Universidad Politécnica
de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Matemáticas e Informática

Trabajo Fin de Grado

Bachelor's Thesis

Descubrimiento Causal con Redes Bayesianas

Causal Discovery with Bayesian Networks

Autor: José María De Miguel Contreras
Tutores: Concha Bielza, Pedro Larrañaga

Madrid, Junio 2025

Este Trabajo Fin de Grado se ha depositado en la ETSI Informáticos de la Universidad Politécnica de Madrid para su defensa.

Trabajo Fin de Grado
Grado en Matemáticas e Informática

Título: Descubrimiento Causal con Redes Bayesianas

Junio 2025

Autor: José María De Miguel Contreras
Tutores: Concha Bielza, Pedro Larrañaga
Departamento de Inteligencia Artificial
Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid

Resumen

Los algoritmos de descubrimiento causal que dependen de *tests de independencia condicional* suelen asumir que el conjunto completo de variables está disponible desde el inicio, una suposición que deja de cumplirse cuando las variables llegan de manera secuencial como *streaming features*. Por ejemplo, aquellas variables que no pueden recolectarse completamente al inicio como las derivadas de redes sociales que dependen de noticias en tiempo real. Presentamos *FCI-FS*, una extensión del algoritmo *Fast Causal Inference (FCI)* para *feature streams (FS)* y el primer método basado en restricciones que es incremental, correcto, completo, y que actualiza el *grafo causal* cada vez que llega un nuevo lote de variables. *FCI-FS* trata el grafo aprendido en el paso previo como un *partial ancestral graph (PAG)* y, aprovechando un nuevo resultado teórico que muestra que cada arista eliminada en un PAG anterior permanece ausente al revelarse nuevas variables, *FCI-FS* reutiliza información previa de tests de independencia condicional para evitar tests redundantes. Una implementación *open-source* en Python, construida sobre la librería *pgmpy*, demuestra empíricamente las mejoras en tiempo de computación y eficiencia en el número de tests de independencia condicional logradas por *FCI-FS*.

Palabras clave: descubrimiento causal, aprendizaje automático, flujos de características, causalidad, redes Bayesianas.

Abstract

Causal discovery algorithms that rely on conditional independence (CI) tests usually assume that the full set of variables is available from the outset, an assumption that breaks down when features arrive over time as streaming features. For example, features that cannot be fully collected upfront, such as those derived from social networks that depend on real-time news. We introduce FCI-FS, an extension of the Fast Causal Inference (FCI) algorithm for feature streams (FS) and the first sound, complete, and incremental constraint-based method that refreshes the causal graph each time a new batch of features arrives. FCI-FS treats the graph learned in the previous step as a partial ancestral graph (PAG) and, leveraging a new theoretical result that shows every edge deleted in an earlier PAG remains absent after new variables are revealed, FCI-FS reuses prior CI information to avoid redundant tests. An open-source Python implementation, built on the pgmpy library, empirically highlights the runtime and CI tests efficiency gains achieved by FCI-FS.

Key words: causal discovery, machine learning, feature streams, causality, Bayesian networks.

Agradecimientos

Quiero mostrar mi más profundo agradecimiento a mis tutores Pedro Larrañaga y Concha Bielza, por acogerme en su grupo de investigación y permitirme explorar ideas nuevas y abrirme la mente. Su ejemplo y su compromiso con la investigación y con la excelencia son verdaderamente inspiradores, y su guía ha sido fundamental durante este último año para desarrollar ideas emocionantes que puedan aportar un granito de arena en el estado del arte.

En segundo lugar quiero agradecer a las instituciones que me han apoyado durante estos meses y durante toda la carrera, especialmente a la beca general del estado español del año 2024/2025, número 24AE/1405247. A la beca de colaboración para investigación del 2024/2025 número 24CO1/007185, y a la beca Santander ayuda económica dirigidas al estudiantado con los mejores expedientes académicos.

Finalmente quiero dar un fuerte agradecimiento a mi familia por su apoyo incondicional. A mi hermano Vicente por su espíritu aventurero, a mi padre, por su cariño y sabios consejos, a mis tíos Monchi y Elvira, por su constante respaldo, y a mi primo Guillermo, por su ejemplo inspirador. Finalmente, dedico especialmente este trabajo a quien sigue motivándome desde el cielo, a mi madre.

Acknowledgments

I would like to express my deepest gratitude to my advisors, Pedro Larrañaga and Concha Bielza, for welcoming me into their research group and allowing me to explore new ideas and open my mind. Their example and their commitment to research and excellence are truly inspiring, and their guidance has been fundamental over this past year in developing exciting ideas that may contribute, even in a small way, to the state of the art.

Secondly, I want to thank the institutions that have supported me during these months and throughout my entire academic journey, especially the Spanish government's general grant for the year 2024/2025, number 24AE/1405247; the research collaboration grant for 2024/2025, number 24CO1/007185; and the Santander scholarship for financial aid aimed at students with outstanding academic records.

Finally, I would like to give my heartfelt thanks to my family for their unconditional support: to my brother Vicente for his adventurous spirit, to my father for his love and wise advice, to my uncles Monchi and Elvira for their constant support, and to my cousin Guillermo for being an inspiring example. Above all, I dedicate this work especially to the one who continues to motivate me from heaven—my mother.

Contents

1. Introduction **1**

2. Prerequisites **7**
2.1. Terminology 7
2.2. Statistical and Causal Semantics Underlying DAGs 9
2.3. Statistical and Causal Semantics Underlying MAGs 10

3. Limits of Causal Sufficiency with Incremental Feature Streams **15**

4. The FCI Algorithm **17**

5. Design of the Proposed Algorithm **23**
5.1. The FCI-FS Algorithm 23
5.2. Example Execution of the FCI-FS Algorithm 29
5.3. Correctness of the FCI-FS Algorithm 29

6. Experiments **33**
6.1. Synthetic Datasets 33
6.2. Real-World Dataset 35

7. Impact Analysis **41**
7.1. Environmental Impact 41
7.2. Possible Adverse Effects 42
7.3. Development Decisions Influenced by Impact Considerations 42

8. Conclusion **43**

Bibliography **45**

Anexos **51**

A. Informe de Originalidad de Turnitin **51**

Chapter 1

Introduction

Discovering the causal mechanism governing an industrial or scientific process has always attracted attention due to its numerous practical applications, such as understanding whether the behavior of a component causes failure in industrial machinery or determining if a treatment causes improvement in a patient's condition. Often, learning causal relationships through randomized controlled trials is infeasible or impossible. Consequently, under Reichenbach's common-cause assumption, or "no correlation without causation", i.e., for every pair of dependent features, either one causes the other or there is a third feature causing both [1, 2], causal discovery becomes an essential tool to learn cause-and-effect relationships from observational data, providing insights beyond mere correlation and having numerous applications in fields such as genomics and decision-making systems.

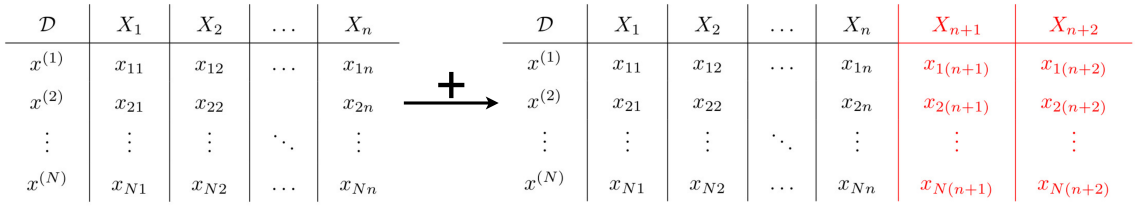
With the ever-growing dimensionality of data and the rapid development of new machine learning techniques, causal discovery encounters both fresh challenges and novel opportunities. Central to these challenges is the fact that knowledge of a system under study often evolves; for instance, in biological and medical research, initial studies may focus on well-established clinical indicators, but as science advances, features previously overlooked or unknown can become crucial. This dynamic scenario illustrates the concept of feature streams, where the set of observed variables is not static but continuously evolving while the number of instances remains constant (see Figure 1.1). Feature streams inherently emerge in situations where (1) the production of all features is resource-intensive, making it impractical to await their complete generation, (2) it is currently impossible to acquire all potential features, for instance, features derived from a social network that depend on up-to-date news, or (3) in incremental feature subset selection [3], features previously deemed irrelevant may become relevant as new knowledge about the system becomes available. Consequently, feature streams have become a highly researched topic within the machine learning community due to their applications in incremental feature subset selection and the implementation of feature streams in high-dimensional domains, including social networks [4], malicious URL detection [5], or long-lead prediction of extreme flood events [6].

Constraint-based algorithms [7, 8] use CI tests to recover the causal structure of data generated by a distribution with underlying causal semantics. Traditional constraint-based methods always assume that all features are available from the beginning and that no new features will emerge. These algorithms can be broadly categorized into two main types: global and local constraint-based algorithms.

The first category, global constraint-based algorithms, aims to reconstruct the causal graph encompassing all features of the dataset. A primary representative of this approach is the PC algorithm [7], which assumes causal sufficiency, i.e., all common confounders (common causes of at least two observed features, whose omission induces spurious correlations) are observed and no selection variables (which control sample inclusion and can introduce selection bias) are present. This algorithm has inspired extensive research over the decades [4, 9–17], and under suitable sparsity and regularity assumptions it has been proved to be asymptotically consistent even in high dimensionality scenarios [18], that means that the probability that the PC outputs the exact Markov equivalence class of the true causal DAG tends to one for an arbitrarily large sample size. However, the assumption of causal sufficiency is often problematic as it is typically violated in practice due to unobserved common causes. To address this limitation, the FCI algorithm [19] avoids causal sufficiency by learning the Markov equivalence class of a maximal ancestral graph (MAG), a generalization of DAGs that accommodates latent confounders and selection variables (see Section 2.3). The structure learned by FCI is represented by PAGs (Definition 2.3.3), and notably, FCI shares the same adjacency identification phase as the PC algorithm (Chapter 4). Both the PC and the FCI have been proved to recover the true causal DAG given perfect CI information [9, 20].

The second category, local constraint-based algorithms, focuses on learning local structures around individual variables. These algorithms are particularly useful for tasks such as local causal discovery and variable selection. Additionally, they can construct the skeleton of a DAG inductively by sequentially evaluating each new feature to determine if previously considered features belong to its local structure [21]. For a given variable X, two types of local structures are commonly targeted: (1) the Markov blanket of X, denoted as $MB(X)$ [21, 22], defined as the minimal set of variables that renders X independent from all other variables; and (2) the parents and children of X, represented as $PC(X)$ [21, 23, 24], corresponding to variables directly adjacent to X in the DAG. An essential property leveraged by these algorithms to reduce computational cost is the symmetry of the parents and children set: if $Y \in PC(X)$, then it follows that $X \in PC(Y)$. Typically, these methods initialize with an empty local structure for each new variable, iteratively adding and removing adjacencies through alternating forward phases (which introduce new adjacencies by adding elements to the local structure) and backward phases (which refine the structure by removing previously added adjacencies).

