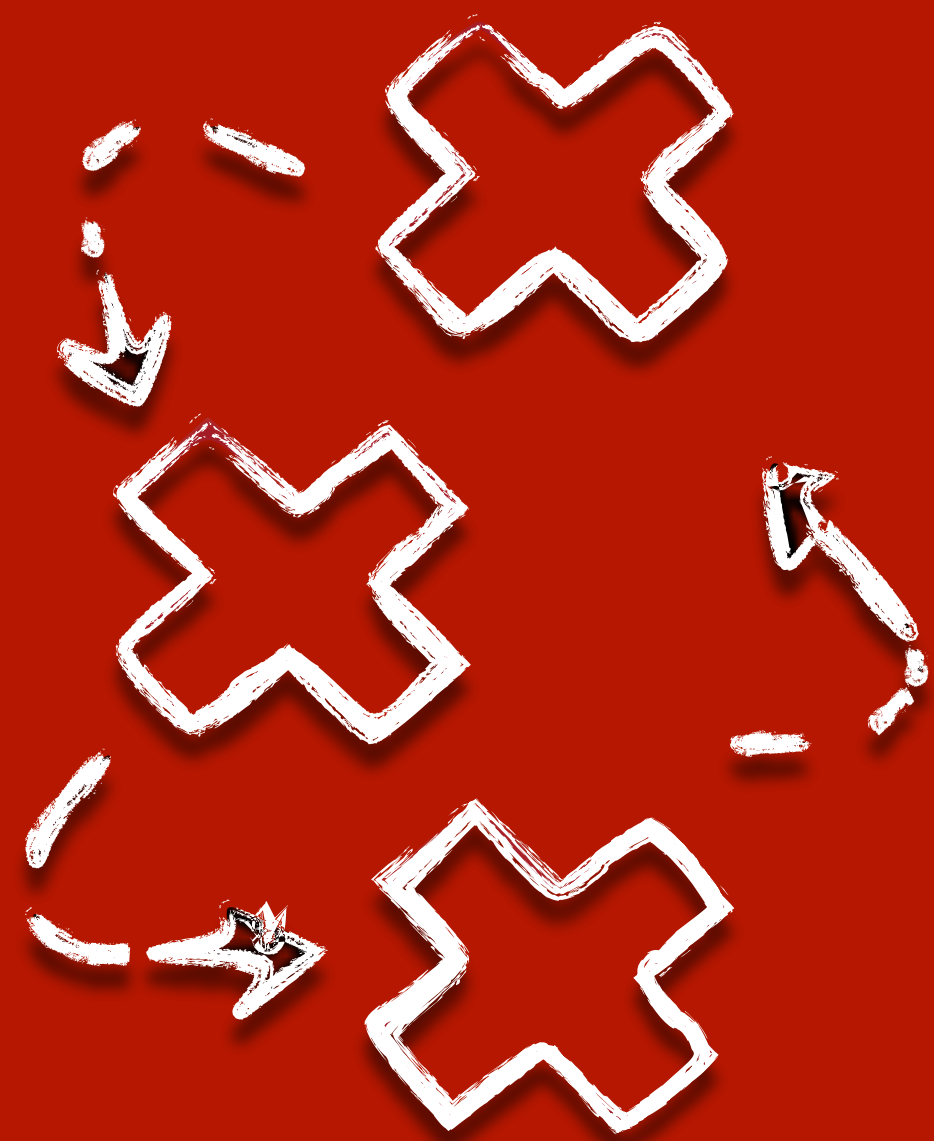




Seminar: Advanced topics in causality and causal ML research

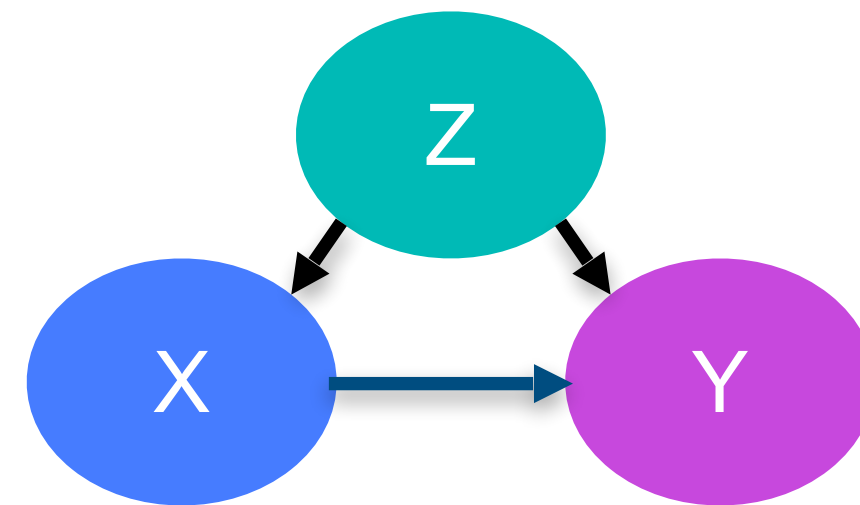
Lecture 2: Causal Discovery

Lecturer: Sara Magliacane

UdS - Summer 2026

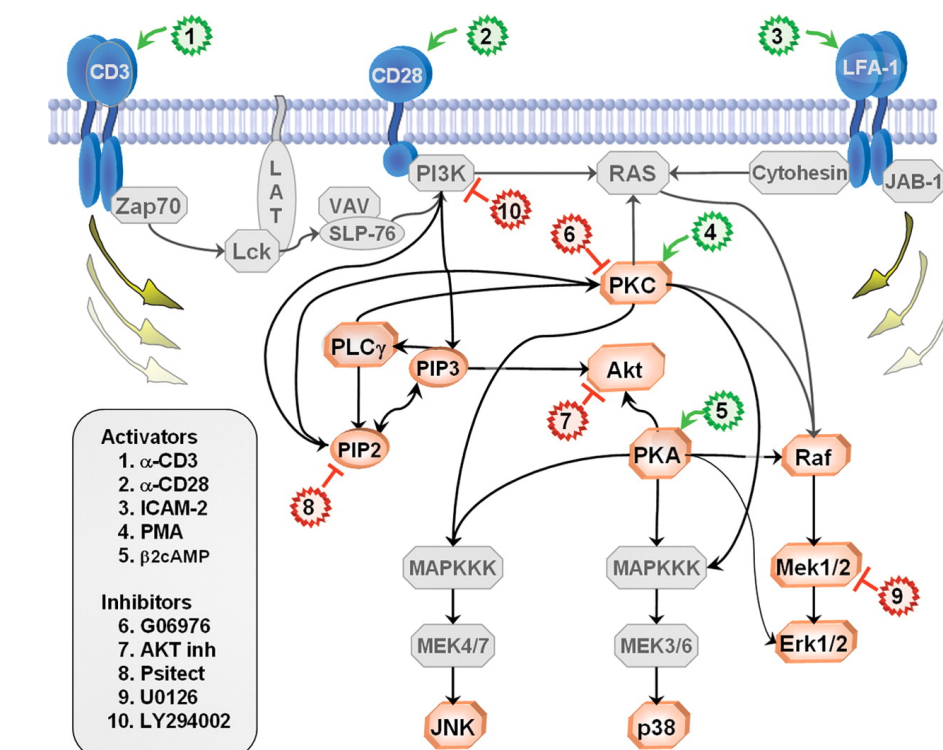
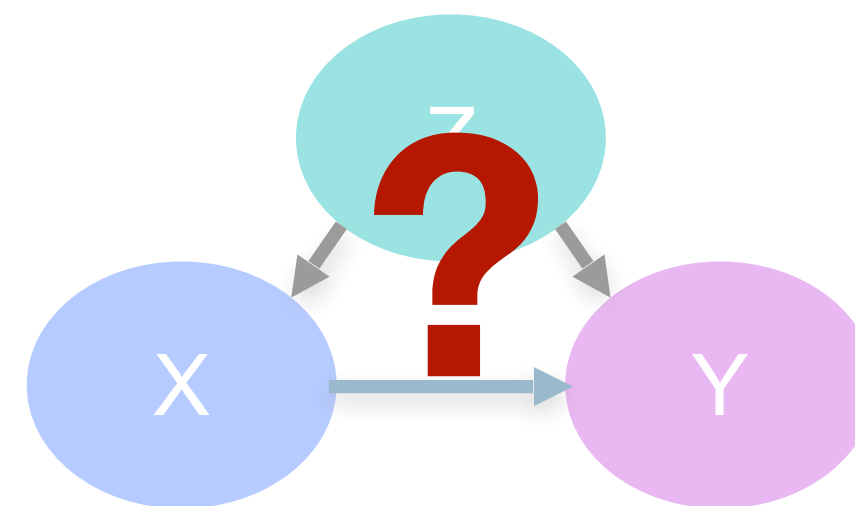
Causal discovery - learning causal relations from data

- In causal inference, we often assume that we know that X causes Y , and that Z is valid adjustment set - the task is to estimate **how much**



Causal discovery - learning causal relations from data

- In causal inference, we often assume that we know that X causes Y , and that Z is valid adjustment set - the task is to estimate **how much**



- What if don't know (all of) these causal relations?
 - E.g. we don't know if Z is a valid adjustment set, or even if X causes Y ?
- Causal discovery** is the task of learning causal relations from (obs.) **data**

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

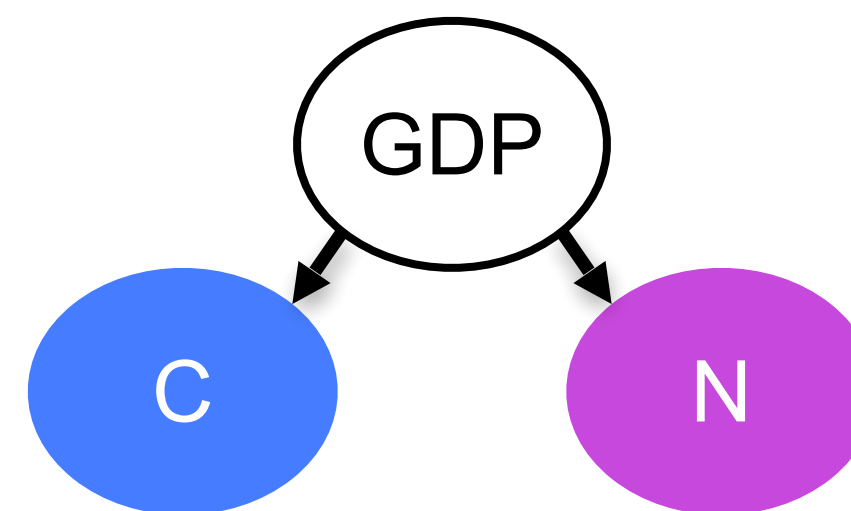
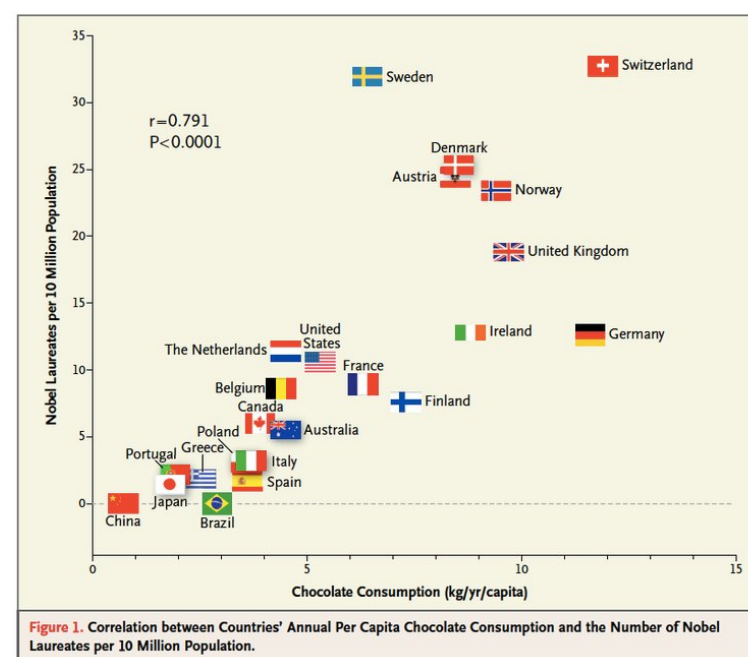
- Observational and Interventional data
- Output: parents of Y, I-MEC
- ICP, GIES, JCI

Common assumptions

- If P is **Markov** and **faithful** to G , then for any disjoint $\mathbf{A}, \mathbf{B}, \mathbf{C} \subseteq \mathbf{V}$:

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

- Causal sufficiency** - no **latent** confounders (common causes), no selection bias



Common assumptions

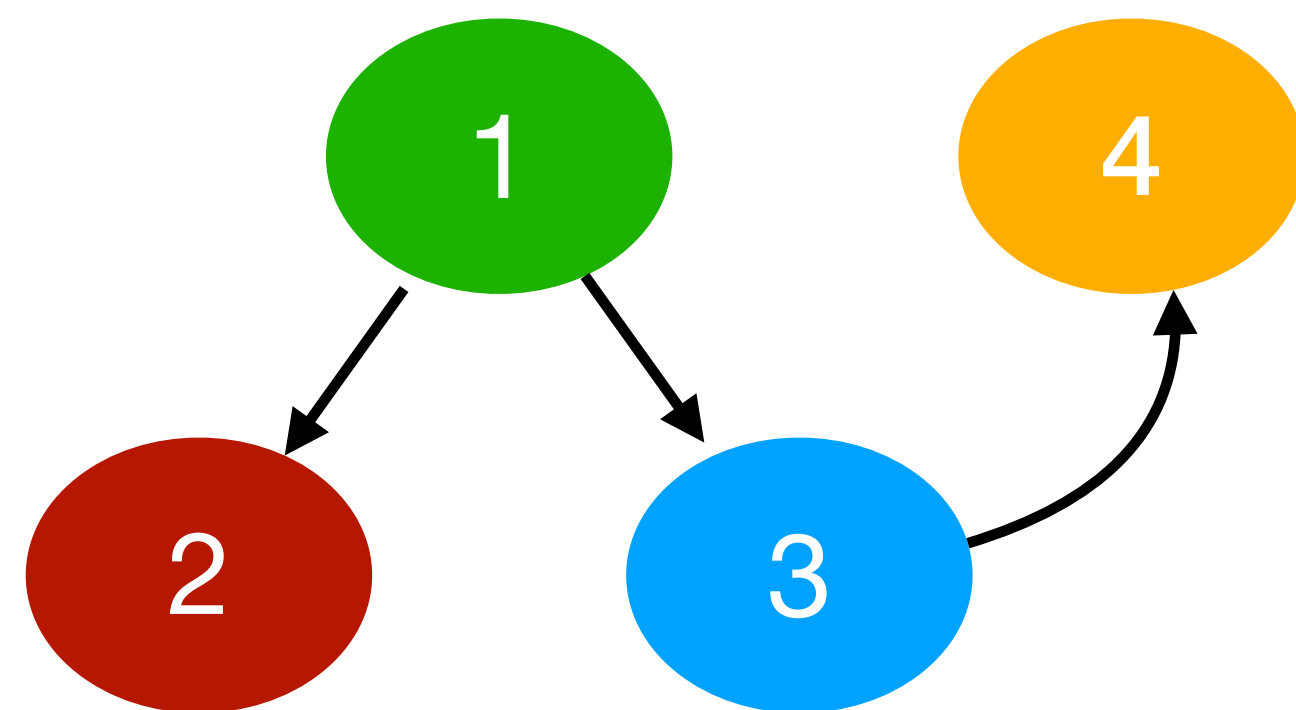
- If P is **Markov** and **faithful** to G , then for any disjoint $\mathbf{A}, \mathbf{B}, \mathbf{C} \subseteq \mathbf{V}$:

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

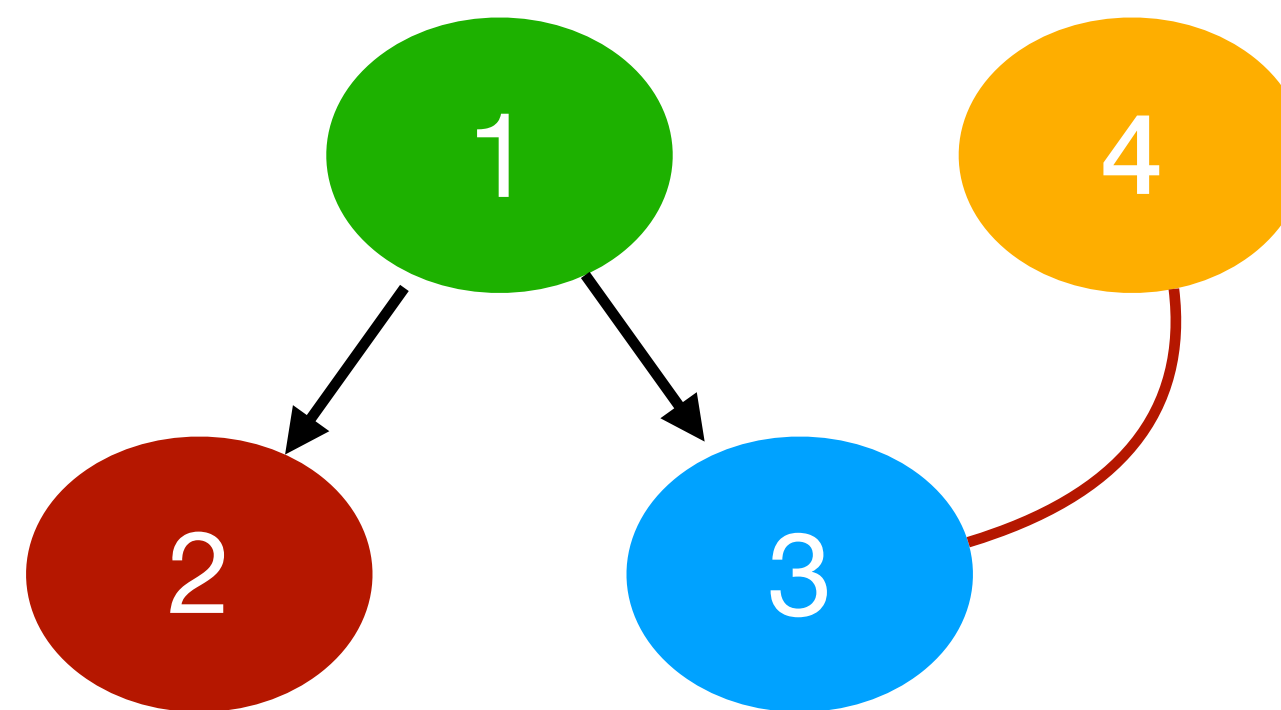
- **Causal sufficiency** - no latent confounders (common causes), no selection bias
- **Acyclicity** - the graph is acyclic, i.e., it is a **Directed Acyclic Graph (DAG)**
- (... We will not discuss it in detail, but there are methods also for cycles + causal insufficiency, e.g., sigma separation, Joint Causal Inference)

Recap: Directed vs mixed vs undirected graphs

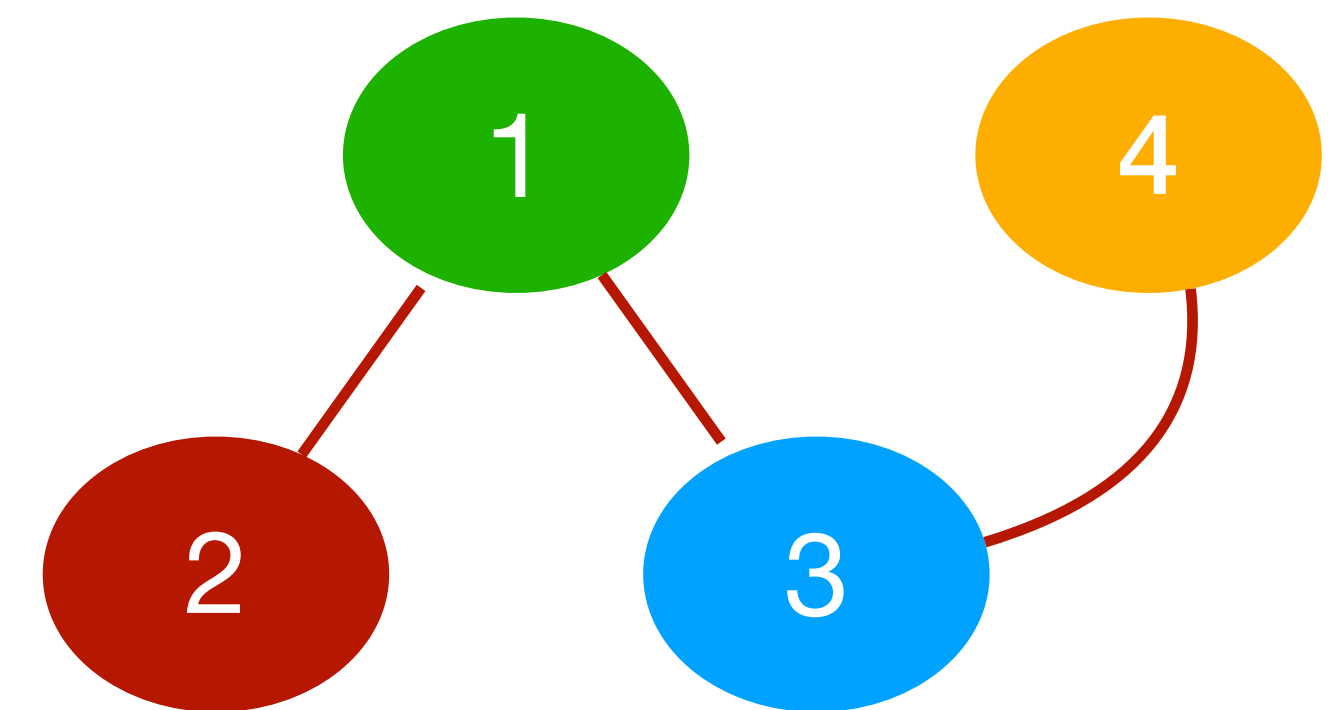
- A graph G is a tuple $G = (\mathbf{V}, \mathbf{E})$:
 - $\mathbf{V}$ is the set of **nodes** (vertices)
 - $\mathbf{E}$ is the set of **edges** between two nodes, i.e. $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$
 - If all edges are **directed** $\rightarrow$ then the graph is **directed**



Directed graph



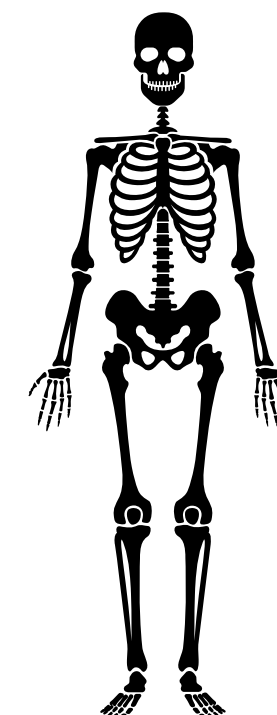
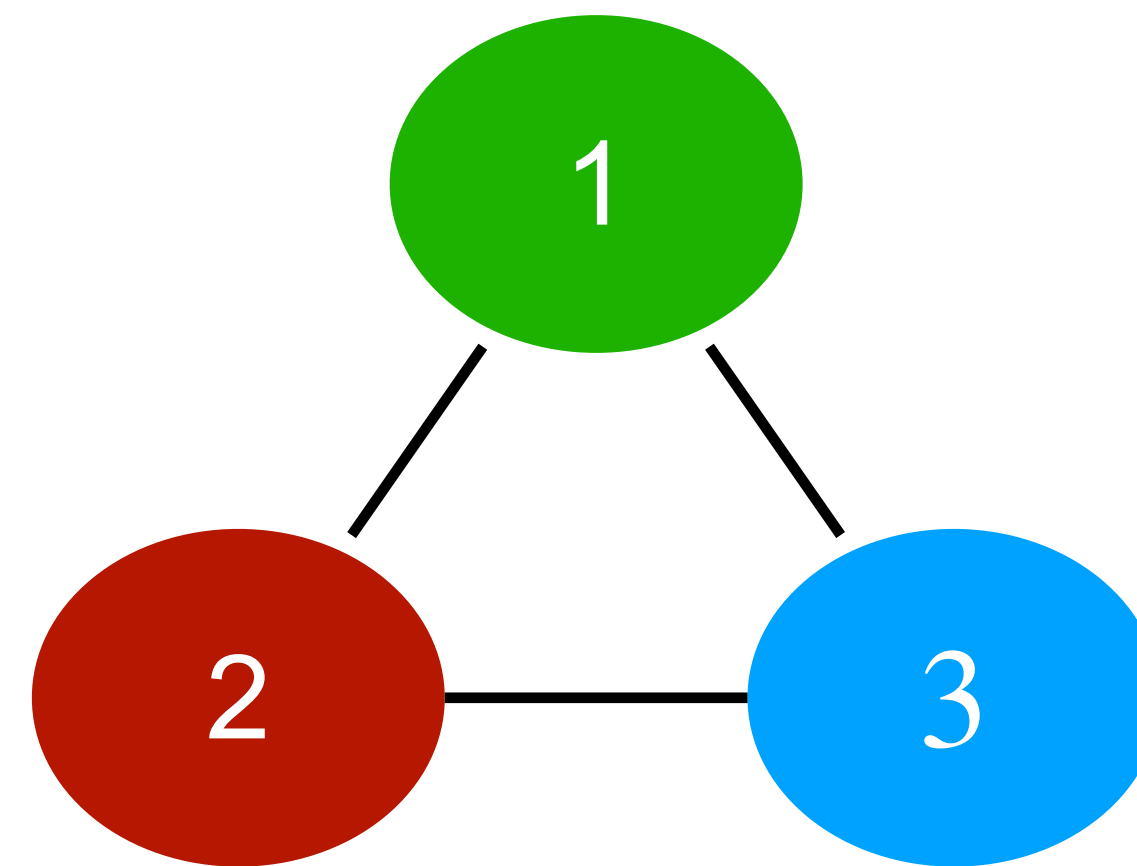
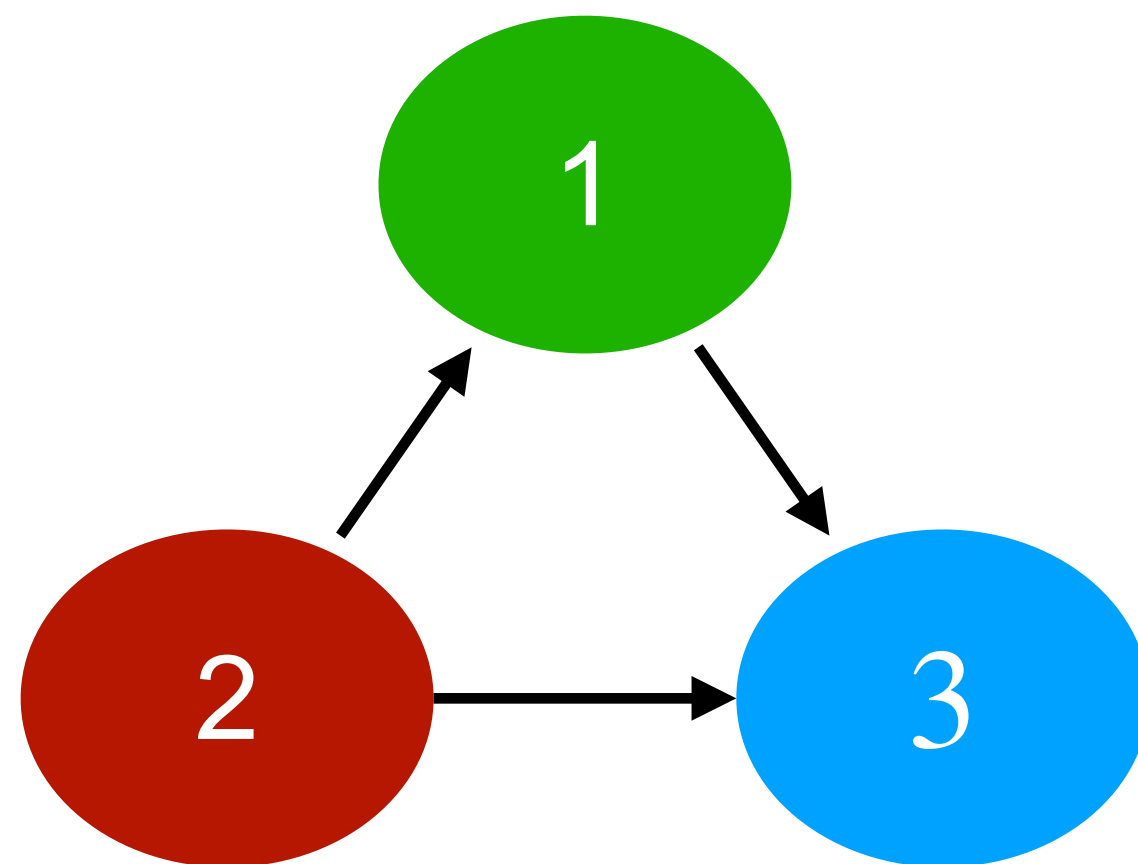
Mixed graph



Undirected graph

Graph terminology: skeletons

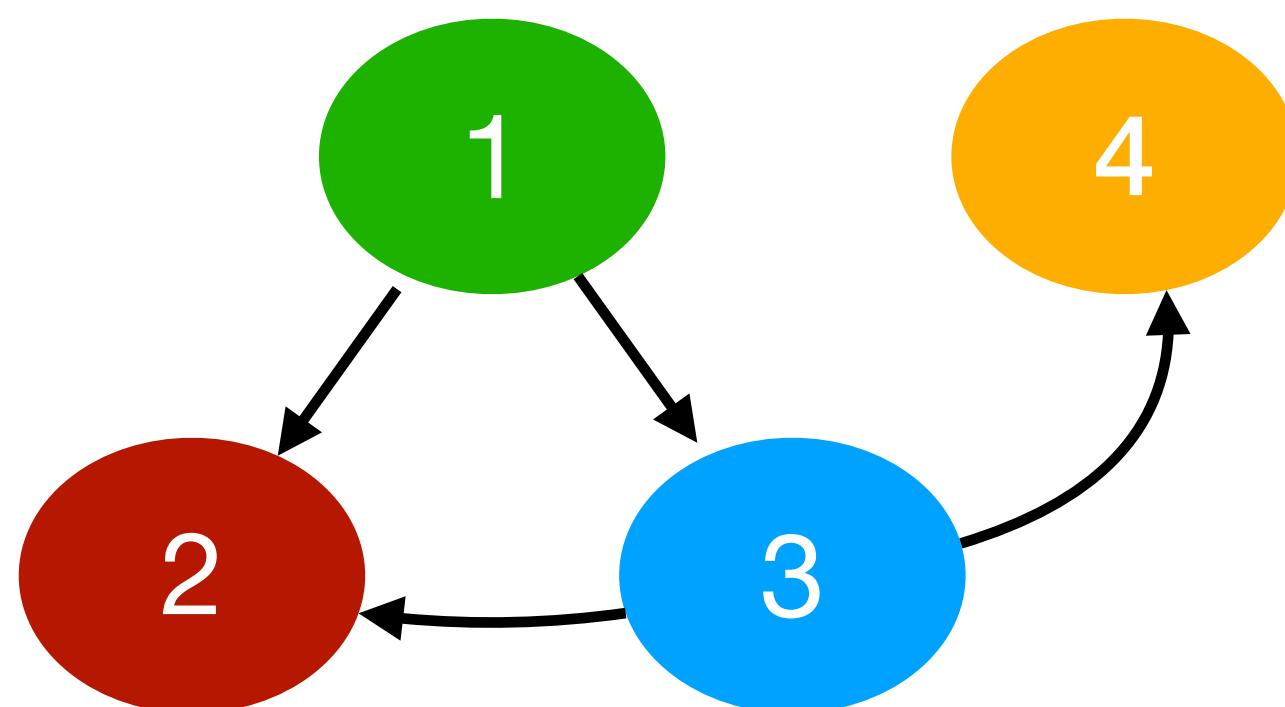
- The skeleton of a DAG $G = (\mathbf{V}, \mathbf{E})$ is the undirected graph $U = (\mathbf{V}, \mathbf{E}')$ that has **an undirected edge** $(i, j) \in \mathbf{E}'$ for every **directed edge** $i \rightarrow j \in \mathbf{E}$ (and no other edges)



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

Directed graphs: paths vs directed paths

- A **path** between **node i and j** is a sequence of **distinct nodes** $(i, \dots, j)$ such that each two **consecutive nodes** are **adjacent (connected)**
- A **directed path** between **node i and node j** is a path where **all edges point towards j**, i.e. $i \rightarrow \dots \rightarrow j$



