Causal Inference
CS 477-677

Causal Models With Hidden Variables

Ilya Shpitser

Outline

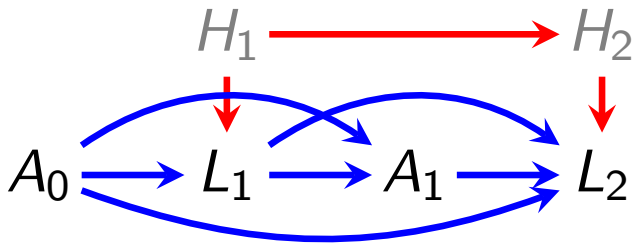
1. Review
2. Hidden Variable Example
3. G-Computation With Hidden Variables
4. Mixed Graphs And Latent Projections
5. Kernels
6. Fixing: Generalizing The G-Formula
7. The ID Algorithm (New Version)
8. Effect Modification

Review

- Causal models of a DAG.
- The g-formula.
- Mediation (splitting causal effects).
- Estimation methods.
- Only talked about dealing with hidden variables using instrumental variables.
- Hidden variables are a common problem in practice.
- Let's talk about how to handle them in general today!

First Example

- Saw this example (without H_1, H_2) in homework extra credit.
- These types of variables are very common
- Think unobserved state where we get to see only parts L_1, L_2 (related to Hidden Markov Models (HMMs)).

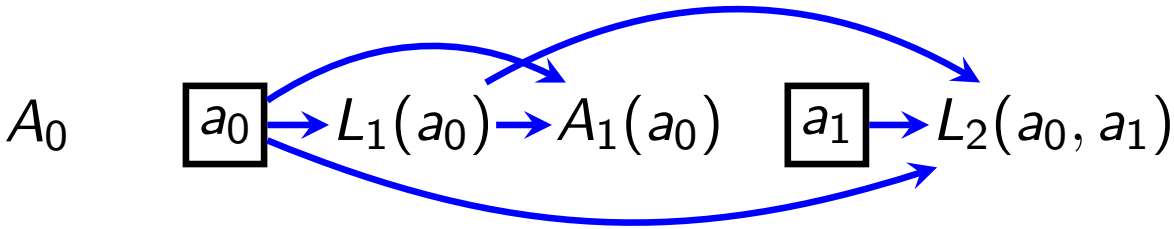


- Interested in the ACE $E[L_2(a_0, a_1)] - E[L_2(a_0', a_1')]$, as before.
- Need $\perp\!\!\!\perp$ assumptions to identify.
- Let's try to construct SWIGs for this problem.

SWIGs With No Hidden Variables

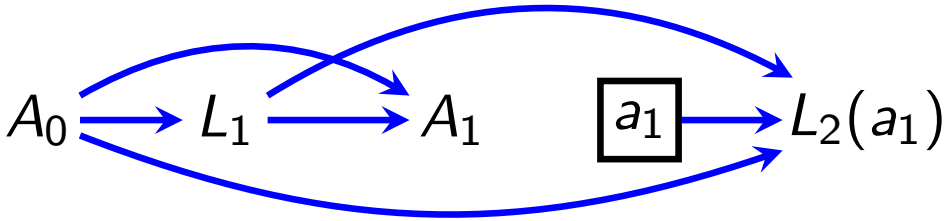
$$\{L_2(a_0, a_1), L_1(a_0)\} \perp\!\!\!\perp A_0$$

due to:



$$L_2(a_1) \perp\!\!\!\perp A_1 \mid L_1, A_0$$

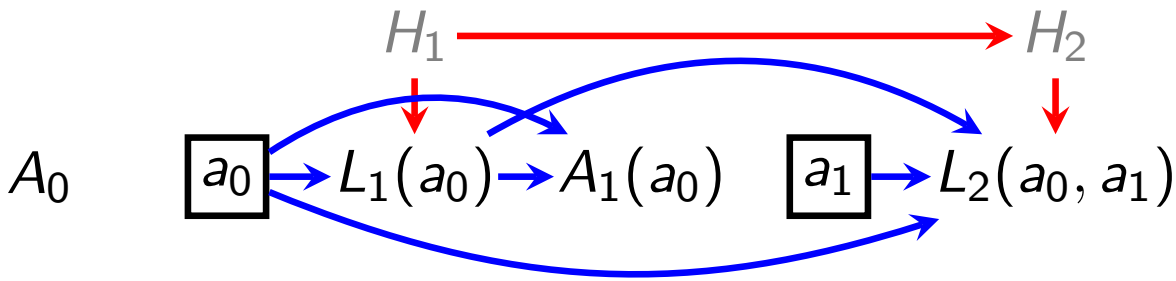
due to:



SWIGs With Hidden Variables

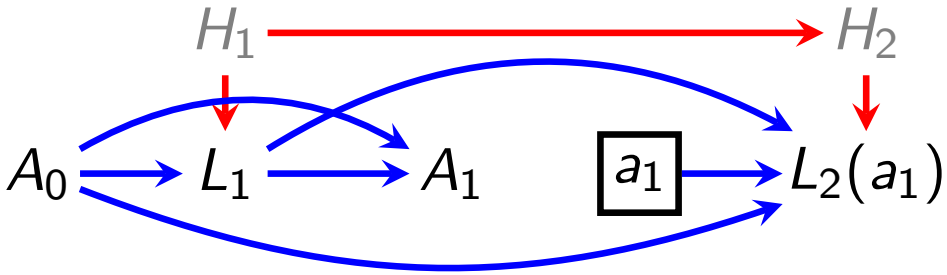
$$\{L_2(a_0, a_1), L_1(a_0)\} \perp\!\!\!\perp A_0$$

due to:



$$L_2(a_1) \perp\!\!\!\perp A_1 \mid L_1, A_0$$

due to:



Same Derivation As Before

$$
\begin{aligned}
p(L_2(a_0, a_1)) &=^{p} \sum_{l_1} p(L_2(a_0, a_1) \mid L_1(a_0) = l_1) p(L_1(a_0) = l_1) \\
&=^{1} \sum_{l_1} p(L_2(a_0, a_1) \mid L_1(a_0) = l_1, a_0) p(L_1(a_0) = l_1 \mid a_0) \\
&=^{c} \sum_{l_1} p(L_2(a_1) \mid l_1, a_0) p(l_1 \mid a_0) \\
&=^{2} \sum_{l_1} p(L_2(a_1) \mid A_1 = a_1, l_1, a_0) p(l_1 \mid a_0) \\
&=^{c} \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0)
\end{aligned}
$$

$$\{L_2(a_0, a_1), L_1(a_0)\} \perp\!\!\!\perp A_0 \tag{1}$$

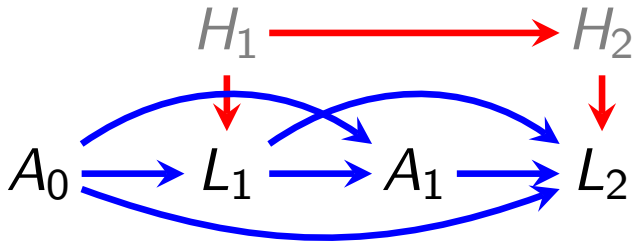
$$L_2(a_1) \perp\!\!\!\perp A_1 \mid L_1, A_0 \tag{2}$$

Hidden Variables: Friends Or Foes

- In machine learning, hidden variables are your friends – they help you to learn good representation of the data
  - Clustering
  - Representation learning
  - Latent variable graphical models (HMMs, Kalman filters, etc.)
- In ML, assume whatever we like about hidden variables to get good generalization error.
- In causal inference, hidden variables are your enemies – they create confounding.
- Saw a simple example of a non-identified ACE due to a hidden U.
- In causal inference, assume the worst – U is very large and complicated.
- Do not want to model U explicitly. Why?
  - No data on U.
  - No way to check if we are wrong – cannot use observed data!
  - Get severe bias if we are wrong.

Modeling Around Hidden Variables

- Under



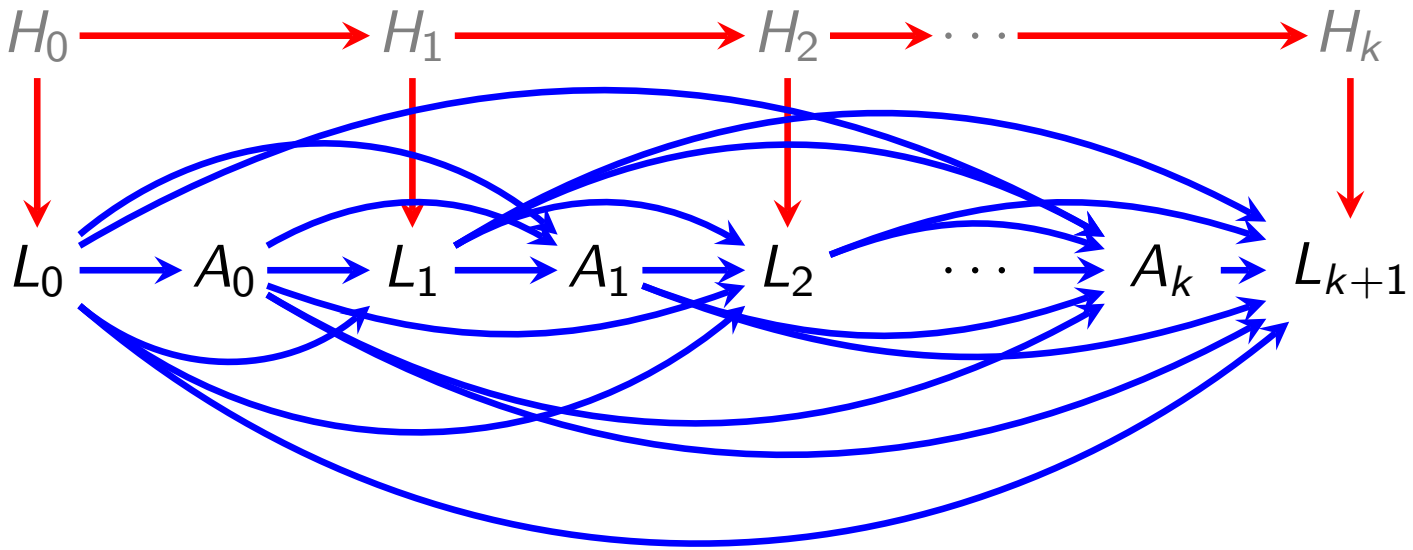
- We get:

$$p(L_2(a_0, a_1)) = \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0)$$

- Note: H_1, H_2 is not mentioned anywhere!
- We assume the worst about H_1, H_2. Can't test any model we assume. Can't cross-validate.
- So model around them!

