Causal Inference for High-Dimensional Data

Atlantic Causal Conference

Overview

- Conditional Independence
- Directed Acyclic Graph (DAG) Models
  - factorization and d-separation
  - Markov equivalence
- Structure Learning : Fully observed
  - The PC Algorithm
  - The IDA Algorithm
- Structure Learning with Hidden Variables
  - The FCI Algorithm and RFCI algorithms
  - Current Work: Nested Markov Models; Inequalities
- Causal DAG Models

Conditional Independence

Independence

Recall the following equivalent characterizations of independence, $X \perp\!\!\!\perp Y$:

$$\begin{aligned} P(X = x \mid Y = y) &= P(X = x) \\ P(Y = y \mid X = x) &= P(Y = y) \\ P(X = x, Y = y) &= P(X = x)P(Y = y) \\ P(X = x, Y = y) &= f(x)g(y), \quad \text{for some fns } f(\cdot) \text{ and } g(\cdot). \end{aligned}$$

Intuitively, if $X \perp\!\!\!\perp Y$ then *knowledge of* X *provides no information about* Y.

If X and Y are binary variables then

$$X \perp\!\!\!\perp Y \;\Leftrightarrow\; RD(X, Y) = 0 \;\Leftrightarrow\; \log RR(X, Y) = 0 \;\Leftrightarrow\; \log OR(X, Y) = 0.$$

If X and Y are jointly Gaussian then:

$$X \perp\!\!\!\perp Y \;\Leftrightarrow\; Cov(X, Y) = 0 \;\Leftrightarrow\; Corr(X, Y) = 0$$

Conditional Independence (I)

Characterizations of conditional independence, $X \perp\!\!\!\perp Y \mid Z$:

$$
\begin{aligned}
P(X = x \mid Y = y, Z = z) &= P(X = x \mid Z = z), \\
P(X = x, Y = y \mid Z = z) &= P(X = x \mid Z = z) P(Y = y \mid Z = z), \\
P(X = x, Y = y, Z = z) &= f(x, z) g(y, z), \quad \text{for some fns } f(\cdot, \cdot) \text{ and } g(\cdot, \cdot), \\
P(X = x, Y = y, Z = z) &= P(X = x, Z = z) P(Y = y, Z = z) / P(Z = z).
\end{aligned}
$$

Intuitively, *if $X \perp\!\!\!\perp Y \mid Z$, then if Z is known, X provides no further knowledge of Y, and Y provides no further knowledge of X.*

The above extends in the obvious way to define $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}$ for vectors $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$.

Conditional Independence (II)

If X and Y are binary variables then

$$\begin{aligned} X \perp\!\!\!\perp Y \mid Z &\Leftrightarrow RD(X, Y \mid Z = z) = 0 \\ &\Leftrightarrow \log RR(X, Y \mid Z = z) = 0 \\ &\Leftrightarrow \log OR(X, Y \mid Z = z) = 0. \end{aligned}$$

i.e. we have independence in every stratum defined by Z.

If X, Y, Z follow a joint normal distribution then

$$\begin{aligned} X \perp\!\!\!\perp Y \mid Z &\Leftrightarrow Cov(X, Y \mid Z) = 0 \\ &\Leftrightarrow Corr(X, Y \mid Z) = 0 = 0 \\ &\Leftrightarrow \beta_{yx.z} = 0, \end{aligned}$$

where $\beta_{yx.z}$ is the coefficient on X in the linear regression of Y on X and Z.

‘Graphoid’ Axioms of Conditional Independence

- Symmetry

$$X \perp\!\!\!\perp Y \mid Z \quad \Rightarrow \quad Y \perp\!\!\!\perp X \mid Z$$

- Decomposition

$$X \perp\!\!\!\perp Y, W \mid Z \quad \Rightarrow \quad X \perp\!\!\!\perp Y \mid Z \text{ and } X \perp\!\!\!\perp W \mid Z$$

- Weak Union

$$X \perp\!\!\!\perp Y, W \mid Z \quad \Rightarrow \quad X \perp\!\!\!\perp Y \mid Z, W$$

- Contraction

$$X \perp\!\!\!\perp Y \mid Z \text{ and } X \perp\!\!\!\perp W \mid Y, Z \quad \Rightarrow \quad X \perp\!\!\!\perp Y, W \mid Z$$

- Intersection (not true in general, but holds if density is strictly positive):

$$X \perp\!\!\!\perp Y \mid W, Z \text{ and } X \perp\!\!\!\perp W \mid Y, Z \quad \Rightarrow \quad X \perp\!\!\!\perp Y, W \mid Z$$

Example where Intersection Fails

Consider three binary random variables (X, Y, Z)

$$X \sim \text{Bernoulli}(0.5) \qquad X = Y = Z.$$

Then we have $X \perp\!\!\!\perp Y \mid Z$, and $Y \perp\!\!\!\perp Z \mid X$, but $X \not\perp\!\!\!\perp Y, Z$.

Another counterexample

Might expect that $X \perp\!\!\!\perp Y$ and $X \perp\!\!\!\perp Z$ imply $X \perp\!\!\!\perp Y, Z$ but this is not true in general:

Let X and Y be two independent coin flips;

let Z be a bell that rings whenever $X = Y$.

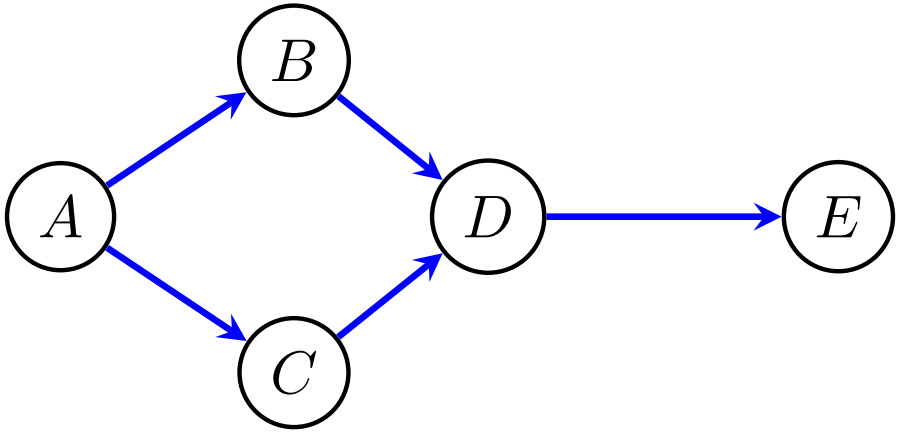
Then we have $Z \perp\!\!\!\perp X$ and $Z \perp\!\!\!\perp Y$ but not $Z \perp\!\!\!\perp Y, X$.

(This implication does hold under multivariate normality, but not more generally.)

$\Rightarrow$ working with conditional independence can be quite tricky

DAG Models

Directed Acyclic Graphs (DAGs)



- The graph is *acyclic* because we do not permit directed cycles: $A \rightarrow \cdots \rightarrow A$
- If there is an edge $A \rightarrow B$ then A is said to be a *parent* of B, and B is a *child* of A.
  Two vertices joined by an edge are said to be *adjacent*.
- The set of parents of a vertex A, is denoted pa(A).

  Example:

$$\text{pa}(D) = \{B, C\}.$$

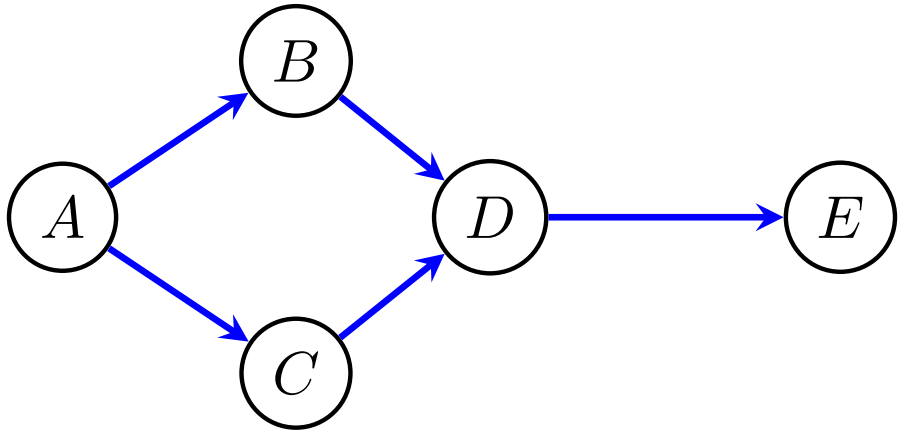
- We use $\mathbf{V}$ to denote the set of all vertices in the DAG.

Factorization Associated with a DAG

We associate the following factorization of a joint distribution with a DAG:

$$P(\mathbf{V}) = \prod_{X \in \mathbf{V}} P(X \mid \mathsf{pa}(X))$$

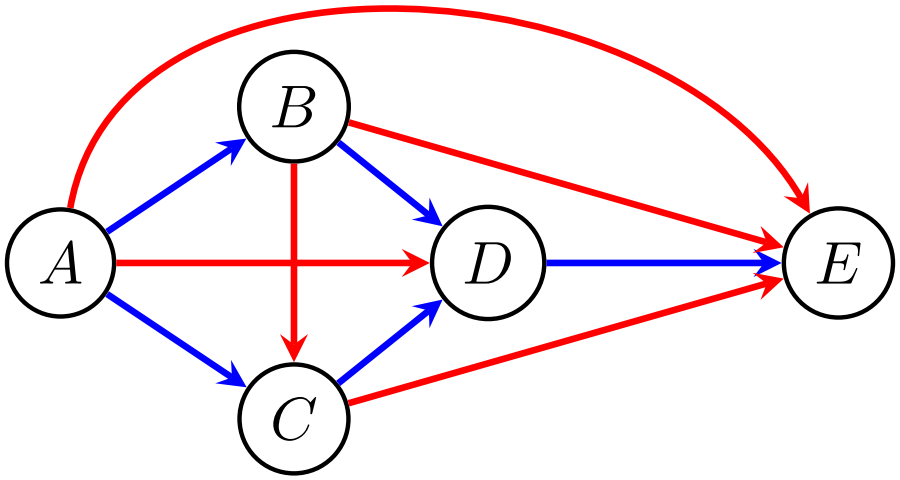
Example:



$$P(A, B, C, D, E)$$
$$= \quad P(A) \times P(B \mid A) \times P(C \mid A) \times P(D \mid B, C) \times P(E \mid D)$$

Complete DAGs

A DAG is *complete* if every pair of vertices are adjacent.

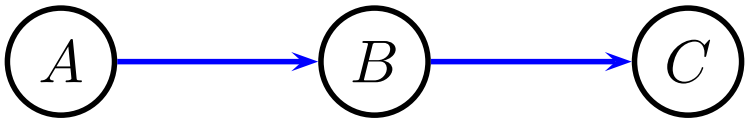


Every distribution factorizes w.r.t. a complete graph via the chain rule of probability:

$$
\begin{aligned}
& P(A, B, C, D, E) \\
= \quad & P(A)\, P(B \mid A)\, P(C \mid A, B)\, P(D \mid A, B, C)\, P(E \mid A, B, C, D)
\end{aligned}
$$

Factorization wr.t. a graph that is not complete implies conditional independence.

Conditional Independence



$$\Rightarrow \quad P(A,B,C) = P(A)P(B|A)P(C|B)$$

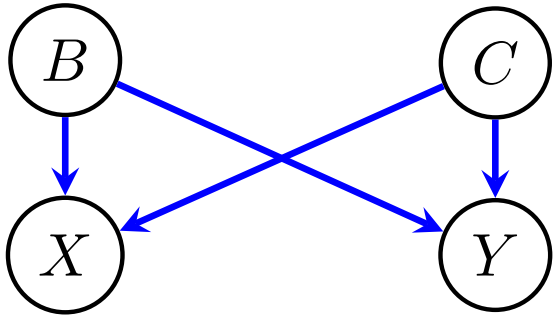
$$\Rightarrow \qquad A \perp\!\!\!\perp C \mid B$$

Q: Is there a general graphical condition that we can use to read conditional independence from the graph?

