Causal Inference
CS 477-677

Learning The Structure Of A Directed Acyclic Graph

Ilya Shpitser

Outline

Review

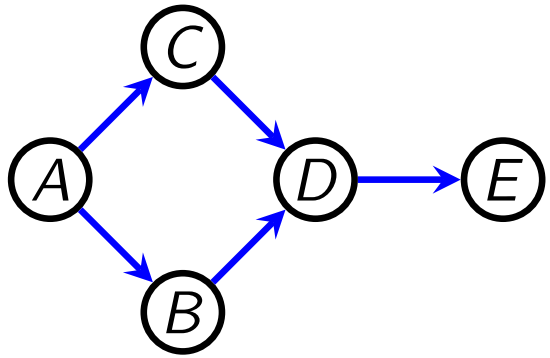
- Counterfactuals as functions of observed data.
- Estimation strategies.
- Causal contrasts: average causal effect, direct and indirect effects.
- Counterfactual reasoning for learning policies.
- Represented causal assumptions via graphs (SWIGs).
- Assumed we knew the graph.
- In many domains, structural knowledge is hard to come by (genomics, for instance).
- What do we do then?

Review

- Counterfactuals as functions of observed data.
- Estimation strategies.
- Causal contrasts: average causal effect, direct and indirect effects.
- Counterfactual reasoning for learning policies.
- Represented causal assumptions via graphs (SWIGs).
- Assumed we knew the graph.
- In many domains, structural knowledge is hard to come by (genomics, for instance).
- What do we do then? Learn the graph!

DAG Models (Refresher)

- Three definitions of a (statistical) DAG model.



- Factorization:

$$p(A, B, C, D, E) = p(E \mid D)p(D \mid B, C)p(C \mid A)p(B \mid A)p(A)$$

- Local Markov property:

$$(C \perp\!\!\!\perp B \mid A), (D \perp\!\!\!\perp A \mid B, C), (E \perp\!\!\!\perp A, B, C \mid D).$$

- Global Markov property: for any $\vec{A}, \vec{B}, \vec{C}$,

  if $\vec{A}$ is d-separated from $\vec{B}$ given $\vec{C}$ in $\mathcal{G}$ then $\vec{A} \perp\!\!\!\perp \vec{B} \mid \vec{C}$ in $p(\vec{V})$.

How To Learn A DAG From Data?

- Global Markov property is a one way implication.
- If we know the DAG, and want to know what it implies about the data, it's perfect!
- But now we want to know what the data implies about the DAG.
- Need another assumption to reverse implication.
- Faithfulness: for any $\vec{A}, \vec{B}, \vec{C}$,

  $\vec{A}$ is d-separated from $\vec{B}$ given $\vec{C}$ in $\mathcal{G}$ if and only if $\vec{A} \perp\!\!\!\perp \vec{B} \mid \vec{C}$ in $p(\vec{V})$.

- Somewhat controversial assumption.

Faithfulness

- Think of faithfulness is an additional property that $p(\vec{V})$ factorizing according to $\mathcal{G}$ may satisfy.
- Most $p(\vec{V})$ will be faithful (unfaithful distributions form a set of measure zero).
- So we should be ok, right? Two caveats:
  - In finite samples, may not be able to tell unfaithful from nearly unfaithful $p(\vec{V})$, and there are many more of those.
  - Nature does not pick distributions at random. May evolve unfaithful situations for evolutionary reasons (homeostasis).
  - In practice: should justify why faithfulness is sensible for the problem.

Structure Learning Algorithms

- Assume there is a graph $\mathcal{G}(\vec{V})$ with k vertices, and a distribution $p(\vec{V})$ factorizing relative to $\mathcal{G}$.
- INPUT: a dataset $\vec{X}_{n\times k}$ (assumed sampled independently from $p(\vec{V})$.
- OUTPUT: a set of graphs consistent with what we know about $\vec{X}_{n\times k}$ (hopefully including $\mathcal{G}$).

Structure Learning Algorithms

- Assume there is a graph $\mathcal{G}(\vec{V})$ with k vertices, and a distribution $p(\vec{V})$ factorizing relative to $\mathcal{G}$.
- INPUT: a dataset $\vec{X}_{n\times k}$ (assumed sampled independently from $p(\vec{V})$.
- OUTPUT: a set of graphs consistent with what we know about $\vec{X}_{n\times k}$ (hopefully including $\mathcal{G}$).
- This is an unsupervised learning problem.
- We want to find a sensible causal description of the data.
- Lots of ways of doing this!

