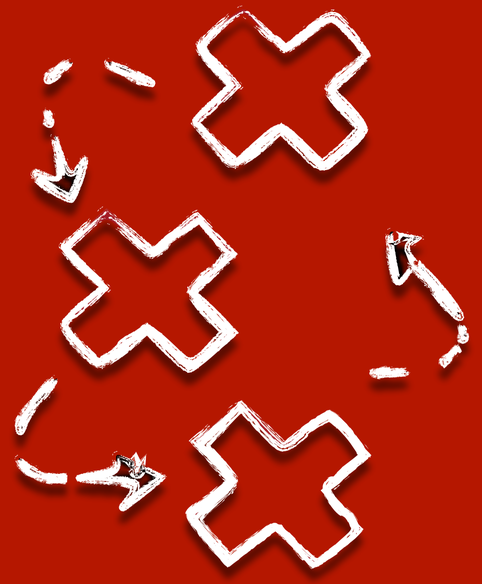


Causal Data Science

Lecture 9.1 Constraint-based causal discovery

Lecturer: Sara Magliacane

UvA - Spring 2024

Overview on where we are in the course

1	Introduction
2	Probability recap
3	Causal graphs
4	Interventions
5	Covariate adjustment
6	Frontdoor criterion, Instrumental variables
7	Counterfactuals and potential outcomes
8	Estimating causal effects, Missing data
9	Constraint based structure learning
10	Score based structure learning
11	Advanced structure learning and transportability
12	Causality-inspired ML

Background on causal graphs

We know the causal graph, how do we estimate causal effects?

What happens if the graph is unknown?

Causal discovery simplified overview

Constraint-based causal discovery

- Conditional independence tests
- Observational data
- Output: MEC
- SGS, PC, FCI

Score-based causal discovery

- Penalised likelihood
- Observational data
- Output: MEC
- GES, MMHC

Restricted models

- Nonlinear additive noise, Linear Non-Gaussianity
- Observational data
- Output: DAG
- RESIT, LINGAM

Interventional causal discovery / causal invariance

- Observational and Interventional data
- Output: parents of Y, I-MEC
- ICP, GIES, JCI

Causal discovery simplified overview

Constraint-based causal discovery

- Conditional independence tests
- Observational data
- Output: MEC
- SGS, PC

Score-based causal discovery

- Penalised likelihood
- Observational data
- Output: MEC
- GES, MMHC

Restricted models

- Nonlinear additive noise, Linear Non-Gaussianity
- Observational data
- Output: DAG
- RESIT, LINGAM

Interventional causal discovery / causal invariance

- Observational and Interventional data
- Output: parents of Y, I-MEC
- ICP, GIES, JCI

Perfect maps - definition

- If P is Markov and faithful to G, we say that **G is a perfect map of P.**

  Then, for any disjoint $\mathbf{A}, \mathbf{B}, \mathbf{C} \subseteq \mathbf{V}$:

$$\mathbf{A} \perp_G \mathbf{B} \mid \mathbf{C} \iff X_{\mathbf{A}} \perp\!\!\!\perp_P X_{\mathbf{B}} \mid X_{\mathbf{C}}$$

Perfect maps - definition

- If P is Markov and faithful to G, we say that **G is a perfect map of P.**

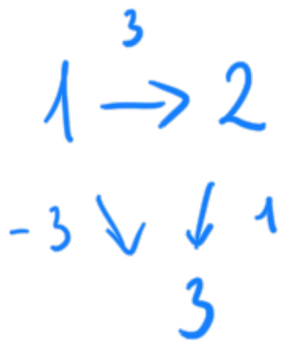
  Then, for any disjoint $\mathbf{A}, \mathbf{B}, \mathbf{C} \subseteq \mathbf{V}$:

$$\mathbf{A} \perp_G \mathbf{B} \,|\, \mathbf{C} \iff X_{\mathbf{A}} \perp\!\!\!\perp_P X_{\mathbf{B}} \,|\, X_{\mathbf{C}}$$

- In some special cases, the ground truth causal graph is not a perfect map of the distribution P
- We will now see an example of a faithfulness violation SCM, in which there are more conditional independences than what we would expect from the graph based on d-separations

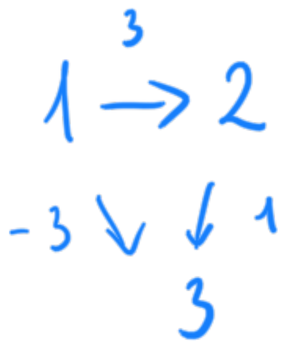
Faithfulness violations example - cancelling paths

$$
\begin{cases}
X_1 = \epsilon_1 \\
X_2 = 3X_1 + \epsilon_2 \\
X_3 = X_2 - 3X_1 + \epsilon_3 \\
\epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\
\epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3
\end{cases}
$$



Faithfulness violations example - cancelling paths

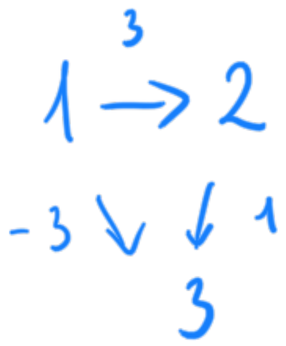
$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3X_1 + \epsilon_2 \\ X_3 = X_2 - 3X_1 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$



$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3\epsilon_1 + \epsilon_2 \\ X_3 = 3\epsilon_1 + \epsilon_2 - 3\epsilon_1 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

Faithfulness violations example - cancelling paths

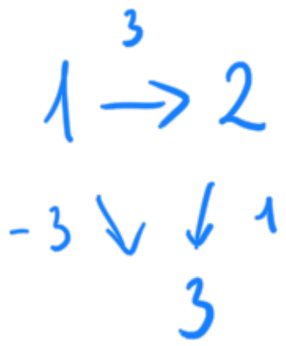
$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3X_1 + \epsilon_2 \\ X_3 = X_2 - 3X_1 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$



$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3\epsilon_1 + \epsilon_2 \\ X_3 = \cancel{3\epsilon_1} + \epsilon_2 - \cancel{3\epsilon_1} + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

Faithfulness violations example - cancelling paths

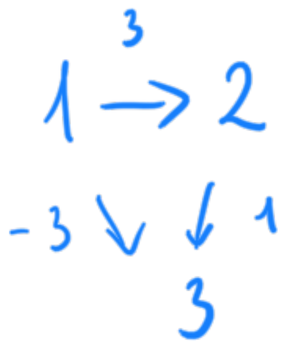
$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3X_1 + \epsilon_2 \\ X_3 = X_2 - 3X_1 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$



$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3\epsilon_1 + \epsilon_2 \\ X_3 = \epsilon_2 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

Faithfulness violations example - cancelling paths

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3X_1 + \epsilon_2 \\ X_3 = X_2 - 3X_1 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

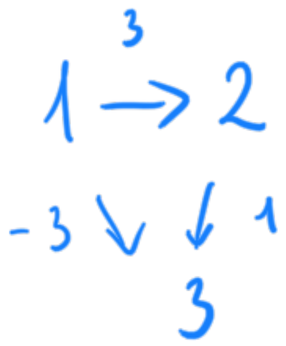


$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3\epsilon_1 + \epsilon_2 \\ X_3 = \epsilon_2 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, , \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$$X_1 \not\perp\!\!\!\perp X_2$$

Faithfulness violations example - cancelling paths

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3X_1 + \epsilon_2 \\ X_3 = X_2 - 3X_1 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$



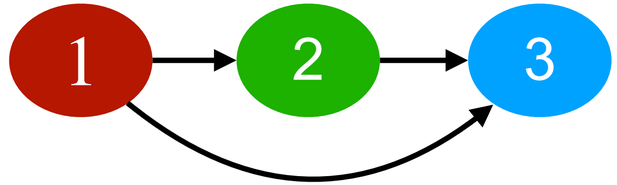
$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3\epsilon_1 + \epsilon_2 \\ X_3 = \epsilon_2 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, , \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$X_1 \not\perp\!\!\!\perp X_2$ $X_2 \not\perp\!\!\!\perp X_3$ $X_1 \perp\!\!\!\perp X_3$

Faithfulness violations example - cancelling paths

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = 3X_1 + \epsilon_2 \\ X_3 = X_2 - 3X_1 + \epsilon_3 \\ \epsilon_1, \epsilon_2, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_2, \epsilon_1 \perp\!\!\!\perp \epsilon_3, \epsilon_2 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$X_1 \not\perp\!\!\!\perp X_2 \quad X_2 \not\perp\!\!\!\perp X_3 \quad X_1 \perp\!\!\!\perp X_3$



$X_1 \not\perp\!\!\!\perp_d X_2 \quad X_2 \not\perp\!\!\!\perp_d X_3 \quad \boxed{X_1 \not\perp\!\!\!\perp_d X_3}$

The conditional independences don't fit the true causal graph, but they fit another one.

Perfect maps - non-existence

- Not every distribution p has a perfect map

$$
\begin{cases}
X_1 = \epsilon_1 \\
X_2 = X_1 \\
X_3 = X_2 + \epsilon_3 \\
\epsilon_1, \epsilon_3 \sim N(0,1) \\
\epsilon_1 \perp\!\!\!\perp \epsilon_3
\end{cases}
$$

Deterministic function

Perfect maps - existence

- Not every distribution p has a perfect map

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = X_1 \\ X_3 = X_2 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

Deterministic function

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = \epsilon_1 \\ X_3 = \epsilon_1 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$X_1 \not\perp\!\!\!\perp X_2$

$X_2 \not\perp\!\!\!\perp X_3$ $X_1 \not\perp\!\!\!\perp X_3$

$X_1 \perp\!\!\!\perp X_3 \mid X_2$

Perfect maps - existence

- Not every distribution p has a perfect map

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = X_1 \\ X_3 = X_2 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

Deterministic function

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = \epsilon_1 \\ X_3 = \epsilon_1 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$$X_1 \not\perp\!\!\!\perp X_2$$

$$X_2 \not\perp\!\!\!\perp X_3$$

$$X_1 \not\perp\!\!\!\perp X_3$$

$$X_1 \perp\!\!\!\perp X_3 \mid X_2$$

$$p(\epsilon_1 \mid \epsilon_1, \epsilon_1 + \epsilon_3) = p(\epsilon_1 \mid \epsilon_1)$$

$$X_2 \perp\!\!\!\perp X_3 \mid X_1$$

Perfect maps - existence

- Not every distribution p has a perfect map

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = X_1 \\ X_3 = X_2 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

Deterministic function

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = \epsilon_1 \\ X_3 = \epsilon_1 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$X_1 \not\perp\!\!\!\perp X_2$

$X_2 \not\perp\!\!\!\perp X_3$

$X_1 \not\perp\!\!\!\perp X_3$

$X_1 \perp\!\!\!\perp X_3 \mid X_2$

$X_2 \perp_d X_3 \mid X_1$?

$X_2 \perp\!\!\!\perp X_3 \mid X_1$

Perfect maps - existence

- Not every distribution p has a perfect map

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = X_1 \\ X_3 = X_2 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

Deterministic function

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = \epsilon_1 \\ X_3 = \epsilon_1 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$X_1 \not\perp\!\!\!\perp X_2$

$X_2 \not\perp\!\!\!\perp X_3$

$X_1 \not\perp\!\!\!\perp X_3$

$X_1 \perp\!\!\!\perp X_3 \mid X_2$

$X_2 \not\perp_d X_3 \mid X_1$

$X_2 \perp\!\!\!\perp X_3 \mid X_1$

Perfect maps - existence

- Not every distribution p has a perfect map

$$\begin{cases} X_1 = \epsilon_1 \\ X_2 = X_1 \\ X_3 = X_2 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases} \qquad \begin{cases} X_1 = \epsilon_1 \\ X_2 = \epsilon_1 \\ X_3 = \epsilon_1 + \epsilon_3 \\ \epsilon_1, \epsilon_3 \sim N(0,1) \\ \epsilon_1 \perp\!\!\!\perp \epsilon_3 \end{cases}$$

$X_1 \not\perp\!\!\!\perp X_2$

$X_2 \not\perp\!\!\!\perp X_3$

$X_1 \not\perp\!\!\!\perp X_3$

$X_1 \perp\!\!\!\perp X_3 \mid X_2$

$X_2 \perp\!\!\!\perp X_3 \mid X_1$

There is a (d-connecting) path between X_1 and X_3

X_1 and X_3 are not adjacent

X_2 and X_3 are not adjacent

This is a contradiction, there exists no Bayesian network that can represent these conditional in/dependences as d-separations perfectly!

Perfect maps - definition

- If P is Markov and faithful to G, we say that **G is a perfect map of P.**

  Then, for any disjoint $\mathbf{A}, \mathbf{B}, \mathbf{C} \subseteq \mathbf{V}$:

$$\mathbf{A} \perp_G \mathbf{B} \mid \mathbf{C} \iff X_{\mathbf{A}} \perp\!\!\!\perp_P X_{\mathbf{B}} \mid X_{\mathbf{C}}$$

- In some special cases, the ground truth causal graph is not a perfect map of the distrib
- We will n there are more the graph based on d-separations

For the rest of the course, we will assume the true causal graph is a perfect map of the distribution, i.e. it is Markov and faithful to P.

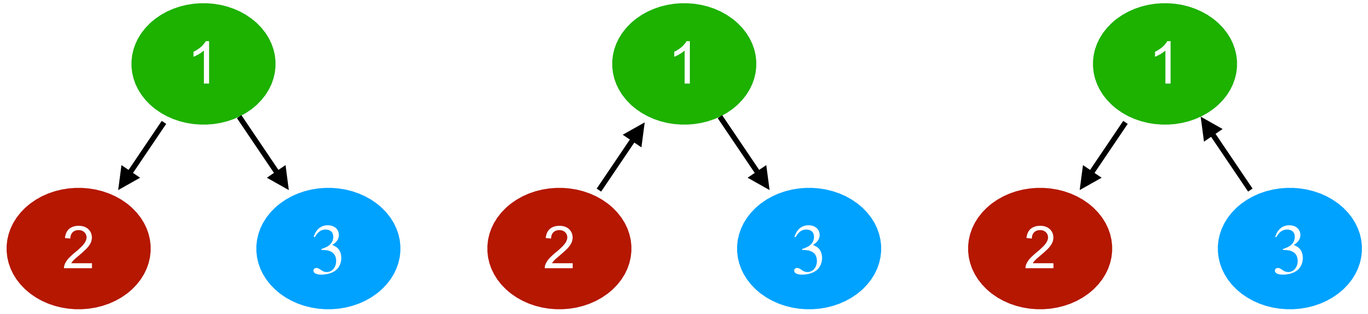
Perfect maps - Markov equivalence

- If P is Markov and faithful to G, we say that **G is a perfect map of P.**

  Then, for any disjoint $\mathbf{A}, \mathbf{B}, \mathbf{C} \subseteq \mathbf{V}$:

$$\mathbf{A} \perp_G \mathbf{B} \,|\, \mathbf{C} \iff X_{\mathbf{A}} \perp\!\!\!\perp_P X_{\mathbf{B}} \,|\, X_{\mathbf{C}}$$

- We call all DAGs with the same d-separations **Markov equivalent**
- Their set is the **Markov equivalence class (MEC)**



$$X_1 \not\perp\!\!\!\perp X_2$$
$$X_1 \not\perp\!\!\!\perp X_3$$
$$X_2 \not\perp\!\!\!\perp X_3$$
$$X_1 \not\perp\!\!\!\perp X_2 \,|\, X_3$$
$$X_1 \not\perp\!\!\!\perp X_3 \,|\, X_2$$
$$X_2 \perp X_3 \,|\, X_1$$

Markov equivalence example

$X \not\perp Y$ $X \perp Y \mid Z$

Which graphs on X, Y, Z are perfect maps of these conditional independences?