A: Yes. The absence of a special kind of path, called 'd-connecting' between X and Y given $\mathbf{Z}$ in graph G will indicate $X \perp\!\!\!\perp Y \mid \mathbf{Z}$ in any distribution factorizing according to G.

d-separation

Genetics Example (I)



Beth and Charlie have two children, Xavier and Yugo.

Here each variable represents a genotype taking one of three states:

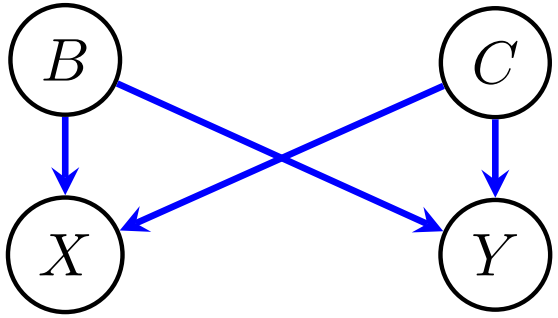
(AA) *homozygous dominant*, (Aa) *heterozygous*, (aa) *homozygous recessive*.

Intuitively, if we know B and C then X tells us nothing about Y, so we have:

$$X \perp\!\!\!\perp Y \mid B, C$$

but unconditionally we have $X \not\perp\!\!\!\perp Y$.

Genetics Example (II)



(Under random mating) we would expect that Beth and Charlie's genes are independent, so we have:

$$B \perp\!\!\!\perp C.$$

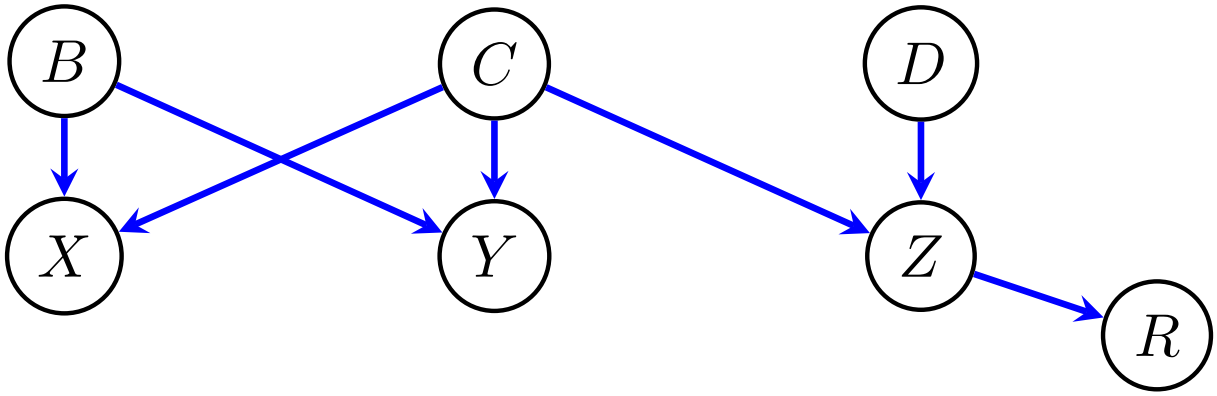
However, if we know X or Y (or both) then clearly B and C are dependent.

For example, if $X = Aa$, then if $B = aa$, we know that C is either Aa or AA.

Hence:

$$B \not\perp\!\!\!\perp C \mid X, \qquad B \not\perp\!\!\!\perp C \mid Y, \qquad B \not\perp\!\!\!\perp C \mid X, Y$$

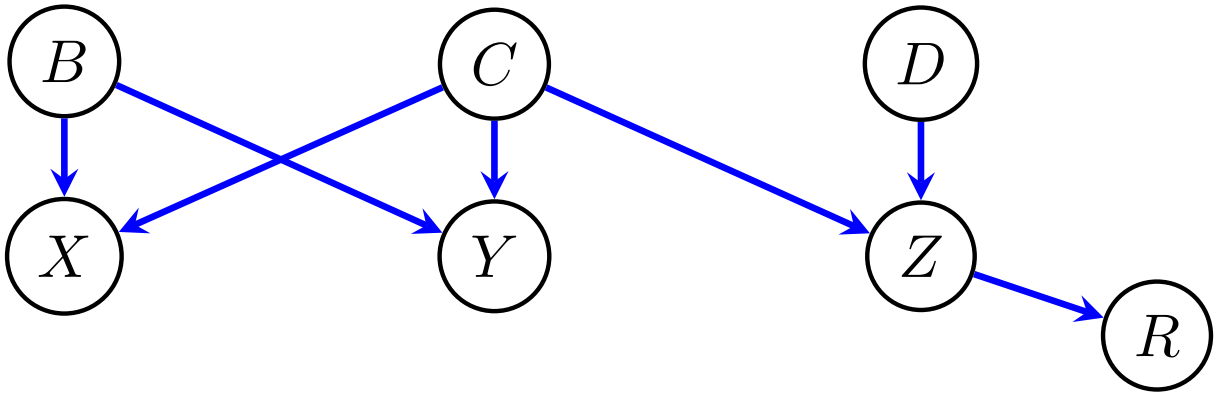
Genetics Example (III)



Charlie also has a child Zane, with Dorothy. R is the presence or absence in Zane of the recessive trait associated with the gene (e.g. red hair, corresponding to aa).

Knowing R, D and C become dependent: for example, if Zane does not have red hair, but Dorothy is aa, then we know that C is either AA or Aa.

Genetics Example (III)



Charlie also has a child Zane, with Dorothy. R is the presence or absence in Zane of the phenotype associated with the gene (e.g. red hair, corresponding to aa).

Knowing R, D and C become dependent: for example, if Zane does not have red hair, but Dorothy is aa, then we know that C is either AA or Aa.

Further, given R, D and X become dependent, since if (continuing the example) we know that C is either AA or Aa then this makes it less likely that Xavier or Yugo have red hair (than if it were possible that C is aa).

Similarly, D and Y are dependent given R.

Paths and Colliders

A *path* π between vertices X and Y consists of a sequence of distinct vertices that are adjacent. For example:

$$X \rightarrow A \leftarrow B \leftarrow C \rightarrow D \rightarrow Y$$

A non-endpoint vertex V on a path (between X and Y) is said to be a *collider* if the path takes the form $X \cdots \rightarrow V \leftarrow \cdots Y$.

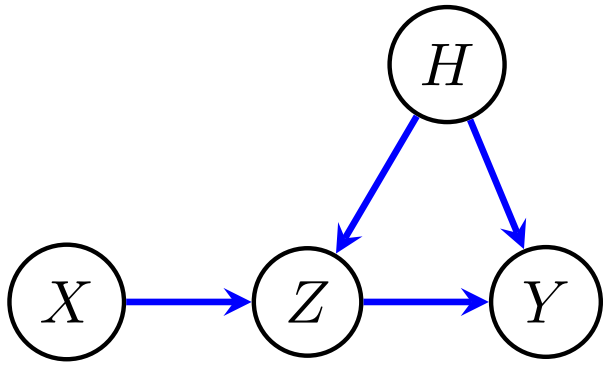
Non-endpoints that are not colliders are called *non-colliders*:

$$X \cdots \leftarrow V \rightarrow \cdots Y, \quad X \cdots \rightarrow V \rightarrow \cdots Y, \quad X \cdots \leftarrow V \leftarrow \cdots Y$$

Subtlety regarding colliders

A vertex is a collider or non-collider *with respect to* a given path.

A vertex may be a collider on one path and a non-collider on another (even between the same endpoints).



Here Z is a collider on the path $X \rightarrow Z \leftarrow H \rightarrow Y$,

but Z is a non-collider on the path $X \rightarrow Z \rightarrow Y$.

Grammar: . . . V is a collider **with respect to the path** . . .

Unconditional d-connection

A path π is said to *d-connect* X and Y *unconditionally* (or given the empty set, $\emptyset$) if X and Y are the endpoints of π, and there are no colliders on π.

In this case π takes one of three forms:

$$X \rightarrow \cdots \rightarrow Y$$

$$X \leftarrow \cdots \leftarrow Y$$

$$X \leftarrow \cdots \leftarrow T \rightarrow \cdots \rightarrow Y$$

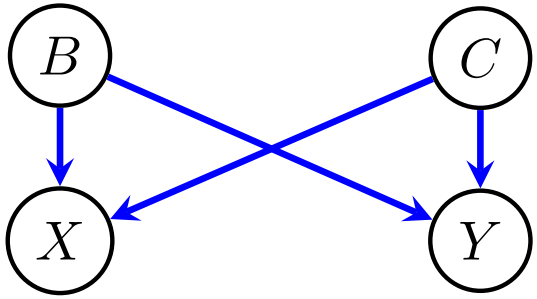
Such collider-free paths are sometimes called *treks*.

If there is no path d-connecting X and Y (given $\emptyset$) then X and Y are said to be *d-separated* (given $\emptyset$).

Unconditional d-connection & marginal independence

Lemma: In any distribution P factorizing according to a graph G, if X and Y are d-separated given the empty set then $X \perp\!\!\!\perp Y$ in P.

Example:



There are no paths d-connecting B and C given $\emptyset$ hence $B \perp\!\!\!\perp C$.

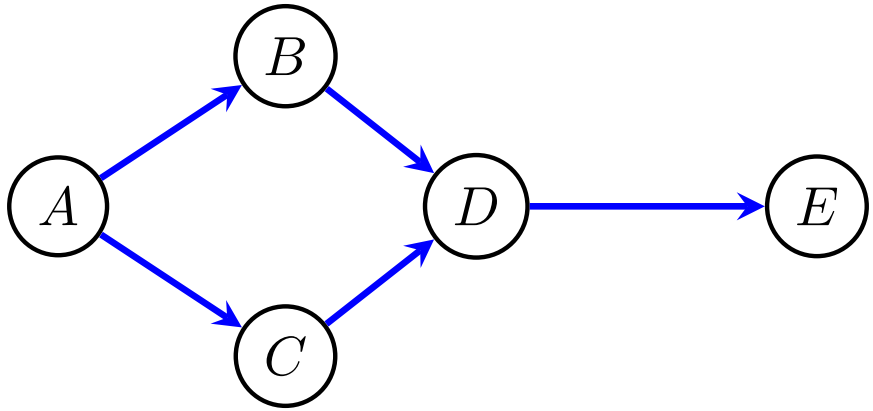
X and Y are d-connected given $\emptyset$, so the graph does not imply $X \perp\!\!\!\perp Y$.

Ancestors and Descendants

In a DAG a vertex X is said to be an *ancestor* of Y if there is a directed path containing one or more edges from X to Y: $X \to \cdots \to Y$, or if $X = Y$.

If X is an ancestor of Y then Y is said to be a *descendant* of X.

Note that every vertex is its own descendant and its own ancestor; every parent of X is an ancestor of X; every child of X is a descendant of X.



$$\mathsf{an}(E) = \{A, B, C, D, E\}, \qquad \mathsf{an}(D) = \{A, B, C, D\},$$
$$\mathsf{de}(A) = \{A, B, C, D, E\}, \qquad \mathsf{de}(B) = \{B, D, E\}.$$

Conditional d-connection & conditional independence

A path π is said to *d-connect* X *and* Y *conditional on a set* $\mathbf{Z}$ if X and Y are the endpoints of π and:

(i) Every non-collider on π is not in $\mathbf{Z}$, and

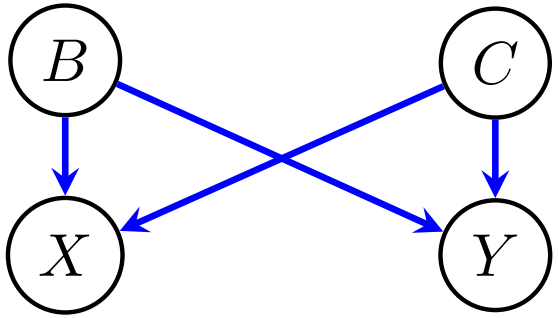
(ii) Every collider on π is an ancestor of $\mathbf{Z}$ (or is in $\mathbf{Z}$).