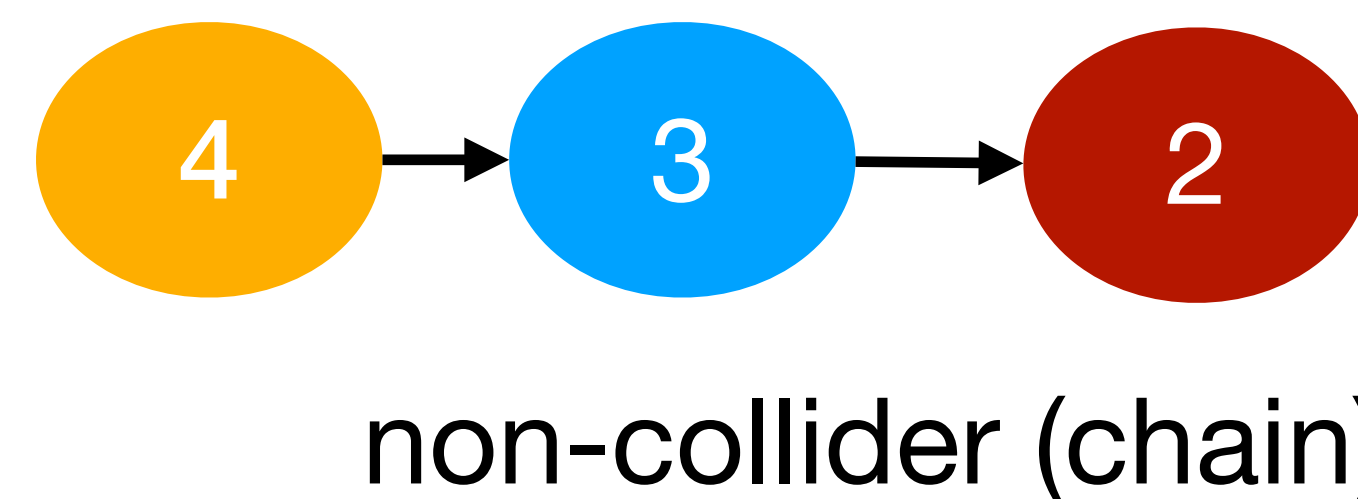
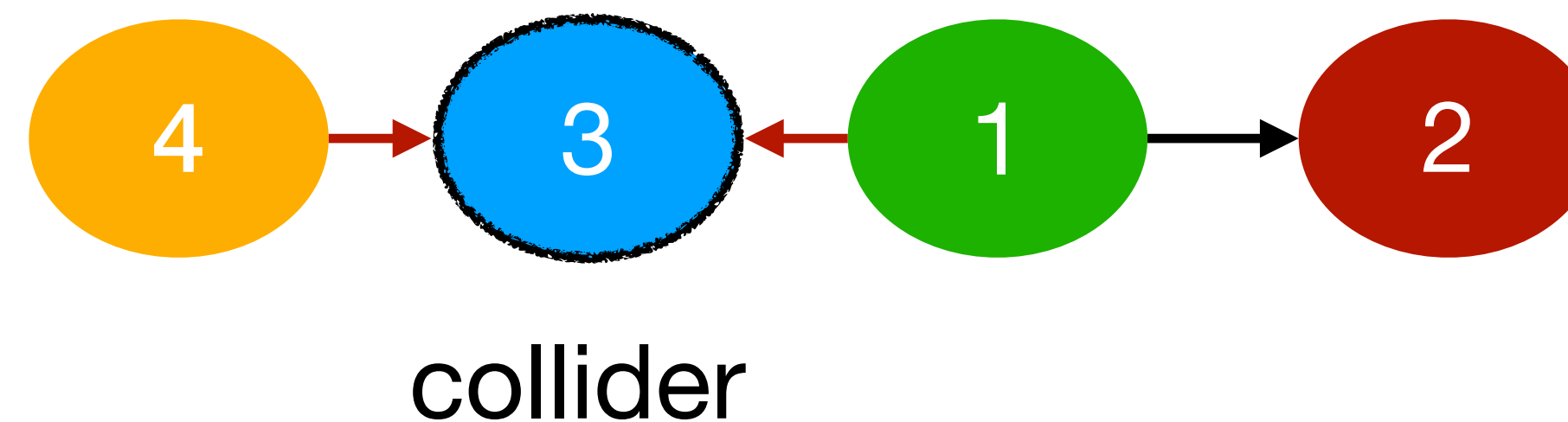
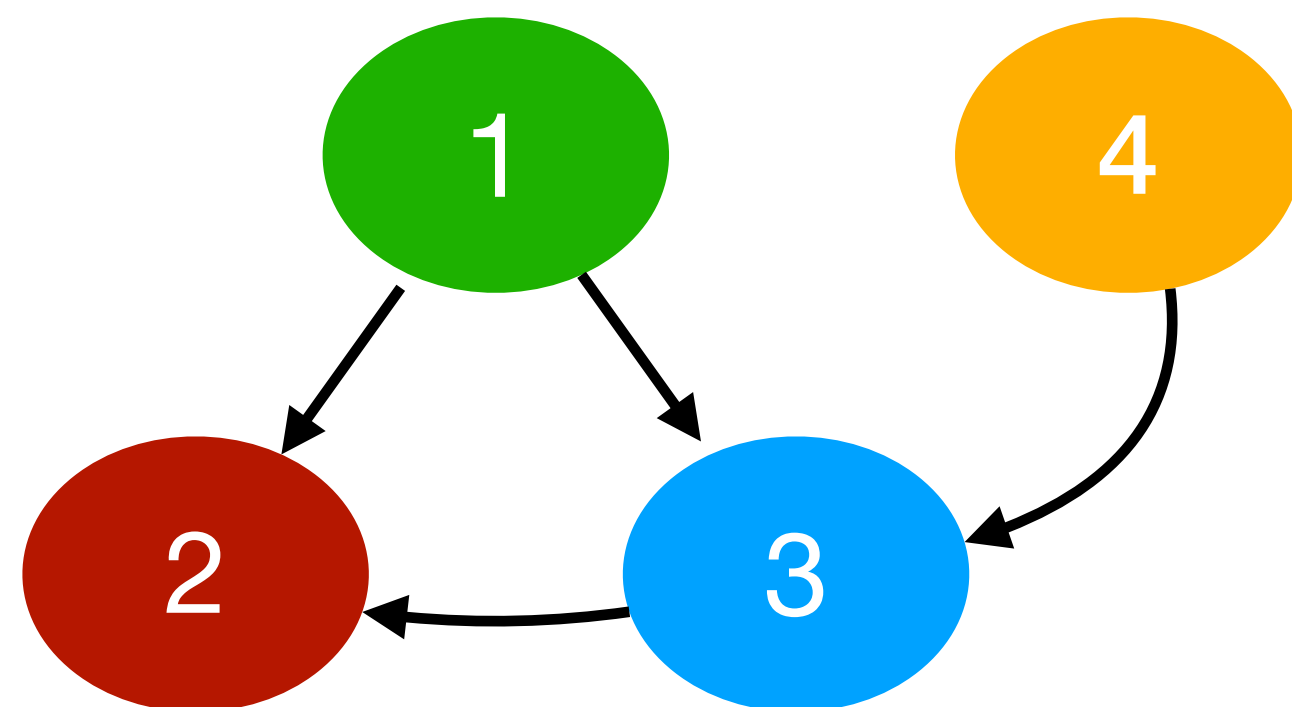
Directed path from 1 to 4



A path from 4 to 1, but not a directed path

Graph terminology: collider on a path

- A **collider** k on a **path** $\pi = (i, \dots, j)$ is a non-endpoint node ($k \neq i, j$) s.t. the path π contains $\rightarrow k \leftarrow$, nodes without this pattern are **non-colliders**



More graph terminology: v-structures/immoralities

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

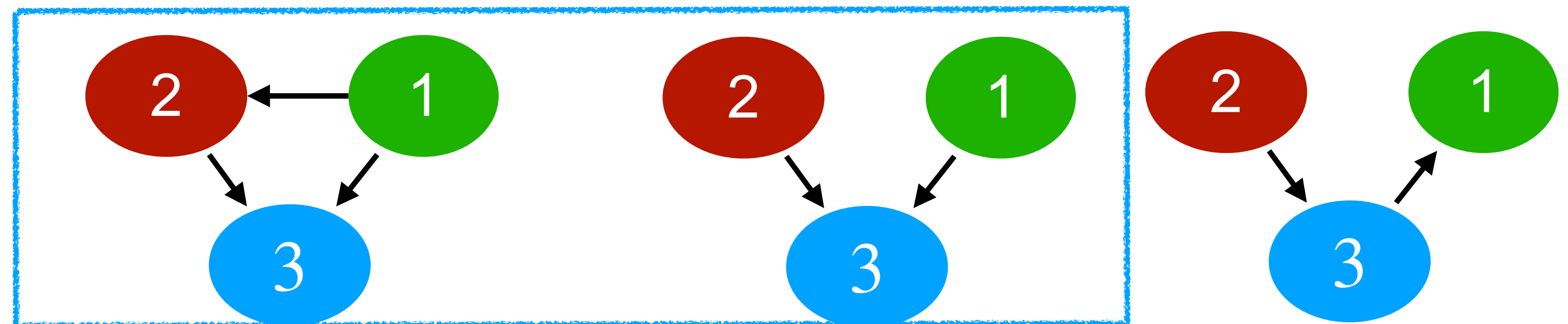
This edge is called a shield

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 $i \rightarrow j \leftarrow k$ in G and i is not adjacent to k

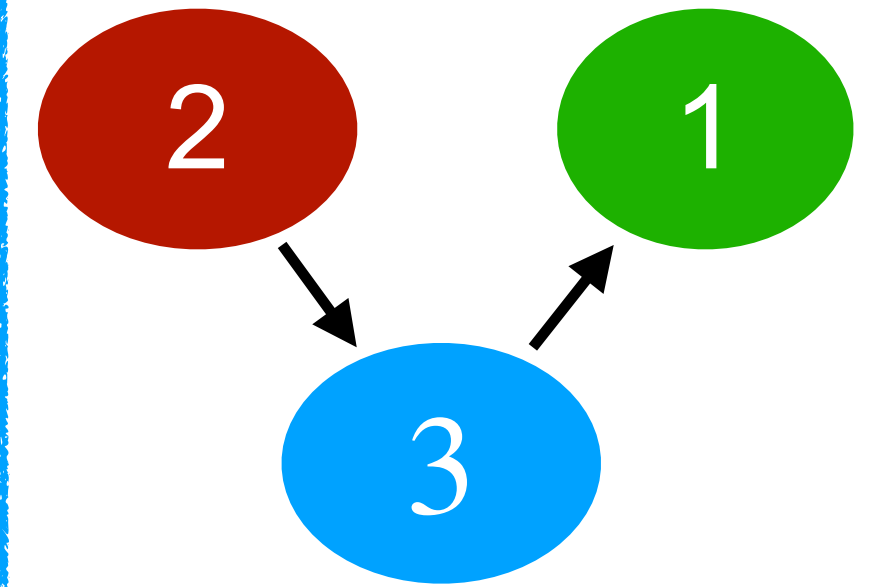
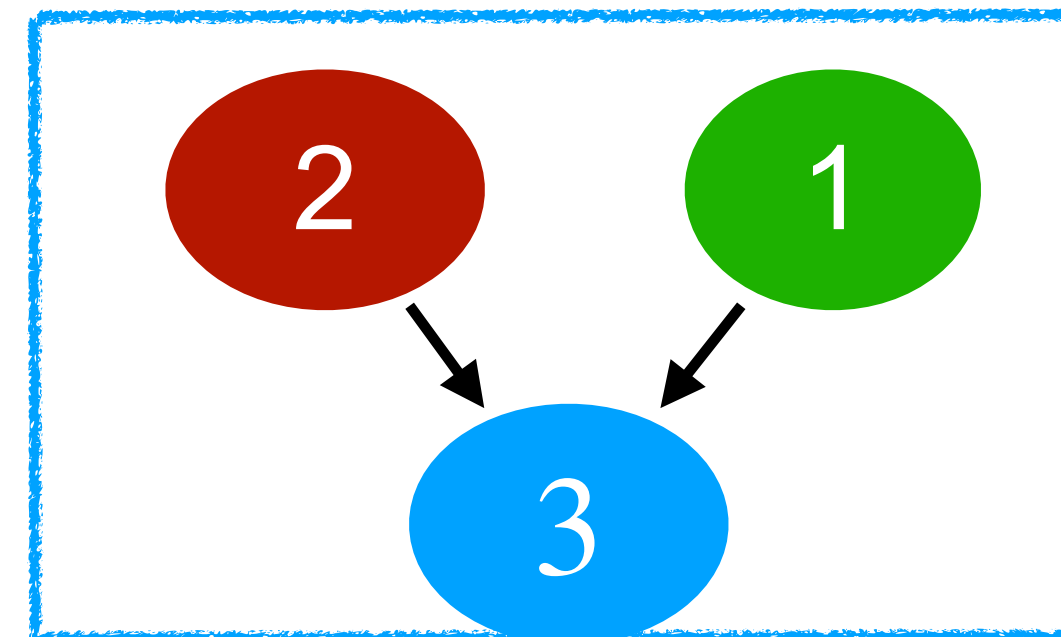
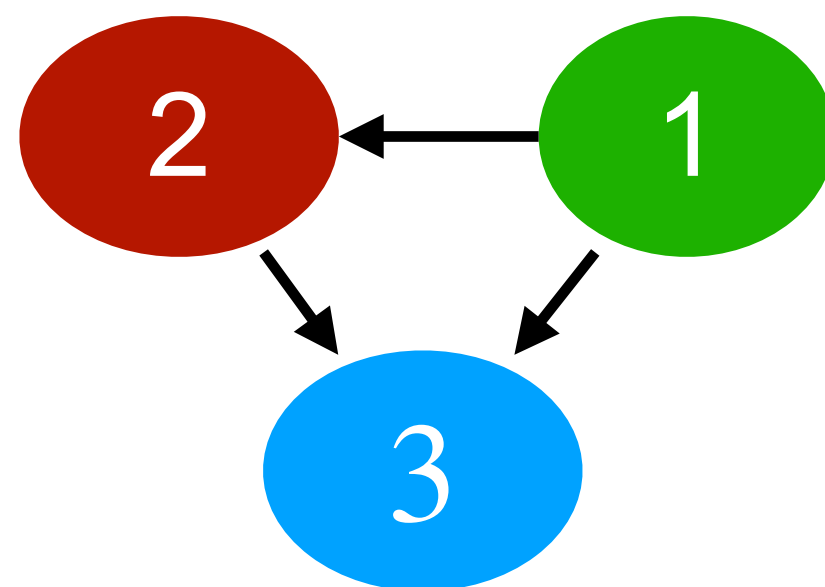
In which graphs is 3 a collider on the path $(2,3,1)$?



More graph terminology: v-structures/immoralities

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

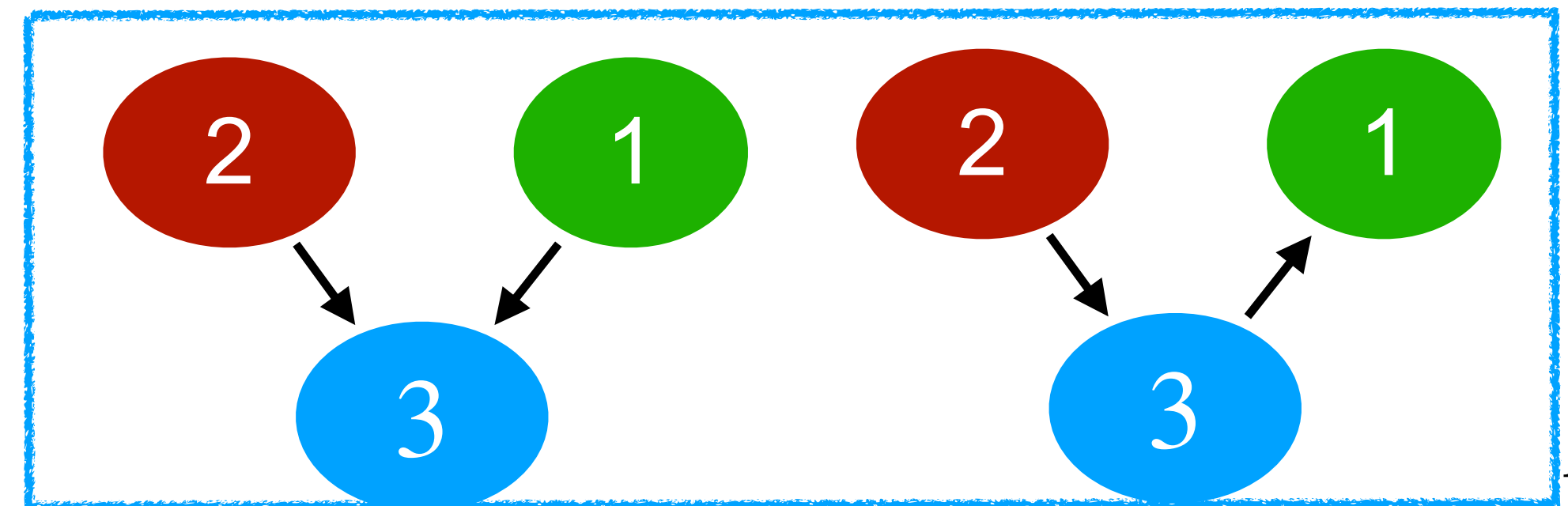
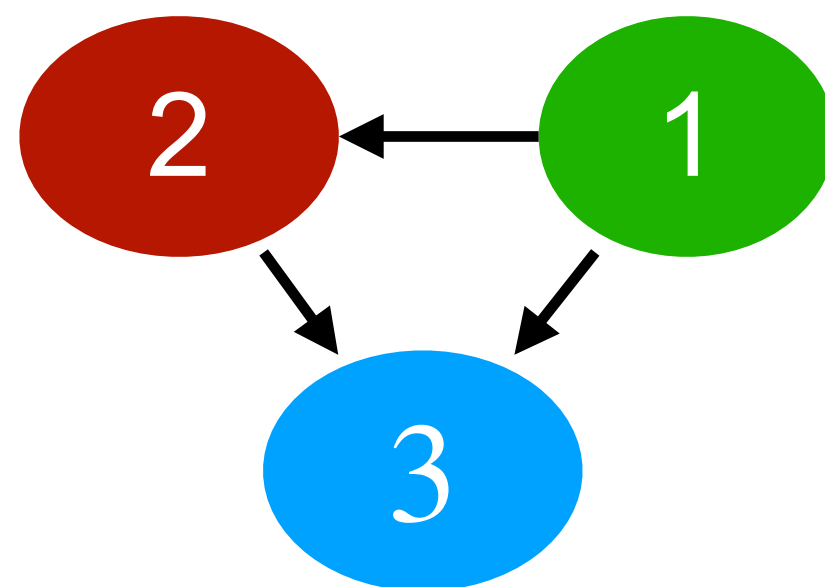
In which graphs is 3 part of a v-structure (2,3,1)?



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 ... this will be a “candidate v-structure”

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

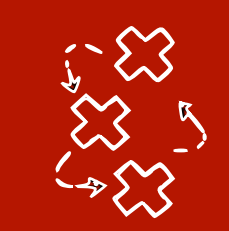


Markov equivalence class and CPDAGs

- (Verma and Pearl 1990) show that all DAGs in a **Markov equivalence class (MEC)**, i.e., all graphs with the same d-separations/conditional independences, have the **same skeleton** and the **same v-structures**
- We represent the orientations (edge marks) all DAGs in a 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 (MEC), i.e., all graphs with the same d-separations/conditional independences, have the **same skeleton** and the **same v-structures**
- We represent the orientations (edge marks) all DAGs in a 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 $i \rightarrow j$ 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 $j \rightarrow i$

$$X \rightarrow Z \rightarrow Y$$

$$\begin{aligned} X \not\perp_d Y \quad \checkmark \\ X \perp_d Y | Z \quad \checkmark \end{aligned}$$

$$X \leftarrow Z \rightarrow Y$$

$$\begin{aligned} X \not\perp_d Y \quad \checkmark \\ X \perp_d Y | Z \quad \checkmark \end{aligned}$$

$$X \leftarrow Z \leftarrow Y$$

$$\begin{aligned} X \not\perp_d Y \quad \checkmark \\ X \perp_d Y | Z \quad \checkmark \end{aligned}$$

$$X - Z - Y$$

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)**
- 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** *(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)

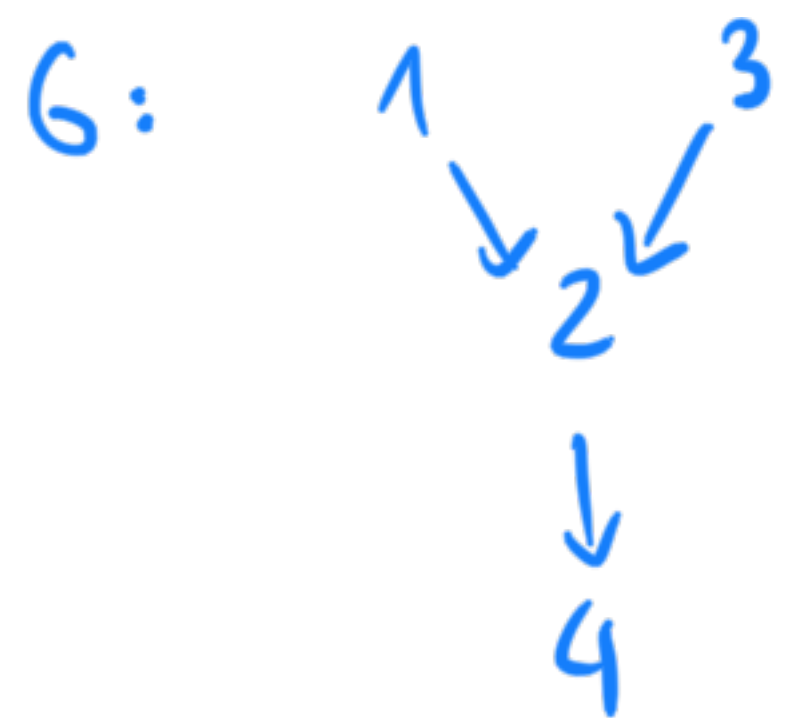
Step 1: Skeleton learning

- 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 $\mathbf{S} \subseteq \mathbf{V} \setminus \{i, j\}$
 - Check **if $X_i \perp\!\!\!\perp X_j \mid X_{\mathbf{S}}$** for any $\mathbf{S}$ in data
 - If this is true, by faithfulness $i \perp_G j \mid \mathbf{S}$, so we can **remove $i - j$ in U**

Step 1: Skeleton learning - example

True causal graph

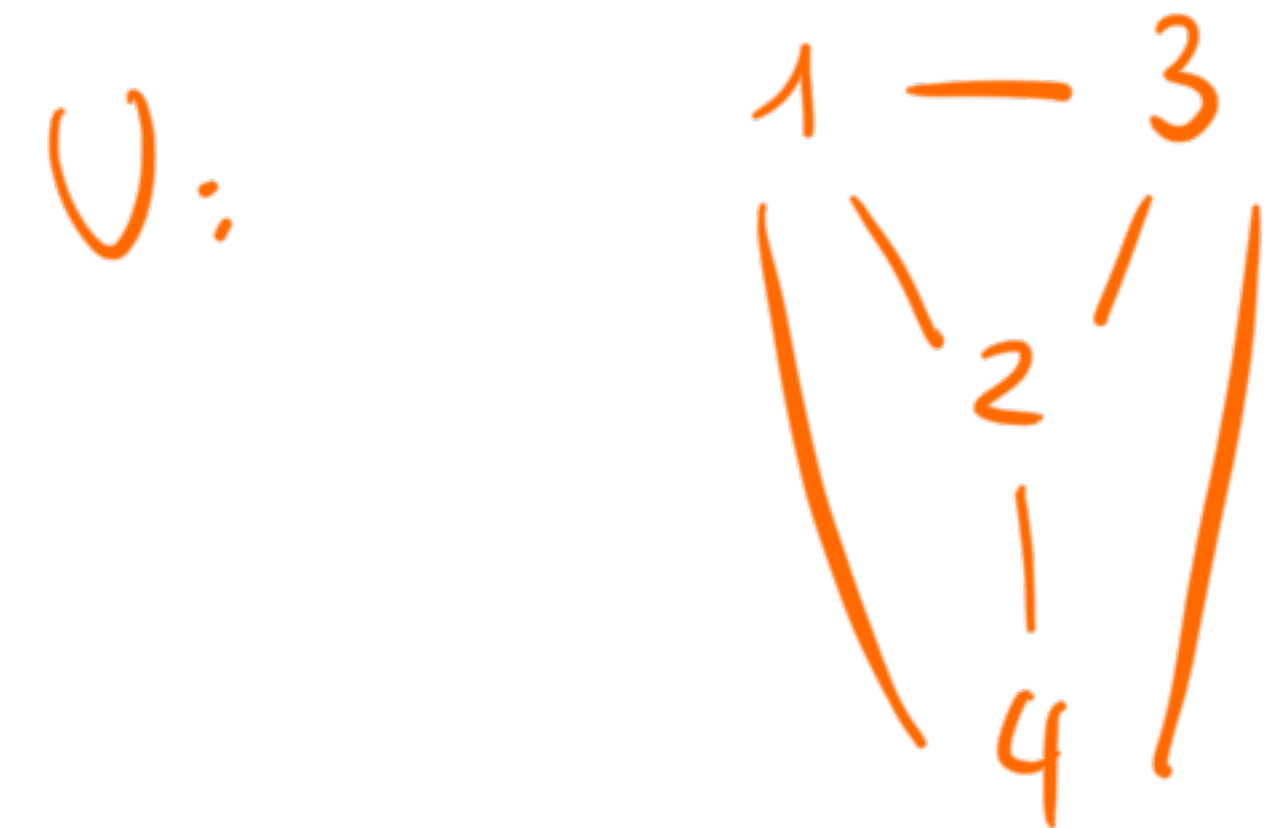


P: $P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

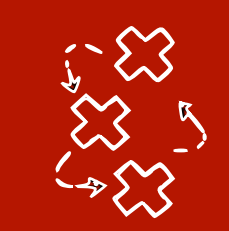
Data

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

Algorithm output

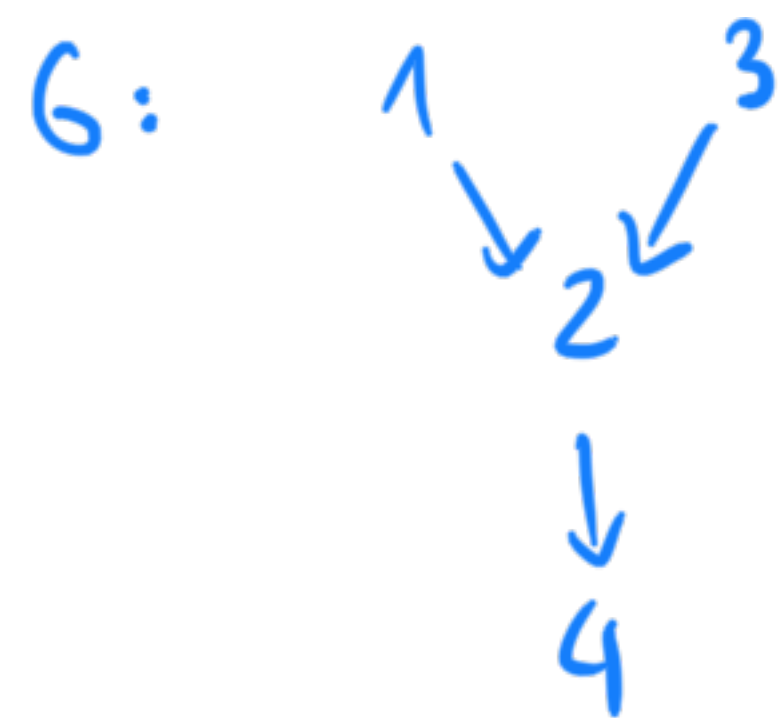


For this example, we will pretend we do not know the true graph, but only the CIs that we can derive from data



Step 1: Skeleton learning - example

True causal graph



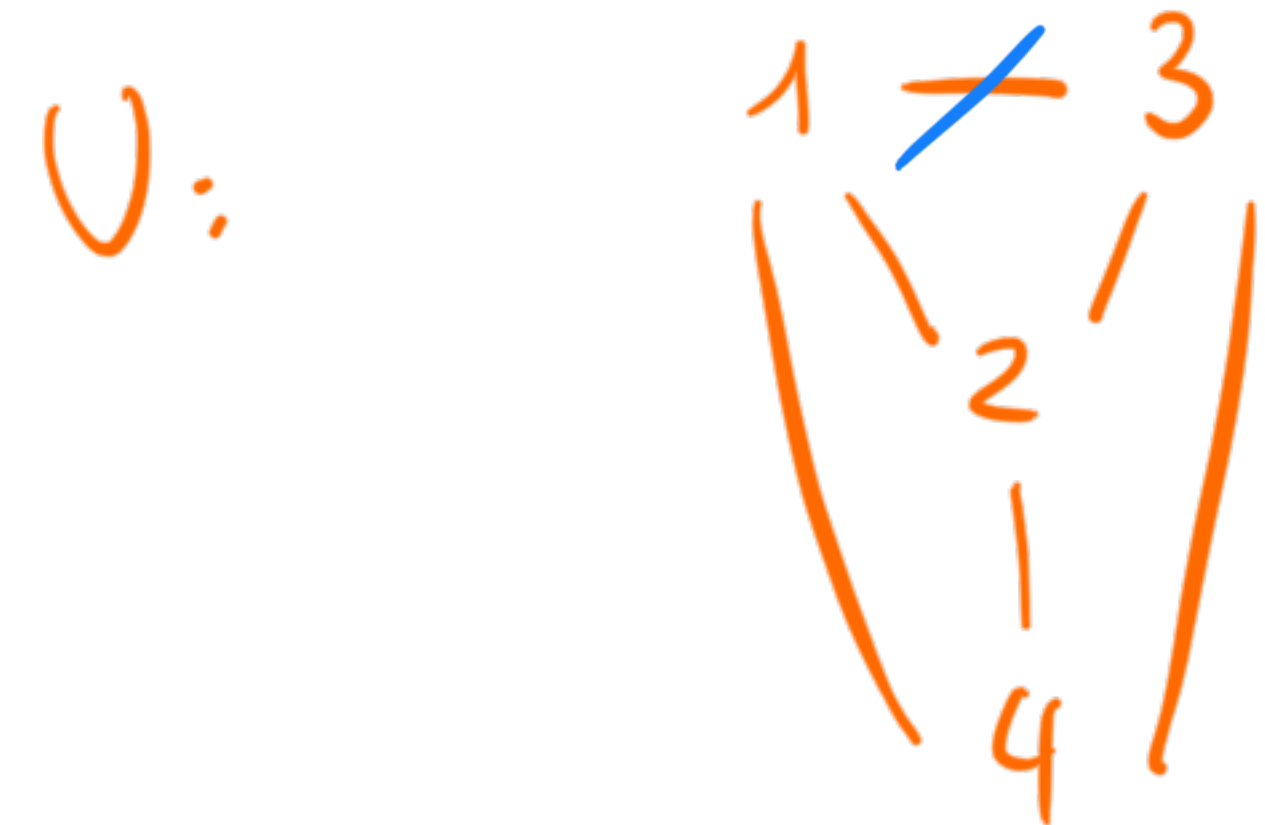
P: $P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

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

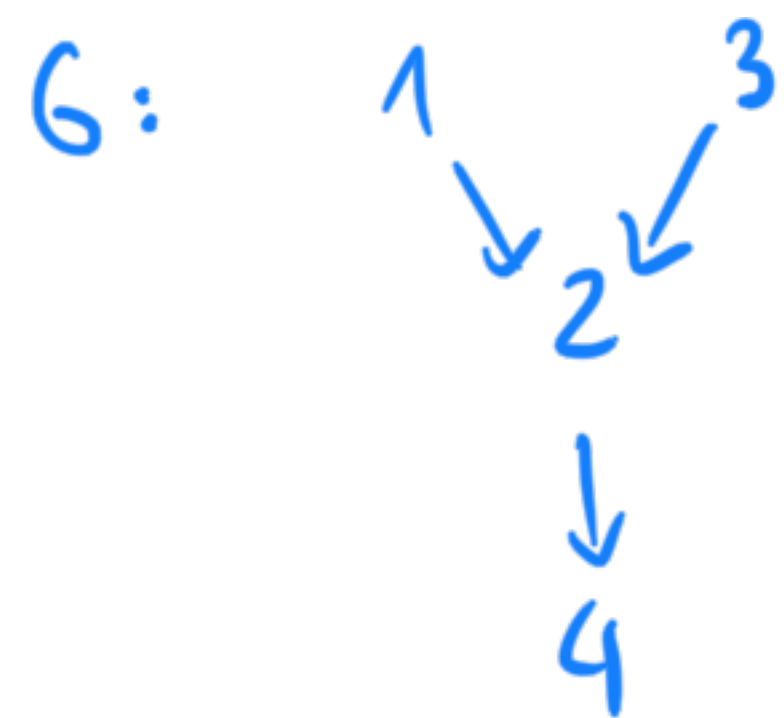
$X_1 \perp\!\!\!\perp X_3$

Algorithm output



Step 1: Skeleton learning - example

True causal graph



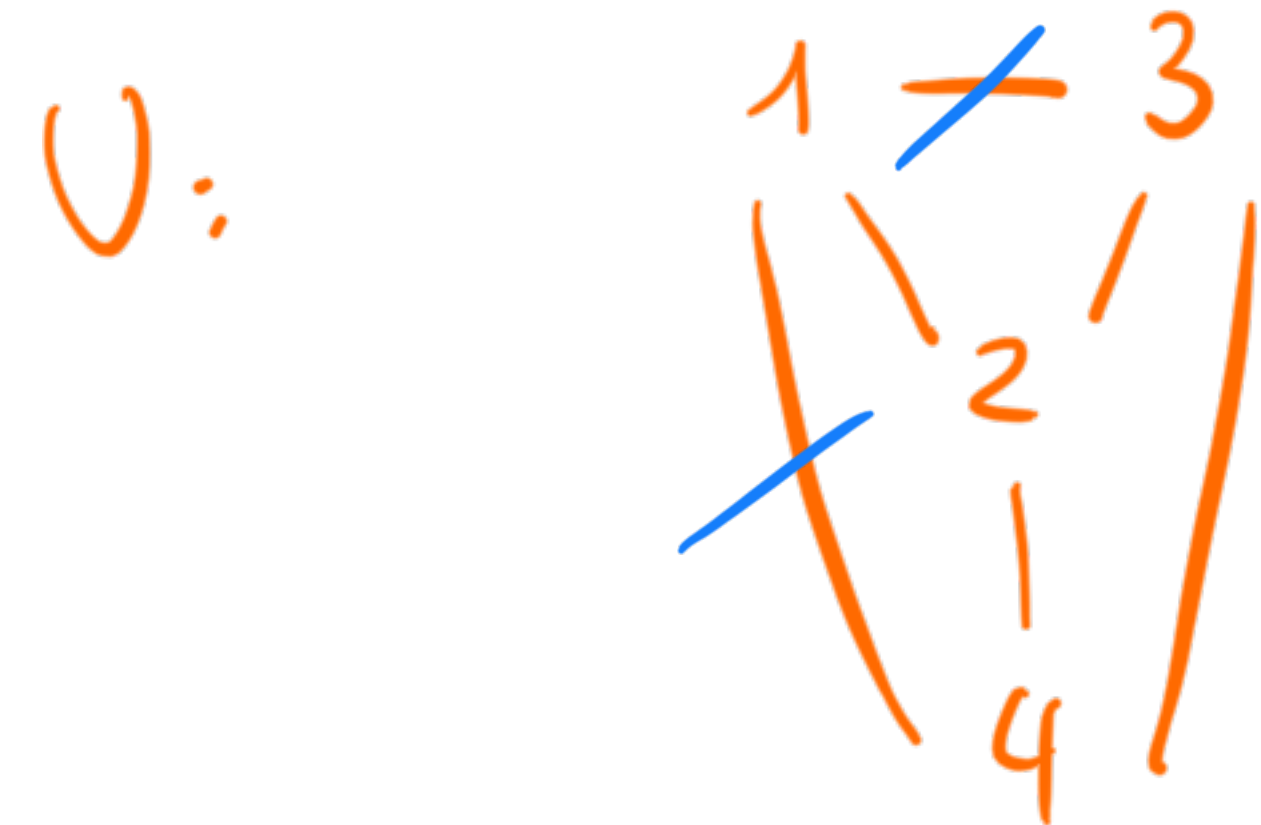
P: $P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