Hint: we can start by orienting this undirected graph

Markov equivalence example

$X \not\perp Y$ $X \perp\!\!\!\perp Y \mid Z$

Which graphs on X, Y, Z are perfect maps of these conditional independences?

$X - Z - Y$

$X \to Z \to Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

Markov equivalence example

$X \not\perp Y \quad X \perp\!\!\!\perp Y \mid Z$

Which graphs on X, Y, Z are perfect maps of these conditional independences?

$X - Z - Y$

$X \to Z \to Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

$X \leftarrow Z \leftarrow Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

Markov equivalence example

$X \not\perp Y \quad X \perp\!\!\!\perp Y \mid Z$

Which graphs on X, Y, Z are perfect maps of these conditional independences?

$X - Z - Y$

$X \to Z \to Y$ $\quad X \not\perp_d Y$ ✓ $\quad X \perp_d Y \mid Z$ ✓

$X \leftarrow Z \to Y$ $\quad X \not\perp_d Y$ ✓ $\quad X \perp_d Y \mid Z$ ✓

$X \leftarrow Z \leftarrow Y$ $\quad X \not\perp_d Y$ ✓ $\quad X \perp_d Y \mid Z$ ✓

These three graphs represent a Markov Equivalence Class

Markov equivalence example

$X \not\perp Y \quad X \perp\!\!\!\perp Y \mid Z$

Which graphs on X, Y, Z are perfect maps of these conditional independences?

$X - Z - Y$

$X \to Z \to Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

$X \leftarrow Z \to Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

$X \leftarrow Z \leftarrow Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

$X \to Z \leftarrow Y$

$X \perp_d Y$ ×

$X \not\perp_d Y \mid Z$ ×

Markov equivalence example

$X \not\perp Y \quad X \perp\!\!\!\perp Y \mid Z$

Which graphs on X, Y, Z are perfect maps of these conditional independences?

$X - Z - Y$

$X \to Z \to Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

$X \leftarrow Z \to Y$

$X \not\perp_d Y$ ✓

$X \perp_d Y \mid Z$ ✓

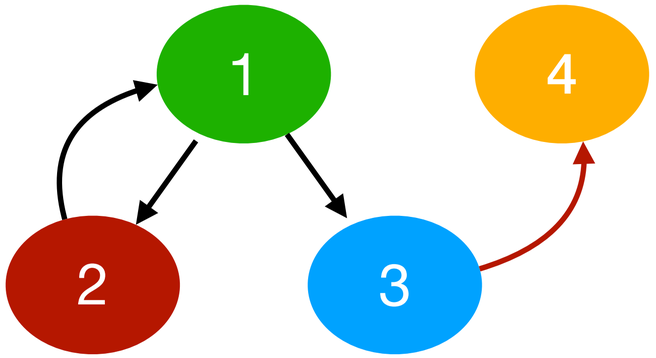
$X \leftarrow Z \leftarrow Y$

$X \not\perp_d Y$ ✓

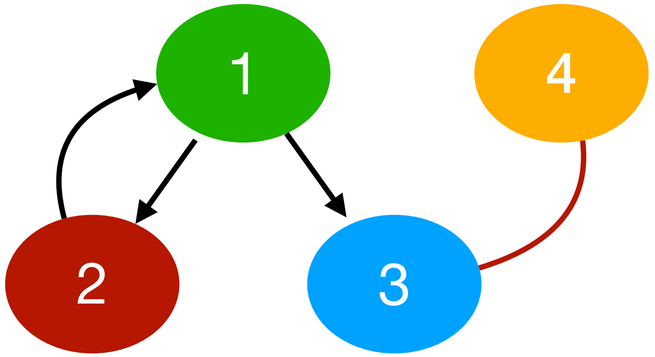
$X \perp_d Y \mid Z$ ✓

Recap from class 3: Directed graphs vs mixed graphs

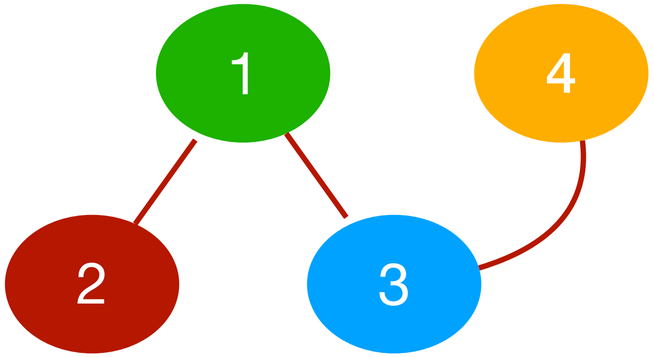
- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
  - $\mathbf{V}$ is the set of **nodes** (vertices)
  - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$
  - If all edges are **directed** $\rightarrow$ then the graph is **directed**



Directed graph



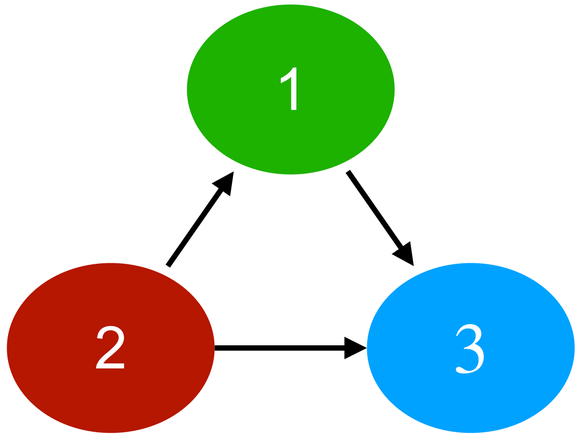
Mixed graph

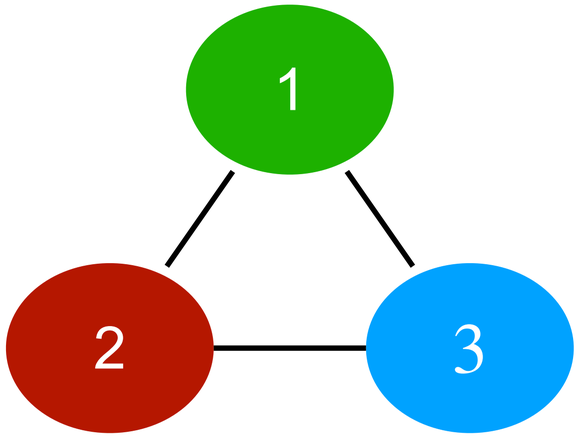


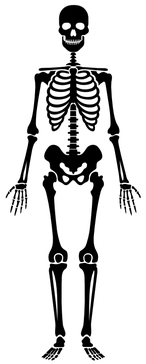
Undirected graph

More graph terminology: skeletons

- The skeleton of a DAG $G = (\mathbf{V}, \mathbf{E})$ is the undirected graph $U = (\mathbf{V}, \mathbf{E}')$ that has **an undirected edge** $(i, j) \in \mathbf{E}'$ for every **directed edge** $i \to j \in \mathbf{E}$ and no other edges



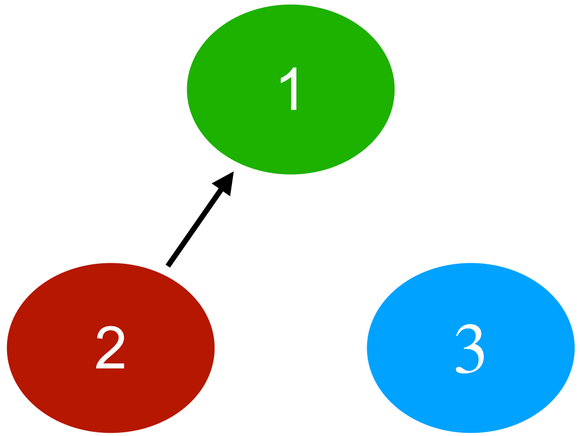




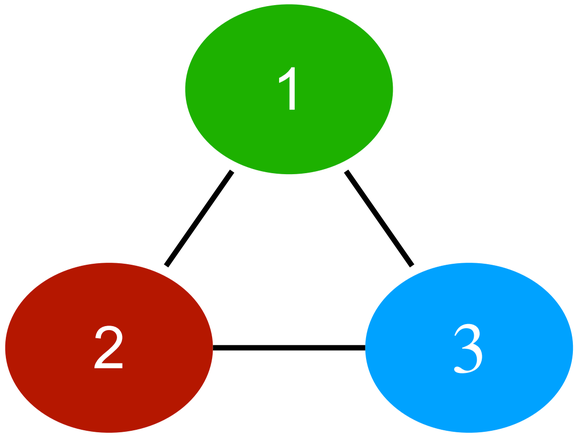
In other words a graph that is the same, but without the arrowheads

Skeleton exercise

- The skeleton of a DAG $G = (\mathbf{V}, \mathbf{E})$ is the undirected graph $U = (\mathbf{V}, \mathbf{E}')$ that has **an undirected edge** $(i, j) \in \mathbf{E}'$ for every **directed edge** $i \to j \in \mathbf{E}$ and no other edges



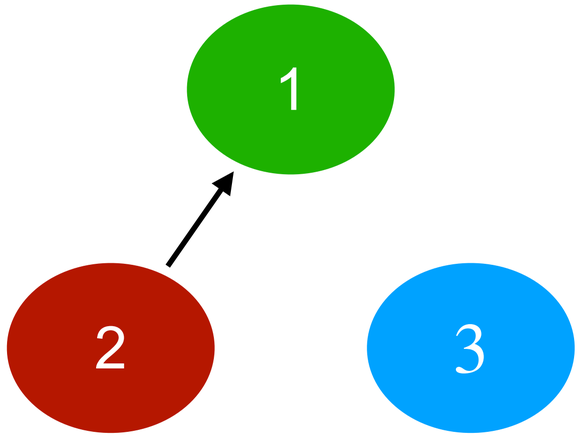
G



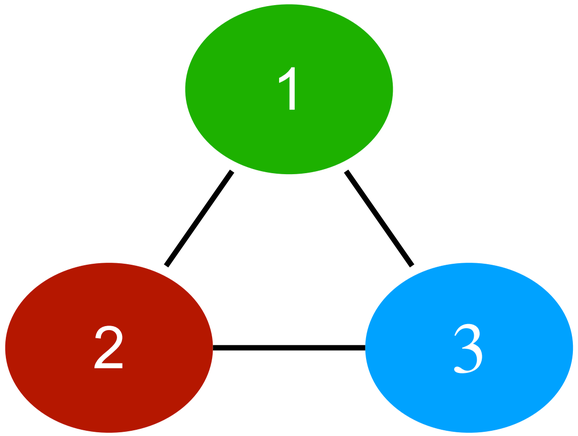
Is this the skeleton of G?

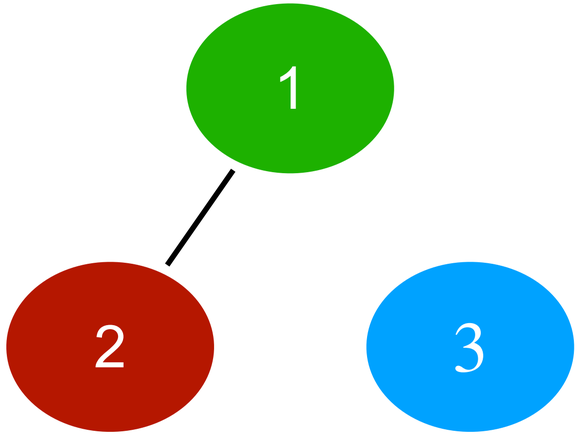
Skeleton exercise

- The skeleton of a DAG $G = (\mathbf{V}, \mathbf{E})$ is the undirected graph $U = (\mathbf{V}, \mathbf{E}')$ that has **an undirected edge** $(i, j) \in \mathbf{E}'$ for every **directed edge** $i \rightarrow j \in \mathbf{E}$ and no other edges



G

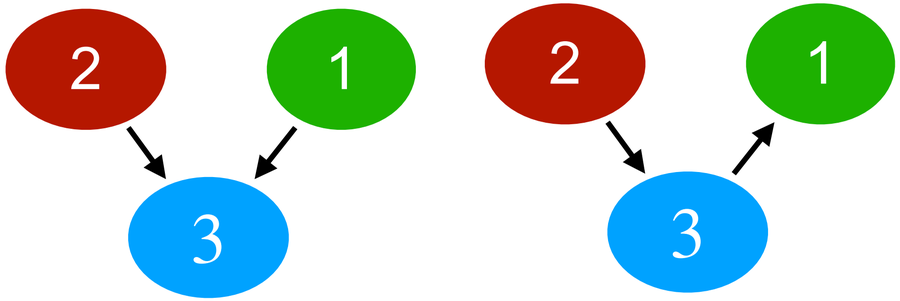




Is this the skeleton of G?

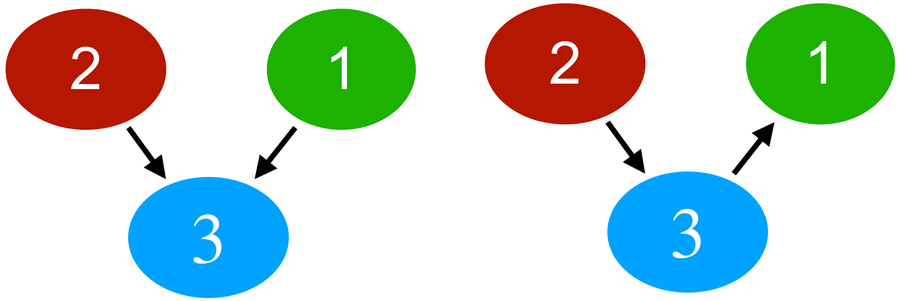
More graph terminology: adjacent

- Nodes i and j in a DAG G are **adjacent/neighbours** if $i \to j$ or $j \to i$ in G



More graph terminology: adjacent

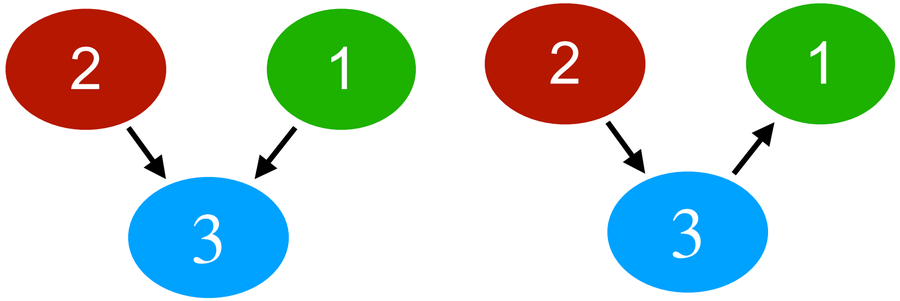
- Nodes i and j in a DAG G are **adjacent/neighbours** if $i \to j$ or $j \to i$ in G
  - I.e., they are connected by an undirected edge in the skeleton of G
  - We denote adjacency with $i - j$, while $i \not- j$ means non-adjacent



More graph terminology: adjacent

- Nodes i and j in a DAG G are **adjacent/neighbours** if $i \to j$ or $j \to i$ in G
  - I.e., they are connected by an undirected edge in the skeleton of G
  - We denote adjacency with $i - j$, while $i \not- j$ means non-adjacent

Is 2 adjacent to 3?

