Causal Inference
CS 477-677

Causal Models Of A Directed Acyclic Graph

Ilya Shpitser

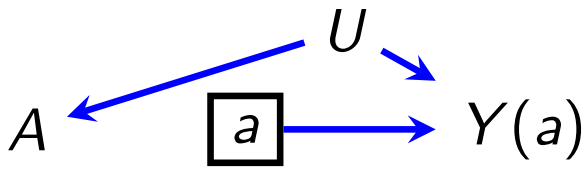
Outline

Review

- Discussed Hume's definition and Lind's experiment.
- Discussed Neyman's potential outcomes for inference with randomized treatments.
- Two schools of thought on causality since then
  - Graph based (Wright's pedigree analysis to structural equation models).
  - Potential outcome based (Neyman to Rubin: conditionally ignorable models, etc.)
- Dichotomy exists in the field to this day.
- Today we see why the dichotomy is false!
- Today we also see how statistical and causal DAGs are different.

Review: Conditional Ignorability

- Assume an observed vector of baseline factors/confounders U, binary treatment A, outcome Y.
- Assume both potential outcomes on Y: $Y(1), Y(0)$.
- Consistency $Y = Y(A)$ and conditional ignorability $\{Y(1), Y(0)\} \perp\!\!\!\perp A \mid U$.
- Had an informal picture:

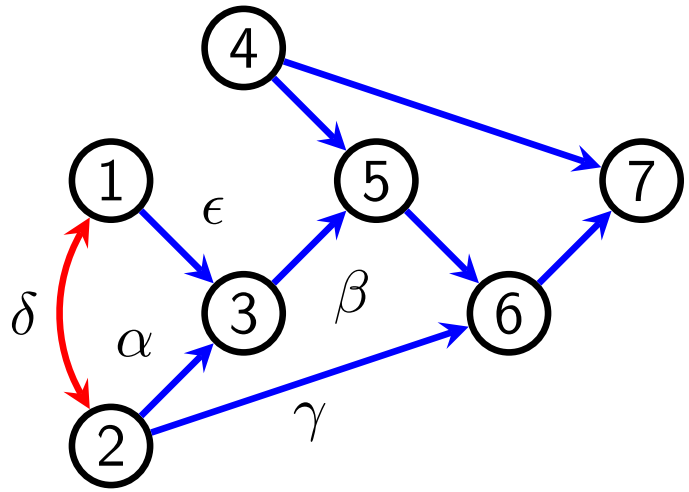


- Have two versions of A, one represents assignment probability, and one hypothetical assignment.
- Conditional ignorability follows by d-separation.

Review: Structural Equation Models

- Assume linear regression for every variable given parents:

$$Y = w_0 + \sum_{X_i} w_i \cdot X_i + \epsilon_Y$$



- Functions of coefficients as degrees of inbreeding, total, and mediated effects.

Generalizing Structural Equation Models

- Obvious point: no particular reason to do linear models.
- Can use any model at all to relate Y and $\text{pa}_{\mathcal{G}}(Y)$.
- The trick is not being confused about what hypothetical experiments mean.
- The result is Pearl's functional model or non-parametric structural equation model (with independent errors) (NPSEM-IE).

Non-Parametric Structural Equation Model

- Given a DAG $\mathcal{G}$, for every variable X, posit a mechanism f_X and noise term ϵ_X.
- These determine value of X in terms of values of parents of X:

$$X \leftarrow f_X(\text{pa}_{\mathcal{G}}(X), \epsilon_X).$$

- Functions are unrestricted.

Non-Parametric Structural Equation Model

- Given a DAG $\mathcal{G}$, for every variable X, posit a mechanism f_X and noise term ϵ_X.
- These determine value of X in terms of values of parents of X:

$$X \leftarrow f_X(\text{pa}_{\mathcal{G}}(X), \epsilon_X).$$

- Functions are unrestricted.
- Note: imperative assignment, not equality!

Non-Parametric Structural Equation Model

- Given a DAG $\mathcal{G}$, for every variable X, posit a mechanism f_X and noise term ϵ_X.
- These determine value of X in terms of values of parents of X:

$$X \leftarrow f_X(\text{pa}_{\mathcal{G}}(X), \epsilon_X).$$

- Functions are unrestricted.
- Note: imperative assignment, not equality!
- Assume independent (not necessarily Gaussian) errors:

$$p(\epsilon_{X_1}, \ldots \epsilon_{X_k}) = \prod_{i=1}^{k} p(\epsilon_{X_i}).$$

- Note: if ϵ are fixed, the model is entirely deterministic.

Observed Distributions in the NPSEM-IE

- What is $p(\vec{X} = \vec{x})$ for an NPSEM-IE of a DAG $\mathcal{G}$?

$$p(\vec{X} = \vec{x}) = \prod_{X \in \vec{X}} \sum_{\{\epsilon_X : f_X(\vec{x}_{\text{pa}_\mathcal{G}(X)}, \epsilon_X) = \vec{x}_X\}} p(\epsilon_X)$$

- In words: add up all probability mass of ϵ variables that make f create observed $\vec{x}$.
- Basically equivalent to computing

$$\prod_{X \in \vec{X}} p(X \mid \text{pa}_\mathcal{G}(X)).$$

- Different from Bayesian networks since we can deal with interventions.

Interventions in the NPSEM-IE

- Recall, imperative assignment:

$$X \leftarrow f_X(\text{pa}_{\mathcal{G}}(X), \epsilon_X).$$

- We allow external changes to model, where any X is assigned to a constant instead:

$$X \leftarrow x.$$

- Note: everything else stays the same!
- Have a new model where some functions are now constants.
- How does this model behave?

Post-Intervention Distributions in the NPSEM-IE

- Say $\vec{x}$ is an assignment to all variables $\vec{X} \setminus \{A\}$.
- What is $p(\vec{X} \setminus \{A\} = \vec{x} \mid \text{do}(A = a))$ (a distribution over $X(a)$ for all $X \in \vec{X} \setminus \{A\}$) for an NPSEM-IE of a DAG $\mathcal{G}$?
- Replace f_A by "$A \leftarrow a$." Recompute as usual.
- Define $\vec{x}^*$ as $\vec{x}$ for all variables that are not A and a for A.