(Note that if $\mathbf{Z} = \emptyset$ then this definition reduces to the previous one.)

If there is no path d-connecting X and Y given $\mathbf{Z}$ then X and Y are said to be *d-separated given* $\mathbf{Z}$.

Theorem: In any distribution P factorizing according to G, if X and Y are d-separated given $\mathbf{Z}$ then $X \perp\!\!\!\perp Y \mid \mathbf{Z}$ in P.

Genetics Example Revisited (I)



There is no path d-connecting X and Y given $\mathbf{Z} = \{B, C\}$:

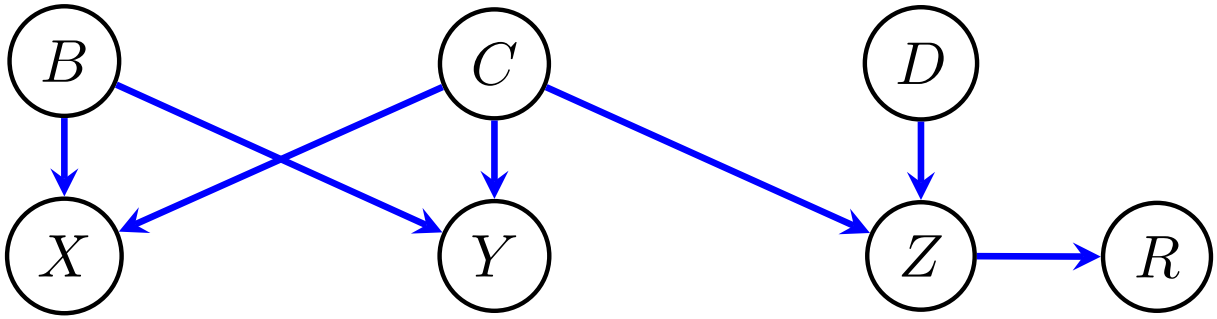
- On the path $X \leftarrow B \rightarrow Y$, B is a non-collider and in $\mathbf{Z}$;
- Similarly on the path $X \leftarrow C \rightarrow Y$, C is a non-collider and in $\mathbf{Z}$.

Hence the graph implies $X \perp\!\!\!\perp Y \mid \{B, C\}$.

However, the path $B \rightarrow X \leftarrow C$ d-connects given $\mathbf{Z} = \{X\}$, hence the graph does not imply $B \perp\!\!\!\perp C \mid X$.

The same holds for B and C given $\{Y\}$, also B and C given $\{X, Y\}$.

Genetics Example Revisited (II)



- C and D are d-connected given R by the path $C \rightarrow Z \leftarrow D$ because Z is a collider on the path, and is an ancestor of R, hence the graph does not imply $C \perp\!\!\!\perp D \mid R$.
- X and D are d-connected given R by the path $X \leftarrow C \rightarrow Z \leftarrow D$.
- B and D are d-separated given R since there are two paths:

$$B \rightarrow Y \leftarrow C \rightarrow Z \leftarrow D$$

$$B \rightarrow X \leftarrow C \rightarrow Z \leftarrow D$$

but X and Y are both colliders and neither Y nor X is an ancestor of R.

d-separation for sets of variables

So far we have considered d-separation between single variables X, Y given a set $\mathbf{Z}$.

We extend this to sets of variables as follows:

A set $\mathbf{X}$ is d-separated from a set $\mathbf{Y}$ given $\mathbf{Z}$ if for all $X \in \mathbf{X}$, $Y \in \mathbf{Y}$, X and Y are d-separated given $\mathbf{Z}$.

Theorem: In any distribution P factorizing according to G, if $\mathbf{X}$ and $\mathbf{Y}$ are d-separated given $\mathbf{Z}$ then $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}$ in P.

d-separation implies factorization

We have seen that if P factorizes according to G then d-separation implies conditional independence.

The reverse implication also holds:

If P has a joint density (w.r.t. a product measure) then if for all disjoint sets $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$, where $\mathbf{Z}$ may be empty,

$$\mathbf{X} \text{ and } \mathbf{Y} \text{ are d-separated given } \mathbf{Z} \text{ in } G \quad \Rightarrow \quad \mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z} \text{ in } P,$$

then P factorizes w.r.t. G.

Completeness of d-separation

We have seen that d-separation in G implies conditional independence in any distribution factorizing according to G.

In general d-connection does not imply dependence (because, for example, a joint distribution in which all variables are jointly independent trivially factorizes according to every graph).

However, we have the following:

If X and Y are d-connected given $\mathbf{Z}$ in G then there exists a distribution P that factorizes according to G, and $X \not\perp\!\!\!\perp Y \mid \mathbf{Z}$ in P.

(Further, it can be shown that for smooth distributions over the set of distributions that factorize, d-connection implies dependence for almost all distributions factoring according to G.)

Composition revisited

Note that if $\mathbf{X}$ is d-separated from $\mathbf{Y}$ given $\mathbf{Z}$ and $\mathbf{X}$ is d-separated from $\mathbf{W}$ given $\mathbf{Z}$ then $\mathbf{X}$ is d-separated from $\mathbf{W} \cup \mathbf{Y}$ given $\mathbf{Z}$.

This is the intuitive implication that we saw earlier did not hold in general for conditional independence.

However, it does hold for those conditional independence relations that follow from factorization w.r.t. a DAG.

Simple properties of d-connection

- If there is an edge $X \rightarrow Y$ or $X \leftarrow Y$ then X and Y are d-connected given every set $\mathbf{W} \subseteq \mathbf{V} \setminus \{X, Y\}$.

Simple properties of d-connection

- If there is an edge $X \rightarrow Y$ or $X \leftarrow Y$ then X and Y are d-connected given every set $\mathbf{W} \subseteq \mathbf{V} \setminus \{X, Y\}$.
- In a DAG $\mathcal{G}$ if there is no edge between X and Y then either
  - X is d-separated from Y given pa(X), *or*
  - Y is d-separated from X given pa(Y), (or both).

Simple properties of d-connection

- If there is an edge $X \rightarrow Y$ or $X \leftarrow Y$ then X and Y are d-connected given every set $\mathbf{W} \subseteq \mathbf{V} \setminus \{X, Y\}$.
- In a DAG $\mathcal{G}$ if there is no edge between X and Y then either
  - X is d-separated from Y given pa(X), *or*
  - Y is d-separated from X given pa(Y), (or both).

  Consequence:

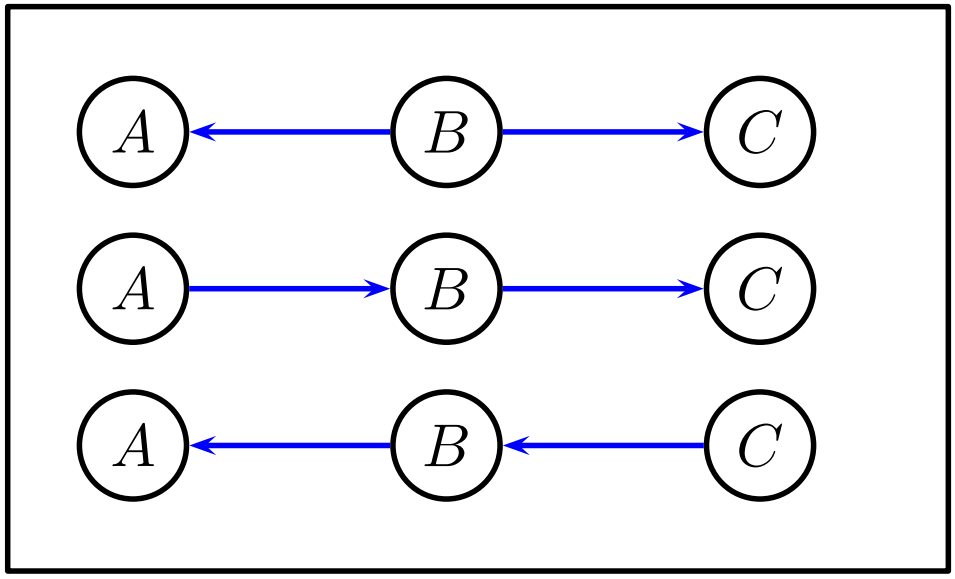
  There is an edge between X and Y in $\mathcal{G}$ iff X and Y are d-connected given every subset $\mathbf{W} \subseteq \mathbf{V} \setminus \{X, Y\}$.

  Equivalently:

  There is no edge between X and Y in $\mathcal{G}$ iff there exists a set $\mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\}$ such that X and Y are d-separated given $\mathbf{S}_{XY}$.

Markov Equivalence

Different graphs can represent the same set of distributions:



$$\Rightarrow \qquad A \perp\!\!\!\perp C \mid B$$

If two graphs G_1 and G_2 imply the same set of conditional independence relations via d-separation then they are said to be *Markov equivalent*.

The set of all DAGs that imply a given set of d-separation relations is called a Markov equivalence class.

Markov equivalence

If G_1 and G_2 are Markov equivalent then P factorizes w.r.t. G_1 if and only if P factorizes w.r.t. G_2.

Hence Markov equivalent DAGs are associated with the same sets of distributions

Characterization of Markov equivalence

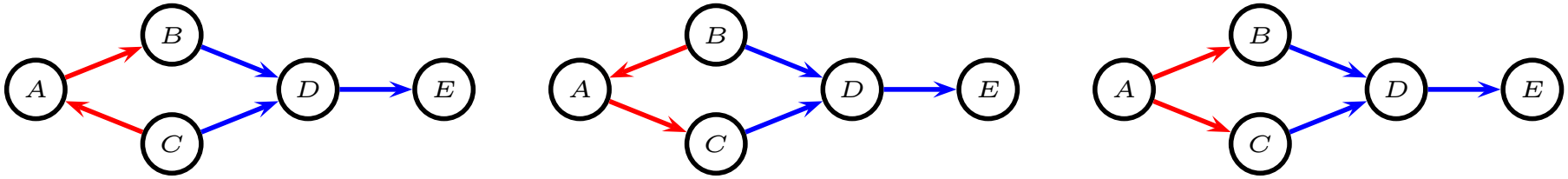
A set of vertices forms an *unshielded collider* in G if there are edges $X \to Y \leftarrow Z$, but X and Z are not adjacent.

Theorem: Two DAGs G_1 and G_2 are Markov equivalent if G_1 and G_2 have:

(a) the same adjacencies;

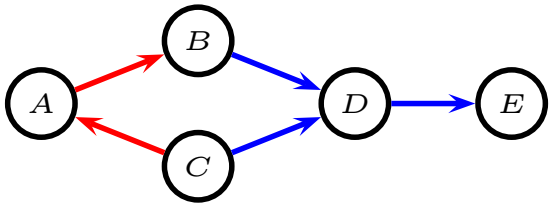
(b) the same unshielded colliders.

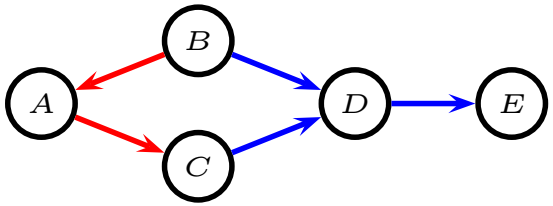
Example:

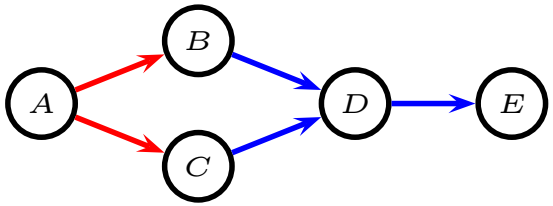


These graphs form a Markov equivalence class.

Representing a Markov equivalence class of DAGs

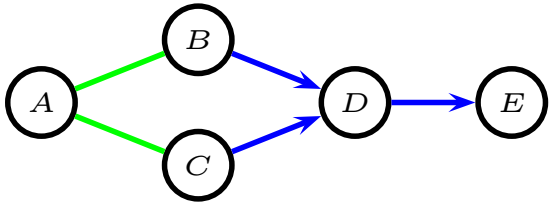






These graphs form a Markov equivalence class.

