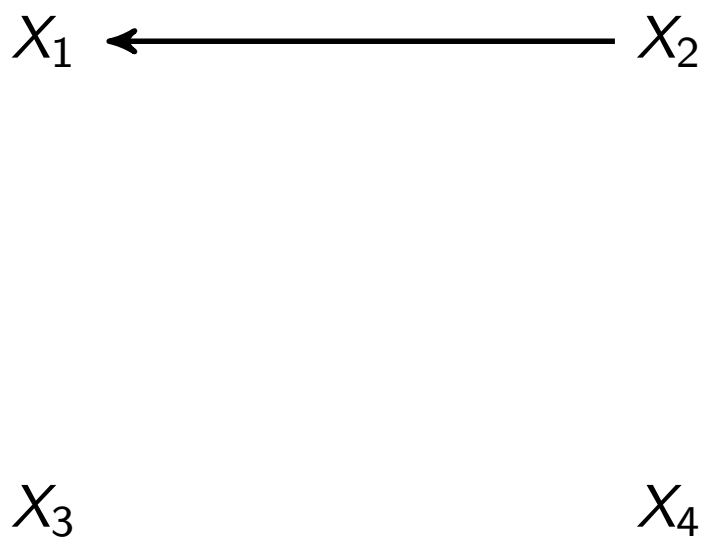
Adds and removes directed edges to optimize a BIC score.

X_1 X_2

X_3 X_4

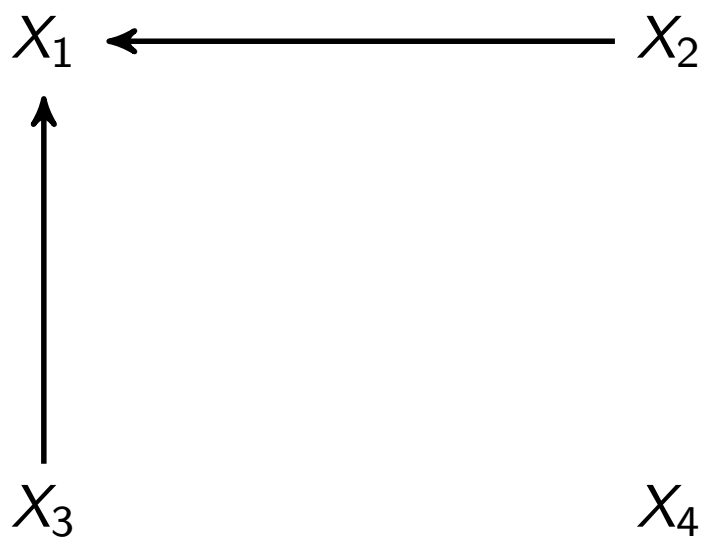
Score-based learning: the GES algorithm

Adds and removes directed edges to optimize a BIC score.



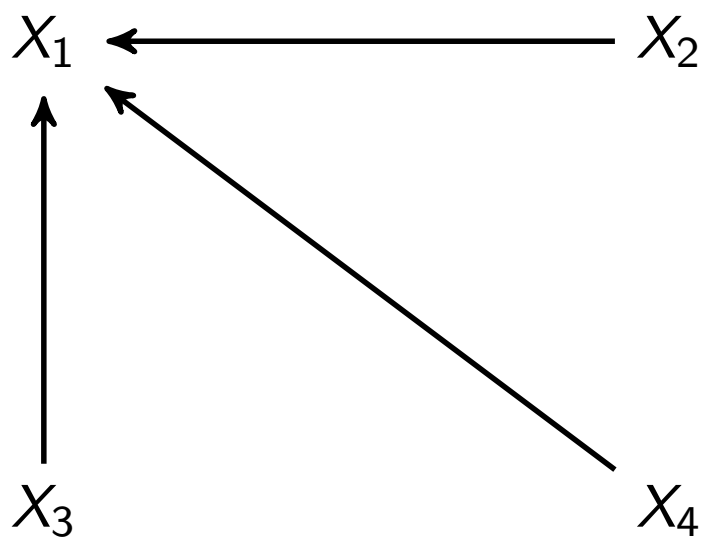
Score-based learning: the GES algorithm

Adds and removes directed edges to optimize a BIC score.