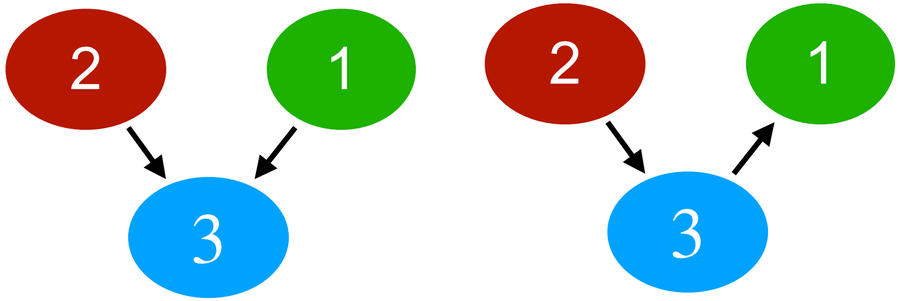


More graph terminology: adjacent

- Nodes i and j in a DAG G are **adjacent/neighbours** if $i \to j$ or $j \to i$ in G
  - I.e., they are connected by an undirected edge in the skeleton of G
  - We denote adjacency with $i - j$, while $i \not- j$ means non-adjacent

Is 2 adjacent to 3?

Is 2 adjacent to 1?



More graph terminology: v-structures/immoralities

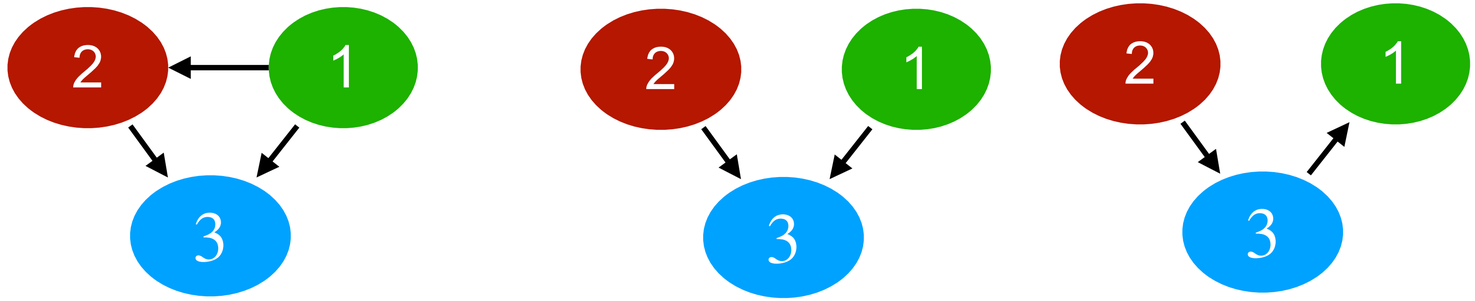
- A triple of nodes (i, j, k) in a DAG G is a **v-structure (unshielded collider)** if $i \to j \leftarrow k$ in G and i **is not adjacent to** k

This edge is called a shield

It's also called an immorality, because the two parents i and j who have a common child k are not "married" (adjacent)

More graph terminology: v-structures/immoralities

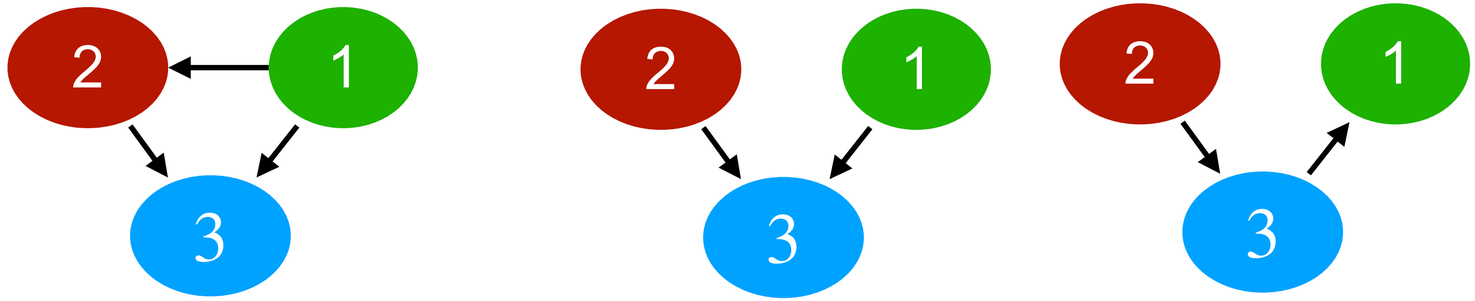
- A triple of nodes (i, j, k) in a DAG G is a **v-structure (unshielded collider)** if $i \to j \leftarrow k$ in G and i **is not adjacent to** k



More graph terminology: v-structures/immoralities

- A triple of nodes (i, j, k) in a DAG G is a **v-structure (unshielded collider)** if $i \to j \leftarrow k$ in G and i **is not adjacent to** k

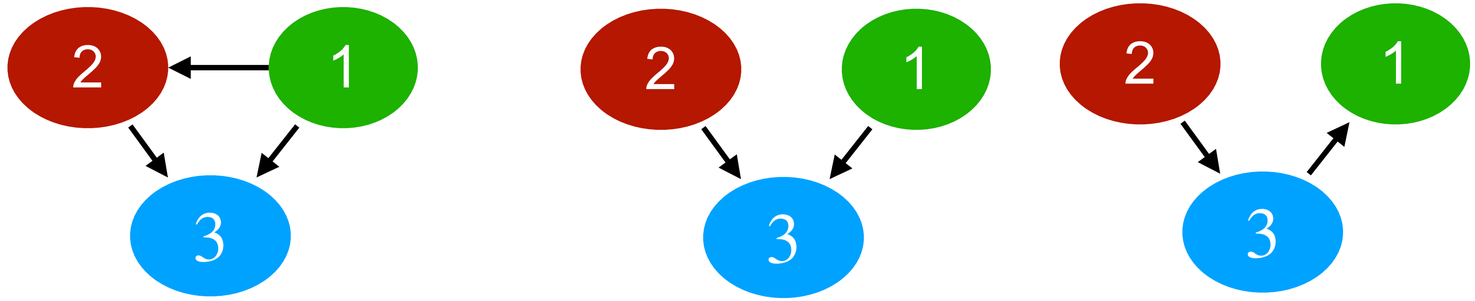
In which graphs is 3 a collider on the path between 2 and 1?



More graph terminology: v-structures/immoralities

- A triple of nodes (i, j, k) in a DAG G is a **v-structure (unshielded collider)** if $i \to j \leftarrow k$ in G and i **is not adjacent to** k

In which graphs is 3 part of a v-structure (2,3,1)?

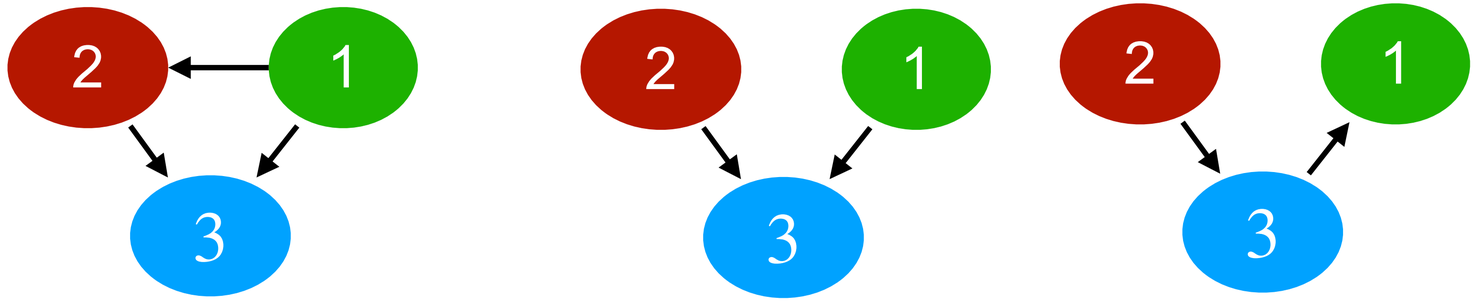


More graph terminology: unshielded triple

- A triple of nodes (i, j, k) in a DAG G is a **v-structure (unshielded collider)** if $i \rightarrow j \leftarrow k$ in G and i **is not adjacent to** k in G

- A triple of nodes (i, j, k) in a DAG G is a **an unshielded triple** if $i - j, j - k$ and i **is not adjacent to k** in G

In which graphs is (2,3,1) an unshielded triple?



Markov equivalence class and CPDAGs

essential graphs Summary graphs

- (Verma and Pearl 1990) show that all DAGs in a Markov equivalence class have the **same skeleton** and the **same v-structures**

Markov equivalence class and CPDAGs

- (Verma and Pearl 1990) show that all DAGs in a Markov equivalence class have the **same skeleton** and the **same v-structures**

- We can represent the skeleton and the orientations (edge marks) all DAGs in a Markov equivalence class (MEC) have in common with a **Complete Partially Directed Acyclic Graph (CPDAG):**

Markov equivalence class and CPDAGs

- (Verma and Pearl 1990) show that all DAGs in a Markov equivalence class have the **same skeleton** and the **same v-structures**

- We can represent the skeleton and the orientations (edge marks) all DAGs in a Markov equivalence class (MEC) have in common with a **Complete Partially Directed Acyclic Graph (CPDAG):**
  - We have a **directed** edge $i \to j$ if **all DAGs** in the MEC have $i \to j$

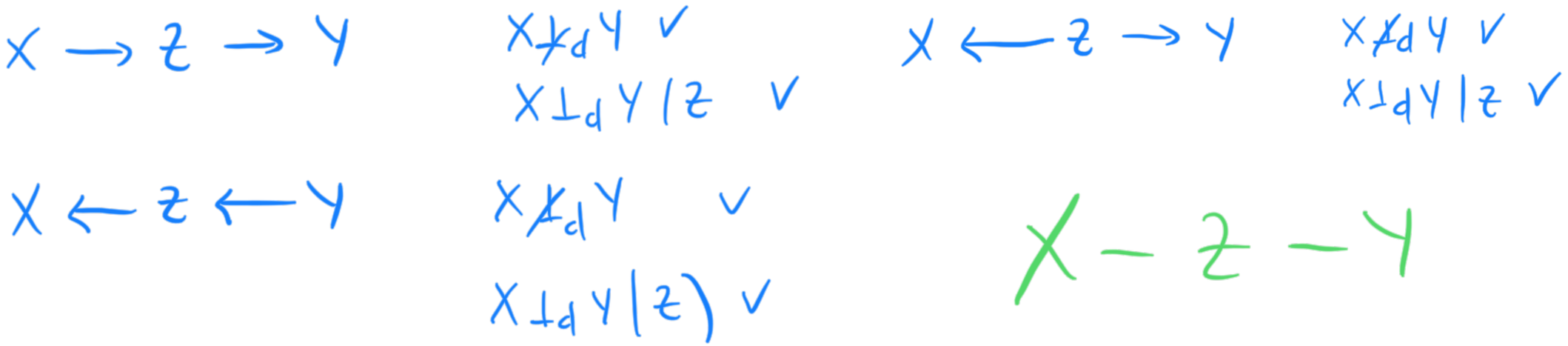
Markov equivalence class and CPDAGs

- (Verma and Pearl 1990) show that all DAGs in a Markov equivalence class have the **same skeleton** and the **same v-structures**

- We can represent the skeleton and the orientations (edge marks) all DAGs in a Markov equivalence class (MEC) have in common with a **Complete Partially Directed Acyclic Graph (CPDAG):**
  - We have a **directed** edge $i \rightarrow j$ if **all DAGs** in the MEC have $i \rightarrow j$
  - We have an **undirected** edge $i - j$ if **some DAGs** in the MEC have $i \rightarrow j$ and others have $j \rightarrow i$

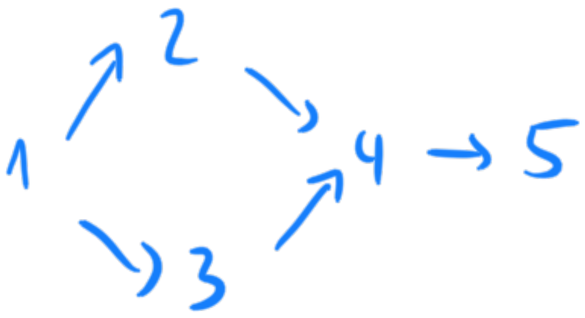
Markov equivalence class and CPDAGs

- **Complete Partially Directed Acyclic Graph (CPDAG):**
  - We have a directed edge $i \to j$ if all DAGs in the MEC have $i \to j$
  - We have an undirected edge $i - j$ if some DAGs in the MEC have $i \to j$ and others have $j \to i$



CPDAG example 2

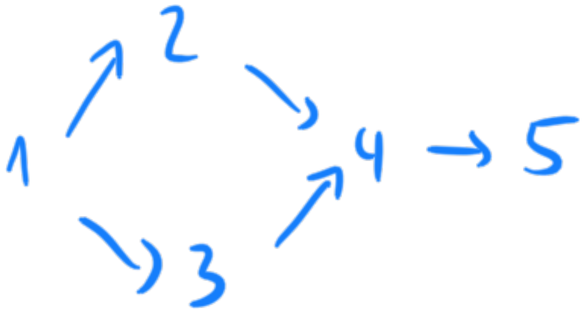
- (Verma and Pearl 1990) show that all DAGs in a Markov equivalence class have the **same skeleton** and the **same v-structures**

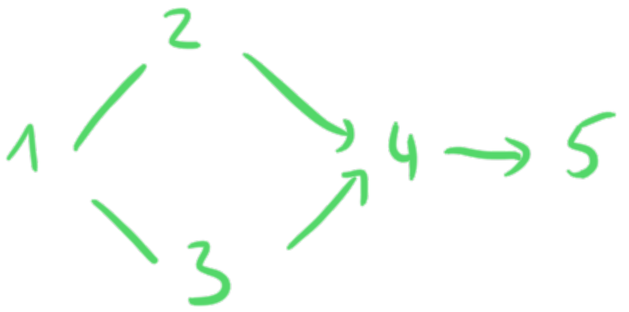


- A triple of nodes (i, j, k) in a DAG G is a **v-structure**
  - if $i \rightarrow j \leftarrow k$ in G and
  - **i is not adjacent to k**

CPDAG example 2

- (Verma and Pearl 1990) show that all DAGs in a Markov equivalence class have the **same skeleton** and the **same v-structures**

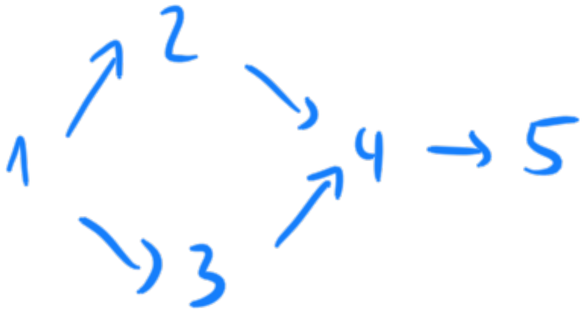


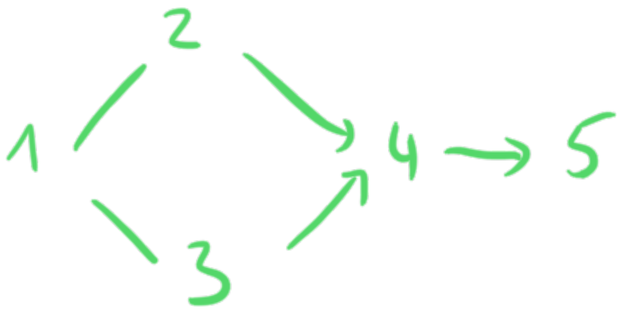


If the true graph is known we can compute the CPDAG representing the MEC

CPDAG example 2

- (Verma and Pearl 1990) show that all DAGs in a Markov equivalence class have the **same skeleton** and the **same v-structures**





If the true graph is known we can compute the CPDAG representing the MEC

What if it isn't known?!

