TECHNISCHE UNIVERSITÄT DARMSTADT

AIML Lab

Winter Semester 2025/26 Lecture

Causality for AI & ML

"Dealing with Uncertainty"

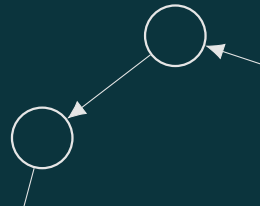
Prof. Dr. Kristian Kersting

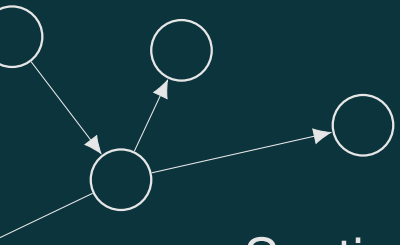
Moritz Willig — Today's speaker

Tim Woydt

Florian Busch

Matej Zečević

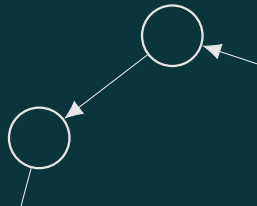




Section

0

Recap: Causal Discovery

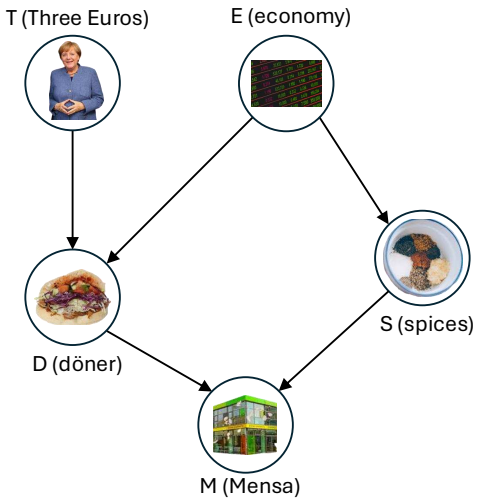


PC – Exercise

1. Independency and d-separation

Write down all pairwise and conditional independencies for the displayed Mensa graph. You can ignore all independencies that do not have a minimal conditioning set.

Minimal conditioning set: If $X \perp\!\!\!\perp Y$ but also $X \perp\!\!\!\perp Y \mid Z$, you don't have to write down $X \perp\!\!\!\perp Y \mid Z$, as the minimal conditioning set for X and Y is the empty set.



PC – Exercise

1. Independency and d-separation

Pairwise:

$T \perp E$

$T \perp S$

Conditioning on one variable:

$T \perp M|D$

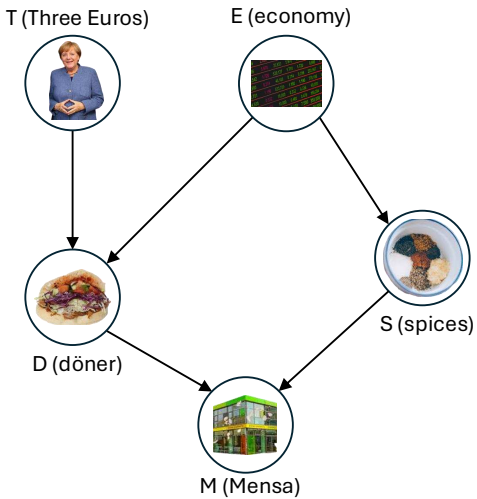
$D \perp S|E$

Conditioning on two variables:

$E \perp M|D,S$

Finished

(Hint: no conditioning will make directly connected variables dependent)

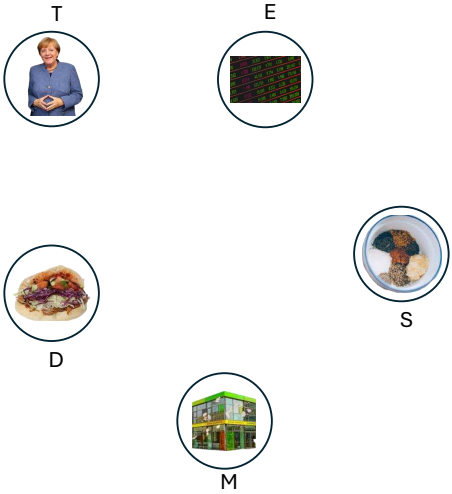


PC – Exercise

2. PC Algorithm

Imagine you want to apply the PC algorithm discovery on the Mensa problem. First, you apply pairwise and conditional independency tests on the data and record the results. Assume that this gave you exactly the independencies that you determined in the exercise from the previous slide.

Making use of these independency statements, apply the full PC algorithm. How does the final CPDAG look like?



PC – Exercise

2. PC Algorithm

Pairwise:

$T \perp E$

$T \perp S$

Conditioning on one variable:

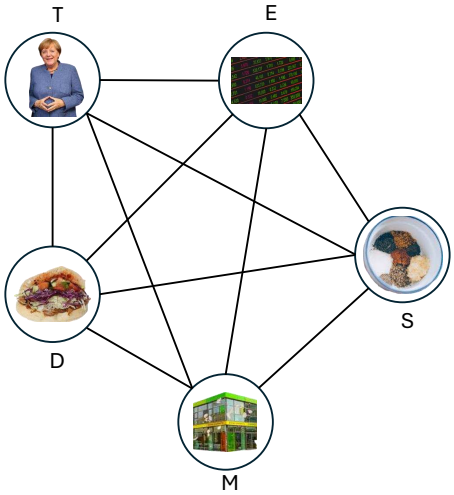
$T \perp M|D$

$D \perp S|E$

Conditioning on two variables:

$E \perp M|D,S$

Step 1: Initialization



PC – Exercise

2. PC Algorithm

Pairwise:

$T \perp E$

$T \perp S$

Conditioning on one variable:

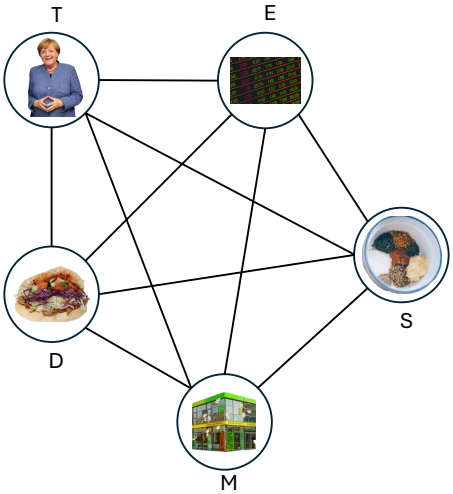
$T \perp M|D$

$D \perp S|E$

Conditioning on two variables:

$E \perp M|D,S$

Step 2: Remove edges according to independency statements



PC – Exercise

2. PC Algorithm

Pairwise:

$T \perp E$

$T \perp S$

Conditioning on one variable:

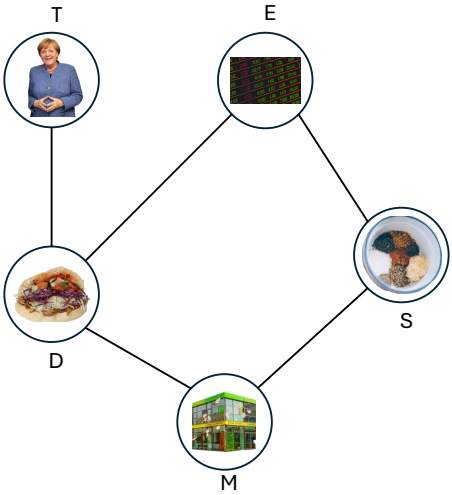
$T \perp M|D$

$D \perp S|E$

Conditioning on two variables:

$E \perp M|D,S$

Step 2: Remove edges according to independency statements



PC – Exercise

2. PC Algorithm

Pairwise:

$T \perp E$

$T \perp S$

Conditioning on one variable:

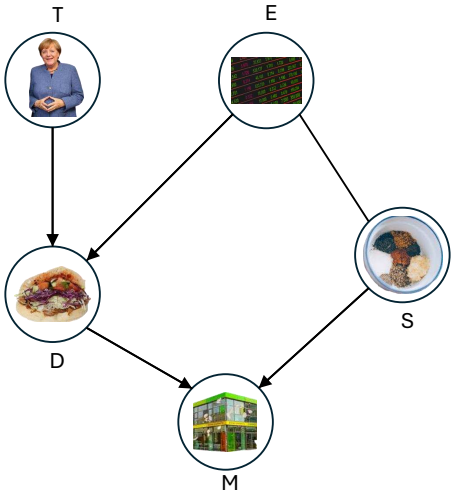
$T \perp M|D$

$D \perp S|E$

Conditioning on two variables:

$E \perp M|D,S$

Step 3: Direct v-structures



PC – Exercise

2. PC Algorithm

Pairwise:

$T \perp E$

$T \perp S$

Conditioning on one variable:

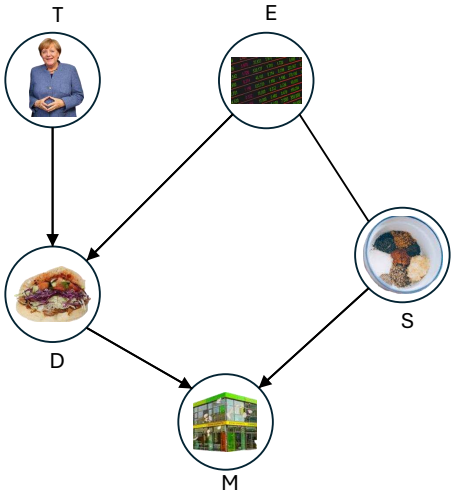
$T \perp M|D$

$D \perp S|E$

Conditioning on two variables:

$E \perp M|D,S$

Step 4: Meek rules (no changes)



Exercise

CD with known causal order

Imagine you are in a real-world setting, where you have a dataset and want to apply causal discovery. In this specific case, assume you know when each variable has been measured and that there is a clear temporal **order** according to which all variables have been recorded (e.g., A has always been recorded before B). Also assume causal sufficiency to hold.

Now, argue why (or why not) an independency based causal discovery method should always be able to return a specific DAG instead of only a Markov equivalence class here.

Exercise

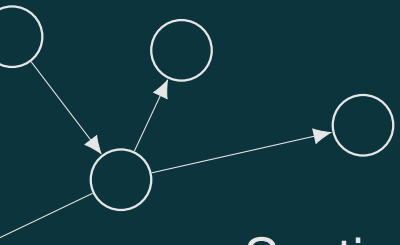
CD with known causal order

Setting

- Assumption: Causal sufficiency and causal order is given
- Also assume the standard assumptions: Markov and faithfulness
- Acyclicity follows from the provided causal order

Proof Sketch

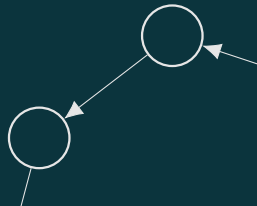
- Imagine applying PC or GES
- The true skeleton is always discovered, i.e., there is an edge in the skeleton if and only if there is an edge in the true graph (only the direction is unknown)
- Causal sufficiency implies that any edge must be directed; there can be no unobserved common causes (confounders; e.g., see $\longleftrightarrow$ in FCI)
- Direct all edges according to the causal order → fully directed DAG



Section

1

Measuring Causality



Failure Modes of CD

All previous discovery causal discovery techniques prove appealing identification guarantees under strong/optimistic assumptions.

In practice these are often violated:

- **Acyclicity:** Excludes systems with feedback loops.

Failure Modes of CD

All previous discovery causal discovery techniques prove appealing identification guarantees under strong/optimistic assumptions.

In practice these are often violated:

- **Acyclicity:** Excludes systems with feedback loops.
- **Presence of Noise:** Deterministic relations are hard to direct.

Failure Modes of CD

All previous discovery causal discovery techniques prove appealing identification guarantees under strong/optimistic assumptions.

In practice these are often violated:

- **Acyclicity:** Excludes systems with feedback loops.
- **Presence of Noise:** Deterministic relations are hard to direct.
- **Infinite Data:** Identification is often only proven in the sample limit (under access to infinite data).
  - In practice independence tests are may fail due to outliers in the data.

Failure Modes of CD

All previous discovery causal discovery techniques prove appealing identification guarantees under strong/optimistic assumptions.

In practice these are often violated:

- **Acyclicity:** Excludes systems with feedback loops.
- **Presence of Noise:** Deterministic relations are hard to direct.
- **Infinite Data:** Identification is often only proven in the sample limit (under access to infinite data).
  - In practice independence tests are may fail due to outliers in the data.
- **Causal Sufficiency:** Observation of all variables.