Although a growing body of work tackles incremental causal discovery in streaming settings, every method proposed to date is local; indeed, naively extending constraint-based algorithms to feature streams would require repeating CI tests each time a new feature arrives, with the worst-case complexity growing expo-



$\mathcal{D}$	X_1	X_2	$\dots$	X_n
$x^{(1)}$	x_{11}	x_{12}	$\dots$	x_{1n}
$x^{(2)}$	x_{21}	x_{22}		x_{2n}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$x^{(N)}$	x_{N1}	x_{N2}	$\dots$	x_{Nn}

+

$\mathcal{D}$	X_1	X_2	$\dots$	X_n	X_{n+1}	X_{n+2}
$x^{(1)}$	x_{11}	x_{12}	$\dots$	x_{1n}	$x_{1(n+1)}$	$x_{1(n+2)}$
$x^{(2)}$	x_{21}	x_{22}		x_{2n}	$x_{2(n+1)}$	$x_{2(n+2)}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
$x^{(N)}$	x_{N1}	x_{N2}	$\dots$	x_{Nn}	$x_{N(n+1)}$	$x_{N(n+2)}$

Figure 1.1: Dataset $\mathcal{D}$, which receives two new features, where X_i denotes a feature and $x^{(i)} = (x_{i2}, \dots, x_{in})$ is the i-th vector of instantiations. For any feature X_i the value x_{ik} is taken at the instantiation $x^{(k)}$.

nentially with the number of variables. Such recomputation is infeasible in real-time applications. As a result, existing causal discovery algorithms for streaming features deliberately trade global guarantees for the runtime improvements and reduced number of CI tests offered by local constraint-based algorithms. Yu et al. [25, 26] introduced S-CDFSF, a two-phase algorithm that learns the skeleton of a DAG for each new feature using the symmetry property of the local PC set. Specifically, it first executes a forward phase to add adjacencies by attempting to incorporate the new feature into the candidate parents and children (CPC) of existing features, followed by a backward phase that removes false positives. Guo and Yang [27], Yang et al. [28, 29] developed several related approaches: the CSBS, CSSU, and PRCDSF algorithms, respectively. Each one begins with a relevance analysis step. In this step, previously observed features are added to a temporary Markov blanket of the newly arrived feature. Subsequently, each algorithm performs a redundancy analysis to prune the provisional blanket and updates the blankets of previously seen features containing the new feature. Once all features have arrived, these methods utilize the resulting skeleton to orient the DAG through a greedy hill-climbing search using the *minimum description length* score. Furthermore, You et al. [30] introduced algorithms for local causal discovery that specifically focus on scenarios involving a single target variable whose causal relationships evolve continuously with the arrival of new features. They proposed two algorithms: OL_{aMB}, which maintains a running "approximate Markov blanket" by discarding streaming features that become conditionally independent from the target, and LCSL_{SF}, which converts the remaining features into a directed local DAG (which contains the target variable and its local structure) by identifying v-structures and propagating directions using Meek's rules [31]. These approaches ensure a rapid updated map of the target's direct causes and effects.

Despite these significant contributions, the existing algorithms exhibit several limitations (see Table 1.1). First, they commonly assume that the features stream sequentially one by one into the network. This simplification often does not reflect real-world scenarios in which it might be advantageous to evaluate groups of features collectively, e.g., Image analysis often involves extracting several descriptors that come in groups to represent different visual aspects, but computing all these features can be extremely time-consuming. For this reason, group feature streams can be leveraged to process the data more efficiently [32]. Second, excepting the LCSL_{SF} algorithm, current algorithms either require wait-

Algorithm	Learns Global Structure	Group Streaming	Incremental	Avoids Causal Sufficiency	Correctness
S-CDFS [26]	✓	✗	✗	✗	✗
CSBS [27]	✓	✗	✗	✗	✗
CSSU [28]	✓	✗	✗	✗	✗
PRCDSF [29]	✓	✗	✗	✗	✗
$LCSL_{SF}$ [30]	✗	✗	✓	✗	✗

Table 1.1: High-level comparison of current algorithms for causal structure learning with feature streams. Columns indicate whether the method learns the full global structure or only local structures around a target variable, supports group streaming (Group Streaming), operates incrementally, avoids assuming causal sufficiency, and guarantees correctness. Ticks (✓) indicate the property is present, and crosses (✗) indicate it is absent.

ing until all features have arrived to orient the skeleton of the DAG or entirely omit orientations. Consequently, if additional features appear after training, the model must be re-trained from scratch. As a result, these methods are not incremental, as the causal structure becomes available only after processing all features, as in traditional constraint-based methods. This limitation is particularly critical in contexts where features take considerable time to arrive and in high-dimensionality scenarios. On the other hand, the $LCSL_{SF}$ allows incrementally adding new incoming features, but it learns only the local DAG around the target variable, not paying attention to the full global causal structure. Third, these algorithms rely on the local direct Markov property (Section 2.2), which guarantees that two features not connected in the causal structure become conditionally independent when conditioned on the parents of one of them. This property inherently depends on the assumption of causal sufficiency, which is violated in the presence of latent confounders [9, pp. 168–173], and therefore cannot be guaranteed in the context of incremental feature streams where only a subset of features is considered and new potential confounders can arrive (Chapter 3). Finally, to the best of our knowledge, existing algorithms do not yet provide formal guarantees of correctness under perfect CI information to recover the true causal structure, which may reduce confidence in their long-term performance for incremental or open-ended learning scenarios.

To avoid the previously mentioned limitations, this work proposes a new version of the FCI algorithm for feature streams, called FCI-FS, an incremental global constraint-based causal discovery algorithm for group feature streams. The algorithm updates the graph each time a new batch of features arrives, leveraging the previously learned PAG to reduce the number of CI tests, which constitute the primary computational cost in constraint-based methods, and is asymptotically correct. Spirtes et al. [19] developed the FCI, which does not require the causal sufficiency assumption. The initial version of FCI identified the correct adjacencies, given the perfect CI information; however, the output was not maximally informative (i.e., more information about the edge orientations of

the causal structure could be obtained from data). Its output was a partially oriented inducing path graph (POIPG), which represents the concept of inducing path graphs (IPG) introduced by Verma and Pearl [8] to graphically represent marginalized distributions and latent confounders. Later, Richardson and Spirtes [33] introduced MAGs, which allowed for both selection and latent variables and had deeper statistical foundations, and Zhang [34] (in the appendix) showed that POIPGs are just incomplete PAGs, and made the FCI complete by adding graphical orientation rules [20], interpreting the output of FCI as PAGs using the notation of MAGs instead of IPGs. Furthermore, to improve computational complexity, Spirtes [35] introduced the anytime FCI that interrupts the adjacency phase, and Colombo et al. [36] introduced a much faster sound and complete version suitable for high-dimensional graphs, the RFCI algorithm, but both versions produce possibly less informative structures.

Since most of the software available for causal discovery is not incremental, this work also introduces an initial Python implementation that can be found at GitHub [1]. The code is based mainly on the package pgmpy [37], with modifications both in the graph structure and in the constraints-based methods. Although this work focuses solely on proposing a new version of the FCI, to keep the initial testing manageable, this initial version is based on the PC, which has a simpler structure differing in its pseudo-code very little since the FCI is a modification of the PC that avoids causal sufficiency. This simplification leaves some edges even when features are CI, thus reducing the performance with graphical scores; however, the behavior of the PC approximates the FCI enough to validate the theoretical results of the proposed FCI-FS empirically. This initial implementation was used to evaluate FCI-FS on two types of datasets. The first type is a synthetic dataset generated from 256 randomly constructed sparse graphs, each comprising 200 variables. Sparse graphs were specifically chosen because global constraint-based methods, such as the PC algorithm and its variants, are explicitly optimized for sparse graph scenarios, allowing them to perform fewer CI tests [7]. The training was conducted incrementally in batches, using eight distinct variable orderings per graph and experimenting with various parameter settings. The performance of the proposed FCI-FS algorithm was benchmarked against both the traditional, non-incremental PC algorithm [9] and the PC-stable variant [12], with both methods applied to the dataset after all features had become entirely available. The second type is the real-world dataset ApisTox [38], which we use to learn the causal structure of pesticide toxicity data.

The remainder of this work is organized as follows: Chapter 2 introduces prerequisites, including terminology and the causal and statistical semantics of DAGs and their latent-variable counterparts, MAGs. Chapter 3 discusses why the causal sufficiency assumption fails in the context of feature streams and shows how MAGs can solve this issue. Chapter 4 introduces the original FCI algorithm, laying the groundwork for the proposed FCI-FS algorithm, whose formal presentation and proof of correctness are provided in Chapter 5. Chapter 6 demonstrates the performance of FCI-FS on synthetic and real-world datasets, and Chapter 7 analyzes the impact. Finally, Chapter 8 concludes the work.

[1] https://github.com/Jose-Maria-De-Miguel/FCI-FS.git.

Chapter 2

Prerequisites

This chapter introduces basic probabilistic and graph terminology and concepts about graphical models and their causal and probabilistic interpretations.

2.1. Terminology

We denote a probability distribution as p. Features are denoted by capital letters (e.g., X, Y, Z), potentially with subindices to distinguish features ($e.g., X_1, X_2$, Y_1, Y_2), sets of features are indicated in boldface (e.g., $\mathbf{X}$ and $\mathbf{Y}$), and the disjoint union of sets of features as $\mathbf{X} \sqcup \mathbf{Y}$. We consider a *graph* $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ which includes a finite set of features $\mathcal{X} = \{X_1, \ldots, X_n\}$ and edges $\mathcal{E}$, which may be *undirected* ($X - Y$), *directed* ($X \to Y$), *bidirected* ($X \leftrightarrow Y$), *nondirected* (○–○), *partially undirected* (○–), or *partially directed* (○→). Edge *endpoints* are shown with a *tail* (-), an *arrowhead* (>), or a *circle* (o). The *asterisk* (*) acts as a wildcard, standing for any of these endpoint symbols. A graph is *complete* if all its nodes are connected to each other. A graph is called *directed* if it only contains directed edges, *undirected* if all edges are undirected, and *mixed* if it contains a combination of both directed and undirected edges. A *simple* graph does not allow for self-adjacency or multiple edges between two features.

Given adjacent features X and Y, Y is the *parent* of X if $Y \to X \in \mathcal{E}$, *child* if $X \to Y \in \mathcal{E}$, *neighbor* if $Y - X \in \mathcal{E}$, and *spouse* if $Y \leftrightarrow X \in \mathcal{E}$. The respective sets of parents, children, neighbors, and spouses $\mathcal{G}$ are denoted as $Pa_{\mathcal{G}}(X)$, $Ch_{\mathcal{G}}(X)$, $Ne_{\mathcal{G}}(X)$, and $Sp_{\mathcal{G}}(X)$. A *path* is a sequence of distinct adjacent features $\langle X_1, X_2, \ldots, X_n \rangle$. A *directed path* has all edges consistently oriented, with each arrow pointing in the same direction (e.g., $X_1 \to X_2 \to \cdots \to X_n$); for example, a sequence like $X \to Y \leftarrow Z$ is not a directed path. If there is a directed path from Y to X we say Y is an *ancestor* of X and X is a *descendant* of Y; the respective sets of ancestors and descendants are $An_{\mathcal{G}}(X)$ and $De_{\mathcal{G}}(Y)$. A *directed cycle* occurs in $\mathcal{G}$ when $X \to Y \in \mathcal{E}$ and $X \in De_{\mathcal{G}}(Y)$. An *almost directed cycle* occurs when $Y \leftrightarrow X \in \mathcal{E}$ and $X \in An_{\mathcal{G}}(Y)$. Let $\langle X, Y, Z \rangle$ be a path, the middle vertex, Y, is called a *collider* if both arrows of the path are pointing at Y (the path takes the form $X \ast\!\!\to Y \leftarrow\!\!\ast\ Z$); if X and Z are not adjacent, we call Y

an *unshielded* collider. Any triple (X, Y, Z) where Y is an unshielded collider is called a *v-structure*. A *triangle* is a triple (X, Y, Z), where all the vertices are adjacent. An *inducing path* concerning a vertex set $\mathbf{L}$ has every internal vertex that is not in $\mathbf{L}$ as a collider on the path, and each collider is an ancestor of at least one of the path's endpoints; henceforth, if the set $\mathbf{L}$ is not mentioned we presume that $\mathbf{L} = \emptyset$ (e.g., in Figure 2.1a, $\langle X_1, X_2, X_3, X_4, X_5, X_6 \rangle$ is an inducing path between X_1 and X_6). The simplest inducing path consists of one vertex and two adjacent features $X \ast\!\!\ast Y$.