Constraint-based causal discovery

- **Idea:** we perform conditional independence tests on **observational** data and use them to constrain the possible graphs using d-separation

$X \not\perp\!\!\!\perp Y$ $X \perp\!\!\!\perp Y \mid Z$

$X - Z - Y$

Constraint-based causal discovery

- **Idea:** we perform conditional independence tests on **observational** data and use them to constrain the possible graphs using d-separation

- In general, we can narrow down the possible graphs only up to their **Markov equivalence class (MEC)**

Constraint-based causal discovery

- **Idea:** we perform conditional independence tests on **observational** data and use them to constrain the possible graphs using d-separation

- In general, we can narrow down the possible graphs only up to their **Markov equivalence class (MEC)**

- The output of the algorithms we will see (e.g. SGS, PC) is a **CPDAG**, a mixed graph in which directed edges represent causal relations on which all DAGs in the MEC agree - these relations are **identifiable**

SGS algorithm (Spirtes, Glymour, Scheines)

- Assuming P is Markov and faithful to an unknown graph G

- We can estimate a CPDAG from samples of P in three steps:
  1. Determine the **skeleton**
  2. Determine the **v-structures**
  3. Direct as many remaining edges as possible

SGS algorithm (Spirtes, Glymour, Scheines)

- Assuming P is Markov and faithful to an unknown graph G

- We can estimate a CPDAG from samples of P in three steps:
  1. Determine the **skeleton**
  2. Determine the **v-structures** *(given the tests in the previous phase)*
  3. Direct as many remaining edges as possible

- **Note**: the directed parts of the CPDAG will agree with G, but some parts might stay undirected

Step 1: Skeleton learning

- Given $G = (\mathbf{V}, \mathbf{E})$, nodes $i, j \in \mathbf{V}, i \neq j$ then:
  - If i **is adjacent to** j, they cannot be d-separated by any subset of remaining nodes (and vice-versa)

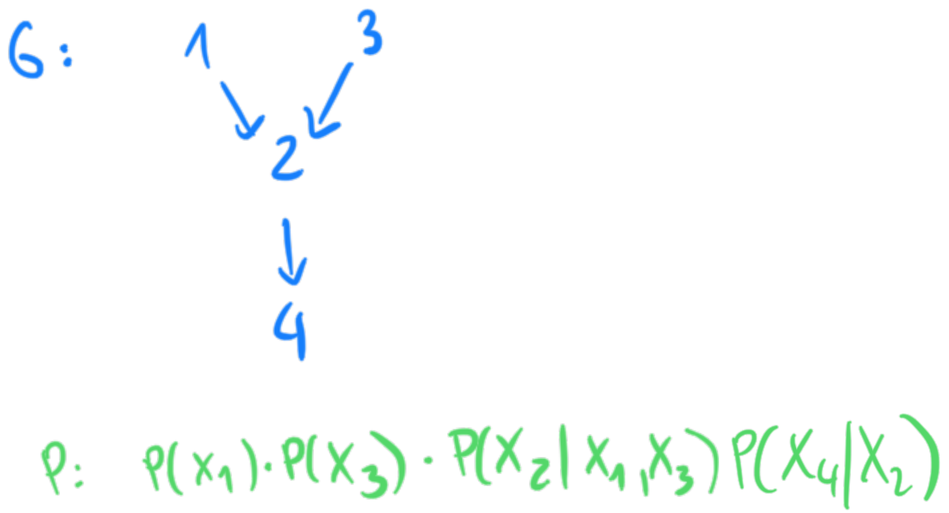
Step 1: Skeleton learning

- Given $G = (\mathbf{V}, \mathbf{E})$, nodes $i, j \in \mathbf{V}, i \neq j$ then:
  - If i **is adjacent to** j, they cannot be d-separated by any subset of remaining nodes (and vice-versa)

1. Start with **completely connected undirected graph** U
2. For each pair $i, j \in \mathbf{V}, i \neq j$, and for any subset $\mathbf{S} \subseteq \mathbf{V} \backslash \{i, j\}$
   - Check **if** $X_i \perp\!\!\!\perp X_j \,|\, X_{\mathbf{S}}$ for any $\mathbf{S}$ in data
     - If this is true, by faithfulness $i \perp_G j \,|\, \mathbf{S}$, so we can **remove** $i - j$ **in** U

Step 1: Skeleton learning - example

True causal graph



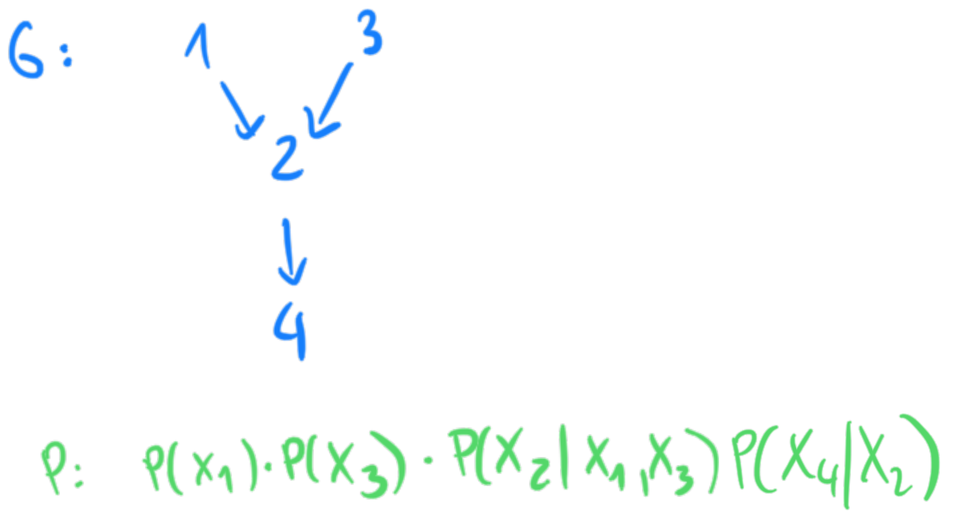
Data

Algorithm output

For this example, we will pretend we do not know the true graph, but only the CIs that we can derive from data

Step 1: Skeleton learning - example

True causal graph

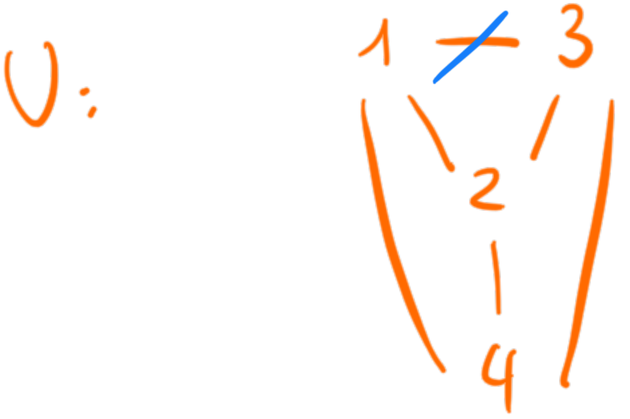


Data

X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0

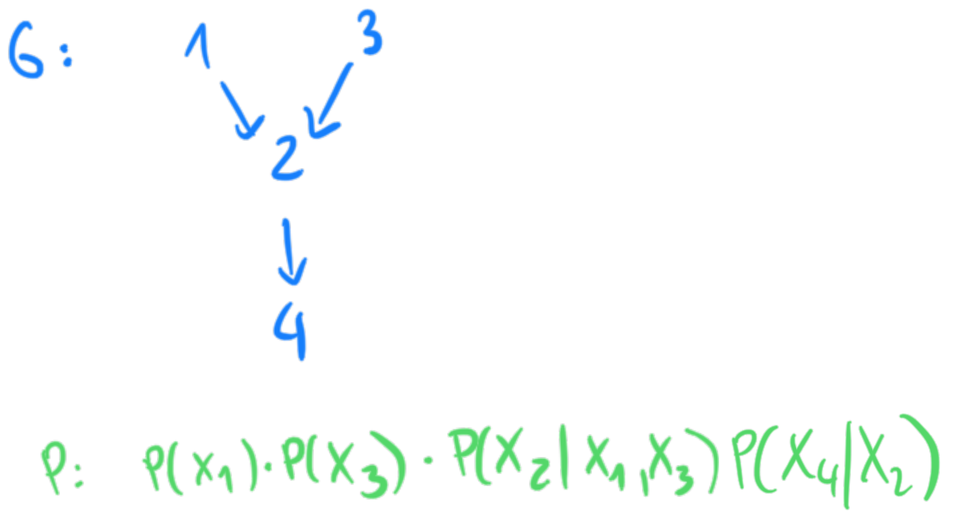
$X_1 \perp\!\!\!\perp X_3$

Algorithm output



Step 1: Skeleton learning - example

True causal graph

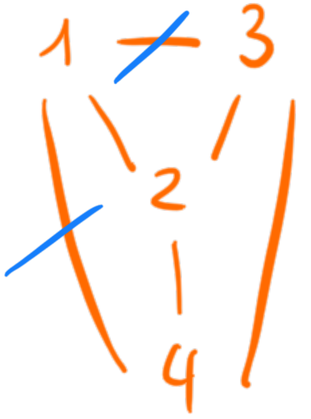


Data

X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0

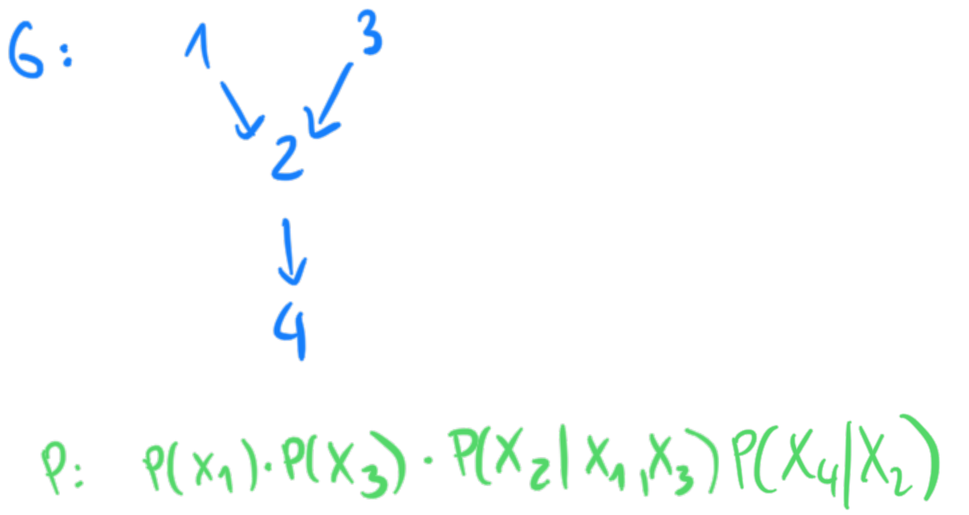
$X_1 \perp\!\!\!\perp X_4 \mid X_2$

Algorithm output



Step 1: Skeleton learning - example

True causal graph

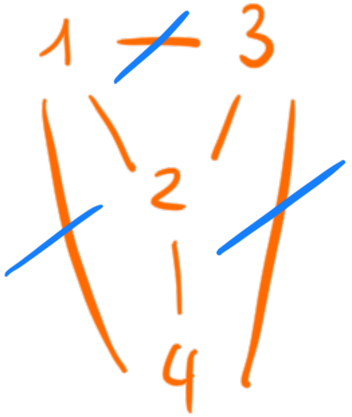


Data

X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0

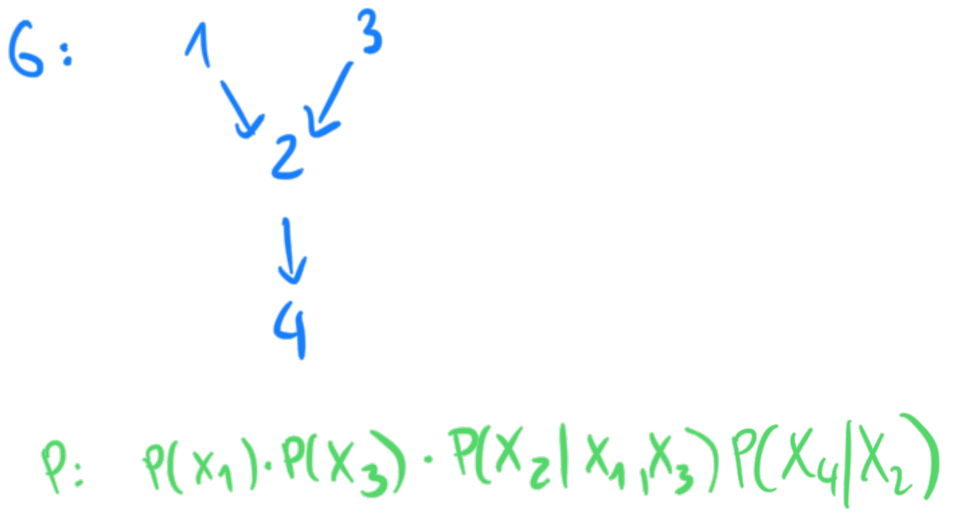
$X_3 \perp\!\!\!\perp X_4 \mid X_2$

Algorithm output



Step 1: Skeleton learning - example

True causal graph



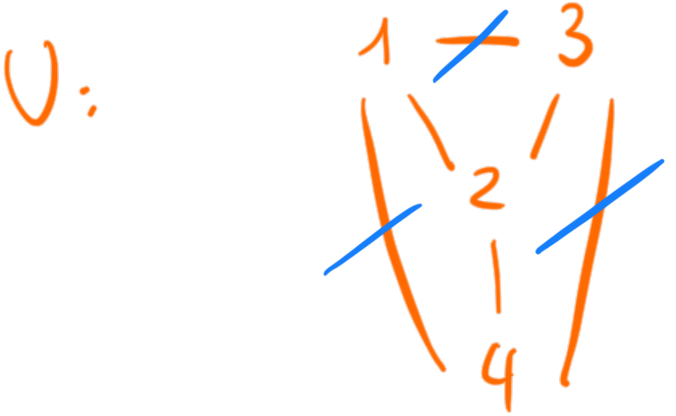
Data

X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$

$X_3 \perp\!\!\!\perp X_4 \mid X_2, X_1$

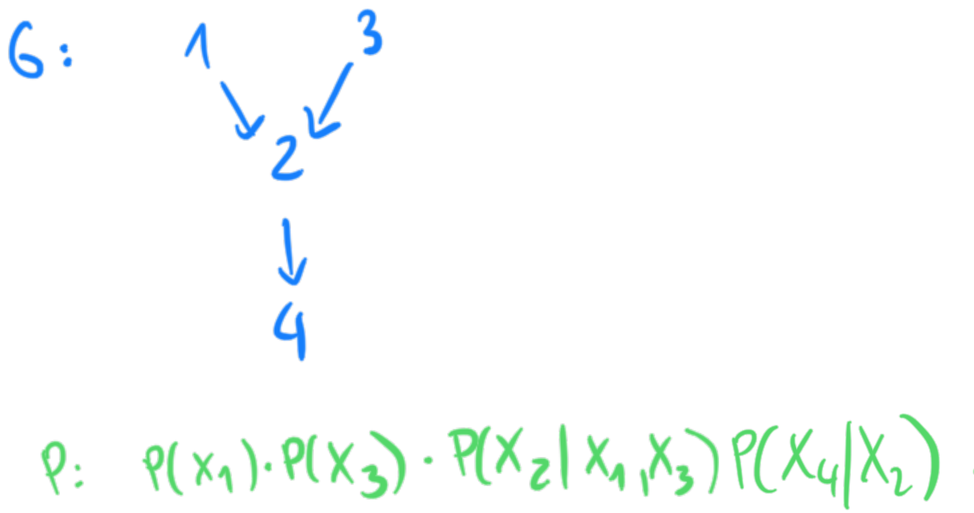
Algorithm output



Nothing changes, we can stop

Step 1: Skeleton learning - example

True causal graph



Data

Algorithm output

Step 1 finds the correct skeleton

SGS algorithm (Spirtes, Glymour, Scheines)