Failure Modes of CD

All previous discovery causal discovery techniques prove appealing identification guarantees under strong/optimistic assumptions.

In practice these are often violated:

- **Acyclicity:** Excludes systems with feedback loops.
- **Presence of Noise:** Deterministic relations are hard to direct.
- **Infinite Data:** Identification is often only proven in the sample limit (under access to infinite data).
  - In practice independence tests are may fail due to outliers in the data.
- **Causal Sufficiency:** Observation of all variables.
- ...

How can we assess the quality of our causal discovery algorithms? -> Graph Metrics

Precision, Recall and Accuracy

Most simple approach: Count the number of correctly/incorrectly predicted edges.

Accuracy: $\text{Acc} = \frac{TP + TN}{\#\text{ total possible edges}}$

Precision, Recall and Accuracy

Most simple approach: Count the number of correctly/incorrectly predicted edges.

Accuracy: Acc = $\frac{TP+TN}{\#\text{ total possible edges}}$

Take care: Many graphs are sparse! For 100 nodes there are 10,000 possible edges. But a sparse graph might only have 500 edges.

An algorithm which does not predict a single edge still has an accuracy of 95%!

Precision, Recall and Accuracy

Most simple approach: Count the number of correctly/incorrectly predicted edges.

Accuracy: Acc = $\frac{TP+TN}{\#\text{ total possible edges}}$

Take care: Many graphs are sparse! For 100 nodes there are 10,000 possible edges. But a sparse graph might only have 500 edges.

An algorithm which does not predict a single edge still has an accuracy of 95%!

Precision: "How much percent of the predicted edges are actually correct?"
Prec = $\frac{TP}{TP+FP}$

Recall: How much percent of the true edges are actually predicted?"
Rec = $\frac{TP}{TP+FN}$

Precision, Recall and Accuracy

Most simple approach: Count the number of correctly/incorrectly predicted edges.

Accuracy: Acc = $\frac{TP+TN}{\#\text{ total possible edges}}$

Take care: Many graphs are sparse! For 100 nodes there are 10,000 possible edges. But a sparse graph might only have 500 edges.

An algorithm which does not predict a single edge still has an accuracy of 95%!

Precision: "How much percent of the predicted edges are actually correct?"
Prec = $\frac{TP}{TP+FP}$

Recall: How much percent of the true edges are actually predicted?"
Rec = $\frac{TP}{TP+FN}$

Often reported as a combined F_1**-score:**
F_1-score $= 2\frac{prec \cdot recall}{precision+recall}$

Structural Hamming Distance

Context: Hamming Distances are measures that determine the number of locations where two entities differ. (Historically developed for error correction in strings).

Idea: Apply the same to graphs! Check where individual edges differ between the ground truth (GT) and prediction (pred) graph and count up all the errors:

$$SHD(\mathcal{G}_{pred}, \mathcal{G}_{GT}) := \sum_{i \in [0..N]} \sum_{j \in [0..N]} \mathbf{1}((e_{i,j} \in \mathcal{G}_{pred}) \neq (e_{i,j} \in \mathcal{G}_{GT}))$$

Acid, Silvia, and Luis M. de Campos. "Searching for Bayesian network structures in the space of restricted acyclic partially directed graphs." Journal of artificial intelligence research 18 (2003): 445-490.

Tsamardinos, Ioannis, Laura E. Brown, and Constantin F. Aliferis. "The max-min hill-climbing Bayesian network structure learning algorithm." Machine learning 65.1 (2006): 31-78.

Structural Hamming Distance

Context: Hamming Distances are measures that determine the number of locations where two entities differ. (Historically developed for error correction in strings).

Idea: Apply the same to graphs! Check where individual edges differ between the ground truth (GT) and prediction (pred) graph and count up all the errors:

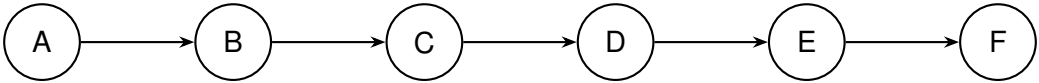
$$SHD(\mathcal{G}_{pred}, \mathcal{G}_{GT}) := \sum_{i \in [0..N]} \sum_{j \in [0..N]} \mathbf{1}((e_{i,j} \in \mathcal{G}_{pred}) \neq (e_{i,j} \in \mathcal{G}_{GT}))$$

Counting Direction Errors Twice: One could consider it a worse mistake to predict an edge in the wrong direction than not to predict it (or leaving it undirected). The above formula counts wrongly directed edges with double the error - once for $e_{i,j}$ and again for $e_{j,i}$.

Acid, Silvia, and Luis M. de Campos. "Searching for Bayesian network structures in the space of restricted acyclic partially directed graphs." Journal of artificial intelligence research 18 (2003): 445-490.

Tsamardinos, Ioannis, Laura E. Brown, and Constantin F. Aliferis. "The max-min hill-climbing Bayesian network structure learning algorithm." Machine learning 65.1 (2006): 31-78.

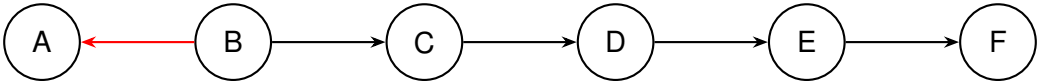
“Not all edges are the same”



Wrongly directing some edges in the above graph might be worse for some than for others.

Wahl, Jonas, and Jakob Runge. "Separation-based distance measures for causal graphs." arXiv preprint arXiv:2402.04952 (2024).

“Not all edges are the same”



Wrongly directing some edges in the above graph might be worse for some than for others.

Flipping A→B: Conditional (in)dependencies between A and all other variables (except B) are affected. Independencies between B,…,F, however, stay intact.

Wahl, Jonas, and Jakob Runge. "Separation-based distance measures for causal graphs." arXiv preprint arXiv:2402.04952 (2024).

“Not all edges are the same”

Wrongly directing some edges in the above graph might be worse for some than for others.

Flipping A→B: Conditional (in)dependencies between A and all other variables (except B) are affected. Independencies between B,…,F, however, stay intact.

Flipping C→D: Conditional (in)dependency statements between variables on both sides of the flipped edge are broken. Predicting this edge wrong has a much stronger impact on the correct prediction of many causal queries.