Games Vs Nature (Hidden Variable Version)

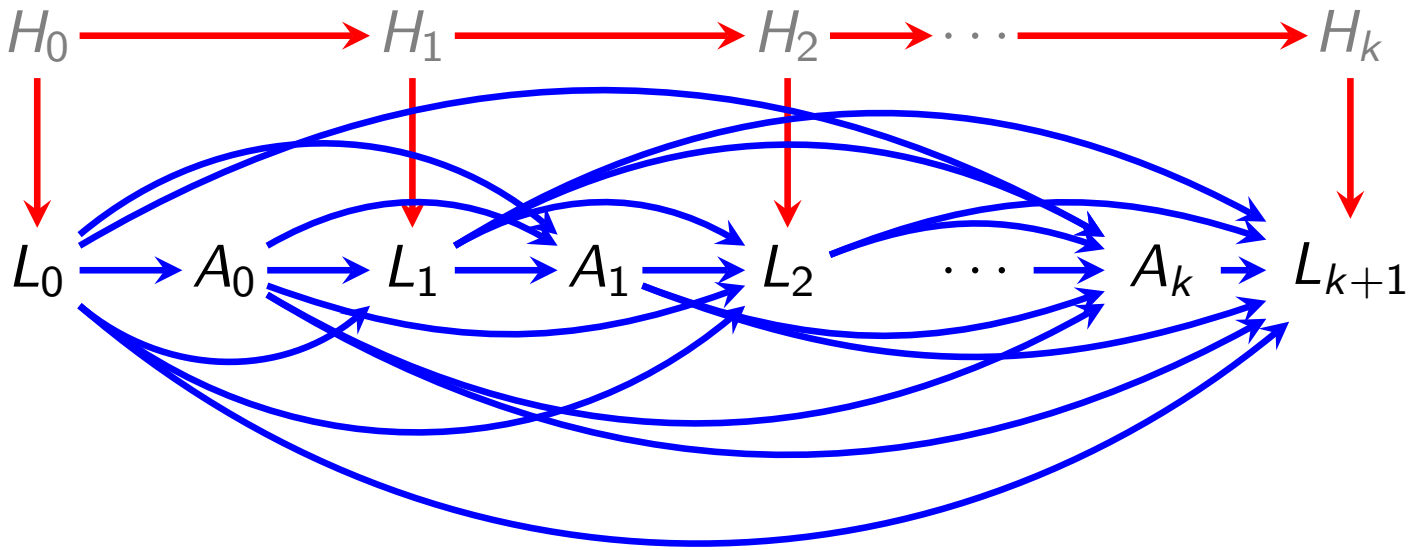
- General version of the "game vs Nature" (k treatments, k outcomes).



- Interested in $E[L_{k+1}(a_1, \ldots, a_k)] - E[L_{k+1}(a'_1, \ldots, a'_k)]$.
- Could identify this with no H. Can we do this now?

Games Vs Nature (Hidden Variable Version)

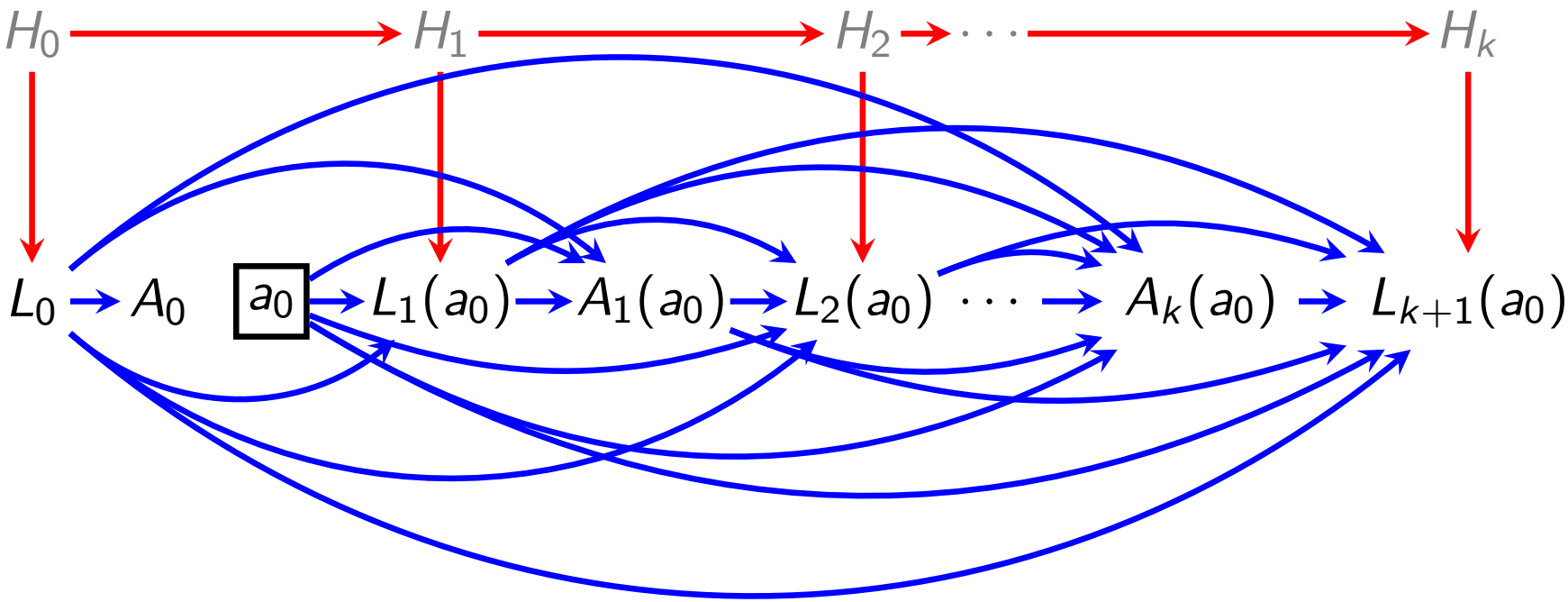
- General version of the "game vs Nature" (k treatments, k outcomes).



- Interested in $E[L_{k+1}(a_1, \ldots, a_k)] - E[L_{k+1}(a'_1, \ldots, a'_k)]$.
- Could identify this with no H. Can we do this now?
- Yes, via the same g-computation algorithm. Why?

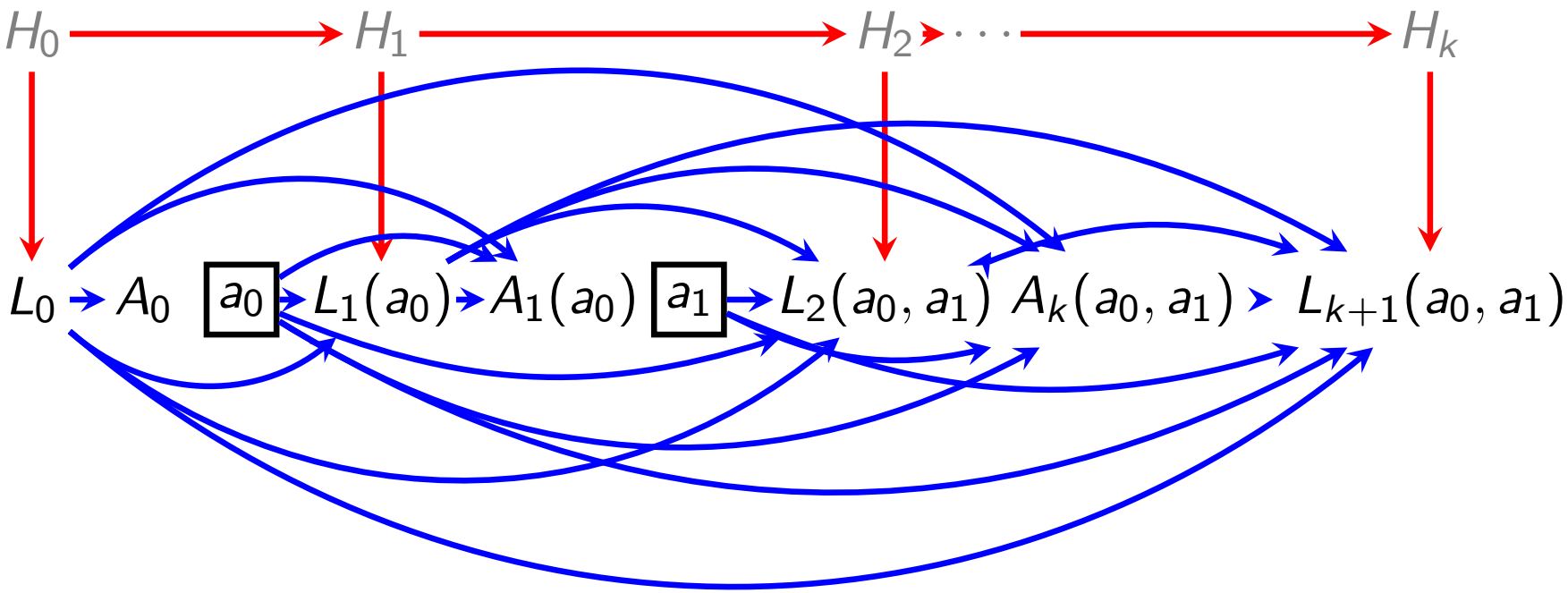
Games Vs Nature (Hidden Variable Version)

- General version of the "game vs Nature" (k treatments, k outcomes).



Games Vs Nature (Hidden Variable Version)

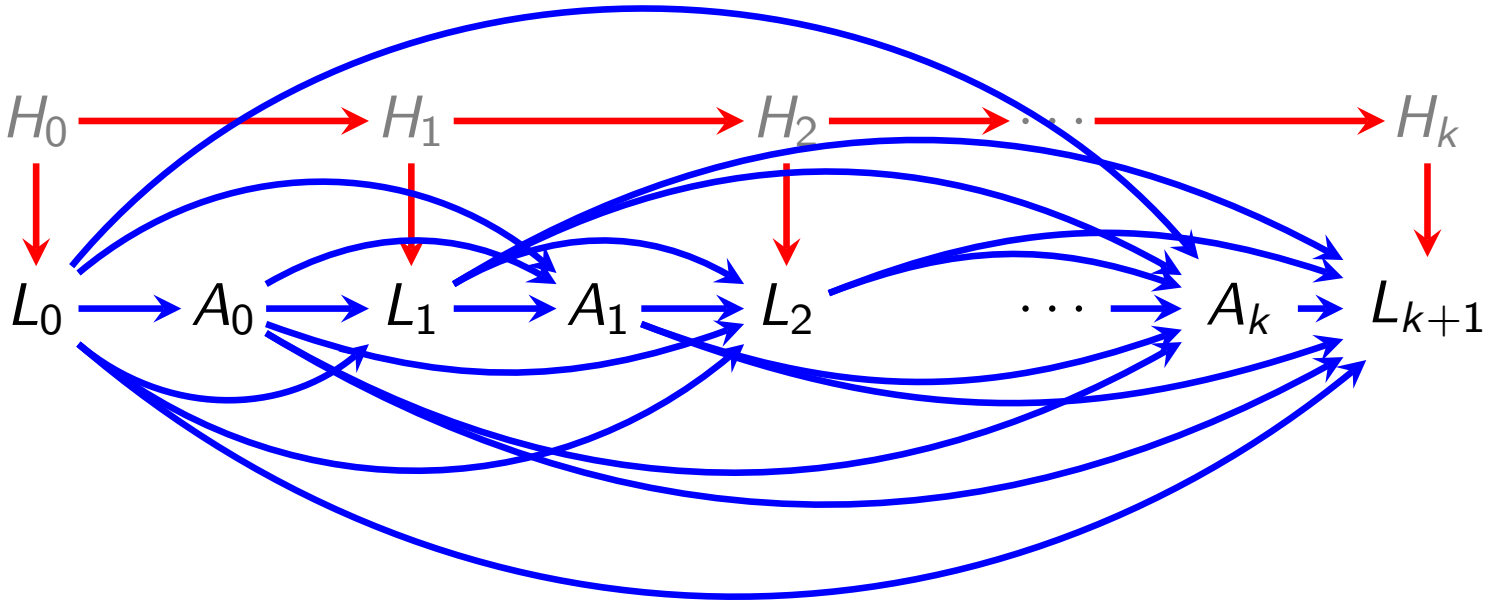
- General version of the "game vs Nature" (k treatments, k outcomes).