An *independence model* $\mathcal{I}$ over a set of variables $\mathcal{X}$ is a set of conditional independencies between subsets of variables, denoted as $\langle \mathbf{X}, \mathbf{Y} | \mathbf{Z} \rangle$, with nonempty disjoint sets $\mathbf{X}$ and $\mathbf{Y}$, and possibly empty $\mathbf{Z}$. The conditional independencies generated by a probability distribution, p, are indicated as $\mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z}$, and as $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}$ if $\mathbf{Z}$ is empty. The conditional independencies of a graph $\mathcal{G}$ are denoted as $\mathbf{X} \perp_m \mathbf{Y} | \mathbf{Z}$ (see Definition 2.3.2). The CI model derived from a probability distribution p is denoted;

$$\mathcal{I}(p) = \{\mathbf{X} \perp\!\!\!\perp \mathbf{Y} | \mathbf{Z} : \mathbf{X}, \mathbf{Y}, \mathbf{Z} \subset \mathcal{X} \text{ are disjoint sets, and } \mathbf{X}, \mathbf{Y} \neq \emptyset\} \tag{2.1}$$

Let $\mathcal{X} = \mathbf{O} \sqcup \mathbf{L} \sqcup \mathbf{S}$ be a set of variables, where $\mathbf{O}$ are observable variables, $\mathbf{L}$ are latent variables, and $\mathbf{S}$ are selection variables. We denote the marginalization of latent variables using the notation $[_{\mathbf{L}}$ (e.g., marginalizing $\mathbf{L}$ in the graph $\mathcal{G}$, $\mathcal{G}[_{\mathbf{L}}$). To condition on a subset of selection variables, we denote it with the notation $[^{\mathbf{S}}$ (e.g., conditioning $\mathbf{S}$ in the distribution p, $p[^{\mathbf{S}}$). An independence model under these operations behaves as follows [33, pg. 980]: to marginalize an independence model over $\mathbf{L}$, we retain only the subset of independencies that do not involve variables in $\mathbf{L}$:

$$\mathcal{I}[_{\mathbf{L}} \equiv \{\langle \mathbf{X}, \mathbf{Y} | \mathbf{Z} \rangle \in \mathcal{I} : (\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}) \cap \mathbf{L} = \emptyset\}$$

Similarly, the conditioned independence model conditioned by $\mathbf{S}$ is defined as:

$$\mathcal{I}[^{\mathbf{S}} \equiv \{\langle \mathbf{X}, \mathbf{Y} | \mathbf{Z} \rangle : \langle \mathbf{X}, \mathbf{Y} | \mathbf{Z} \cup \mathbf{S} \rangle \in \mathcal{I}; (\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}) \cap \mathbf{S} = \emptyset\}$$

Combining both, we obtain the following.

$$\mathcal{I}[^{\mathbf{S}}_{\mathbf{L}} \equiv \{\langle \mathbf{X}, \mathbf{Y} | \mathbf{Z} \rangle : \langle \mathbf{X}, \mathbf{Y} | \mathbf{Z} \cup \mathbf{S} \rangle \in \mathcal{I}; (\mathbf{X} \cup \mathbf{Y} \cup \mathbf{Z}) \cap (\mathbf{L} \cup \mathbf{S}) = \emptyset\} \tag{2.2}$$

To deal with latent confounders or selection bias, MAGs prove especially beneficial by depicting marginalization and conditioning without directly including hidden variables in the graph (see Section 2.3 and Chapter 4).

2.2. Statistical and Causal Semantics Underlying DAGs

Under the assumptions of acyclicity, causal sufficiency, and the common-cause principle (see Chapter 1) Pearl [1] introduces three models graphically represented by DAGs. The first model, Bayesian networks (BNs), consists of purely statistical models that compactly encode joint probability distributions, facilitating reasoning under uncertainty. In contrast, causal BNs (CBNs) are BNs that interpret directed edges explicitly as causal relationships, allowing for reasoning about interventions and addressing cause-effect questions. The third model, structural causal models (SCMs), combines three distinct elements: (1) a set of structural equations quantitatively specifying how each feature is generated from its parent features, (2) a set of exogenous random variables (noise) described by a joint probability distribution, and (3) a causal graph representing these functional relationships. Markovian SCMs are a subclass of SCMs represented by DAGs, characterized by having all noise variables independent of each other, allowing counterfactual reasoning queries[1], which support richer and more human-like causal reasoning than the previous formulations.

Only CBNs and SCMs inherently carry causal interpretations. Given two features X and Y, if $X \in Pa_{\mathcal{G}}(Y)$ then X is a direct cause of Y. Otherwise, if $X \in An_{\mathcal{G}}(Y)$, X is an indirect cause of Y. Thus, when referring to learning causal DAGs, the focus is on learning the causal graphs representing either a CBN or a Markovian SCM (excluding noise terms). A fundamental condition for both CBNs and Markovian SCMs is the fulfillment of the causal Markov condition.

Definition 2.2.1 (Causal Markov condition [39])**.** *Let $\mathcal{G}$ be a causal graph with a feature set $\mathcal{X}$, and let p be a probability distribution over $\mathcal{X}$ generated by the causal structure represented by $\mathcal{G}$. Then, $\mathcal{G}$ and p satisfy the causal Markov condition if and only if every X in $\mathcal{X}$ is independent of $\mathcal{X} \backslash (Pa_{\mathcal{G}}(X) \sqcup De_{\mathcal{G}}(X))$ given $Pa_{\mathcal{G}}(X)$.*

A consequence of the causal Markov condition is that two nonadjacent features in $\mathcal{G}$ become conditionally independent with p when conditioned on the parents of one of them. We will refer to this as the *local Markov property*. This entails a slightly different meaning from Lauritzen et al. [40], but pays off by allowing us to easily distinguish between the independence of two specific features and the independence between a feature and all its nondescendants. Additionally, the causal Markov condition has two consequences. The first one, the *pairwise Markov property*, guarantees that for every two nonadjacent features, there exists a subset of the feature space such that, when conditioned on, the pair becomes independent. This is a broader property than the *local Markov property* because it does not specify which sets render them independent. The second consequence is that the causal Markov condition allows the joint distribution to factor causally into the product of components defined by causal relationships:

[1] Counterfactuals are "what if" queries—hypothetical statements about what would have happened to an outcome if a different intervention had occurred. In Pearl's ladder of causation [1], counterfactual reasoning forms the highest rung, enabling questions not just about seeing (associations) or doing (interventions), but imagining alternative realities and their consequences.

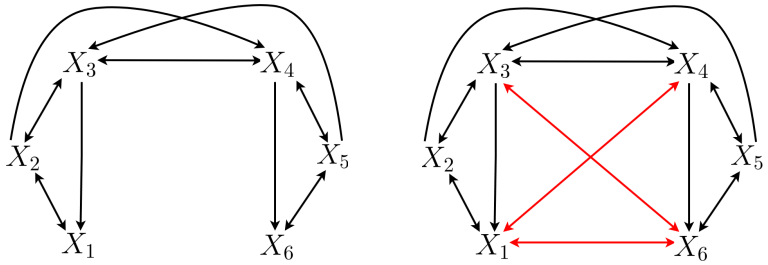


Figure 2.1: Two mixed graphs: (a) is an AG that is not X_6 maximal and (b) is the equivalent MAG.

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i | Pa_{\mathcal{G}}(X_i))$$

Despite the causal semantics of the causal Markov condition, it only specifies a set of local independencies. To capture the full independence model implied by a DAG, we use a graphical criterion known as *d-separation*—a special case of *m-separation* for DAGs (see Definition 2.3.2), which determines CI holds.

2.3. Statistical and Causal Semantics Underlying MAGs

Since feature streams allow the incorporation of new potential confounders, the available variables do not necessarily satisfy the causal sufficiency assumption. However, DAGs do not allow for the representation of latent confounders or selection variables [19]. Nevertheless, there is a graphical model that generalizes DAGs by enabling the implicit incorporation of hidden variables: ancestral graphs (AGs). AGs allow for both the marginalization of the graphical model to represent latent confounders and conditioning to represent selection bias.

Definition 2.3.1 (Ancestral graph [20, 33])**.** *A mixed graph (see Section 2.1), $\mathcal{G}$, is ancestral if the following three conditions hold:*

1. *It does not contain any directed cycle (e.g., $Y = X_1 \to X_2 \cdots \to X_n = Y$).*
2. *It does not contain any almost directed cycle (e.g., $Y = X_1 \to X_2 \cdots \to X_{n-1} \leftrightarrow X_n = Y$).*
3. *If it has the edge $X - Y$, then $Pa_{\mathcal{G}}(X) = Sp_{\mathcal{G}}(X) = Pa_{\mathcal{G}}(Y) = Sp_{\mathcal{G}}(Y) = \emptyset$ (neither X nor Y have spouses or parents).*

It is simple to see that DAGs are AGs with all their vertices directed. The main reason we are interested in AGs is the notion of m-separation, which generalizes d-separation from DAGs to ancestral graphs and serves as a graphical criterion for reading conditional independencies from graphs.

Definition 2.3.2 (m-separation [33]). *Let $\mathcal{G}$ be an AG over a set of random variables $\mathcal{X}$, and let $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathcal{X}$ be disjoint sets of variables. A path $\pi = \langle X = X_1, \ldots, X_n = Y \rangle$ between a variable, $X \in \mathbf{X}$ and a variable $Y \in \mathbf{Y}$ is said to be* blocked *by $\mathbf{Z}$ if and only if at least one of the following conditions holds:*

- *π contains a triple $\langle X_i, X_{i+1}, X_{i+2} \rangle$ where X_{i+1} is not a collider (e.g., $X_i - X_{i+1} - X_{i+2}$, $X_i \leftrightarrow X_{i+1} \rightarrow X_{i+2}$, etc.) such that X_{i+1} is in $\mathbf{Z}$.*
- *π contains a collider X_i (i.e., $X_i \ast\!\!\rightarrow X_{i+1} \leftarrow\!\!\ast\ X_{i+2}$) such that $X_{i+1} \notin An_{\mathcal{G}}(\mathbf{Z})$.*

If all paths between any variable in $\mathbf{X}$ and any variable in $\mathbf{Y}$ are blocked by $\mathbf{Z}$, then $\mathbf{X}$ and $\mathbf{Y}$ are said to be m-separated *given $\mathbf{Z}$ in $\mathcal{G}$, we denote this as $\mathbf{X} \perp_m \mathbf{Y} \mid \mathbf{Z}$. If two variables cannot be m-separated, we say they are unseparable. This allows us to have a CI model derived from an ancestral graph $\mathcal{G}$,*

$$\mathcal{I}(\mathcal{G}) = \{\mathbf{X} \perp_m \mathbf{Y} | \mathbf{Z} : \mathbf{X}, \mathbf{Y}, \mathbf{Z} \subset \mathcal{X} \text{ are disjoint sets, and } \mathbf{X}, \mathbf{Y} \neq \emptyset\} \tag{2.3}$$

The major limitation of AGs is that they do not satisfy the pairwise Markov property, meaning that two non-adjacent features may still be unseparable, because unseparability occurs if and only if the features are connected by an inducing path [41, pg. 53; 33, pg. 984]; for instance, X_1 and X_6 are unseparable in Figure 2.1a. To address this issue, Richardson and Spirtes [33] introduced MAGs, which represent the same independencies of AGs and satisfy the pairwise Markov property, a condition known as maximality (e.g., Figure 2.1b is the MAG corresponding to Figure 2.1a). Because DAGs are a special case of MAGs, one can graphically marginalize over a latent set $\mathbf{L}$ and condition on a set $\mathbf{S}$ in any DAG or, more generally, in an AG $\mathcal{G}$ to obtain a MAG $\mathcal{M} = \mathcal{G}[^{\mathbf{S}}_{\mathbf{L}}$ that represents the marginalized independence model $\mathcal{I}(\mathcal{G})[^{\mathbf{S}}_{\mathbf{L}}$ by using the following construction (An example of each step of the construction in Figure 2.2 which is inspired by Verma and Pearl [8]):

Input: An AG (or DAG) $\mathcal{G}$ over the feature space $\mathcal{X} = \mathbf{X} \sqcup \mathbf{S} \sqcup \mathbf{L}$ (example input in Figure 2.2, where $\mathbf{X} = \{X_1, X_2, X_3, X_4\}$, $\mathbf{L} = \{Y\}$, and $\mathbf{S} = \emptyset$).

Output: A MAG $\mathcal{M}$ over $\mathbf{X}$ (Figure 2.2c).

