Causal Discovery

Constraint-Based Methods

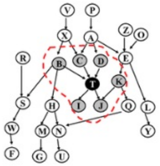
Daryna Nedilko, University of Copenhagen, 2025

UNIVERSITY OF COPENHAGEN

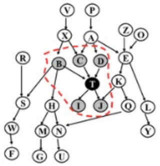
Constraint-Based Learning

- Input: Data
- Output: Learned data relationships in the form of **graph**
- Main idea: Use **Conditional Independence (CI)** tests to learn the structure.
    - Test if $X \perp\!\!\!\perp Y \mid S$, where S is a set of variables (can be empty).

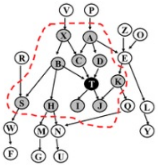
Discovery: Local vs. Global vs. Intermediate



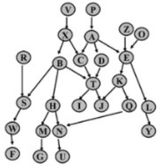
Problem #1: Consider a target variable T and discover Markov Blanket of T.



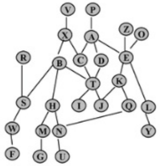
Problem #2: Consider a target variable T and discover Parents and Children of T.



Problem #3: Consider a target variable T and discover regions (e.g., of depth 2 edges) around T.



Problem #4: Discover directed graph.



Problem #5: Discover undirected graph.

- **Local examples**: 1, 2
- **Global examples**: 4, 5
- **Intermediate**: 3

Algorithms

- Global Discovery Algorithms
  - SGS (Spirtes-Glymour-Scheines)
  - **PC(Peter-Clark)** and variations(more conservative, stable)
  - TPDA Three Phase Dependency Algorithm
  - Recursive autonomy identification (RAI)
- Local Discovery Algorithms
  - Markov Blankets
  - Parent-and-Child
- Algorithms assuming the existence of latent variables
  - **FCI- Fast Causal Inference** ...and later variations(more conservative, faster and with mixed variables)

PC (Peter-Clark)

- A global constraint-based algorithm.
- Uses CI tests to eliminate and orient edges.
- Output: **CPDAG** (Completed Partially Directed Acyclic Graph) that represents an MEC of DAGs.
- Key assumptions:
    - **Causal Sufficiency:** No unmeasured confounders.
    - **Causal Markov Condition:** Independence of non-descendants given parents.
    - **Faithfulness:** No extra conditional independencies beyond the graph.

PC (Peter-Clark): Algorithm

- **Skeleton Identification**:
  - Start with a fully connected undirected graph.
  - Remove edges if variables are conditionally independent ($X \perp\!\!\!\perp Y | S$ removes X-Y).
- **Orientation:**
  - Identify v-structures ($X \rightarrow Z \leftarrow Y$, if $X \perp\!\!\!\perp Y | \emptyset$ and $X \not\perp\!\!\!\perp Y | Z$). Perform the adjacency phase with conditioning sets of increasing size.
  - Propagate orientations using adjacency rules("Meek Rules"): For each triple of variables such that $A \rightarrow B$ - C, and A and C are not adjacent, orient the edge $B - C$ as $B \rightarrow C$.
- **Output**: Completed Partially Directed Acyclic Graph (CPDAG) representing Markov equivalence class

If the conditional independence decisions are correct and all assumptions are fulfilled, the PC algorithm is guaranteed to converge to the true Markov Equivalence Class in the large sample limit.

PC (Peter-Clark): Example

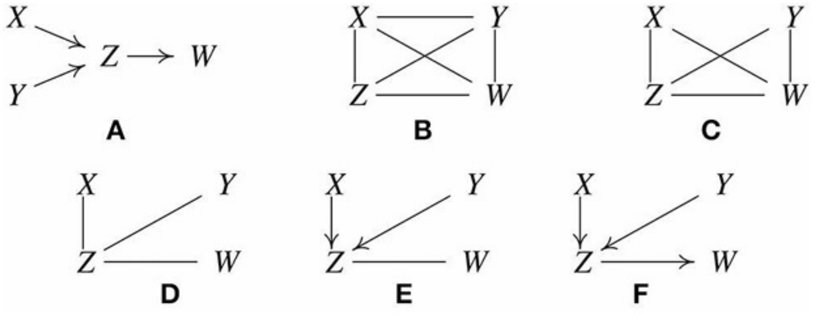


FIGURE 1 | Illustration of how the PC algorithm works. **(A)** Original true causal graph. **(B)** PC starts with a fully-connected undirected graph. **(C)** The $X - Y$ edge is removed because $X \perp\!\!\!\perp Y$. **(D)** The $X - W$ and $Y - W$ edges are removed because $X \perp\!\!\!\perp W \mid Z$ and $Y \perp\!\!\!\perp W \mid Z$. **(E)** After finding v-structures. **(F)** After orientation propagation.