G-Computation Assumptions Still Hold)

- The following assumptions (sequential ignorability) still hold via SWIGs:

$$(\forall i \in \{1, \ldots, k+1\}) \left(\{L_i(a_{<i}), \ldots, L_{k+1}(a_{<k+1})\} \perp\!\!\!\perp A_{i-1} \mid \text{past of } A_{i-1}\right).$$



Sequential G-Formula Still Holds

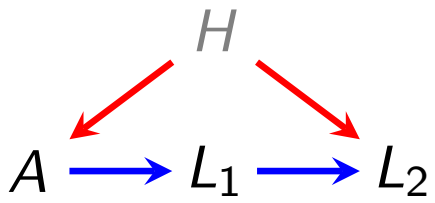
- Since same assumptions hold, derivation from last lecture still works:

$$p(L_{k+1}(a_0, \ldots, a_k)) = \sum_{L_{<(k+1)}} \left(\prod_{j=0}^{k+1} p(L_j \mid L_{<j}, a_{<j}) \right)$$

- Observation due to Robins, 1986.
- Assumptions are much more realistic – we allow a lot of confounding.
- As before, we do not model H_i, assume the worst.
- The big assumption is this: each A_i is determined by the observable past only, not H.
- Often makes sense – doctors decide based on what they observe.

Second Example

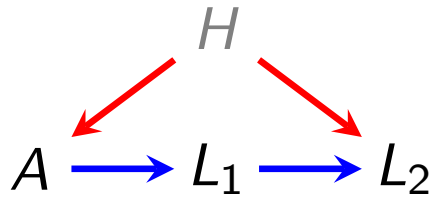
- What if A is determined by H directly? Simple example:



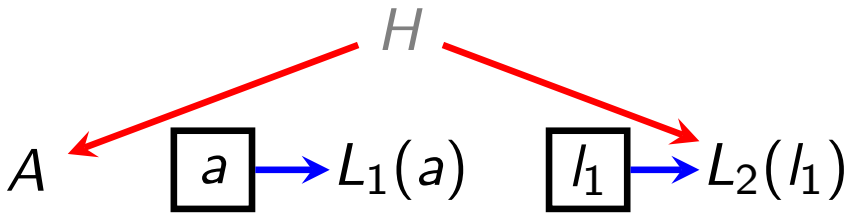
- Interested in the ACE $E[L_2(a)] - E[L_2(a')]$, as before.
- Need $\perp\!\!\!\perp$ assumptions to identify.
- Let's try to construct SWIGs for this problem.

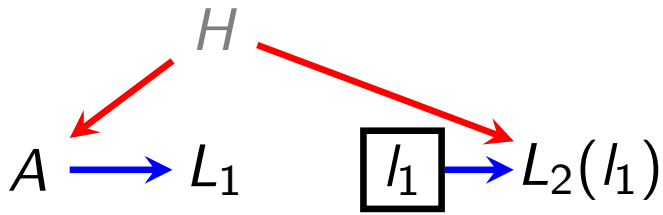
Second Example

- What if A is determined by H directly? Simple example:



- Interested in the ACE $E[L_2(a)] - E[L_2(a')]$, as before.
- Need $\perp\!\!\!\perp$ assumptions to identify.
- Let's try to construct SWIGs for this problem.





(1) $L_1(a) \perp\!\!\!\perp A$

(2) $L_2(l_1) \perp\!\!\!\perp L_1(a)$

(3) $L_2(l_1) \perp\!\!\!\perp L_1 \mid A$

(4) $L_2(l_1, a) = L_2(l_1)$.

Derivation

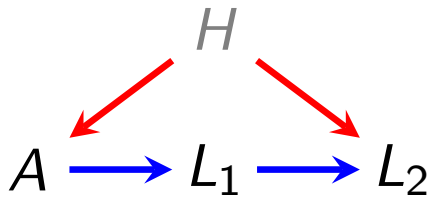
$$
\begin{aligned}
p(L_2(a)) &=^{p} \sum_{l_1} p(L_2(a)) \mid L_1(a) = l_1) p(L_1(a) = l_1) \\
&=^{1} \sum_{l_1} p(L_2(a)) \mid L_1(a) = l_1) p(l_1 \mid a) \\
&=^{c} \sum_{l_1} p(L_2(a, l_1)) \mid L_1(a) = l_1) p(l_1 \mid a) \\
&=^{4} \sum_{l_1} p(L_2(l_1)) \mid L_1(a) = l_1) p(l_1 \mid a) \\
&=^{2} \sum_{l_1} p(L_2(l_1)) p(l_1 \mid a) \\
&=^{3} \sum_{l_1} \left(\sum_{a'} p(L_2 \mid l_1, a') p(a') \right) p(L_1 = l_1 \mid a)
\end{aligned}
$$

(1) $L_1(a) \perp\!\!\!\perp A$ (2) $L_2(l_1) \perp\!\!\!\perp L_1(a)$

(3) $L_2(l_1) \perp\!\!\!\perp L_1 \mid A$ (4) $L_2(l_1, a) = L_2(l_1)$.

Front-Door Formula

- Counterintuitive: there is a latent H causing both A and Y, but we can deal with it (!)



$$p(L_2(a)) = \sum_{l_1} \left(\sum_{a'} p(L_2 \mid l_1, a') p(a') \right) p(L_1 = l_1 \mid a)$$

- Answer does not look like g-formula.
- Shown by Pearl in 1995, called the front-door formula.
- (Pearl called adjustment formula/conditional ignorability "back-door formula.")
- We know sometimes H leads to non-identification.
- When can we get identification even if H is there?

Dealing With Hidden Variables in General

- First observation: different hidden variable DAGs can lead to same derivation:

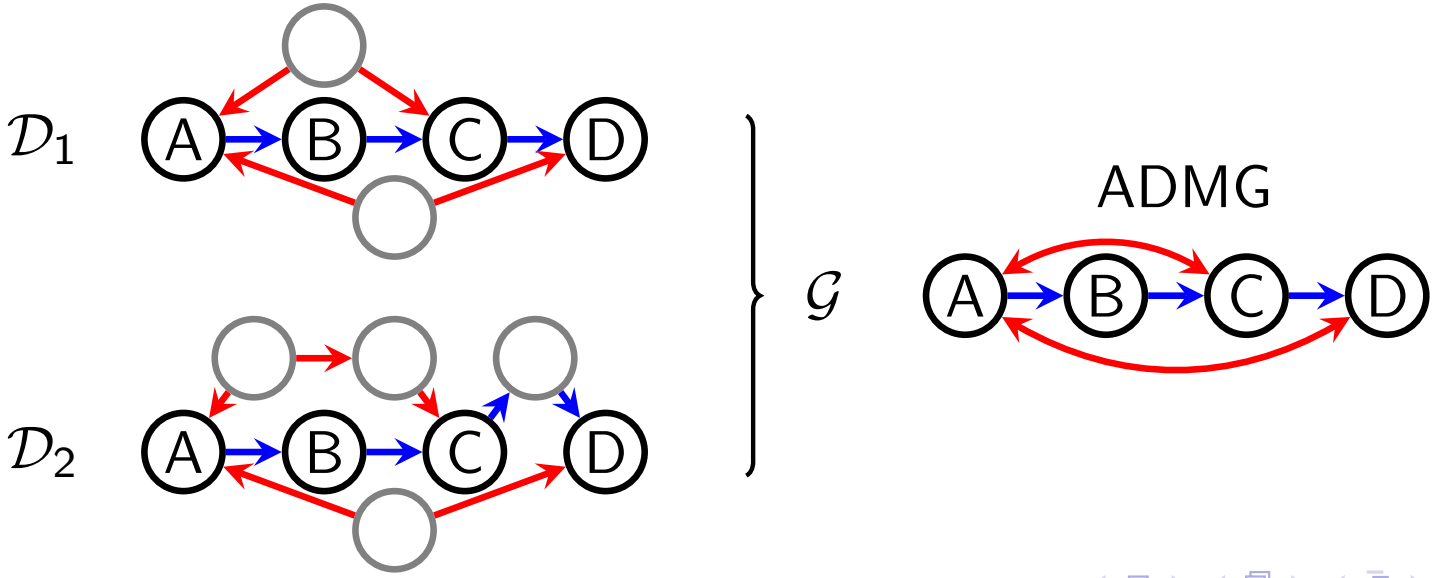
- Why? Only use d-separation in SWIGs where we don't mention H.
- Want to combine multiple hidden variable DAGs (that share these d-separation statements in all SWIGs) into one graph.

Latent Projections

- Will use a generalization of a DAG called Acyclic Directed Mixed Graphs (ADMGs).
- Only has vertices corresponding to observed variables (all H vertices removed).
- Has two types of edges:
  - $A \rightarrow B$ (meaning $A \rightarrow H_1 \rightarrow H_2 \rightarrow \ldots \rightarrow B$ via a path only through H)
  - $A \leftrightarrow B$ (meaning $A \leftarrow H_1 \rightarrow H_2 \rightarrow \ldots \rightarrow B$ via a path with no colliders only through H)

Latent Projections

- Will use a generalization of a DAG called Acyclic Directed Mixed Graphs (ADMGs).
- Only has vertices corresponding to observed variables (all H vertices removed).
- Has two types of edges:
  - $A \to B$ (meaning $A \to H_1 \to H_2 \to \ldots \to B$ via a path only through H)
  - $A \leftrightarrow B$ (meaning $A \leftarrow H_1 \to H_2 \to \ldots \to B$ via a path with no colliders only through H)



More on Latent Projections

- Above way of construct edges is called the latent projection.
- Allow one $\rightarrow$ and $\leftrightarrow$ between a node pair.
- Meaning: both direct causation and unobserved confounding exists.

More on Latent Projections

- Above way of construct edges is called the latent projection.
- Allow one $\rightarrow$ and $\leftrightarrow$ between a node pair.
- Meaning: both direct causation and unobserved confounding exists.

- Construct SWIGs from ADMGs the same way:

More on Latent Projections

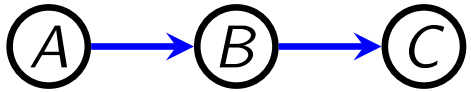
- Above way of construct edges is called the latent projection.
- Allow one $\rightarrow$ and $\leftrightarrow$ between a node pair.
- Meaning: both direct causation and unobserved confounding exists.