1. **Construct** a graph $\mathcal{M}$ on $\mathbf{X}$ by connecting X_i and X_j whenever there exists an inducing path contained in $\mathbf{X}$ with respect to $\mathbf{L} \sqcup \mathbf{S}$ linking them in $\mathcal{G}$ (Figure 2.2b from the example, where $\mathbf{L} \sqcup \mathbf{S} = \{Y\}$).
2. **Assign edge directions:** For each pair of adjacent features (X_i, X_j) place an arrowhead pointing at X_i whenever $X_i \notin An_{\mathcal{G}}(\{X_j\} \cup \mathbf{S})$ or place a tail otherwise, and proceed symmetrically for X_j (Figure 2.2c).

The resulting MAG satisfies $\mathcal{I}(\mathcal{M}) = \mathcal{I}(\mathcal{G}[^{\mathbf{S}}_{\mathbf{L}}) = \mathcal{I}(\mathcal{G})[^{\mathbf{S}}_{\mathbf{L}}$. Notice that to expand an AG into a MAG, it is only necessary to add bidirected edges between any pair of nonadjacent features that are connected by an inducing path. In a MAG, edge types convey clear causal meanings: $X \rightarrow Y$ implies that X is a cause of Y or of a selection variable, while Y is not a cause of X; $X \leftrightarrow Y$ indicates that X and Y are correlated solely because of an unobserved confounder, with neither causing the other or any selection variable; and $X - Y$ means either X causes Y, Y causes X, or one (or both) causes a selection variable [20].

Several different MAGs can encode the same independencies, but all of these can be encapsulated within a Markov equivalence class, denoted as $[\mathcal{M}]$. To provide a graphical characterization of Markov equivalence, it is essential to introduce the concept of a discriminating path between two features X_1, X_n relative to a feature Y. Such discriminating paths have the form $\langle X_1, X_2, \ldots, X_{n-1} = Y, X_n \rangle$, and must satisfy three properties: (1) the path includes at least three vertices; (2) Y is a nonendpoint vertex on the path and is adjacent to X_n; and (3) X_1 is not adjacent to X_n, and every feature along the path between X_1 and Y acts as both a collider on the path and a parent of X_n (see Figure 2.3). With these definitions in place, it becomes possible to consider the following proposition [42].

Proposition 2.3.1. *Two MAGs with the same set of features are Markov equivalent if and only if*

(i) Their adjacencies are identical.

(ii) Their unshielded colliders are identical.

(iii) Whenever a path is a discriminating path for vertex Y in both MAGs, Y assumes the same role on each path: it is a collider on both paths or neither.

Just as the Markov equivalence class of a DAG can be captured by a *partially directed acyclic graph* (PDAG), the Markov equivalence class of a MAG is represented by a partial ancestral graph (PAG).

Definition 2.3.3 (PAG)**.** *Given the Markov equivalence class of a MAG $\mathcal{M}$, $[\mathcal{M}]$, a partial ancestral graph (PAG), $\mathcal{P}$, that represents $[\mathcal{M}]$ is a graphical structure with six types of edges: $-, \rightarrow, \leftrightarrow, \circ\!\!-\!\!\circ, \circ\!\!-, \circ\!\!\rightarrow$, and three types of marks $o, >, -$ such that (i) $\mathcal{P}$ has the same adjacencies as any member of $[\mathcal{M}]$; and (ii) every non-circle mark is an invariant mark across all the MAGs in $[\mathcal{M}]$, i.e., all the MAGs in $[\mathcal{M}]$ must have that exact mark.*

If, furthermore, $\mathcal{P}$ satisfies that every circle mark indicates that there are equivalent MAGs with different marks on that end of the edge, then we call it a *maximally informative PAG*. The FCI algorithm (Chapter 4) outputs a PAG representing the equivalence class of the corresponding generating MAG. If the PAG is maximally informative, then the FCI version is complete; this was achieved by Zhang [20] by adding direction rules to the algorithm.

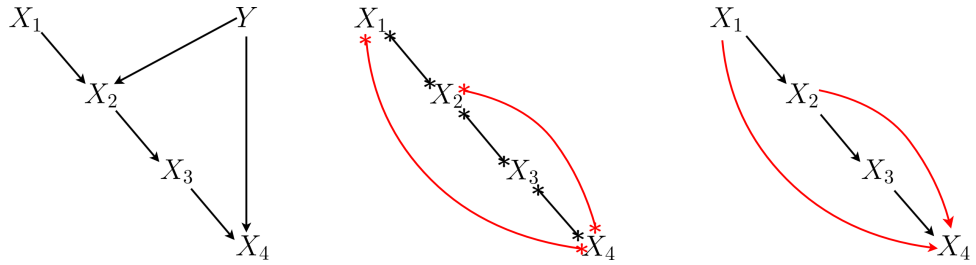


Figure 2.2: Graphical result of the steps to marginalize feature Y in the DAG from (a), in (b) the result from the first step, making adjacent all the features connected by an inducing path given Y. The wildcard '$*$' is used to denote that the directions have not been added yet. In (c), the directions are added according to the second step of the construction.

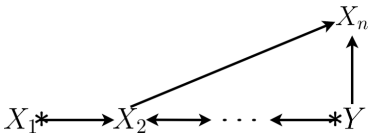


Figure 2.3: Illustration of a discriminating path in which vertex Y is discriminated. Endpoints labelled '$<$' or '$-$' are fixed; every discriminating path must include them. The wildcard '$*$' indicates that any endpoint symbol may appear in that position.

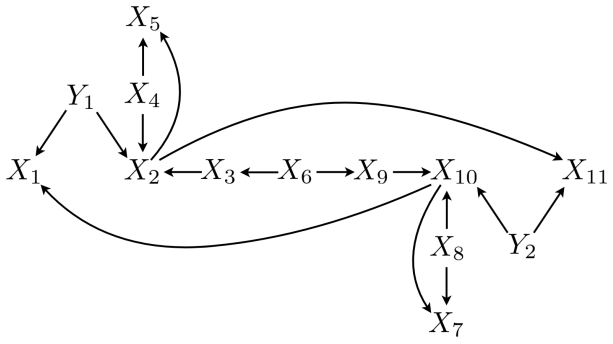
Chapter 3

Limits of Causal Sufficiency with Incremental Feature Streams

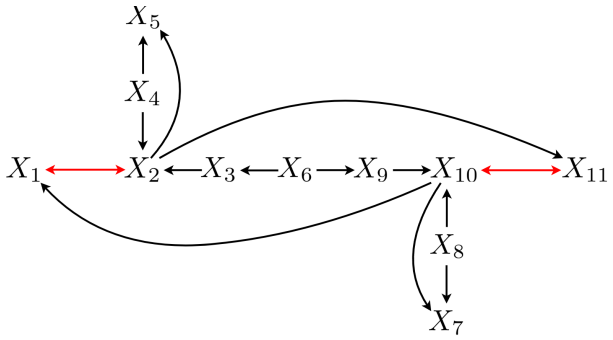
Causally sufficient discovery methods, such as the PC algorithm or the local structure learning methods presented in the introduction, rely fundamentally on the *local direct Markov property*. This property states that any two features not adjacent in the causal DAG become independent once conditioned on the parents of one of these features. This property is fundamental in limiting the search space for separating sets and thereby reducing algorithmic complexity; however, it does *not* hold in the presence of latent confounders. Consequently, it generally fails in scenarios involving incremental feature streams because the causal MAGs must be available at every stage, even before all features have been observed. Intermediate MAGs can therefore contain latent confounders, violating the causal sufficiency assumption during those intermediate steps. An illustrative example from Spirtes et al. [9, Chapter 6] demonstrates this issue. Consider the DAG $\mathcal{G}$ in Figure 3.1a under the latent confounders, Y_1 and Y_2. The corresponding MAG is shown in Figure 3.1b. A causally sufficient algorithm trying to learn its MAG would incorrectly conclude that the features X_1 and X_{11} are unseparable because they remain dependent given only their observed adjacencies. However, X_1 and X_{11} can, in fact, be d-separated by conditioning on the set $\{X_2, X_6, X_{10}\}$.

As a consequence of the previous result, in a MAG, two m-separable features, X_i and X_j, may only be separated once conditioned on nonadjacent variables. To solve this, Spirtes et al. [9, Chapter 6] introduced two types of sets which play a similar role to that of parents and adjacent features in DAGs.

The first one is the set **D-SEP**(X_i, X_j). We say that a feature X_k is in **D-SEP**(X_i, X_j) in $\mathcal{M}$ if and only if $X_k \neq X_i$ and there exists a path between X_i and X_k on which every nonendpoint vertex is a collider in the path and an ancestor of X_i or X_j. Specifically, $Pa_{\mathcal{M}}(X_i) \subseteq$ **D-SEP**$(X_i, \cdot)$, the equality holding for DAGs. In a MAG, two separable features, X_i and X_j, become independent once conditioned on



(a)



(b)

Figure 3.1: (a) DAG defined over variables $\mathcal{X} = \mathbf{O} \sqcup \mathbf{L}$, where observable variables are $\mathbf{O} = \{X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}, X_{11}\}$ and latent variables are $\mathbf{L} = \{Y_1, Y_2\}$, with its corresponding MAG in (b). Figures inspired by Spirtes [35, Chapter 6]

D-SEP(X_i, X_j), or **D-SEP**(X_j, X_i) (generally, **D-SEP**$(X_i, X_j) \neq$ **D-SEP**(X_j, X_i)). For example, in Figure 3.1b, **D-SEP**$(X_1, X_{11}) = \{X_2, X_3, X_4, X_{10}\}$ and **D-SEP**$(X_{11}, X_1) = \{X_2, X_8, X_9, X_{10}\}$. X_1 and X_{11} can be separated given either of the previous two sets.

The second set is merely a relaxed version of the first one, it consists of possible features that could be in **D-SEP**(X_j, X_i). We say a feature X_k is in **Possible-D-SEP**(X_i, X_j) in a PAG $\mathcal{P}$ if and only if $X_k \neq X_i$ and all the nonendpoints are colliders in the path, or have their orientation hidden because they form a triangle. Since the definition of **Possible-D-SEP**(X_i, X_j) does not consider the second term, we will denote it as **Possible-D-SEP**(X_i). If $\mathcal{P}$ is a PAG of $[\mathcal{M}]$, containing all its unshielded colliders, then **D-SEP**$(X_i, \cdot) \subseteq$ **Possible-D-SEP**(X_i). For example, in Figure 3.1b, **Possible-D-SEP**$(X_1, X_{11}) = \{X_2, X_3, X_4, X_5, X_{10}\}$ and **Possible-D-SEP**$(X_{11}, X_1) = \{X_2, X_7, X_8, X_9, X_{10}\}$. Finally, since in the causal discovery tasks it is not possible to recover all the edge marks specified, it is not possible to determine all the **D-SEP**s, but given the unshielded colliders, it is possible to recover **Possible-D-SEP**.

Chapter 4

The FCI Algorithm

Constraint-based algorithms descending from the PC algorithm share the same basic properties and assumptions (see Table 4.1 for a glossary); however, FCI stands out because it accommodates latent confounders and selection bias by operating with MAGs instead of DAGs. Specifically, the PC algorithm assumes two conditions regarding the data-generating distribution p: (i) the causal Markov condition (Definition 2.2.1), stating that the distribution factorizes according to a DAG, $\mathcal{G}$, so that $\mathcal{I}(\mathcal{G}) \subseteq \mathcal{I}(p)$ (see Equation 2.1, and Equation 2.3), (ii) and the causal faithfulness assumption, which posits that every CI in p corresponds to a d-separation in $\mathcal{G}$, hence $\mathcal{I}(\mathcal{G}) \supseteq \mathcal{I}(p)$. Together, these conditions guarantee that d-separation characterizes all and only the independencies present in p. On the other hand, FCI extends this framework to cases with latent and selection variables. It assumes that the data-generating distribution is faithful to a DAG with variables that remain latent, whose independencies—after marginalization and conditioning—are represented by m-separations in a MAG. This guarantees that m-separation characterizes all and only the independencies present in the data-generating distribution, which is the marginal of a distribution that includes all hidden variables. By reading m-separations from observed CI tests, FCI can therefore consistently recover the underlying causal MAG.

Identifying CIs from finite, noisy data is inherently a statistical task, requiring hypothesis testing, large sample sizes, and careful error control. Nevertheless, to specifically concentrate on inferring causal structure from CI relations, constraint-based methods commonly adopt the assumption of a perfect oracle of CI information [20, 35, 36]. Such an oracle always correctly answers queries about CIs among observed variables. Combined with the previous assumptions, every CI provided by the oracle precisely corresponds to those present in the causal MAG being inferred. Henceforth, we assume that all CI queries are answered by a perfect oracle. However, one can only approximate such an ideal oracle in empirical scenarios.