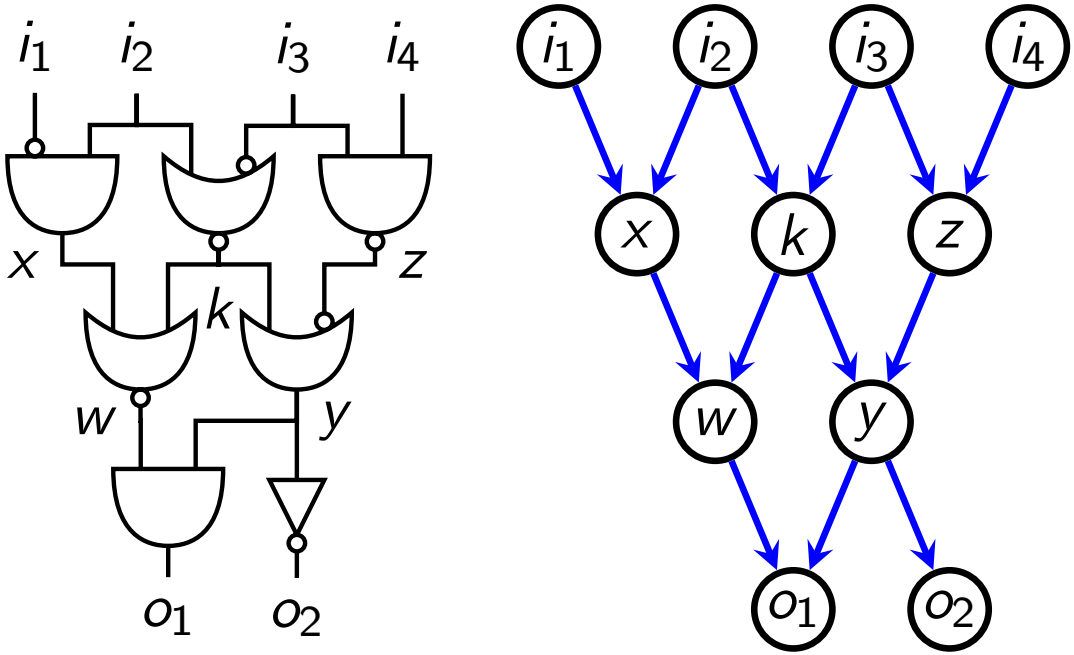
$$p(\vec{X} \setminus \{A\} = \vec{x} \mid \text{do}(X = a)) = \prod_{\tilde{X} \in \vec{X} \setminus \{A\}} \sum_{\{\epsilon_{\tilde{X}} : f_{\tilde{X}}(\vec{x}^*_{\text{pa}_{\mathcal{G}}(\tilde{X})}, \epsilon_{\tilde{X}}) = \vec{x}_{\tilde{X}}\}} p(\epsilon_{\tilde{X}})$$

- Basically equivalent to computing

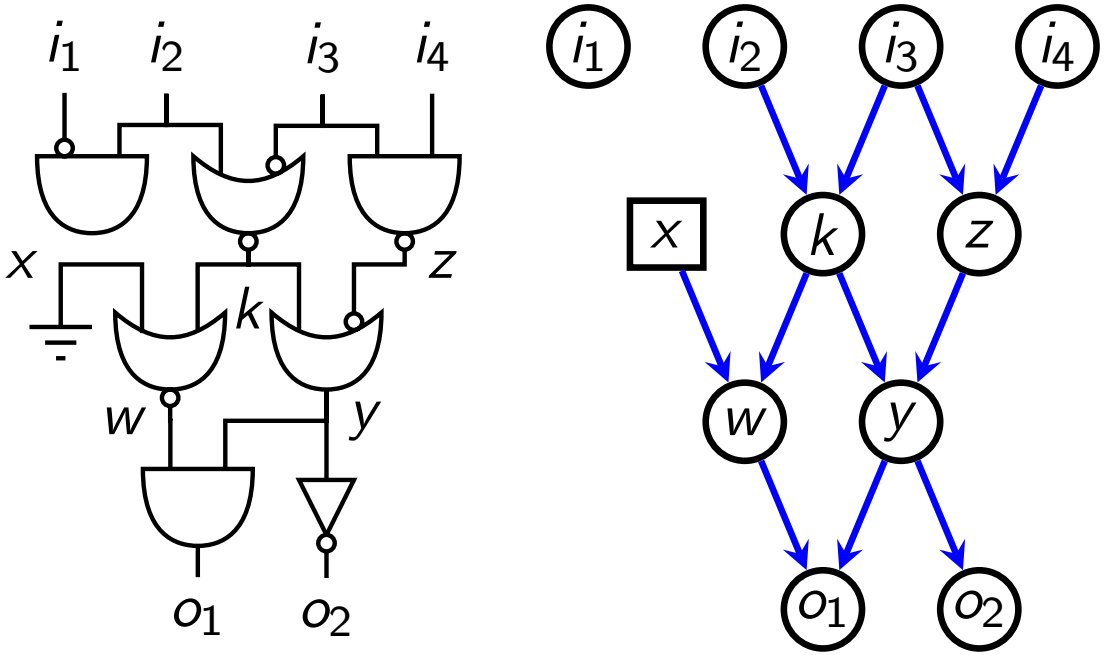
$$\prod_{\tilde{X} \in \vec{X} \setminus \{A\}} p(\tilde{X} \mid \text{pa}_{\mathcal{G}}(\tilde{X}))|_{A=a}$$

- This formula is very important, we will come back to it later.

NPSEM-IE: A Familiar Example



NPSEM-IE: A Familiar Example



Generalizing Ignorable Models

- Obvious point: no particular reason ignorability or conditional ignorability should hold.
- How do we impose arbitrary restrictions on counterfactuals?
- Lots of approaches, but we will use DAGs to avoid being confused.
- Humans tend to have a strong visual system, weak algebraic system.
- Hard to think about a big set of algebraic constraints.
- Easy to think about a graph!

Counterfactual Model On A DAG

- Given a DAG $\mathcal{G}$, for every variable X, posit existence of a set of potential outcome random variables: $X(\text{pa}_{\mathcal{G}}(X))$.
- Assume consistency as usual: $X(A) = X$.
- Derive other potential outcomes using recursive substitution:

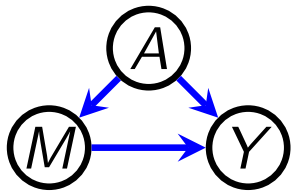
$$X(\vec{a}) = X(\text{pa}_{\mathcal{G}}(X) \cap \vec{A} = \vec{a}, \{\text{pa}_{\mathcal{G}}(X) \setminus \vec{A}\}(\vec{a})).$$

Counterfactual Model On A DAG

- Given a DAG $\mathcal{G}$, for every variable X, posit existence of a set of potential outcome random variables: $X(\text{pa}_{\mathcal{G}}(X))$.
- Assume consistency as usual: $X(A) = X$.
- Derive other potential outcomes using recursive substitution:

$$X(\vec{a}) = X(\text{pa}_{\mathcal{G}}(X) \cap \vec{A} = \vec{a}, \{\text{pa}_{\mathcal{G}}(X) \setminus \vec{A}\}(\vec{a})).$$

- Example: $Y(a) = Y(M(a), a)$.



Assumpions on Counterfactuals

- Only assumed consistency so far.
- Will also assume

$$\left\{ \{ V(\vec{a}_V) \mid \vec{a}_V \in \mathfrak{X}_{\mathbf{pa}_{\mathcal{G}}(V)} \} \middle| V \in \vec{V} \right\}$$

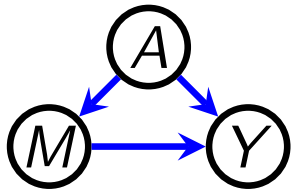
are mutually independent. ($\mathfrak{X}_{\vec{S}}$ is state space of variables in $\vec{S}$).

Assumpions on Counterfactuals

- Only assumed consistency so far.
- Will also assume

$$\left\{ \{ V(\vec{a}_V) \mid \vec{a}_V \in \mathfrak{X}_{\mathrm{pa}_{\mathcal{G}}(V)} \} \middle| V \in \vec{V} \right\}$$

  are mutually independent. ($\mathfrak{X}_{\vec{S}}$ is state space of variables in $\vec{S}$).
- Example: $A \perp\!\!\!\perp M(a) \perp\!\!\!\perp Y(m, a')$ for all a, a', m.