A whole class may be represented via a single graph with directed and undirected edges:



Use directed edges when an edge has the same orientation for all graphs in an equivalence class; otherwise use an undirected edge.

The resulting graph is called the ‘essential graph’ or the CPDAG (Completed Partially Directed Acyclic Graph) for the equivalence classe.

Faithfulness Assumption

If $P(X_V)$ factorizes according to a DAG $\mathcal{G}$ then:

$$\mathbf{X} \text{ d-separated from } \mathbf{Y} \text{ given } \mathbf{Z} \quad \Rightarrow \quad \mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z} \ [P]$$

However, in general there may be additional independence relations, so it need not be the case that:

$$\mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z} \quad \Rightarrow \quad \mathbf{X} \text{ d-separated from } \mathbf{Y} \text{ given } \mathbf{Z}$$

The faithfulness assumption assumes that there are no additional independence relations, so that:

$$\mathbf{X} \text{ d-separated from } \mathbf{Y} \text{ given } \mathbf{Z} \quad \Leftrightarrow \quad \mathbf{X} \perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z} \ [P]$$

Structure Learning

- Q: Given a distribution P that factorizes according to a unknown DAG $\mathcal{G}$, that satisfies the faithfulness assumption, what can be inferred regarded the structure of $\mathcal{G}$?
- Simple case: Fully observed

  $\mathcal{G}$ has vertex set V; we are given the joint distribution $P(X_V)$.
- Hard case: With hidden variables

  $\mathcal{G}$ has vertex set $O \cup H$; we are given the marginal distribution $P(X_O)$

Structure Learning: Fully observed

Structure Learning: Fully observed

Recall: no edge between X and Y iff there exists $\mathbf{S}_{XY} \subseteq \mathbf{V} \setminus \{X, Y\}$ such that X is d-separated from Y given $\mathbf{S}_{XY}$.

Suppose there is no edge between X and Y in $\mathcal{G}$, and $X \rightarrow Z \leftarrow Y$ forms an unshielded collider. If X and Y are d-separated given $\mathbf{S}_{XY}$ then $Z \notin \mathbf{S}_{XY}$.

(*Why?*)

$\Rightarrow$: If for every 'missing' edge (X, Y) in $\mathcal{G}$ we have a set $\mathbf{S}_{XY}$ such that X is d-separated from Y given $\mathbf{S}_{XY}$ then this is sufficient to identify the Markov equivalence class containing $\mathcal{G}$.

Inferring the Markov equivalence class from the separating sets S_{XY}

0. Start with a complete undirected graph.

1. For every pair (X, Y) if there exists a separating set $\mathbf{S}_{XY}$ then remove the edge $X \text{ — } Y$.

2. If we have $X \text{ — } Z \text{ — } Y$ but no edge between X and Y, and $Z \notin \mathbf{S}_{XY}$ then orient as $X \to Z \leftarrow Y$.

3. Apply the following rules:

   (a) If X and Z are not adjacent by $X \to Y \text{ — } Z$ then orient as $Y \to Z$.

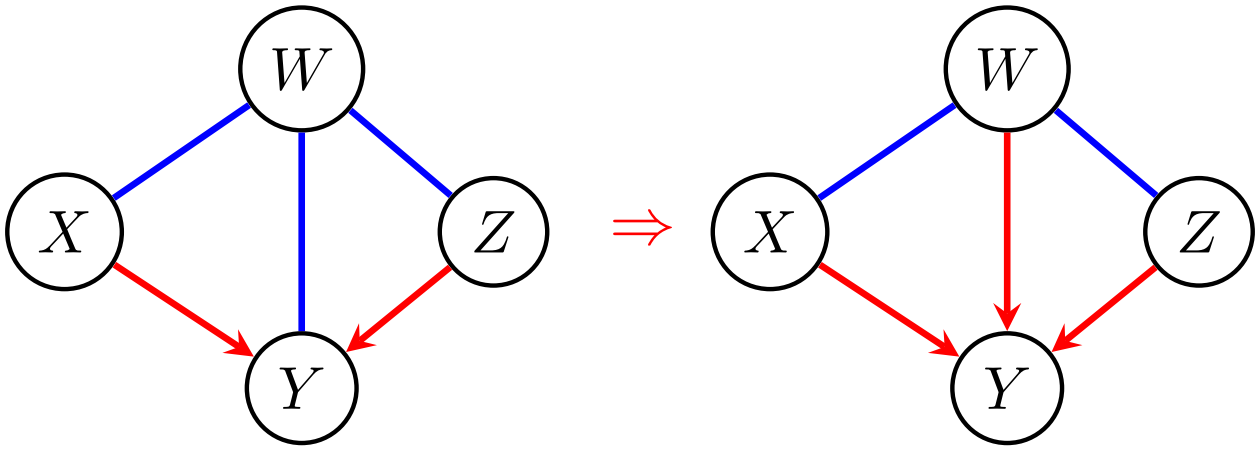
   (b) If $X \to Y \to Z$ and $X \text{ — } Z$ then orient as $X \to Z$.

   (c) If X and Z are not adjacent, but $X \text{ — } W \text{ — } Z$ and $X \to Y \leftarrow Z$ and $W \text{ — } Y$ then orient as $W \to Y$

Inferring the Markov equivalence class from the separating sets S_{XY} (contd.)

3. Apply the following rules:

   (c) If X and Z are not adjacent, but $X - W - Z$ and $X \to Y \leftarrow Z$ and $W - Y$ then orient as $W \to Y$ (*Why?*)



Simple strategy for structure learning

- For every pair of vertices (X, Y) search for a set $\mathbf{S}_{XY}$ such that:

  $X \perp\!\!\!\perp Y \mid \mathbf{S}_{XY}$

  $\Rightarrow$ Faithfulness implies X is d-separated from Y given $\mathbf{S}_{XY}$ in $\mathcal{G}$.

- Reconstruct the Markov equivalence class from the separating sets $\mathbf{S}_{XY}$

Problem: On p variables, for each pair (X, Y) there are potentially 2^{p-2} separating sets

$\Rightarrow$ Computationally intractable.

The PC Algorithm (I)

Addresses tractability via two ideas:

(I) If (X, Y) not adjacent then X and Y are d-separated either given pa(X) or pa(Y):

$\Rightarrow$ If all edges between a vertex T and X, and between T and Y have already been removed, then sets $\mathbf{S}^*$ s.t. $T \in \mathbf{S}^*$ are need not be considered in search for a separating set $\mathbf{S}_{XY}$.

$\Rightarrow$ Restrict search for separating sets to sets $\mathbf{S}$ such that either:

$\mathbf{S} \subseteq$ Adj(X) or $\mathbf{S} \subseteq$ Adj(Y);

here Adj(V) is the set of vertices in the graph that are still adjacent to V, after we have removed edges between vertices (X^*, Y^*) for which separating sets have already been found.

The PC Algorithm (II)

Addresses tractability via two ideas:

(II) Consider all potential separating sets of size k for every pair, before sets of size $k+1$.

Thus:

$k=0$ For each pair (X,Y), test $X \perp\!\!\!\perp Y \mid \emptyset$

(If the independence holds, remove the edge.)

$k=1$ For each pair (X,Y) adjacent, with $\max\{|\text{Adj}(X)|, |\text{Adj}(Y)|\} \geq 2$,
test $X \perp\!\!\!\perp Y \mid \mathbf{S}$, where $|\mathbf{S}| = 1$ and $\mathbf{S} \subseteq \text{Adj}(X)$ or $\mathbf{S} \subseteq \text{Adj}(Y)$

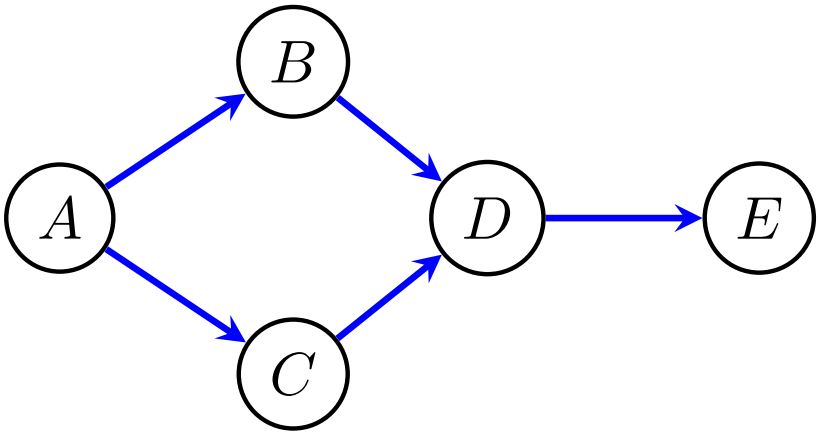
$\vdots \qquad \vdots \qquad \vdots$

k For each pair (X,Y) adjacent, with $\max\{|\text{Adj}(X)|, |\text{Adj}(Y)|\} \geq k+1$,
test $X \perp\!\!\!\perp Y \mid \mathbf{S}$, where $|\mathbf{S}| = k$ and $\mathbf{S} \subseteq \text{Adj}(X)$ or $\mathbf{S} \subseteq \text{Adj}(Y)$.

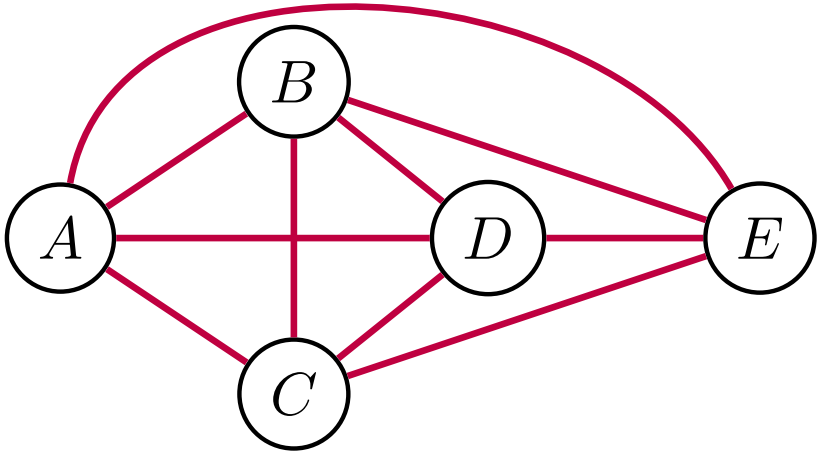
Stop when we reach a k such that for all (X,Y), $\max\{|\text{Adj}(X)|, |\text{Adj}(Y)|\} < k+1$.

Build the graph representing the Markov equivalence class from the separating sets $\mathbf{S}_{XY}$.