Building on these foundations, the FCI algorithm [19] explores the space of maximally informative PAGs in three successive stages of adjacency, orientation, and propagation (Algorithm 1). During the *adjacency phase*, the FCI constructs a

provisional skeleton by exhaustively testing CIs among currently adjacent pairs of variables, but only considering conditioning sets that are subsets of their respective adjacent variables. These separating sets (denoted as $Sep[X_j, X_i]$) are those that render the pairs conditionally independent (Algorithm 2) and, when necessary, expand the separating sets with seemingly nonadjacent features contained in **Possible-D-SEP**$(\cdot)$ (first using Algorithm 3 to orient edges and then Algorithm 4 to remove them). In the *orientation phase*, it uncovers v-structures exactly as the PC algorithm does (Algorithm 3). Finally, in the *propagation phase*, it augments the PC propagation step with ten additional deterministic rules from Zhang [20] to orient as many remaining edges as possible, thereby achieving completeness by obtaining a maximally informative PAG. This extension is essential because latent or selection variables violate the local Markov property, so two nodes may become conditionally independent only after conditioning on variables outside their immediate adjacencies. Together, the three phases yield a maximally informative PAG that precisely represents the causal equivalence class compatible with the oracle's independence information. Variants of the FCI, such as RFCI [36], follow the same principles while modifying certain steps and returning their output types. For instance, the RFCI instead of using PAGs, uses RFCI-PAGs. Nevertheless, to keep it short, we will not specify the pseudocode of those versions.

For the initial skeleton (Algorithm 2), we have chosen its stable version from Colombo et al. [12] since it guarantees correctness for the oracle version, and stability for the empirical version. In each algorithm, the edge endmarks have the following meaning: the three types of marks: $>, o$, and $-$, have the meaning specified in Definition 2.3.3, and the metasymbol $*$ is used to represent any of the three previous marks and to denote that the edge mark stays the same after applying a step of the algorithm. For simplicity, we consider that the separating sets are symmetrical in all the algorithms (i.e., $Sep[X_i, X_j] = Sep[X_j, X_i]$).

Algorithm 1 Schematic overview of the FCI algorithm [19]

Input: dataset $\mathcal{D}$ over feature set $\mathcal{X}$
Output: maximally informative PAG $\mathcal{P}$

/* Adjacency phase */
1: Create initial skeleton $\mathcal{P}$ using Algorithm 2
2: Orient v-structures in $\mathcal{P}$ using Algorithm 3
3: Eliminate adjacencies of separable features, update $\mathcal{P}$, and eliminate all directions using Algorithm 4
/* Orientation phase */
4: Orient v-structures in $\mathcal{P}$ using Algorithm 3
/* Propagation phase */
5: Propagate orientation in $\mathcal{P}$ with graphical rules $\mathcal{R}1 - \mathcal{R}10$ from Zhang [20], orienting as many edge marks as possible
6: **return** $\mathcal{P}$

Algorithm 2 Stable algorithm for the initial skeleton [12]

Input: dataset $\mathcal{D}$ over variable set $\mathcal{X}$
Output: undirected skeleton $\mathcal{P}$ and separating sets $\mathrm{Sep}[\cdot,\cdot]$

$\mathcal{P} \leftarrow$ complete undirected graph over $\mathcal{X}$, with all its vertices as $\circ\!\!-\!\!\circ$
$\ell \leftarrow 0$
repeat
  for all features X_i in $\mathcal{X}$ **do**
    $\mathrm{a}(X_i) \leftarrow \mathrm{adj}_{\mathcal{P}}(X_i)$
  end for
  for all pairs (X_i, X_j) **s.t.** $X_j \in \mathrm{a}(X_i)$ **and** $|\mathrm{a}(X_i)\backslash\{X_j\}| \geq \ell$ **do**
    $\mathcal{S}_\ell \leftarrow$ all subsets of $\mathrm{a}(X_i)\backslash\{X_j\}$ of size ℓ
    repeat
      Select a new subset $\mathbf{S} \in S_\ell$
      $S_\ell \leftarrow S_\ell \backslash \{\mathbf{S}\}$
      if X_i and X_j are conditionally independent given $\mathbf{S}$ **then**
        Remove edge $X_i \ast\!\!\ast\ X_j$ from $\mathcal{P}$
        $\mathrm{Sep}[X_i, X_j] \leftarrow \{\mathbf{S}\}$
      end if
    until X_i and X_j are no longer adjacent **or** $\mathcal{S}_\ell = \emptyset$
  end for
  $\ell \leftarrow \ell + 1$
until all pairs of adjacent vertices (X_i, X_j), satisfy $|a(X_i)\backslash\{X_j\}| < \ell$
return $\mathcal{P}, \mathrm{Sep}[\cdot,\cdot]$

Algorithm 3 Orientation of v-structures during the adjacency and orientation phases [7]

Input: undirected skeleton $\mathcal{P}$ and separating sets $\mathrm{Sep}[\cdot,\cdot]$
Output: partially oriented graph $\mathcal{P}$ where every *confirmed* collider is oriented as $X \ast\!\!\rightarrow Z \leftarrow\!\!\ast\ Y$

for all unshielded triples $\langle X, Z, Y\rangle$ in $\mathcal{P}$ **do**
  if $Z \notin \mathrm{Sep}[X, Y]$ **then**
    Orient $X \ast\!\!\ast\ Z \ast\!\!\ast\ Y$ as $X \ast\!\!\rightarrow Z \leftarrow\!\!\ast\ Y$
  end if
end for
return $\mathcal{P}$

Algorithm 4 Pruning edges between features separated by nonadjacent separating sets to obtain the final skeleton during the adjacency phase [19, 43]

Input: partially oriented graph $\mathcal{P} = (\mathcal{X}, \mathcal{E})$, separating sets $\mathrm{Sep}[\cdot, \cdot]$
Output: updated graph $\mathcal{P}$, separating sets $\mathrm{Sep}[\cdot, \cdot]$

for all $X_i \in \mathcal{X}$ **do**
  $\mathrm{PDS}(X_i) \leftarrow \text{Possible-D-SEP}_{\mathcal{P}}(X_i)$
  for all $X_j \in \mathrm{adj}_{\mathcal{P}}(X_i)$ **do**
    $\ell \leftarrow 0$
    repeat
      $\mathcal{S}_\ell \leftarrow$ all subsets of $\mathrm{PDS}(X_i) \backslash \{X_j\}$ of size ℓ
      repeat
        Select a new subset $\mathbf{S} \in S_\ell$
        $S_\ell \leftarrow S_\ell \backslash \{\mathbf{S}\}$
        if X_i and X_j are conditionally independent given $\mathbf{S}$ **then**
          Remove edge $X_i \ast\!\!\ast\ X_j$ from $\mathcal{P}$
          $\mathrm{Sep}[X_i, X_j] \leftarrow S$
        end if
      until X_i and X_j are no longer adjacent **or** $\mathcal{S}_\ell = \emptyset$
      $\ell \leftarrow \ell + 1$
    until X_i and X_j are no longer adjacent **or** $|\mathrm{PDS}(X_i) \backslash \{X_j\}| \leq \ell$
  end for
end for
for all remaining edges $X_i \ast\!\!\ast\ X_j$ **do**
  orient $X_i \ast\!\!\ast\ X_j$ as $X_i \circ\!\!-\!\!\circ X_j$
end for
return $\mathcal{P}$, Sep

Property / Concept	**Brief definition**
Causal faithfulness	Every conditional independence in the distribution p corresponds to an m-separation in the true causal graph.
Causal Markov condition	The distribution p satisfies the causal Markov condition concerning a graph $\mathcal{G}$ if every variable is independent of its non-descendants given its parents.
Causal sufficiency	All common causes of the measured variables are themselves included in the graph.
D-SEP(X_i, X_j)	Set of colliders/ancestors either **D-SEP**(X_i, X_j) or **D-SEP**(X_j, X_i) m-separates X_i and X_j in a MAG.
Local Markov property	Two non-adjacent variables become independent once conditioned on the parents of either of them.
Pairwise Markov property	For any non-adjacent pair, there exists *some* conditioning set that renders them independent by m-separation.
Partial ancestral graph (PAG)	A graphical model with different types of edge marks that represents the Markov equivalence class of MAGs.
Possible-D-SEP(X_i)	Approximation that contains **D-SEP**$(X_i, \cdot)$. Substitute for **D-SEP** when the true MAG is unknown.

Table 4.1: Glossary of the main properties and concepts used in the paper

Chapter 5

Design of the Proposed Algorithm

5.1. The FCI-FS Algorithm

Although the FCI deals with the problem of having only a subset of features, it is not feature-wise incremental (and, to the best of our knowledge, nor are any of the algorithms in the current literature). Thus, each time new features stream in, the causal MAG must be retrained from the beginning. Nevertheless, this is quite ineffective since many of the CI tests, which are time-consuming operations, would have to be repeated. To solve this problem, it is necessary to make an assumption, which we call the *extended causal sufficiency assumption* that states that new incoming features correspond to previously latent features, now becoming observable, and that they respect the acyclicity of the system. That is, new features correspond to causes, effects, or both, which were present in the model but previously unobserved, and now these features can be incorporated without adding any directed or almost directed cycles to the MAG representing our knowledge about the system at the moment. In a distribution faithful to a DAG with latent variables (a MAG), this means that we will have two distributions, one with the new streaming features, p_{new}, and the previous distribution with only the old features, p_{old}, and that p_{old} is the marginal of p_{new}.

As a consequence of this assumption, given a perfect oracle of CI information, the introduction of new features augments the independence model of the original MAG by adding new independencies without eliminating the existing ones. For instance, consider the graph in Figure 5.1a, the simplest CI the oracle can read are: $X_1 \perp_m X_3|X_2$, $X_1 \perp_m X_5|X_2$, $X_1 \perp_m X_4$, $X_2 \perp_m X_4$, $X_3 \perp_m X_5|X_2$. Now, suppose additional features Y_1 and Y_2 arrive through a feature stream that respects the extended causal sufficiency assumption. The inclusion of these new features enables the oracle to detect new CI information involving Y_1 and Y_2, specifically: $X_1 \perp_m Y_1$, $X_1 \perp_m Y_2$, $X_2 \perp_m Y_1$, $X_2 \perp_m Y_2$, $X_3 \perp_m X_4|Y_1$, $X_4 \perp_m X_5|Y_2$, $Y_1 \perp_m Y_2$. Under the assumption of faithfulness, the new distribution will remain faithful to the MAG shown in Figure 5.1b. Notably, despite the

addition of new features, all independencies represented by the original graph in Figure 5.1a persist in the new graph depicted in Figure 5.1b. We call this property the *independence preservation property*.