Wahl, Jonas, and Jakob Runge. "Separation-based distance measures for causal graphs." arXiv preprint arXiv:2402.04952 (2024).

Structural Intervention Distance (SID)

Unlike the SHD, which counts edge errors, the SID quantifies how ‘useful’ a graph is for causal inference.

The **Structural Intervention Distance** counts the number of pairs (X, Y) where the predicted graph would lead to an incorrect prediction for the query $P(Y|do(X))$ using parent adjustment.

- A single edge flip in the wrong position can drastically increase SID.
- Some modifications (e.g., edge additions) to the graph might not have an impact on the SID.

Structural Intervention Distance (SID)

Unlike the SHD, which counts edge errors, the SID quantifies how ‘useful’ a graph is for causal inference.

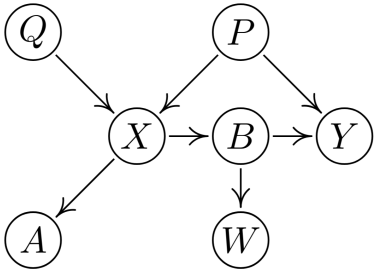
The **Structural Intervention Distance** counts the number of pairs (X, Y) where the predicted graph would lead to an incorrect prediction for the query $P(Y|do(X))$ using parent adjustment.

- A single edge flip in the wrong position can drastically increase SID.
- Some modifications (e.g., edge additions) to the graph might not have an impact on the SID.

Complexity: $O(p \cdot log_2(p) \cdot f(p))$ where p is the number of nodes. $f(n)$ is the complexity of squaring a matrix. Naive implementations take $O(p^3)$, faster ones $O(p^{2.37\ldots})$. The complexity of is $O(p^4 \cdot log_2(p))$ for a naive implementations.

Peters and Bühlmann. “Structural Intervention Distance for Evaluating Causal Graphs”. In: Neural Computation 27.3 2015

Reminder: Adjustment Sets

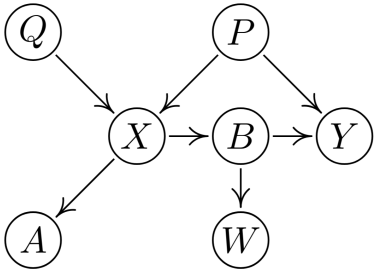


Adjustment Set for $Y|do(X)$: No $Z \in \mathbf{Z}$ is a descendant of any W which lies on a directed path from X to Y and $\mathbf{Z}$ blocks all non-directed paths from X to Y.

"Sets $\mathbf{Z} = \{P, Q\}$ and $\mathbf{Z} = \{P, A\}$ are valid adjustment sets for $Y|do(X)$; $\mathbf{Z} = \{P\}$ is the smallest adjustment set. Any set containing W cannot be a valid adjustment set."

Peters and Bühlmann. "Structural Intervention Distance for Evaluating Causal Graphs". In: Neural Computation 27.3 2015

Reminder: Adjustment Sets



Adjustment Set for $Y|do(X)$: No $Z \in \mathbf{Z}$ is a descendant of any W which lies on a directed path from X to Y and $\mathbf{Z}$ blocks all non-directed paths from X to Y.

Parent Adjustment: Specifically $\mathbf{Z} = pa(X)$ is always a valid adjustment set for $Y|do(X)$!

Peters and Bühlmann. "Structural Intervention Distance for Evaluating Causal Graphs". In: Neural Computation 27.3 2015

Structural Intervention Distance (SID)

The SID counts the number of incorrectly predicted adjustment sets:

$$\text{SID}(\mathcal{G}, \mathcal{H}) = \# \left\{ (i,j), i \neq j \mid \begin{cases} j \in \mathbf{DE}_i^{\mathcal{G}} & \text{if } j \in \mathbf{PA}_i^{\mathcal{H}} \\ \mathbf{PA}_i^{\mathcal{H}} \text{ is not a valid adj. set for } (\mathcal{G}, i, j) & \text{if } j \notin \mathbf{PA}_i^{\mathcal{H}} \end{cases} \right\}$$

Peters and Bühlmann. “Structural Intervention Distance for Evaluating Causal Graphs”. In: Neural Computation 27.3 2015

Structural Intervention Distance (SID)

The SID counts the number of incorrectly predicted adjustment sets:

$$\text{SID}(\mathcal{G}, \mathcal{H}) = \# \left\{ (i,j), i \neq j \mid \begin{cases} j \in \mathbf{DE}_i^{\mathcal{G}} & \text{if } j \in \mathbf{PA}_i^{\mathcal{H}} \\ \mathbf{PA}_i^{\mathcal{H}} \text{ is not a valid adj. set for } (\mathcal{G}, i, j) & \text{if } j \notin \mathbf{PA}_i^{\mathcal{H}} \end{cases} \right\}$$

Zero Distance: If two graphs $\mathcal{G}, \mathcal{H}$ are equal, the SID is zero:
$\mathcal{G} = \mathcal{H} \Rightarrow \text{SID}(\mathcal{G}, \mathcal{H}) = 0$

The SID is ***not*** **symmetric**: $\text{SID}(\mathcal{G}, \mathcal{H}) = A \not\Rightarrow \text{SID}(\mathcal{G}, \mathcal{H}) = A$

Bounds: $\text{SHD}(\mathcal{G}, \mathcal{H}) = 1 \Rightarrow \text{SID}(\mathcal{G}, \mathcal{H}) \leq 2 \cdot (p - 1)$

Subgraph Distance: $\mathcal{G} \leq \mathcal{H} \Leftrightarrow \text{SID}(\mathcal{G}, \mathcal{H}) = 0$

Peters and Bühlmann. "Structural Intervention Distance for Evaluating Causal Graphs". In: Neural Computation 27.3 2015

Structural Intervention Distance (SID)

The SID counts the number of incorrectly predicted adjustment sets:

$$\text{SID}(\mathcal{G}, \mathcal{H}) = \#\left\{(i,j), i \neq j \mid \begin{cases} j \in \mathbf{DE}_i^{\mathcal{G}} & \text{if } j \in \mathbf{PA}_i^{\mathcal{H}} \\ \mathbf{PA}_i^{\mathcal{H}} \text{ is not a valid adj. set for } (\mathcal{G}, i, j) & \text{if } j \notin \mathbf{PA}_i^{\mathcal{H}} \end{cases}\right\}$$