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

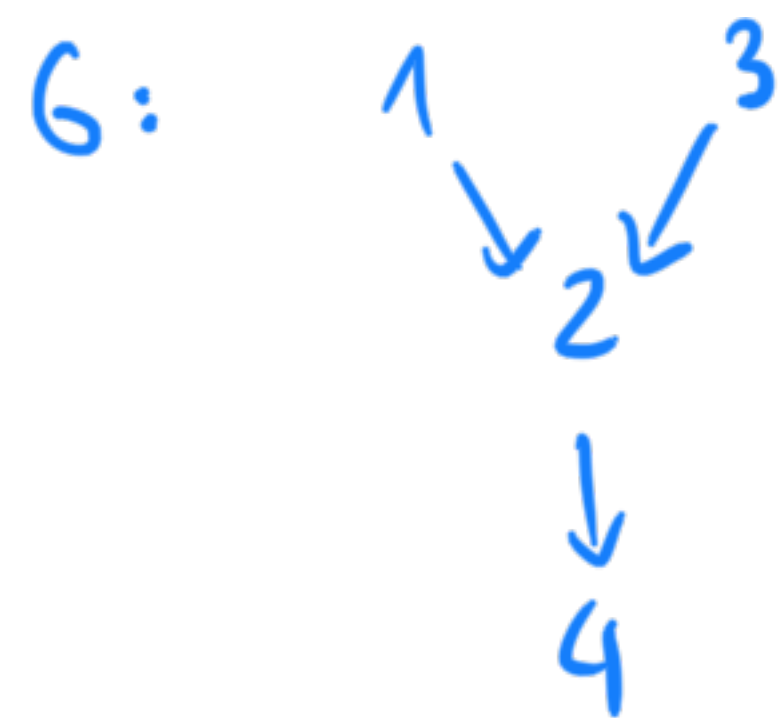
$X_1 \perp\!\!\!\perp X_4 | X_2$

Algorithm output



Step 1: Skeleton learning - example

True causal graph



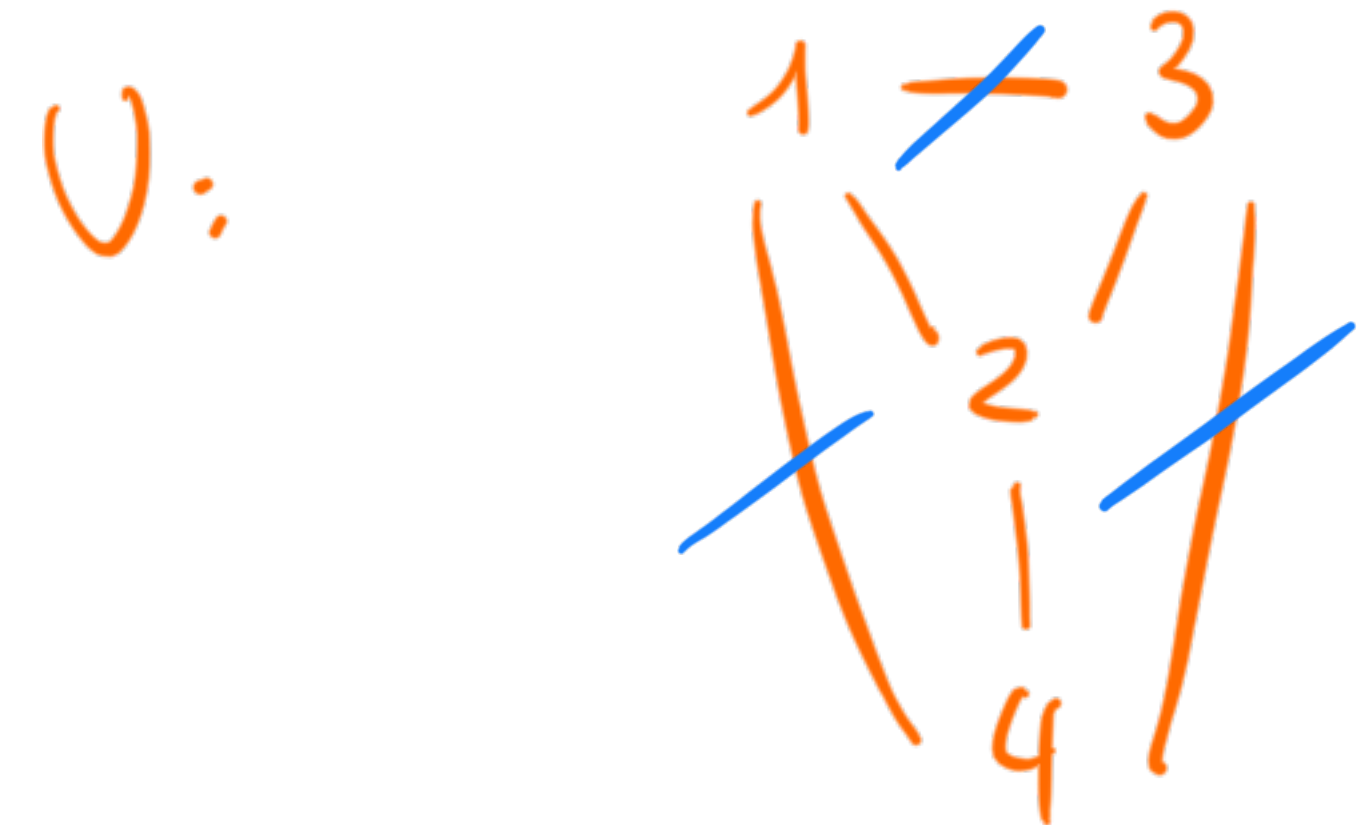
P: $P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

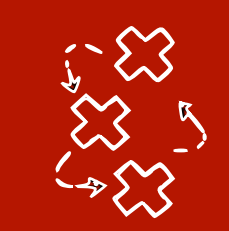
Data

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

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

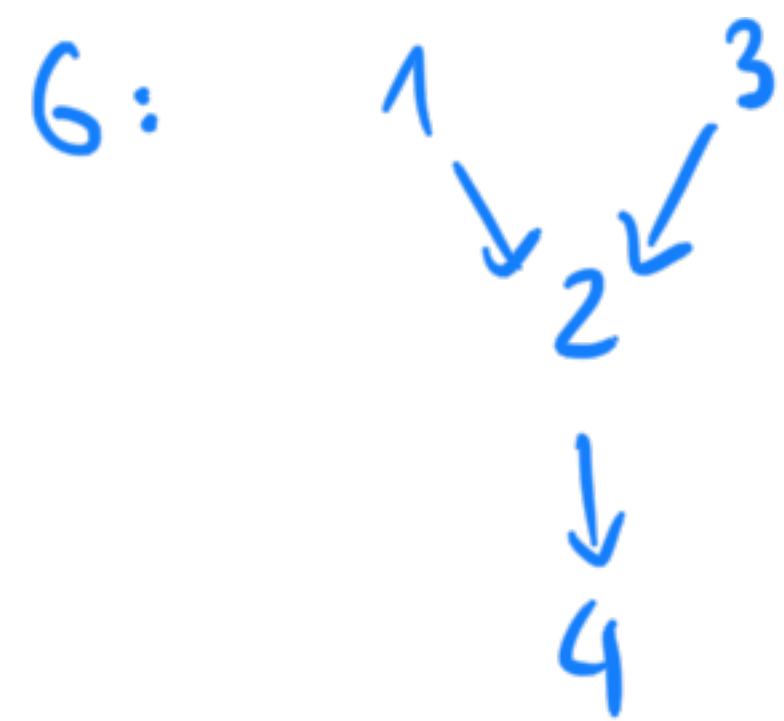
Algorithm output





Step 1: Skeleton learning - example

True causal graph



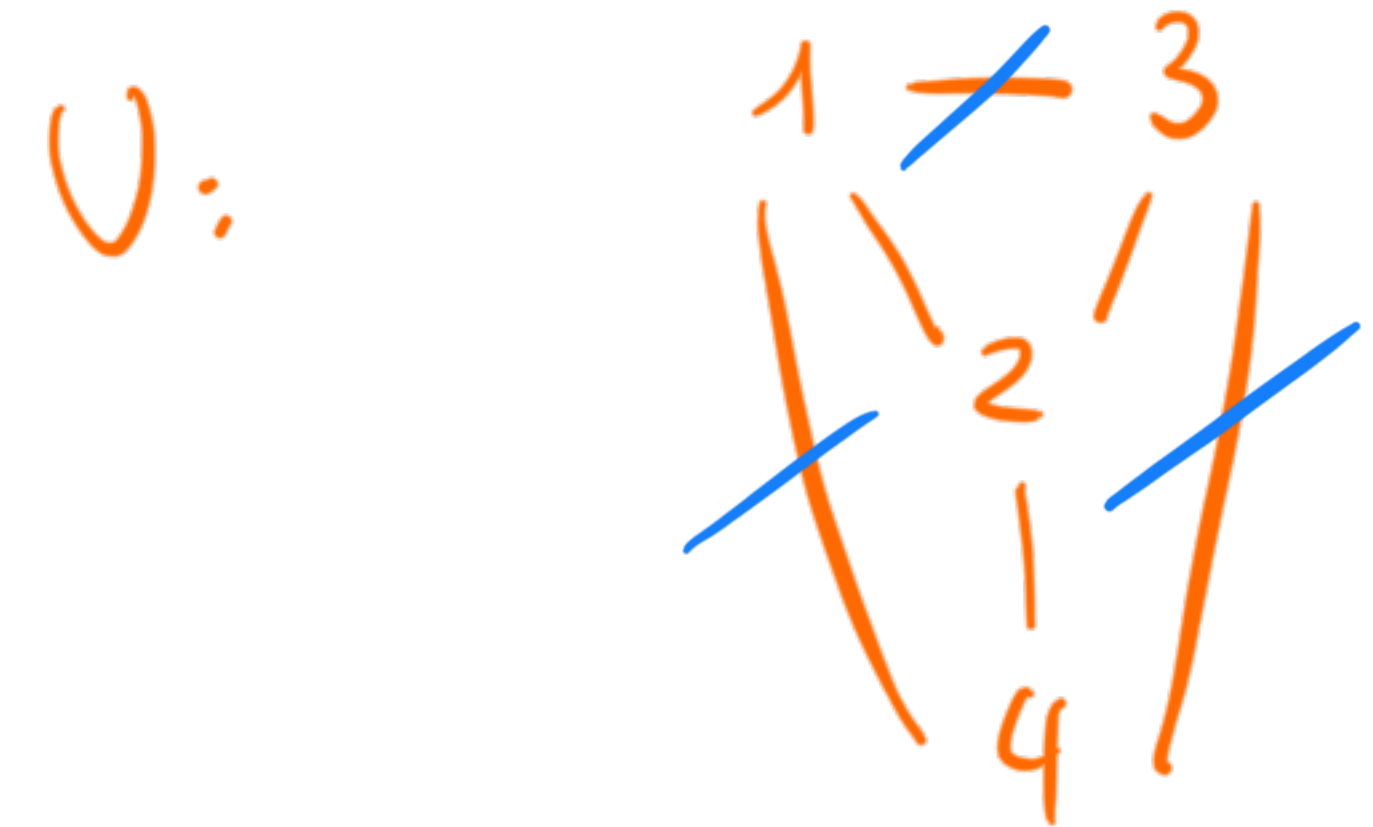
P: $P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

Data

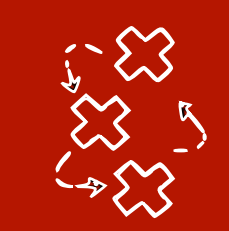
X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0

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

Algorithm output

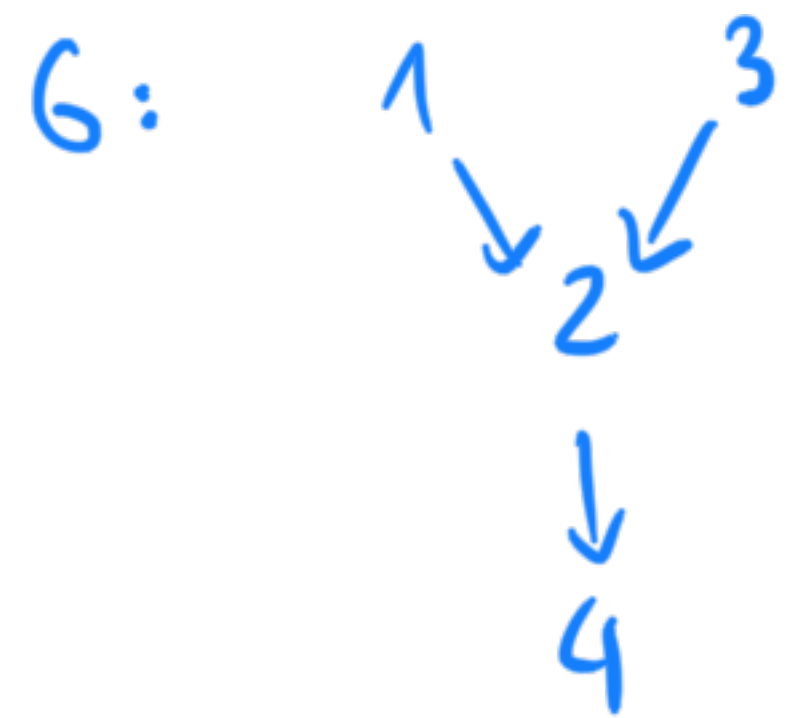


**Nothing
changes, we
can stop**



Step 1: Skeleton learning - example

True causal graph

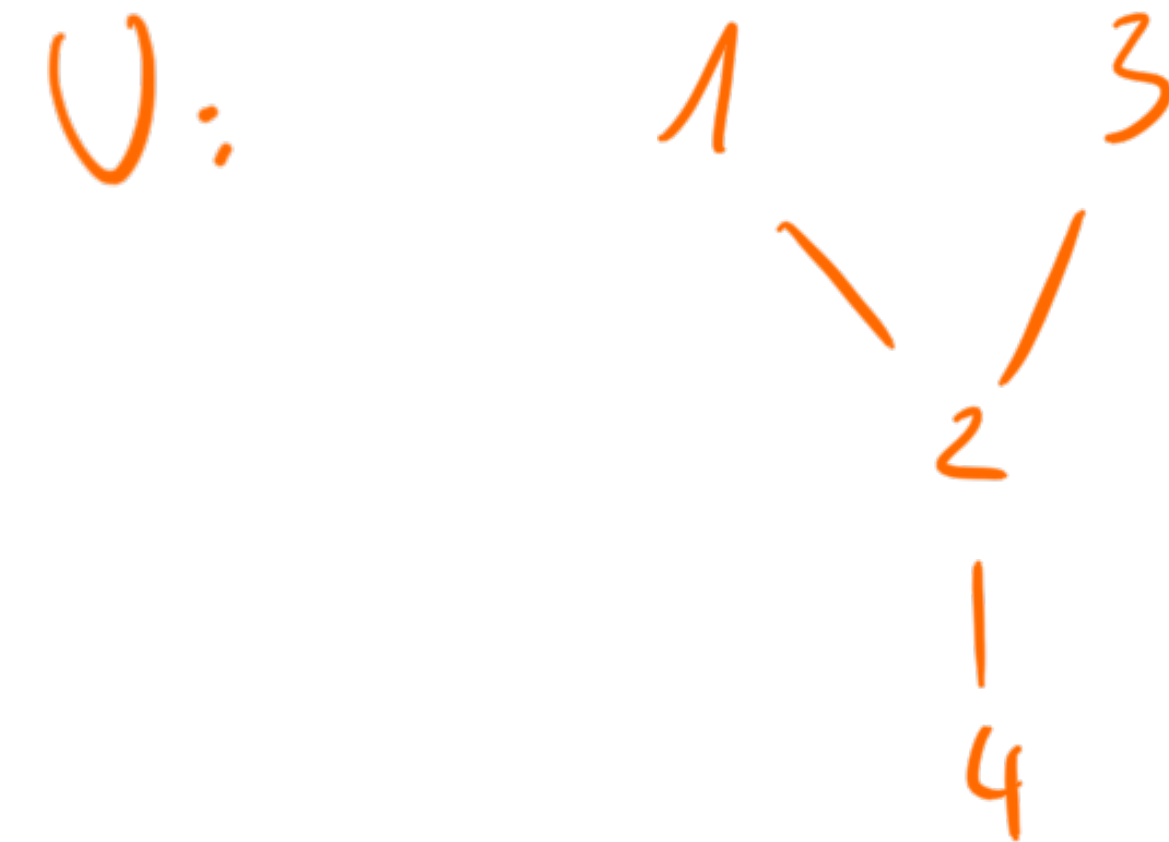


P: $P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

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

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. $i \not\sim k$, 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. $i \neq k$, 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
 - For all $S \subseteq V \setminus \{i, j, k\}$ check if $X_i \not\perp\!\!\!\perp X_k \mid X_j \cup X_S$ in data
 - If this is true, **$i \rightarrow j \leftarrow k$ is a v-structure**

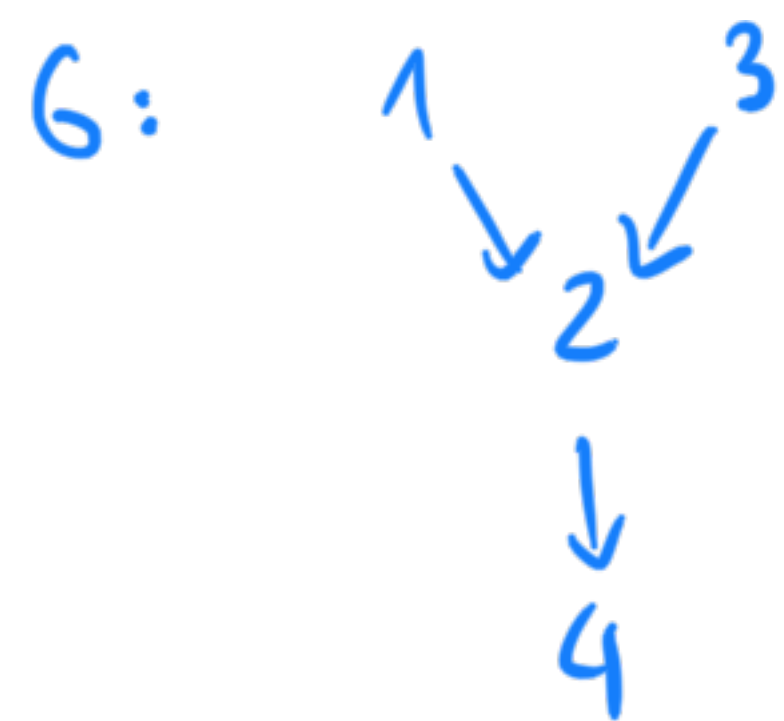
Step 2: Determine v-structures

- 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_k are always dependent **given any conditioning set containing X_j**

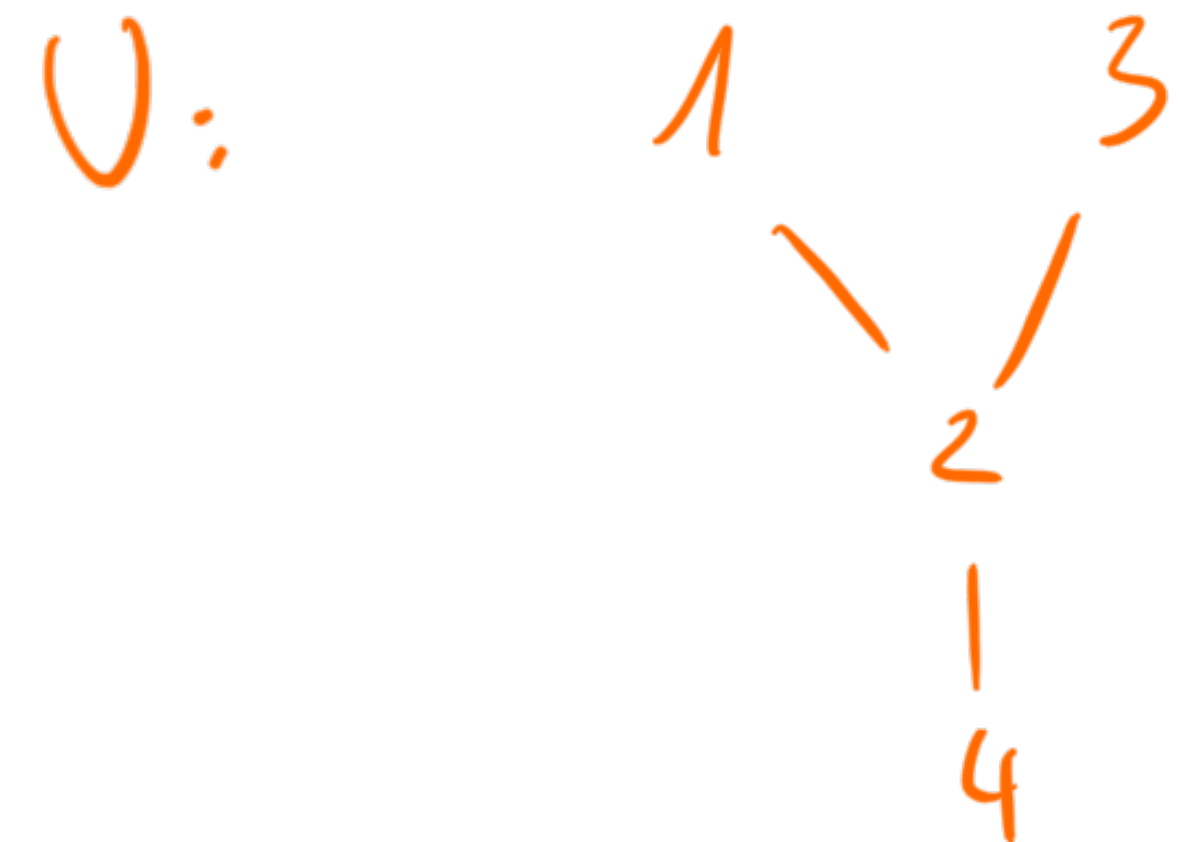
1. Start from the skeleton U from previous step
2. For each unshielded triple (i, j, k) in U , i.e. $i - j, j - k$ and $i \neq k$ in U
 - For all $S \subseteq V \setminus \{i, j, k\}$ check if $X_i \not\perp\!\!\!\perp X_k \mid X_j \cup X_S$ in data
 - If this is true, **$i \rightarrow j \leftarrow k$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

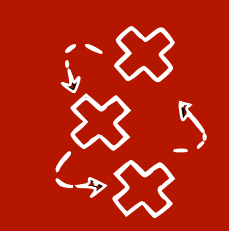
Step 2: Determine v-structures - example



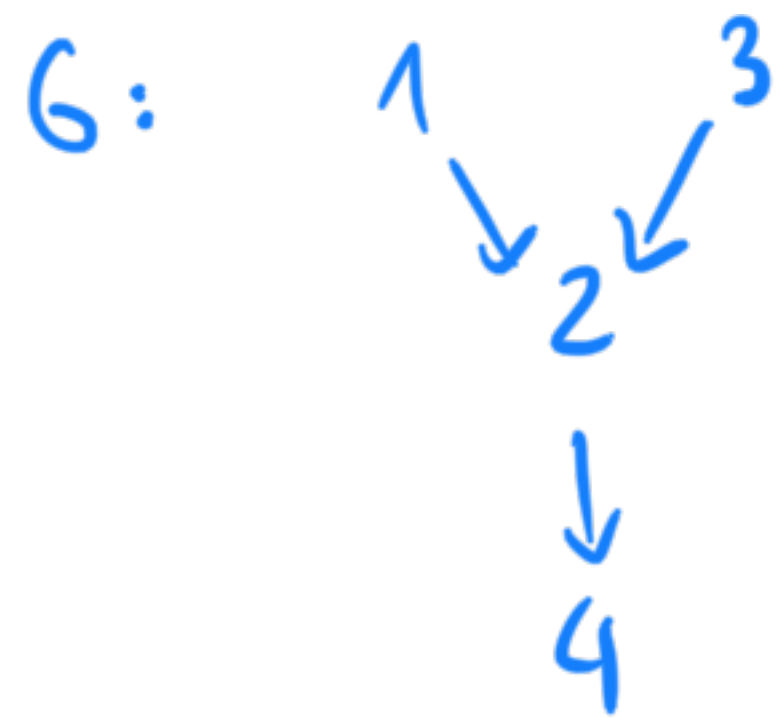
P: $P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

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



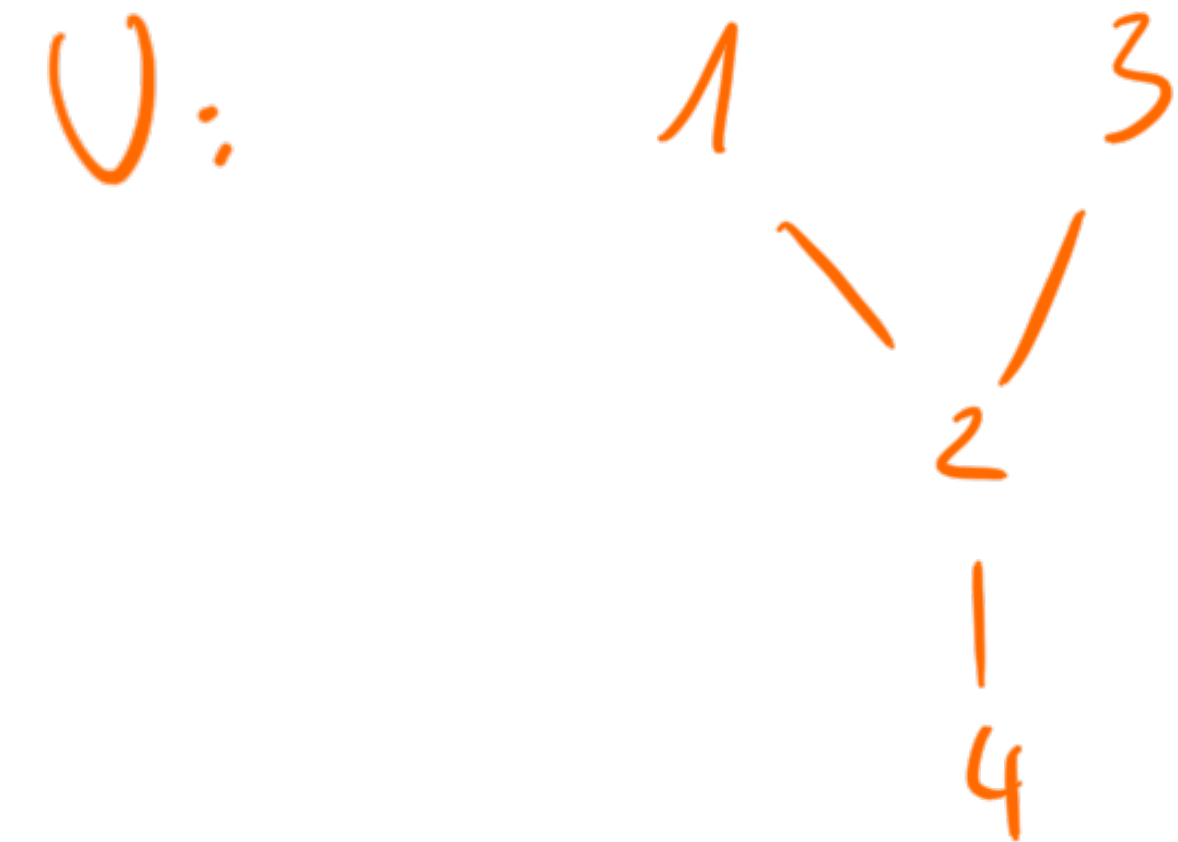


Step 2: Determine v-structures - example



$$P: P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$$

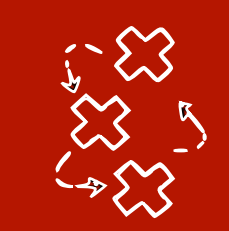
X_1	X_2	X_3	X_4
1	1	1	0
1	0	0	1
0	1	0	1
1	0	1	0
1	0	0	0



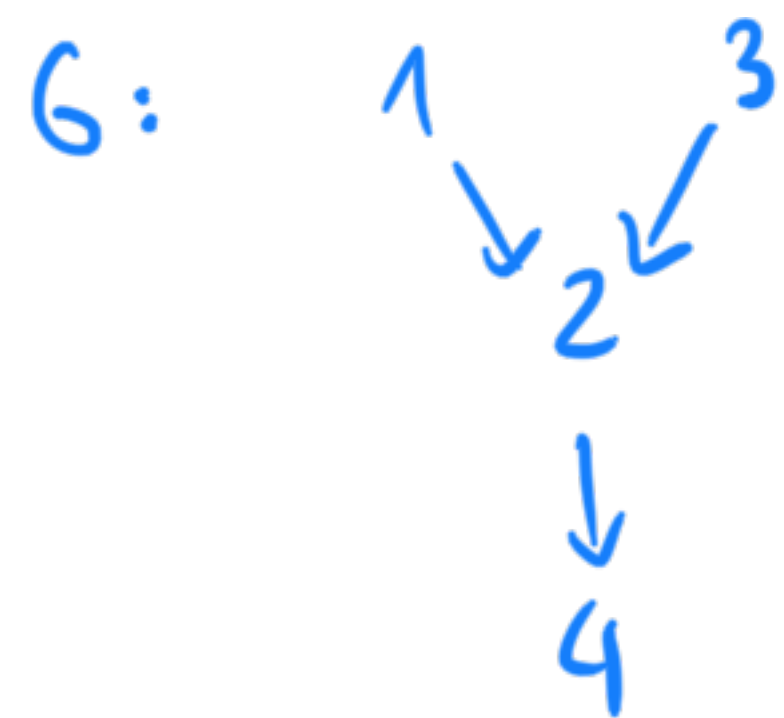
1-2-4: $X_1 \perp\!\!\!\perp X_4 | X_2$
 $X_1 \perp\!\!\!\perp X_4 | X_2, X_3$ ✗

**We can reuse
the CIs from
phase 1**

$$\begin{aligned} X_1 &\perp\!\!\!\perp X_3 \\ X_1 &\perp\!\!\!\perp X_4 | X_2 \\ X_3 &\perp\!\!\!\perp X_4 | X_2 \\ X_1 &\perp\!\!\!\perp X_4 | X_2, X_3 \\ X_3 &\perp\!\!\!\perp X_4 | X_2, X_1 \end{aligned}$$

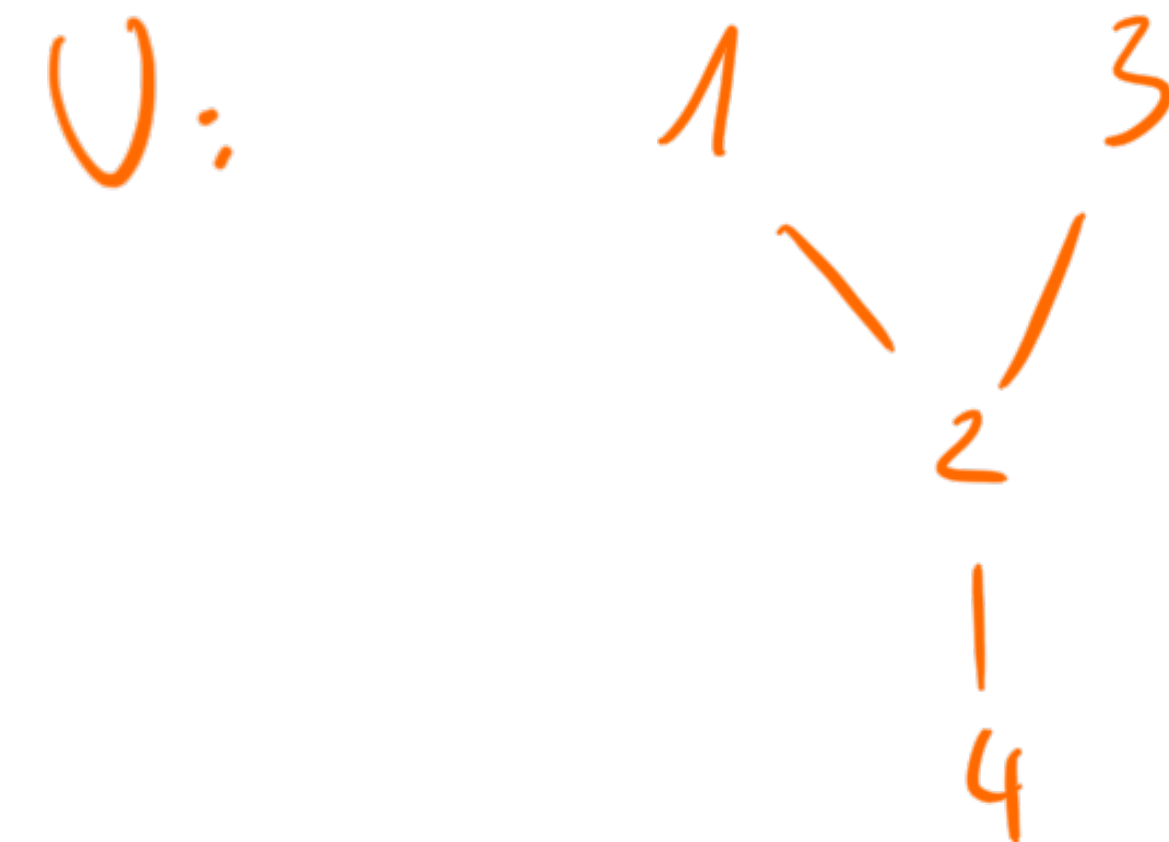


Step 2: Determine v-structures - example



$P: P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$

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



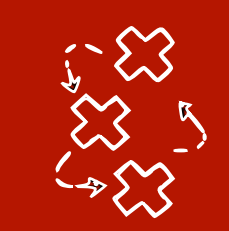
3-2-4

We can reuse
the CIs from
phase 1

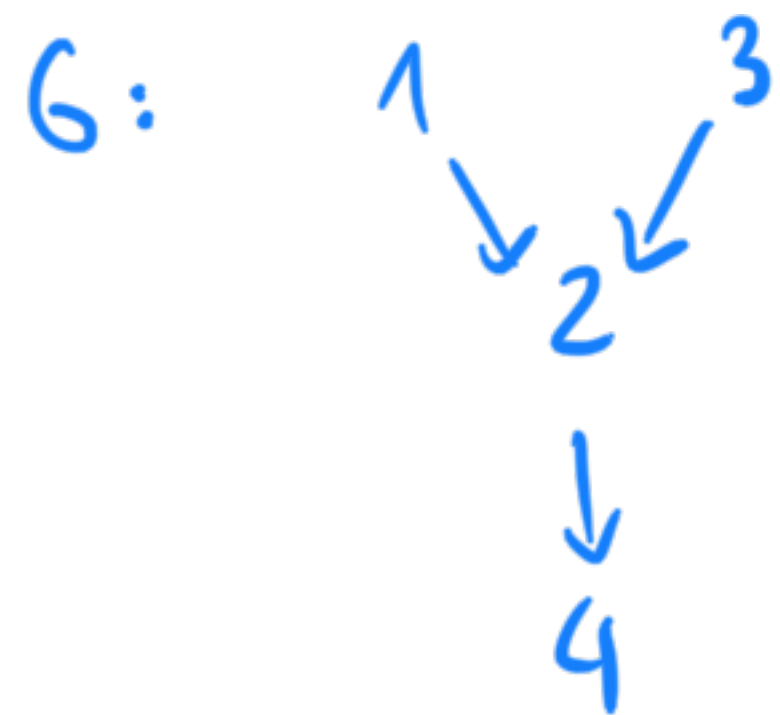
$$\begin{aligned} X_1 &\perp\!\!\!\perp X_3 \\ X_1 &\perp\!\!\!\perp X_4 | X_2 \\ X_3 &\perp\!\!\!\perp X_4 | X_2 \\ X_1 &\perp\!\!\!\perp X_4 | X_2, X_3 \\ X_3 &\perp\!\!\!\perp X_4 | X_2, X_1 \end{aligned}$$

$$\begin{aligned} X_3 &\perp\!\!\!\perp X_4 | X_2 \\ X_3 &\perp\!\!\!\perp X_4 | X_2, X_1 \end{aligned}$$