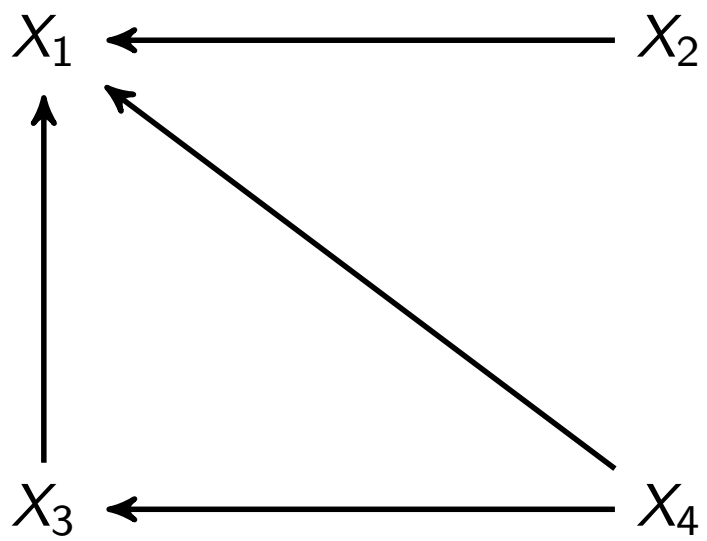
Score-based learning: the GES algorithm

Adds and removes directed edges to optimize a BIC score.



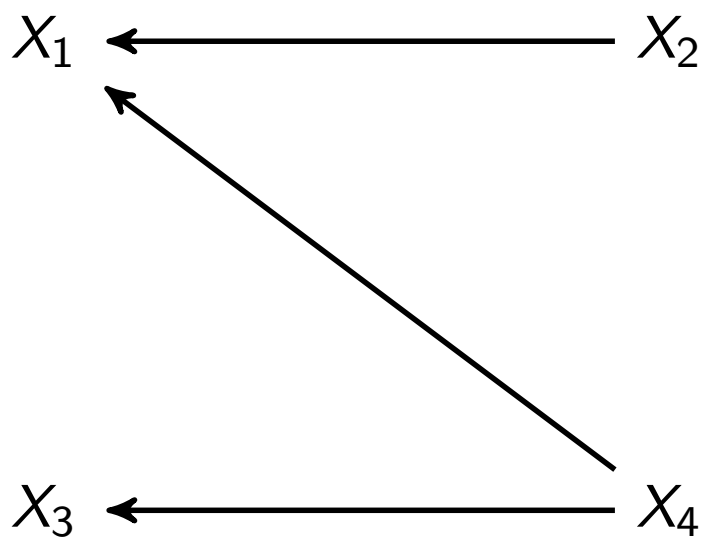
Score-based learning: the GES algorithm

Adds and removes directed edges to optimize a BIC score.



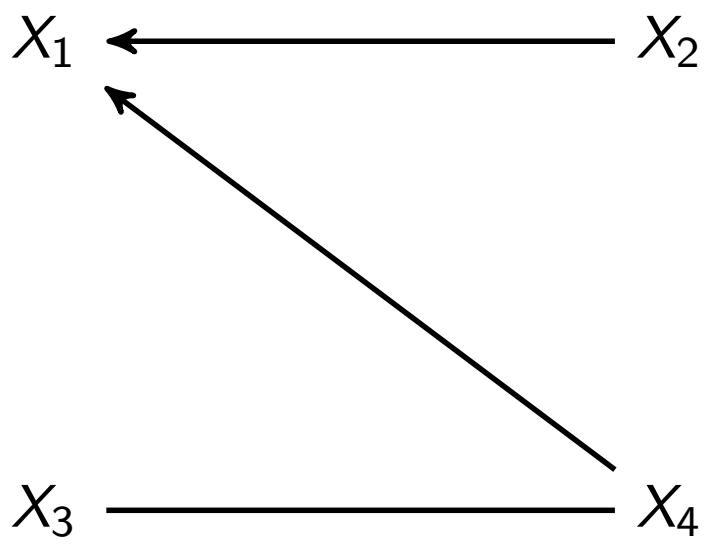
Score-based learning: the GES algorithm

Adds and removes directed edges to optimize a BIC score.



Score-based learning: the GES algorithm

Adds and removes directed edges to optimize a BIC score.