Types Of Structure Learning

- Constraint based learning (today):
  - Find constraints that hold in $\vec{X}_{n \times k}$.
  - Rule out graphs inconsistent with constraints we found.
  - Return what's left.
- Score based learning (next time):
  - Assign a score to any graphical model.
  - Scores typically reward fit, but also regularize (want a concise description of the data).
  - Do search (model selection) for high scoring models given $\vec{X}_{n \times k}$.
- These are asymptotically equivalent, but behave differently in finite samples.
- "Parametric" structure learning:
  - Make strong additional assumptions on $p(\vec{V})$.
  - Orient edges using those assumptions.
  - Examples: additive noise models, structure learning as classification, etc.

Constraint-Based Structure Learning

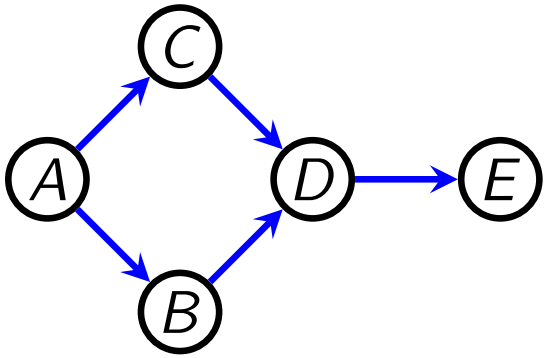
- Say we had a big list of all conditional independence constraints in $p(\vec{V})$ faithful with respect to $\mathcal{G}$.
- What can we say about $\mathcal{G}$?

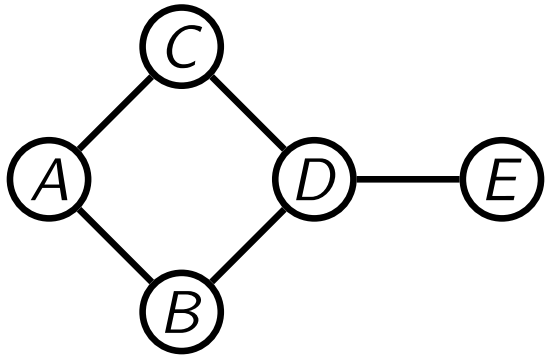
Constraint-Based Structure Learning

- Say we had a big list of all conditional independence constraints in $p(\vec{V})$ faithful with respect to $\mathcal{G}$.
- What can we say about $\mathcal{G}$?
- If for some $A, B \in \vec{V}$, there is some $\vec{S} \subseteq \vec{V} \setminus \{A, B\}$, such that $A \perp\!\!\!\perp B \mid \vec{S}$, then A and B do not have an edge in common!

Constraint-Based Structure Learning

- Say we had a big list of all conditional independence constraints in $p(\vec{V})$ faithful with respect to $\mathcal{G}$.
- What can we say about $\mathcal{G}$?
- If for some $A, B \in \vec{V}$, there is some $\vec{S} \subseteq \vec{V} \setminus \{A, B\}$, such that $A \perp\!\!\!\perp B \mid \vec{S}$, then A and B do not have an edge in common!
- Since we can do this for every pair (A, B), we can use conditional independence constraints to learn the skeleton of the graph:





- We know where the edges are (and are not), but not how they are oriented.

Learning Edge Orientations

- Say we had a big list of conditional independences and the skeleton.
- Can we use this information to orient edges?

Learning Edge Orientations

- Say we had a big list of conditional independences and the skeleton.
- Can we use this information to orient edges?
- Yes!
  - Colliders: if $A \perp\!\!\!\perp B \mid \vec{S}$, and $A - C - B$ is in the skeleton, and $C \notin \vec{S}$, then this can only happen if $A \rightarrow C \leftarrow B$.
  - Acyclicity: if we have $A_1 \rightarrow A_2 \rightarrow \ldots \rightarrow A_k$ and $A_1 - A_k$, then this edge better be $A_1 \rightarrow A_k$.
  - How many rules like that do we need? Maybe a lot more?
  - What if we cannot orient edges?

Issues With Constraint Based Structure Learning

- Unclear how many edge orientation rules we need.
- What if we can't orient some edge?
- How many tests do we need to do in the worst case?