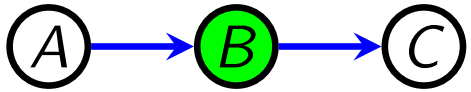
- Construct SWIGs from ADMGs the same way:

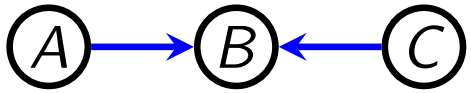
- How do we generalize d-separation from DAGs with $\rightarrow$ to ADMGs with both $\rightarrow$ and $\leftrightarrow$?

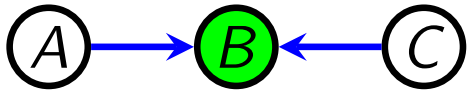
m-separation ("m" for "mixed.")

- Treat colliders and non-colliders the same as with d-separation.
- As before, can condition on descendants of colliders instead.

$A \not\perp\!\!\!\perp C$: 

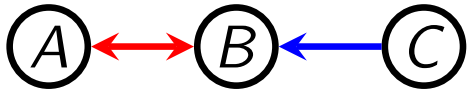
$A \perp\!\!\!\perp C \mid B$: 

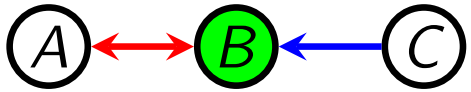
$A \perp\!\!\!\perp C$: 

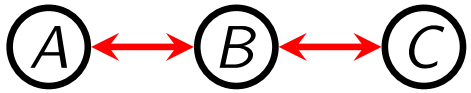
$A \not\perp\!\!\!\perp C \mid B$: 

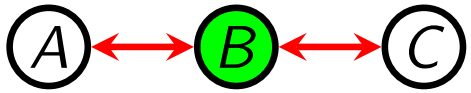
$A \not\perp\!\!\!\perp C$: 

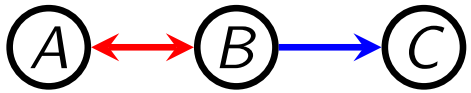
$A \perp\!\!\!\perp C \mid B$: 

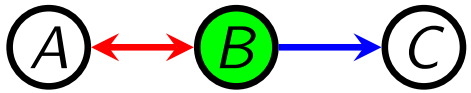
$A \perp\!\!\!\perp C$: 

$A \not\perp\!\!\!\perp C \mid B$: 

$A \perp\!\!\!\perp C$: 

$A \not\perp\!\!\!\perp C \mid B$: 

$A \not\perp\!\!\!\perp C$: 

$A \perp\!\!\!\perp C \mid B$: 

Identification In Hidden Variable DAG Models

We have all the ingredients now:

- A lifted representation of hidden variable DAG models (latent projection ADMGs).
- A way of constructing SWIGs to think about counterfactual independence.
- Rules of inference: consistency, probability manipulations, adding/dropping things due to independence.
- Enough to deal with any specific graph we get.

Identification In Hidden Variable DAG Models

We have all the ingredients now:

- A lifted representation of hidden variable DAG models (latent projection ADMGs).
- A way of constructing SWIGs to think about counterfactual independence.
- Rules of inference: consistency, probability manipulations, adding/dropping things due to independence.
- Enough to deal with any specific graph we get.

Questions:

- When is identification possible and when is it not?
- Can we construct a systematic algorithm to check this, and give us a formula when it is possible?

The Causal Effect Identification Problem

- Implicitly posed by Rubin in the 1970s.
- Explicitly posed by Pearl in the 1990s.
- Pearl came up with "do-calculus" in 1994: 3 rules to manipulate: $p(Y \mid \text{do}(a), Z) \equiv p(Y(a) \mid Z(a))$ using graphs.
- Tian (Pearl's student) came up with a closed form algorithm (2 pages), got simplified to the **ID** algorithm (7 lines).
- Will show you the new way – 1 line formula.
- Equivalent to **ID** and do-calculus, but simpler.
- Complete: either succeeds or fails. If succeeds, gives formula. If fails, effect not identified. So no other algorithm can succeed either.
- Generalizes: adjustment formula, front-door formula, g-computation, etc.

One Line **ID**: Good News/Bad News

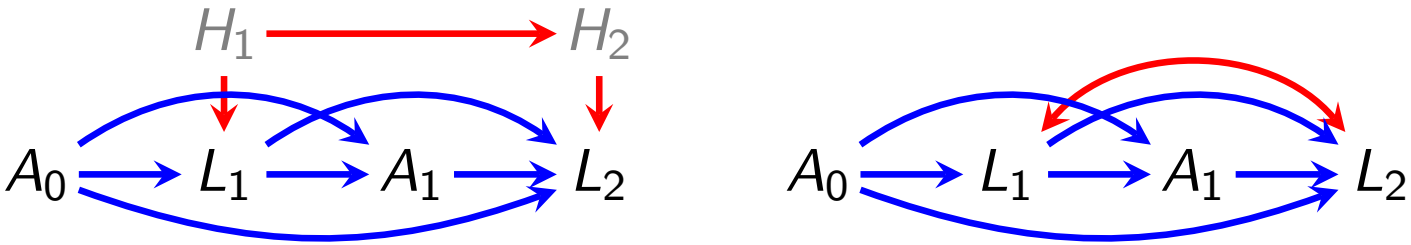
- Good news: a lot simpler then even five years ago.
- Bad news: still somewhat subtle.
- Bad news: will need to introduce quite a bit of new stuff.
- Good news: new stuff will help you really understand hidden variable models. In general, not just in causal settings.

One Line **ID**: Good News/Bad News

- Good news: a lot simpler then even five years ago.
- Bad news: still somewhat subtle.
- Bad news: will need to introduce quite a bit of new stuff.
- Good news: new stuff will help you really understand hidden variable models. In general, not just in causal settings.
  - Kernels: moving beyond conditional distributions.
  - Districts: how do hidden variable models factorize.
  - Fixing: generalizing g-formula to the hidden variable case.
  - Fixable: when can we apply fixing.
  - Putting it all together.

Kernels: Generalized Conditional Distributions

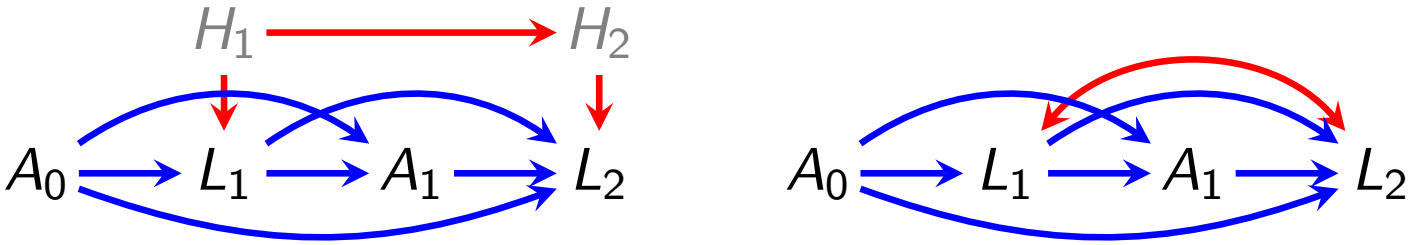
- Recall two treatment case:



$$p(L_2(a_0, a_1)) = \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0)$$

Kernels: Generalized Conditional Distributions

- Recall two treatment case:



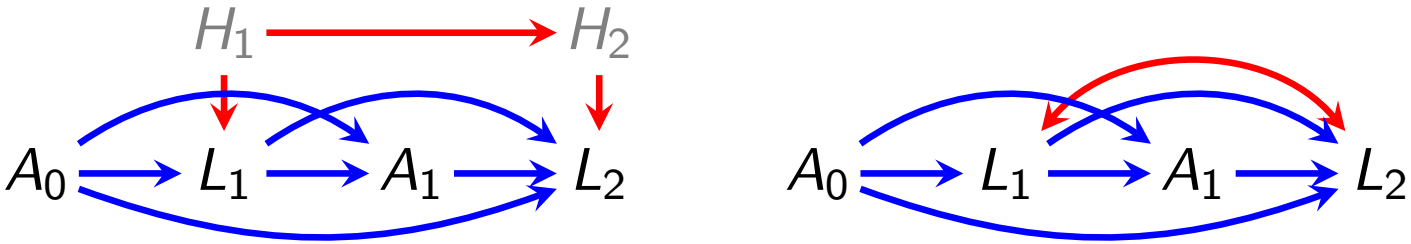
$$p(L_2(a_0, a_1)) = \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0)$$

- Think of this as a "conditional" distribution (map from values a_0, a_1 to a normalized distribution):

$$q(L_2 \mid a_0, a_1) \equiv \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0).$$

Kernels: Generalized Conditional Distributions

- Recall two treatment case:



$$p(L_2(a_0, a_1)) = \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0)$$

- Think of this as a "conditional" distribution (map from values a_0, a_1 to a normalized distribution):

$$q(L_2 \mid a_0, a_1) \equiv \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0).$$

- This was not obtained from $p(L_2, A_1, A_0)$ by conditioning on A_1, A_0. If we did that we would get:

$$p(L_2 \mid a_0, a_1) = \sum_{l_1} p(L_2 \mid a_1, l_1, a_0) p(l_1 \mid a_0, a_1)$$

Kernels: Generalized Conditional Distributions

- Kernels $q(\vec{V} \mid \vec{w})$ act just like a conditional probability $p(\vec{V} \mid \vec{w})$.
- Every value $\vec{w}$ maps to a distribution (density) over $\vec{V}$:
  - Normalized to sum to 1.
  - Every kernel probability $q(\vec{V} = \vec{v} \mid \vec{w}) \geq 0$.
- In general not obtained from $p(\vec{W}, \vec{V})$ by conditioning on $\vec{W}$.
- Kernels are important in hidden variable models, as we will see.
- Conditional independences in kernels are very interesting things! Will come back to this later.
- Define marginalization, conditioning normally. For any $\vec{A} \subseteq \vec{V}$,

$$q(\vec{A} \mid \vec{W}) \equiv \sum_{\vec{V} \setminus \vec{A}} q(\vec{V} \mid \vec{W})$$

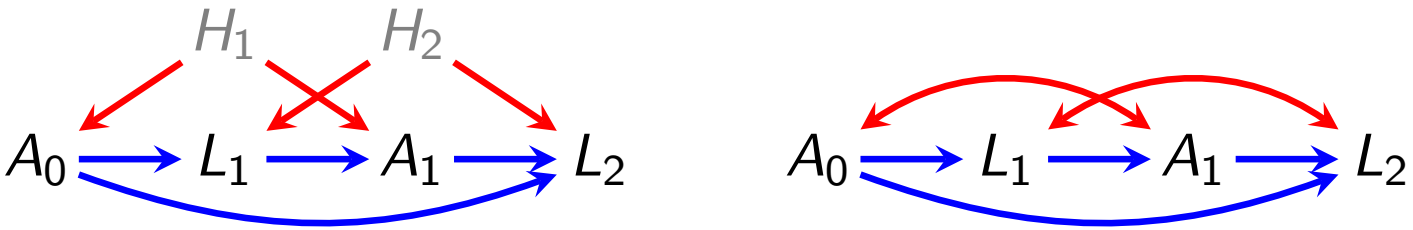
$$q(\vec{V} \setminus \vec{A} \mid \vec{A}, \vec{W}) \equiv \frac{q(\vec{V} \mid \vec{W})}{q(\vec{A} \mid \vec{W})}.$$

How To Factorize Hidden Variable DAGs?