Zero Distance: If two graphs $\mathcal{G}, \mathcal{H}$ are equal, the SID is zero:
$\mathcal{G} = \mathcal{H} \Rightarrow \text{SID}(\mathcal{G}, \mathcal{H}) = 0$

The SID is ***not*** **symmetric**: $\text{SID}(\mathcal{G}, \mathcal{H}) = A \not\Rightarrow \text{SID}(\mathcal{G}, \mathcal{H}) = A$

Bounds: $\text{SHD}(\mathcal{G}, \mathcal{H}) = 1 \Rightarrow \text{SID}(\mathcal{G}, \mathcal{H}) \leq 2 \cdot (p - 1)$

Subgraph Distance: $\mathcal{G} \leq \mathcal{H} \Leftrightarrow \text{SID}(\mathcal{G}, \mathcal{H}) = 0$

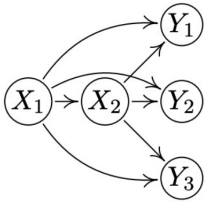
CPDAGs: The SID is extended to CPDAGs by iterating over the DAGs of the CPDAG.

Peters and Bühlmann. “Structural Intervention Distance for Evaluating Causal Graphs”. In: Neural Computation 27.3 2015

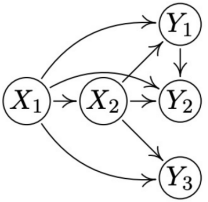
SID Example

$\mathcal{H}_1$ **adds** an edge over $\mathcal{G}$.
$\mathcal{H}_2$ **reverses** an edge over $\mathcal{G}$.

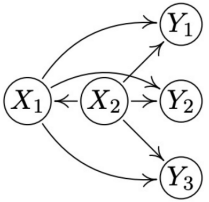
$\text{SHD}(\mathcal{G}, \mathcal{H}_1) = 1$
$\text{SHD}(\mathcal{G}, \mathcal{H}_2) = 2$
or $\text{SHD}(\mathcal{H}_2) = 1$ depending on whether the reversal error is counted twice or not.



true graph $\mathcal{G}$



graph $\mathcal{H}_1$



graph $\mathcal{H}_2$

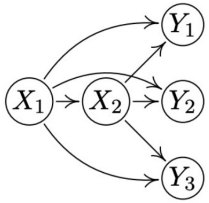
Peters and Bühlmann. "Structural Intervention Distance for Evaluating Causal Graphs". In: Neural Computation 27.3 2015

SID Example

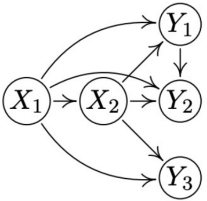
$\mathcal{H}_1$ **adds** an edge over $\mathcal{G}$.
$\mathcal{H}_2$ **reverses** an edge over $\mathcal{G}$.

$\text{SHD}(\mathcal{G}, \mathcal{H}_1) = 1$
$\text{SHD}(\mathcal{G}, \mathcal{H}_2) = 2$
or $\text{SHD}(\mathcal{H}_2) = 1$ depending on whether the reversal error is counted twice or not.

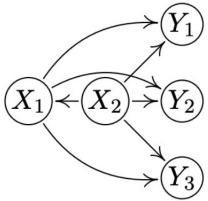
$\text{SID}(\mathcal{G}, \mathcal{H}_1) = 0$ ($\mathcal{G}$ is a subgraph of $\mathcal{H}_1$)
$\text{SID}(\mathcal{G}, \mathcal{H}_2) = 8$



true graph $\mathcal{G}$



graph $\mathcal{H}_1$



graph $\mathcal{H}_2$

Peters and Bühlmann. "Structural Intervention Distance for Evaluating Causal Graphs". In: Neural Computation 27.3 2015

‘Gaming Benchmarks?’ - Var-Sortability [1]

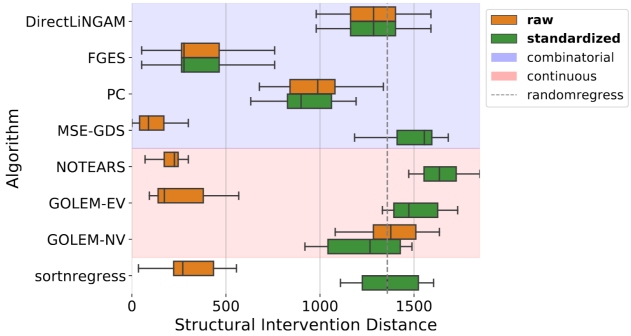
Var-Sortability: Some models, e.g. ANMs, feature increasing variance with the depth of a variable in the graph. Simply directing variables by $var_{min} \rightarrow var_{max}$ achieves a good performance.

[1] Reisach, A; Seiler, C and Weichwald, S. "Beware of the simulated dag! Causal discovery benchmarks may be easy to game." Advances in Neural Information Processing Systems 34 (2021): 27772-27784.

‘Gaming Benchmarks?’ - Var-Sortability [1]

Var-Sortability: Some models, e.g. ANMs, feature increasing variance with the depth of a variable in the graph. Simply directing variables by $var_{min} \rightarrow var_{max}$ achieves a good performance.

Renormalizing variables prevents this ‘trick’. Some algorithms seem to rely on this property!



Hint: If a system is known to feature var-sortable data, there is nothing wrong with leveraging this information! Just don’t assume this property by default!

[1] Reisach, A; Seiler, C and Weichwald, S. "Beware of the simulated dag! Causal discovery benchmarks may be easy to game." Advances in Neural Information Processing Systems 34 (2021): 27772-27784.

‘Gaming Benchmarks?’ - R^2-Sortability [2]

A solution to avoid Var-Sortability in benchmarks is to scale all variables back to unit variance. Variable order can still be predicted by considering the *coefficient of determination* R^2 of a variable from its parents.

R^2 can’t directly be measured (because the parents are a-priori unknown). [2] proposes to determine R^2 by simply considering all correlated variables as a proxy:

$$R^2 = 1 - \frac{\mathrm{Var}\left(X_t - \mathbb{E}[X_t \mid X_{\{1,\dots,d\}\setminus\{t\}}]\right)}{\mathrm{Var}\left(X_t\right)}$$

[2] Reisach, Alexander, et al. "A scale-invariant sorting criterion to find a causal order in additive noise models." Advances in Neural Information Processing Systems 36 (2023): 785-807.