Proposition 5.1.1 (Independence preservation property)**.** *Let $\mathcal{G}$ be a DAG over the variable set $\mathcal{X} = \mathbf{O}_1 \sqcup \mathbf{X}_{new} \sqcup \mathbf{L}_2 \sqcup \mathbf{S}$, and set $\mathbf{L}_1 = \mathbf{X}_{new} \sqcup \mathbf{L}_2$. Denote by $\mathcal{M}_1$ the MAG over the feature set $\mathbf{O}_1$ obtained from $\mathcal{G}$ by marginalizing over $\mathbf{L}_1$ and conditioning on the selection variables $\mathbf{S}$. If the batch $\mathbf{X}_{new}$ becomes observable, define the enlarged observed set $\mathbf{O}_2 = \mathbf{O}_1 \sqcup \mathbf{X}_{new}$ and let $\mathcal{M}_2$ be the MAG over $\mathbf{O}_2$ obtained by marginalizing the remaining latent variables $\mathbf{L}_2$ and (again) conditioning on $\mathbf{S}$. Then the independence model encoded by $\mathcal{M}_1$ is contained in the one encoded by $\mathcal{M}_2$:*

$$\mathcal{I}(\mathcal{M}_1) \subseteq \mathcal{I}(\mathcal{M}_2).$$

Proof. By construction, $\mathcal{I}(\mathcal{M}_1) = \mathcal{I}(\mathcal{G})[^{\mathbf{S}}_{\mathbf{L}_1}$, $\mathcal{I}(\mathcal{M}_2) = \mathcal{I}(\mathcal{G})[^{\mathbf{S}}_{\mathbf{L}_2}$, . Because $\mathbf{L}_2 \subseteq \mathbf{L}_1$, $\mathcal{M}_2$ marginalizes out the *fewer* variables more than $\mathcal{M}_1$. From the definition in Equation 2.2, marginalizing over a smaller set can only reveal additional conditional independencies and never remove those already present. Hence, every separation encoded by $\mathcal{M}_1$ is also encoded by $\mathcal{M}_2$, proving $\mathcal{I}(\mathcal{M}_1) \subseteq \mathcal{I}(\mathcal{M}_2)$. □

By Proposition 5.1.1, when a new batch of features $\mathbf{X}_{\text{new}}$ arrives, every pair of variables that was nonadjacent in the previous MAG remains nonadjacent in the updated one. Consequently, the skeleton of the existing PAG can serve as the starting point, removing many CI tests—the principal computational bottleneck of constraint-based algorithms—and allowing our incremental variant, FCI-FS, to update the network far more efficiently than rerunning the nonincremental version of the FCI algorithm from scratch. A schematic overview of the FCI-FS algorithm is presented in Algorithm 5. A graphical example of the execution of the FCI-FS can be found in Section 5.2.

The FCI-FS algorithm (Algorithm 5) has the same three phases as the original FCI algorithm (Algorithm 1): an adjacency phase to build the skeleton of the network, an orientation phase to orient v-structures, and a propagation phase to propagate orientations from v-structures. The only changes in the FCI-FS algorithm compared to FCI are confined to the input and the adjacency phase. Algorithm 5 allows incremental input, permitting the structure learned from a previous execution of the algorithm to be used. Suppose we have previously learned the graph shown in Figure 5.1a. After receiving two new features, Y_1 and Y_2, we want to update this graph to account for these additions. Here, $\mathcal{P}_{old}$ denotes the graph from the previous execution, and $Sep_{old}[X, Y]$ denotes, for each pair of nonadjacent features (X, Y) in $\mathcal{P}_{old}$, the separating set responsible for making X and Y conditionally independent. The modifications in the adjacency phase avoid redundant CI tests by adjusting the first step (line 1) of Algorithm 1, which creates the initial skeleton (Algorithm 6), and the third step (line 3), which builds the final skeleton from the initial one (Algorithm 7), using more complex separating sets. These modifications, described in detail below, specifically leverage the previously obtained structures to prevent unnecessary repetition of CI tests.

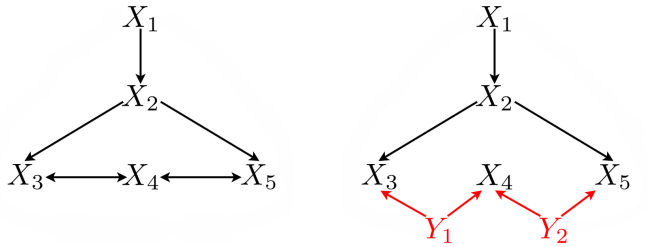


Figure 5.1: In (a), the initial MAG. In (b), the new MAG (which is a DAG) results from receiving Y_1 and Y_2 through a feature stream.

To create the initial skeleton, Algorithm 6 has a similar structure compared to Algorithm 2: starting with a dense graph and checking CI tests over sets of increasing size (the size is noted as ℓ) to remove edges, but it differs from traditional nonincremental algorithms in three remarkable ways. The first difference is that the input has a more sophisticated structure: while Algorithm 2 receives only a dataset, Algorithm 6 also receives the PAG and the separating sets from the previous iteration, both using only the old features, as well as an augmented dataset over the combined space of old and new features. The second difference, implemented in the initialization on lines 1–2, leverages the old graph via Proposition 5.1.1: instead of creating the complete graph over the entire feature space, it keeps the old graph over the old features, creates the complete graph over *only* the new features, then joins each new feature with each old feature, and, instead of initializing an empty separating-set map, starts with the sets from the previous iteration. The final difference appears on line 9 in the choice of possible separating sets: the algorithm skips any candidate set that contains no new variables when the tested pair consists of old variables, because those independencies were already examined in the previous graph and need not be retested; nevertheless, it still guarantees that pairs of old features that were spuriously correlated in the earlier graph will be separated once their latent common causes become observable.

We build the initial skeleton based on the *stable* adjacency phase of Colombo et al. [12] (Algorithm 2) because it retains the correctness of the FCI algorithm under oracle CI information but removes the order dependence of the original procedure; by storing the adjacencies of each feature level-wise (lines 5-7 of Algorithm 6) instead of doing it in the loop of line 8 in Algorithm 6, it yields the same skeleton regardless of how new incoming variables are enumerated. This property is crucial in our incremental framework to reduce the instability of the algorithm, but overall, the FCI-FS is not globally stable.

Once the initial skeleton has been constructed, the algorithm must prune any remaining adjacencies whose endpoints can be separated only by conditioning sets that include variables nonadjacent to both endpoints; these "remote" separating sets are precisely those addressed in the final skeleton in the adjacency phase of FCI-FS. Algorithm 7 follows the same sequence of steps as the FCI, but with a single refinement at line 6: before scheduling a CI test, it checks whether

the candidate variables and separating set have already been examined in an earlier iteration and, if so, skips the redundant test. Although modest, this change prevents the re-evaluation of independencies already certified in prior runs, thereby reducing the total number of CI tests. In practice, this leads to substantial wall-time savings when successive feature batches are small relative to the accumulated feature set, while preserving the soundness and completeness guarantees of the original FCI, as we will see in Section 5.3.

Algorithm 5 Schematic overview of FCI-FS

Input: previous PAG $\mathcal{P}_{old}$, previous separating set $\mathrm{Sep}_{old}[\cdot,\cdot]$, and augmented dataset $\mathcal{D}$ with new features $\mathbf{X}_{new}$

Output: New maximally informative PAG $\mathcal{P}_{new}$ based on $\mathcal{P}_{old}$

/* Adjacency phase */

1: Departing from $\mathcal{P}_{old}$, create initial skeleton $\mathcal{P}_{new}$ using Algorithm 6

2: Orient v-structures in $\mathcal{P}_{new}$ using Algorithm 3

3: Eliminate adjacencies of separable features, update $\mathcal{P}_{new}$, and eliminate all directions using Algorithm 7

/* Orientation phase */

4: Orient v-structures in $\mathcal{P}_{new}$ using Algorithm 3

/* Propagation phase */

5: Propagate orientation in $\mathcal{P}_{new}$ with graphical rules $\mathcal{R}1 - \mathcal{R}10$ from Zhang [20], orienting as many edge marks as possible.

6: **return** $\mathcal{P}_{new}$

Algorithm 6 New FCI-FS for the initial skeleton with feature streams

Input: previous PAG $\mathcal{P}_{old}$, previous separating set $\mathrm{Sep}_{old}[\cdot,\cdot]$, and augmented dataset $\mathcal{D}$ with new features $\mathbf{X}_{new}$

Output: new undirected skeleton $\mathcal{P}_{new}$ and separating sets $\mathrm{Sep}_{new}[\cdot,\cdot]$

$\mathcal{P}_{new} \leftarrow$ create the complete undirected graph over $\mathbf{X}_{new}$, join this graph with $\mathcal{P}_{old}$ by adding an edge between each variable in $\mathcal{P}_{old}$ and each feature in $\mathbf{X}_{new}$. Direct all the edges of the resulting graph as $\circ\!\!-\!\!\circ$
$\mathrm{Sep}_{new}[\cdot,\cdot] \leftarrow \mathrm{Sep}_{old}[\cdot,\cdot]$
$\ell \leftarrow 0$
repeat
  for all vertices X_i in $\mathcal{P}_{new}$ **do**
    $\mathrm{a}(X_i) \leftarrow \mathrm{adj}_{\mathcal{P}_{new}}(X_i)$
  end for
  for all pairs (X_i, X_j) **s.t.** $X_j \in \mathrm{a}(X_i)$ **and** $|\mathrm{a}(X_i)\backslash\{X_j\}| \geq \ell$ **do**
    $\mathcal{S}_\ell \leftarrow$ all subsets $\mathbf{S} \subseteq \mathrm{a}(X_i)\backslash\{X_j\}$ of size ℓ **s.t.** $(\{X_i, X_j\} \sqcup \mathbf{S}) \cap \mathbf{X}_{new} \neq \emptyset$
    repeat
      Select a new subset $\mathbf{S} \in S_\ell$
      $S_\ell \leftarrow S_\ell \backslash \{\mathbf{S}\}$
      if X_i and X_j are conditionally independent given $\mathbf{S}$ **then**
        Remove edge $X_i \ast\!\!\ast X_j$ from $\mathcal{P}_{new}$
        $\mathrm{Sep}_{new}[X_i, X_j] \leftarrow \{\mathbf{S}\}$
      end if
    until X_i and X_j are no longer adjacent **or** $\mathcal{S}_\ell = \emptyset$
  end for
  $\ell \leftarrow \ell + 1$
until all pairs of adjacent vertices (X_i, X_j), satisfy $|a(X_i)\backslash\{X_j\}| < \ell$
return $\mathcal{P}_{new}, \mathrm{Sep}_{new}[\cdot,\cdot]$

Algorithm 7 FCI-FS for pruning nonadjacent separating sets to obtain final skeleton

Input: partially oriented graph $\mathcal{P}_{new} = (\mathcal{X}, \mathcal{E})$, separating sets $\mathrm{Sep}_{new}[\cdot,\cdot]$, incoming variables $\mathbf{X}_{new}$
Output: updated graph $\mathcal{P}_{new}$, separating sets $\mathrm{Sep}_{new}[\cdot,\cdot]$

for all vertices X_i in $\mathcal{P}_{new}$ **do**
  $\mathrm{PDS}(X_i) \leftarrow$ **Possible-D-SEP**(X_i)
  for all $X_j \in \mathrm{adj}_{\mathcal{P}_{\mathrm{new}}}(X_i)$ **do**
    $\ell \leftarrow 0$
    repeat
      $\mathcal{S}_\ell \leftarrow$ all subsets $\mathbf{S} \subseteq \mathrm{PDS}(X_i)\backslash\{X_j\}$ of size ℓ **s.t.** $(\{X_i, X_j\} \sqcup \mathbf{S}) \cap \mathbf{X}_{new} \neq \emptyset$
      repeat
        Select a new subset $\mathbf{S} \in S_\ell$
        $S_\ell \leftarrow S_\ell \backslash \{\mathbf{S}\}$
        if X_i and X_j are conditionally independent given $\mathbf{S}$ **then**
          Remove edge $X_i \ *\!\!-\!\!* \ X_j$ from $\mathcal{P}$
          $\mathrm{Sep}_{new}[X_i, X_j] \leftarrow S$
        end if
      until X_i and X_j are no longer adjacent **or** $\mathcal{S}_\ell = \emptyset$
      $\ell \leftarrow \ell + 1$
    until X_i and X_j are no longer adjacent **or** $|\mathrm{PDS}(X_i) \backslash \{X_j\}| \leq \ell$
  end for
end for
for all remaining edges $X_i \ *\!\!-\!\!* \ X_j$ **do**
  orient $X_i \ *\!\!-\!\!* \ X_j$ as $X_i \circ\!\!-\!\!\circ X_j$
end for
return $\mathcal{P}_{new}$, $\mathrm{Sep}_{new}[\cdot,\cdot]$

5.2. Example Execution of the FCI-FS Algorithm

Consider an illustrative execution of the FCI-FS algorithm. Suppose we initially have learned the PAG depicted in Figure 5.2a by employing FCI-FS, and now a new streaming feature, Y, arrives under the extended causal sufficiency assumption. We assume an oracle provides the CI statements relevant to Y: specifically, $Y \perp_m X_2 \mid X_3$ and $Y \perp_m X_4 \mid X_2, X_3, X_4$. FCI-FS will accept three inputs: (1) The previous PAG, $\mathcal{P}_{old}$, as shown in Figure 5.2a. (2) The set of previously identified separating sets, Sep_{old}, for nonadjacent pairs: specifically, (X_2, X_4) with $\text{Sep}_{old}[X_2, X_4] = X_3$, and (X_3, X_5) with $\text{Sep}_{old}[X_3, X_5] = X_2$. (3) The augmented dataset $\mathcal{D}$ incorporates the new feature Y.