- A DAG gives us Markov factorization:

$$p(\vec{V}) = \prod_{V \in \vec{V}} p(V \mid \mathrm{pa}_{\mathcal{G}}(V)).$$

- Want to factorize a hidden variable DAG, but cannot use H:

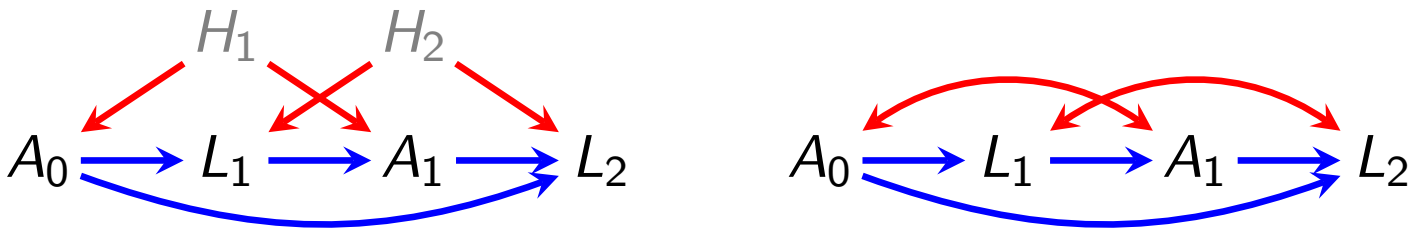


- Don't get to use e.g. $p(L_2 \mid A_1, H_2)$.
- What do we do?

Can Use Kernels To Factorize

- Factorize observed marginal as:

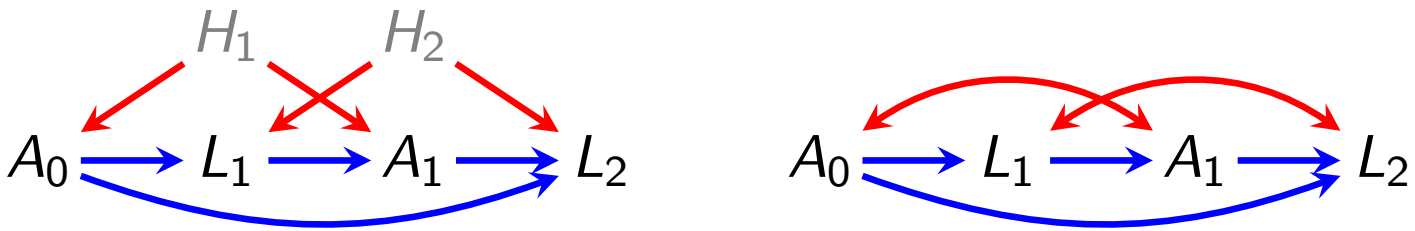
$$\begin{aligned} p(A_0, L_1, A_1, L_2) &= q(A_0, A_1 \mid L_1) q(L_2, L_1 \mid A_0, A_1) \\ &= (p(A_1 \mid L_1, A_0) p(A_0)) \cdot (p(L_2 \mid A_1, L_1, A_0) p(L_1 \mid A_0)) \,. \end{aligned}$$



Can Use Kernels To Factorize

- Factorize observed marginal as:

$$\begin{aligned} p(A_0, L_1, A_1, L_2) &= q(A_0, A_1 \mid L_1) q(L_2, L_1 \mid A_0, A_1) \\ &= (p(A_1 \mid L_1, A_0) p(A_0)) \cdot (p(L_2 \mid A_1, L_1, A_0) p(L_1 \mid A_0)) \,. \end{aligned}$$



- Just chain rule and term rearranging.
- So does this buy us anything?

Kernel Factorizations Vs DAG Factorizations

- DAG factorization is chain rule + dropping terms due to $\perp\!\!\!\perp$.
- Impose order $\prec$ on variables in $\vec{V}$, and get this:

$$p(\vec{V}) = \prod_{V \in \vec{V}} p(V \mid \{W \mid W \text{ is earlier than V according to } \prec\})$$
$$= \prod_{V \in \vec{V}} p(V \mid \mathrm{pa}_{\mathcal{G}}(V)).$$

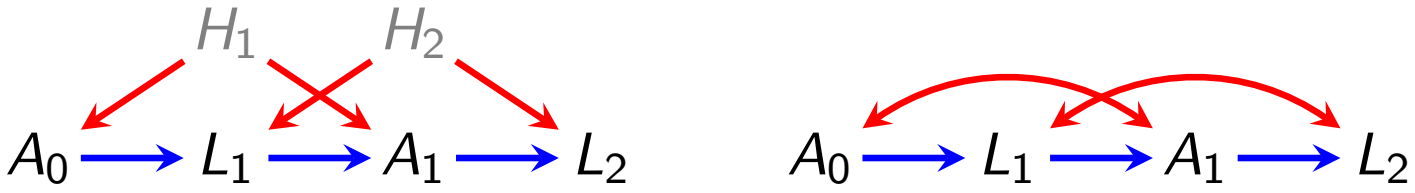
- Because V is independent of non-parental non-descendants given parents.
- Can play the same game with kernel factorization.

Kernel Factorizations Vs DAG Factorizations

- If we drop $A_0 \rightarrow L_2$ from above, get:

$$p(A_0, L_1, A_1, L_2) = q(A_0, A_1 \mid L_1) q(L_2, L_1 \mid A_0, A_1),$$

- where L_2 is independent of A_0 in $q(L_2, L_1 \mid A_0, A_1)$.

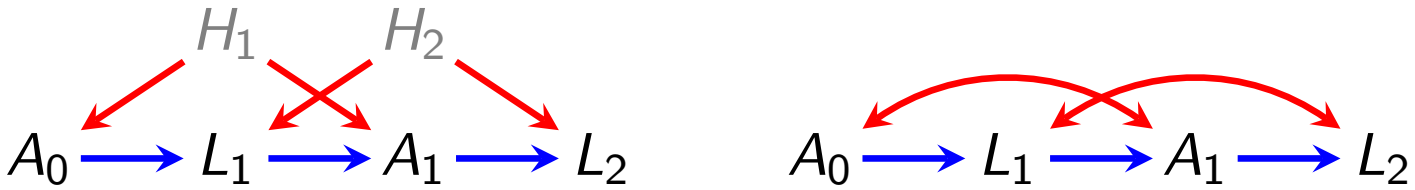


Kernel Factorizations Vs DAG Factorizations

- If we drop $A_0 \rightarrow L_2$ from above, get:

$$p(A_0, L_1, A_1, L_2) = q(A_0, A_1 \mid L_1) q(L_2, L_1 \mid A_0, A_1),$$

- where L_2 is independent of A_0 in $q(L_2, L_1 \mid A_0, A_1)$.



- Equivalent to:

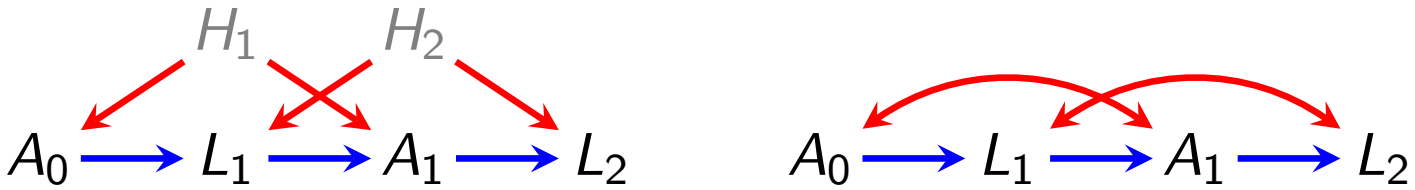
$$\sum_{L_1} p(L_2 \mid L_1, A_1, A_0) p(L_1 \mid A_0) \text{ is not a function of } A_0.$$

Kernel Factorizations Vs DAG Factorizations

- If we drop $A_0 \rightarrow L_2$ from above, get:

$$p(A_0, L_1, A_1, L_2) = q(A_0, A_1 \mid L_1) q(L_2, L_1 \mid A_0, A_1),$$

- where L_2 is independent of A_0 in $q(L_2, L_1 \mid A_0, A_1)$.



- Equivalent to:

$$\sum_{L_1} p(L_2 \mid L_1, A_1, A_0) p(L_1 \mid A_0) \text{ is not a function of } A_0.$$

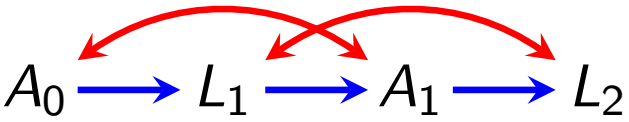
- This is a generalized independence constraint or Verma constraint.
- Thomas Verma was Pearl's student, appears in a lot of early d-separation, etc. papers.

Kernel Factorizations In General

- DAG factorization has terms for single variables given parents.
- We saw an example with terms for sets of variables.
- What do sets correspond to in general?

Kernel Factorizations In General

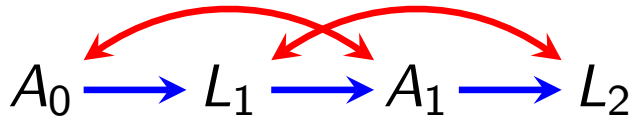
- DAG factorization has terms for single variables given parents.
- We saw an example with terms for sets of variables.
- What do sets correspond to in general?
- Districts. A district is a spanning tree of $\leftrightarrow$ edges. Example:



- Two districts here: $\{A_0, A_1\}$, $\{L_1, L_2\}$.

Kernel Factorizations In General

- DAG factorization has terms for single variables given parents.
- We saw an example with terms for sets of variables.
- What do sets correspond to in general?
- Districts. A district is a spanning tree of $\leftrightarrow$ edges. Example:



- Two districts here: $\{A_0, A_1\}$, $\{L_1, L_2\}$.
- Can view factorization as a product over kernels, one kernel per district:

$$\begin{aligned} p(A_0, L_1, A_1, L_2) &= q(A_0, A_1 \mid L_1) q(L_2, L_1 \mid A_0, A_1), \\ &= \prod_{\vec{D} \in \mathcal{D}(\mathcal{G})} q(\vec{D} \mid \mathrm{pa}_{\mathcal{G}}(\vec{D}) \setminus \vec{D}), \end{aligned}$$

- $\mathcal{D}(\mathcal{G})$ means "set of districts in $\mathcal{G}$." In a DAG, district = single vertex.

Generalizing The G-formula

- In a DAG, g-formula was dropping terms $p(A \mid \mathrm{pa}_{\mathcal{G}}(A))$.
- In a hidden variable DAG, cannot observe all $\mathrm{pa}_{\mathcal{G}}(A))$.
- Have to generalize "dropping terms."
- Will define fixing operation on graphs and kernels.
- Start with $p(\vec{V})$, this is (vacuously) a kernel, too.