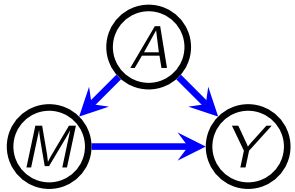


Assumpions on Counterfactuals

- Only assumed consistency so far.
- Will also assume

$$\left\{ \{ V(\vec{a}_V) \mid \vec{a}_V \in \mathfrak{X}_{\mathrm{pa}_{\mathcal{G}}(V)} \} \middle| V \in \vec{V} \right\}$$

  are mutually independent. ($\mathfrak{X}_{\vec{S}}$ is state space of variables in $\vec{S}$).
- Example: $A \perp\!\!\!\perp M(a) \perp\!\!\!\perp Y(m, a')$ for all a, a', m.



- Implies conditional ignorability: $Y(m) \perp\!\!\!\perp M(A) \mid A$.

Two Definitions Of The Same Model

- Structural equations and counterfactuals give the same model – NPSEM-IE.
- Random variable $X(\text{pa}_{\mathcal{G}}(X))$ is given by $f_X(\text{pa}_{\mathcal{G}}(X), \epsilon_X)$.
- Counterfactual Independence assumption

$$\left\{ \{V(\vec{a}_V) \mid \vec{a}_V \in \mathfrak{X}_{\text{pa}_{\mathcal{G}}(V)}\} \middle| V \in \vec{V} \right\}$$

- is equivalent to independent errors assumption

$$p(\epsilon_{X_1}, \ldots \epsilon_{X_k}) = \prod_{i=1}^{k} p(\epsilon_{X_i}).$$

Two Definitions Of The Same Model

- Structural equations and counterfactuals give the same model – NPSEM-IE.
- Random variable $X(\mathrm{pa}_{\mathcal{G}}(X))$ is given by $f_X(\mathrm{pa}_{\mathcal{G}}(X), \epsilon_X)$.
- Counterfactual Independence assumption

$$\left\{ \{V(\vec{a}_V) \mid \vec{a}_V \in \mathfrak{X}_{\mathrm{pa}_{\mathcal{G}}(V)}\} \middle| V \in \vec{V} \right\}$$

- is equivalent to independent errors assumption

$$p(\epsilon_{X_1}, \ldots \epsilon_{X_k}) = \prod_{i=1}^{k} p(\epsilon_{X_i}).$$

- If you like random variables, use counterfactual view, if you like mechanisms, use structural equations.
- Both are useful.

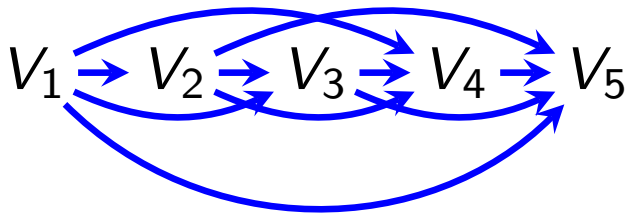
The G-Formula

- Causal models of DAGs are very nice.
- For any $\vec{Y}, \vec{A} \subseteq \vec{V}$, can express $p(\vec{Y}(\vec{a}))$ as a function of observed data.

$$p(\vec{Y}(\vec{a})) = \sum_{\vec{V} \setminus (\vec{Y} \cup \vec{A})} \prod_{V \in \vec{V} \setminus \vec{A}} p(V \mid \mathrm{pa}_{\mathcal{G}}(V)) \Big|_{\vec{A}=\vec{a}}.$$

- Intuition: dropping terms that are not causally relevant post-intervention.
- Example: $p(V_5(v_2 = 1, v_4 = 0))$ is equal to

$$\sum_{V_1, V_3} p(V_5 \mid v_4 = 0, V_3, v_2 = 1, V_1) p(V_3 \mid v_2 = 1, V_1) p(V_1) \text{ in:}$$



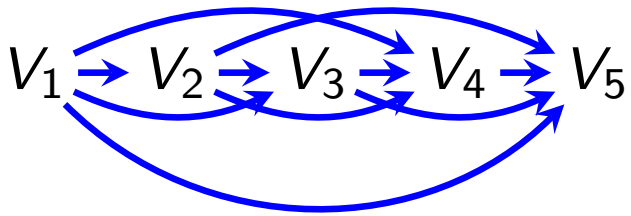
The G-Formula

- Causal models of DAGs are very nice.
- For any $\vec{Y}, \vec{A} \subseteq \vec{V}$, can express $p(\vec{Y}(\vec{a}))$ as a function of observed data.

$$p(\vec{Y}(\vec{a})) = \sum_{\vec{V} \setminus (\vec{Y} \cup \vec{A})} \prod_{V \in \vec{V} \setminus \vec{A}} p(V \mid \mathrm{pa}_{\mathcal{G}}(V)) \Big|_{\vec{A}=\vec{a}}.$$

- Intuition: dropping terms that are not causally relevant post-intervention.
- Example: $p(V_5(v_2 = 1, v_4 = 0))$ is equal to

$$\sum_{V_1, V_3} p(V_5 \mid v_4 = 0, V_3, v_2 = 1, V_1) p(V_3 \mid v_2 = 1, V_1) p(V_1) \text{ in:}$$



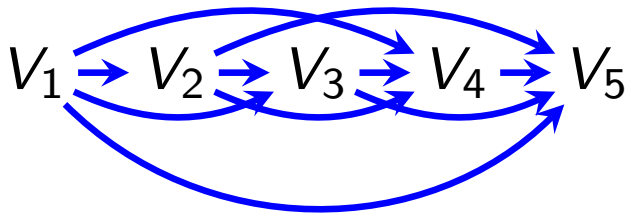
The G-Formula

- Causal models of DAGs are very nice.
- For any $\vec{Y}, \vec{A} \subseteq \vec{V}$, can express $p(\vec{Y}(\vec{a}))$ as a function of observed data.

$$p(\vec{Y}(\vec{a})) = \sum_{\vec{V} \setminus (\vec{Y} \cup \vec{A})} \prod_{V \in \vec{V} \setminus \vec{A}} p(V \mid \mathrm{pa}_{\mathcal{G}}(V)) \Big|_{\vec{A}=\vec{a}}.$$

- Intuition: dropping terms that are not causally relevant post-intervention.
- Example: $p(V_5(v_2 = 1, v_4 = 0))$ is equal to

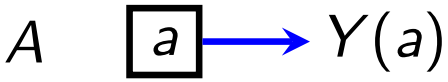
$$\sum_{V_1, V_3} p(V_5 \mid v_4 = 0, V_3, v_2 = 1, V_1) p(V_3 \mid v_2 = 1, V_1) p(V_1) \text{ in:}$$

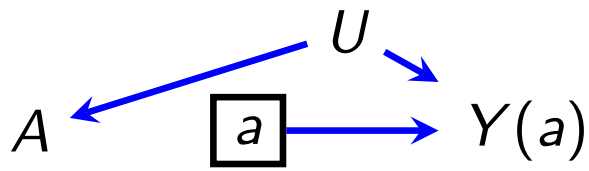


- Recovers what we did earlier.