Example of PC Algorithm; Initialization

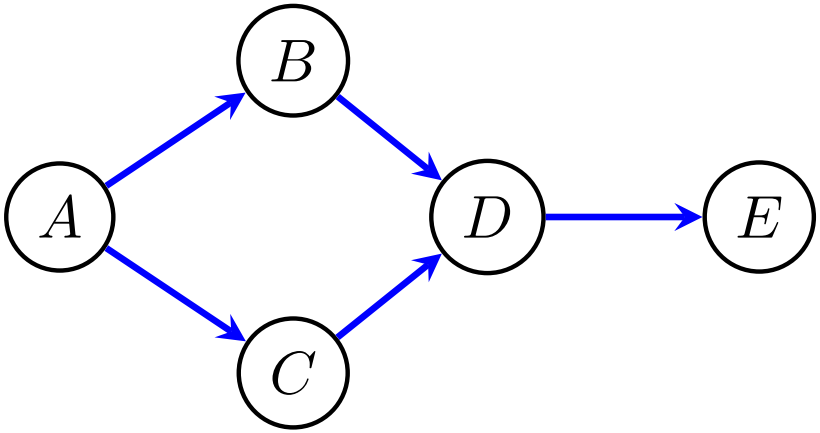


Unknown true graph

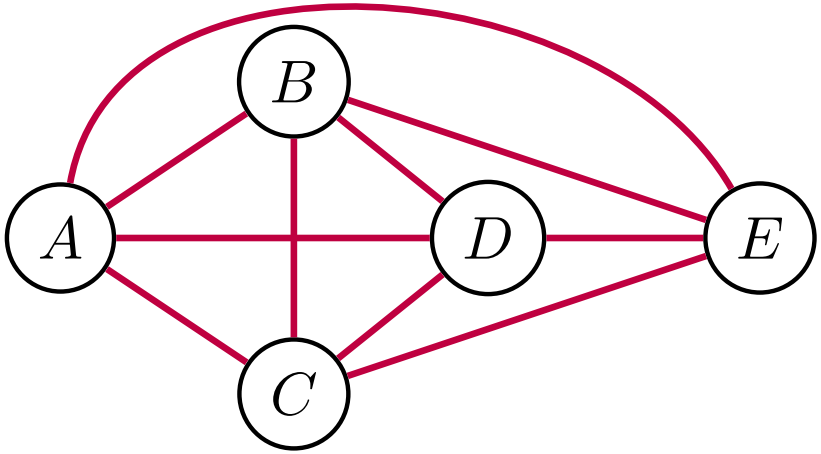


Initial Graph

Example of PC Algorithm; $k = 0$



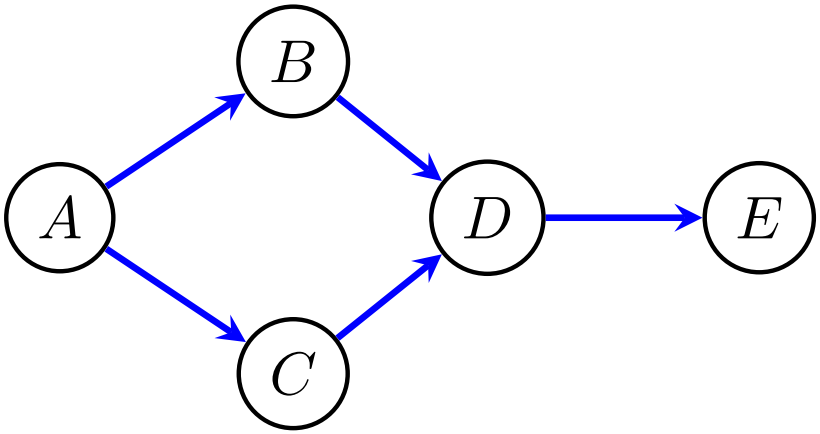
Unknown true graph



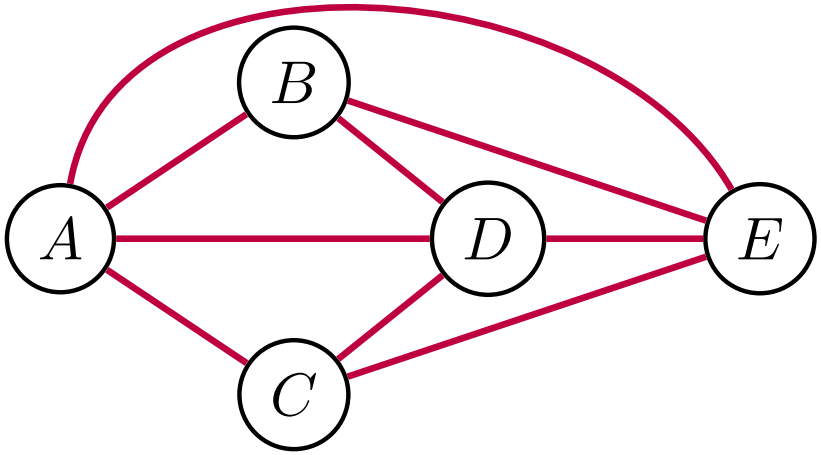
Graph after $k = 0$

There is no pair of variables d-separated given $\emptyset$, so Graph unchanged.

Example of PC Algorithm; $k = 1$ with (B, C)



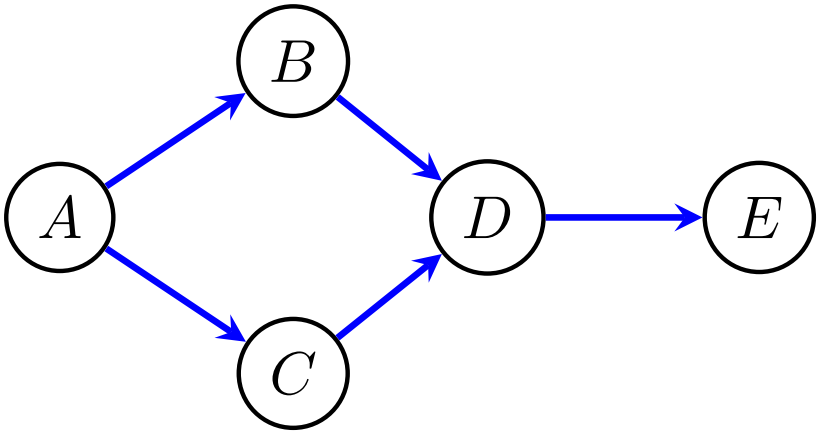
Unknown true graph



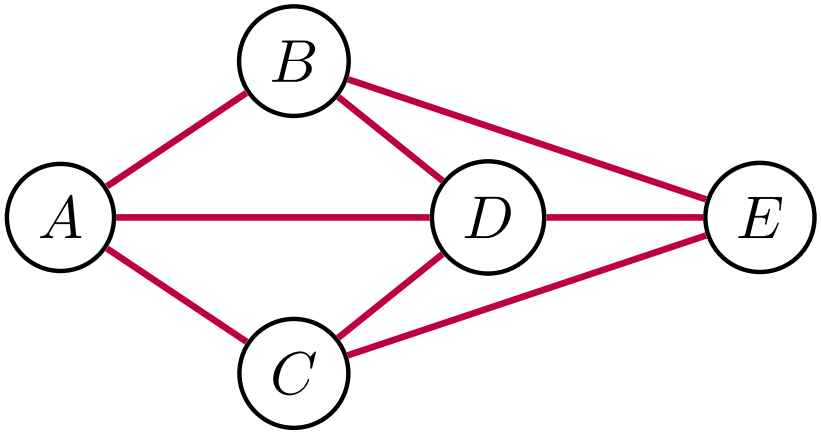
Graph

Since B and C are d-separated given $\{A\}$ we remove the B — C edge and record $S_{BC} = \{A\}$.

Example of PC Algorithm; $k = 1$ with (A, E)



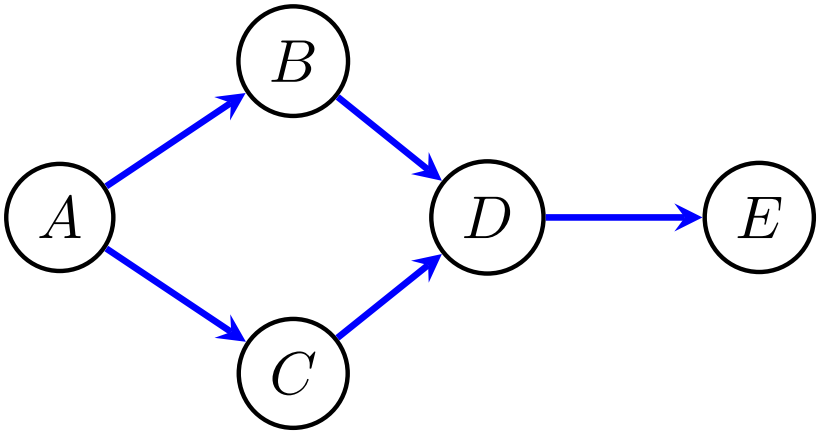
Unknown true graph



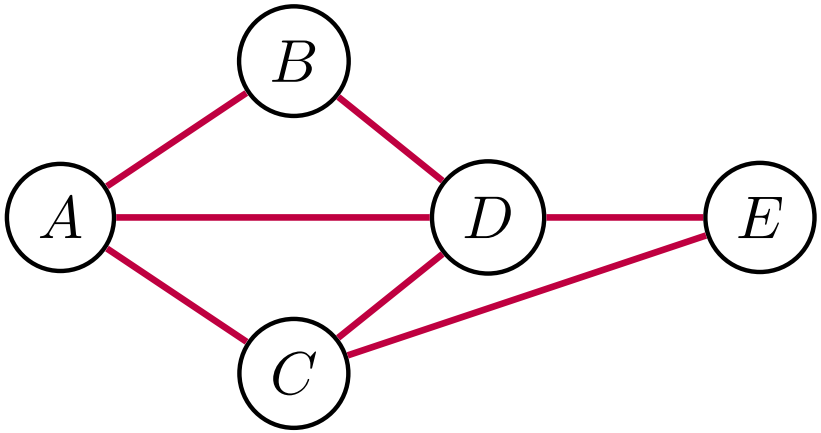
Graph

Since A and E are d-separated given $\{D\}$ we remove the A — E edge and record $S_{AE} = \{D\}$.

Example of PC Algorithm; $k = 1$ with (B, E)



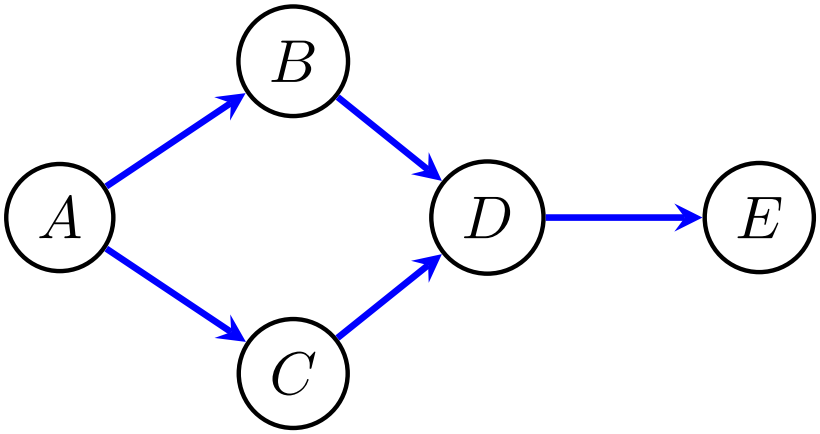
Unknown true graph



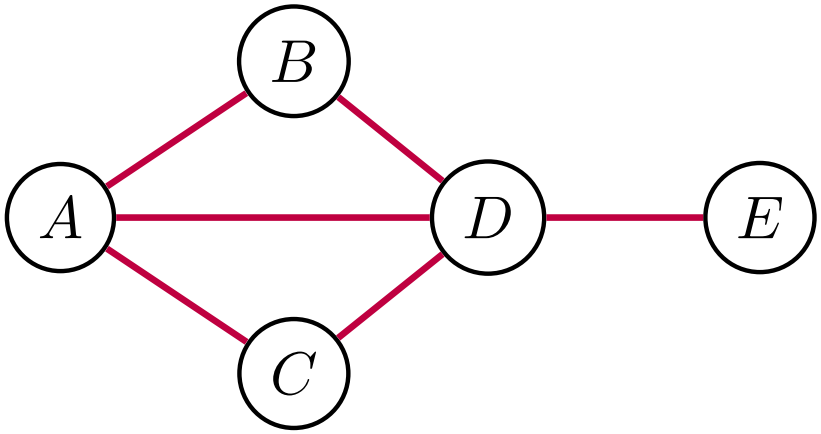
Graph

Since B and E are d-separated given $\{D\}$ we remove the B — E edge and record $S_{BE} = \{D\}$.