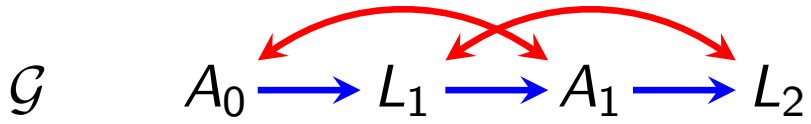
Fixing Operation On Graphs

- Fixing a vertex A in a graph: make A a square, remove all incoming arrows to A (both $\rightarrow$ and $\leftrightarrow$).
- Like doing a SWIG, but also remove A. Example, fixing A_1:

- Will denote operation of fixing A by $\phi_A(\mathcal{G})$.

Fixing Operation On Kernels

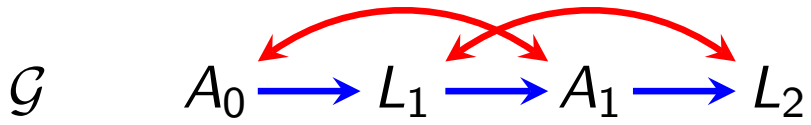
- Fixing a variable A in a kernel $q(\vec{V} \mid \vec{W})$: divide by $q(A \mid \mathrm{nd}_{\mathcal{G}}(A), \vec{W})$ ($\mathrm{nd}_{\mathcal{G}}(A)$ is "non-descendants of A.")
- Example:



$$\begin{aligned}\phi_{A_1}(p(A_0, L_1, A_1, L_2); \mathcal{G}) &\equiv \frac{p(A_0, L_1, A_1, L_2)}{p(A_1 \mid A_0, L_1)} \\ &= p(L_2 \mid A_1, L_1, A_0)p(L_1 \mid A_0)p(A_0) \\ &\equiv q(L_2, L_1, A_0 \mid A_1).\end{aligned}$$

Fixing Operation On Kernels

- Fixing a variable A in a kernel $q(\vec{V} \mid \vec{W})$: divide by $q(A \mid \text{nd}_{\mathcal{G}}(A), \vec{W})$ ($\text{nd}_{\mathcal{G}}(A)$ is "non-descendants of A.")
- Example:



$$\begin{aligned}\phi_{A_1}(p(A_0, L_1, A_1, L_2); \mathcal{G}) &\equiv \frac{p(A_0, L_1, A_1, L_2)}{p(A_1 \mid A_0, L_1)} \\ &= p(L_2 \mid A_1, L_1, A_0)p(L_1 \mid A_0)p(A_0) \\ &\equiv q(L_2, L_1, A_0 \mid A_1).\end{aligned}$$

- Use same notation $\phi_A(.)$, but now mention graph $\mathcal{G}$ also, not just starting kernel.
- Without graph don't know what non-descendants are!
- Important note: unlike $p(.|.)$, $q(.|.)$ notation is ambiguous, have to tell you how I got the $q(.|.)$, usually via a sequence of $\phi(.)$.

Fixing As G-formula

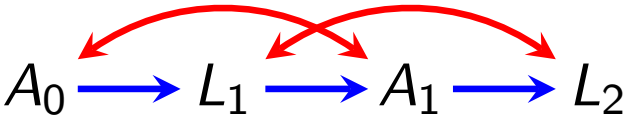
- Fixing in a kernel is sort of like g-formula when there are hidden variables.
- In a DAG can always use g-formula.
- In an ADMG can not always fix – depends on the graph.
- Will try to apply fixing on graph/kernel together multiply times.
- Use graph to check if we can fix.

Fixable Vertices

- A is fixable if there does not exist a single B with these paths:

$$A \rightarrow V_1 \rightarrow V_2 \rightarrow \ldots \rightarrow B$$
$$A \leftrightarrow Z_1 \leftrightarrow Z_2 \leftrightarrow \ldots \leftrightarrow B$$

- Example, can fix A_1, cannot fix A_0:

$\mathcal{G}$ 

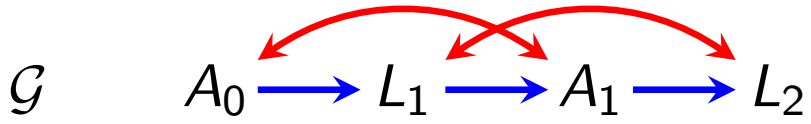
Fixable Vertices

- A is fixable if there does not exist a single B with these paths:

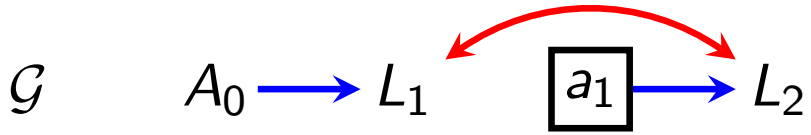
$$A \rightarrow V_1 \rightarrow V_2 \rightarrow \ldots \rightarrow B$$

$$A \leftrightarrow Z_1 \leftrightarrow Z_2 \leftrightarrow \ldots \leftrightarrow B$$

- Example, can fix A_1, cannot fix A_0:



- Once we fix A_1, can now fix A_0 also:

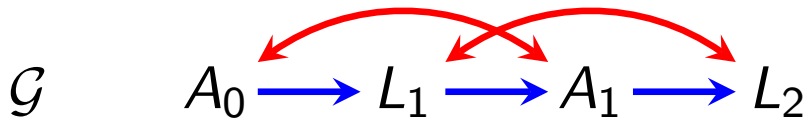


Fixable Vertices

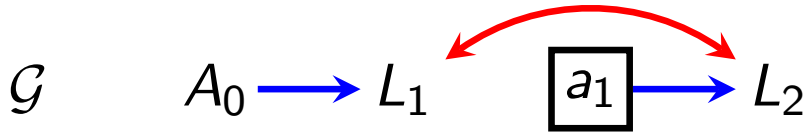
- A is fixable if there does not exist a single B with these paths:

$$A \rightarrow V_1 \rightarrow V_2 \rightarrow \ldots \rightarrow B$$
$$A \leftrightarrow Z_1 \leftrightarrow Z_2 \leftrightarrow \ldots \leftrightarrow B$$

- Example, can fix A_1, cannot fix A_0:



- Once we fix A_1, can now fix A_0 also:



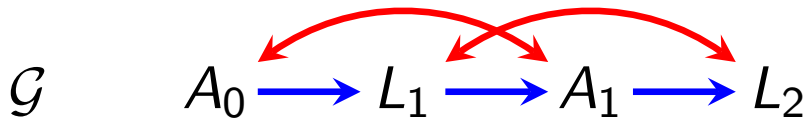
- Fixing a set in a graph: $\phi_{\vec{A}}(\mathcal{G})$, find an order $\{A_1, \ldots A_k\}$ in $\vec{A}$ such that can fix in that order in $\mathcal{G}$.
- If possible, can also fix $\vec{A}$ in a kernel: $\phi_{\vec{A}}(q; \mathcal{G})$ in the same order.

Fixable Vertices

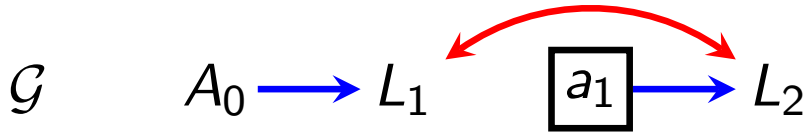
- A is fixable if there does not exist a single B with these paths:

$$A \rightarrow V_1 \rightarrow V_2 \rightarrow \ldots \rightarrow B$$
$$A \leftrightarrow Z_1 \leftrightarrow Z_2 \leftrightarrow \ldots \leftrightarrow B$$

- Example, can fix A_1, cannot fix A_0:



- Once we fix A_1, can now fix A_0 also:



- Fixing a set in a graph: $\phi_{\vec{A}}(\mathcal{G})$, find an order $\{A_1, \ldots A_k\}$ in $\vec{A}$ such that can fix in that order in $\mathcal{G}$.
- If possible, can also fix $\vec{A}$ in a kernel: $\phi_{\vec{A}}(q; \mathcal{G})$ in the same order.
- If not possible to fix in any order, operation is undefined.
- If possible in some order, then order does not matter (not obvious!)

One Line **ID** Algorithm

- Input: ADMG $\mathcal{G}$ with vertex set $\vec{V}$, any outcome set $\vec{Y}$, any treatment set $\vec{A}$.
- Question: is $p(\vec{Y}(\vec{a}))$ identified from $p(\vec{V})$ in any hidden variable DAG causal model represented by $\mathcal{G}$?
- Specific value $\vec{a}$ does not matter.
  1. Construct SWIG $\mathcal{G}(\vec{a})$ for fixing $\vec{a}$.
  2. Define $\vec{Y}^*$ that are ancestors of $\vec{Y}$ in $\mathcal{G}(\vec{a})$ and are not square nodes.
  3. Construct a graph $\mathcal{G}_{\vec{Y}^*}$ containing only $\vec{Y}^*$ (and edges between those nodes).
  4. If fixing all sets below is defined, return

$$\sum_{\vec{Y}^* \setminus \vec{Y}} \prod_{\vec{D} \in \mathcal{D}(\mathcal{G}_{\vec{Y}^*})} \phi_{\vec{V} \setminus \vec{D}}(p(\vec{V}); \mathcal{G}).$$

  4. Otherwise return **not identified**.

One Line **ID** Algorithm

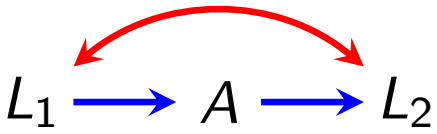
- Input: ADMG $\mathcal{G}$ with vertex set $\vec{V}$, any outcome set $\vec{Y}$, any treatment set $\vec{A}$.
- Question: is $p(\vec{Y}(\vec{a}))$ identified from $p(\vec{V})$ in any hidden variable DAG causal model represented by $\mathcal{G}$?
- Specific value $\vec{a}$ does not matter.
  1. Construct SWIG $\mathcal{G}(\vec{a})$ for fixing $\vec{a}$.
  2. Define $\vec{Y}^*$ that are ancestors of $\vec{Y}$ in $\mathcal{G}(\vec{a})$ and are not square nodes.
  3. Construct a graph $\mathcal{G}_{\vec{Y}^*}$ containing only $\vec{Y}^*$ (and edges between those nodes).
  4. If fixing all sets below is defined, return

$$\sum_{\vec{Y}^* \setminus \vec{Y}} \prod_{\vec{D} \in \mathcal{D}(\mathcal{G}_{\vec{Y}^*})} \phi_{\vec{V} \setminus \vec{D}}(p(\vec{V}); \mathcal{G}).$$

  4. Otherwise return **not identified**.
- Very abstract, let's do some examples!

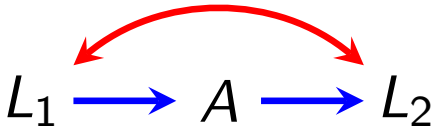
Example 1

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.

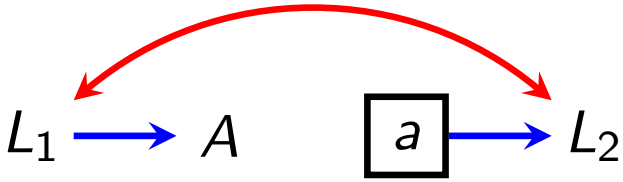


Example 1

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.