‘Gaming Benchmarks?’ - R^2-Sortability [2]

A solution to avoid Var-Sortability in benchmarks is to scale all variables back to unit variance. Variable order can still be predicted by considering the *coefficient of determination* R^2 of a variable from its parents.

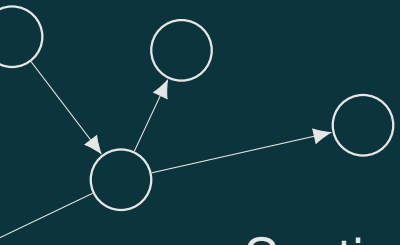
R^2 can’t directly be measured (because the parents are a-priori unknown). [2] proposes to determine R^2 by simply considering all correlated variables as a proxy:

$$R^2 = 1 - \frac{\mathrm{Var}\left(X_t - \mathbb{E}[X_t \mid X_{\{1,\dots,d\}\setminus\{t\}}]\right)}{\mathrm{Var}\left(X_t\right)}$$

The approach achieves competitive results on some data.

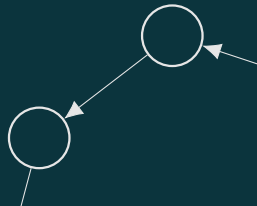
Take away: Be aware of the implicit biases in your data. For developing robust algorithms leverage known properties, but try not to game your benchmarks.

[2] Reisach, Alexander, et al. "A scale-invariant sorting criterion to find a causal order in additive noise models." Advances in Neural Information Processing Systems 36 (2023): 785-807.



Section 2

Graph Uncertainty



Iterating MECs

Causal discovery algorithms can often only identify causal graphs up to their Markov equivalence class (MEC). Given a CPDAG representing a MEC, we still want to analyze some causal properties of the MEC.

For computing causal effect bounds from a MEC we might want to enumerate all DAGs in a MEC an take their minimum/maximum.

Iterating MECs

Causal discovery algorithms can often only identify causal graphs up to their Markov equivalence class (MEC). Given a CPDAG representing a MEC, we still want to analyze some causal properties of the MEC.

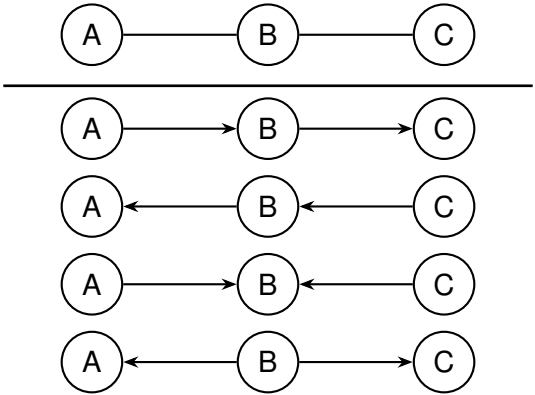
For computing causal effect bounds from a MEC we might want to enumerate all DAGs in a MEC an take their minimum/maximum.

Naïve Approach: Iterate all over all $2^{|u|}$ possible edge directions of all undirected edges u.

Some of those graphs, however, need to be rejected! E.g. ones that would induce cycles.

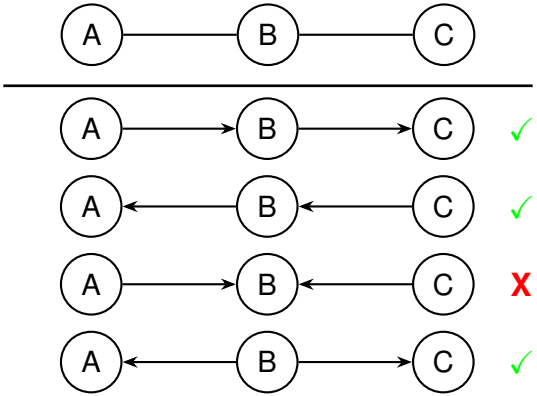
Iterating MECs

Given two undirected neighboring edges in a graph:



Iterating MECs

Given two undirected neighboring edges in a graph:



The third case would have been directed during discovery. When iterating CPDAGs, we need to reject combinations that create v-structures.

Iterating MECs

Naïve Approach Complexity: $\mathcal{O}(2^n)$.

Causal Order Iteration Complexity [1]: $\mathcal{O}(n!)$.

Recent Advances (Thm. 3 in [2]):
The Markov equivalence class [G] of a CPDAG $\mathcal{G}$ can be enumerated with worst-case delay $\mathcal{O}(n + m)$.

(n=number of nodes, m=number of edges)

[1] Christopher Meek. "Causal inference and causal explanation with background knowledge". In Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence, UAI'95, pages 403–410, 1995.
[2] Wienöbst, Marcel, et al. "Efficient enumeration of markov equivalent dags." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 37. No. 10. 2023.

Bounded Effects

Iterate all possible DAGs of a MEC/CPDAG and compute causal effects, $\mathbf{C} = \{c_1, \ldots, c_m\}$.

Estimate bound by taking minimum and maximum of all estimates: causal effect bound = $[\min(\mathbf{C}), \max(\mathbf{C})]$.

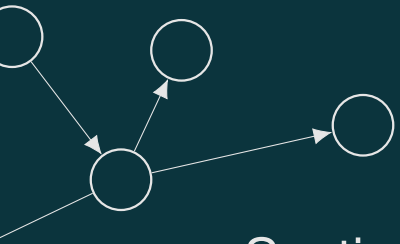
Bounded Effects

Iterate all possible DAGs of a MEC/CPDAG and compute causal effects,
$\mathbf{C} = \{c_1, \ldots, c_m\}$.

Estimate bound by taking minimum and maximum of all estimates:
causal effect bound = $[\min(\mathbf{C}), \max(\mathbf{C})]$.

Deciding based on bounds:

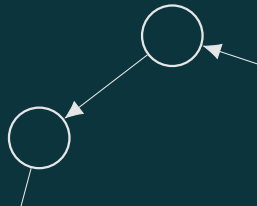
1. Bounds are **strictly positive [negative]** (e.g, $[0.5, 2.3]$)
   $\Rightarrow$ Positive [Negative] treatment effect "The medication always helps [harms]".
2. Bounds are **positive [negative]** but include zero (e.g, $[0.0, 2.3]$)
   $\Rightarrow$ "The medication might help [harm] or have no effect".
3. Bounds span into the **positive and negative domain** (e.g, $[-10.1, 2.1]$)
   $\Rightarrow$ No qualitative decision possible. "Medication might help or harm…".