- Assuming P is Markov and faithful to an unknown graph G

- We can estimate a CPDAG from samples of P in three steps:
  1. ~~Determine the **skeleton**~~
  2. Determine the **v-structures** *(given the tests in the previous phase)*
  3. Direct as many remaining edges as possible

- **Note**: the directed parts of the CPDAG will agree with G, but some parts might stay undirected

Step 2: Determine v-structures

- A triple of nodes (i, j, k) in a DAG G is a **an unshielded triple** if $i - j, j - k$ and i **is not adjacent to k**, i.e. $i \neq k$, in G

Step 2: Determine v-structures

- A triple of nodes (i, j, k) in a DAG G is a **an unshielded triple** if $i - j, j - k$ and i **is not adjacent to k**, i.e. $i \not- k$, in G

1. Start from the skeleton U from previous step
2. For each unshielded triple (i, j, k) in U, i.e. $i - j, j - k$ **and** $i \not- k$ **in** U
   - For all $\mathbf{S} \subseteq \mathbf{V} \backslash \{i, j, k\}$ check if $X_i \not\perp\!\!\!\perp X_k \mid X_j \cup X_{\mathbf{S}}$ in data
     - If this is true, $i \rightarrow j \leftarrow k$ **is a v-structure**

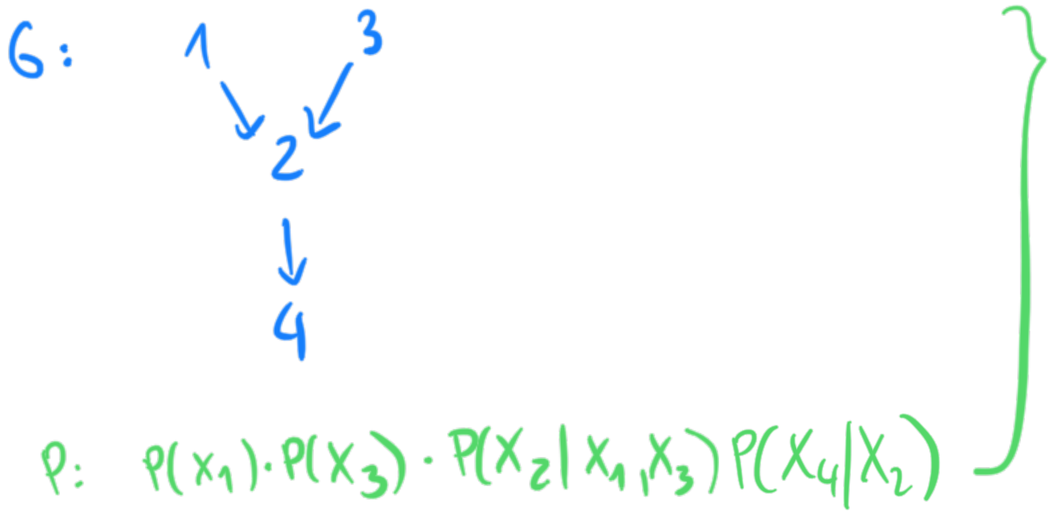
Step 2: Determine v-structures

- A triple of nodes (i, j, k) in a DAG G is a **an unshielded triple** if $i - j, j - k$ and i **is not adjacent to k**, i.e. $i \neq k$ in G

Keep in mind: for unshielded triples (i, j, k) we check if X_i, X_j are always dependent **given any conditioning set containing** X_j

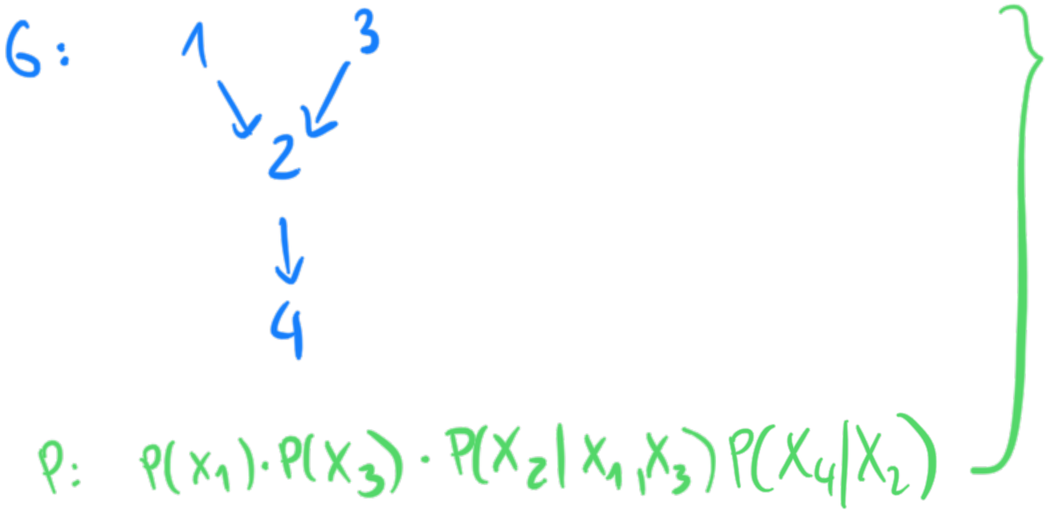
1. Start from the skeleton U from pre
2. For each unshielded triple (i, j, k) in U, i.e. $i - j, j - k$ **and** $i \neq k$ **in** U
   - For all $\mathbf{S} \subseteq \mathbf{V} \backslash \{i, j, k\}$ check if $X_i \not\perp\!\!\!\perp X_k \mid X_j \cup X_{\mathbf{S}}$ in data
     - If this is true, $i \rightarrow j \leftarrow k$ **is a v-structure**

Step 2: Determine v-structures - example

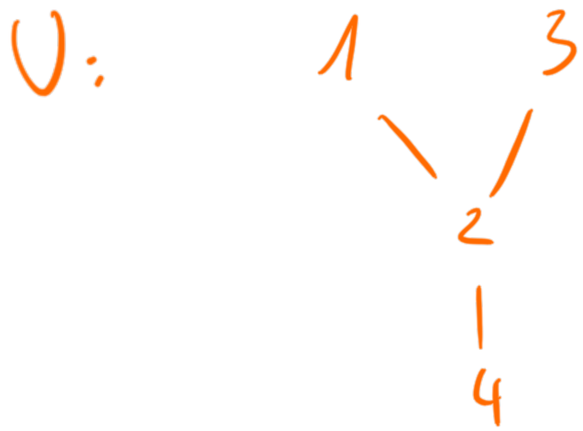


X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0

Step 2: Determine v-structures - example



X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0



We can reuse the CIs from phase 1

$X_1 \perp\!\!\!\perp X_3$

$X_1 \perp\!\!\!\perp X_4 | X_2$

$X_3 \perp\!\!\!\perp X_4 | X_2$

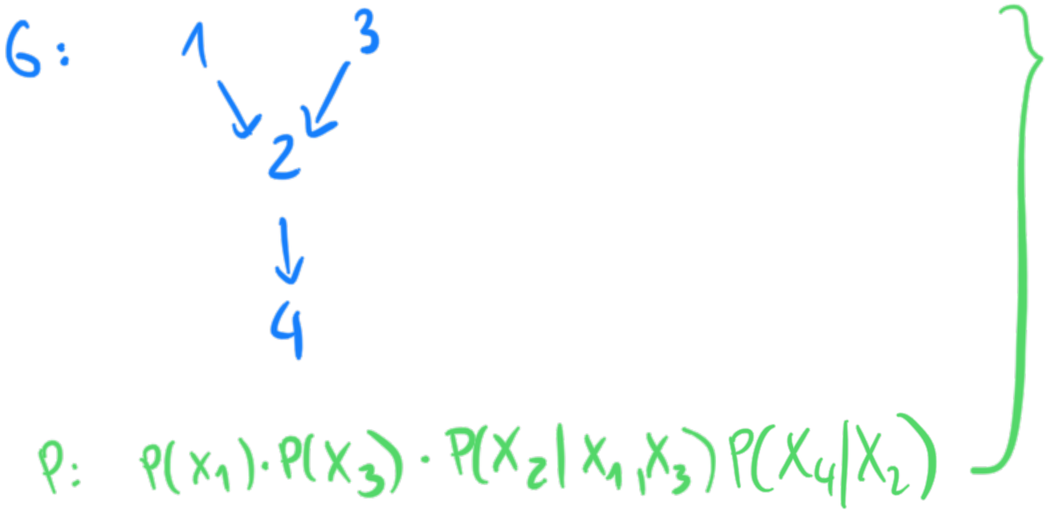
$X_1 \perp\!\!\!\perp X_4 | X_2, X_3$

$X_3 \perp\!\!\!\perp X_4 | X_2, X_1$

1 – 2 – 4: $X_1 \perp\!\!\!\perp X_4 | X_2$

$X_1 \perp\!\!\!\perp X_4 | X_2, X_3$ X

Step 2: Determine v-structures - example



X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0

We can reuse the CIs from phase 1

$$X_1 \perp\!\!\!\perp X_3$$
$$X_1 \perp\!\!\!\perp X_4 | X_2$$
$$X_3 \perp\!\!\!\perp X_4 | X_2$$
$$X_1 \perp\!\!\!\perp X_4 | X_2, X_3$$
$$X_3 \perp\!\!\!\perp X_4 | X_2, X_1$$

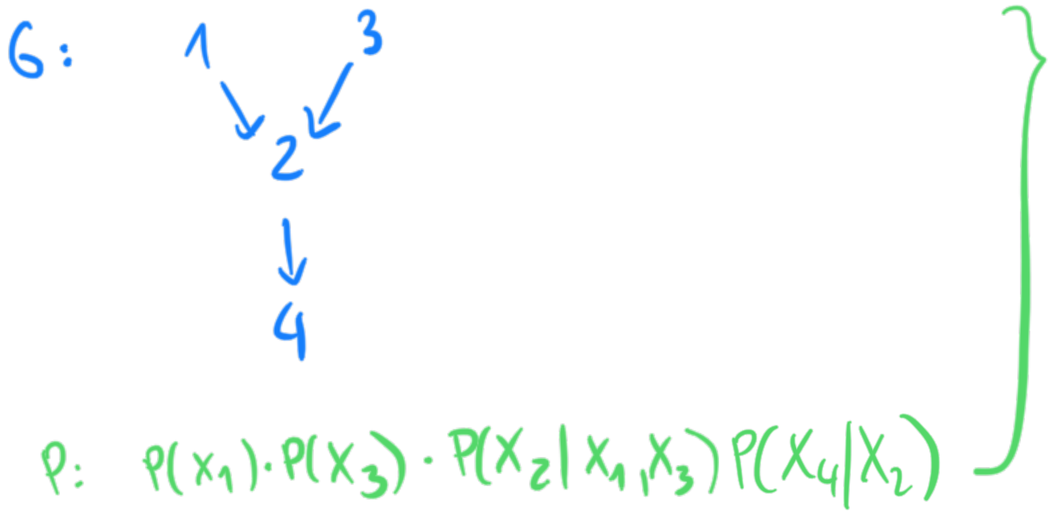
3 - 2 - 4

$X_3 \perp\!\!\!\perp X_4 | X_2$

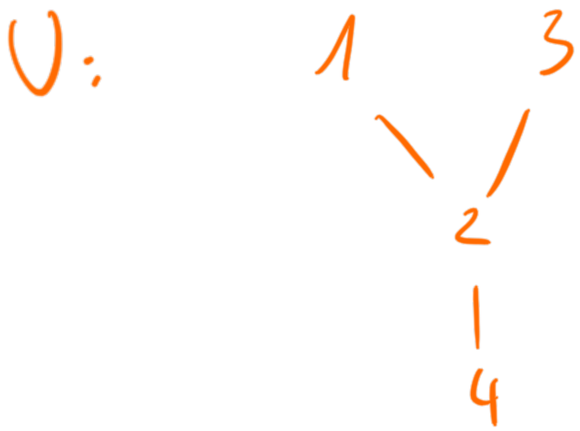
$X_3 \perp\!\!\!\perp X_4 | X_2, X_1$

X

Step 2: Determine v-structures - example



X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0



We can reuse the CIs from phase 1

$X_1 \perp\!\!\!\perp X_3$
$X_1 \perp\!\!\!\perp X_4 | X_2$
$X_3 \perp\!\!\!\perp X_4 | X_2$
$X_1 \perp\!\!\!\perp X_4 | X_2, X_3$
$X_3 \perp\!\!\!\perp X_4 | X_2, X_1$