Algorithm 0.2: GES(SCORE)

Input: Samples of the vector $X = (X_1, ..., X_p)$
Output: CPDAG $\mathcal{G}$

1. Form the empty graph $\mathcal{G}$ on vertex set X.
2. Let $S(\mathcal{G}, D)$ be the SCORE for $\mathcal{G}$ with data D.
3. $\langle \mathcal{G}, S \rangle \leftarrow$ FORWARDEQUIVALENCESEARCH$(\mathcal{G}, S)$
4. $\mathcal{G} \leftarrow$ BACKWARDEQUIVALENCESEARCH$(\mathcal{G}, S)$
5. **return** $\mathcal{G}$.

Algorithm 0.3: FORWARDEQUIVALENCESEARCH$(\mathcal{G}, S)$

Input: Samples of the vector $X = (X_1, ..., X_p)$
Output: CPDAG $\mathcal{G}$, SCORE S

1. **while** $E_0 \neq \emptyset$
2. $\quad E_0 \leftarrow T_0 \leftarrow \emptyset$. $S_0 \leftarrow 0$.
3. $\quad$ **for each** edge $E = X_i \rightarrow X_j$ s.t. $X_i \notin \text{Adj}(X_j, \mathcal{G})$
4. $\quad\quad$ Let $T' \leftarrow$ vertices X_k s.t. $X_k - X_j$ and $X_k \notin \text{Adj}(X_i, \mathcal{G})$
5. $\quad\quad$ **for each** subset $T \in T'$
6. $\quad\quad\quad \mathcal{G}' \leftarrow$ a DAG in $\mathcal{G}$
7. $\quad\quad\quad S' \leftarrow S +$ SCOREEDGEADDITION$(\mathcal{G}, E, T)$
8. $\quad\quad\quad$ **if** $S' > S$ and $S' > S_0$ and VALIDINSERT$(\mathcal{G}, E, T)$

$$\textbf{then} \begin{cases} E_0 \leftarrow E \\ T_0 \leftarrow T \\ S_0 \leftarrow S' \end{cases}$$

9. $\quad\quad$ **end**
10. $\quad$ **end**
11. $\quad$ **if** $E_0 \neq \emptyset$

$$\textbf{then} \begin{cases} \text{Add } E_0 \text{ to } \mathcal{G}. \\ \textbf{for each } T \in T_0 \textbf{ if } T - X_i \text{ in } \mathcal{G}, \text{ orient } T - X_i \text{ as } T \rightarrow X_i. \\ S \leftarrow S_0 \\ \mathcal{G} \leftarrow \text{REBUILD}(\mathcal{G}) \end{cases}$$

13. **end**
14. **return** $\langle \mathcal{G}, S \rangle$.

Algorithm 0.4: BackwardEquivalenceSearch$(\mathcal{G}, S)$

Input: Samples of the vector $X = (X_1, ..., X_p)$
Output: CPDAG $\mathcal{G}$

while $E_0 \neq \emptyset$
  $E_0 \leftarrow H_0 \leftarrow \emptyset$. $S_0 \leftarrow 0$.
  for each edge E between X_i and X_j in $\mathcal{G}$
    Let $H' \leftarrow$ vertices X_k s.t. $X_k - X_j$ and $X_k \in \text{Adj}(X_i, \mathcal{G})$
    for each subset $H \in H'$
      $\mathcal{G}' \leftarrow$ a DAG in $\mathcal{G}$
      $S' \leftarrow S + \text{ScoreEdgeDeletion}(\mathcal{G}, E, H)$
      if $S' > S$ and $S' > S_0$ and $\text{ValidDelete}(\mathcal{G}, E, H)$

$$\textbf{then} \begin{cases} E_0 \leftarrow E \\ H_0 \leftarrow H \\ S_0 \leftarrow S' \end{cases}$$

    end
  end
  if $E_0 \neq \emptyset$

$$\textbf{then} \begin{cases} \text{Remove } E_0 \text{ from } \mathcal{G}. \\ \textbf{for each } H \in H_0 \textbf{ if } X_i - H \text{ in } \mathcal{G}, \text{ orient } X_i - H \text{ as } X_i \rightarrow H. \\ S \leftarrow S_0 \\ \mathcal{G} \leftarrow \text{Rebuild}(\mathcal{G}) \end{cases}$$

end
return $\mathcal{G}$.