Issues With Constraint Based Structure Learning

- Unclear how many edge orientation rules we need.
- What if we can't orient some edge?
- How many tests do we need to do in the worst case?
- For k vertices, $O(\binom{k}{2} \cdot 2^{k-2})$ tests, must check all possible subsets $\vec{S} \subseteq \vec{V} \setminus \{A, B\}$, for every A, B pair.

Issues With Constraint Based Structure Learning

- Unclear how many edge orientation rules we need.
- What if we can't orient some edge?
- How many tests do we need to do in the worst case?
- For k vertices, $O(\binom{k}{2} \cdot 2^{k-2})$ tests, must check all possible subsets $\vec{S} \subseteq \vec{V} \setminus \{A, B\}$, for every A, B pair.
- How do we learn conditional independence holds from a finite dataset $\vec{X}_{n \times k}$?

Issues With Constraint Based Structure Learning

- Unclear how many edge orientation rules we need.
- What if we can't orient some edge?
- How many tests do we need to do in the worst case?
- For k vertices, $O(\binom{k}{2} \cdot 2^{k-2})$ tests, must check all possible subsets $\vec{S} \subseteq \vec{V} \setminus \{A, B\}$, for every A, B pair.
- How do we learn conditional independence holds from a finite dataset $\vec{X}_{n \times k}$?
- Hypothesis testing (recall: Fisher's test). Tests can have type 1 and type 2 errors. If we orient based on erroneous tests, we get in trouble.

Addressing Structure Learning Issues

- If the graph is very dense (lots of edges), cannot avoid doing many tests.
- If the graph is sparse (few edges), can do better by being clever with tests.
- Try small sets first, remove edges as you go.
- Show a small set of rules orients as many edges as possible.
- Later: how to deal with hypothesis testing?

The PC Algorithm

- Start with a complete undirected graph $\mathcal{G}$ with n vertices.
  - For $i = 0, \ldots, n-2$,
    - For every adjacent pair A, B in $\mathcal{G}$,
    - if $(A \perp\!\!\!\perp B \mid \vec{S})$, $(|\vec{S}| = i,\ \vec{S} \subset \text{nb}_{\mathcal{G}}(A) \cup \text{nb}_{\mathcal{G}}(B) \setminus \{A, B\}))$, remove edge $A - B$ from $\mathcal{G}$, add $\vec{S}$ to **sepset**(A, B), **sepset**(B, A).
- For any non-adjacent A, B such that $A - C - B$ in $\mathcal{G}$, if $C \notin$ **sepset**(A, B), orient $A \rightarrow C \leftarrow B$.
- Repeat as long as orientations are possible:
  - If $A \rightarrow B - C$, and A not adjacent to C, orient $B \rightarrow C$ (no new colliders).
  - If $A_1 \rightarrow A_2 \rightarrow \ldots \rightarrow A_k$, and $A_1 - A_k$, orient $A_1 \rightarrow A_k$ (no cycles).
  - If $A - B \rightarrow C$, $A - D \rightarrow C$, $A - C$, B not adjacent to D, orient $A \rightarrow C$ (no new colliders and no cycles).

Observational Equivalence (Review)

- Consider the following two DAGs:

- Local Markov property gives same independence: $(A \perp\!\!\!\perp C \mid B)$.
- In fact, the only independence in this model.
- If one graph is causal, the other isn't...
- These graphs are called observationally equivalent.
- Big problem for structure learning! Can only learn graph up to equivalence class.

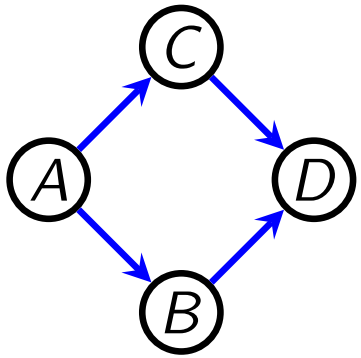
Observational Equivalence

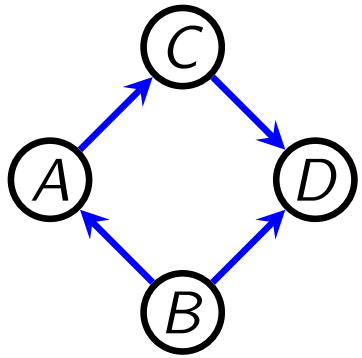
- Simple characterization of equivalence of DAGs:

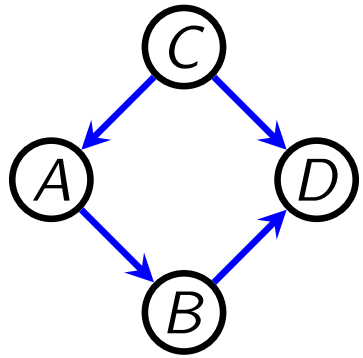
Theorem (Verma and Pearl)

Two distinct DAGs $\mathcal{G}_1$, $\mathcal{G}_2$ represent the same statistical DAG model if and only if they share skeletons and unshielded colliders ($A \to C \leftarrow B$, A, B not adjacent).