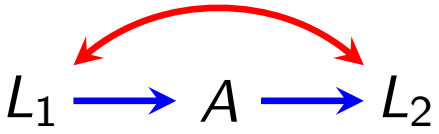


1 Construct SWIG $\mathcal{G}(a)$:

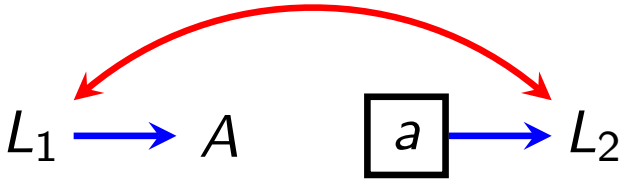


Example 1

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.



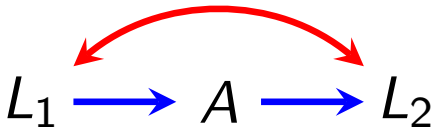
1 Construct SWIG $\mathcal{G}(a)$:



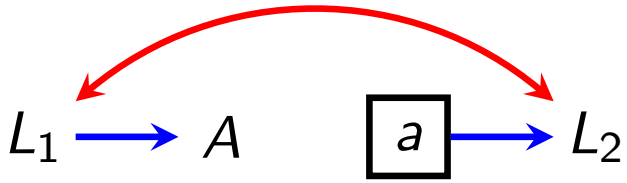
2 $\vec{Y}^* = \{L_2\}$.

Example 1

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.



1 Construct SWIG $\mathcal{G}(a)$:

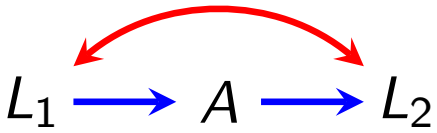


2 $\vec{Y}^* = \{L_2\}$.

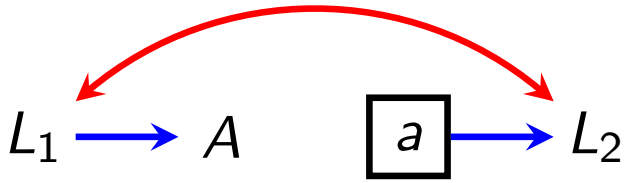
3 Construct $\mathcal{G}_{\vec{Y}^*}$:

Example 1

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.



1 Construct SWIG $\mathcal{G}(a)$:



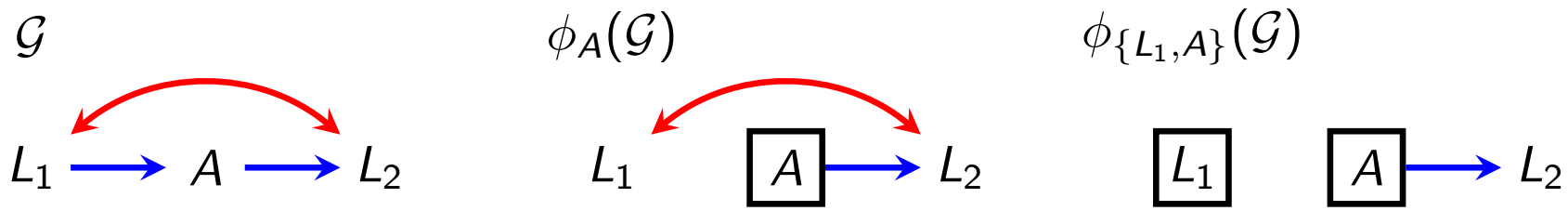
2 $\vec{Y}^* = \{L_2\}$.

3 Construct $\mathcal{G}_{\vec{Y}^*}$:

4 Only one district $\{L_2\}$. Is fixing $\phi_{\{L_1,A\}}(\mathcal{G})$ defined?

Example 1 (Continued)

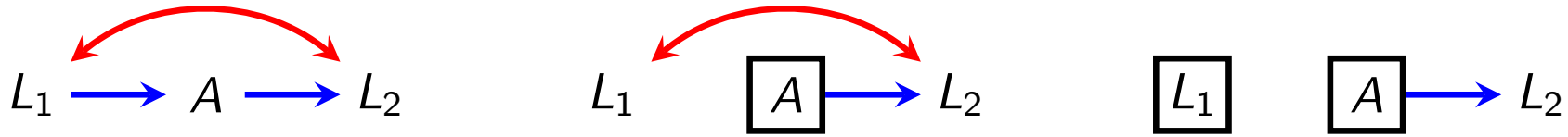
4 Yes, can fix A, then L_1:



Example 1 (Continued)

- Yes, can fix A, then L_1:

$\mathcal{G}$ $\qquad\qquad$ $\phi_A(\mathcal{G})$ $\qquad\qquad$ $\phi_{\{L_1,A\}}(\mathcal{G})$



- Thus, return

$$p(L_2(a)) = \sum_{\vec{Y}^* \setminus \vec{Y}} \prod_{\vec{D} \in \mathcal{D}(\mathcal{G}_{\vec{Y}^*})} \phi_{\vec{V} \setminus \vec{D}}(p(\vec{V}); \mathcal{G})$$

$$= \phi_{\{L_1,A\}}(p(L_1, A, L_2); \mathcal{G})$$

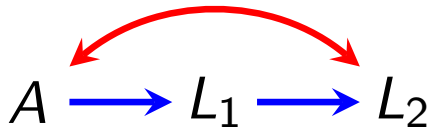
$$= \phi_{\{L_1\}}\left(\frac{p(L_1, A, L_2)}{p(A \mid L_1)}; \phi_A(\mathcal{G})\right) \equiv \phi_{\{L_1\}}(q(L_1, L_2 \mid A); \phi_A(\mathcal{G}))$$

$$= \frac{q(L_1, L_2 \mid A)}{q(L_1 \mid A, L_2)}$$

$$= \frac{p(L_2 \mid A, L_1)p(L_1)}{\frac{p(L_2|A,L_1)p(L_1)}{\sum_{L_1} p(L_2|A,L_1)p(L_1)}} = \sum_{L_1} p(L_2 \mid A, L_1)p(L_1)$$

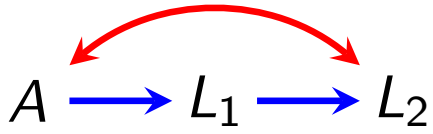
Example 2

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.

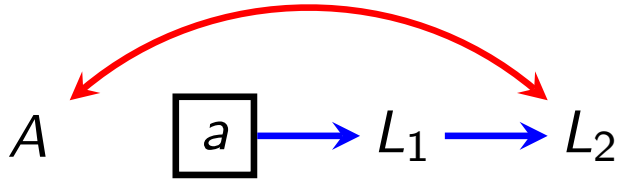


Example 2

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.

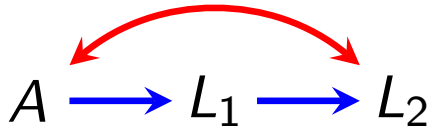


1 Construct SWIG $\mathcal{G}(a)$:

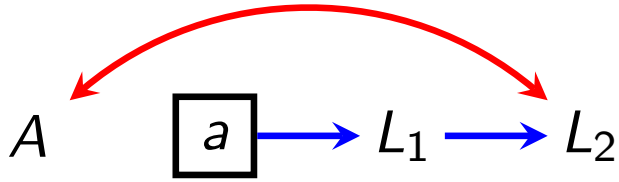


Example 2

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.



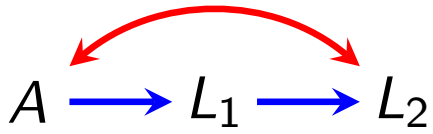
1 Construct SWIG $\mathcal{G}(a)$:



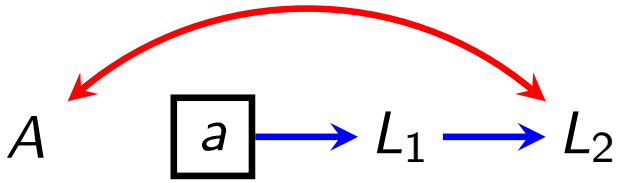
2 $\vec{Y}^* = \{L_2, L_1\}$.

Example 2

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.



1 Construct SWIG $\mathcal{G}(a)$:

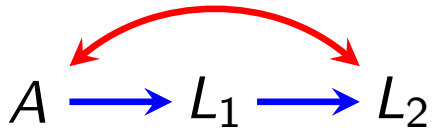


2 $\vec{Y}^* = \{L_2, L_1\}$.

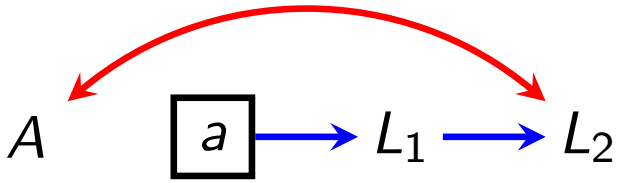
3 Construct $\mathcal{G}_{\vec{Y}^*}$:

Example 2

- Target: $p(\vec{Y}(\vec{a})) = p(L_2(a))$.



1 Construct SWIG $\mathcal{G}(a)$:



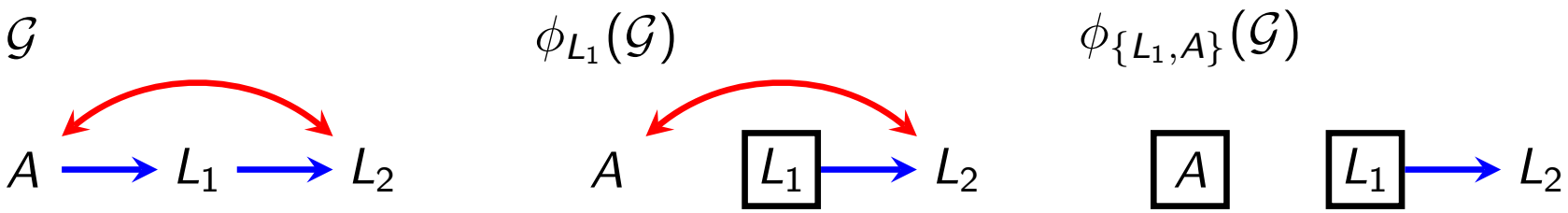
2 $\vec{Y}^* = \{L_2, L_1\}$.