Example of PC Algorithm; $k = 1$ with (C, E)



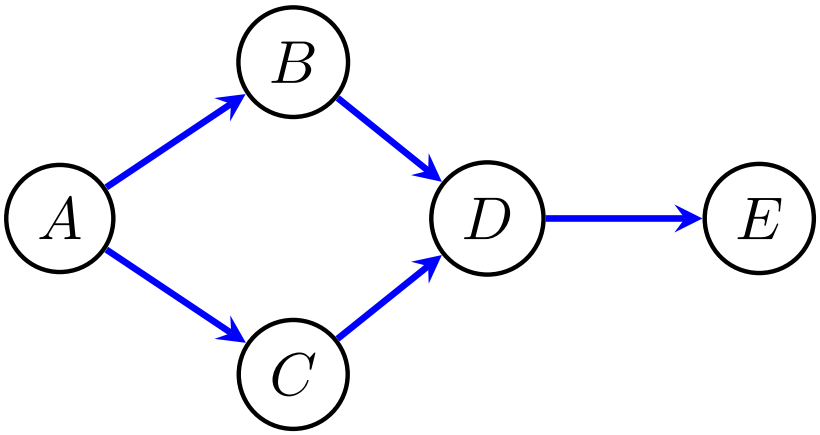
Unknown true graph



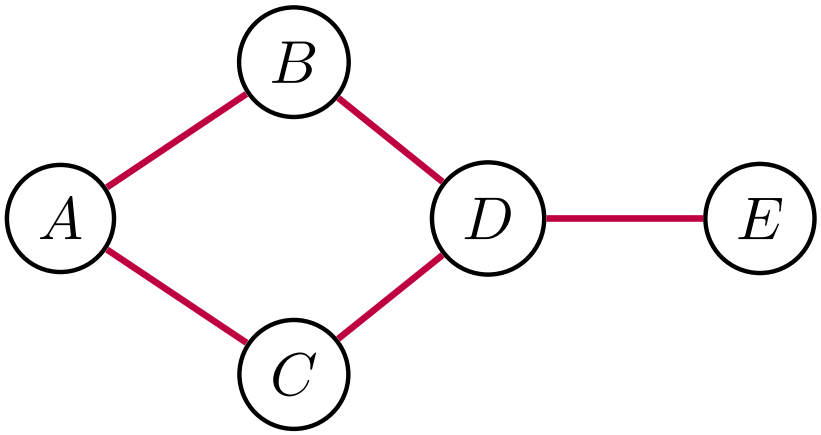
Graph after $k = 1$

Since C and E are d-separated given $\{D\}$ we remove the C — E edge and record $S_{CE} = \{D\}$. This completes this stage.

Example of PC Algorithm; $k = 2$ with (A, D)



Unknown true graph



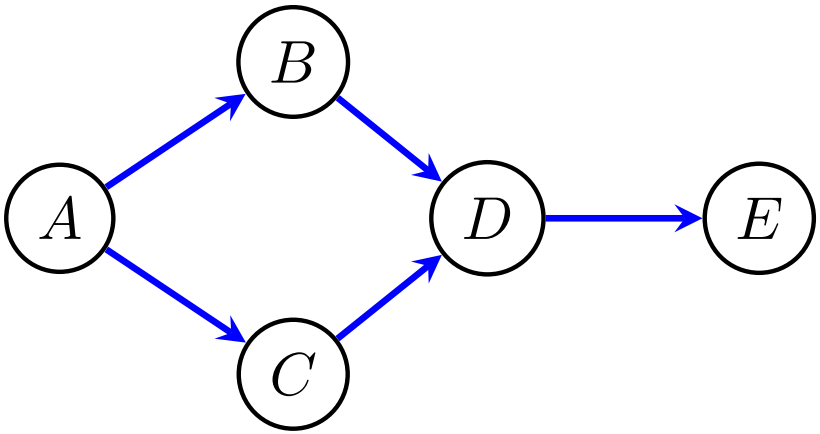
Graph after $k = 2$

Since A and D are d-separated given $\{B, C\}$ we remove the A — D edge and record $S_{AD} = \{B, C\}$.

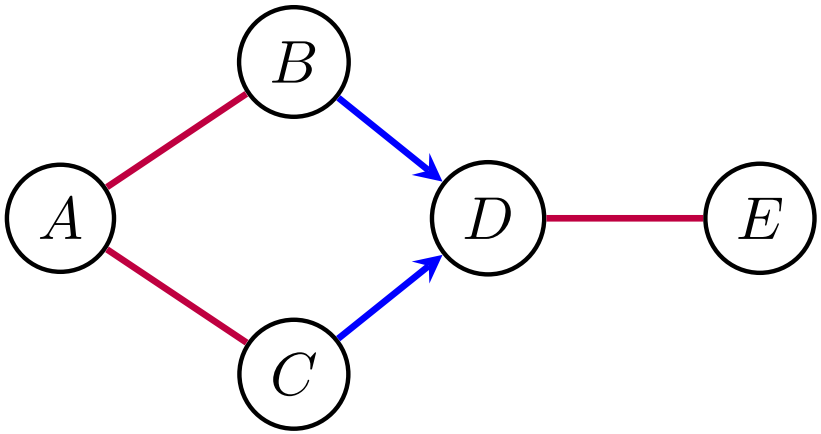
Note that the pairs (A, B) and (A, C) do not need to be considered here. *(Why?)*

This completes this stage; clearly there is no need to check $k = 3$.

Orienting unshielded colliders from separating sets



Unknown true graph

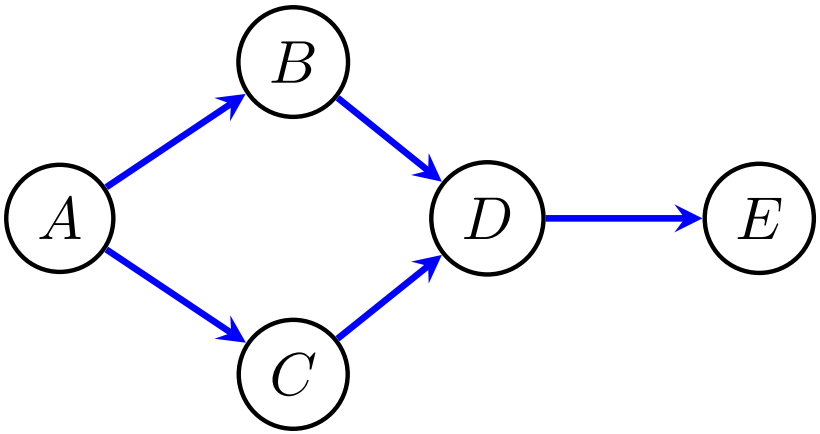


Graph after $k = 1$

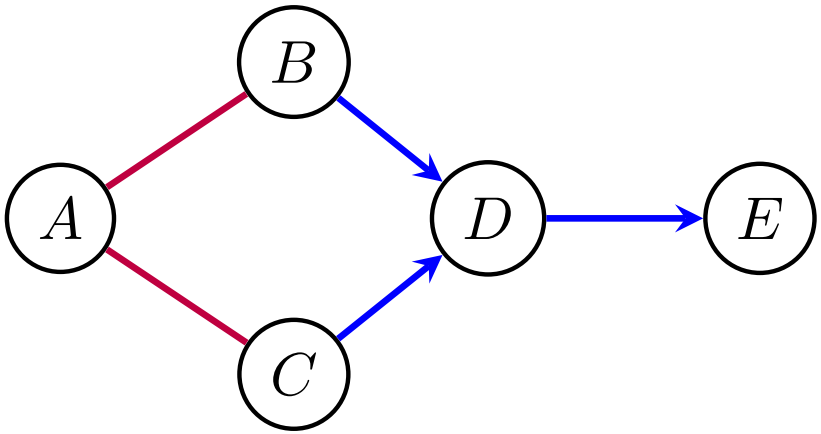
Since $D \notin S_{BC} = \{A\}$, we orient $B \rightarrow D \leftarrow C$.

The other triples (B, A, C), (A, B, D), (A, C, D), (B, D, E) and (C, D, E) do not lead to further orientation; since the 'middle' vertex is in each separating set.

Additional orientations to form CPDAG



Unknown true graph



CPDAG

Since (B, D, E) is not a collider, but $B \to D$ we can orient $D - E$ as $D \to E$.

Properties of the PC Algorithm

- Polynomial time on sparse graphs: complexity bounded by $O(p^q)$ where q is the max number of vertices adjacent to any vertex in $\mathcal{G}$.

Given a sample the algorithm performs hypothesis tests to judge conditional independence relations .

The resulting 'Sample PC Algorithm' is:

- Consistent for faithful distributions, but not uniformly consistent.
  (Sprites *et al*, 1993; Robins *et al*, 1999, 2003)

- Uniformly consistent under additional assumptions.
  (Kalisch & Buhlmann 2007)

  K& B consider a high-dimensional asymptotic regime where both $n, p \to \infty$. More formally, they consider a sequence of graphs $\mathcal{G}_n$ each with vertex set $\mathbf{V}_n$, and faithful distributions $P(\mathbf{V}_n)$.

Theorem [K& B, 2007] If the following four conditions hold:

(A1) The distribution P_n is multivariate Gaussian, faithful to a DAG $\mathcal{G}_n$.

(A2) $\mathcal{G}_n$ has p_n vertices where $p_n = O(n^a)$, with $0 \leq a < \infty$.

(A3) In $\mathcal{G}_n$, the graphs are sparse relative to sample size in that:

$$\max_{X \in \mathbf{V}_n} |\text{Adj}_{\mathcal{G}_n}(X)| = O(n^{1-b}) \quad \text{for some } 0 < b \leq 1.$$

(A4) Non-zero partial correlations $\rho_{n;X,Y|\mathbf{Z}}$ are bounded from below and above:

$$\inf \left\{ |\rho_{n;X,Y|\mathbf{Z}}| \;\middle|\; \{X,Y\} \cup \mathbf{Z} \subseteq \mathbf{V}_n \text{ with } \rho_{n;X,Y|\mathbf{Z}} \neq 0 \right\} \geq c_n,$$
$$\text{where } c_n^{-1} = O(n^d) \text{ for some } 0 < d < b/2 < 1/2$$

$$\sup_{X,Y,\mathbf{Z}} |\rho_{n;X,Y|\mathbf{Z}}| \leq M < 1,$$

Then there exists a sequence of significance levels α_n such that:

$$P[\hat{\mathcal{G}}^{CP}(\alpha_n) = \mathcal{G}_n^{CP}] \;=\; 1 - O(\exp(-Cn^{1-2d})) \to 1 \quad \text{as } n \to \infty$$

where $\mathcal{G}_n^{CP}$ and $\hat{\mathcal{G}}^{CP}(\alpha_n)$ are, respectively, the true CPDAG corresponding to $\mathcal{G}_n$ and the CPDAG resulting from the PC algorithm with sig. level α_n; C is a positive constant.

Simulation results (from K&B 2007)

p	n	$E[N]$	TPR	FPR
9	50	1.4	0.61 (0.03)	0.023 (0.005)
27	100	2.0	0.70 (0.02)	0.011 (0.001)
81	150	2.4	0.753 (0.007)	0.0065 (0.0003)
243	200	2.8	0.774 (0.004)	0.0040 (0.0001)
729	250	3.2	0.794 (0.004)	0.0022 (0.00004)
2187	300	3.5	0.805 (0.002)	0.0012 (0.00002)