- Example:



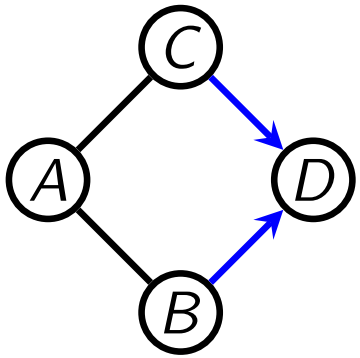




- All DAGs give the same model: $(B \perp\!\!\!\perp C \mid A)$, $(D \perp\!\!\!\perp A \mid B, C)$.

Equivalence Classes Of DAGs And Patterns

- Can represent all DAGs in an equivalence class by a single mixed graph called a pattern.
- Pattern has two types of edges, directed ($\rightarrow$) and undirected ($-$).
- Any DAG in an equivalence class of a pattern can be constructed by orienting $-$ edges without creating new colliders.
- Example: this pattern gives three DAGs on the previous slide.



Completeness Of The PC Algorithm

- Assume the PC algorithm has access to a conditional independence oracle (all tests are correct).
- Then the PC algorithm is complete (always returns a pattern representation of the equivalence class).

Statistical Issues

- How to do conditional independence hypothesis testing?
- If conditioning set $\vec{S}$ is large, need strong assumptions or lots of data.
- In practice, use parametric tests, based on partial correlations, written $\rho_{XY.Z}$.
- Can also use non-parametric tests.

Partial Correlation

- (Pearson's) correlation ρ_{XY} is defined as

$$\rho_{XY} \equiv \frac{Cov(X,Y)}{\sigma_X \cdot \sigma_Y} = \frac{E[X - E[X]] \cdot E[Y - E[Y]]}{\sigma_X \cdot \sigma_Y},$$

  σ_X : standard deviation of X, σ_Y: standard deviation of Y.
- Partial correlation given $\vec{Z}$, $\rho_{XY.\vec{Z}}$ is defined as $\rho_{r_X r_Y}$ where r_X is the residual of (linear) regressing X against $\vec{Z}$, and r_Y is the residual of (linear) regressing Y against $\vec{Z}$.
- Sample version $\hat{\rho}_{XY.\vec{Z}}$ substitutes sample estimates.
- In certain models (e.g. $p(\vec{V})$ is multivariate normal), $\rho_{XY.\vec{Z}} = 0$ is equivalent to $X \perp\!\!\!\perp Y \mid \vec{Z}$.
- Standard tests exist for this in the stats literature.
- The point: polynomial amount of work, modest sample requirements.

Testing No Partial Correlation

- Can test using Fisher's z-transform of sample partial correlation:

$$z(\hat{\rho}_{XY.\vec{Z}}) = 0.5 \cdot \log\left(\frac{1+\hat{\rho}_{XY.\vec{Z}}}{1-\hat{\rho}_{XY.\vec{Z}}}\right).$$

- Reject if test statistic is in "unusual location" in its distribution (approximately normal):

$$\sqrt{N - |\vec{Z}| - 3} \cdot |z(\hat{\rho}_{XY.\vec{Z}})| > \Phi^{-1}(1-\alpha/2).$$

- $\Phi(.)$ is a cdf of normal, $\Phi(.)$ is a quantile function of normal.
- α is significance level (e.g. 95%). Absolute value and $\alpha/2$ because this is a two-sided test.
- $\sqrt{N-3}$ holds for $\vec{Z} = \emptyset$, subtract degrees of freedom for larger $\vec{Z}$.

Structure Learning In Practice

- PC Algorithm can be sensitive to test errors.
- Wrong edges in skeleton propagate to orientations in hard to analyze ways.
- Lots of work on robust versions.
- Lack of uniform consistency: probability of error does not just depend on sample size n (!), also depends on true graph $\mathcal{G}$.
- This is a general issue with structure learning problems.
- What if there are hidden variables?

Next time: Score Based Structure Learning.