X

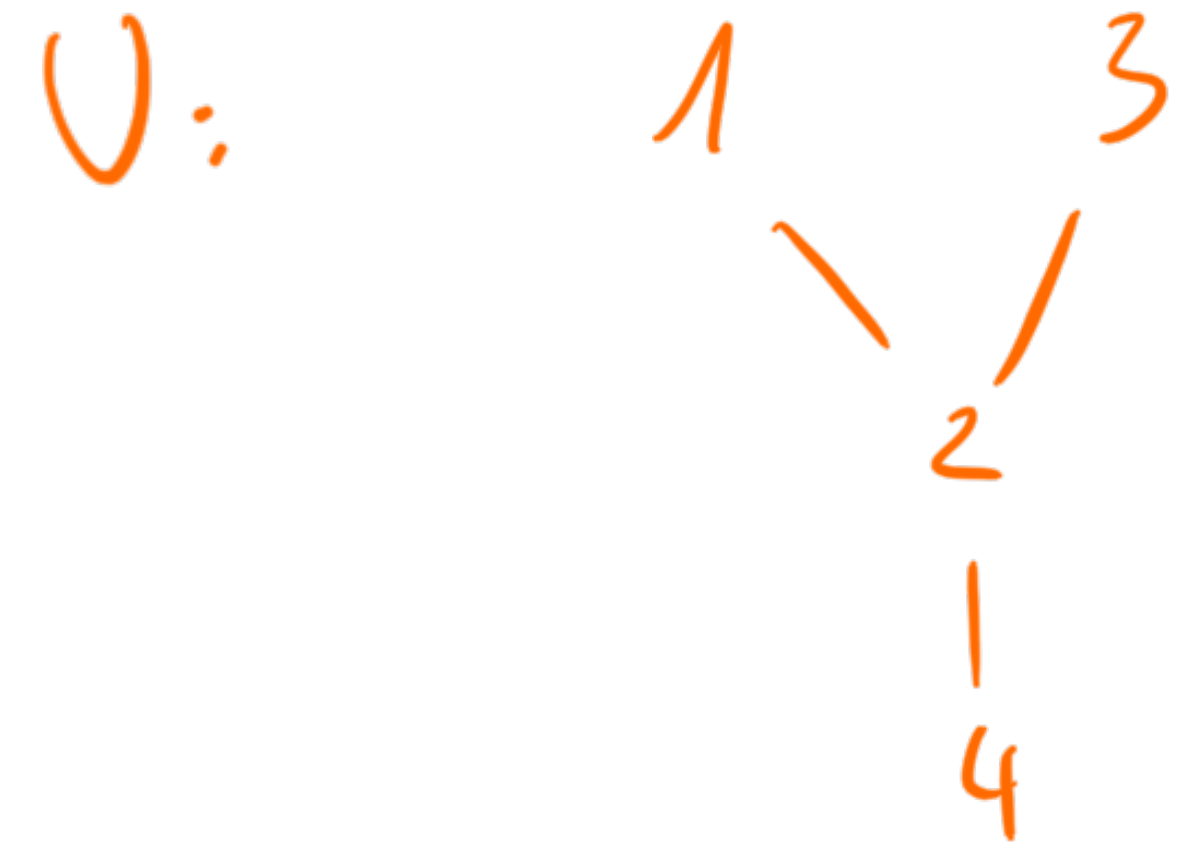


Step 2: Determine v-structures - example



$$P: P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$$

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



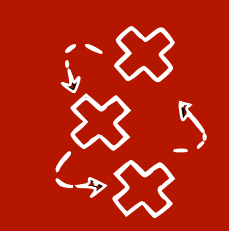
1-2-3

We can reuse
the CIs from
phase 1

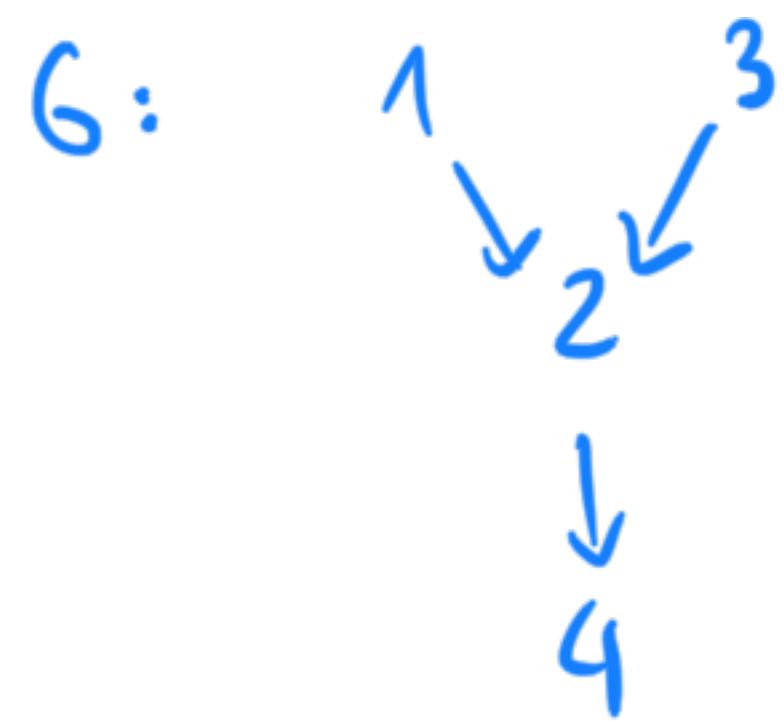
$$\begin{aligned} X_1 &\perp\!\!\!\perp X_3 \\ X_1 &\perp\!\!\!\perp X_4 | X_2 \\ X_3 &\perp\!\!\!\perp X_4 | X_2 \\ X_1 &\perp\!\!\!\perp X_4 | X_2, X_3 \\ X_3 &\perp\!\!\!\perp X_4 | X_2, X_1 \end{aligned}$$

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

$$\begin{aligned} X_1 &\not\perp\!\!\!\perp X_3 | X_2 \quad \checkmark \\ X_1 &\not\perp\!\!\!\perp X_3 | X_2, X_4 \quad \checkmark \end{aligned}$$



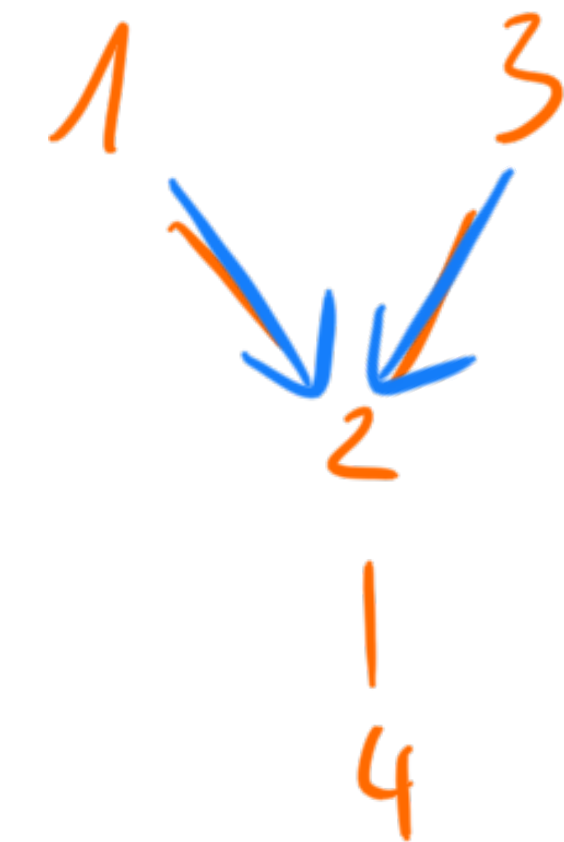
Step 2: Determine v-structures - example



$$P: P(X_1) \cdot P(X_3) \cdot P(X_2 | X_1, X_3) P(X_4 | X_2)$$

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

U:



1 - 2 - 3

~~$X_1 \perp\!\!\!\perp X_3 | X_2$~~ ✓

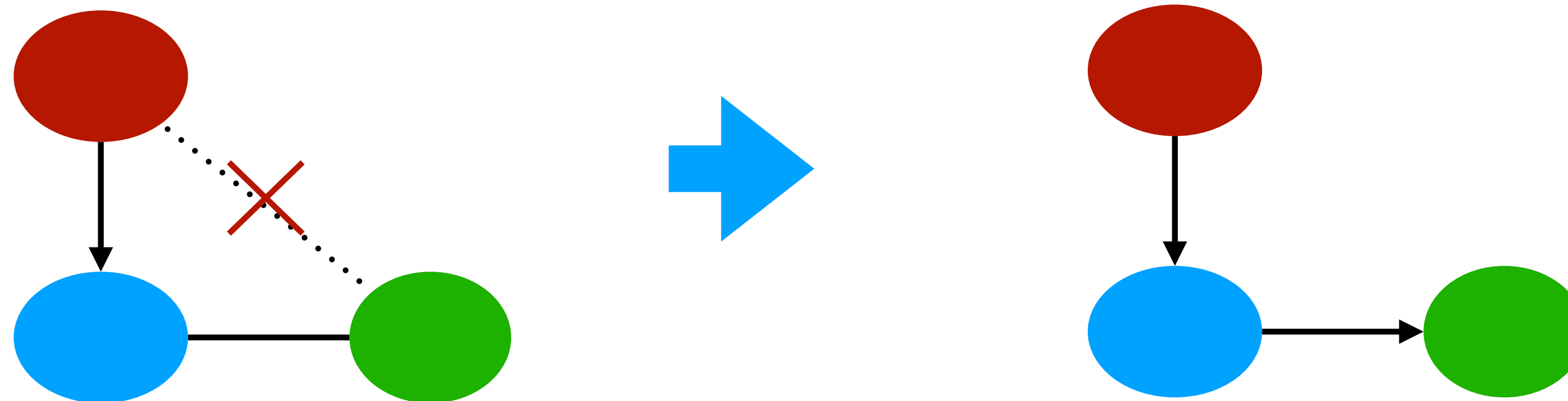
~~$X_1 \perp\!\!\!\perp X_3 | X_2, X_4$~~ ✓

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: Meek's rules (1995) - Rule 1

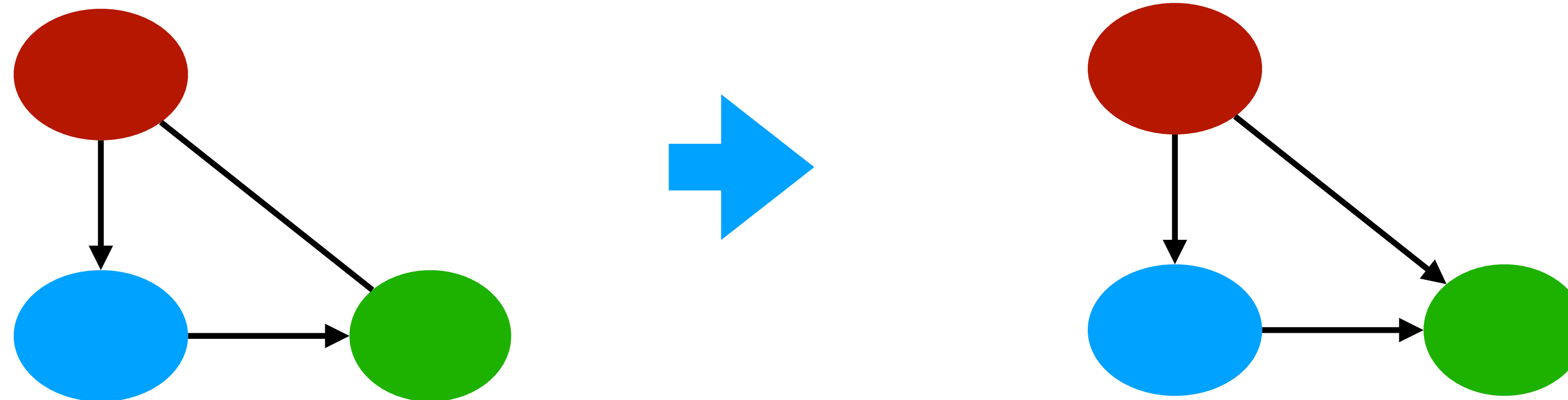
Sound and complete rules for additional orientations (also with added background knowledge)



Otherwise new v-structure (but we had tested them all in Step 2, so contradiction)

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

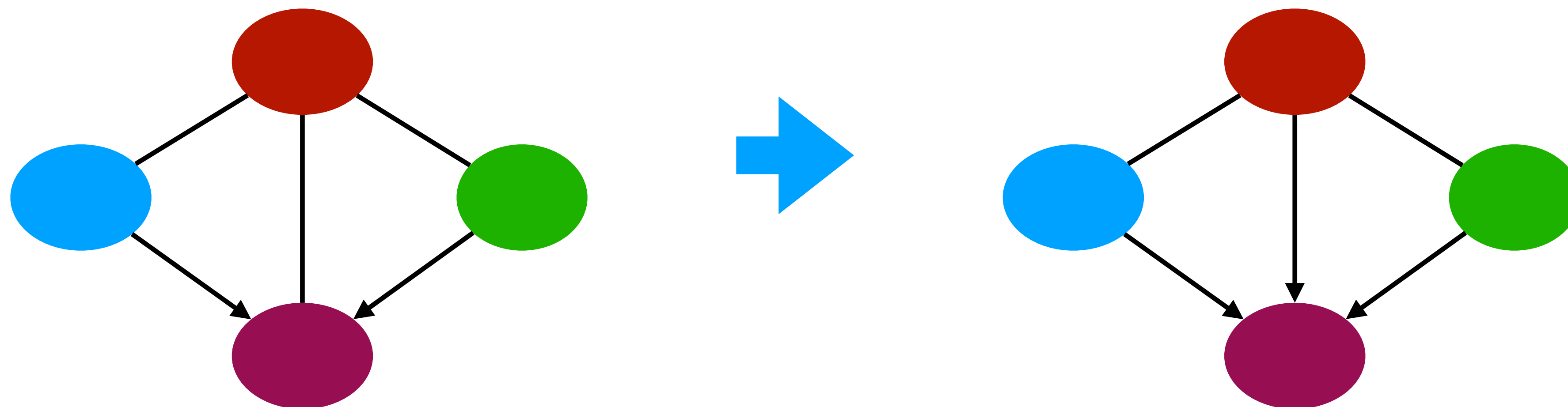
Sound and complete rules for additional orientations (also with added background knowledge)



Otherwise cycle

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

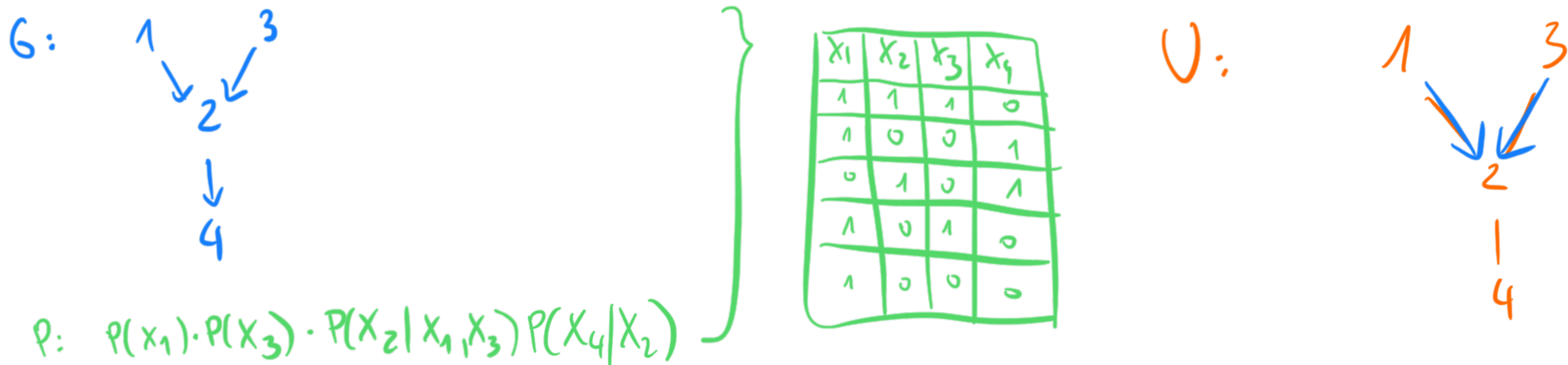
Sound and complete rules for additional orientations (also with added background knowledge)



Otherwise cycle or new v-structure

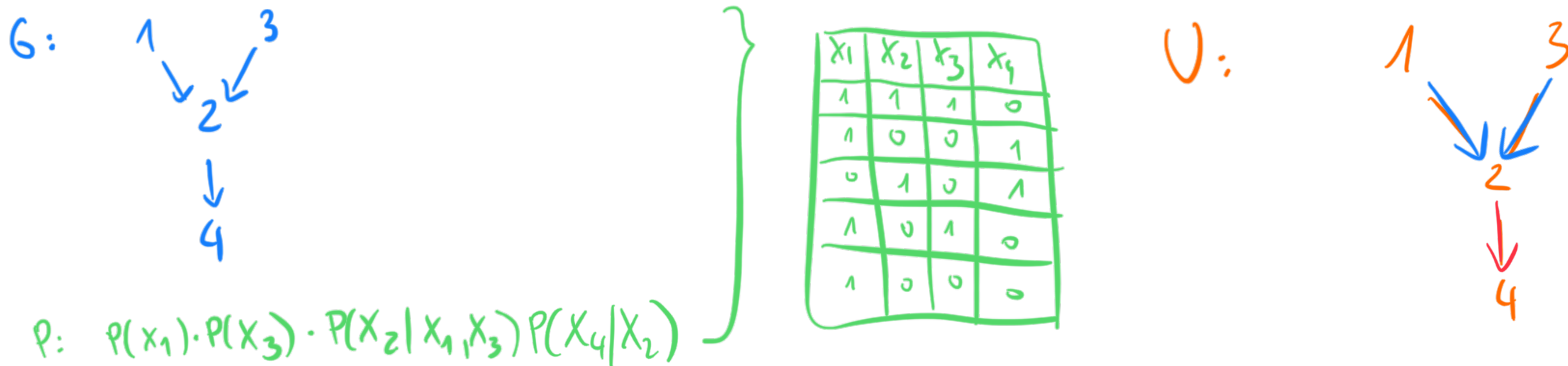
Step 3: Direct as many edges as possible - example

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



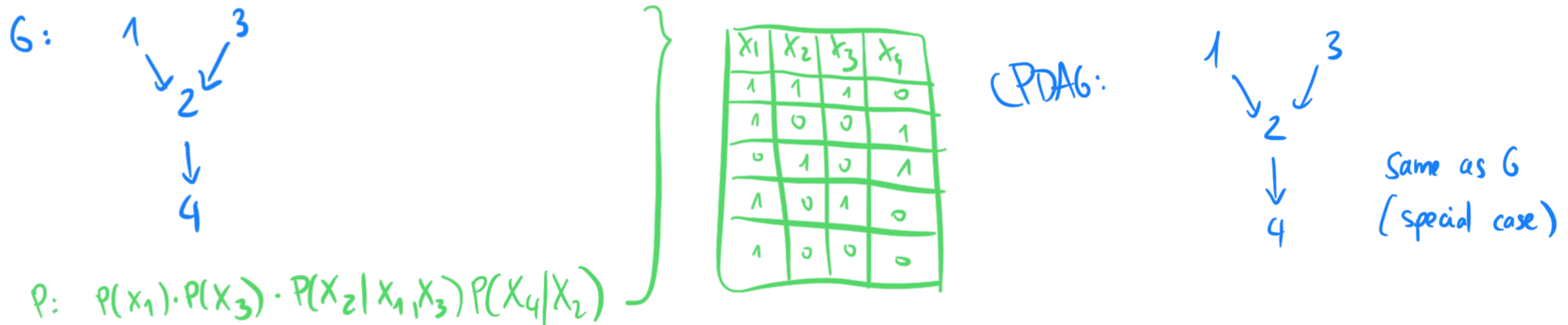
Step 3: Direct as many edges as possible - example

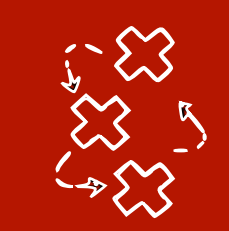
- Cannot create cycles or new v-structures
- Some of the edges can be oriented to disallow these situations to happen



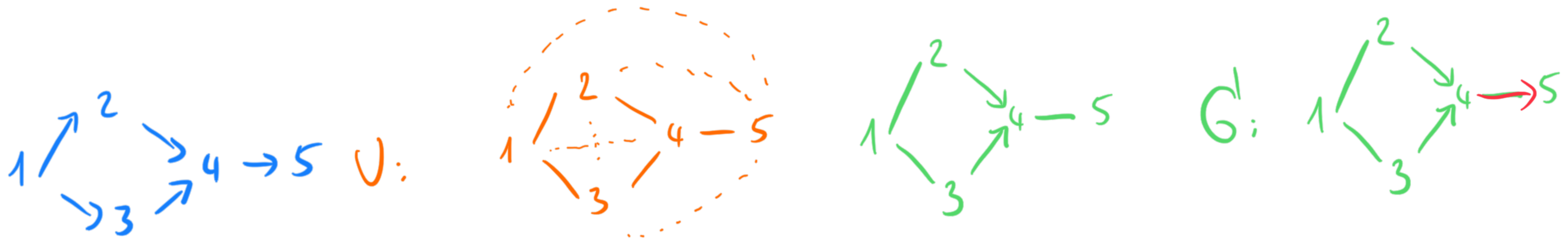
Step 3: Direct as many edges as possible - example

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





Other more complex example



$$\begin{aligned} X_1 &\perp\!\!\!\perp X_4 \mid X_2, X_3 \\ X_2 &\perp\!\!\!\perp X_3 \mid X_1 \\ X_3 &\perp\!\!\!\perp X_5 \mid X_4 \\ X_2 &\perp\!\!\!\perp X_5 \mid X_4 \\ X_1 &\perp\!\!\!\perp X_5 \mid X_4 \end{aligned}$$

1-2-4

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

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

1-3-4

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

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

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

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

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

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

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

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
- **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 $\text{Pa}(i)$ or $\text{Pa}(j)$
- Determine the skeleton in an optimised way
 - Since we do not know the parents we will use the nodes that are adjacent, $\text{Adj}(i)$ or $\text{Adj}(j)$ 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 $\text{Pa}(i)$ or $\text{Pa}(j)$
 - Determine the skeleton in an optimised way
 - Since we do not know the parents we will use the nodes that are adjacent, $\text{Adj}(i)$ or $\text{Adj}(j)$ 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 $\text{Pa}(i)$ or $\text{Pa}(j)$
 - Determine the skeleton in an optimised way
 - Since we do not know the parents we will use the nodes that are adjacent, $\text{Adj}(i)$ or $\text{Adj}(j)$ in U at a given iteration (*superset*)
1. Start with **completely connected undirected graph U**
 2. For $k = 0, 1, 2, \dots, p - 2$
 - If $i - j$ in U and there exists a set $S \subseteq \text{Adj}(i) \setminus \{j\}$ of size at least k

PC algorithm skeleton learning

- If i is not adjacent to j , then they can be d-separated by $\text{Pa}(i)$ or $\text{Pa}(j)$
- Determine the skeleton in an optimised way
 - Since we do not know the parents we will use the nodes that are adjacent, $\text{Adj}(i)$ or $\text{Adj}(j)$ in U at a given iteration (*superset*)
- 1. Start with **completely connected undirected graph U**
- 2. For $k = 0, 1, 2, \dots, p - 2$
 - If $i - j$ in U and there exists a set $S \subseteq \text{Adj}(i) \setminus \{j\}$ of size at least k
 - If holds $X_i \perp\!\!\!\perp X_j \mid S$, then **remove $i - j$ in U** (and add S to $\text{SepSet}(i, j)$)

PC algorithm skeleton learning

- If i is not adjacent to j , then they can be d-separated by $\text{De}(i)$ or $\text{De}(j)$
 - Determine the **skeleton in an optimal order** and X_j where we store the results of the conditional independence tests
 - Since we do not know the parents adjacent, $\text{Adj}(i)$ or $\text{Adj}(j)$ 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
1. Start with **completely connected**
 2. For $k = 0, 1, 2, \dots, p - 2$
 - If $i - j$ in U and there exists a set $S \subseteq \text{Adj}(i) \setminus \{j\}$ of size at least k
 - If holds $X_i \perp\!\!\!\perp X_j \mid S$, then **remove $i - j$ in U** (and add S to SepSet(i, j))
- We stop when there are no more untested sets of size k in the adjacencies of any variable

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**

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

What if we want to take into account the uncertainty?

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

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

Score-based causal discovery

- **Constraint-based causal discovery:** test conditional independences in the data and use them to constrain the possible causal graphs **(up to MEC)**
- **Score-based causal discovery:** find the graph that maximises a **score** $S(G, D)$ (fit of graph G on data D)

$$\hat{G} = \operatorname{argmax}_{G \in \mathcal{G}} S(G, D)$$

Model selection

Score-based causal discovery

- **Constraint-based causal discovery:** test conditional independences in the data and use them to constrain the possible causal graphs **(up to MEC)**
- **Score-based causal discovery:** find the graph that maximises a **score** $S(G, D)$ (fit of graph G on data D)
 - This is also called **Bayesian network structure learning**
 - We will use some specific properties for the score, e.g. **score equivalence:**
 - All DAGs in a MEC have the same score

Score-based causal discovery

- **Constraint-based causal discovery:** test conditional independences in the data and use them to constrain the possible causal graphs (**up to MEC**)
- **Score-based causal discovery:** We usually need to know the parametric model to compute the score, which is often based on the likelihood. This gives a **score** $S(G, D)$ (fit of graph to data).
 - This is also called **Most commonly linear Gaussian or discrete**
- We will use some specific properties for the score, e.g. **score equivalence:**
 - All DAGs in a MEC have the same score

An alternative view: Structural causal models (SCMs)

- Let G be a **causal graph**
- We can write each variable X_i for $i \in \mathbf{V}$ as a **function of its parents** in G and a **noise term** ϵ_i in a **structural equation**:

$$X_i \leftarrow h_i(X_{\text{Pa}(i)}, \epsilon_i)$$

- We assume all noises are **independent of each other** $\forall i \neq j : \epsilon_i \perp\!\!\!\perp \epsilon_j$

An alternative view: Structural causal models (SCMs)

- Let G be a **causal graph**
- We can write each variable X_i for $i \in \mathbf{V}$ as a **function of its parents** in G and a **noise term** ϵ_i in a **structural equation**:

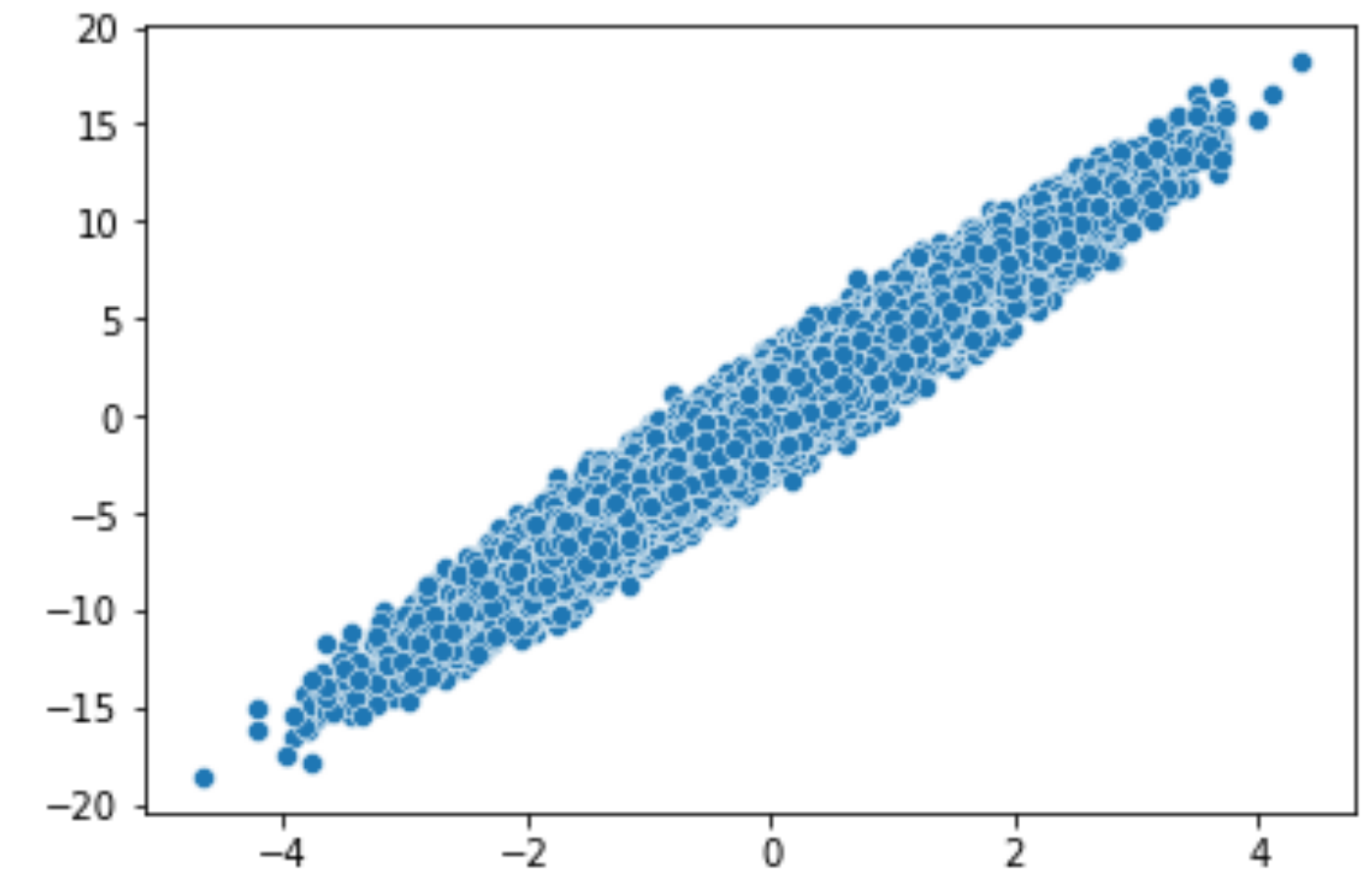
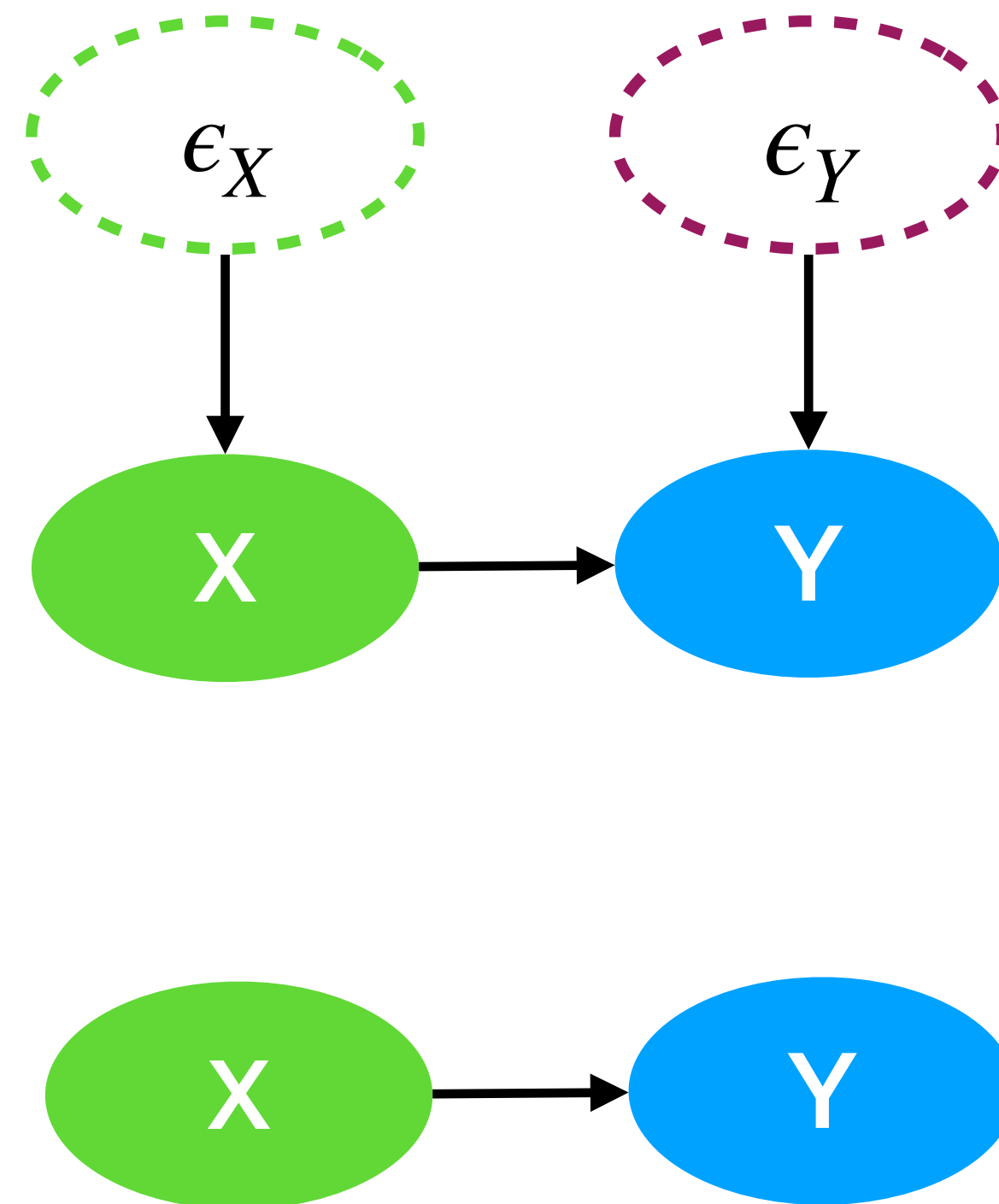
$$X_i \leftarrow h_i(X_{\text{Pa}(i)}, \epsilon_i)$$

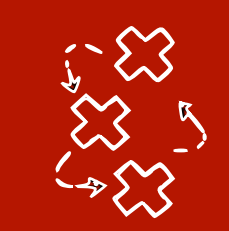
often linear
often Gaussian

- We assume all noises are **independent of each other** $\forall i \neq j : \epsilon_i \perp\!\!\!\perp \epsilon_j$

Example 3.2 in Elements of Causal Inference

$$\begin{cases} X \leftarrow \epsilon_X \\ Y \leftarrow 4 \cdot X + \epsilon_Y \\ \epsilon_X, \epsilon_Y \sim \mathcal{N}(0,1) \end{cases}$$