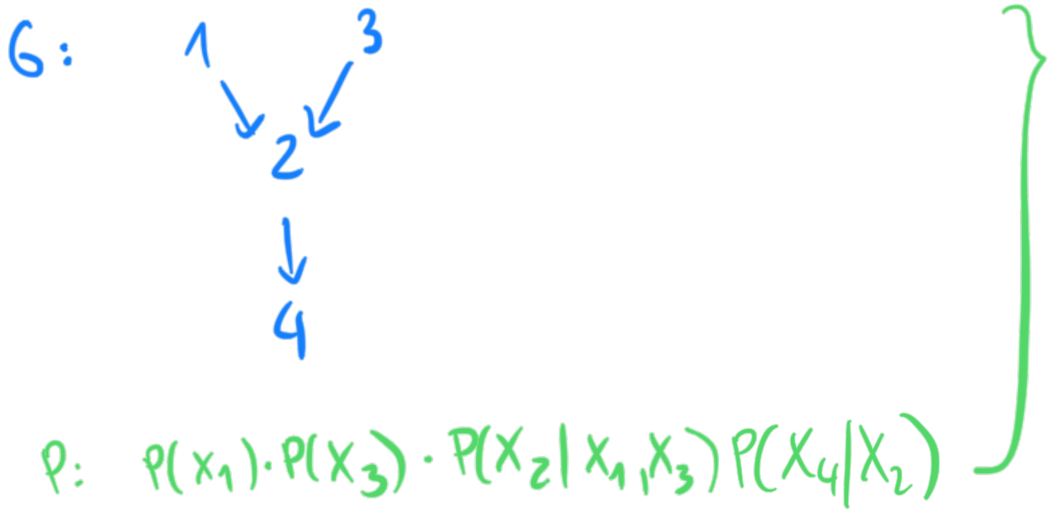
1 - 2 - 3

$X_1 \not\perp\!\!\!\perp X_3 | X_2$ ✓

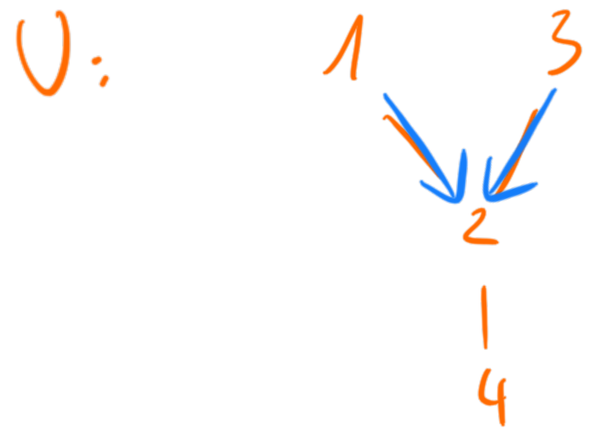
$X_1 \not\perp\!\!\!\perp X_3 | X_2, X_4$ ✓

$X_1 \perp\!\!\!\perp X_3$ has $\mathbf{Z} = \emptyset$, so it does not contain X_2 and we ignore it

Step 2: Determine v-structures - example



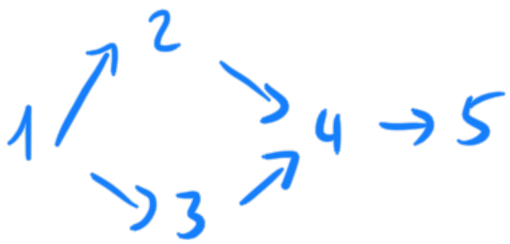
X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0



$1 - 2 - 3$ $X_1 \not\perp X_3 | X_2$ ✓

$X_1 \not\perp X_3 | X_2, X_4$ ✓

Step 2: Determine v-structures - example 2



$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$

$X_2 \perp\!\!\!\perp X_3 \mid X_1$

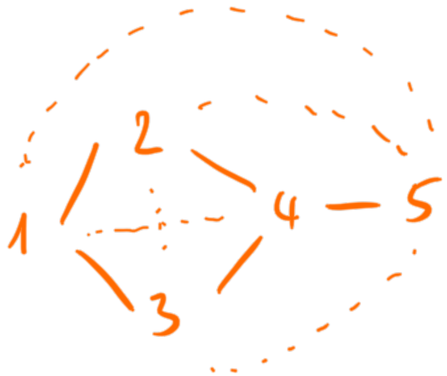
$X_3 \perp\!\!\!\perp X_5 \mid X_4$

$X_2 \perp\!\!\!\perp X_5 \mid X_4$

$X_1 \perp\!\!\!\perp X_5 \mid X_4$

Step 2: Determine v-structures - example 2

U:



$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$

$X_2 \perp\!\!\!\perp X_3 \mid X_1$

$X_3 \perp\!\!\!\perp X_5 \mid X_4$

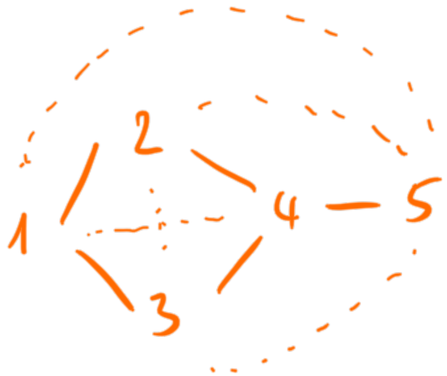
$X_2 \perp\!\!\!\perp X_5 \mid X_4$

$X_1 \perp\!\!\!\perp X_5 \mid X_4$

Step 2: Determine v-structures - example 2

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$
$X_2 \perp\!\!\!\perp X_3 \mid X_1$
$X_3 \perp\!\!\!\perp X_5 \mid X_4$
$X_2 \perp\!\!\!\perp X_5 \mid X_4$
$X_1 \perp\!\!\!\perp X_5 \mid X_4$

U:



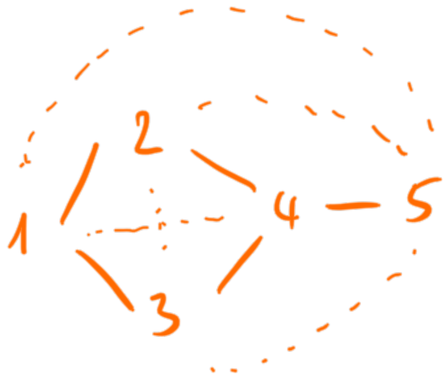
1 - 2 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_2$
$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ X

Step 2: Determine v-structures - example 2

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$
$X_2 \perp\!\!\!\perp X_3 \mid X_1$
$X_3 \perp\!\!\!\perp X_5 \mid X_4$
$X_2 \perp\!\!\!\perp X_5 \mid X_4$
$X_1 \perp\!\!\!\perp X_5 \mid X_4$

U:



1 - 2 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_2$

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ X

1 - 3 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_3$

$X_1 \perp\!\!\!\perp X_4 \mid X_3, X_2$

Step 2: Determine v-structures - example 2

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$

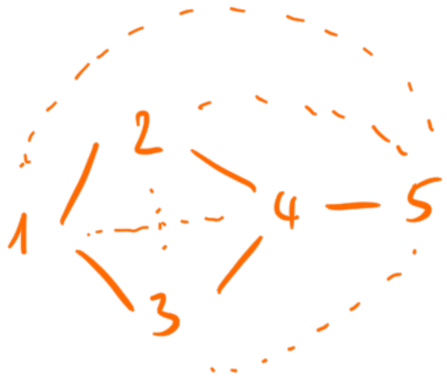
$X_2 \perp\!\!\!\perp X_3 \mid X_1$

$X_3 \perp\!\!\!\perp X_5 \mid X_4$

$X_2 \perp\!\!\!\perp X_5 \mid X_4$

$X_1 \perp\!\!\!\perp X_5 \mid X_4$

U:



1 - 2 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_2$

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ X

1 - 3 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_3$

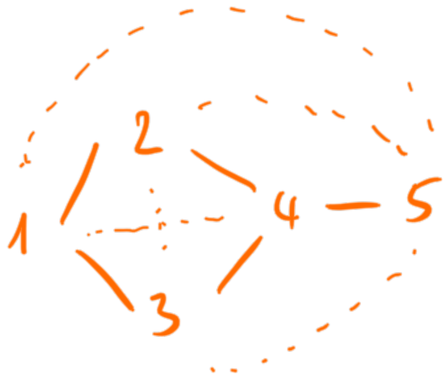
$X_1 \perp\!\!\!\perp X_4 \mid X_3, X_2$

3 - 1 - 2: $X_3 \perp\!\!\!\perp X_2 \mid X_1$ X

Step 2: Determine v-structures - example 2

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$
$X_2 \perp\!\!\!\perp X_3 \mid X_1$
$X_3 \perp\!\!\!\perp X_5 \mid X_4$
$X_2 \perp\!\!\!\perp X_5 \mid X_4$
$X_1 \perp\!\!\!\perp X_5 \mid X_4$

U:



1 - 2 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_2$

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ X

1 - 3 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_3$

$X_1 \perp\!\!\!\perp X_4 \mid X_3, X_2$

3 - 1 - 2: $X_3 \perp\!\!\!\perp X_2 \mid X_1$ X

2 - 4 - 5: $X_2 \perp\!\!\!\perp X_5 \mid X_4$ X

Step 2: Determine v-structures - example 2

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$

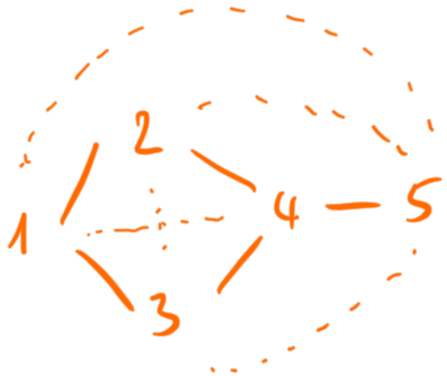
$X_2 \perp\!\!\!\perp X_3 \mid X_1$

$X_3 \perp\!\!\!\perp X_5 \mid X_4$

$X_2 \perp\!\!\!\perp X_5 \mid X_4$

$X_1 \perp\!\!\!\perp X_5 \mid X_4$

U:



1 - 2 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_2$

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ ✗

1 - 3 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_3$

$X_1 \perp\!\!\!\perp X_4 \mid X_3, X_2$

3 - 1 - 2: $X_3 \perp\!\!\!\perp X_2 \mid X_1$ ✗

2 - 4 - 5: $X_2 \perp\!\!\!\perp X_5 \mid X_4$ ✗

3 - 4 - 5: $X_3 \perp\!\!\!\perp X_5 \mid X_4$ ✗

Step 2: Determine v-structures - example 2

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$

$X_2 \perp\!\!\!\perp X_3 \mid X_1$

$X_3 \perp\!\!\!\perp X_5 \mid X_4$

$X_2 \perp\!\!\!\perp X_5 \mid X_4$

$X_1 \perp\!\!\!\perp X_5 \mid X_4$

1 - 2 - 4

$X_2 \perp\!\!\!\perp X_3 \mid X_1$ has $\mathbf{Z} = \{X_1\}$, so $\mathbf{Z}$ does not contain X_4 and we ignore it

1 - 3 - 4

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ ✗

$X_1 \perp\!\!\!\perp X_4 \mid X_3$

$X_1 \perp\!\!\!\perp X_4 \mid X_3, X_2$

3 - 1 - 2: $X_3 \perp\!\!\!\perp X_2 \mid X_1$ ✗

2 - 4 - 5: $X_2 \perp\!\!\!\perp X_5 \mid X_4$ ✗

3 - 4 - 5: $X_3 \perp\!\!\!\perp X_5 \mid X_4$ ✗

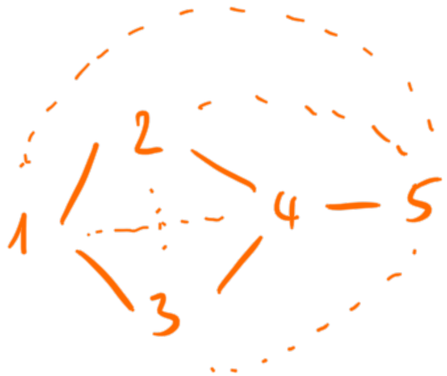
2 - 4 - 3 $X_2 \not\perp\!\!\!\perp X_3 \mid X_4$ $X_2 \not\perp\!\!\!\perp X_3 \mid X_4, X_1$

$X_2 \not\perp\!\!\!\perp X_3 \mid X_4, X_1, X_5$ $X_2 \not\perp\!\!\!\perp X_3 \mid X_4, X_5$

Step 2: Determine v-structures - example 2

$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$
$X_2 \perp\!\!\!\perp X_3 \mid X_1$
$X_3 \perp\!\!\!\perp X_5 \mid X_4$
$X_2 \perp\!\!\!\perp X_5 \mid X_4$
$X_1 \perp\!\!\!\perp X_5 \mid X_4$

U:



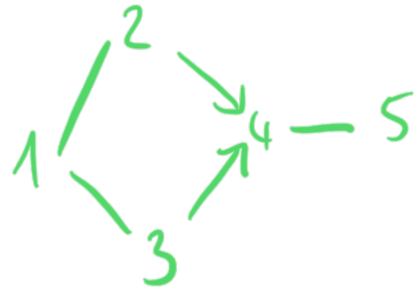
1 - 2 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_2$
$X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3$ ✗

1 - 3 - 4

$X_1 \not\perp\!\!\!\perp X_4 \mid X_3$
$X_1 \perp\!\!\!\perp X_4 \mid X_3, X_2$

G:



3 - 1 - 2: $X_3 \perp\!\!\!\perp X_2 \mid X_1$ ✗

2 - 4 - 5: $X_2 \perp\!\!\!\perp X_5 \mid X_4$ ✗

3 - 4 - 5: $X_3 \perp\!\!\!\perp X_5 \mid X_4$ ✗

2 - 4 - 3 $X_2 \not\perp\!\!\!\perp X_3 \mid X_4$ $X_2 \not\perp\!\!\!\perp X_3 \mid X_4, X_1$
$X_2 \not\perp\!\!\!\perp X_3 \mid X_4, X_1, X_5$ $X_2 \not\perp\!\!\!\perp X_3 \mid X_4, X_5$

SGS algorithm (Spirtes, Glymour, Scheines)

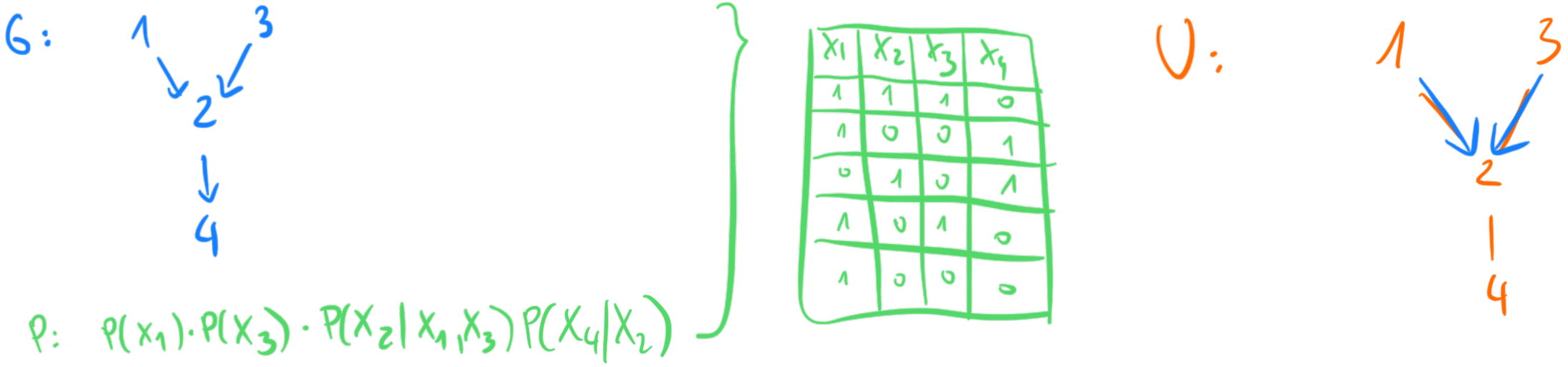
- Assuming P is Markov and faithful to an unknown graph G

- We can estimate a CPDAG from samples of P in three steps:
  1. ~~Determine the **skeleton**~~
  2. ~~Determine the **v-structures** *(given the tests in the previous phase)*~~
  3. Direct as many remaining edges as possible

- **Note**: the directed parts of the CPDAG will agree with G, but some parts might stay undirected