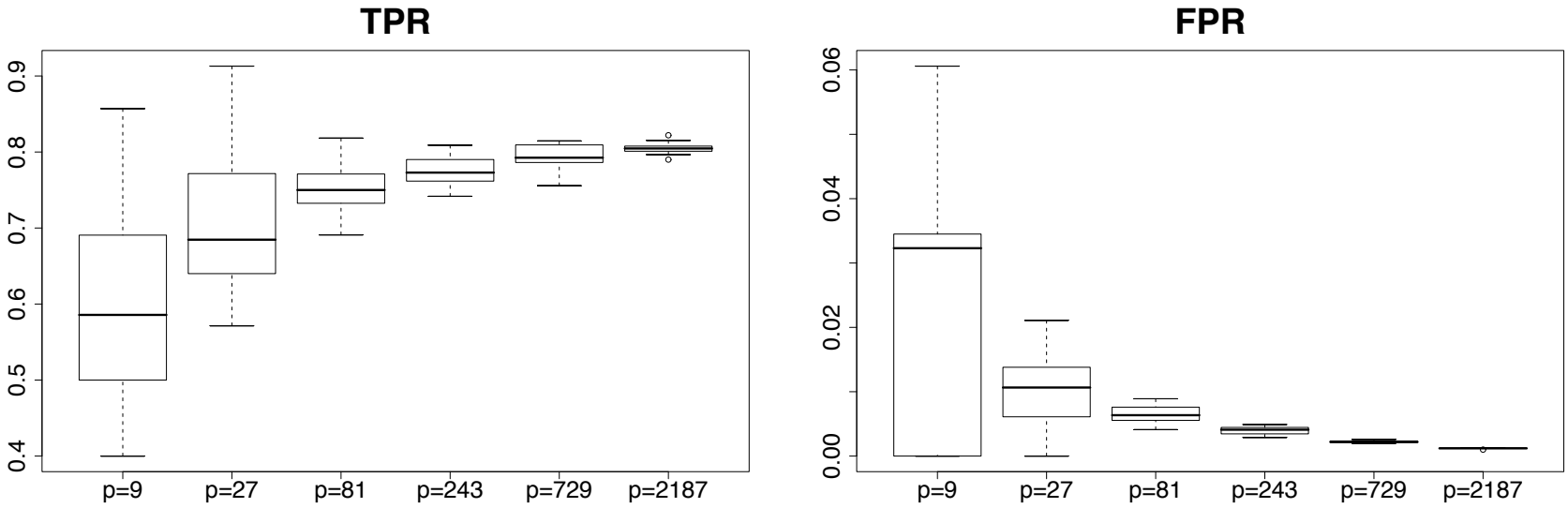
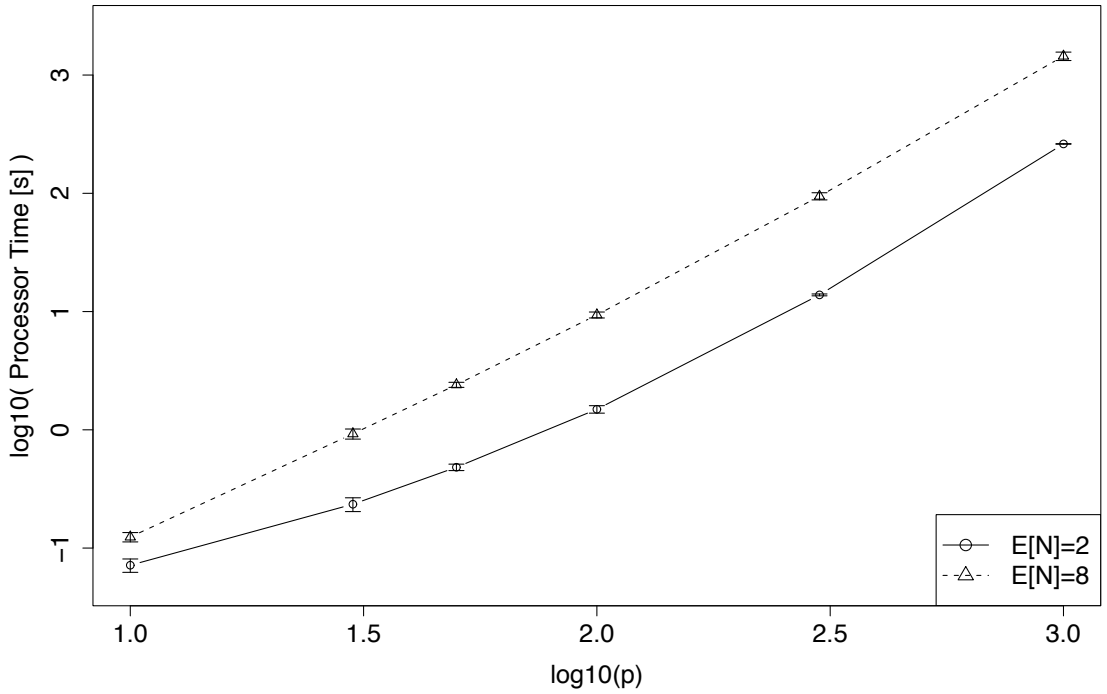


Figure 3: While the number of variables p increases exponentially, the sample size n increases linearly and the expected neighborhood size $E[N]$ increases sub-linearly, the TPR increases and the FPR decreases. See Table 1 for a more detailed specification of the parameters.

Run times in R (from K&B 2007)



$n = 1000$, $\alpha = 0.01$, p varied from 10 to 1000;

25 mins with $E[N] = 8$, $p = 1000$; about 1s for $p = 100$.

Machine specs: AMD Athlon 64 X2 Dual Core Processor 5000+ with 2.6GHz and 4GB RAM

running on Linux with R 2.4.1, `pcalg` package.

Building on PC: The IDA Algorithm

Maathuis *et al.* (2009) used CPDAGs produced by the PC algorithm, to try to find the parents of a given target variable (Y), and to judge the strength of effect.

Basic idea: consider each DAG compatible with the CPDAG given by PC;

if there is an edge $X \to Y$ in this DAG then estimate the coefficient $\hat{\beta}_{Y \leftarrow X}$ via regression, else set $\hat{\beta}_{Y \leftarrow X} = 0$.

For each variable X report the (multi-set) of coefficients resulting from this procedure.

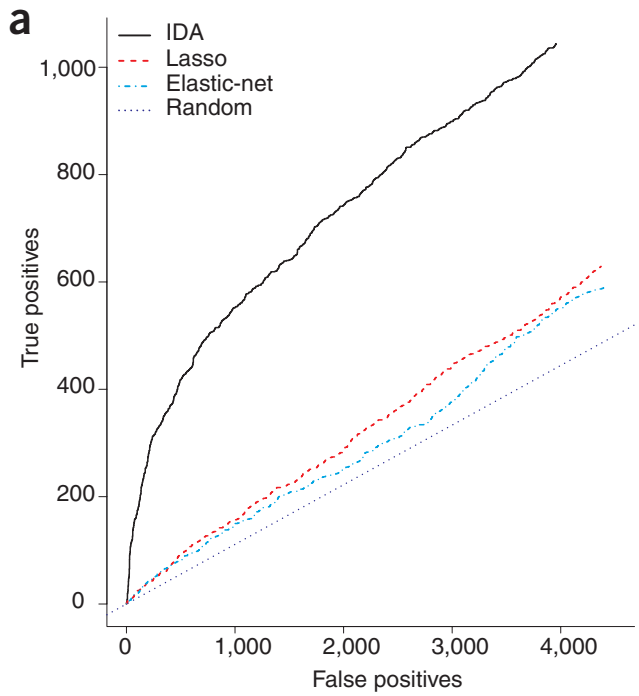
Use the resulting distribution to rank variables in importance.

Implicitly puts a uniform structure prior within equivalence classes.

(Note all variables are scaled to have the same variance.)

Experimental validation

Maathuis *et al.* (2010), *Nature Methods*, applied IDA to observational data on yeast gene expression with $n = 63, p = 5361$. They used data from 234 deletion experiments to determine effects on the other genes (giving 234×5360 effects); top 10% taken as Target Set. IDA was compared to Lasso and Elastic-Net:



Other approaches to learning DAGs

Many other algorithms have been presented for learning the structure of a CPDAG from data:

- Computation of marginal likelihoods under conjugate priors:
  - Multinomial likelihood - Dirichlet prior (Cooper & Herskovits,1992)
  - Normal likelihood - Inverse Wishart prior (Geiger & Heckerman 1994,1995)
- Searches based on BIC scores (Madigan and Raftery, 1994)
- Greedy Equivalence Search (Chickering 2002; Chickering & Meek 2002)

Search-based methods face challenges for large p due to the huge number of graphs to consider.

Let's get serious...

The PC Algorithm and all the methods described so far assume every variable in the underlying DAG is observed. *Really ?!*

Structure Learning – the hard case

when hidden variables may be present

Structure Learning: Not fully observed

- Q: Given a distribution P that factorizes according to a unknown DAG $\mathcal{G}$, that satisfies the faithfulness assumption, what can be inferred regarded the structure of $\mathcal{G}$?
- Simple case: Fully observed
- Hard case: With hidden variables

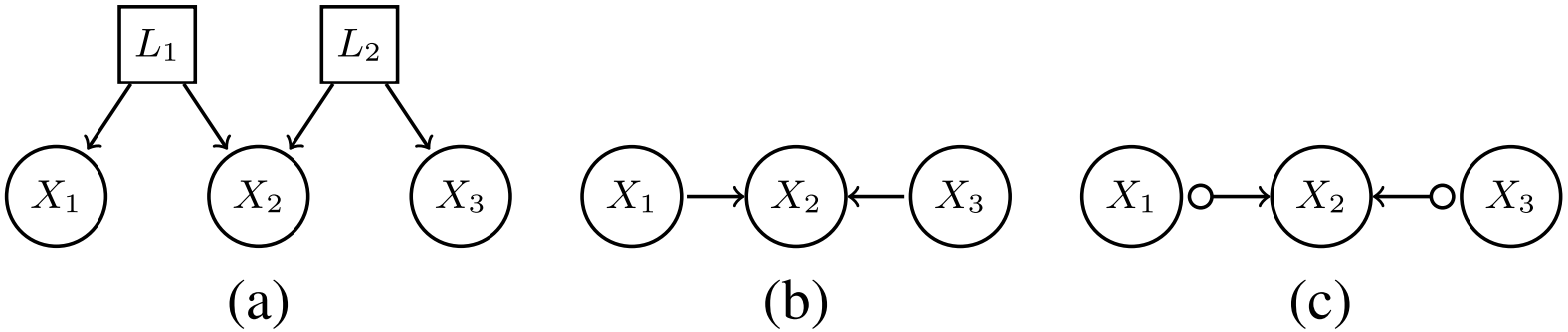
  $\mathcal{G}$ has vertex set $O \cup H$; we are given the marginal distribution $P(X_O)$.

  We make no assumptions regarding the number of H variables, nor their state spaces. (This rules out e.g. applying the EM algorithm.)

Simplest idea

Work out which DAGs with hidden variables are compatible with independence relations in the observed margin.

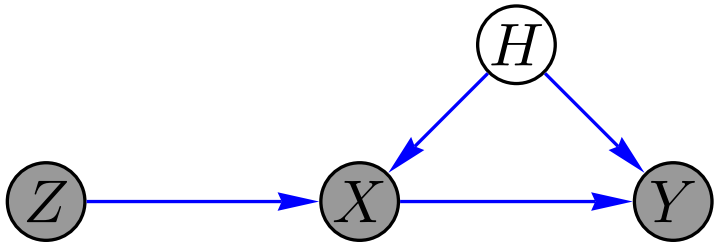
Example: (a) the true graph, L_1, L_2 unobserved, this implies $X_1 \perp\!\!\!\perp X_3$; (b) another graph also implying this; (c) A 'Partial Ancestral Graph' (PAG) representing an equivalence class. (Figure from Colombo *et al.* (2012))



Even for this 'simple' strategy, there turn out to be many subtle and important differences that arise when hidden confounders may be present.

Difference (I)

There can be observed variables which are not adjacent, but for which no subset of the other observed variables d-separates them.



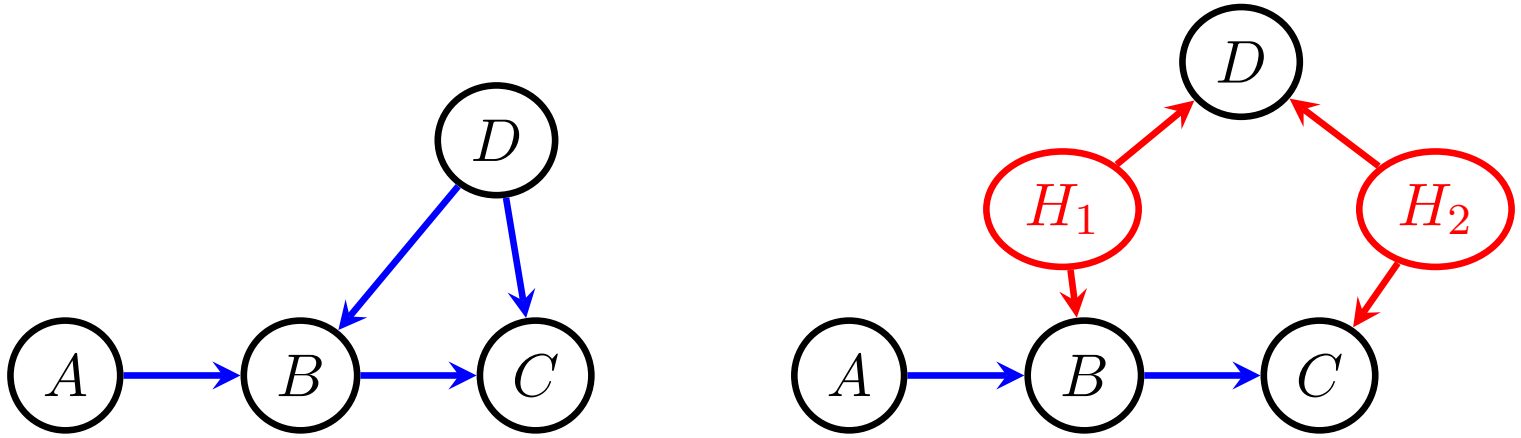
Here H is unobserved.