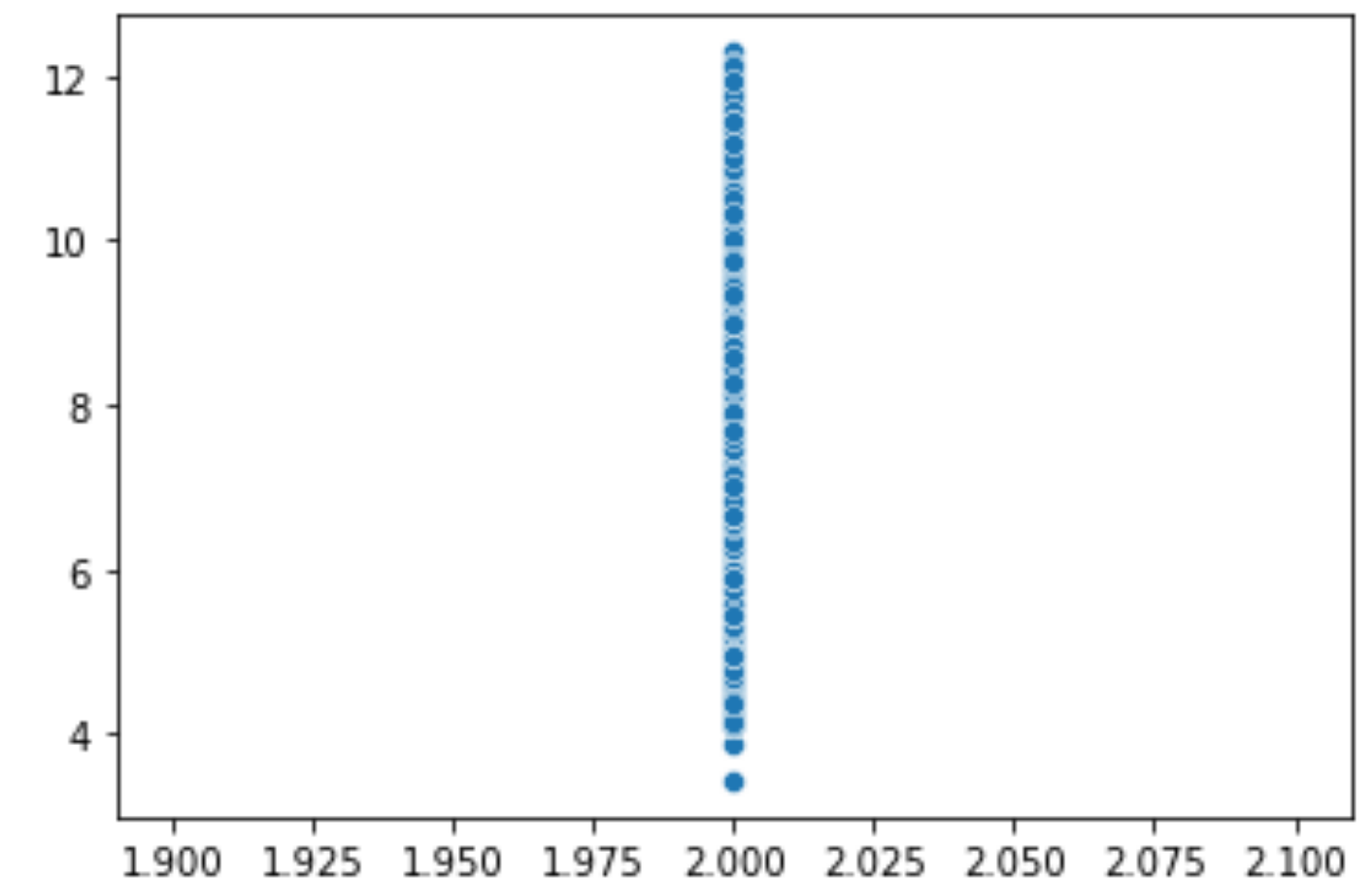
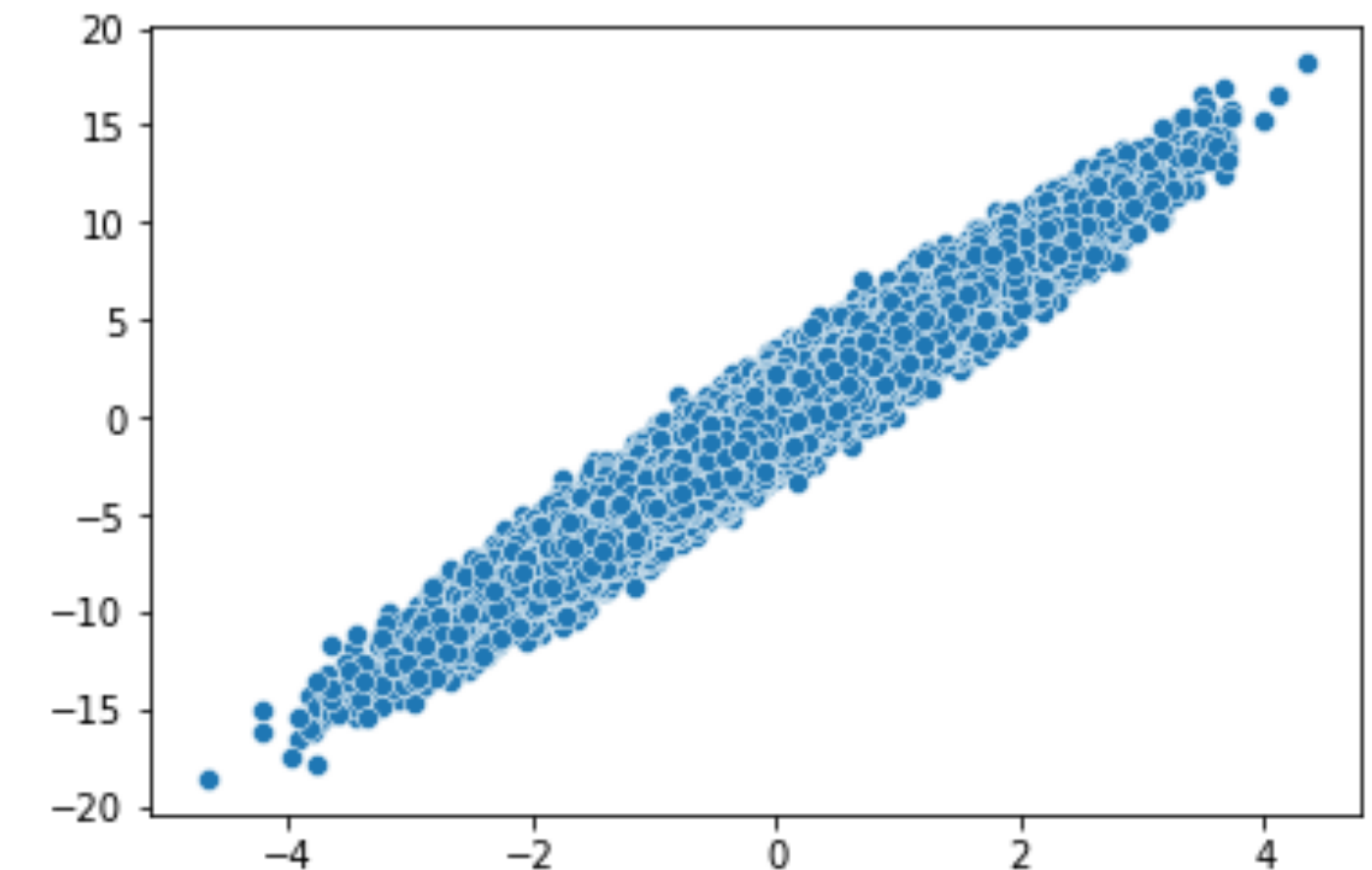
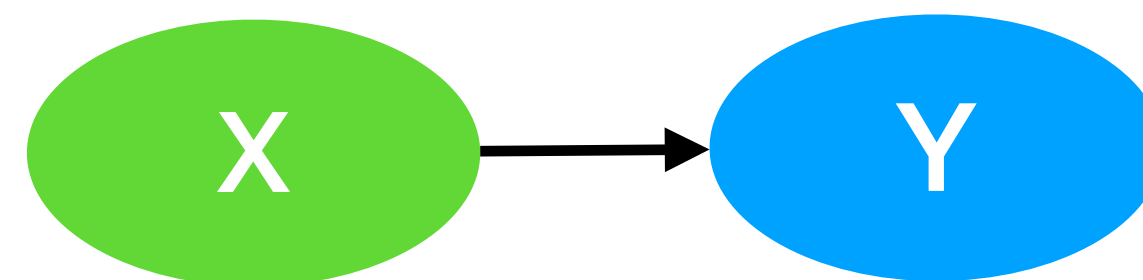
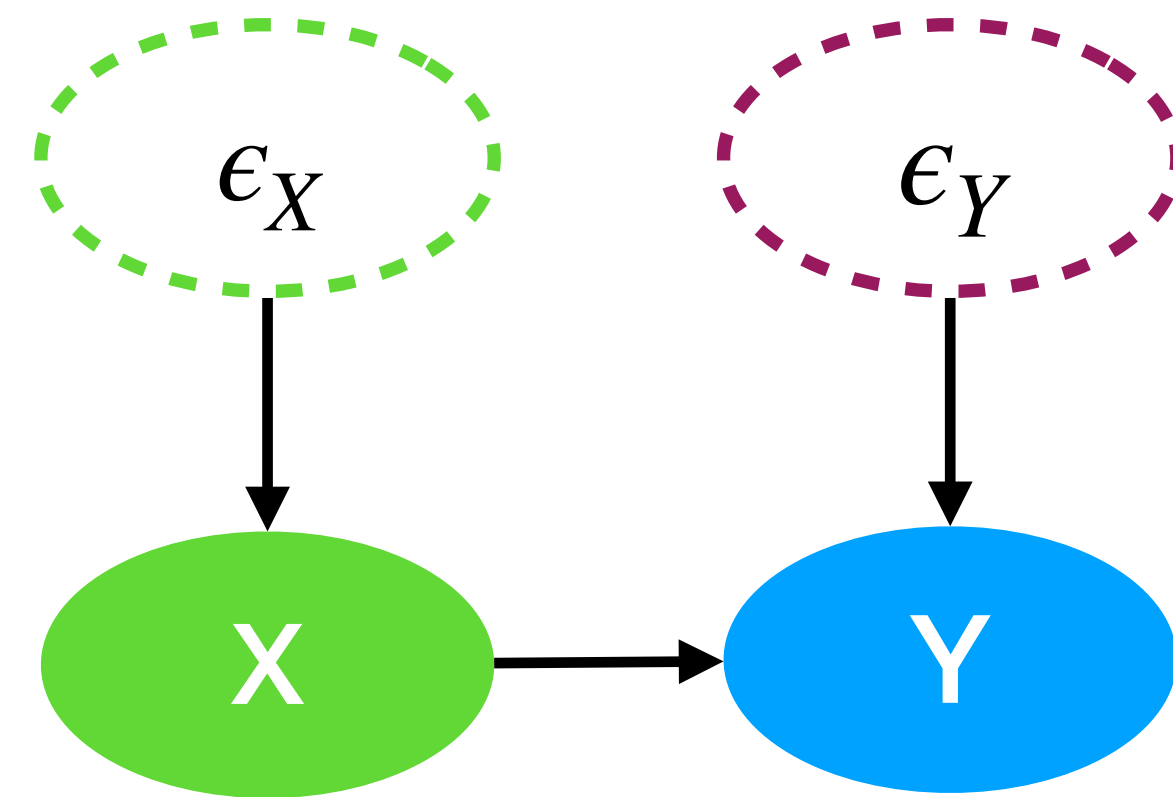


Perfect intervention example

$$\begin{cases} X \leftarrow \epsilon_X \\ Y \leftarrow 4 \cdot X + \epsilon_Y \\ \epsilon_X, \epsilon_Y \sim \mathcal{N}(0,1) \end{cases}$$

$do(X=2):$

$$\begin{cases} X \leftarrow 2 \\ Y \leftarrow 4 \cdot X + \epsilon_Y \end{cases}$$



More general types of interventions: soft interventions

- A soft intervention on an variable just changes its equation in the SCM
 - Usually we assume that the parents of Y after intervention are either the same or a subset of the parents in observational data
- A perfect intervention is just a special case of a soft intervention

$$\begin{cases} X \leftarrow \epsilon_x \\ Y \leftarrow 4 \cdot X + \epsilon_Y \end{cases}$$

$$\epsilon_X, \epsilon_Y \sim \mathcal{N}(0,1)$$

$$\begin{cases} X \leftarrow \epsilon_x \\ Y \leftarrow 4 \cdot X + \epsilon_Y \end{cases}$$

$$Y \leftarrow 2 \cdot X + \epsilon_Y$$

$$\begin{cases} X \leftarrow \epsilon_x \\ Y \leftarrow 4 \cdot X + \epsilon_Y + \epsilon' \end{cases}$$

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$

i.i.d. = independent and identically distributed

In other words, we sampled them independently one from each other and they come from the same P

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$

- Attempt 1: Bayes theorem

$$P(G, \theta | D) = \frac{P(D | G, \theta)P(G, \theta)}{P(D)}$$

$$\begin{cases} X_2 \leftarrow \epsilon_{X_2} \\ X_1 \leftarrow 3 \cdot X_2 + \epsilon_{X_1} \\ X_3 \leftarrow 5 \cdot X_1 + 4 \cdot X_2 + \epsilon_{X_3} \end{cases}$$

$$\epsilon_{X_1}, \epsilon_{X_2}, \epsilon_{X_3} \sim \mathcal{N}(0, 1)$$

G graph structure, **θ** parameters,
e.g. coefficients in a linear SCM

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$

- Attempt 1: Bayes theorem

$$P(G, \theta | D) = \frac{P(D | G, \theta) P(G, \theta)}{P(D)}$$

Prior (assumed to be uniform)

Normalisation constant

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$

- Attempt 1: Bayes theorem

Likelihood, depends on G and θ

$$P(G, \theta | D) = \frac{P(D | G, \theta)P(G, \theta)}{P(D)}$$

- Find G, θ that maximise $P(D | G, \theta)$

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$
- Attempt 1: Find G, θ that maximise $P(D | G, \theta)$

$$P(D | G, \theta) = \prod_{i=1}^n P(x_1^i, \dots, x_p^i | G, \theta)$$

↑
i.i.d. n samples

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$
- Attempt 1: Find G, θ that maximise $P(D | G, \theta)$

$$P(D | G, \theta) = \prod_{i=1}^n P(x_1^i, \dots, x_p^i | G, \theta) = \prod_{i=1}^n \prod_{j=1}^p P(x_j^i | x_{\text{Pa}_G(j)}^i, \theta)$$

i.i.d. n samples

BN factorization

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$
- Attempt 1: Find G, θ that maximise $P(D \mid G, \theta)$

$$P(D \mid G, \theta) = \prod_{i=1}^n P(x_1^i, \dots, x_p^i \mid G, \theta) = \prod_{i=1}^n \prod_{j=1}^p P(x_j^i \mid x_{\text{Pa}_G(j)}^i, \theta)$$

$$\log P(D \mid G, \theta) = \sum_i^n \sum_j^p \log P(x_j^i \mid x_{\text{Pa}_G(j)}^i, \theta)$$

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$
- Attempt 1: Find G, θ that maximise $P(D | G, \theta)$

$$\log P(D | G, \theta) = \sum_i^n \sum_j^p \log P(x_j^i | x_{\text{Pa}(j)_G}^i, \theta)$$

Maximum likelihood estimation

$$(G^{MLE}, \theta^{MLE}) = \operatorname{argmax}_{G, \theta} \log P(D | G, \theta)$$

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$

- Attempt 1: Find G, θ that maximise $P(D | G, \theta)$

$$\log P(D | G, \theta) = \sum_i^n \sum_j^p \log P(x_j^i | x_{\text{Pa}(j)_G}^i, \theta)$$

Maximum likelihood estimation

$$(G^{MLE}, \theta^{MLE}) = \operatorname{argmax}_{G, \theta} \log P(D | G, \theta)$$

- **Issue:** this will result in fully connected DAGs (*overfitting*)

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$
- **Attempt 2:** G, θ maximise $P(D \mid G, \theta) +$ **encourage sparsity with penalty**

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of n **i.i.d.** samples, where each $x_{\mathbf{V}}^i = (x_1^i, \dots, x_p^i)$ is a realisation for a unit i for all random variables $X_{\mathbf{V}}$
- **Attempt 2:** G, θ maximise $P(D | G, \theta) +$ **encourage sparsity with penalty**
- Typically we use **BIC (Bayesian information criterion)**

$$BIC(D, G) := 2 \cdot \log p(D | G, \theta_G^{MLE}) - \log(n) \cdot \text{\#parameters}$$

$$G^{BIC} = \operatorname{argmax}_G BIC(D, G)$$

E.g., number of non-zero coefficients for linear SCMs -> edges in G

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of i.i.d. samples
- For each G , we can compute $BIC(D, G)$ as:

1. Given G we find

$$\theta_G^{MLE} = \operatorname{argmax}_{\theta_G} \log P(D | G, \theta_G) = \sum_i^n \sum_j^p \log P(x_j^i | x_{\text{Pa}_G(j)}^i, \theta_G)$$

Bayesian Information Criterion (BIC)

- We have a dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ of i.i.d. samples
- For each G , we can compute $BIC(D, G)$ as:

1. Given G we find

$$\theta_G^{MLE} = \operatorname{argmax}_{\theta_G} \log P(D | G, \theta_G) = \sum_i^n \sum_j^p \log P(x_j^i | x_{\text{Pa}_G(j)}^i, \theta_G)$$

2. Compute $BIC(D, G) = 2\log P(D | G, \theta_G^{MLE}) - \log(n) \cdot \text{\#edges in } G$

Bayesian Information Criterion (BIC) - properties

- **Score equivalence:** all DAGs in a MEC get the same score

Bayesian Information Criterion (BIC) - properties

- **Score equivalence:** all DAGs in a MEC get the same score
- **Decomposable:** we can decompose the score as the sum of the contributions for each variable and its parents
 - If we want to compare two similar DAGs, we just need to check the variables with different edges, the rest stays the same

Bayesian Information Criterion (BIC) - properties

- **Score equivalence:** all DAGs in a MEC get the same score
- **Decomposable:** we can decompose the score as the sum of the contributions for each variable and its parents
 - If we want to compare two similar DAGs, we just need to check the variables with different edges, the rest stays the same
- **Local consistency :** adding **superfluous** edges decreases score, instead adding an edge that adds a “correct” dependence based on the data increases it

Number of DAGs with n nodes ?

A003024 as a simple table

n	$a(n)$
0	1
1	1
2	3
3	25
4	543
5	29281
6	3781503
7	1138779265
8	783702329343
9	1213442454842881
10	4175098976430598143
11	31603459396418917607425
12	521939651343829405020504063
13	18676600744432035186664816926721
14	1439428141044398334941790719839535103

Number of DAGs grows
super-exponentially!

Number of DAGs with n nodes ?

A003024 as a simple table

n	a(n)
0	1
1	1
2	3
3	25
4	543
5	29281
6	3781503
7	1138779265
8	783702329343
9	1213442454842881
10	4175098976430598143
11	31603459396418917607425
12	521939651343829405020504063
13	18676600744432035186664816926721
14	1439428141044398334941790719839535103

Number of DAGs grows
super-exponentially!

$$G^{BIC} = \operatorname{argmax}_G \operatorname{BIC}(D, G)$$

We cannot do exhaustive
search for BIC

Greedy search for graphs (informal)

- Given data D and score $S(D, G)$, iterate:
 - Candidate graph G with score S
 - **Get neighbouring graphs** $\mathcal{G}$
 - Find **best scoring neighbour** $G^* = \operatorname{argmax}_{G' \in \mathcal{G}} S(D, G')$
 - If $S(D, G^*) > S(D, G)$
 - Take G^* as new candidate graph G and **iterate**
- Otherwise return G

Greedy Equivalence Search (GES)

- Optimise **Bayesian Information Criterion (BIC)**
- Reduce search space by **searching over CPDAGs** instead of DAGs

Greedy Equivalence Search (GES)

- Optimise **Bayesian Information Criterion (BIC)**
- Reduce search space by **searching over CPDAGs** instead of DAGs
- BIC is score-equivalent, so DAGs in same Markov equivalence class (so represented by same CPDAG) have the same score (*so you can pick any*)
- It can be shown that searching over CPDAGs with local consistency allows us to find the **global optimum** (*in large sample limit*) even with **greedy search**

Naive Equivalence Search

1. Start with complete CPDAG
2. Remove edges one by one until local maxima in BIC

decomposability helps us
to recompute only one factor

Computationally inefficient
(especially sparse graphs)

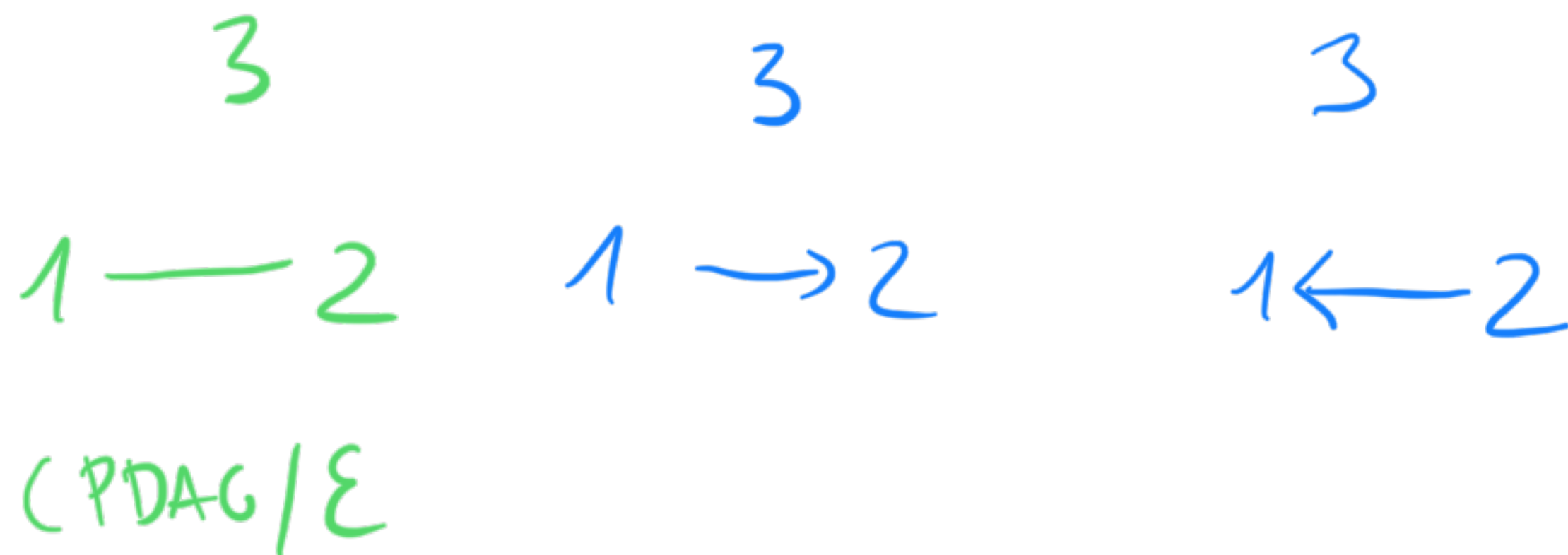
Greedy Equivalence Search (GES)

1. Start with empty CPDAG
2. **Forward (phase 1):** Add edges one by one until local maxima in BIC
3. **Backward (phase 2):** Remove edges one by one until local maxima in BIC

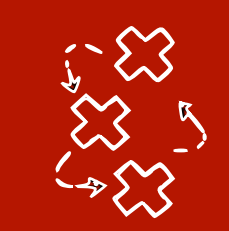
We need to be careful when adding and removing edges (**valid moves**), because we want to have a CPDAG at each step

Greedy Equivalence Search (GES)

1. Start with empty CPDAG
2. **Forward (phase 1):** Add edges one by one until local maxima in BIC

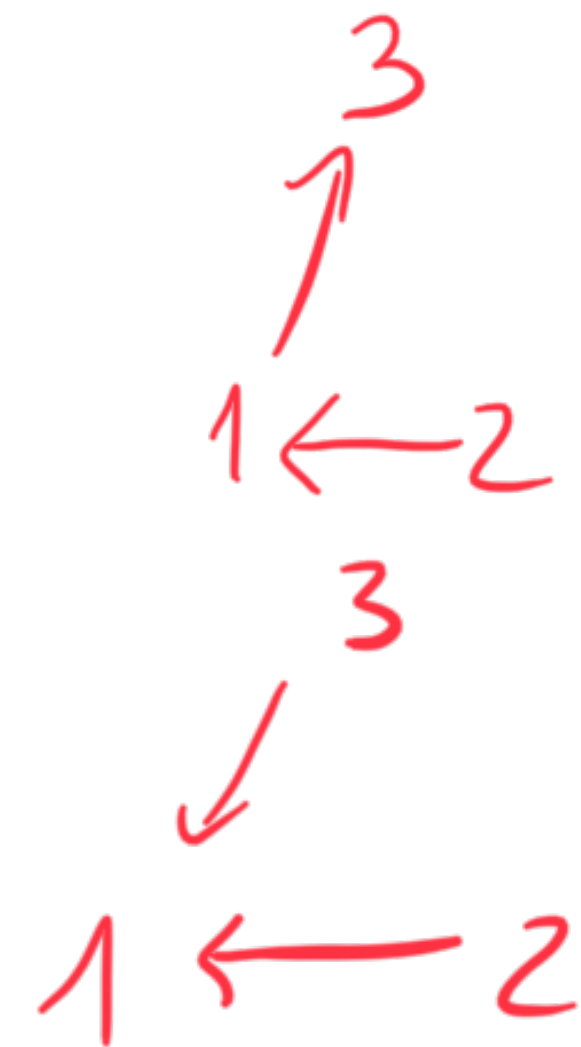
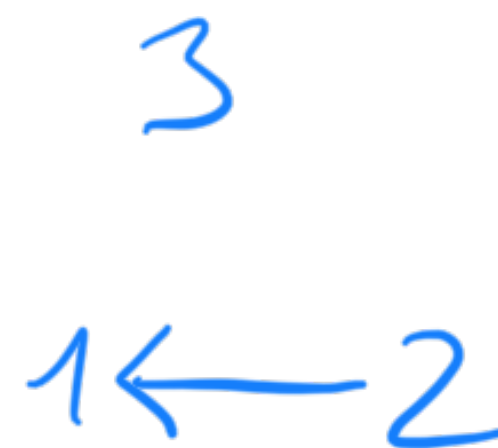
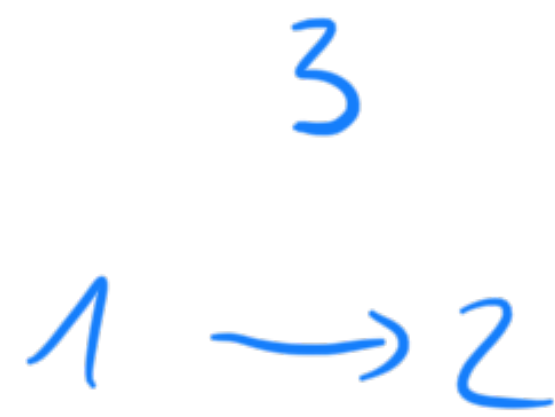
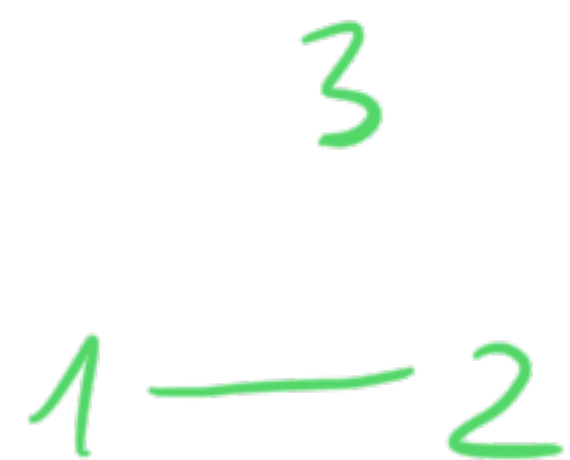


Step 1: Enumerate all DAGs $G \in \varepsilon$ in the initial CPDAG ε



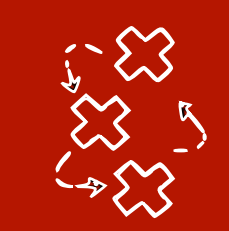
Greedy Equivalence Search (GES)

1. Start with empty CPDAG
2. **Forward (phase 1):** Add edges one by one until local maxima in BIC



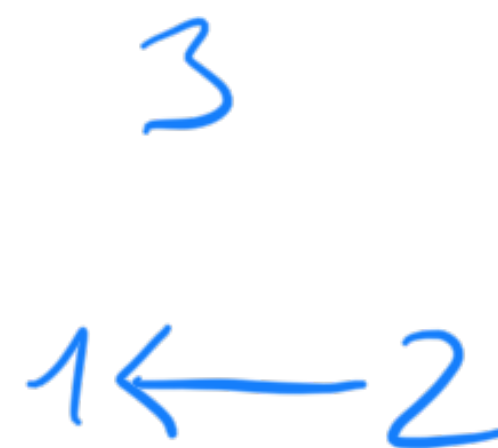
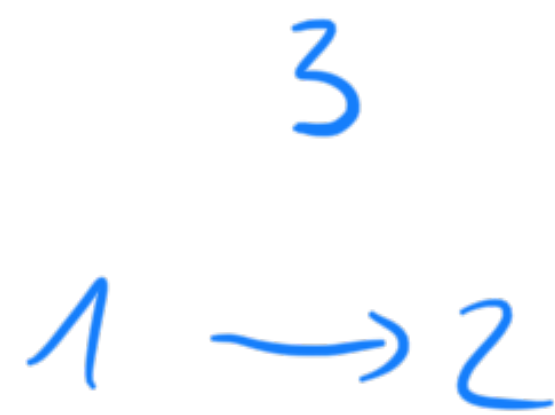
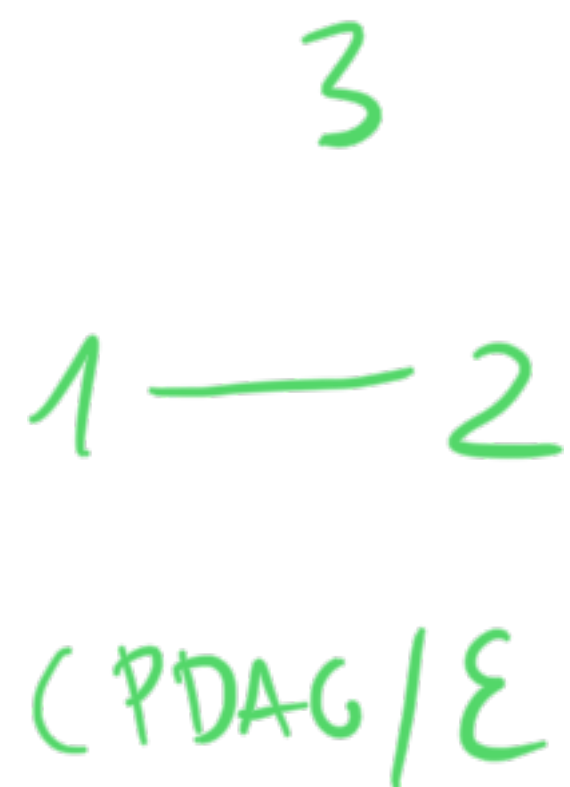
Add an edge
between 3 and 1

Step 2: for each G , enumerate all G' that can be created by adding a single edge

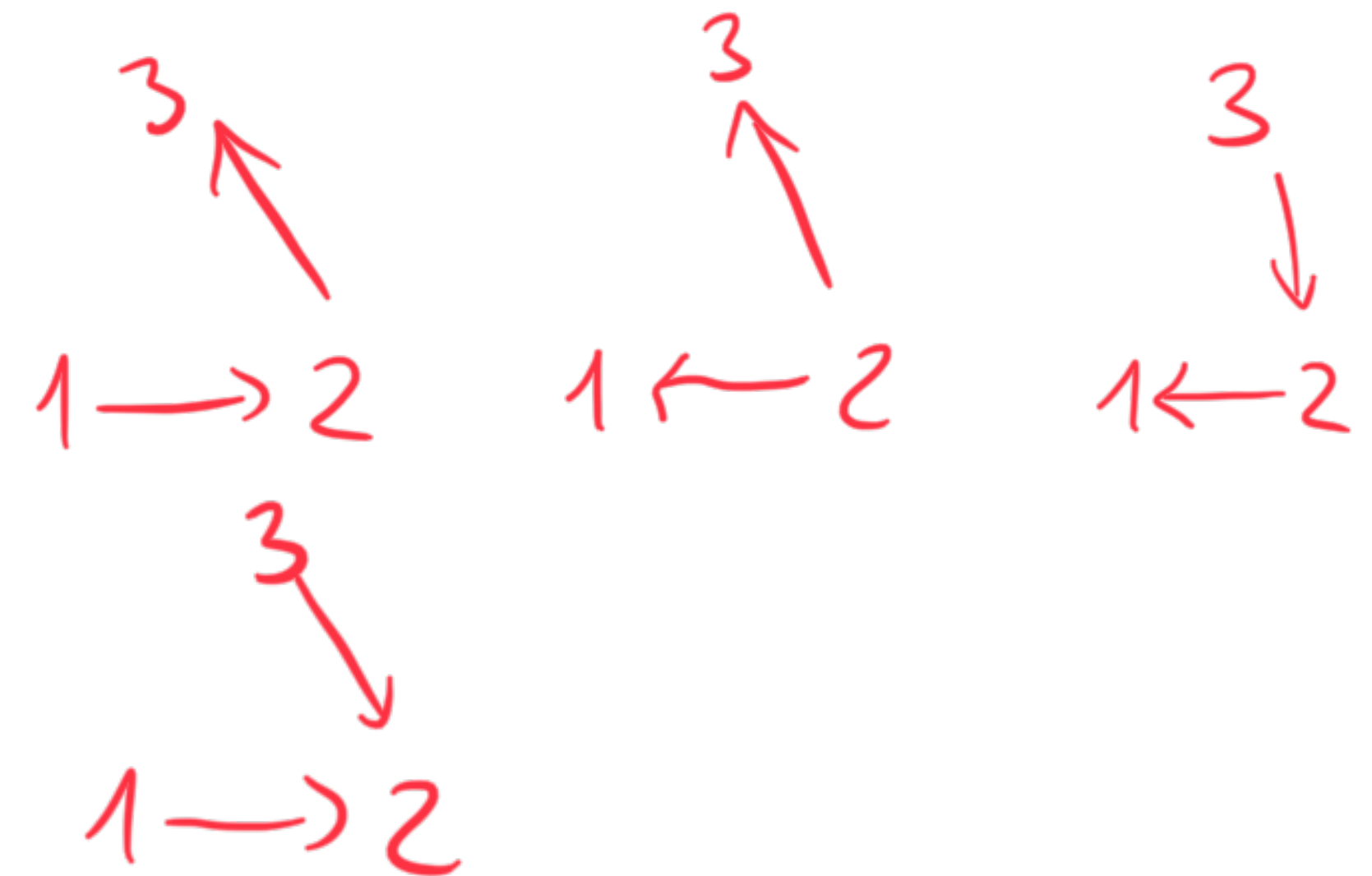


Greedy Equivalence Search (GES)

1. Start with empty CPDAG
2. **Forward (phase 1):** Add edges one by one until local maxima in BIC



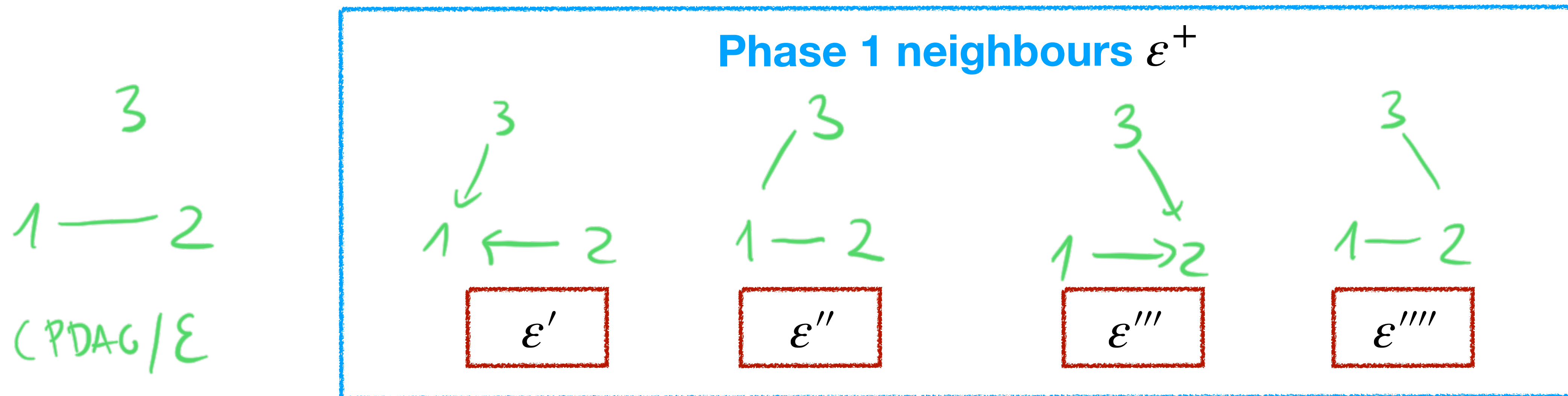
Add an edge
between 3 and 2



Step 2: for each G , enumerate all G' that can be created by adding a single edge

Greedy Equivalence Search (GES)

1. Start with empty CPDAG
2. **Forward (phase 1):** Add edges one by one until local maxima in BIC



Step 3: For each DAG G' add its CPDAG ϵ' to the list of neighbours ϵ^+

Greedy Equivalence Search - **phase 2**: example 2

$\mathcal{E}$:

$1 \rightarrow 2 \rightarrow 3$

$1-2-3$

$1 \leftarrow 2 \rightarrow 3$

$1 \leftarrow 2 \leftarrow 3$

$\Sigma^-(\mathcal{E})$:

Step 1: Enumerate all DAGs $G \in \mathcal{E}$ in the initial CPDAG $\mathcal{E}$

Greedy Equivalence Search - **phase 2:** example 2

$\mathcal{E}$:

$1 \rightarrow 2 \rightarrow 3$

$1-2-3$

$1 \leftarrow 2 \rightarrow 3$

$1 \leftarrow 2 \leftarrow 3$

$\Sigma^-(\mathcal{E})$:

$1 \quad 2 \rightarrow 3$

$1 \quad 2 \leftarrow 3$

$1 \rightarrow 2 \quad 3$

$1 \leftarrow 2 \quad 3$

Step 2: for each G , enumerate all G' that can be created by removing a single edge

Greedy Equivalence Search - **phase 2:** example 2

$\mathcal{E}$:

$1 \rightarrow 2 \rightarrow 3$

$1-2-3$

$1 \leftarrow 2 \rightarrow 3$

$1 \leftarrow 2 \leftarrow 3$

$\Sigma^-(\mathcal{E})$:

$1 \quad 2 \rightarrow 3$

$1 \quad 2 \leftarrow 3$

$1 \quad 2-3$

$1 \rightarrow 2 \quad 3$

$1 \leftarrow 2 \quad 3$

$1-2 \quad 3$

Step 3: For each DAG G' add its CPDAG $\mathcal{E}'$ to the list of neighbours $\mathcal{E}^-$

Score-based causal discovery vs restricted models

- Similarly to constraint-based causal discovery, also GES returns a CPDAG
- To find the true DAG we can
 - Add **background knowledge**
 - Add **experimental/interventional data** or **data from different environments**
 - What if we cannot do that?
- Under some special assumptions, **we can recover the true causal graph:**
 - Nonlinear models with additive noise: Additive Noise Models (ANM)
 - Linear Non-Gaussian Acyclic Models (LINGAM)

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

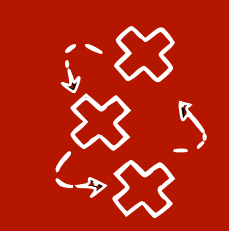
Interventional causal discovery / causal invariance

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

Additive noise models (ANMs)

$$\begin{cases} X = \varepsilon_X \\ Y = f(X) + \varepsilon_Y \end{cases}$$
$$\varepsilon_X \perp\!\!\!\perp \varepsilon_Y$$

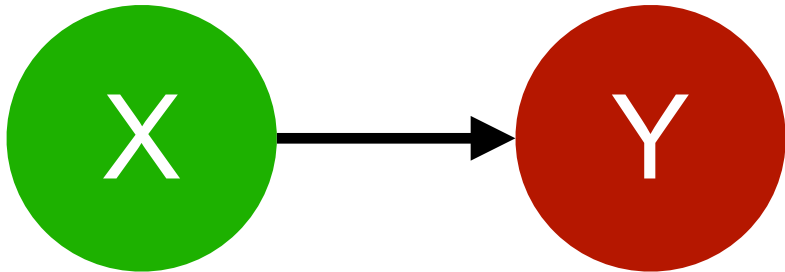
Here ε_i are not gaussian,
but rather uniform



Additive noise models (ANMs)

$$\begin{cases} X = \varepsilon_X \\ Y = f(X) + \varepsilon_Y \end{cases}$$

$\varepsilon_X \perp\!\!\!\perp \varepsilon_Y$



Here ε_i are not gaussian,
but rather uniform

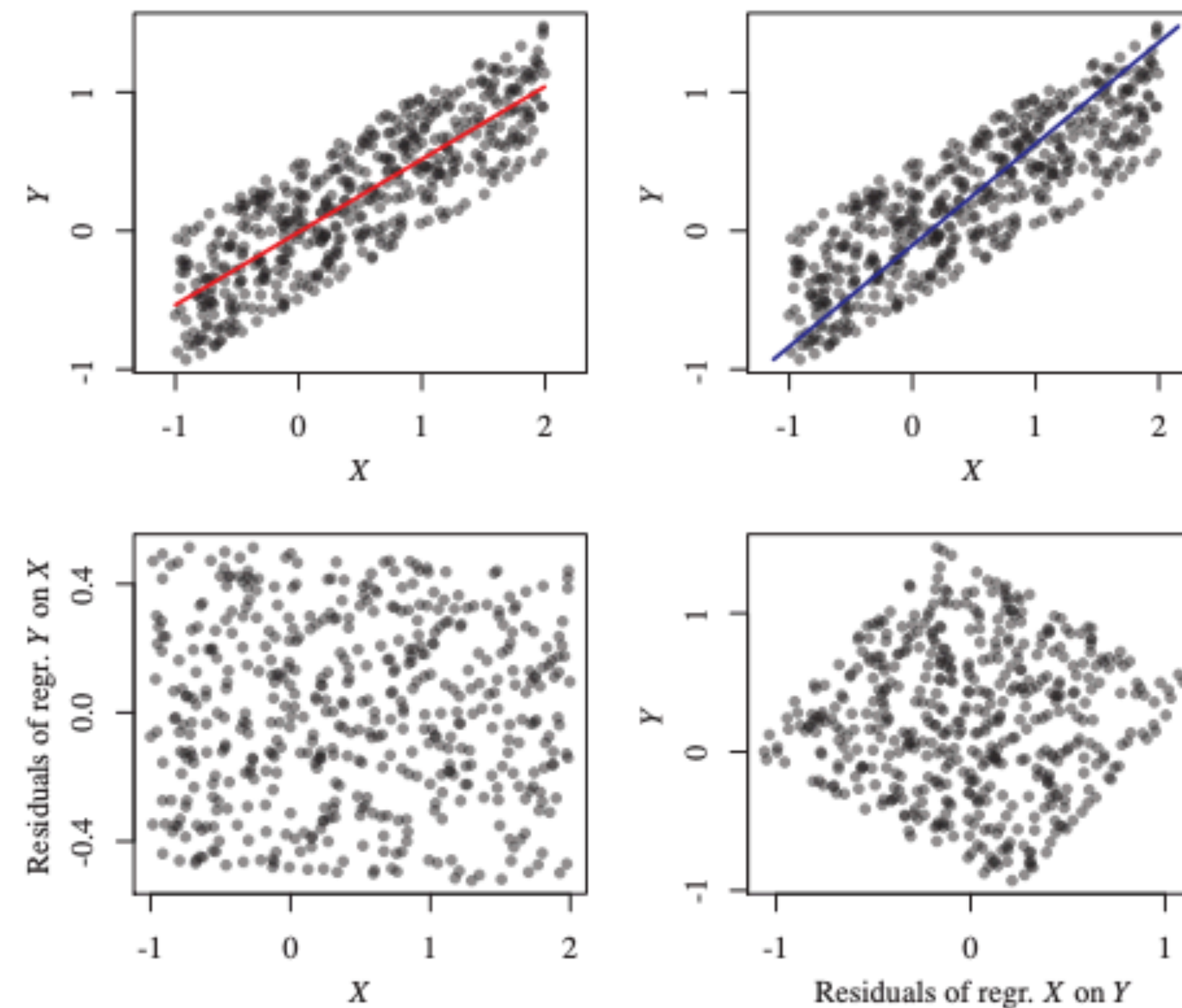


Figure 4.5: We are given a sample from the underlying distribution and perform a linear regression in the directions $X \rightarrow Y$ (left) and $Y \rightarrow X$ (right). The fitted functions are shown in the top row, the corresponding residuals are shown in the bottom row. Only the direction $X \rightarrow Y$ yields independent residuals; see also Figure 4.1.

Linear models with additive Gaussian noise

- If we have the linear SCM

$$Y = \alpha X + \epsilon_Y \text{ such that } \epsilon_Y \perp\!\!\!\perp X$$

Then there exists a $\beta \in \mathbb{R}$ and random variable ϵ_X such that:

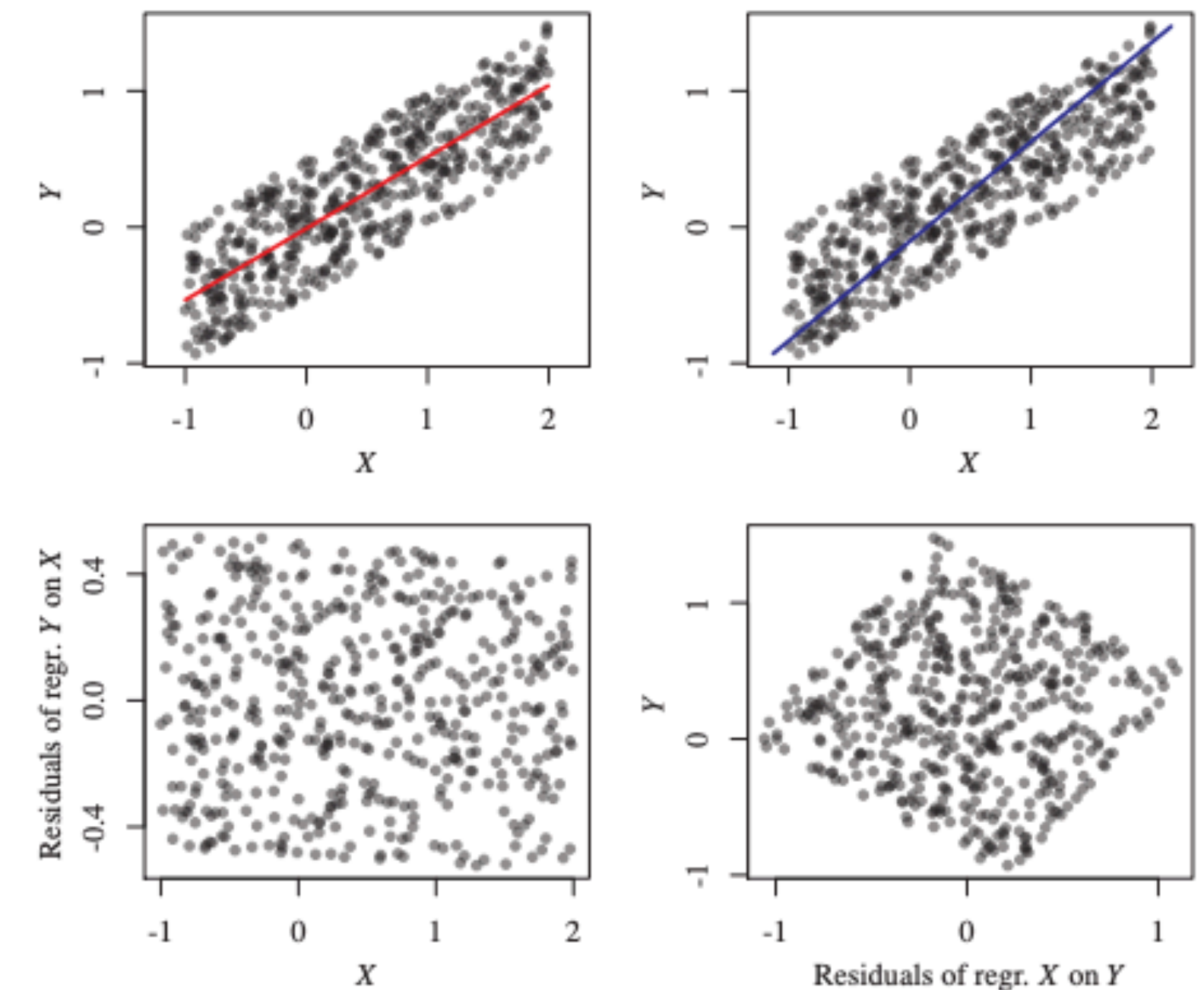
$$X = \beta Y + \epsilon_X \text{ such that } \epsilon_X \perp\!\!\!\perp Y$$

if and only if ϵ_Y and X are Gaussian

*This means we cannot
distinguish $X \rightarrow Y$
or $Y \rightarrow X$!*

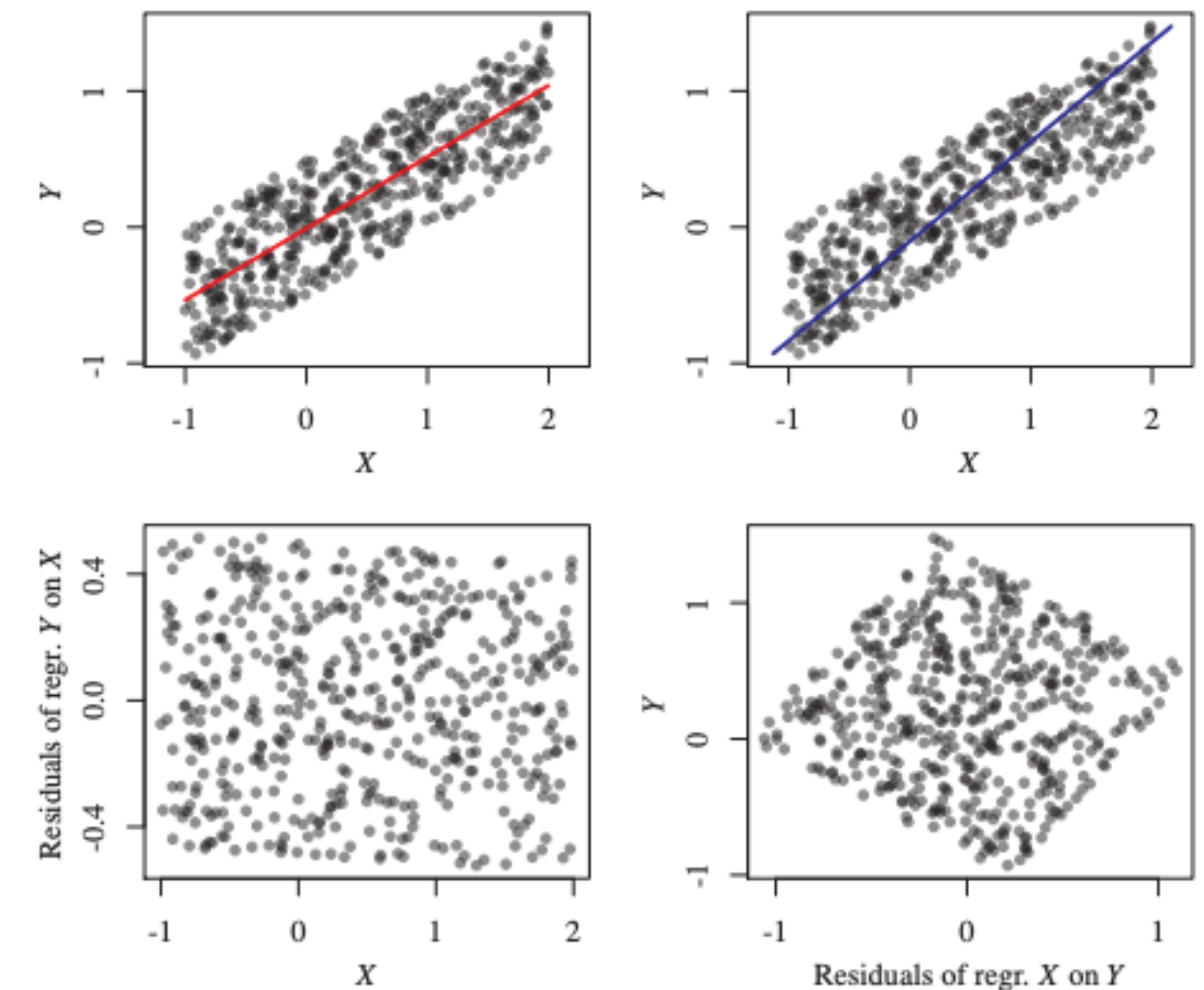
RESIT: regression with subsequent independence test

1. Regress X on Y with (possibly nonlinear) regression and estimate $\hat{f}_Y(X)$
2. Test if $(Y - \hat{f}_Y(X))$ is independent of X



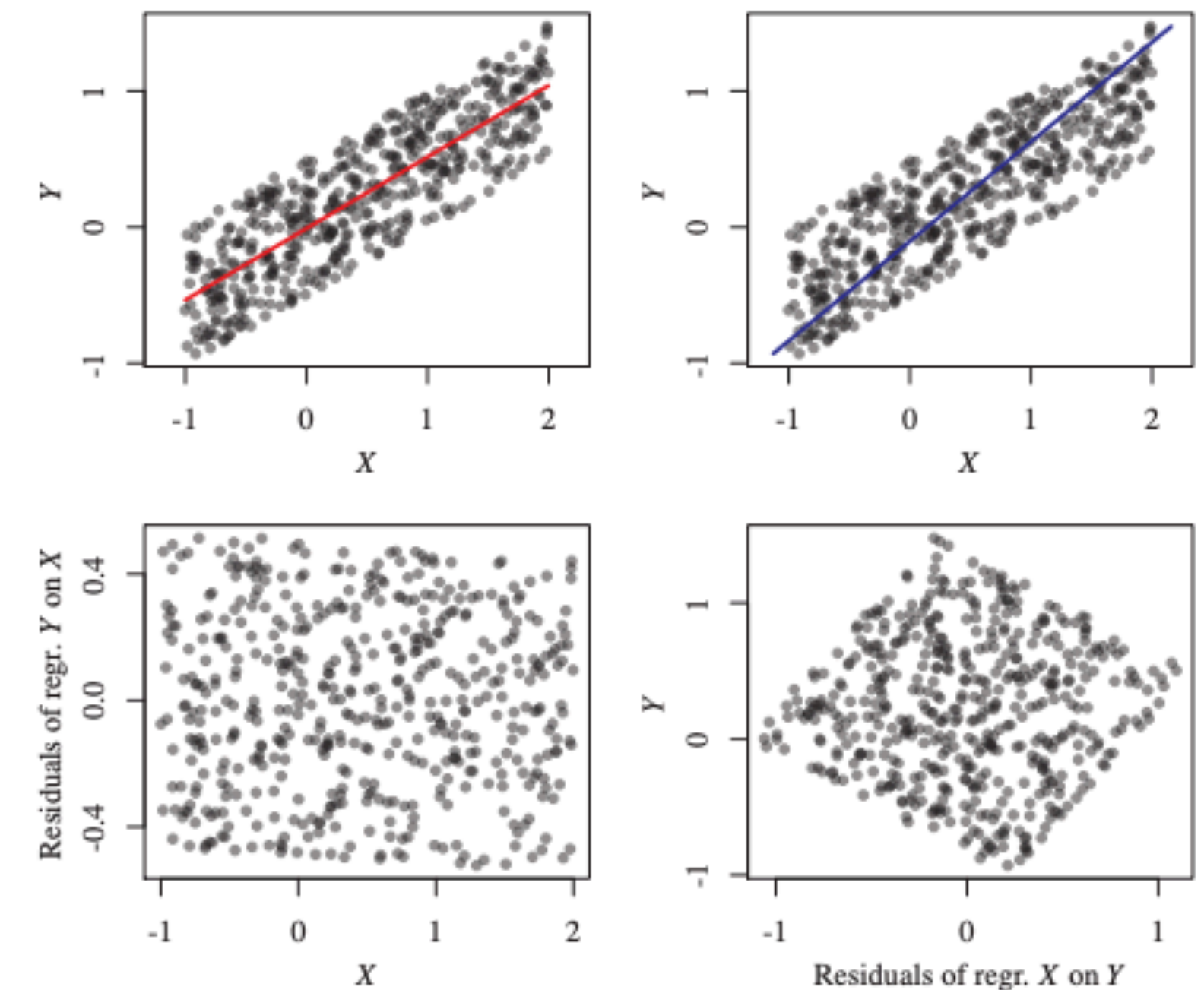
RESIT: regression with subsequent independence test

1. Regress X on Y with (possibly nonlinear) regression and estimate $\hat{f}_Y(X)$
2. Test if $(Y - \hat{f}_Y(X))$ is independent of X
3. Regress Y on X with (possibly nonlinear) regression and estimate $\hat{f}_X(Y)$
4. Test if $(X - \hat{f}_X(Y))$ is independent of Y



RESIT: regression with subsequent independence test

1. Regress X on Y with (possibly nonlinear) regression and estimate $\hat{f}_Y(X)$
2. Test if $(Y - \hat{f}_Y(X))$ is independent of X . **independent**
3. Regress Y on X with (possibly nonlinear) regression and estimate $\hat{f}_X(Y)$
4. Test if $(X - \hat{f}_X(Y))$ is independent of Y
5. If they are dependent in only one direction, if the other **independent** direction is **causal**



Extensions

$$\begin{cases} X = \varepsilon_x \\ Y = f(X) + \varepsilon_y \end{cases}$$

$\varepsilon_x \perp\!\!\!\perp \varepsilon_y$

$$\begin{cases} X = \varepsilon_x \\ Y = g(f(X) + \varepsilon_y) \end{cases}$$

POST-LINEAR

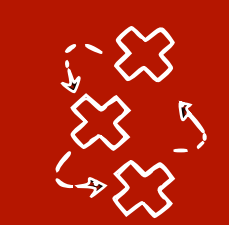
For more details check Chapter 4 in the book: http://web.math.ku.dk/~peters/jonas_files/ElementsOfCausalInference.pdf

Linear Non Gaussian Acyclic Models (LINGAM)

- We can write a linear SCM in matrix notation:

$$X = \mathbf{B}X + \varepsilon \text{ with } \mathbf{B} \in \mathbb{R}^{p \times p}, X \in \mathbb{R}^p, \varepsilon \in \mathbb{R}^p$$

- For Gaussian noise, we cannot distinguish the direction, but **for non-Gaussian noises we can**
 - Assume they are mean zero non Gaussian with positive variance
 - **We don't need faithfulness!** (So it can work on cancelling paths, etc)



Linear Non Gaussian Acyclic Models (LINGAM)

- We can write a linear SCM in matrix notation:

$$X = \mathbf{B}X + \varepsilon \text{ with } \mathbf{B} \in \mathbb{R}^{p \times p}, X \in \mathbb{R}^p, \varepsilon \in \mathbb{R}^p$$

- Because of acyclicity we can show that we can rewrite $\mathbf{B}$ as **strictly lower triangular** by **permuting the variables using a causal ordering**

The diagram illustrates the transformation of matrix $\mathbf{B}$ into a strictly lower triangular matrix $\mathbf{B}_\pi$ using a causal ordering. It shows the following steps:

- A matrix $\mathbf{B}_\pi$ with a blue border containing green 'x' marks in the lower triangular part and blue '0' marks in the upper triangular part.
- A causal ordering $3 \rightarrow 1 \rightarrow 2$ shown with green arrows.
- A matrix $\mathbf{B}$ with a green border, where the upper triangular part is zero and the lower triangular part contains elements B_{21} , B_{13} , and B_{23} .
- A matrix $\mathbf{B}_\pi$ with a red border, where the upper triangular part is zero and the lower triangular part contains elements B_{13} , B_{21} , and B_{23} , corresponding to the causal ordering x_3, x_1, x_2 .

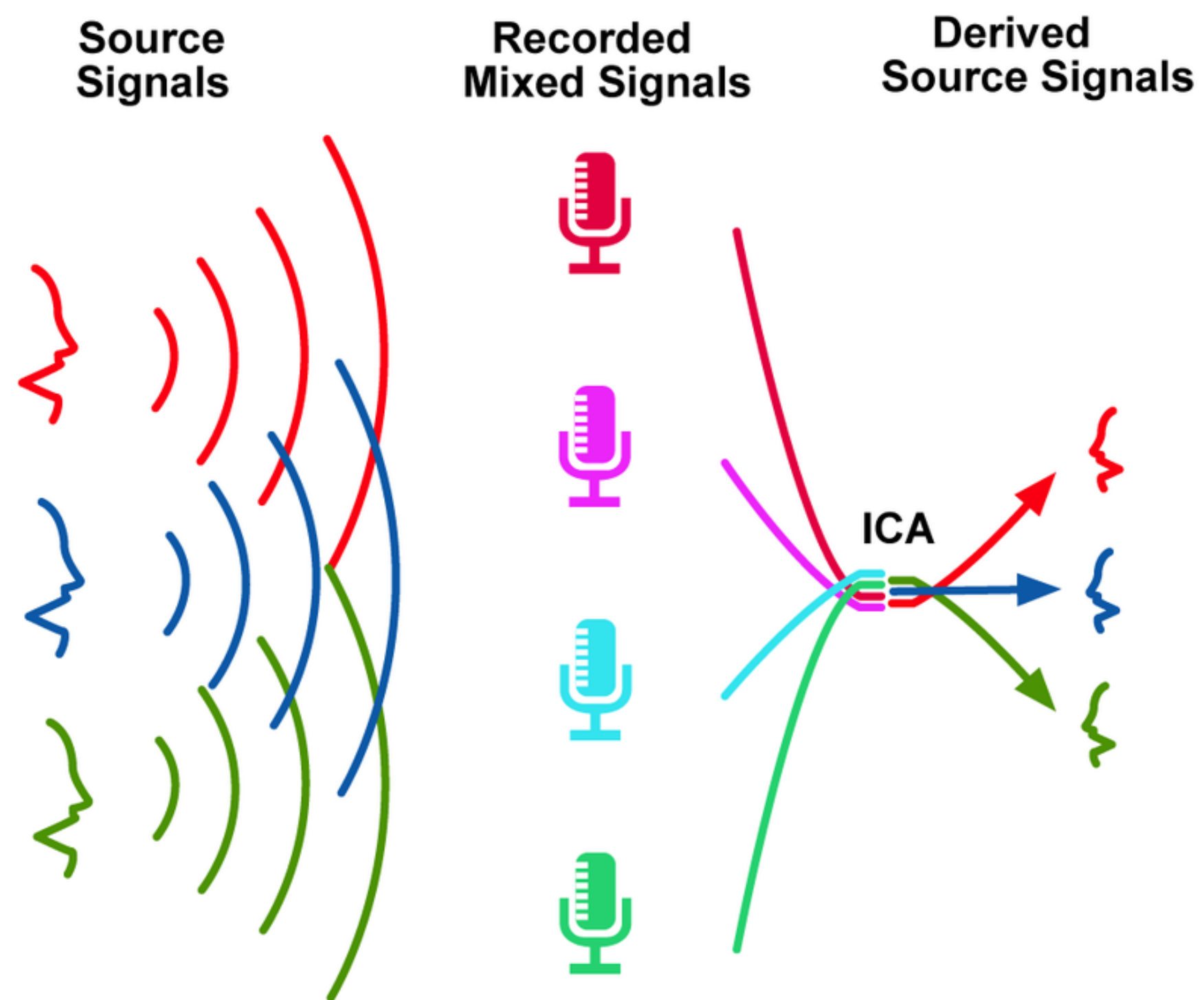
Linear Non Gaussian Acyclic Models (LINGAM)

- We can write a linear SCM in matrix notation:

$$X = \mathbf{B}X + \varepsilon \text{ with } \mathbf{B} \in \mathbb{R}^{p \times p}, X \in \mathbb{R}^p, \varepsilon \in \mathbb{R}^p$$

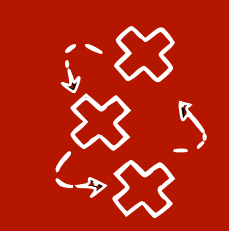
- Because of acyclicity we can show that we can rewrite $\mathbf{B}$ as **strictly lower triangular** by **permuting the variables using a causal ordering**
- **Goal:** estimate $\mathbf{B}$ from data (which also identifies the DAG)
- ICA-LINGAM, DirectLINGAM (and many others)

Independent Component Analysis (ICA)

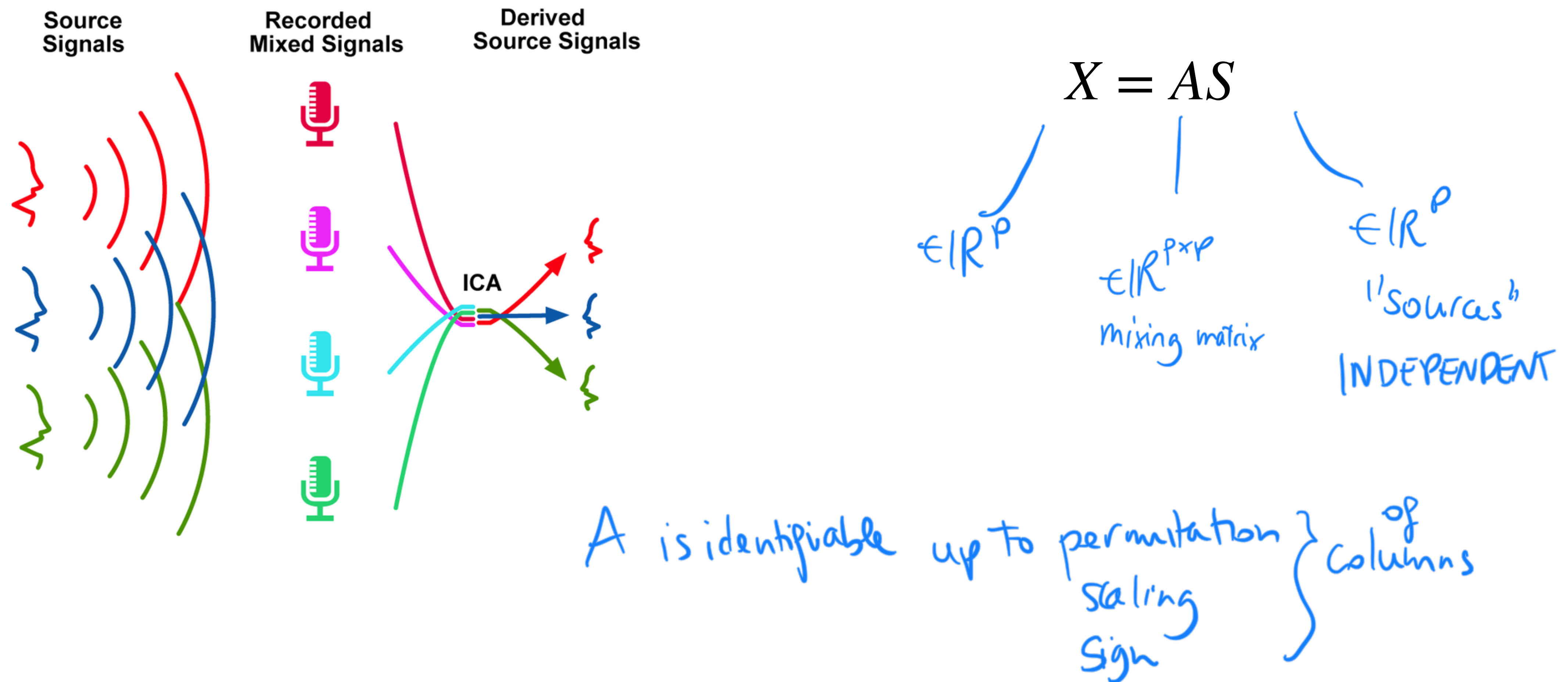


$$X = AS$$

$\in \mathbb{R}^p$ $\in \mathbb{R}^{p \times p}$ $\in \mathbb{R}^p$
 mixing matrix "sources"
 INDEPENDENT



Independent Component Analysis (ICA)



ICA-LINGAM

- A linear SCM $X = \mathbf{B}X + \varepsilon$ can be rewritten as $(I - \mathbf{B})X = \varepsilon$ and

$$X = (I - \mathbf{B})^{-1} \varepsilon$$

mixing matrix
 A

Sources S

ICA-LINGAM

- A linear SCM $X = \mathbf{B}X + \varepsilon$ can be rewritten as $(I - \mathbf{B})X = \varepsilon$ and

$$X = (I - \mathbf{B})^{-1} \varepsilon$$

mixing matrix
 A



up to permutation, scaling...