Counterfactual Independences From A Graph

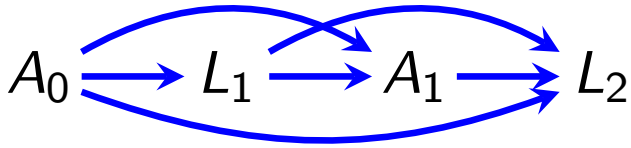
- Learned that DAGs represent independences in observed data distribution.
- Independences defining causal model are not on observed variables, but on counterfactuals.
- Cannot use original graph to read them off.
- Another type of graph works: Single World Intervention Graphs (SWIGs).
- Saw simple examples already:





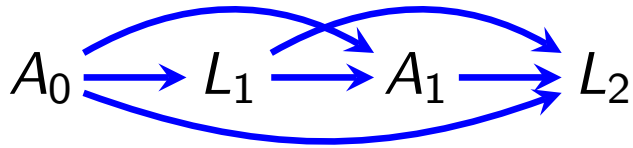
Sequential Treatment Settings

- Many causal problems are "games against Nature."
- We act (assign treatment), then Nature acts, then we act again, etc.



Sequential Treatment Settings

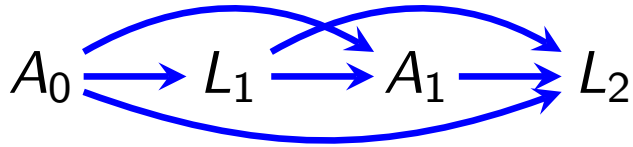
- Many causal problems are "games against Nature."
- We act (assign treatment), then Nature acts, then we act again, etc.



- Many observational datasets are generated in this way (observational studies, healthcare data, etc.)
- Want to estimate causal effects here, e.g. $E[Y(a_1, a_0)] - E[Y(a_1', a_0')]$.
- Will assume consistency, but need other assumptions.
- Is (conditional) ignorability true here?

Sequential Treatment Settings

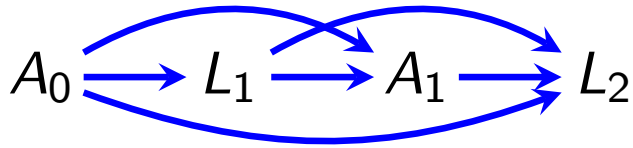
- Many causal problems are "games against Nature."
- We act (assign treatment), then Nature acts, then we act again, etc.



- Many observational datasets are generated in this way (observational studies, healthcare data, etc.)
- Want to estimate causal effects here, e.g. $E[Y(a_1, a_0)] - E[Y(a_1', a_0')]$.
- Will assume consistency, but need other assumptions.
- Is (conditional) ignorability true here?
- No. How to derive assumptions from a graph?

Constructing SWIGs (Sequential Treatment Example)

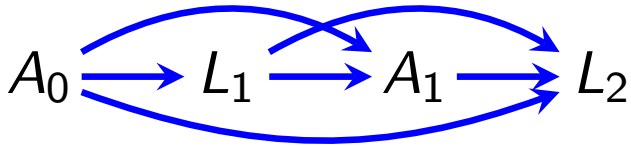
- Start with observed data graph:



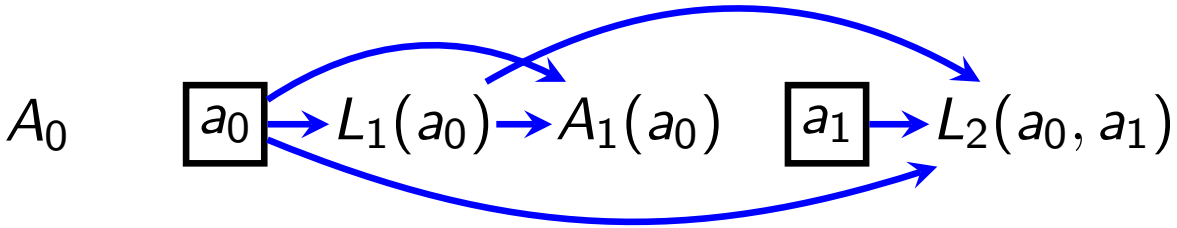
- Step 1: split treatments (A_1, A_2) into constant and random pieces.
- Step 2: constant pieces inherit outgoing edges, random pieces inherit incoming edges.
- Step 3: relabel downstream variables as counterfactuals.

Constructing SWIGs (Sequential Treatment Example)

- Start with observed data graph:

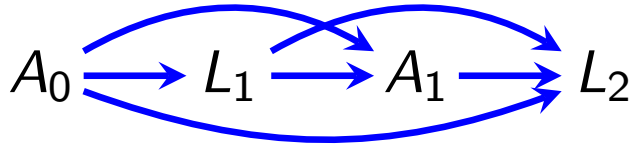


- Step 1: split treatments (A_1, A_2) into constant and random pieces.
- Step 2: constant pieces inherit outgoing edges, random pieces inherit incoming edges.
- Step 3: relabel downstream variables as counterfactuals.

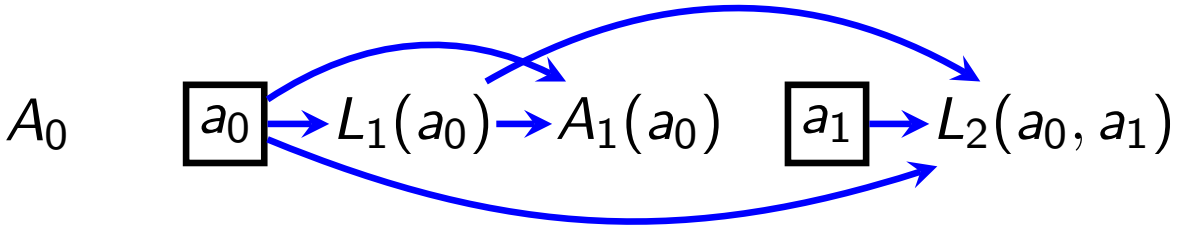


Constructing SWIGs (Sequential Treatment Example)

- Start with observed data graph:



- Step 1: split treatments (A_1, A_2) into constant and random pieces.
- Step 2: constant pieces inherit outgoing edges, random pieces inherit incoming edges.
- Step 3: relabel downstream variables as counterfactuals.