Section

3

Mechanism Uncertainty

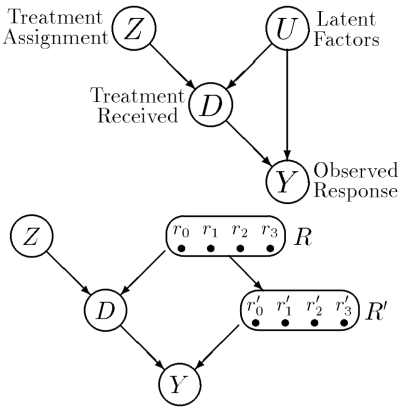


Partial Compliance

Scenario: Data (Z, D, Y) from a medical study.

The latent factors U are assumed to split patient behavior into 4 discrete compliance groups:

1. **Compliers**: Take treatment as assigned ($Z = 1 \to D = 1, Z = 0 \to D = 0$).
2. **Always-takers**: Take treatment regardless of assignment ($D = 1$ always).
3. **Never-takers**: Reject treatment regardless of assignment ($D = 0$ always).
4. **Defiers**: Do the opposite of assignment ($Z = 1 \to D = 0, Z = 0 \to D = 1$).



Balke, Alexander and Pearl, "Judea. Nonparametric bounds on causal effects from partial compliance data". Technical Report R-199, Cognitive Systems Laboratory, UCLA, 1993

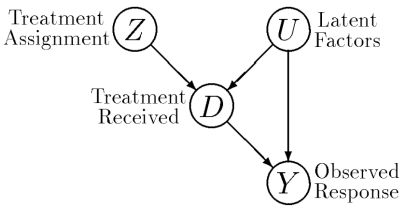
Partial Compliance - cont. I

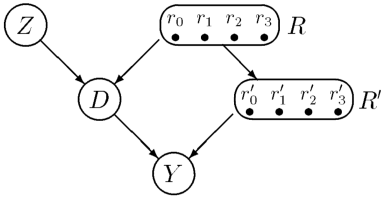
Similarly, the latent factors U affect the outcome:

1. **Helped**: Recover only if treated ($D = 1 \rightarrow Y = 1, D = 0 \rightarrow Y = 0$).
2. **Hurt**: Recover only if not treated ($D = 1 \rightarrow Y = 0$, $D = 0 \rightarrow Y = 1$).
3. **Always-recover**: Recover regardless of treatment ($Y = 1$ always).
4. **Never-recover**: Fail to recover regardless of treatment ($Y = 0$ always).

Objective: Estimate the average causal effect of assigning a treatment:

$$ACE = P(Y{=}1|do(D{=}1)) - P(Y{=}1|do(D{=}0))$$





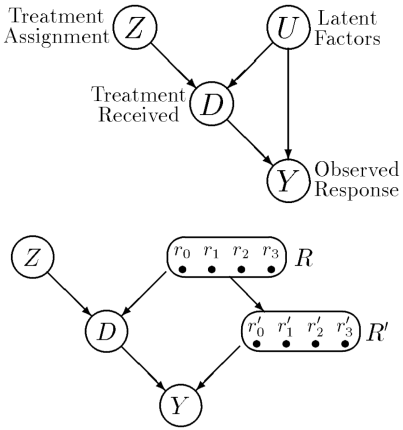
Balke, Alexander and Pearl, "Judea. Nonparametric bounds on causal effects from partial compliance data". Technical Report R-199, Cognitive Systems Laboratory, UCLA, 1993

Partial Compliance - cont. II

Crossing all 4×4 different behaviors for treatment and response obtains 16 different latent groups. We write their probabilities as q_{ij}.

Example: Consider the observed sample:
$Z = 1, D = 1, Y = 1$.
We can not distinguish between a 'complier' and an 'always-taker'. Similarly, we can not distinguish between the 'helped' and 'always-recovers' case.

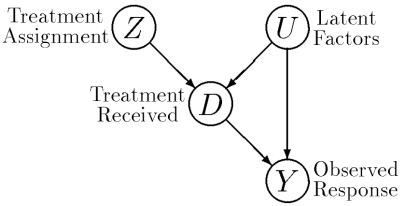
$$P(d, y|z) = p_{dy.z} = \sum_{q_{ij} \text{ consistent with } d,y,z} q_{ij}$$

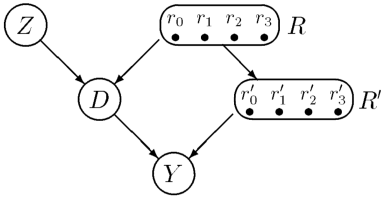


Balke, Alexander and Pearl, "Judea. Nonparametric bounds on causal effects from partial compliance data". Technical Report R-199, Cognitive Systems Laboratory, UCLA, 1993

Partial Compliance - cont. IV

Exact probabilities of ‘compliers’ and ‘always-takers’ is unknown, but we can find upper and lower bounds using **linear programming**.





Balke, Alexander and Pearl, "Judea. Nonparametric bounds on causal effects from partial compliance data". Technical Report R-199, Cognitive Systems Laboratory, UCLA, 1993

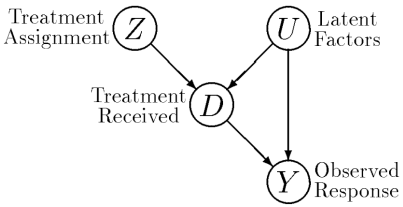
Partial Compliance - cont. IV

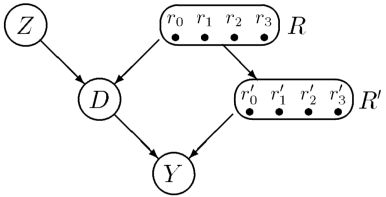
Exact probabilities of 'compliers' and 'always-takers' is unknown, but we can find upper and lower bounds using **linear programming**.