For some omitted details see Chickering (2002) "Optimal structure identification with greedy search," *Journal of Machine Learning Research*

Constraint-based learning: the GES algorithm

The "logic" of GES:

- the highest scoring model will, in the limit $n \to \infty$, be a member of the true equivalence class
- by adding and removing edges to incrementally improve the score, can achieve a global optimum
- some rules that (un)orient edges to account for Markov equivalence and enforce acyclicity

Score

To make greedy search for CPDAGs globally optimal (converge on the highest scoring, and thus true, structure in the limit) the score we use should satisfy three abstract properties. It turns out that the BIC score satisfies these properties.

Consistency: the true structure $\mathcal{G}$ will maximize the score, and all structures $\mathcal{G}'$ that are not Markov equivalent to $\mathcal{G}$ will have strictly lower score.

Score-equivalence: If $\mathcal{G}$ is Markov equivalent to $\mathcal{G}'$, then $S(\mathcal{G}, D) = S(\mathcal{G}', D)$.

Decomposability: $S(\mathcal{G}, D) = \sum_{X_i \in V} s(X_i, \text{Pa}(X_i, \mathcal{G}))$

Score properties

Assuming faithfulness, GES is finds a global optimum for any consistent, score-equivalent, and decomposable score. It turns out the key property, which is a consequence of these 3, is *local consistency*:

Let $\mathcal{G}$ be any DAG, and let $\mathcal{G}'$ be the DAG that results from adding the edge $X_i \rightarrow X_j$. A scoring criterion $S(\mathcal{G}, D)$ is locally consistent if the following two properties hold:

1. If $X_i \not\perp\!\!\!\perp X_j | \text{Pa}(X_j, \mathcal{G})$ then $\lim_{n\rightarrow\infty} \mathbb{P}(S(\mathcal{G}', D) > S(\mathcal{G}, D)) = 1$
2. If $X_i \perp\!\!\!\perp X_j | \text{Pa}(X_j, \mathcal{G})$ then $\lim_{n\rightarrow\infty} \mathbb{P}(S(\mathcal{G}', D) < S(\mathcal{G}, D)) = 1$

Score properties

Assuming faithfulness, GES is finds a global optimum for any consistent, score-equivalent, and decomposable score. It turns out the key property, which is a consequence of these 3, is *local consistency*:

Let $\mathcal{G}$ be any DAG, and let $\mathcal{G}'$ be the DAG that results from adding the edge $X_i \rightarrow X_j$. A scoring criterion $S(\mathcal{G}, D)$ is locally consistent if the following two properties hold:

1. If $X_i \not\perp\!\!\!\perp X_j | \text{Pa}(X_j, \mathcal{G})$ then $\lim_{n\rightarrow\infty} \mathbb{P}(S(\mathcal{G}', D) > S(\mathcal{G}, D)) = 1$
2. If $X_i \perp\!\!\!\perp X_j | \text{Pa}(X_j, \mathcal{G})$ then $\lim_{n\rightarrow\infty} \mathbb{P}(S(\mathcal{G}', D) < S(\mathcal{G}, D)) = 1$

Single-edge score differences $\Leftrightarrow$ conditional independence tests

Theory

The BIC score is consistent, score-equivalent (for simple parametric families), and decomposable. Thus it is also locally consistent.

Theorem: Assume the distribution $p(x)$ is Markov and faithful to some DAG $\mathcal{G}$. Let $\mathcal{C}$ be the CPDAG implied by $\mathcal{G}$. Let $\widehat{\mathcal{C}}$ be the output of GES using a locally consistent score. Then $\lim_{n\to\infty} \mathbb{P}(\widehat{\mathcal{C}} = \mathcal{C}) = 1$.

Virtues and drawbacks of score-based search

Virtues:

- easy to parallelize, easy to scale to large numbers of variables
- less prone to statistical errors than constraint-based search, backwards stage can "recover" from some mistakes

Drawbacks:

- need to specify likelihoods, so not so easy to do nonparametrically (mostly restricted to exponential families, though there has been recent work on nonparametric scores)
- no need[2] to pick tuning parameter α
- somewhat less developed statistical properties

[2] but... may pick any positive constant factor to multiply the sparsity penalty (enforce more sparsity), which may be treated as a tuning parameter