3 Construct $\mathcal{G}_{\vec{Y}^*}$:

4 Two districts: $\{L_1\}$, $\{L_2\}$. Is fixing $\phi_{\{L_1,A\}}(\mathcal{G})$ and $\phi_{\{L_2,A\}}$ defined?

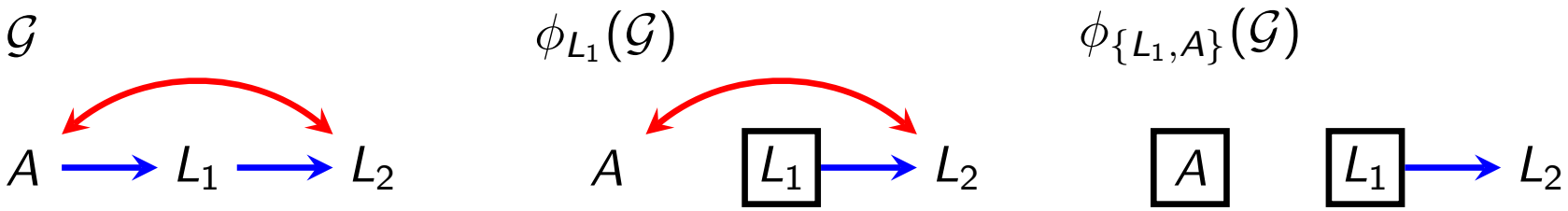
Example 2 (Continued)

4 $\phi_{\{L_1,A\}}(\mathcal{G})$ defined, can fix L_1, then A:

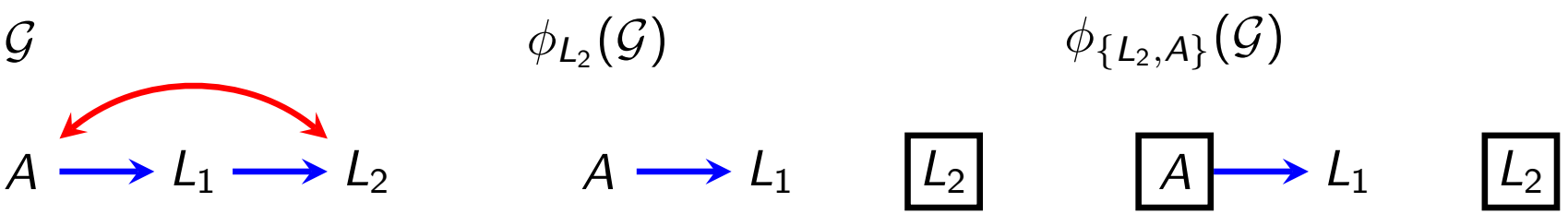


Example 2 (Continued)

4 $\phi_{\{L_1,A\}}(\mathcal{G})$ defined, can fix L_1, then A:



4 $\phi_{\{L_2,A\}}(\mathcal{G})$ defined, can fix L_2, then A:



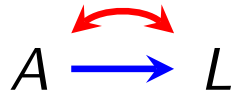
Example 2 (Continued)

4 Thus, return

$$
\begin{aligned}
p(L_2(a)) &= \sum_{\vec{Y}^* \setminus \vec{Y}} \prod_{\vec{D} \in \mathcal{D}(\mathcal{G}_{\vec{Y}^*})} \phi_{\vec{V} \setminus \vec{D}}(p(\vec{V}); \mathcal{G}) \\
&= \sum_{L_1} \phi_{\{L_1, A\}}(p(L_1, A, L_2); \mathcal{G}) \cdot \phi_{\{L_2, A\}}(p(L_1, A, L_2); \mathcal{G}) \\
&= \sum_{L_1} \phi_{\{A\}}\left(\frac{p(L_1, A, L_2)}{p(L_1 \mid A)}; \phi_{L_1}(\mathcal{G})\right) \cdot \phi_{\{A\}}\left(\frac{p(L_1, A, L_2)}{p(L_2 \mid A, L_1)}; \phi_{L_2}(\mathcal{G})\right) \\
&\equiv \sum_{L_1} \phi_{\{A\}}(q(A, L_2 \mid L_1); \phi_{L_1}(\mathcal{G})) \cdot \phi_{\{A\}}(p(L_1, A); \phi_{L_2}(\mathcal{G})) \\
&= \sum_{L_1} \phi_{\{A\}}(q(A, L_2 \mid L_1); \phi_{L_1}(\mathcal{G})) \cdot p(L_1 \mid A) \\
&= \sum_{L_1} \left(\sum_{A'} p(L_2 \mid L_1, A') p(A')\right) \cdot p(L_1 \mid A)
\end{aligned}
$$

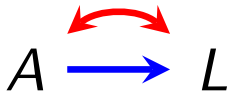
Example 3

- Target: $p(\vec{Y}(\vec{a})) = p(L(a))$.

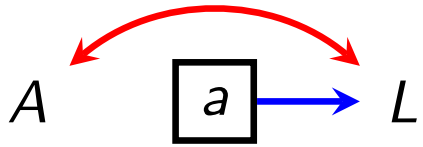


Example 3

- Target: $p(\vec{Y}(\vec{a})) = p(L(a))$.

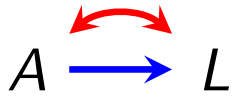


1 Construct SWIG $\mathcal{G}(a)$:

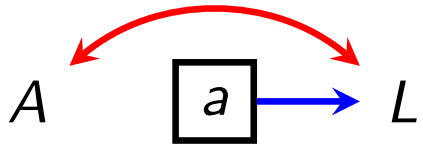


Example 3

- Target: $p(\vec{Y}(\vec{a})) = p(L(a))$.



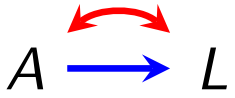
1 Construct SWIG $\mathcal{G}(a)$:



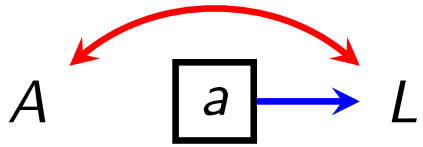
2 $\vec{Y}^* = \{L\}$.

Example 3

- Target: $p(\vec{Y}(\vec{a})) = p(L(a))$.



1 Construct SWIG $\mathcal{G}(a)$:

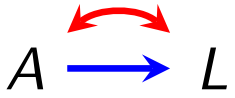


2 $\vec{Y}^* = \{L\}$.

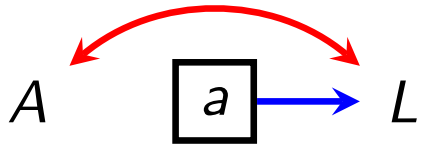
3 Construct $\mathcal{G}_{\vec{Y}^*}$:

Example 3

- Target: $p(\vec{Y}(\vec{a})) = p(L(a))$.



1. Construct SWIG $\mathcal{G}(a)$:



2. $\vec{Y}^* = \{L\}$.
3. Construct $\mathcal{G}_{\vec{Y}^*}$:

4. One district: $\{L\}$. Is fixing $\phi_{\{A\}}(\mathcal{G})$ defined? No. So we fail.

Conditional Causal Effects (Effect Modification)

- May be interested instead in $p(\vec{Y}(\vec{a}) \mid \vec{Z}(\vec{a}))$ instead of $p(\vec{Y}(\vec{a}))$.
- Example "average treatment effect on African-American men over the age of 55:

$$E[Y(\vec{a}) \mid \vec{z}] - E[Y(\vec{a}') \mid \vec{z}].$$

- Called effect modification in Epidemiology.
- Is this harder or easier than identifying $p(\vec{Y}(\vec{a}))$?
- Turns out this is pretty easy if we know how to identify $p(\vec{Y}(\vec{a}))$.

Identifying Conditional Causal Effects

1. Construct a SWIG $\mathcal{G}(\vec{a})$.
2. Let $\vec{W} \subseteq \vec{Z}$ be the set of all W such that all paths from W to $\vec{Y}$ in $\mathcal{G}(\vec{a})$ that start with $\rightarrow W$ or $\leftrightarrow W$ are m-separated by $\vec{Z} \setminus \{W\}$.
3. Try to identify $p(\vec{Y} \cup (\vec{Z} \setminus \vec{W}) \mid \mathrm{do}(\vec{a} \cup \vec{w}))$.
4. If this works and result is $q(\vec{Y}, \vec{Z} \setminus \vec{W} \mid \vec{a}, \vec{w})$, for any value subset $\vec{w}$ of $\vec{z}$, return

$$q(\vec{Y} \mid \vec{Z} \setminus \vec{W}, \vec{a}, \vec{w})|_{\vec{Z} \setminus \vec{W} = \vec{z} \setminus \vec{w}} = \frac{q(\vec{Y}, \vec{Z} \setminus \vec{W} \mid \vec{a}, \vec{w})}{\sum_{\vec{Y}} q(\vec{Y}, \vec{Z} \setminus \vec{W} \mid \vec{a}, \vec{w})}\Big|_{\vec{Z} \setminus \vec{W} = \vec{z} \setminus \vec{w}}.$$

5. Otherwise, return **not identified**.

Identifying Conditional Causal Effects

1. Construct a SWIG $\mathcal{G}(\vec{a})$.
2. Let $\vec{W} \subseteq \vec{Z}$ be the set of all W such that all paths from W to $\vec{Y}$ in $\mathcal{G}(\vec{a})$ that start with $\rightarrow W$ or $\leftrightarrow W$ are m-separated by $\vec{Z} \setminus \{W\}$.
3. Try to identify $p(\vec{Y} \cup (\vec{Z} \setminus \vec{W}) \mid \text{do}(\vec{a} \cup \vec{w}))$.
4. If this works and result is $q(\vec{Y}, \vec{Z} \setminus \vec{W} \mid \vec{a}, \vec{w})$, for any value subset $\vec{w}$ of $\vec{z}$, return

$$q(\vec{Y} \mid \vec{Z} \setminus \vec{W}, \vec{a}, \vec{w})|_{\vec{Z} \setminus \vec{W} = \vec{z} \setminus \vec{w}} = \frac{q(\vec{Y}, \vec{Z} \setminus \vec{W} \mid \vec{a}, \vec{w})}{\sum_{\vec{Y}} q(\vec{Y}, \vec{Z} \setminus \vec{W} \mid \vec{a}, \vec{w})}\Big|_{\vec{Z} \setminus \vec{W} = \vec{z} \setminus \vec{w}}.$$

5. Otherwise, return **not identified**.

- This is also complete (if it fails, no other method can identify without more assumptions).

A Note On Estimation

- For g-computation with hidden variables can use parametric g-formula or marginal structural models (IPW), as before.
- In more complex cases can sometimes use MLE plug-in estimation, or weighting methods.
- Getting "good" estimators is an open problem in general, because you have to estimate weird, variation dependent functions.
- In the special case of discrete statespaces, there is a very clean way out – some out of class reading on this: (Richardson et al, 2017).

Example: Front-Door Formula Estimation

- Recall:

$$p(L_2(a)) = \sum_{l_1} \left(\sum_{a'} p(L_2 \mid l_1, a') p(a') \right) p(L_1 = l_1 \mid a)$$

- Plug-in MLE:
  1. Fit $E[L_2 \mid L_1, A; \alpha]$ and $p(L_1 \mid A; \beta)$ to yield $\hat{\alpha}, \hat{\beta}$.
  2. Estimate $E[L_2(a)]$ as

$$\frac{1}{n} \sum_{i=1}^{n} \left(\frac{1}{n} \sum_{j=1}^{n} E[L_2 \mid L_1^i, A^j; \hat{\alpha}] \right) \cdot p(L_1^i \mid A = 1; \hat{\beta}).$$

- Weighting:
  1. Fit $p(L_1 \mid A; \beta)$ to yield $\hat{\beta}$.
  2. Estimate $E[L_2(a)]$ as

$$\frac{1}{n} \sum_{i=1}^{n} L_2^i \frac{p(L_1^i \mid A = 1; \hat{\beta})}{p(L_1^i \mid A^i; \hat{\beta})}$$

Next time: Causal Decision Theory.