Z and Y are d-connected given $\emptyset$.

Z and Y are d-connected given $\{X\}$.

Difference (II)

For DAGs without hidden variables, *same adjacencies* and *same unshielded colliders* were necessary and sufficient for equivalence $\Rightarrow$ only need to look at structures involving at most 3 vertices.



These graphs are not Markov equivalent over the observed margin.

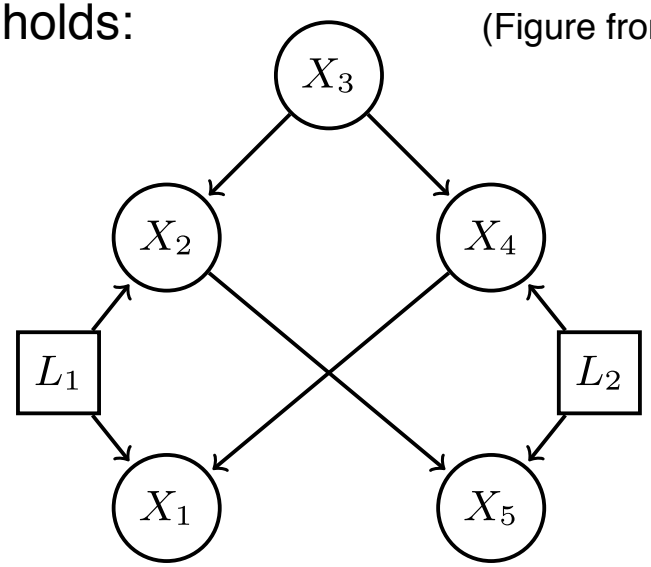
A is d-separated from C given $\{B, D\}$ in the left graph.
A is d-separated from C given $\{B\}$ in the right graph.

$\Rightarrow$ Need to look at more complex structures.

Difference (III)

For DAGs without hidden variables, the search for separating sets for (X, Y) could be restricted to sets $\mathbf{S}$ such that either given Adj(X) or Adj(Y).This no longer holds:

(Figure from Colombo *et al.* (2012))



X_1 and X_5 are d-separated given $\{X_2, X_3, X_4\}$ but no other set; X_3 is not adjacent to X_1 and not adjacent to X_5.

$\Rightarrow$ Major implication: search for sep. sets is not poly-time for sparse graphs.

Summary of results

Results on structure learning DAGs wtih hidden variables from conditional independence:

- Spirtes *et al.* (1993) formulated the FCI algorithm; new interpretation of output by Spirtes, Meek + R (1999).

  Not polynomial time even for sparse graphs.

- Spirtes (2001), publishes an 'anytime' version of FCI (stop at any value of k).

- R+Spirtes (2002) formulate a likelihood for Gaussian data; R (2009), R + Evans (2012) formulated a likelihood for discrete data.

- Ali+R+Spirtes (2009) characterized Markov equivalence, see also Zhao, Zheng & Liu (2005).

- Zhang (2008) developed the analogous notion to the CPDAG construction from separating sets.

RFCI Algorithm

Colombo, Kalisch, Maathuis + R (2012) formulate an FCI-variant that is polynomial-time on sparse graphs.

This paper also gives assumptions under which FCI (and RFCI) are uniformly consistent.

RFCI is potentially less informative than FCI, though in practice there are few differences.

RFCI Run times

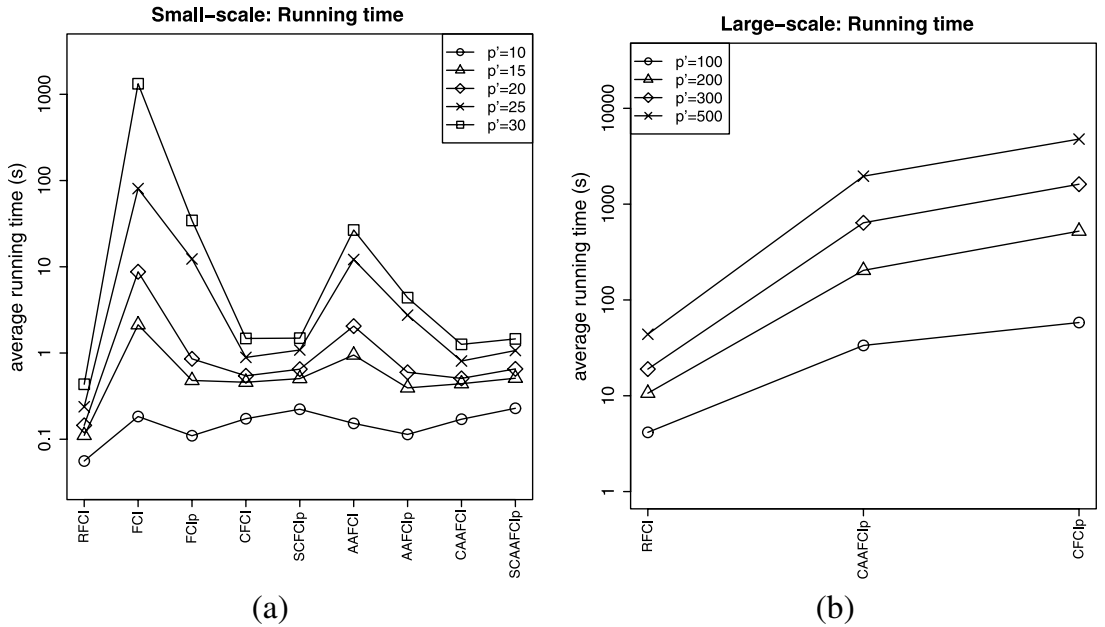


FIG. 7. *Running time of the sample versions of the algorithms, using simulation settings* $n = 1000$ *and* $\alpha = 0.01$, *where the y-axes are drawn in log scale.* (a) *Average running time in seconds of each algorithm over* 50 *replicates, using* $E(N) = 2$; (b) *average running time in seconds of each algorithm over* 91 *replicates* (*see text*), *using* $E(N) = 3$.

(Graph from Colombo, Kalisch, Maathuis, and Richardson (2012))

Yet another difference

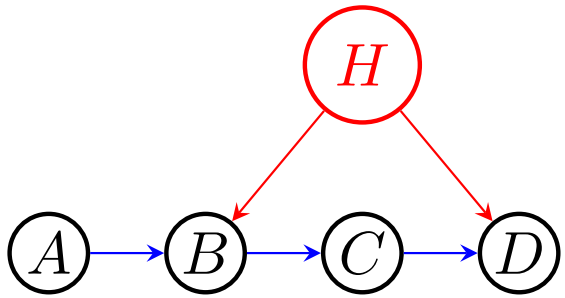
In the fully observed case, two DAGs were Markov equivalent iff they represented the same set of distributions.

$\Rightarrow$ No information is lost by basing structure learning on conditional independence constraints.

But when hidden variables may be present this is no longer true.

$\Rightarrow$ There are other non-parametric constraints, that do not correspond to conditional independence that may be used for structure learning.

'Verma' constraints



This graph implies $A \perp\!\!\!\perp C \mid B$ but also:

$A \perp\!\!\!\perp D \mid C$ in $P^*(a, b, d \mid c) \equiv P(a, b, c, d)/P(c \mid b, a)$.

Constraints of this type were studied by Verma & Pearl (1992), Tian & Pearl (2002) and Shpitser & Pearl (2008)

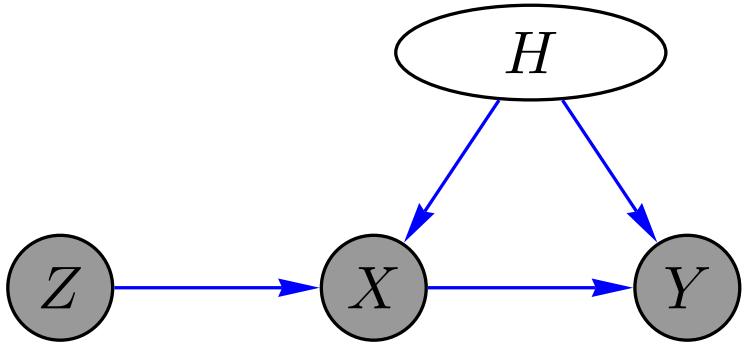
Robins (1999) saw that in conjunction with an extended faithfulness assumption these constraints could be used for structure learning.

R+ Shpitser + Robins (2012) have developed a 'nested Markov property' to capture these constraints via simple graphical operation. This leads to a likelihood, fitting procedures, BIC-based search (see Ilya's Poster!)

Inequality Constraints

There are also inequality constraints implied by hidden variable models.

Pearl's Instrumental Variable (IV) inequalities:



The binary IV model is defined by the inequalities:

$$
\begin{aligned}
p(y=0, x=0 \mid z=0) + p(y=1, x=0 \mid z=1) &\leq 1, \\
p(y=0, x=0 \mid z=1) + p(y=1, x=0 \mid z=0) &\leq 1, \\
p(y=0, x=1 \mid z=0) + p(y=1, x=1 \mid z=1) &\leq 1, \\
p(y=0, x=1 \mid z=1) + p(y=1, x=1 \mid z=0) &\leq 1,
\end{aligned}
$$

Evans (2012) studies the inequalities arising from DAGs with hidden variables.

Past is prologue....

Performing structure learning from DAGs with hidden variables using only conditional independence information serves as a good ‘warm up’ for structural inference using all constraints that are available.

Postscript: In what sense are these DAGs 'causal'?

Though our structure learning algorithm could be presented as solving a problem in multivariate Statistics, the motivation is clearly causal.

WIth some exceptions most statistical causal models are formulated via the potential outcome framework of Neyman (1923).

So. . . where are the potential outcomes?

Dawid (2000): causal interpretation of DAGs without potential outcomes, taking intervention distributions as primitives.

Potential outcomes for DAGs

Consider the simple case in which every variable may be intervened on.

We have a potential outcome $Y(\text{pa}(y))$ for every variable Y and assignment of values to the parents of Y.

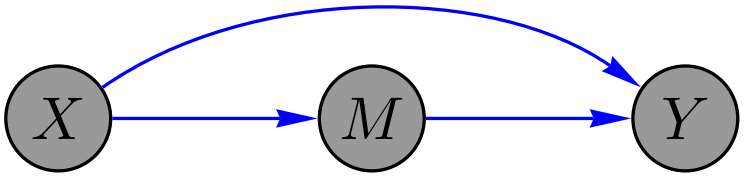
A causal DAG model may be thought of as asserting that:

$$P(Y = y^* \mid \text{pa}(y)) = P(Y(\text{pa}(y)) = y^*).$$

Such constraints are implied by the FFRCISTG model of Robins (1986).

(May also restrict interventions to subsets of variables, though meaning of directed edges is less clear.)

NPSEM approach of Pearl



$$\begin{aligned} X &= f_X(\epsilon_X) \\ M(x) &= f_M(x, \epsilon_M) \\ Y(x, m) &= f_Y(x, m, \epsilon_Y). \end{aligned}$$

Pearl assumes independence of the errors:

$$\epsilon_X \perp\!\!\!\perp \epsilon_M \perp\!\!\!\perp \epsilon_Y.$$

This is a much stronger assumption, that cannot be tested via experiment.

(Leads to identification of PDE and PSDE.)

Thank you!