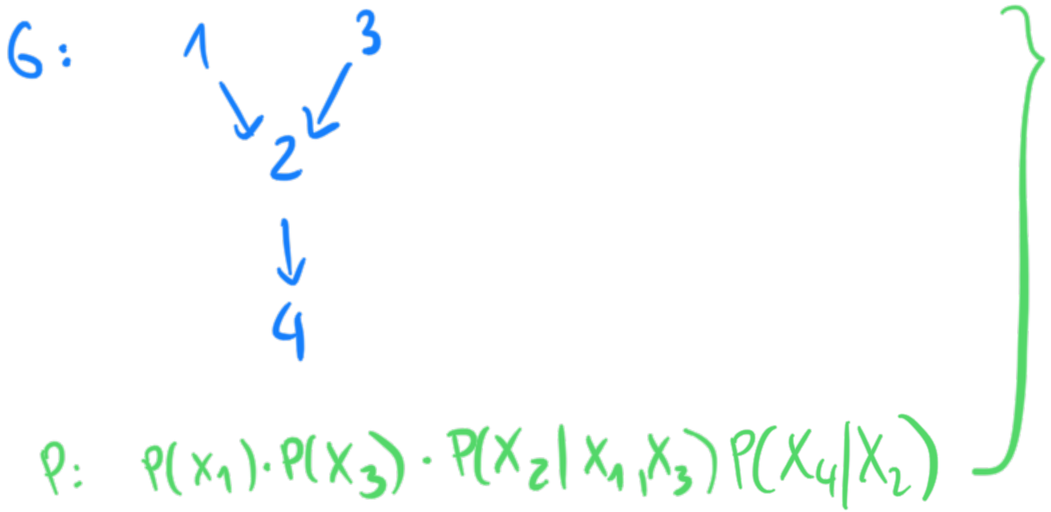
Step 3: Direct as many edges as possible - example

- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen

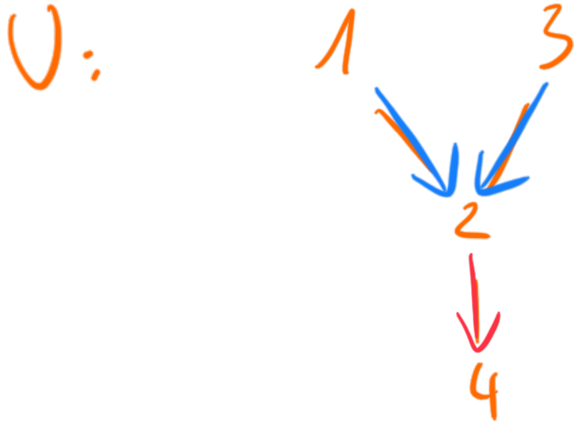


Step 3: Direct as many edges as possible - example

- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen

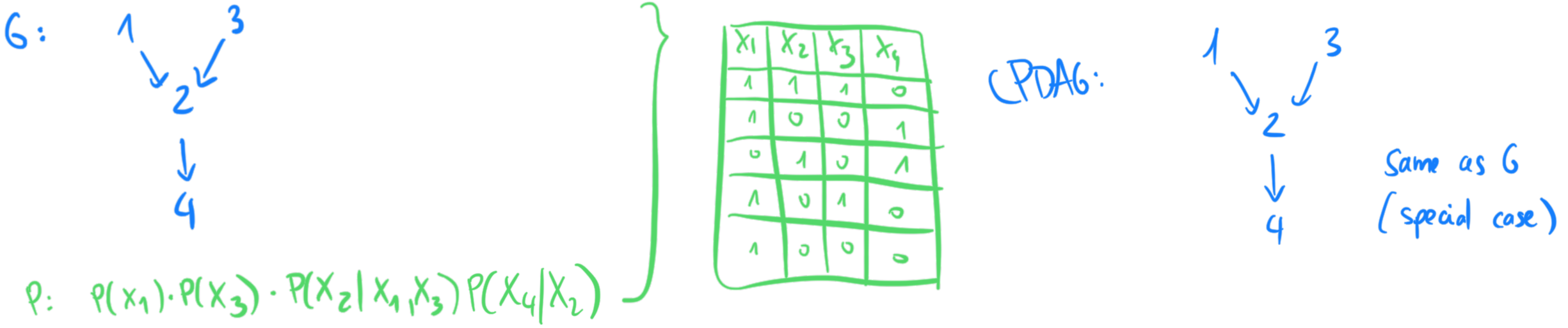


X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0



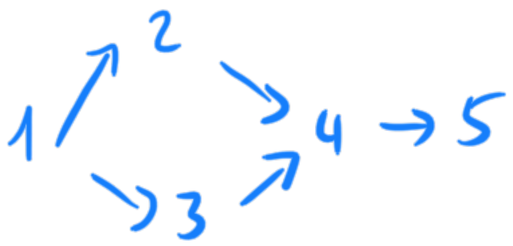
Step 3: Direct as many edges as possible - example

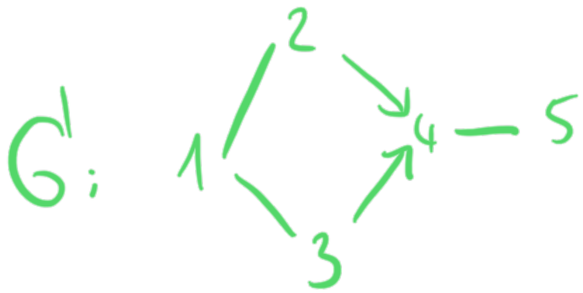
- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen



Step 3: Direct as many edges as possible - example

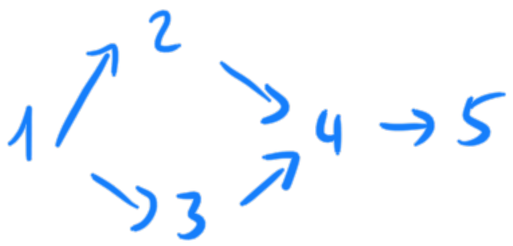
- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen

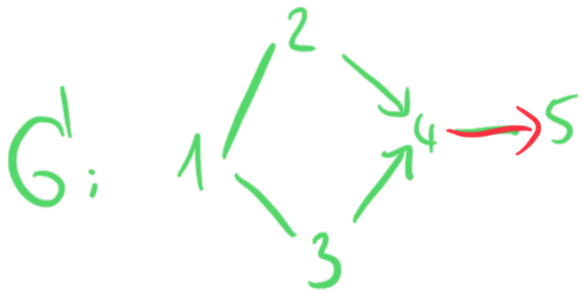




Step 3: Direct as many edges as possible - example

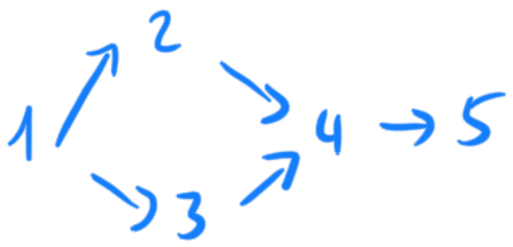
- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen

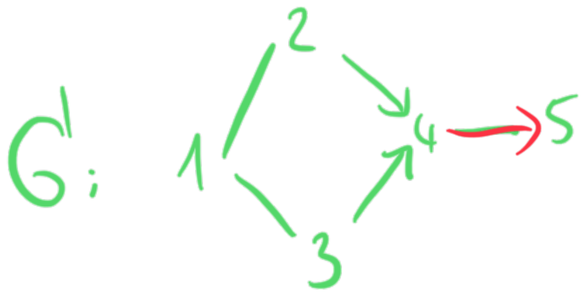


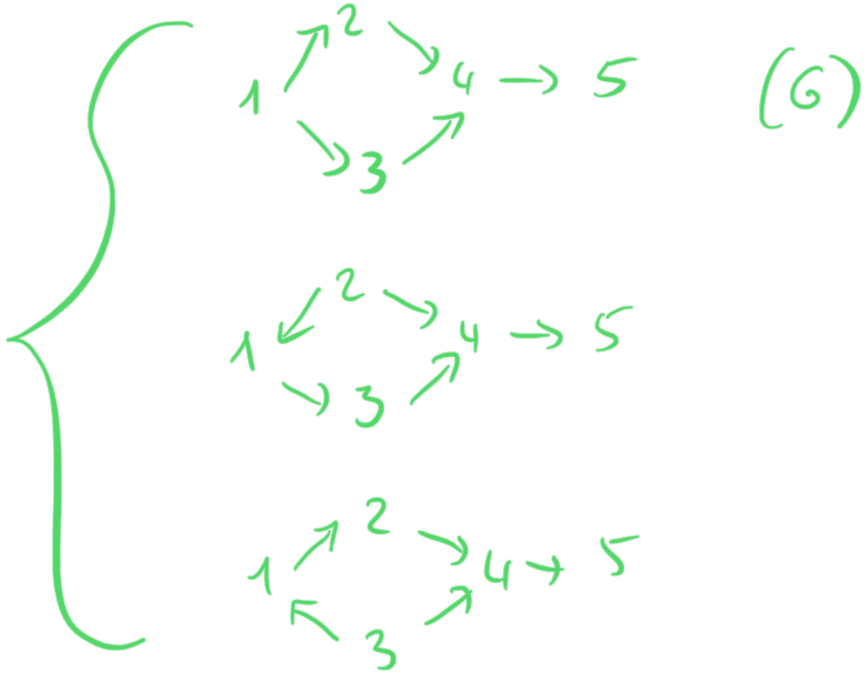


Step 3: Direct as many edges as possible - example

- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen

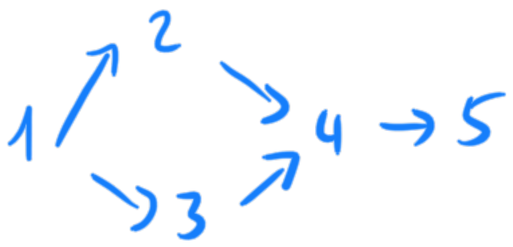


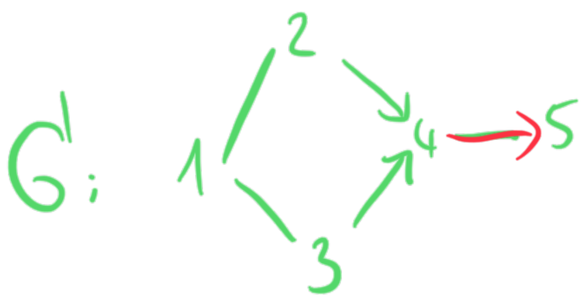


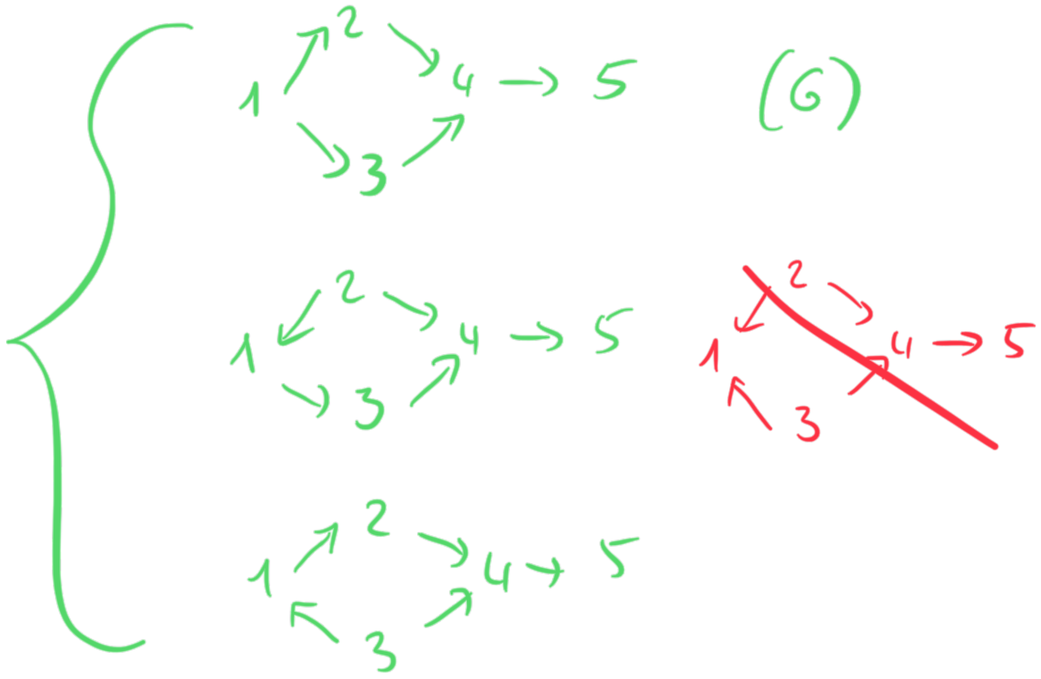


Step 3: Direct as many edges as possible - example

- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen







Step 3: Meek's rules (1995)

Sound and complete rules for additional orientations (also with added background knowledge)

otherwise v-structure

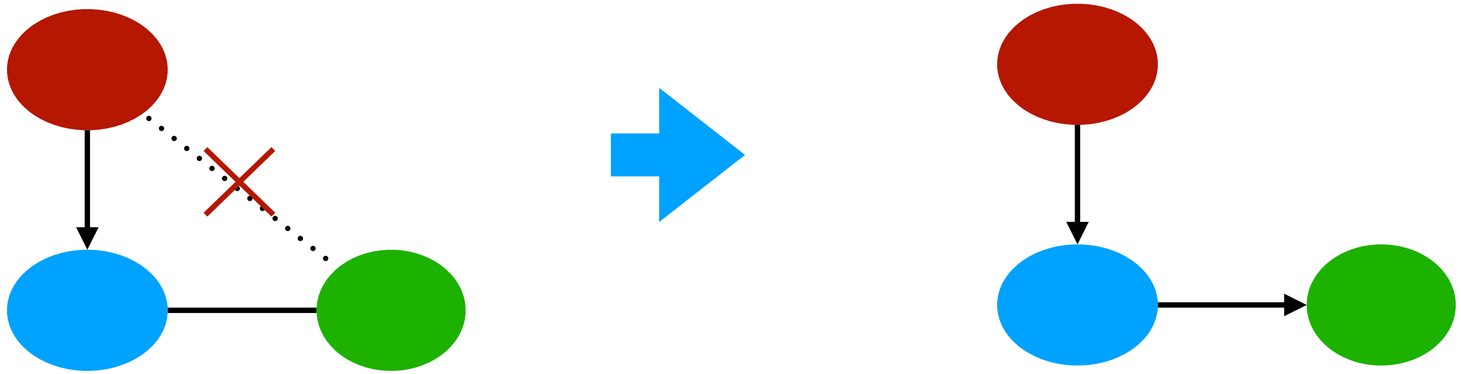
Figure 1: Orientation rules for patterns

otherwise cycle

https://arxiv.org/pdf/1302.4972.pdf

Step 3: Meek's rules (1995) - Rule 1

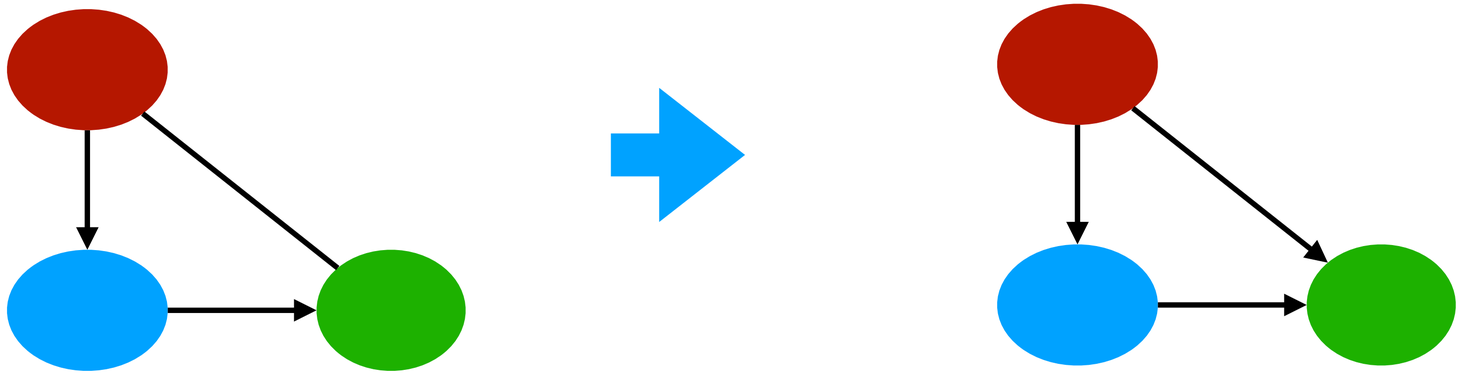
Sound and complete rules for additional orientations (also with added background knowledge)