Upon execution (Algorithm 5, line 1), Algorithm 6 takes these inputs ($\mathcal{P}_{old}$, Sep_{old}, and $\mathcal{D}$). After completing line 1, $\mathcal{P}_{new}$ will match Figure 5.2b. Initially, the algorithm checks separating sets of size $\ell = 0$ without removing any edges, subsequently incrementing ℓ to 1. With $\ell = 1$, since X_1 and X_3 are conditionally independent given X_4, the edge between X_1 and X_3 is removed, and X_4 is recorded as $\text{Sep}_{new}[X_1, X_3]$. After this step, since all vertices have degree three or less and no additional separating sets can be identified, Algorithm 6 terminates, outputting $\mathcal{P}_{new}$ (Figure 5.2c) and Sep_{new}.

Next, Algorithm 5 proceeds to line 2, invoking Algorithm 3 with inputs $\mathcal{P}_{new}$ and Sep_{new}, which produces the updated PAG shown in Figure 5.2d by identifying and orienting new v-structures. Subsequently, Algorithm 5 advances to line 3, triggering Algorithm 7. With input $\mathcal{P}_{new}$, Sep_{new}, and $\mathcal{D}$, this algorithm examines conditional independencies for subsets with nonadjacent features. At $\ell = 3$, it discovers that Y is independent of X_4 given X_1, X_2, X_3 and removes their adjacency. This operation updates $\mathcal{P}_{new}$ as shown in Figure 5.2e, with the separating set $\text{Sep}_{new}[Y, X_4] = \{X_1, X_2, X_3\}$.

Finally, Algorithm 5 again calls Algorithm 3 (line 4), which identifies and orients the remaining v-structures, leading to the final PAG depicted in Figure 5.2f. To complete the process, propagation rules $\mathcal{R}1 - \mathcal{R}10$ from Zhang [20] are applied (line 5), maximizing the orientation of edge marks in $\mathcal{P}_{new}$, producing the final output, graphically represented in Figure 5.2g.

5.3. Correctness of the FCI-FS Algorithm

To the best of our knowledge, existing causal discovery algorithms for streaming features do not provide formal guarantees of soundness or completeness in general, and may introduce incorrect causal relations even under perfect CI information. In contrast, our proposed FCI-FS algorithm provides both (see Theorem 5.3.1), and FCI-FS is therefore the first incremental FCI-style procedure whose output is provably equivalent (Corollary 5.3.3) to what would be obtained by rerunning FCI after each feature arrival. To proceed to the formal proof of Theorem 5.3.1, first, we collect some auxiliary results.

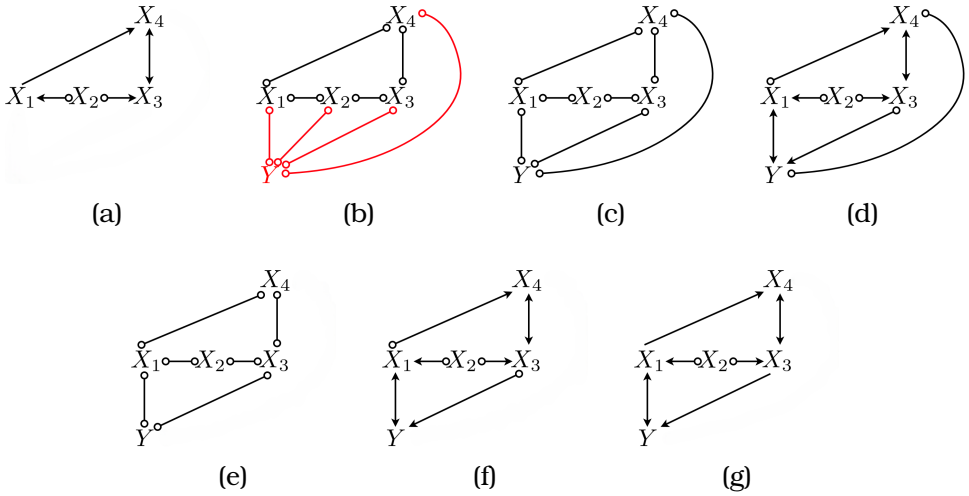


Figure 5.2: Illustrative example of the step-by-step PAG updates performed by the FCI-FS algorithm upon the arrival of a new streaming feature Y.

Proposition 5.3.1 (Spirtes [35, Theorem 6.2])**.** *Let $\mathcal{M}$ be a MAG over the observed variables $\mathbf{O}$. If X_i and X_j are non-adjacent in $\mathcal{M}$ and $X_i \notin An_{\mathcal{M}}(X_j)$, then X_i and X_j are m-separated by **D-SEP**(X_i, X_j).*

Because a MAG contains no directed cycles, Proposition 5.3.1 immediately yields the following corollary.

Corollary 5.3.1. *In a MAG $\mathcal{M}$ over the observed variables $\mathbf{O}$, any non-adjacent pair (X_i, X_j) is m-separated either by **D-SEP**(X_i, X_j) or by **D-SEP**(X_j, X_i).*

The key property that FCI-FS exploits is that all vertices needed for a valid separating set are examined. Since FCI-FS enumerates every subset of **Possible-D-SEP**(X_i) for each X_i (see lines 1-2 for Algorithm 7), we have the following result.

Proposition 5.3.2. *Let the distribution of $\mathcal{X}$ be faithful to a DAG $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ and assume $\mathbf{L}_1 \subseteq \mathcal{X}$ is initially latent. Suppose a subset $\mathbf{X}_{new} \subseteq \mathbf{L}_1$ becomes observable, and we have a perfect oracle for conditional independencies. Let $\mathcal{P}_{\text{new}}$ denote the PAG obtained after step 2 of Algorithm 5. If X_i and X_j are observable with $X_i \notin An_{\mathcal{P}_{\text{new}}}(X_j)$, then every vertex in **D-SEP**(X_i, X_j) belongs to **Possible-D-SEP**(X_i) in $\mathcal{P}_{\text{new}}$.*

Proof. The argument is identical to that of Spirtes et al. [9], in their Lemma 6.4.1 (p. 417), and is omitted for brevity. □

As a consequence, we have the following corollary.

Corollary 5.3.2. *Let the distribution of $\mathcal{X}$ be faithful to a DAG $\mathcal{G} = (\mathcal{X}, \mathcal{E})$ and assume $\mathbf{L}_1 \subseteq \mathcal{X}$ is initially latent. Suppose a subset $\mathbf{X}_{new} \subseteq \mathbf{L}_1$ becomes observable, and we have a perfect oracle for conditional independencies. Let $\mathcal{P}_{\text{new}}$ denote the PAG obtained after step 2 of Algorithm 5. If X_i and X_j are observable and m-separable in $\mathcal{G}_{[\mathbf{L}_1 \setminus \mathbf{X}_{new}}$ then there exists a separating set $\mathbf{S}$ that renders X_i and*

X_j *independent* ***s.t.*** *either* $\mathbf{S} \subset$ ***Possible-D-SEP***(X_i) *or* $\mathbf{S} \subset$ ***Possible-D-SEP***(X_j) *in* $\mathcal{P}_{new}$.

Proof. Assume, for contradiction, that no separating set contained in **Possible-D-SEP**(X_i) or **Possible-D-SEP**(X_j) renders X_i and X_j independent. By Proposition 5.3.2, the sets **D-SEP**(X_i, X_j) and **D-SEP**(X_j, X_i) are, respectively, subsets of **Possible-D-SEP**(X_i) and **Possible-D-SEP**(X_j). Our assumption therefore implies that neither **D-SEP**(X_i, X_j) nor **D-SEP**(X_j, X_i) separates X_i and X_j. Corollary 5.3.1, together with faithfulness, then forces X_i and X_j to be adjacent in the latent projection $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$, contradicting the premise that they are m-separable there. Hence a separating set $\mathbf{S}$ satisfying the stated condition must exist, completing the proof. □

To establish the correctness of the algorithm given its incremental nature, it is necessary to explicitly reference the PAG and the separating set map that FCI-FS receives at each iteration. In practice, the algorithm always receives as input the output from the previous batch of learned features. However, for clarity in constructing an inductive argument in the proof of Theorem 5.3.1, we assume that the input PAG provided to the algorithm is correct and that the accompanying separating set map contains, for each pair of m-separable features in the latent model, at least one valid separating set; concretely, this set would be the one that rendered the pair independent in the previous execution of the FCI-FS procedure whose output is the input in the new iteration.

Proposition 5.3.3. *Let the distribution of* $\mathcal{X}$ *be faithful to a DAG* $\mathcal{G} = (\mathcal{X}, \mathcal{E})$*, and assume that the subset* $\mathbf{L}_1 \subseteq \mathcal{X}$ *is initially latent. Suppose that a subset of these latent variables,* $\mathbf{X}_{new} \subseteq \mathbf{L}_1$*, becomes observable and that a perfect oracle of CI information is available. Then, if the* FCI-FS *algorithm is initialized with a PAG representing* $\mathcal{G}[_{\mathbf{L}_1}$ *along with its corresponding separating set map, the output of the algorithm is a PAG representing* $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$.

Proof. Since the orientation and propagation phases remain unchanged from those in Spirtes et al. [9] and Zhang [20], it suffices to prove the proposition only for the adjacency phase of the FCI-FS algorithm, following an argument analogous to the proofs of Theorems 5.1 (p. 407) and 6.4 (p. 417) in Spirtes et al. [9].

Let $\mathcal{P}_{old}$ and $Sep_{old}[\cdot, \cdot]$ denote the input PAG and its separating set map, respectively, and let $\mathcal{P}_{new}$ represent the output of the algorithm. First, it must be established that if variables X and Y are nonadjacent in $\mathcal{P}_{new}$, they are also nonadjacent in $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$. In Algorithm 6, before any pruning occurs, any pair of vertices is nonadjacent if and only if they were nonadjacent in $\mathcal{P}_{old}$. The independence preservation property (Proposition 5.1.1) guarantees $\mathcal{I}(\mathcal{P}_{old}) \subseteq \mathcal{I}(\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}})$, implying that pairs nonadjacent in $\mathcal{P}_{old}$ remain nonadjacent in $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$. Furthermore, Algorithms 6 and 7 eliminate adjacencies only if the corresponding pair of features is m-separable in $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$. Therefore, any features nonadjacent in $\mathcal{P}_{new}$ must also be nonadjacent in $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$.

Conversely, assume features X and Y are adjacent in $\mathcal{P}_{new}$. Thus, X and Y are not m-separable in $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$ given either **Possible-D-SEP**(X) or **Possible-D-SEP**(Y). Therefore, by Corollary 5.3.2, X and Y cannot be m-separated in $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$ at all, and since $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$ is a MAG, X and Y must be adjacent in $\mathcal{G}[_{\mathbf{L}_1 \setminus \mathbf{X}_{new}}$. □

We then have the main result.

Theorem 5.3.1. *Let the distribution of $\mathcal{X}$ be faithful to a DAG $\mathcal{G} = (\mathcal{X}, \mathcal{E})$, where the feature set $\mathcal{X}$ is partitioned into k batches as $\mathcal{X} = \mathbf{L}_1 \sqcup \mathbf{L}_2 \sqcup \cdots \sqcup \mathbf{L}_k$. Suppose these batches arrive incrementally, and at each batch i, the corresponding set of features $\mathbf{L}_i$ becomes observable. Further, assume that a perfect oracle of CI information is available. Then, if the proposed* FCI-FS *algorithm is executed sequentially each time a new batch of features arrives, its final output after processing all batches is a maximally informative PAG that correctly represents the causal structure encoded by $\mathcal{G}$.*

Proof. The proof is done trivially by induction, using Proposition 5.3.3 and considering that when the first batch arrives, the algorithm receives the empty graph with an empty separation set map and behaves merely as the FCI, so its output is the PAG representing $\mathcal{G}[_{\mathcal{X} \setminus \mathbf{L}_1}$. □

Summing up everything, it is possible to obtain the following corollary.