- Read off independences by d-separation:

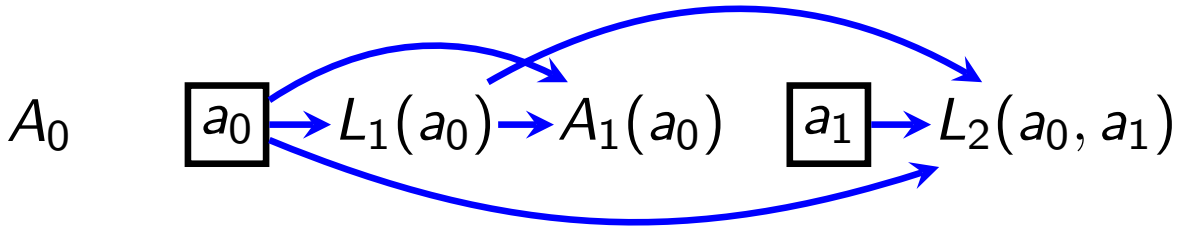
$$L_2(a_0, a_1) \perp\!\!\!\perp A_1(a_0) \mid L_1(a_0), A_0$$

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

A Causal Model Corresponds To Multiple SWIGs

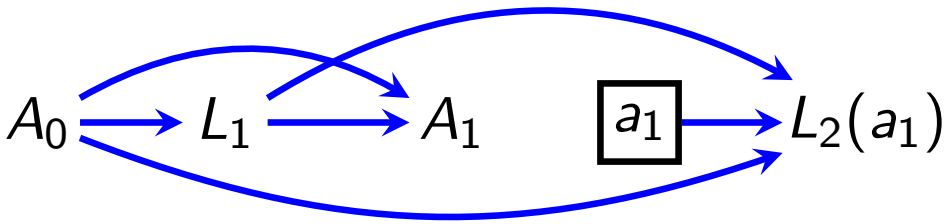
$$\{L_2(a_0, a_1), L_1(a_0)\} \perp\!\!\!\perp A_0$$

due to:



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

due to:



Identifying Two Treatment Effect Using $\perp\!\!\!\perp$ From SWIGs

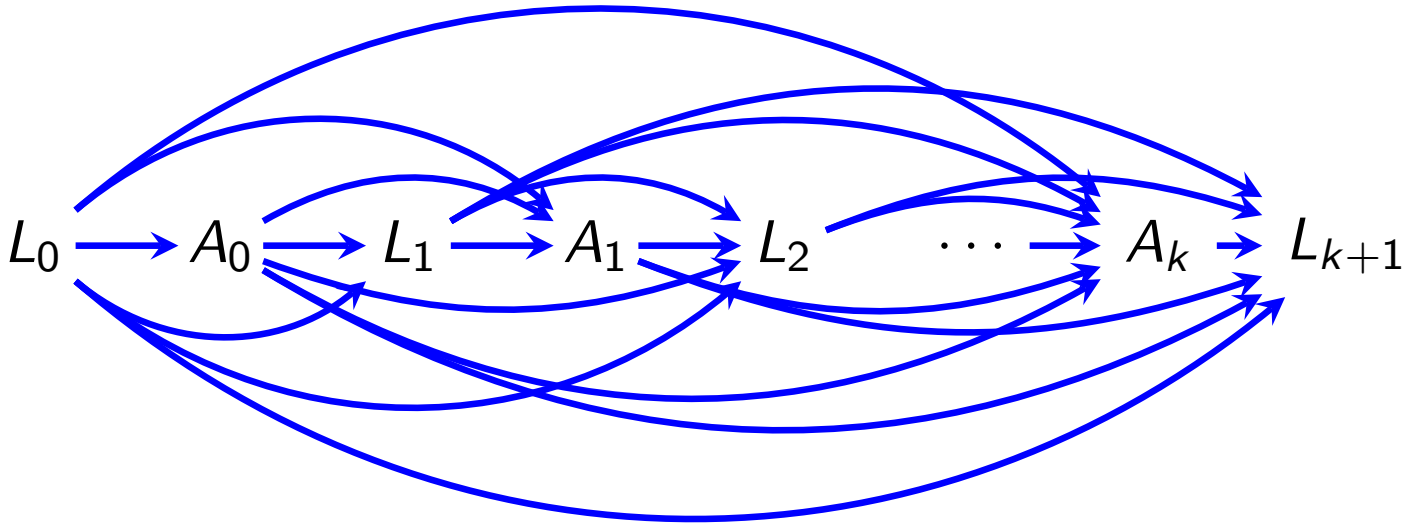
$$
\begin{aligned}
p(L_2(a_0, a_1)) &=^{p} \sum_{l_1} p(L_2(a_0, a_1) \mid L_1(a_0) = l_1) p(L_1(a_0) = l_1) \\
&=^{1} \sum_{l_1} p(L_2(a_0, a_1) \mid L_1(a_0) = l_1, a_0) p(L_1(a_0) = l_1 \mid a_0) \\
&=^{c} \sum_{l_1} p(L_2(a_1) \mid l_1, a_0) p(l_1 \mid a_0) \\
&=^{2} \sum_{l_1} p(L_2(a_1) \mid A_1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

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

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

Games Vs Nature

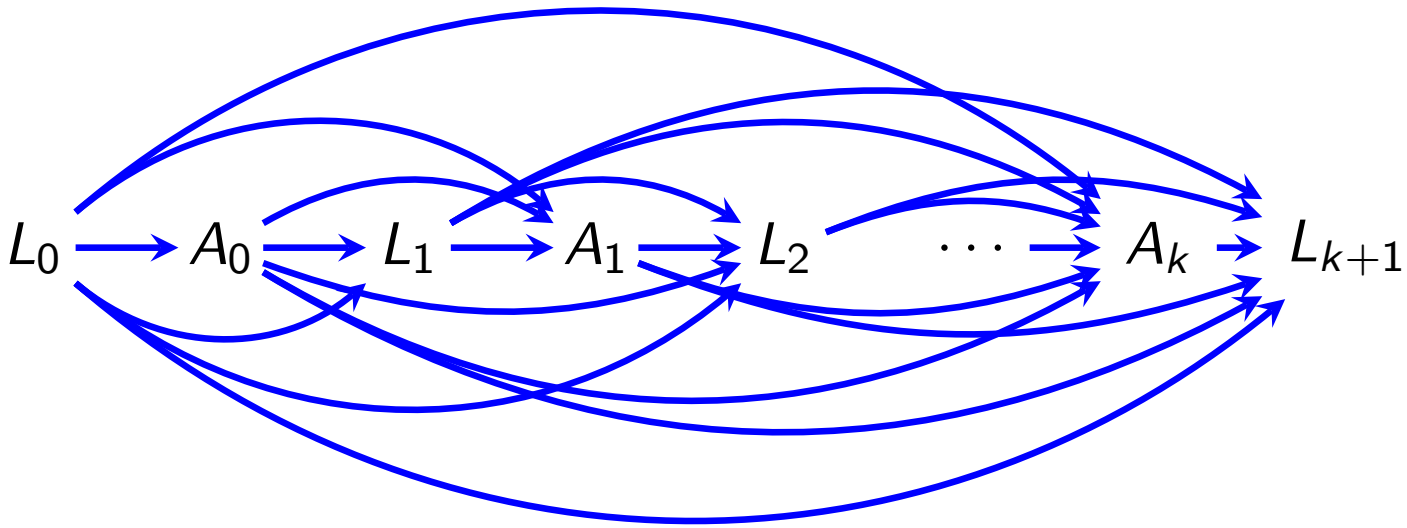
- General version of the "game vs Nature" (k treatments, k outcomes).



- Interested in $E[L_{k+1}(a_1, \ldots, a_k)] - E[L_{k+1}(a'_1, \ldots, a'_k)]$, other things.
- Could identify this for $k = 2$. Can we do this in general?