Otherwise v-structure

Step 3: Meek's rules (1995) - Rule 2

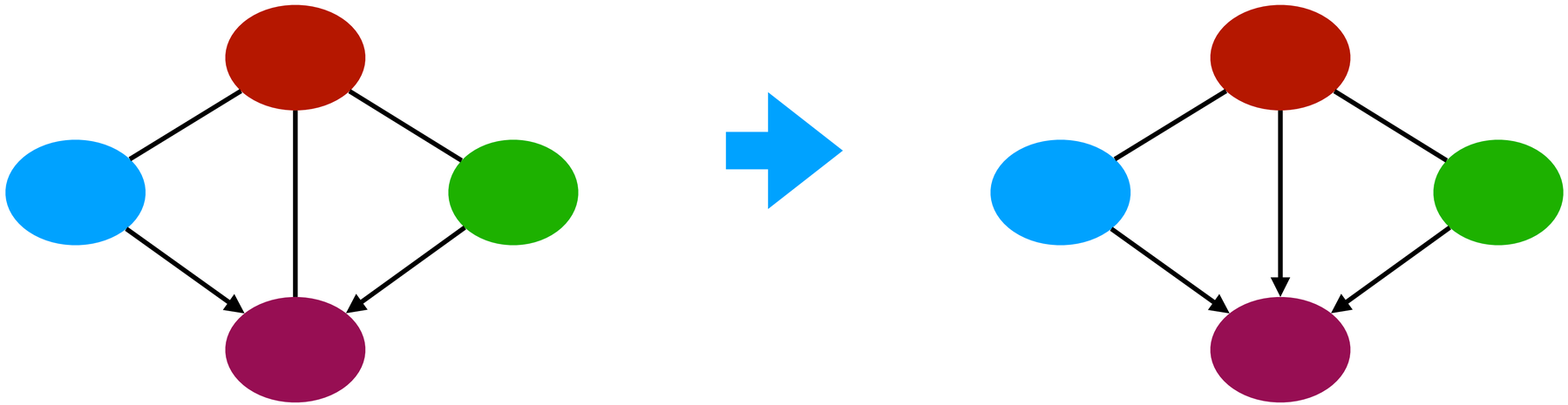
Sound and complete rules for additional orientations (also with added background knowledge)



Otherwise cycle

Step 3: Meek's rules (1995) - Rule 3

Sound and complete rules for additional orientations (also with added background knowledge)



Otherwise cycle or v-structure

Canvas exercise - SGS algorithm

$X_1 \perp\!\!\!\perp X_4$

$X_2 \perp\!\!\!\perp X_4$

$X_1 \perp\!\!\!\perp X_3 \mid X_2$

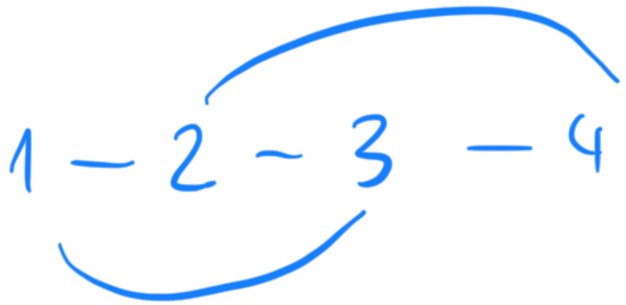
$X_1 \perp\!\!\!\perp X_4 \mid X_2$

$X_2 \perp\!\!\!\perp X_4 \mid X_1$

$X_1 \perp\!\!\!\perp X_4 \mid X_3, X_2$

1) skeleton

1 — 2 — 3 — 4



2) unshielded triples

3) v- structures?

4) additional orientations?

5) fully oriented CPDAG?

6) 1 and 2?

SGS algorithm (Spirtes, Glymour, Scheines)

- Assuming P is Markov and faithful to an unknown graph G

- We can estimate a CPDAG from samples of P in three steps:
  1. Determine the **skeleton**
  2. Determine the **v-structures** *(given the tests in the previous phase)*
  3. Direct as many remaining edges as possible

- **Computationally inefficient:** potentially $O(2^p)$ tests
  - Even if we reuse the test results from skeleton phase in other phases

PC algorithm (Peter Spirtes, Clark Glymour)

- Assuming P is Markov and faithful to an unknown graph G

- We can estimate a CPDAG from samples of P in three steps:
  1. Determine the **skeleton in an optimised way**
  2. Determine the **v-structures** *(given the results in the previous phase)*
  3. Direct as many remaining edges as possible

PC algorithm skeleton learning

- If i **is not adjacent to** j**,** then they can be d-separated by $\mathrm{Pa}(i)$ or $\mathrm{Pa}(j)$
- Determine the **skeleton in an optimised way**
  - Since we do not know the parents we will use the nodes that are adjacent, $\mathrm{Adj}(i)$ or $\mathrm{Adj}(j)$ in U at a given iteration (*superset*)

PC algorithm skeleton learning

- If i **is not adjacent to** j**,** then they can be d-separated by $\mathrm{Pa}(i)$ or $\mathrm{Pa}(j)$
- Determine the **skeleton in an optimised way**
  - Since we do not know the parents we will use the nodes that are adjacent, $\mathrm{Adj}(i)$ or $\mathrm{Adj}(j)$ in U at a given iteration (*superset*)

1. Start with **completely connected undirected graph** U

PC algorithm skeleton learning

- If i **is not adjacent to** j, then they can be d-separated by $\mathrm{Pa}(i)$ or $\mathrm{Pa}(j)$
- Determine the **skeleton in an optimised way**
  - Since we do not know the parents we will use the nodes that are adjacent, $\mathrm{Adj}(i)$ or $\mathrm{Adj}(j)$ in U at a given iteration (*superset*)

1. Start with **completely connected undirected graph** U
2. For $k = 0,1,2\ldots,p-2$
   - If $i - j$ in U and there exists a set $\mathbf{S} \subseteq \mathrm{Adj}(i)\backslash\{j\}$ **of size at least k**

PC algorithm skeleton learning

- If i **is not adjacent to** j, then they can be d-separated by $\mathrm{Pa}(i)$ or $\mathrm{Pa}(j)$
- Determine the **skeleton in an optimised way**
  - Since we do not know the parents we will use the nodes that are adjacent, $\mathrm{Adj}(i)$ or $\mathrm{Adj}(j)$ in U at a given iteration (*superset*)

1. Start with **completely connected undirected graph** U
2. For $k = 0,1,2\ldots,p-2$
   - If $i - j$ in U and there exists a set $\mathbf{S} \subseteq \mathrm{Adj}(i) \backslash \{j\}$ **of size at least k**
     - If holds $X_i \perp\!\!\!\perp X_j \,|\, \mathbf{S}$, then **remove** $i - j$ **in** U (and add $\mathbf{S}$ to $\mathrm{SepSet}(i,j)$)

PC algorithm skeleton learning

- If i **is not adjacent to** j**,** then they c[illegible] $\mathrm{Pa}(i)$ or $\mathrm{Pa}(j)$
- Determine the **skeleton in an optim**
  - Since we do not know the pare[illegible] adjacent, $\mathrm{Adj}(i)$ or $\mathrm{Adj}(j)$ in U at a given iteration (*superset*)

SepSet(i, j) is the separating set of X_i and X_j where we store the results of the conditional independence tests

We stop when there are no more untested sets of size k in the adjacencies of any variable

1. Start with **completely connected**
2. For $k = 0,1,2\ldots, p-2$
   - If $i - j$ in U and there exists a set $\mathbf{S} \subseteq \mathrm{Adj}(i) \backslash \{j\}$ **of size at least k**
     - If holds $X_i \perp\!\!\!\perp X_j \,|\, \mathbf{S}$, then **remove** $i - j$ **in** U (and add $\mathbf{S}$ to $\mathrm{SepSet}(i, j)$)

PC algorithm skeleton example

SepSet(1,3)=∅

$K=0$ $X_1 \perp\!\!\!\perp X_3$:

$X_1 \not\perp\!\!\!\perp X_2$; $X_3 \not\perp\!\!\!\perp X_2$; $X_1 \not\perp\!\!\!\perp X_4$; $X_3 \not\perp\!\!\!\perp X_4$

PC algorithm skeleton example

SepSet(1,3)=∅

SepSet(3,4)={1,2}

$K=0$ $X_1 \perp\!\!\!\perp X_3$: $K=1$ $K=2$:

$X_1 \not\perp\!\!\!\perp X_2$; $X_3 \not\perp\!\!\!\perp X_2$; $X_1 \not\perp\!\!\!\perp X_4$; $X_3 \not\perp\!\!\!\perp X_4$

$X_3 \perp\!\!\!\perp X_4 \mid X_2\, X_1$

PC algorithm determine v-structures

- A triple of nodes (i, j, k) in a DAG G is a **an unshielded triple** if $i - j, j - k$ and i **is not adjacent to k**, i.e. $i \neq k$, in G

1. Start from the skeleton U from previous step
2. For each unshielded triple (i, j, k) in U, i.e. $i - j, j - k$ **and** $i \neq k$ **in** U
   - Check if $j \notin \text{SepSet}(i, k)$
     - If this is true, $i \rightarrow j \leftarrow k$ **is a v-structure**

PC algorithm skeleton example

$k=0 \quad X_1 \perp\!\!\!\perp X_3$

$k=1$

$k=2 \quad X_3 \perp\!\!\!\perp X_4 \mid X_2, X_1$

$1-2-3$ $\quad X_1 \not\perp\!\!\!\perp X_3 \mid X_2$

$X_1 \not\perp\!\!\!\perp X_3 \mid X_2, X_4$

V-structure

$2 \notin \text{SepSet}(1,3)=\emptyset$

PC algorithm skeleton example

$k=0$ $X_1 \perp\!\!\!\perp X_3$

$k=1$

$k=2$ $X_3 \perp\!\!\!\perp X_4 \mid X_2\, X_1$

1-2-3 $X_1 \not\perp\!\!\!\perp X_3 \mid X_2$

$X_1 \not\perp\!\!\!\perp X_3 \mid X_2, X_4$

V-structure

3-2-4 $X_3 \perp\!\!\!\perp X_4 \mid X_2\, X_1$

Note: 1-4-2 is NOT a v-structure

(1 is adjacent to 2)

$1 \in \text{SepSet}(3,4) = \{1,2\}$

$2 \notin \text{SepSet}(1,3) = \emptyset$

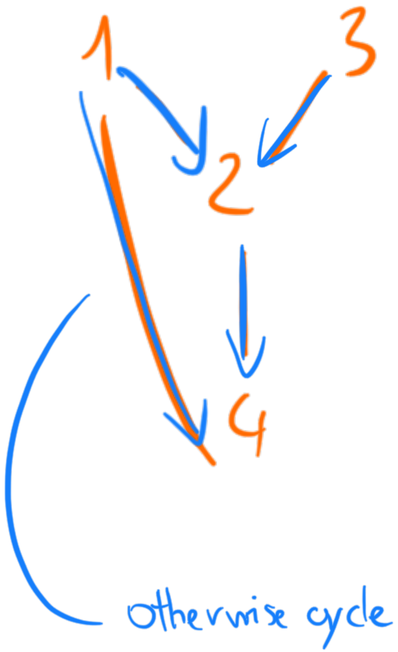
PC algorithm skeleton example

$k=0 \quad X_1 \perp\!\!\!\perp X_3$

$k=2 \quad X_3 \perp\!\!\!\perp X_4 \mid X_2 X_1$

$2 \notin \text{SepSet}(1,3)=\emptyset$

$1 \in \text{SepSet}(3,4)=\{1,2\}$



SGS vs PC

- **SGS:** we can estimate a CPDAG from samples of P in three steps:
  1. Determine the **skeleton**
  2. Determine the **v-structures**
  3. Direct as many remaining edges as possible

- **PC:** we can estimate a CPDAG from samples of P in three steps:
  1. Determine the **skeleton in an optimised way**
  2. Determine the **v-structures in an optimised way**
  3. Direct as many remaining edges as possible

PC algorithm - when does it fail?

- If the conditional independence tests give the wrong result
  - Too few samples
  - A very weak dependence
  - Wrong parametric assumption (e.g. partial correlation on nonlinear data)

If you're curious, you can see here some variants (like conservative PC) that try to circumvent the problem by doing more tests https://rdrr.io/cran/pcalg/man/pc.html

PC algorithm - when does it fail?

- If the conditional independence tests give the wrong result
  - Too few samples
  - A very weak dependence
  - Wrong parametric assumption (e.g. partial correlation on nonlinear data)

- If there are unmeasured confounders or selection bias
  - For example Chocolate - Nobel prizes

PC algorithm - when does it fail?

- If the conditional independence tests give the wrong result
  - Too few samples
  - A very weak dependence
  - Wrong parametric assumption (e.g. partial correlation on nonlinear data)

- If there are unmeasured confounders or selection bias **(causal sufficiency)**

- More advanced algorithms like Fast Causal Inference (FCI)
  - Chapter 6 in [SGS] Causation Prediction and Search
    - https://www.researchgate.net/publication/242448131_Causation_Prediction_and_Search

PC algorithm - when does it fail?

- If the conditional independence tests give the wrong result
  - Too few samples
  - A very weak dependence
  - Wrong parametric assumption (e.g. partial correlation on nonlinear data)

- If there are unmeasured confounders or selection bias **(causal sufficiency)**

Main advantage of constraint-based methods

- More advanced algorithms like Fast Causal Inference (FCI)
  - Chapter 6 in [SGS] Causation Prediction and Search
    - https://www.researchgate.net/publication/242448131_Causation_Prediction_and_Search

Next class:

Constraint-based causal discovery

- Conditional independence tests
- Observational data
- Output: MEC
- SGS, PC, FCI

Score-based causal discovery

- Penalised likelihood
- Observational data
- Output: MEC
- GES, MMHC

Restricted models

- Nonlinear additive noise, Linear Non-Gaussianity
- Observational data
- Output: DAG
- RESIT, LINGAM

Interventional causal discovery / causal invariance

- Observational and Interventional data
- Output: parents of Y, I-MEC
- ICP, GIES, JCI