Constraints:

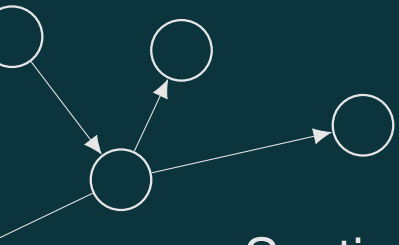
1. Probs. match the data: $p_{dy.z} = P(d, y|z)$.
2. q_{ij}'s must sum to one and be non-negative: $\sum_{i,j\in[0..3]} q_{ij} = 1.0$ and $\forall i, j \in [0..3].q_{ij} \geq 0$
3. Observed $p_{dy.z}$ and their consistent q_{ij} must match:

$$p_{00.0} = q_{00} + q_{01} + q_{10} + q_{11}$$
$$p_{01.0} = q_{20} + q_{22} + q_{30} + q_{32}$$
. . . (full list in the paper)





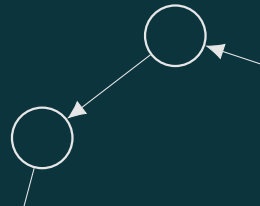
Balke, Alexander and Pearl, "Judea. Nonparametric bounds on causal effects from partial compliance data". Technical Report R-199, Cognitive Systems Laboratory, UCLA, 1993



Section

4

Continuous DAG Constraints



Continuous DAGness

Last lecture we saw the NOTEARS method.

$$\begin{aligned} \min_{W \in \mathbb{R}^{d \times d}} \quad & score(W) \\ \text{s.t.} \quad & h(W) = 0 \end{aligned}$$

- Fits the predicted graph to the data according to some score.
- Minize the 'DAGness' of a graph via some function $h(W)$.
- To allow for standard numerical optimization, $h(W)$ should be a smooth, differentiable function.

Zheng, X., Aragam, B., Ravikumar, P.K. and Xing, E.P., 2018. Dags with no tears: Continuous optimization for structure learning. Advances in neural information processing systems, 31.

Matrix Powers and DAGness

For a weighted adjacency matrix W, an entry $(W^k)_{ij}$ of the k-power of the matrix

$$W^k = \underbrace{W \cdot W \cdot \ldots \cdot W}_{k-times}$$

is the sum of weights along all paths of length k from node i to node j.

Matrix Powers and DAGness

For a weighted adjacency matrix W, an entry $(W^k)_{ij}$ of the k-power of the matrix

$$W^k = \underbrace{W \cdot W \cdot \ldots \cdot W}_{k-times}$$

is the sum of weights along all paths of length k from node i to node j.

The diagonal elements $(W^k)_{ii}$ therefore represent the presence of **self-cycles** $X_i \to \cdots \to X_i$ of length k.

If a graph is a DAG there should be no self-cycles!

graph is DAG $\Leftrightarrow \forall k \in \mathbb{N}.tr(W^k) = 0$

Matrix Exponentials

It is infeasible/expensive to check for every $W^1, W^2, \ldots, W^\infty$.

(although the longest path is bounded by the number of nodes).

Step 1. Recall the Taylor series: $e^x = 1 + x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \ldots$

and apply it to W: $e^W = I + W + \frac{1}{2!}W^2 + \frac{1}{3!}W^3 + \ldots$

Insight: The matrix exponential sums up all the matrix powers!

Matrix Exponentials

It is infeasible/expensive to check for every $W^1, W^2, \ldots, W^\infty$.
(although the longest path is bounded by the number of nodes).

Step 1. Recall the Taylor series: $e^x = 1 + x + \frac{1}{2!}x^2 + \frac{1}{3!}x^3 + \ldots$
and apply it to W: $e^W = I + W + \frac{1}{2!}W^2 + \frac{1}{3!}W^3 + \ldots$

Insight: The matrix exponential sums up all the matrix powers!

Step 2. Take the trace on both sides. The trace is linear $tr(A + B) = tr(A) + tr(B)$:
$tr(e^W) = tr(I) + tr(W) + \frac{1}{2!}tr(W^2) + \frac{1}{3!}tr(W^3) + \ldots$

Traces of all matrix powers should be zero ($tr(W^k) = 0$).
The trace of the identity matrix ($tr(I)$) sums up the ones along the diagonal:
$tr(I) = d$ (d = number of nodes).

Therefore, $tr(e^W) = d$, or equivalently: $tr(e^W) - d = 0$.

DAGness Score

Step 3. Again, recall

$$e^W = I + W + \frac{1}{2!}W^2 + \frac{1}{3!}W^3 + \dots$$

To avoid cancellation effects between individual negative and positive terms, entries of W are point-wise squared ($W \circ W$) to make them positive.

DAGness Score

Step 3. Again, recall

$$e^W = I + W + \frac{1}{2!}W^2 + \frac{1}{3!}W^3 + \dots$$

To avoid cancellation effects between individual negative and positive terms, entries of W are point-wise squared ($W \circ W$) to make them positive.

Step 4. The DAGness score is finally defined as:

$$h(W) = tr(e^{W \circ W}) - d$$

DAGness Score

Step 3. Again, recall

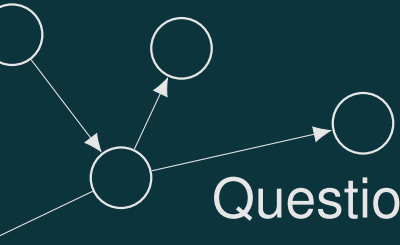
$$e^W = I + W + \frac{1}{2!}W^2 + \frac{1}{3!}W^3 + \dots$$

To avoid cancellation effects between individual negative and positive terms, entries of W are point-wise squared ($W \circ W$) to make them positive.

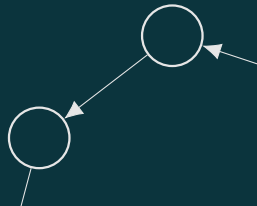
Step 4. The DAGness score is finally defined as:

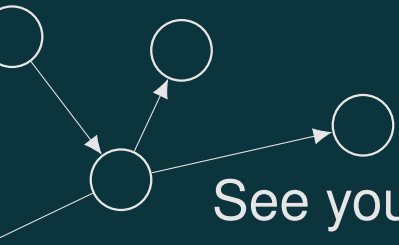
$$h(W) = tr(e^{W \circ W}) - d$$

The matrix exponential (e.g., in `scipy.linalg.expm`) is computed via a 'Scaling and Squaring' approach combined with Padé approximations. Complexity of $\mathcal{O}(d^3)$.



Questions?





See you next week!