Games Vs Nature

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

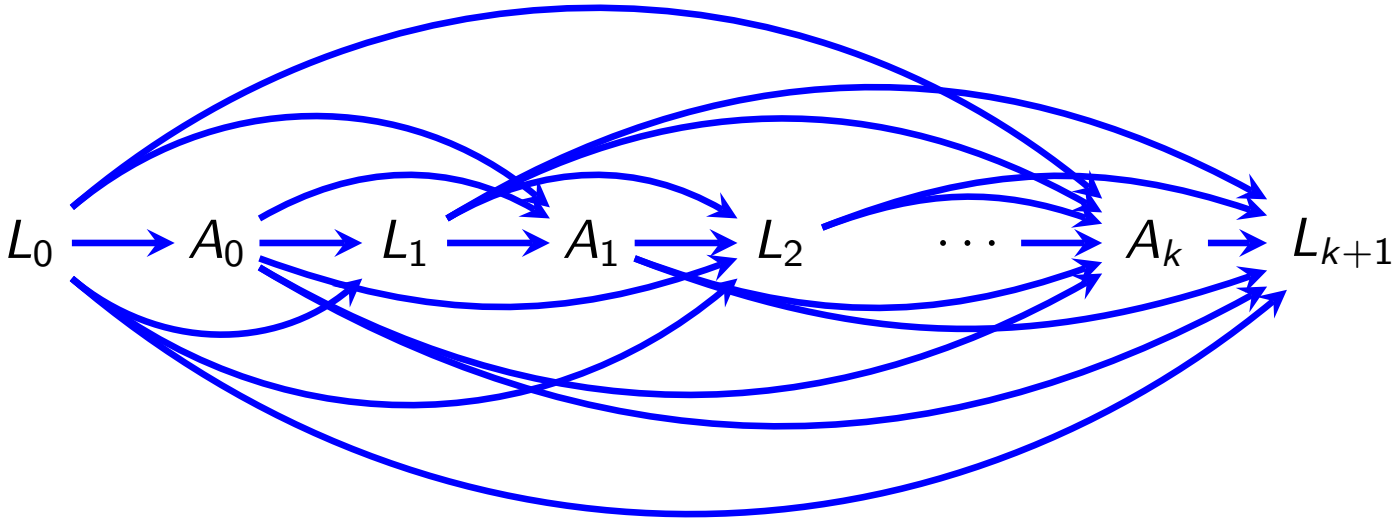


- Interested in $E[L_{k+1}(a_1, \ldots, a_k)] - E[L_{k+1}(a'_1, \ldots, a'_k)]$, other things.
- Could identify this for $k = 2$. Can we do this in general?
- Yes, via the g-computation algorithm.

G-Computation (Old Way)

- Can derive the following assumptions (sequential ignorability) using SWIGs:

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

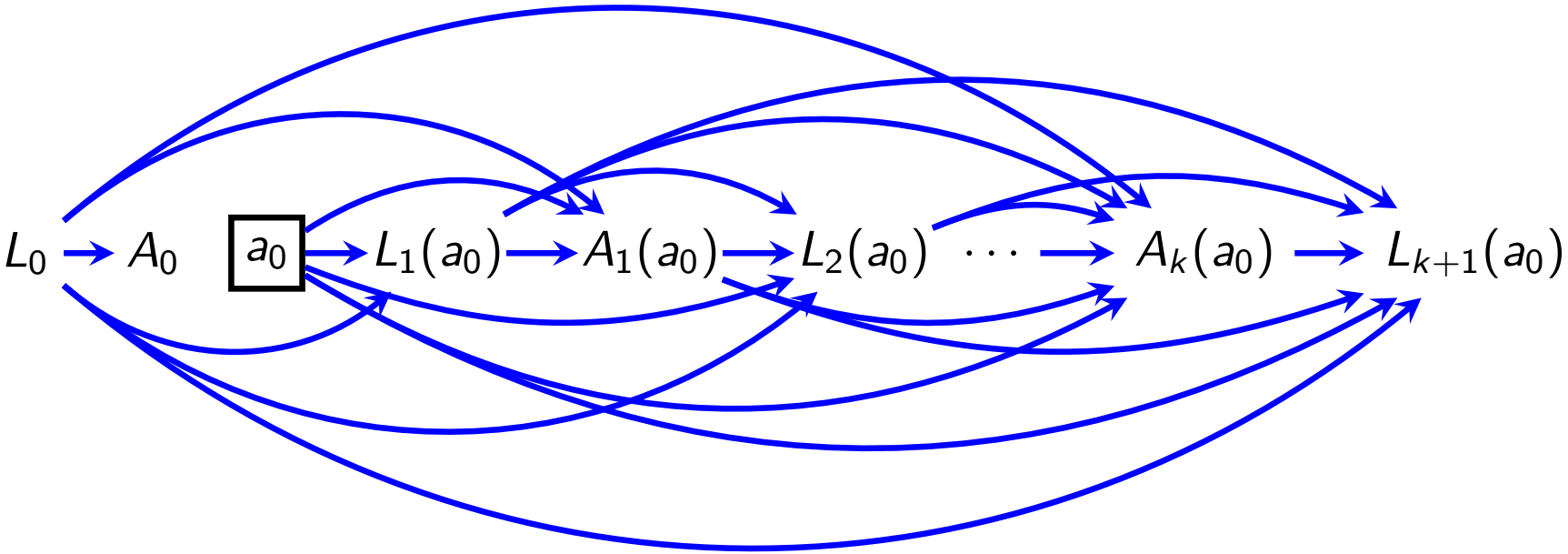


- Notation: $L_{<i} = \{L_1, \ldots, L_{i-1}\}$, $A_{<i} = \{A_1, \ldots, A_{i-1}\}$, past of $A_i = L_{<(i-1)} \cup A_{<i}$.

G-Computation (Old Way)

- Can derive the following assumptions (sequential ignorability) using SWIGs:

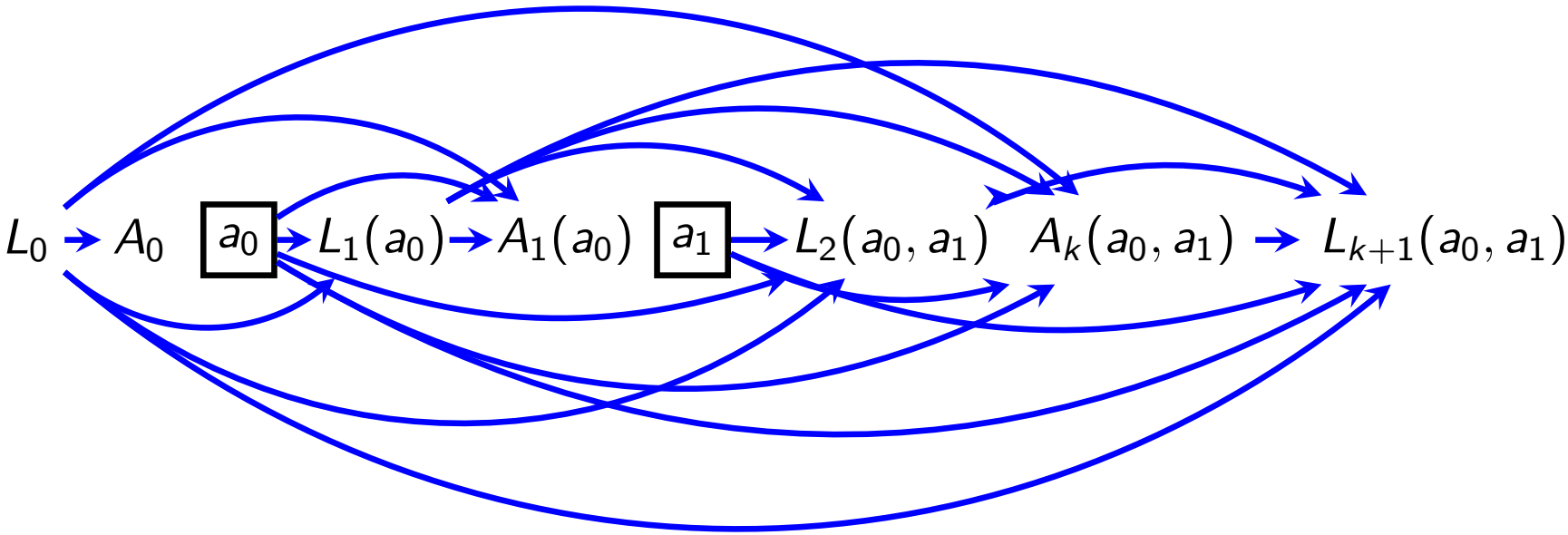
$$(\forall i \in \{1, \ldots, k+1\}) \left(\{L_i(a_{<i}), \ldots, L_{k+1}(a_{<k+1})\} \perp\!\!\!\perp A_{i-1} \mid \text{past of } A_{i-1}\right).$$