PC (Peter-Clark): Pros and Cons

+ Simple to grasp, fundamental to other, more complex algorithms
+ Efficient
+ Asymptotically correct: guaranteed to obtain the correct Markov Equivalence Class as the sample size grows
+ Widely used

- Not stable, early-stage mistakes are crucial
- Does not account for latent confounders
- Computationally intensive and might be too heavy to calculate for high-dimensional data

PC (Peter-Clark): Python libraries

- `causal-learn`:
  - Implements **PC-stable** for order-independent results
  - Handles **missing data** via MV-Fisher_Z test
- `dodiscover`:
  - PyWhy ecosystem integration for full causal workflows
  - Built-in **Oracle CI tests** for validation studies
  - Time-series support via context specification
- `PyPCAlg`:
  - Pure Python port of R pcalg's PC
  - Custom CI test support via user-defined functions

Colab Example: PC with visualization (causal-learn)

FCI (Fast Causal Inference)

- Global constraint-based extension of PC for latent confounders
- Tolerates and sometimes discovers unknown confounding variables
- Uses CI tests with modified orientation rules for latent variables
- Output: **PAG** (Partial Ancestral Graph)(modern implementations, the original algorythm outputs Partially Oriented Inducing Path Graph)
- Key assumptions:
  - **No assumption on Causal Sufficiency:** Allows unmeasured confounders.
  - **Causal Markov Condition:** Retained from PC
  - **Faithfulness:** Retained from PC

FCI: Algorithm

- **Overview:**
  - FCI is similar to PC: it first determines adjacencies, then orients edges.
  - Output: **POIPG**, which is slightly less informative than a PAG.
- **Adjacency Phase:**
  - Determines adjacencies in the POIPG using conditional independence tests with conditioning sets of increasing size to optimise the search for separating sets (Sepsets).
  - Uses Possible-D-Sep(X), a computable superset of D-Sep(X). Therefore, it requires a more complex adjacency search strategy.
- **Orientation Phase:**
  - Orients edges in the POIPG based on the identified separating sets(Possible-D-Sep(X)) and orientation rules.

A lecture that explains FCI and prerequisites clearly: https://youtu.be/85W06fdEa10

FCI: Example

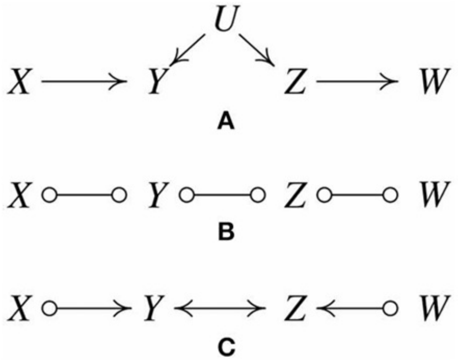


FIGURE 2 | Illustration of how the FCI algorithm is able to determine the existence of latent confunders. **(A)** Original true causal graph. **(B)** After edges are removed because of conditional independence relations. **(C)** The output of FCI, indicating that there is at least one unmeasured confounder of *Y* and *Z*.

FCI: Pros and Cons

+ Handles latent confounders
+ Asymptotically correct
+ Identifies testable confounding patterns

- Computationally intensive
- Not fully stable (requires more data for reliable outputs)
- Has ambiguous $\circ$marks

FCI: Implementation

- **causal-learn**:
  - Implements FCI and RFCI (Really Fast FCI)
  - Supports 12 CI tests
  - Visualizes PAGs
- **R**:
  - pcalg::fci() with gaussCItest
  - CompareCausalNetworks package for biomarker studies