Sources S

LINGAM: independent non Gaussian

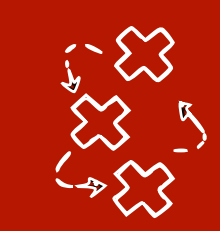
$\equiv$ ICA

ICA-LINGAM

1. Given dataset $D = \{x_{\mathbf{V}}^1, x_{\mathbf{V}}^2, \dots, x_{\mathbf{V}}^n\}$ use ICA to estimate $W = A^{-1} = (I - \mathbf{B})$
2. Find **unique permutation** of rows of W such that $\tilde{W}$ does not have zeros on diagonal
3. Divide each row in $\tilde{W}$ by its diagonal element (so we get all 1 on the diagonal)
4. Compute $\hat{\mathbf{B}} = I - \tilde{W}$
5. Find **causal ordering** described by the permutation matrix P by making $\tilde{\mathbf{B}} = P\hat{\mathbf{B}}P^T$ as close as possible to strictly lower triangular

Other types of data

- All of the methods we saw until now use only **observational data**
- For restricted models this works well, since if the assumptions they make are true, then they can recover the true causal graph
- For score-based and constraint-based models, there are more advanced methods that can also use interventional data
 - For example for GES there is GIES
- If we don't know the targets of the interventions -> Joint Causal Inference



Causal discovery - this class

Constraint-based causal discovery

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

Score-based causal discovery

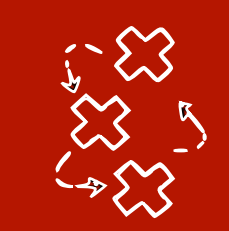
- Penalised likelihood
- Observational data
- Output: MEC
- GES, MMHC

Restricted models

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

Interventional causal discovery / causal invariance

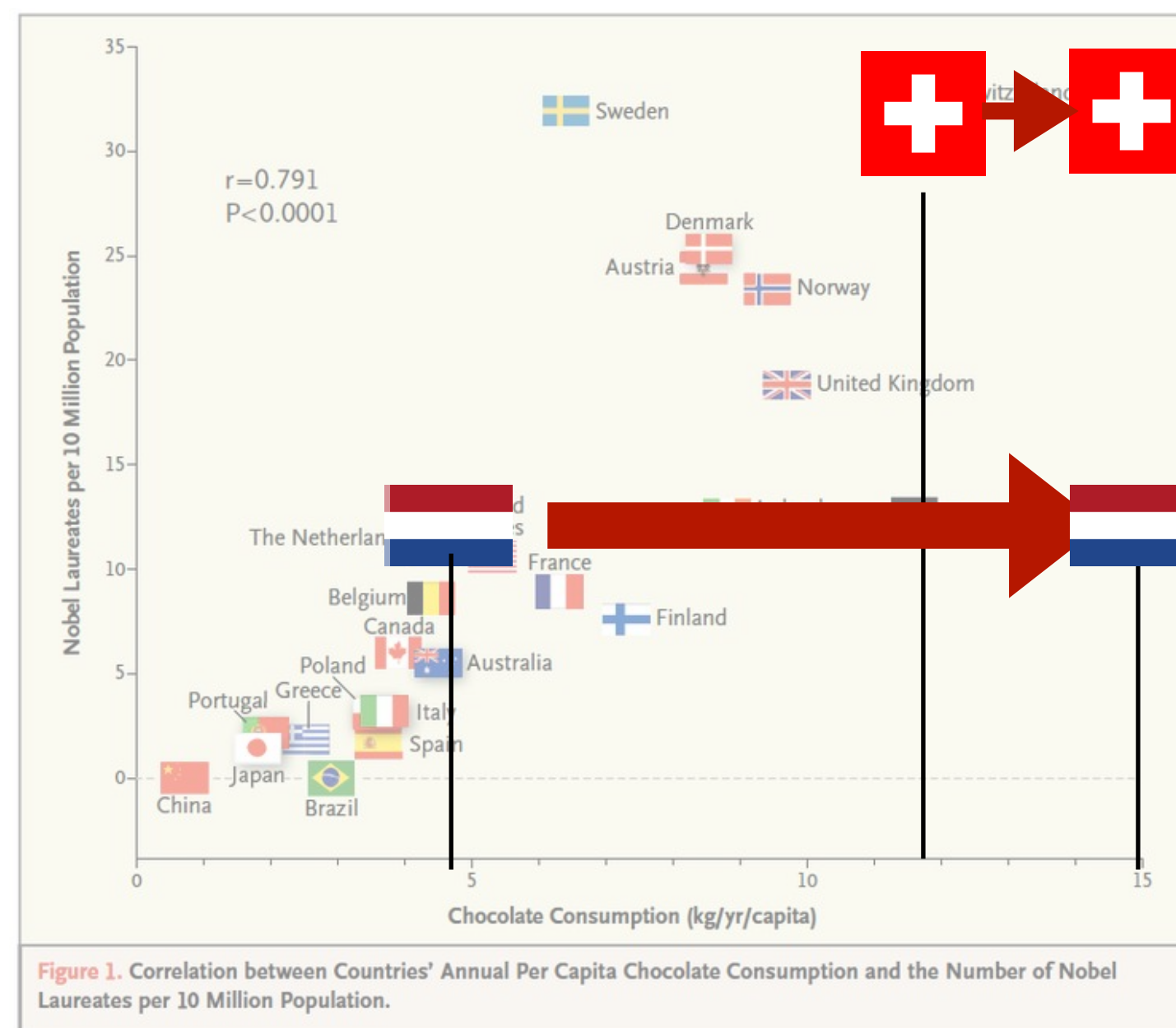
- Observational and Interventional data
- Output: parents of Y, I-MEC
- JCI



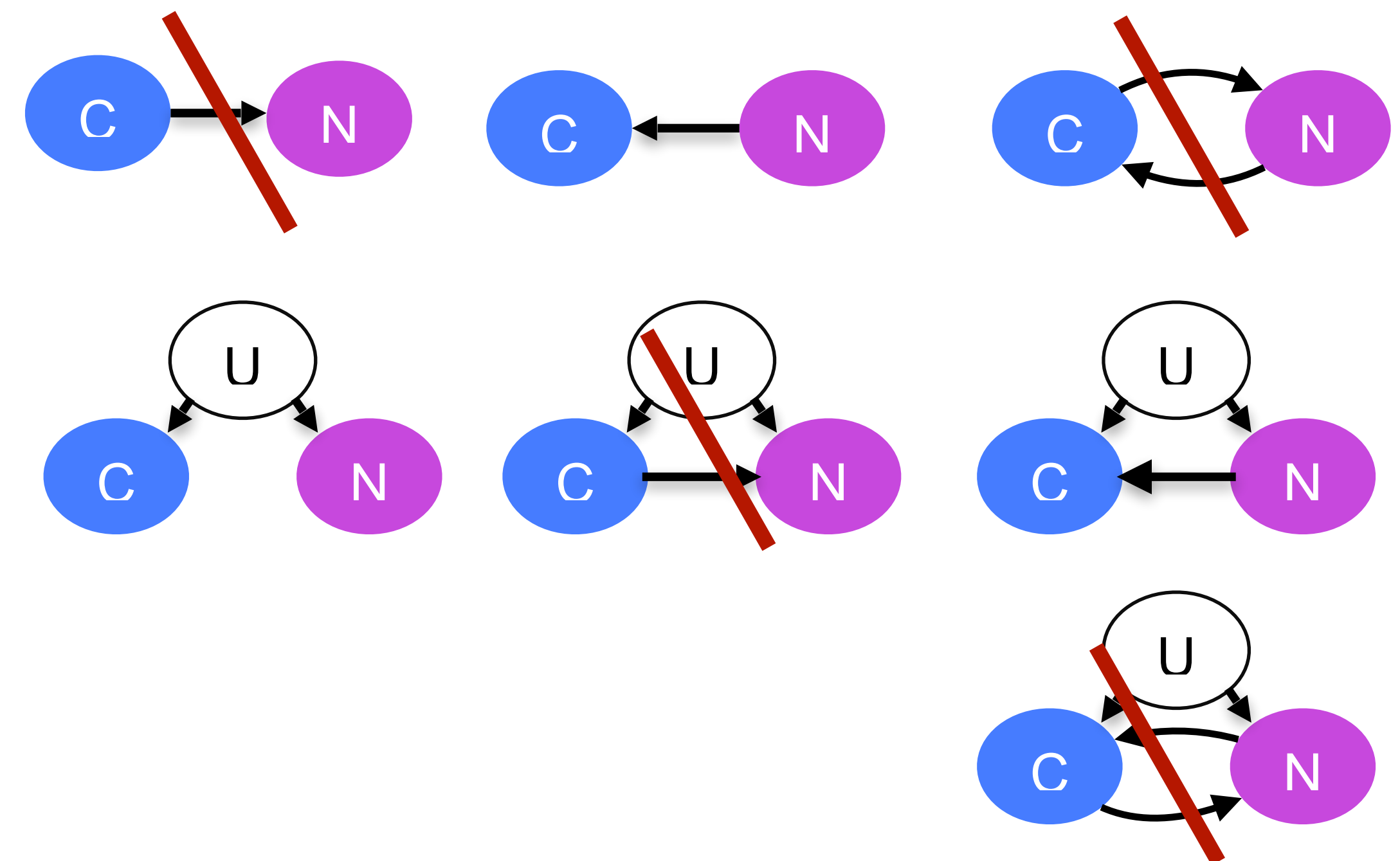
Learning from interventional data - intuition

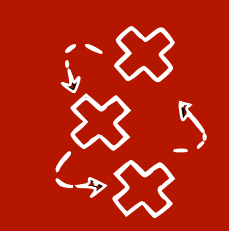
Until now we have only used **observational data**.

Hypothetical world: we perform the experiment and see these results:



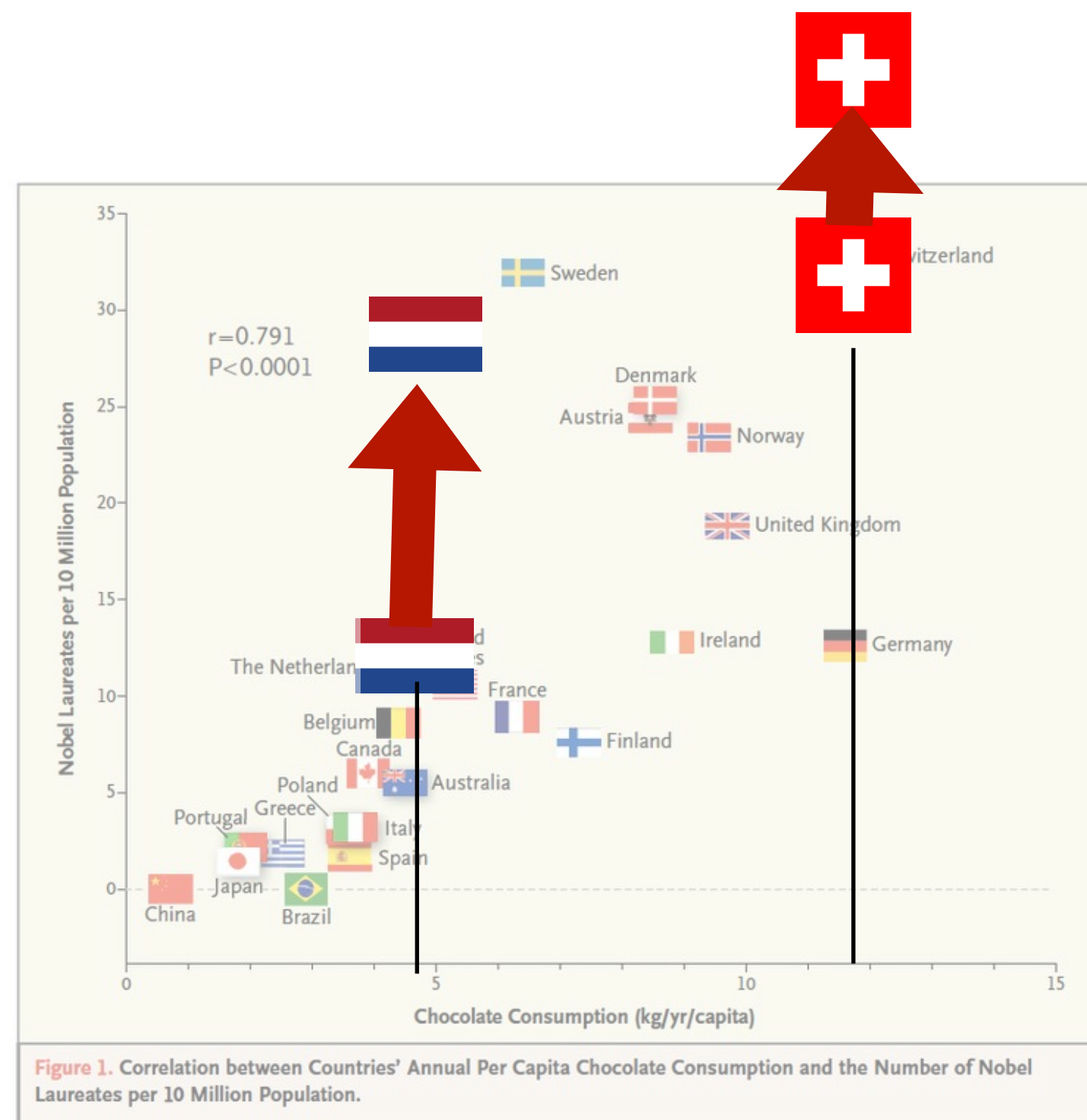
Chocolate does not cause Nobel prizes



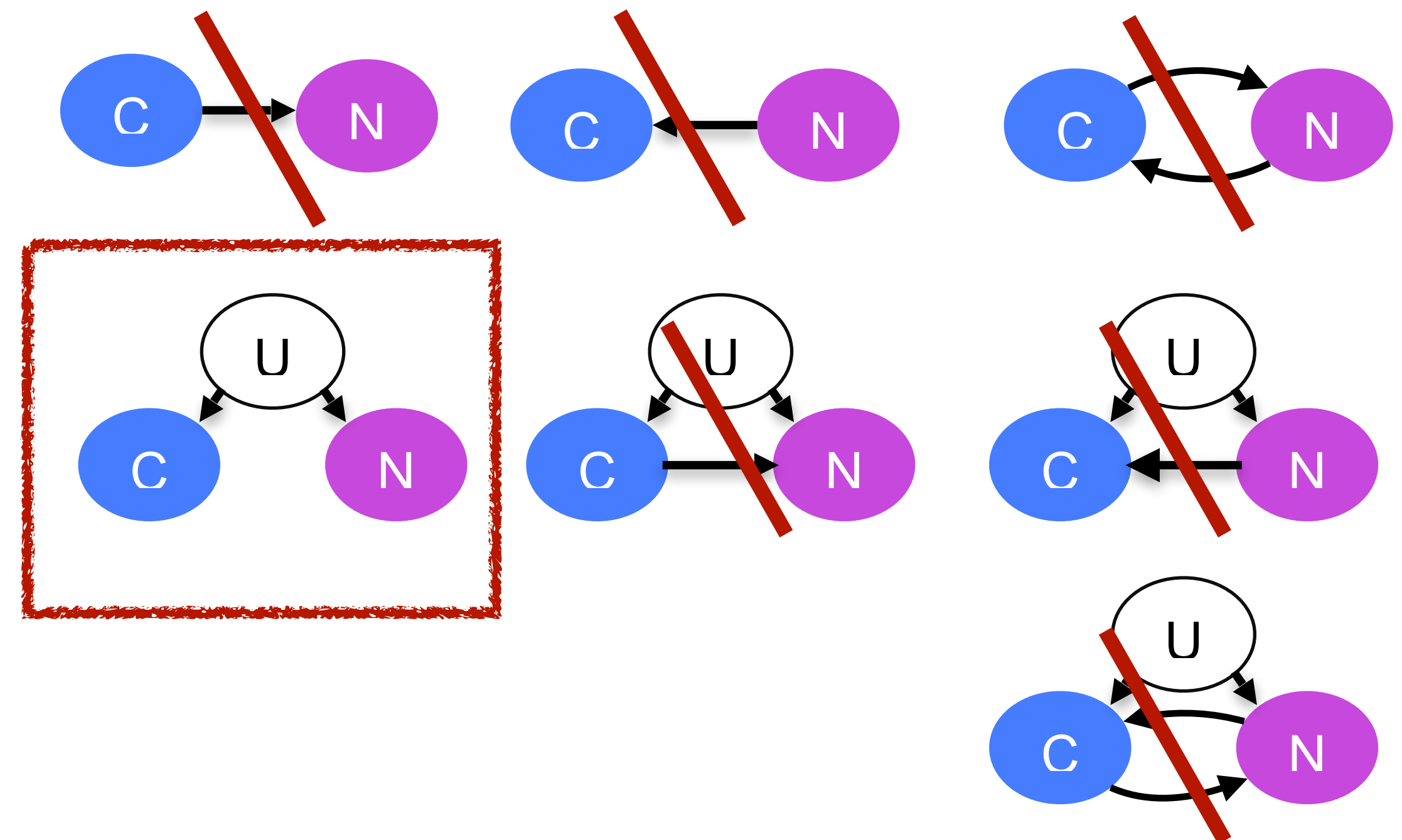


Learning from interventional data - intuition

Hypothetical world: we perform another experiment and see these results:



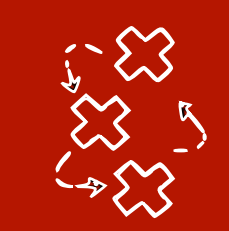
Nobel does not cause chocolate



Joint Causal Inference from Multiple Contexts

Joris M. Mooij, Sara Magliacane, Tom Claassen

- We represent different distributions (including interventional) as an **unknown joint causal graph** (*possibly cyclic or with latent confounders*)
- We **add context variables** so we can **disentangle** changes in distribution across the datasets



Joint Causal Inference intuition

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

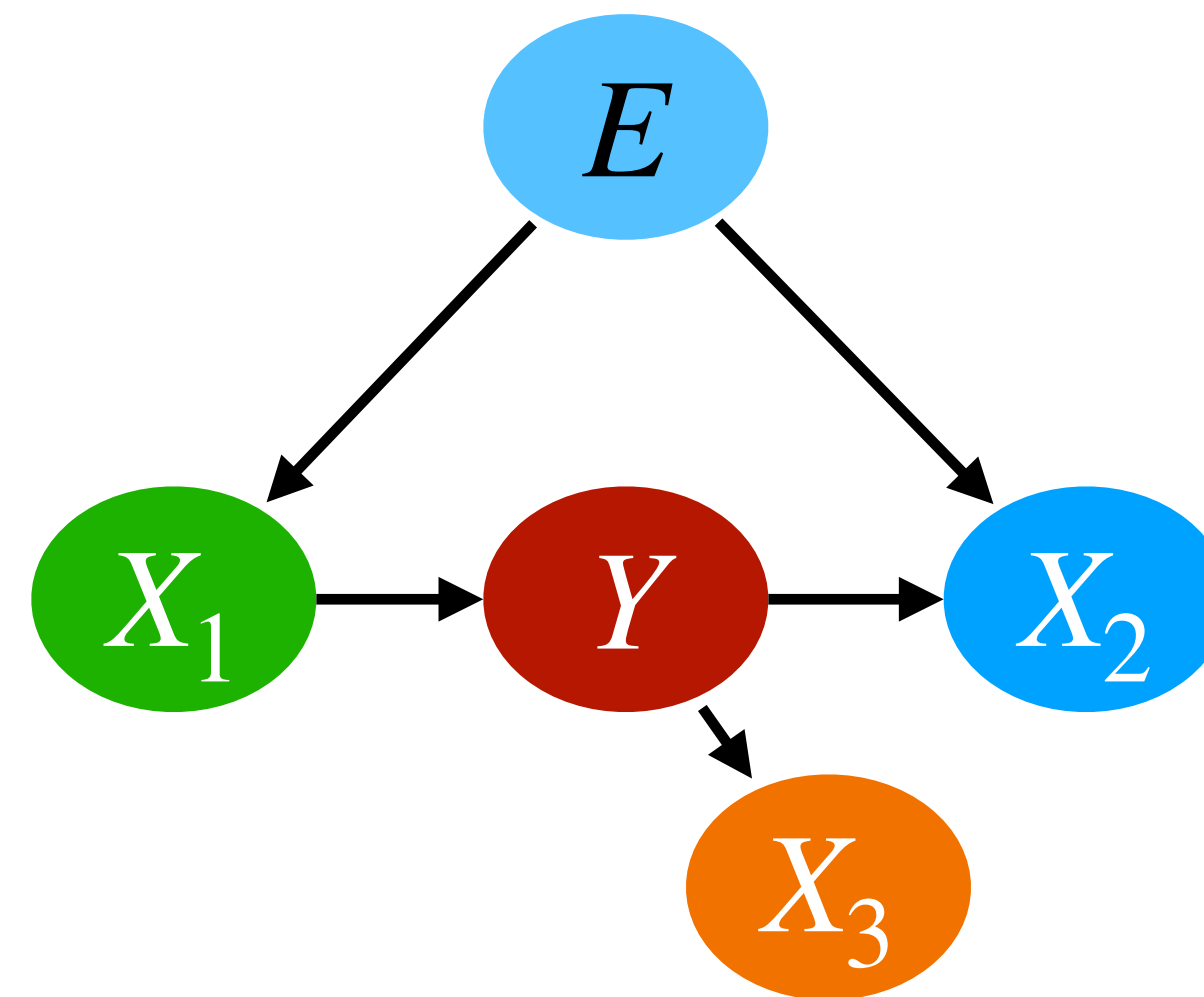
$$E = 0$$

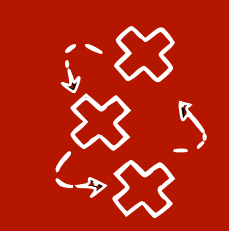
$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 100 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

$$E = 1$$

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = 10Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

$$E = 2$$





Joint Causal Inference intuition

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

$$E = 0$$

$$\begin{aligned} C_1 &= 0 \\ C_2 &= 0 \end{aligned}$$

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 100 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

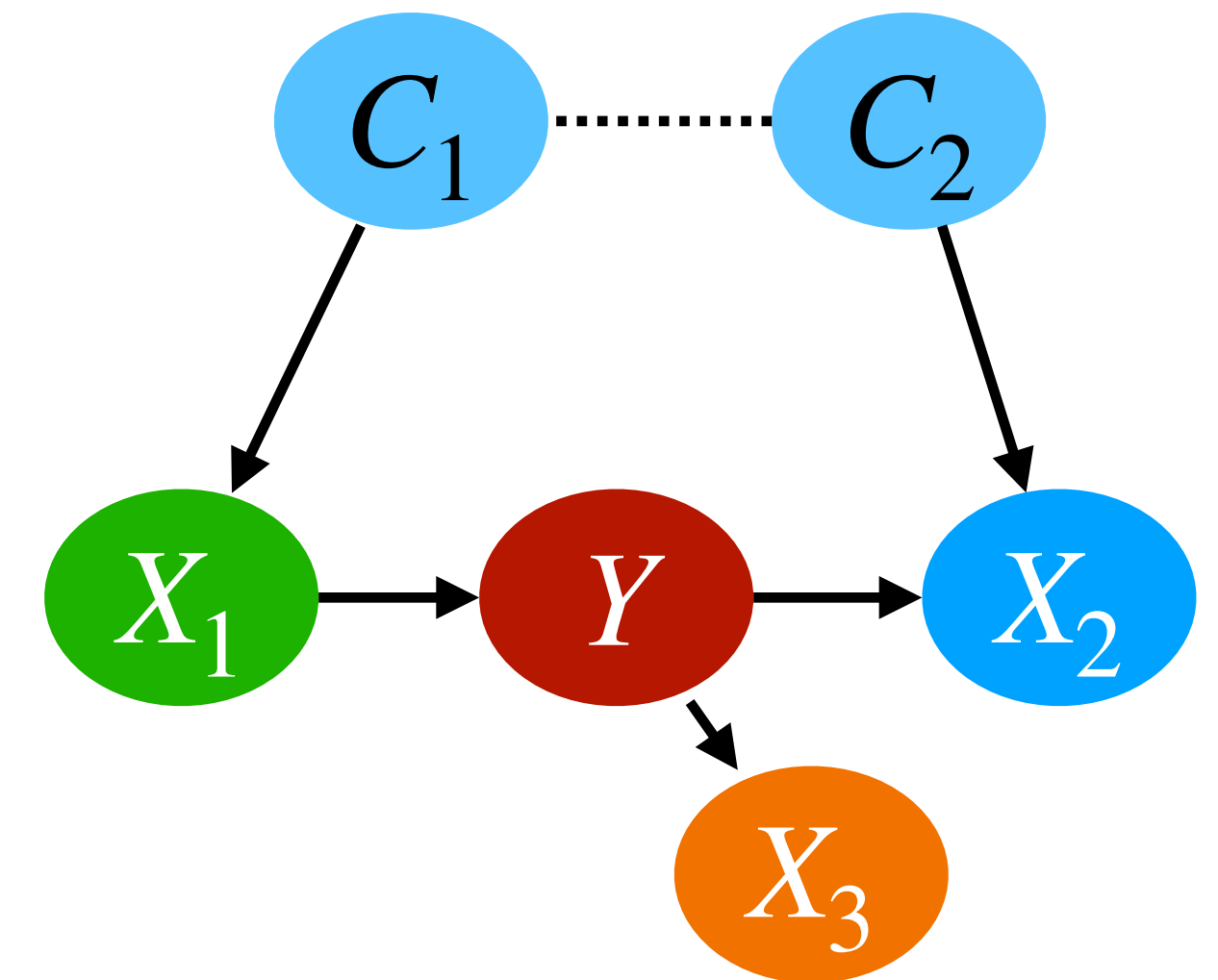
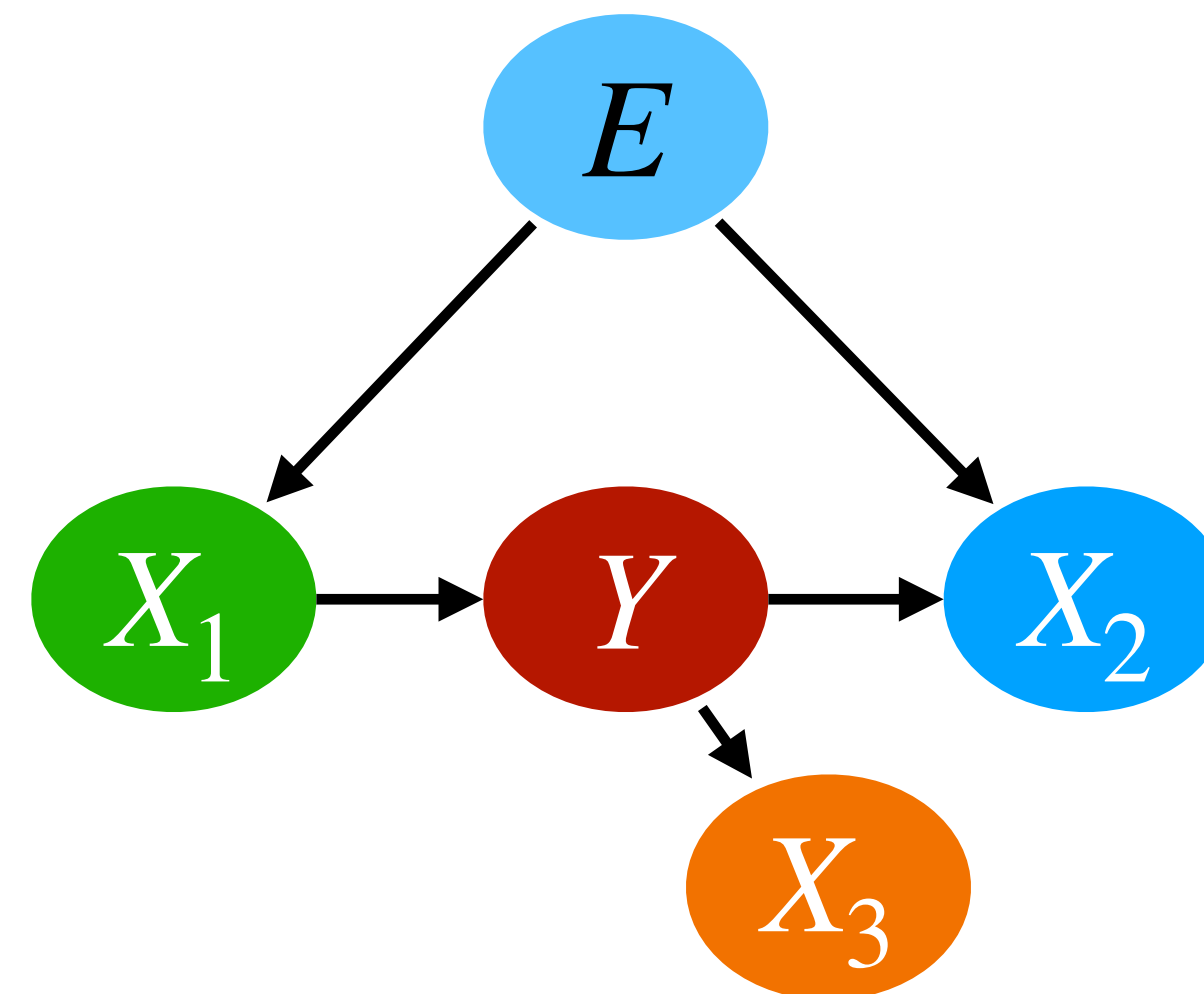
$$E = 1$$

$$\begin{aligned} C_1 &= 1 \\ C_2 &= 0 \end{aligned}$$

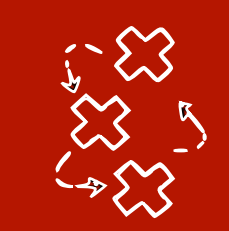
$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = 10Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

$$E = 2$$

$$\begin{aligned} C_1 &= 0 \\ C_2 &= 1 \end{aligned}$$



Adding context variables C_1 and C_2 helps disentangle the changes in each environment



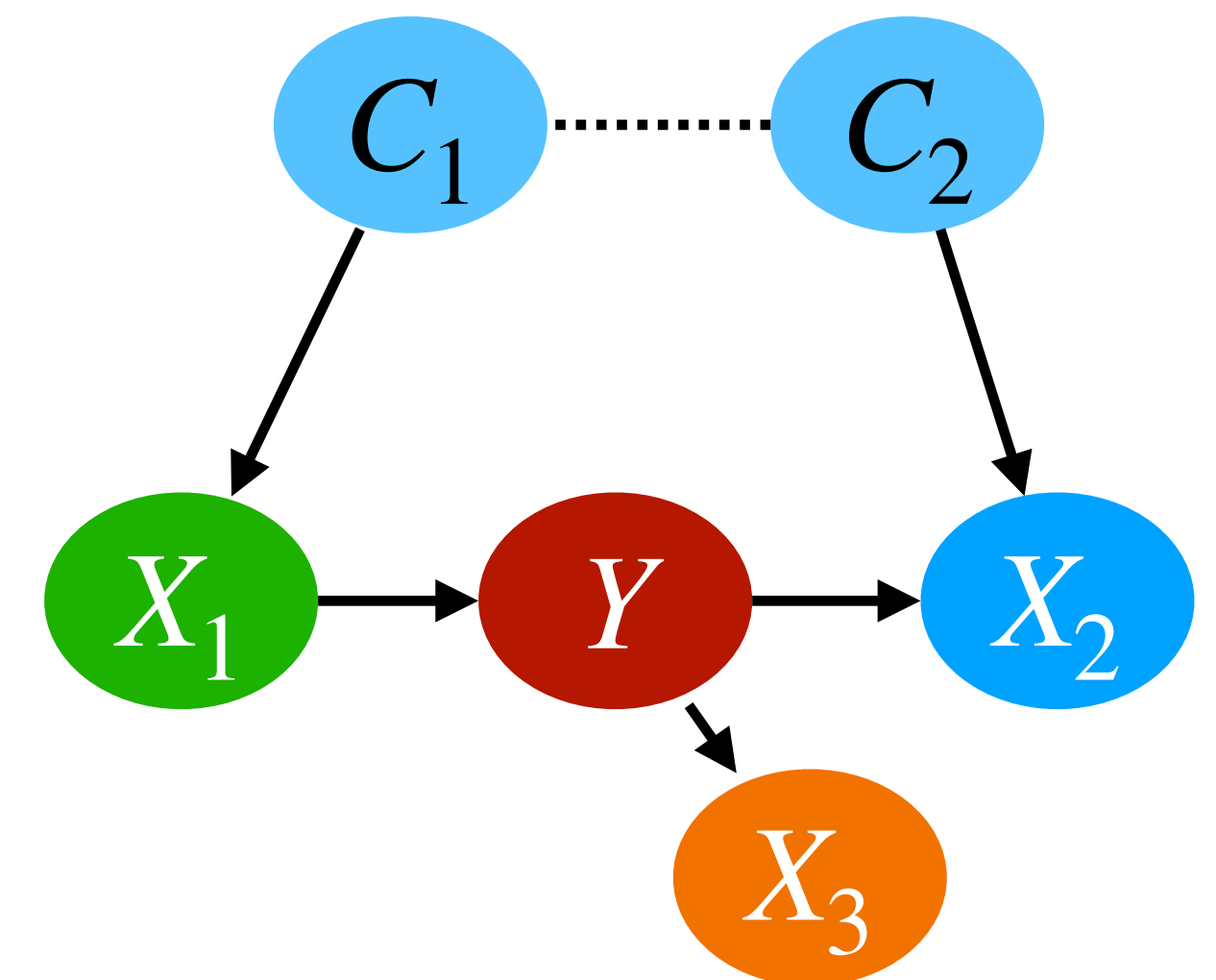
Joint Causal Inference intuition

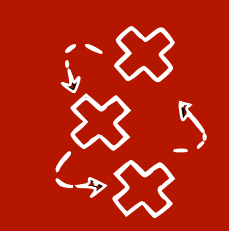
$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases} \quad \begin{array}{l} E = 0 \\ C_1 = 0 \\ C_2 = 0 \end{array}$$

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 100 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases} \quad \begin{array}{l} E = 1 \\ C_1 = 1 \\ C_2 = 0 \end{array}$$

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = 10Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases} \quad \begin{array}{l} E = 2 \\ C_1 = 0 \\ C_2 = 1 \end{array}$$

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = \begin{cases} 10 + \epsilon_1 & \text{if } C_1 = 0 \\ 100 + \epsilon_1 & \text{if } C_1 = 1 \end{cases} \\ Y = 3X_1 + \epsilon_Y \\ X_2 = \begin{cases} -2Y + \epsilon_2 & \text{if } C_2 = 0 \\ 10Y + \epsilon_2 & \text{if } C_2 = 1 \end{cases} \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$





Joint Causal Inference intuition

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

$$E = 0$$

$$\begin{aligned} C_1 &= 10 \\ C_2 &= -2 \end{aligned}$$

$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 100 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = -2Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

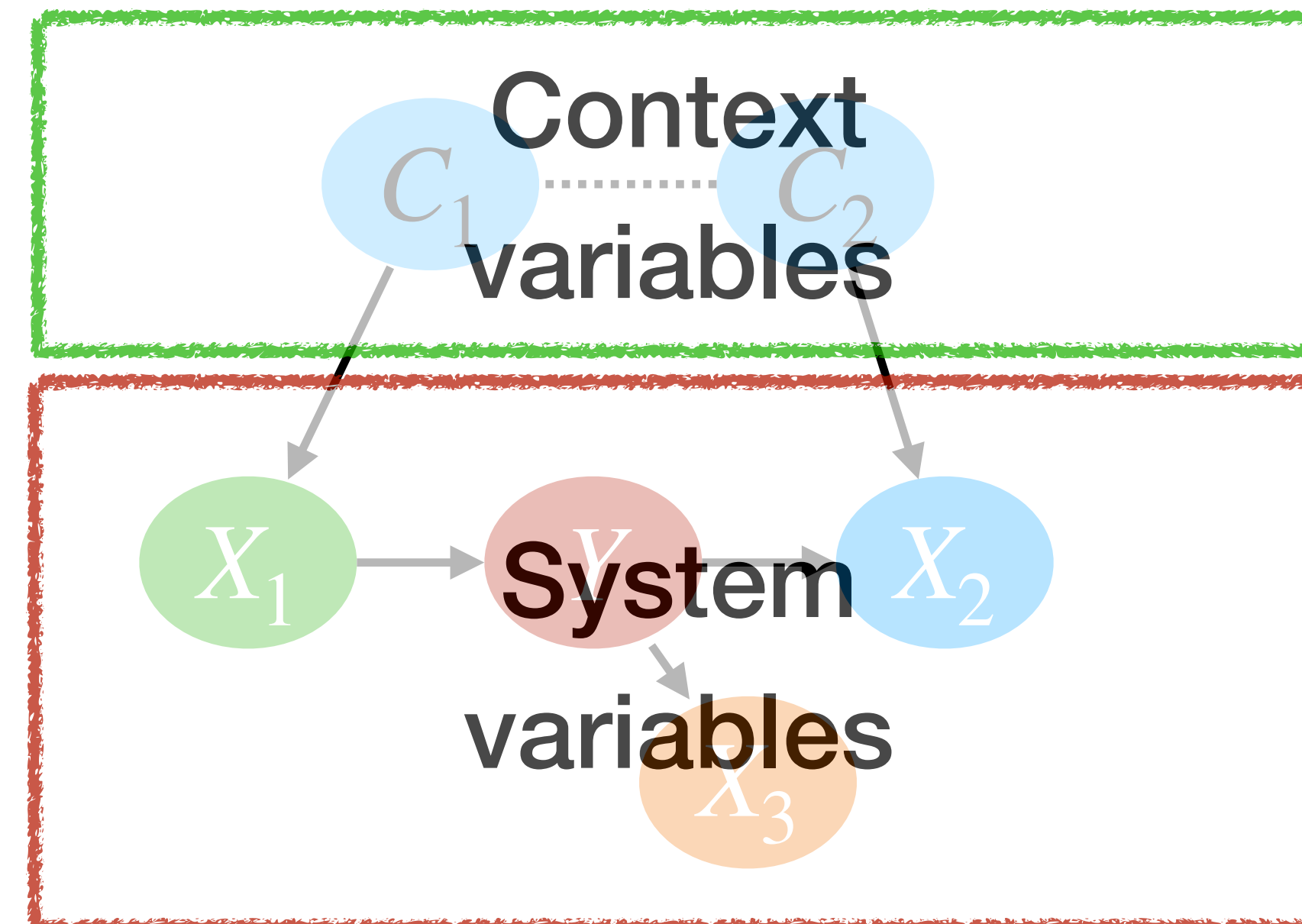
$$E = 1$$

$$\begin{aligned} C_1 &= 100 \\ C_2 &= -2 \end{aligned}$$

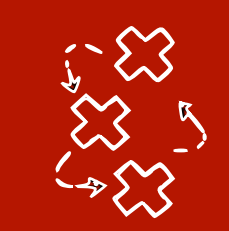
$$\begin{cases} \epsilon_1, \epsilon_2, \epsilon_3, \epsilon_Y \sim \mathcal{N}(0,1) \\ X_1 = 10 + \epsilon_1 \\ Y = 3X_1 + \epsilon_Y \\ X_2 = 10Y + \epsilon_2 \\ X_3 = 2Y + 0.1\epsilon_3 \end{cases}$$

$$E = 2$$

$$\begin{aligned} C_1 &= 10 \\ C_2 &= 10 \end{aligned}$$



The context variables C_1 and C_2 can be also descriptive of the intervention in each environment

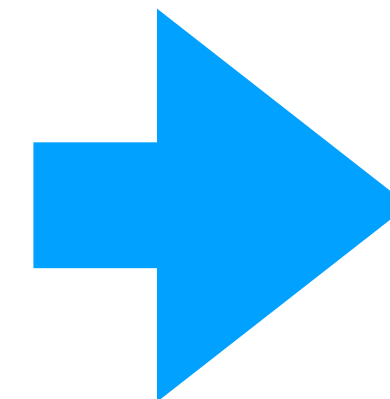


Joint Causal Inference from Multiple Contexts

Joris M. Mooij, Sara Magliacane, Tom Claassen

- We represent different distributions (including interventional) as an **unknown joint causal graph** (possibly cyclic or with latent confounders)
- We **add context variables** so we can **disentangle** changes in distribution across the datasets

	X1	X2	X3
Normal	0,1	2	0
Normal	0,2	3	0
	X1	X2	X3
Gene A	3,1	2	1
Gene A	3,2	3	1
	X1	X2	X3
Gene B	0.2	1	0
Gene B	0.3	1	1
Gene B	0.3	2	1
Gene B	0.4	1	1



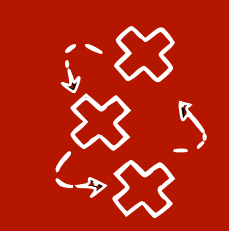
C1	C2	X1	X2	X3
0	0	0,1	2	0
0	0	0,2	3	0
0	0	1,1	2	1
0	0	0,1	3	0
1	0	3,1	2	1
1	0	3,2	3	1
1	0	4	1	1
1	0	3,2	3	1
0	1	0,2	1	0
0	1	0,3	1	1
0	1	0,3	2	1
0	1	0,4	1	1

Joint Causal Inference from Multiple Contexts

Joris M. Mooij, Sara Magliacane, Tom Claassen

- We **add context variables** so we can **disentangle** changes in distribution across the datasets (and optionally background knowledge, e.g. context variables are uncaused)
- We can reuse **any standard method for observational data** that fits any chosen assumptions

C1	C2	X1	X2	X3
0	0	0,1	2	0
0	0	0,2	3	0
0	0	1,1	2	1
0	0	0,1	3	0
1	0	3,1	2	1
1	0	3,2	3	1
1	0	4	1	1
1	0	3,2	3	1
0	1	0,2	1	0
0	1	0,3	1	1
0	1	0,3	2	1
0	1	0,4	1	1

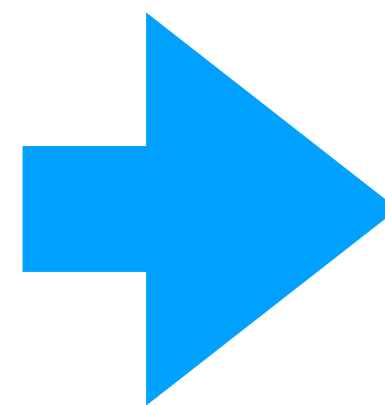


Joint Causal Inference from Multiple Contexts

Joris M. Mooij, Sara Magliacane, Tom Claassen

- We **add context variables** so we can **disentangle** changes in distribution across the datasets (and optionally background knowledge, e.g. context variables are uncaused)
- We can reuse **any standard method for observational data** that fits any chosen assumptions

C1	C2	X1	X2	X3
0	0	0,1	2	0
0	0	0,2	3	0
0	0	1,1	2	1
0	0	0,1	3	0
1	0	3,1	2	1
1	0	3,2	3	1
1	0	4	1	1
1	0	3,2	3	1
0	1	0,2	1	0
0	1	0,3	1	1
0	1	0,3	2	1
0	1	0,4	1	1

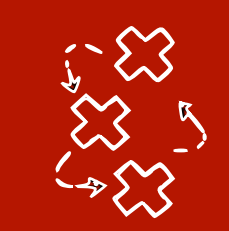


$$X_2 \not\perp C_2$$

$$X_1 \perp C_2 | C_1$$

$$X_2 \perp C_1 | X_3$$

...