G-Computation (Old Way)

- Can derive the following assumptions (sequential ignorability) using SWIGs:

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



G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=2}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot (p(L_1(a_0) \mid L_0) p(L_0))$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=2}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot (p(L_1(a_0) \mid a_0, L_0) p(L_0))$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=2}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot (p(L_1 \mid a_0) p(L_0))$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=3}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot (p(L_2(a_1, a_0) \mid L_1(a_0), L_0)) \cdot$$

$$\left(\prod_{j=0}^{1} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0,\ldots,a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=3}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))\right) \cdot (p(L_2(a_1,a_0) \mid L_1(a_0), a_0, L_0)) \cdot$$

$$\left(\prod_{j=0}^{1} p(L_j \mid L_{<j}, a_{<j})\right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=3}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot (p(L_2(a_1)) \mid L_1, a_0, L_0)) \cdot$$

$$\left(\prod_{j=0}^{1} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0,\ldots,a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=3}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))\right) \cdot (p(L_2(a_1)) \mid a_1, L_1, a_0, L_0)) \cdot$$

$$\left(\prod_{j=0}^{1} p(L_j \mid L_{<j}, a_{<j})\right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=3}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot (p(L_2 \mid a_1, L_1, a_0, L_0)) \cdot \left(\prod_{j=0}^{1} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=3}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot \left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0,\ldots,a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))\right) \cdot p(L_3(a_2,a_1,a_0) \mid L_2(a_1,a_0), L_1(a_0), L_0) \cdot$$

$$\left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j})\right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0,\ldots,a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))\right) \cdot p(L_3(a_2,a_1,a_0) \mid L_2(a_1,a_0), L_1(a_0), a_0, L_0)$$

$$\left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j})\right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot p(L_3(a_2, a_1) \mid L_2(a_1), L_1, a_0, L_0) \cdot$$

$$\left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot p(L_3(a_2, a_1) \mid L_2(a_1), a_1, L_1, a_0, L_0) \cdot$$

$$\left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0,\ldots,a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))\right) \cdot p(L_3(a_2) \mid L_2, a_1, L_1, a_0, L_0) \cdot$$

$$\left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j})\right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0,\ldots,a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))\right) \cdot p(L_3(a_2) \mid a_2, L_2, a_1, L_1, a_0, L_0) \cdot$$

$$\left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j})\right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot p(L_3 \mid a_2, L_2, a_1, L_1, a_0, L_0) \cdot$$

$$\left(\prod_{j=0}^{2} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=4}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot \left(\prod_{j=0}^{3} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

$$= \sum_{L_{<(k+1)}} \left(\prod_{i=5}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)})) \right) \cdot \left(\prod_{j=0}^{4} p(L_j \mid L_{<j}, a_{<j}) \right)$$

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

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

G-Computation Derivation

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

implies $p(L_{k+1}(a_0, \ldots, a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

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

- This is just g-formula. Why did we do this the hard way?

G-Computation Derivation

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

implies $p(L_{k+1}(a_0,\ldots,a_k))$ is equal to

$$\sum_{L_{<(k+1)}} \prod_{i=1}^{k+1} p(L_i(a_{<i}) \mid L_{<i}(a_{<(i-1)}))$$

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

- This is just g-formula. Why did we do this the hard way?
- One reason is to give you an idea for why g-formula is true.
- There is another reason we will revisit when we talk about hidden variables.

Estimation For G-Computation

- Target: $\sum_{L_{<(k+1)}} \left(\prod_{j=0}^{k+1} p(L_j \mid L_{<j}, a_{<j}) \right)$.
- What we did with $k = 1$: model outcome mean (parametric g-formula), or model propensity score (IPW).
- Can generalize to any k!

Parametric G-Formula For Multiple Treatments

- Target: $\sum_{L_{<(k+1)}} \left(\prod_{j=1}^{k+1} p(L_j \mid L_{<j}, a_{<j}) \right)$.
- The parametric g-formula way is to model $p(L_j \mid L_{<j}, a_{<j})$ for each j.
- For L_{k+1} term can model expectation of L_{k+1}, for other terms we model densities.
- Density estimation is harder than mean estimation.
- Have to evaluate intractable sums/integrals for large k!

Sequential Parametric G-Formula Step By Step Guide

Given n data points on $\vec{A}, \mathbf{L}$, and assuming sequential ignorability and consistency:

1. Posit statistical models for $p(L_j \mid A_{<j}, L_{<j}; \alpha_j)$ for $j = 1, \ldots, k+1$.
2. Fit models by MLE, yielding $\hat{\alpha}_j$, $j = 1, \ldots, k-1$.
3. Obtain $E[L_{k+1}(\vec{a})]$ by:
   - For every row $i = 1, \ldots, n$, and timepoint $j = 1, \ldots, k+1$, sequentially sample m_ith sample of r total:
   - $L_j^{m_i} \sim p(L_j \mid \vec{a}_{<j}, L_0^i, L_1^{m_i}, \ldots, L_{j-1}^m; \hat{\alpha}_j)$

$$\text{return } \frac{1}{n \cdot r} \sum_{i=1}^{n} \sum_{m_i} L_{k+1}^{m_i}$$

4. Report confidence intervals using bootstrap. May compute in closed form sometimes, will skip for now.

Sequential Parametric G-Formula Step By Step Guide

Given n data points on $\vec{A}, \mathbf{L}$, and assuming sequential ignorability and consistency:

1. Posit statistical models for $p(L_j \mid A_{<j}, L_{<j}; \alpha_j)$ for $j = 1, \ldots, k+1$.
2. Fit models by MLE, yielding $\hat{\alpha}_j$, $j = 1, \ldots, k-1$.
3. Obtain $E[L_{k+1}(\vec{a})]$ by:
   - For every row $i = 1, \ldots, n$, and timepoint $j = 1, \ldots, k+1$, sequentially sample m_ith sample of r total:
   - $L_j^{m_i} \sim p(L_j \mid \vec{a}_{<j}, L_0^i, L_1^{m_i}, \ldots, L_{j-1}^m; \hat{\alpha}_j)$

$$\text{return } \frac{1}{n \cdot r} \sum_{i=1}^{n} \sum_{m_i} L_{k+1}^{m_i}$$

4. Report confidence intervals using bootstrap. May compute in closed form sometimes, will skip for now.

- Note: Do sampling twice! Once to evaluate integral for $E[L_{k+1}(\vec{a})] = \sum_{L_{<(k+1)}} \left(\prod_{j=1}^{k+1} p(L_j \mid L_{<j}, a_{<j}) \right)$.
- And once to obtain confidence intervals for $E[L_{k+1}(\vec{a})]$.

Sequential Parametric G-Formula Pros/Cons

- Pros:
  - Most efficient thing to do if you know the models.
  - Seems fairly robust to misspecification in practice.
  - Can do in closed form for simple models.
  - Conceptually simple: special case of likelihood weighting after g-formula is applied.
- Cons:
  - Have to do a lot of modeling.
  - Intractable in general.
  - Sampling is computationally intensive (Bayesians can avoid sampling twice – why?)
  - Sampling trajectories can be unstable (more on this in dynamic Bayesian network literature).

Doing Sums Efficiently?

- In machine learning, $\perp\!\!\!\perp$ are exploited to perform intractable sums efficiently using belief propagation.
- Not doing this here, why? Two reasons.

Doing Sums Efficiently?

- In machine learning, $\perp\!\!\!\perp$ are exploited to perform intractable sums efficiently using belief propagation.
- Not doing this here, why? Two reasons.

1 $\perp\!\!\!\perp$ = missing edges. Often no reason to expect missing edges in most causal inference problems!

Example: patient current state depends on entire case history.

Sometimes can redefine state to get a Markov property.

Doing Sums Efficiently?

- In machine learning, $\perp\!\!\!\perp$ are exploited to perform intractable sums efficiently using belief propagation.
- Not doing this here, why? Two reasons.

1 $\perp\!\!\!\perp$ = missing edges. Often no reason to expect missing edges in most causal inference problems!

Example: patient current state depends on entire case history.

Sometimes can redefine state to get a Markov property.

2 We will revisit a second reason when we talk about hidden variables: "Verma constraints."

IPW For Multiple Treatments

- Target: $\sum_{L_{<(k+1)}} \left(\prod_{j=1}^{k+1} p(L_j \mid L_{<j}, a_{<j}) \right)$.
- The IPW way is to model $p(A_j \mid L_{<j}, a_{<j})$ for each j.
- A_j are typically binary, so no need to model densities.
- Method is called marginal structural models.

What Is A Structural Model?

- A statistical model is a set of densities under restrictions.
- Used to restrict observed data distribution.
- Example: $E[Y \mid \vec{x}] = w_0 + \sum_i w_i x_i$.
- A causal model is a set of (counterfactual) densities under ($\perp\!\!\!\perp$) restrictions.
- Example: NPSEM-IE.
- A structural model is (loosely) a parametric model but for counterfactuals.
- Examples: $E[Y(a_1, a_2)] = w_0 + w_1 a_1 + w_2 a_2$,
  $E[Y(a_1, a_2) \mid X] = w_0 + w_1 a_1 + w_2 a_2 + w_x X + w_{2\times x} a_2 x$.
- Cannot fit structural models directly from data!

Marginal Structural Model (MSM)

- A marginal structural model is an assumption on a specific counterfactual marginal (usually outcome):

$$E[L_{k+1}(a_1, a_2, \ldots, a_k)] = w_0 + \sum_{i=1}^{k} w_i a_i$$

- Cannot do regression on $p(\vec{A}, \mathbf{L})$ directly, since $E[L_{k+1}(a_1, a_2, \ldots, a_k)] \neq E[L_{k+1} \mid a_1, a_2, \ldots, a_k]$.
- Under sequential ignorability and consistency, can reweigh the data first using $p(a_j \mid a_{<j}, L_{<j})$, then use a regression model.
- Many regression models allow rows to be weighted (R language's `glm(.)` function has a weights argument).

Sequential IPW (MSM) Step By Step Guide

Given n data points on $\vec{A}, \mathbf{L}$, and assuming sequential ignorability and consistency:

1. Posit statistical models for $p(A_j \mid A_{<j}, L_{<j}; \beta_j)$ for $j = 1, \ldots, k+1$.
2. Fit models by MLE, yielding $\hat{\beta}_j$, $j = 1, \ldots, k-1$.
3. Posit marginal structural model for $E[L_{k+1}(a_1, a_2, \ldots, a_k); \gamma]$.
4. Obtain $E[L_{k+1}(a_1, a_2, \ldots, a_k)]$ by:
   - For every row $i = 1, \ldots, n$, and every $j = 1, \ldots, k$, evaluate $p(a_j \mid L^i_{<j}, a_{<j}; \hat{\beta}_j)$.
   - Fit $E[L_{k+1}(a_1, a_2, \ldots, a_k); \gamma]$ by weighted MLE, with each row i having weight

$$\frac{1}{\prod_{j=1}^{k} p(a_j \mid a_{<j}, L^i_{<j}; \hat{\beta}_j)},$$

   return predicted $E[L_{k+1}(a_1, a_2, \ldots, a_k); \hat{\gamma}]$.
5. Report confidence intervals using bootstrap.

More on Sequential IPW

- Why do we need a structural model for L_{k+1} at all?
- We didn't for $k = 1$!

More on Sequential IPW

- Why do we need a structural model for L_{k+1} at all?
- We didn't for $k = 1$!
- Remember for $k = 1$, IPW reweighted Y for which $\mathbb{I}(A = a) = 1$.
- For large k, number of rows where $\mathbb{I}(\vec{A} = \vec{a})$ goes to 0.
- For small k, may not need a model.

Sequential IPW (MSM) Pros/Cons

- Pros:
  - Easy modeling problem (binary treatments).
  - Easy to fit MSMs (regression models allow row weights), no extra programming!
  - Computationally efficient
- Cons:
  - Statistically inefficient (big intervals).
  - Severe instability issues (dividing by products of small numbers).

Sequential IPW (MSM) Pros/Cons

- Pros:
  - Easy modeling problem (binary treatments).
  - Easy to fit MSMs (regression models allow row weights), no extra programming!
  - Computationally efficient
- Cons:
  - Statistically inefficient (big intervals).
  - Severe instability issues (dividing by products of small numbers).
- Very popular in practice because of simplicity.

Other Estimation Strategies

- As with single treatment case, there are other estimation strategies than IPW and parametric g-formula.
- Notable ones:
  - Robust methods: model both $p(L_j \mid a_{<j}, L_{<j})$ and $p(a_j \mid a_{<j}, L_{<j})$. Remain consistent if either L_j or A_j models are correct.
  - Structural nested models – more complicated structural models that condition on the past.
- Can you think of more? Open problem: likely more are possible, perhaps with machine learning ideas!

Next time: Causal DAG Models With Hidden Variables.