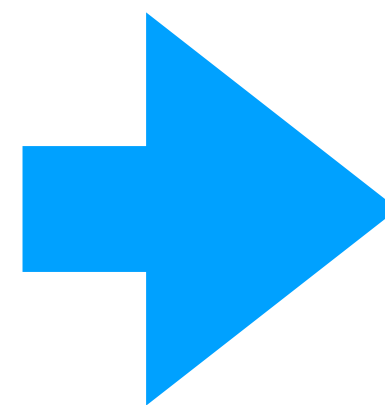


Joint Causal Inference from Multiple Contexts

Joris M. Mooij, Sara Magliacane, Tom Claassen

- We **add context variables** so we can **disentangle** changes in distribution across the datasets (and optionally background knowledge, e.g. context variables are uncaused)
- We can reuse **any standard method for observational data** that fits any chosen assumptions

C1	C2	X1	X2	X3
0	0	0,1	2	0
0	0	0,2	3	0
0	0	1,1	2	1
0	0	0,1	3	0
1	0	3,1	2	1
1	0	3,2	3	1
1	0	4	1	1
1	0	3,2	3	1
0	1	0,2	1	0
0	1	0,3	1	1
0	1	0,3	2	1
0	1	0,4	1	1



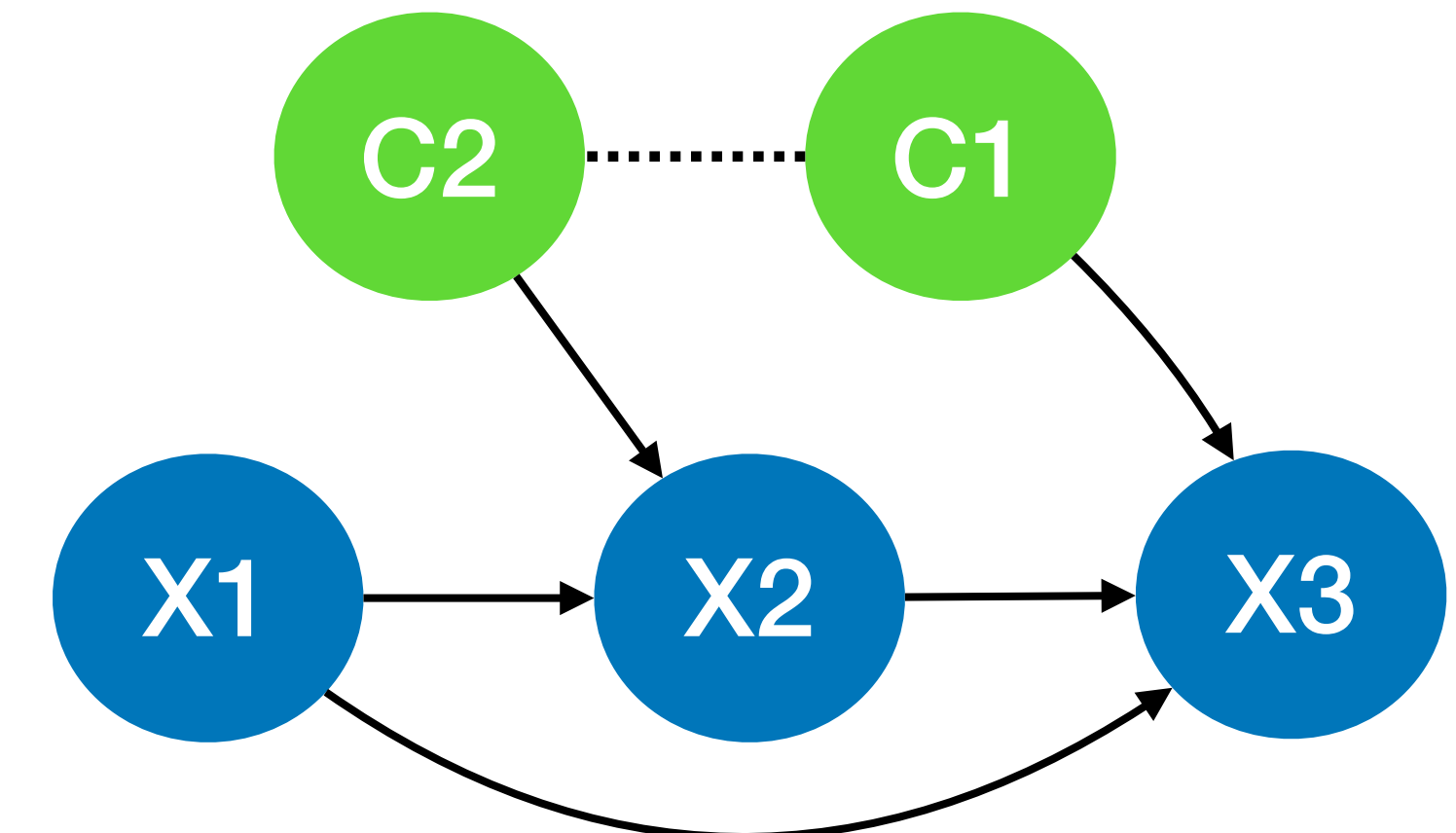
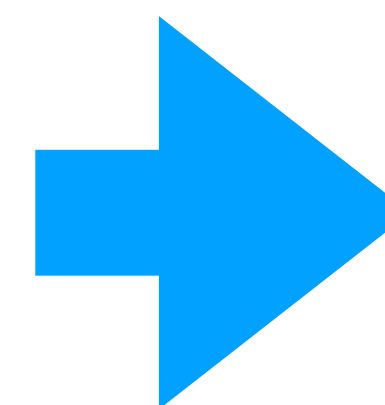
$$X_2 \not\perp C_2$$

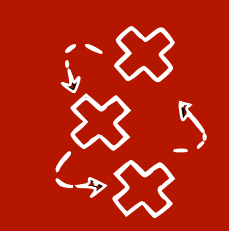
$$X_1 \perp C_2 | C_1$$

$$X_2 \perp C_1 | X_3$$

...

PC-JCI
FCI-JCI



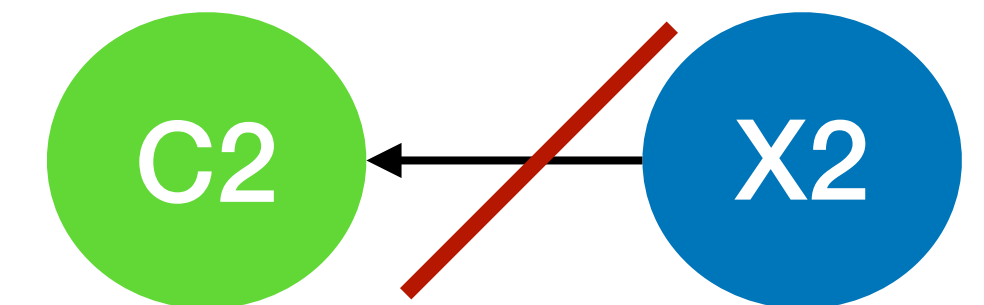


Joint Causal Inference from Multiple Contexts

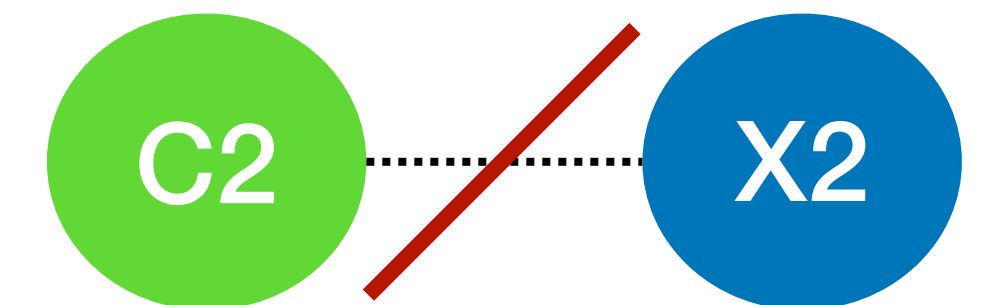
Joris M. Mooij, Sara Magliacane, Tom Claassen

- Additional optional background knowledge based on assumptions:

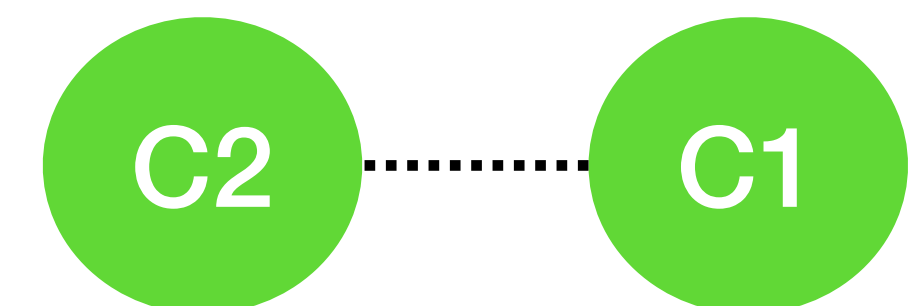
1. No system variable causes any context variable.



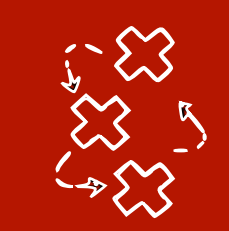
2. No context variable is confounded with a system variable.



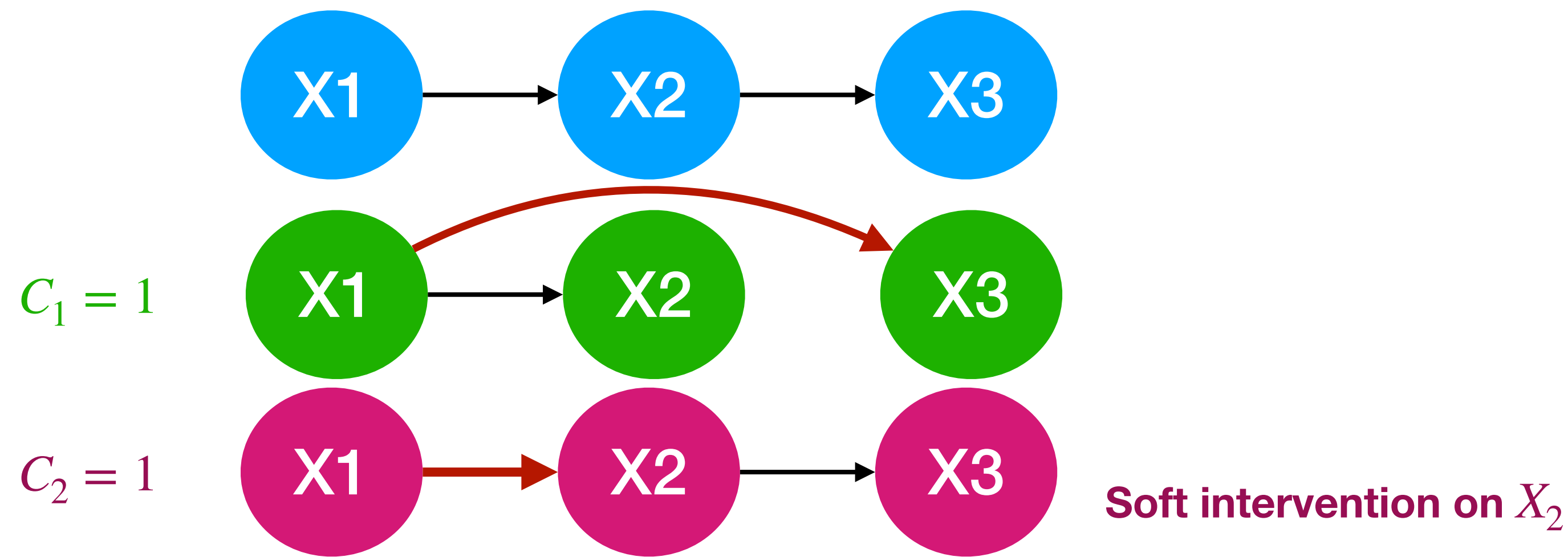
3. The context variables do not cause each other and they are assumed to be confounded.



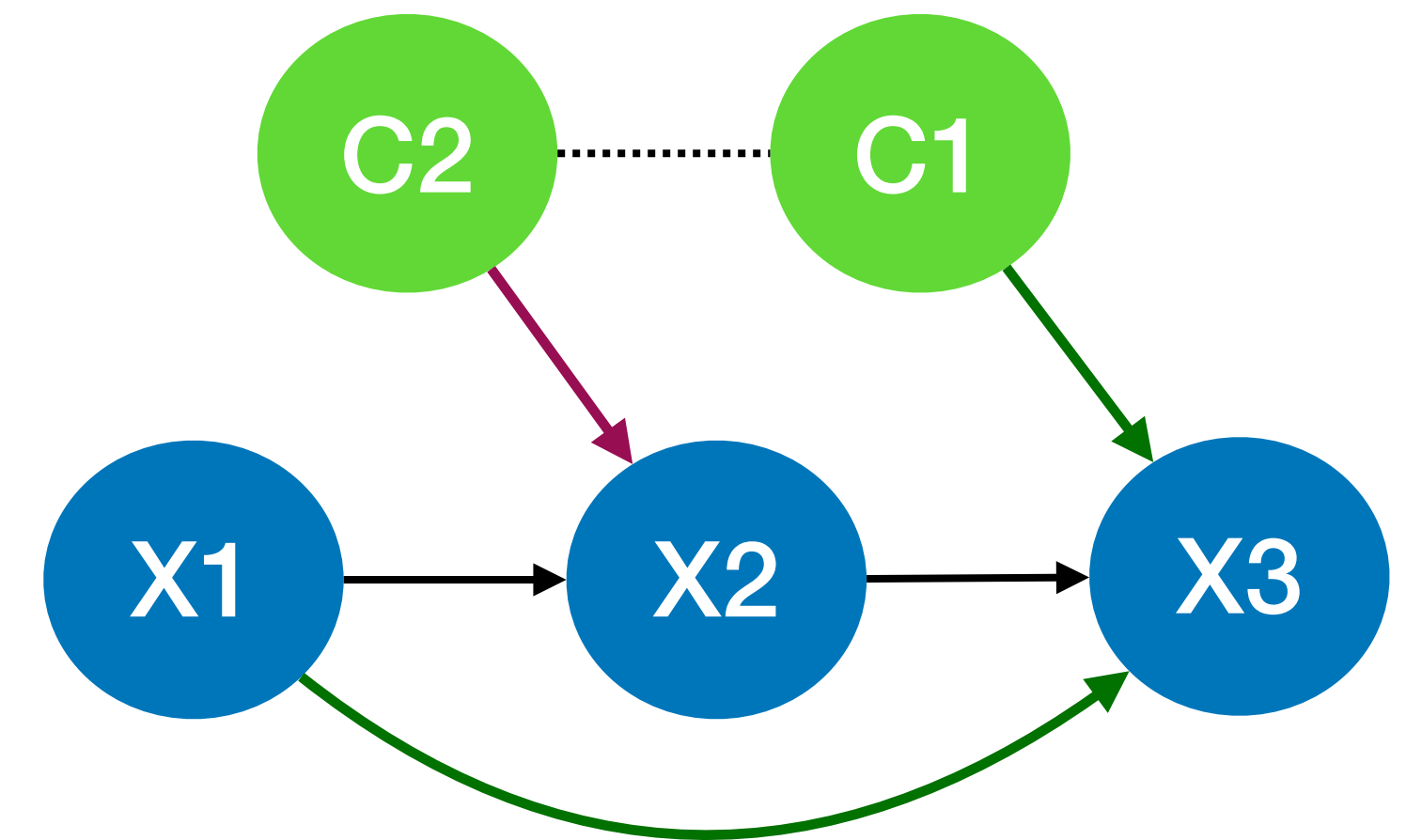
<https://arxiv.org/abs/1611.10351>



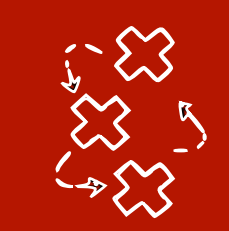
Joint Causal Inference example - setting



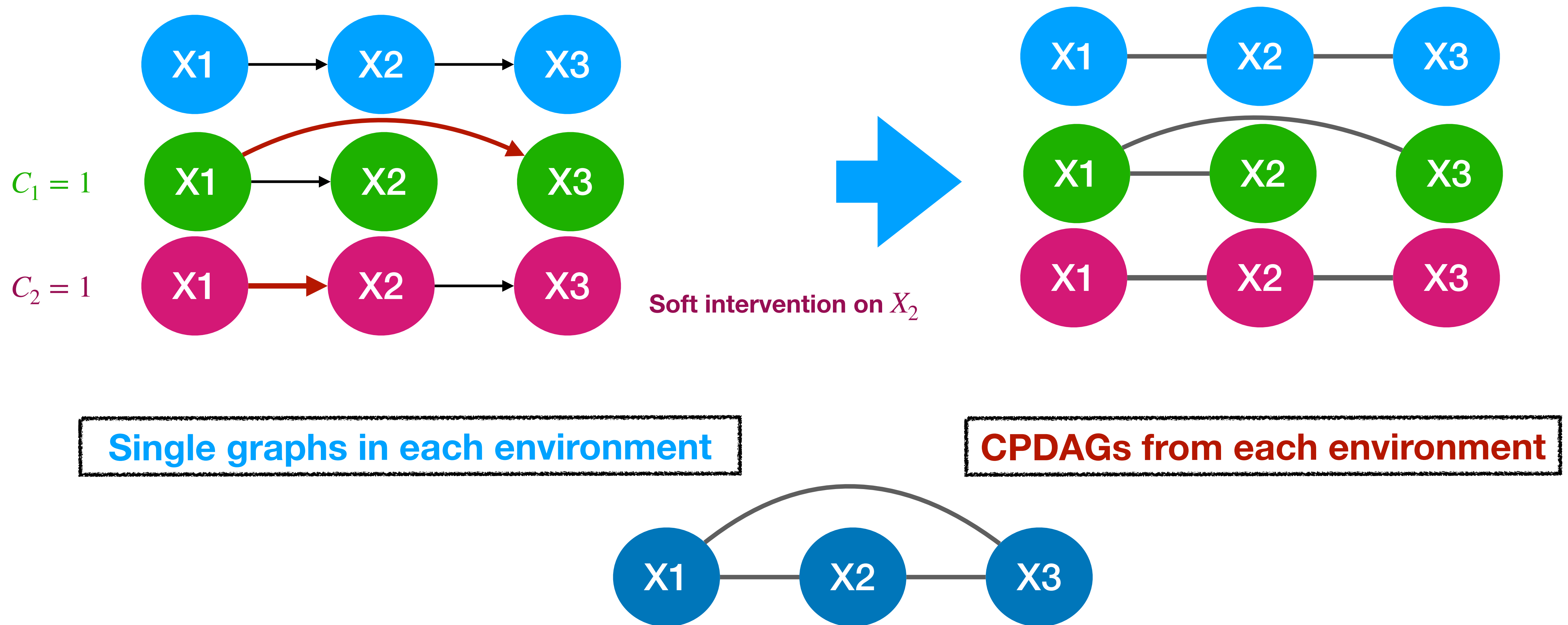
Single graphs in each environment



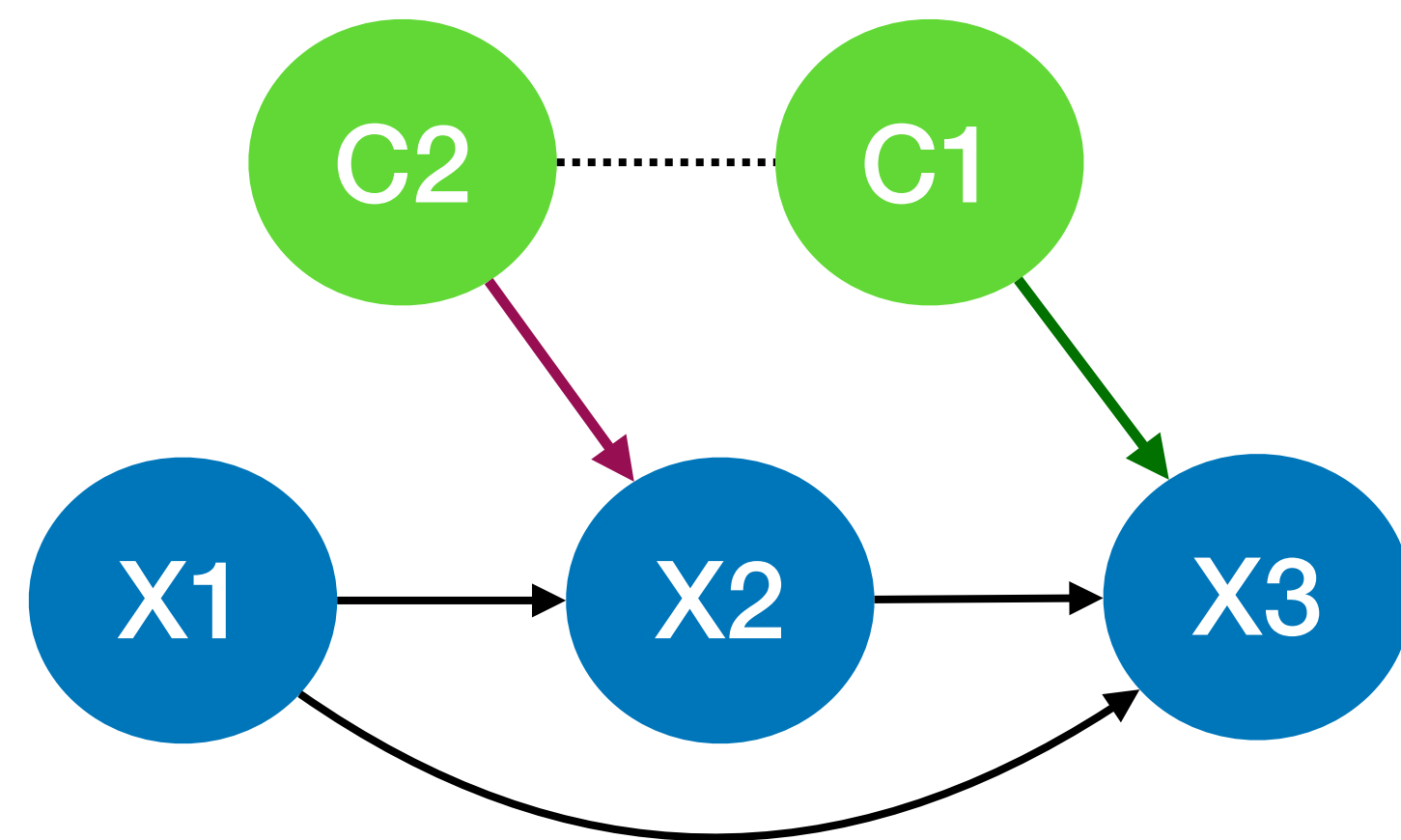
The joint graph is the union of the single graphs + edges from context variables for the causal mechanisms that change



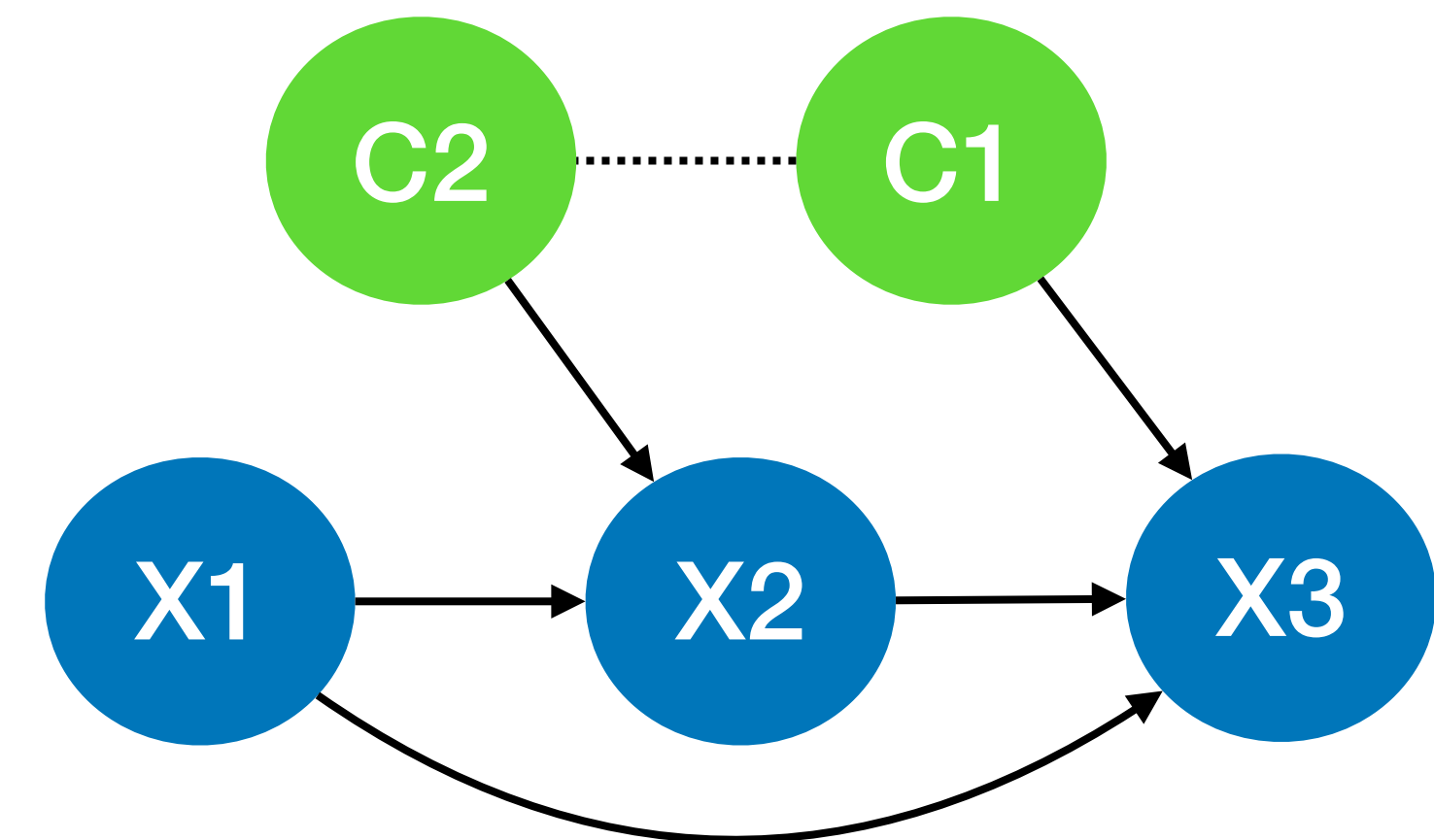
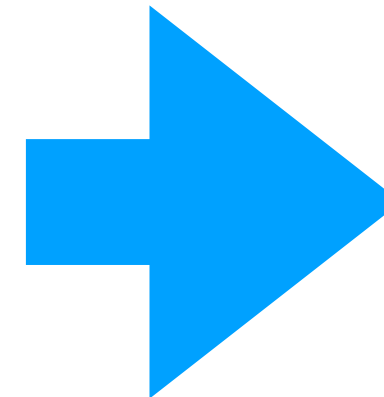
Learning graphs separately in each environment with PC



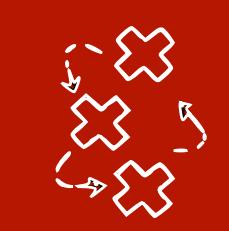
Joint causal inference + PC with multiple context variables



True underlying joint graph with multiple context variable

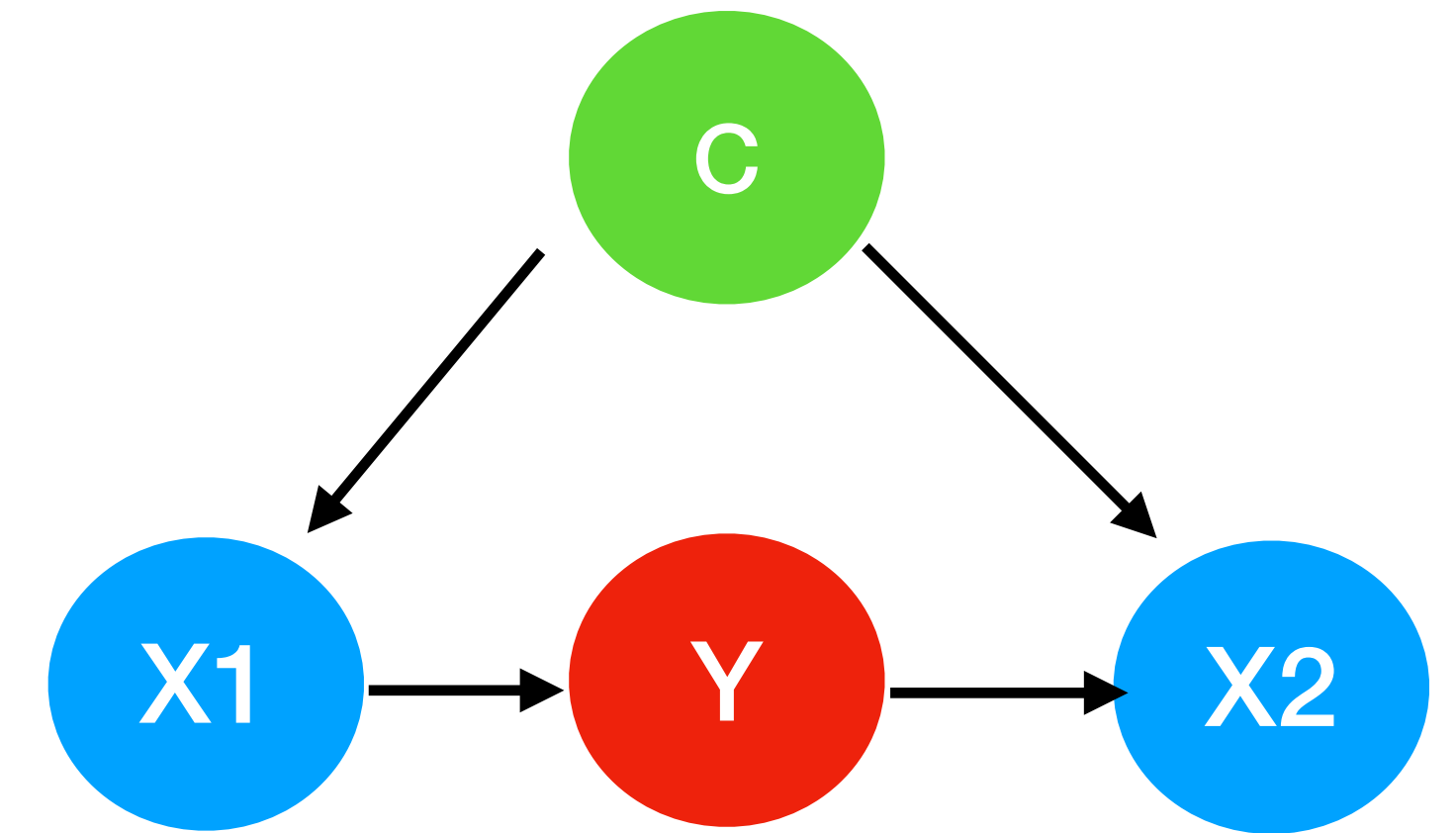


CPDAG with Joint Causal Inference and PC with multiple context variables



Application - Domain adaptation

	C	X1	X2	Y
Source domain	0	0,1	2	0
	0	0,2	3	0
	0	1,1	2	1
	0	0,1	3	0
Target domain	1	3,1	2	?
	1	3,2	3	?
	1	4	2	?
	1	3,2	3	?



We can represent $P(X1, X2, Y, C)$ with an **(unknown)** causal graph

- We can represent the different datasets jointly with Joint Causal Inference
- We can use it to reason about **features that offer a robust prediction of Y**

Causal discovery - summary

- Many more methods, also hybrids across different categories

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

- Issues for practical applications: scalability, integration of BK, reliability

Questions?