Corollary 5.3.3. *Let the distribution of $\mathcal{X}$ be faithful to a DAG $\mathcal{G} = (\mathcal{X}, \mathcal{E})$, where the feature set $\mathcal{X}$ is partitioned into k batches as $\mathcal{X} = \mathbf{L}_1 \sqcup \mathbf{L}_2 \sqcup \cdots \sqcup \mathbf{L}_k$. Suppose these batches arrive incrementally, and at each batch i, the corresponding set of features $\mathbf{L}_i$ becomes observable. Further, assume that a perfect oracle of CI information is available. Then, if the proposed* FCI-FS *algorithm is executed sequentially each time a new batch of features arrives, its output after the i-th batch has been processed, is a maximally informative PAG that correctly represents the causal structure encoded by $\mathcal{G}[_{\mathcal{L}}$, where $\mathcal{L} = \mathcal{X} \setminus (\mathbf{L}_1 \sqcup \cdots \sqcup \mathbf{L}_i) = \mathbf{L}_{i+1} \sqcup \cdots \sqcup \mathbf{L}_k$.*

Chapter 6

Experiments

All the simulations have been performed with the FCI-FS Python repository developed for this work[1]. So far, all our theoretical development has focused on improving the FCI algorithm, but to keep the initial tests manageable, the code in the repository is based on the PC algorithm, not the FCI. There are two reasons for this: the first one is that PAGs require an implementation of graphs where edges can have four different types of marks; this means ten different types of edges, which would increase the complexity of the program. On the other hand, PDAGs only require three types of edges. The second is that, while both FCI and PC are very similar, the PC has simpler semantics, making it better to perform initial tests to show the performance of the proposed modifications in the adjacency phase. However, it is worth noting that since the results seem promising, the package will be extended in the future. In the current code, there are two main scripts, both located in the FCI-FS/main folder. The first consists of the script *PDAG.py*, which was developed to represent the equivalence class of a DAG. The second corresponds to the script *PCFS.py*, which contains the new proposed incremental constraint-based algorithm.

6.1. Synthetic Datasets

The simulation code is in the FCI-FS/main folder, in the *simulation1.py* script, which depends on pgmpy for the generation of random BNs.The simulations have been performed on 256 random BNs of discrete variables, generating 500 samples for each network. Eight different orders of variables were used to learn each network; therefore, for each significance level value, 256 PAGs were learned with the PC-Stable, one for each random graph, without taking into consideration the order, 2048 were learned with the original PC (one per graph and order), and 2048 were learned with the PC-FS (one per graph and order) with two different batches, and all results have been averaged. To test the proposed algorithm, a percentage of the features is randomly extracted, and then the proposed algorithm learns the DAG from the data in two different batches. First,

[1]Available at: https://github.com/Jose-Maria-De-Miguel/FCI-FS.git.

without the extracted variables; later, the variables are added to the network with their incremental version for the feature streams. Given a specific order, all of the sample data have also been used to learn a DAG with the original PC, the PC-Stable. Furthermore, for the CI tests, the chi-square has been used. Two hyper-parameters have been taken into consideration to learn the networks: one of them is the p-value (or pruning parameter) that is used for the CI tests, and the second one is the percentage of extracted variables (we will call this x). The corresponding algorithm will indicate the percentage of features that have been extracted to be learned in the second batch as PC-FSx. For example, PC-FS25 means that 25% of the features were extracted and that PC-FS learned the graph in two batches, first with the 75% of variables that were not extracted, and then the same algorithm was executed again to learn the graph also with the remaining 25% of extracted variables. All the testing has been done with the Tau T2D machine (AMD EPYC Milan) with 8 vCPUs and 32 GiB of RAM.

As far as the performance metrics to evaluate the algorithm are concerned, let TP, FP, TN, and FN denote, respectively, the number of true positive edges (correctly present), false positive edges (spuriously present), true negative edges (correctly absent), and false negative edges (spuriously absent) in the graph learned by the algorithm. From these counts we obtain **precision** P and **recall** R

$$\mathrm{P} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \qquad \mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$$

Precision tells us what fraction of the edges predicted by the algorithm are actually real. Recall tells us what fraction of the real edges the algorithm manages to recover. Their harmonic mean is the the **F1 score**

$$\mathrm{F1} = 2\frac{\mathrm{P}\cdot\mathrm{R}}{\mathrm{P}+\mathrm{R}}$$

Because P and R capture complementary aspects of performance, the $\mathrm{F1}$ score is often preferred as a single summary statistic. A frequently reported alternative (or additional) measure is the **structural Hamming distance** (SHD), defined as the minimum number of arc additions, deletions, and arc orientation changes (additions, reversals or removals) required to transform the learned PDAG into the true PDAG [44]. From its definition, it is possible to obtain the SHD from the FP and FN [45]:

$$\mathrm{SHD} = \mathrm{FP}+\mathrm{FN}$$

For precision, recall, and F1, the metrics range from 0 (worst) to 1 (best), whereas the SHD is an integer metric for which smaller values denote better performance. Figures 6.1 and 6.2 present the mean results over all program runs with 95% confidence intervals. Because every PC-FSx variant aligns closely with PC-Stable, we report the PC-Stable and treat the performance of any PC-FSx as effectively equivalent.

Figure 6.1c reveals that, somewhat counter-intuitively, the PC algorithm achieves a higher $\mathrm{F1}$ score. Although [12] evaluated PC-Stable against the original PC

solely on precision and recall, our analysis also reports the F1 score and shows that the original PC attains a higher value. The reason stems from PC-Stable's order-independent, "level-wise" procedure: once an edge is removed, each node's adjacency set is frozen until the algorithm advances to larger conditioning set sizes, permitting CI tests on richer separating sets. This cautious approach eliminates more false edges and thus boosts precision (Figure 6.1a), but it simultaneously discards a significant number of true edges, sharply reducing recall (Figure 6.1b). Because the F1 score is the harmonic mean of precision and recall, the recall loss outweighs the precision gain, so the original PC, which updates adjacency sets immediately and therefore retains more genuine edges, emerges with the better overall F1.

The SHD metric (Figure 6.2c) behaves as anticipated: with higher p-values (i.e., looser pruning) both algorithms, PC-Stable (and all the PC-FSx), and PC produce similar structures, but as the pruning threshold becomes more stringent, PC-Stable and all the PC-FSx again tend to eliminate more edges than the original PC. Examining the FP and FN in Figures 6.2a and 6.2b respectively clarifies this pattern. PC-Stable and the PC-FSx are notably effective at avoiding FPs, and tightening the pruning parameter alters their FP count only marginally. In contrast, the original PC begins with a considerable number of FPs, yet this number declines steeply as the pruning becomes stricter, eventually approximating the performance of PC-Stable and all the PC-FSx. On the other hand, PC-Stable incurs more FNs because its expanded battery of CI tests uncovers additional independencies; as the CI significance level is lowered, this FN rate worsens further. However, the Stable versions can always get a better SHD than the original ones with higher p-values.

Regarding the complexity, it is important to note that this initial implementation does not yet incorporate every improvement we proposed. In particular, line 9 of Algorithm 6 is still absent from the code, so certain CI tests that have already been executed are repeated unnecessarily. Even so, substantial computational gains remain. Figure 6.3a plots the number of CI tests performed by PC-FSx when the second batch arrives and compares it with the number required to rerun PC or PC-Stable a second time with the new features. Likewise, Figure 6.3b shows the time taken to process the second batch with each incremental algorithm versus the time a complete rerun of a traditional method would demand. Notice how the lower pruning parameter turns into doing fewer CI tests, thus taking less time since the adjacencies get removed faster.

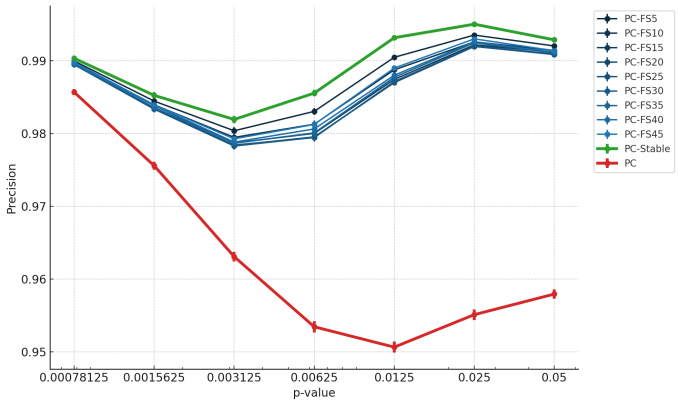
These results show that this preliminary implementation for feature streams delivers significant speed-ups while retaining performance almost indistinguishable from that of PC-Stable, decidedly promising results.

6.2. Real-World Dataset

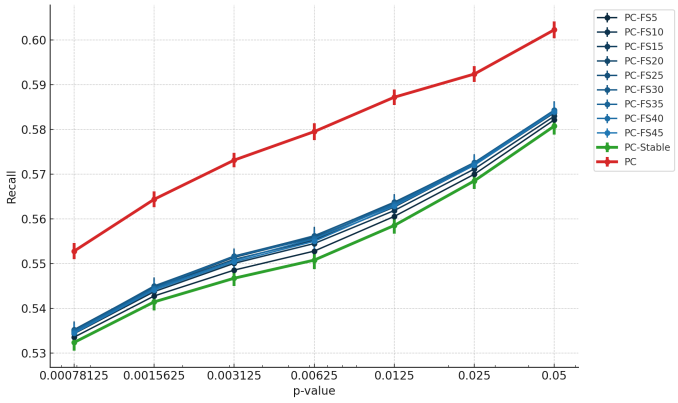
ApisTox [38] is an excellent choice for learning the causal structure of pesticide toxicity data because it is modern and compact. The set contains 1035 pesticide

molecules, each described by 13 categorical or integer-coded variables — no missing values — along with a toxic/nontoxic label (plus an optional three-level severity code). Attributes include chemical identifiers (SMILES, CAS, PubChem CID), year of first report, pesticide class flags (herbicide/fungicide/insecticide), and curated source information drawn from the ECOTOX and PPDB databases. This data set is also well suited to feature-streaming scenarios: one can reveal variables in successive batches, mimicking real scenarios where new descriptors arrive over time. This makes ApisTox an ideal real-world dataset to test for incremental causal-structure learners such as PC-FS.

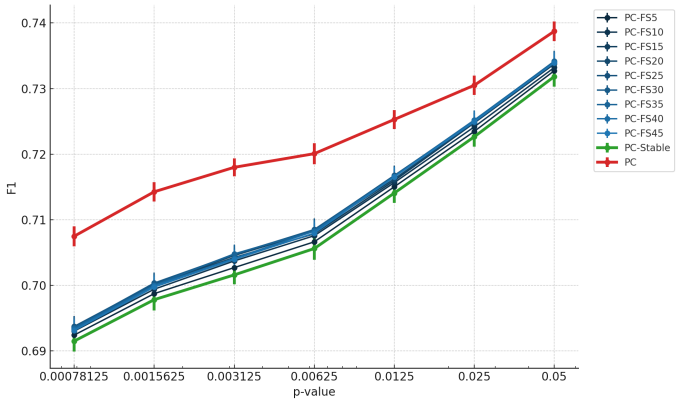
We set the pruning threshold to 0.025 because every PC-FSx variant attains its highest precision and fewest FP while keeping reasonable $\mathrm{F1}$ and SHD; the *time* attribute was removed to focus strictly on atemporal relationships. Structure learning therefore unfolded in three streaming batches: first, the variables *name*, *CID*, *label*, and *ppdbLevel* (Figure 6.4a); second, the arrival of *herbicide*, *insecticide*, *SMILES*, and *toxicityType* produced Figure 6.4b; and finally, adding *CAS*, *source*, *fungicide*, and *otherAgrochemical* yielded Figure 6.4c—a PDAG identical to the graph obtained when we trained the PC-Stable algorithm ourselves on the complete feature set at once, confirming the theoretical results obtained with the synthetic dataset that the incremental procedure preserves most of the ultimate network structure.



(a) Precision score against the p-value.

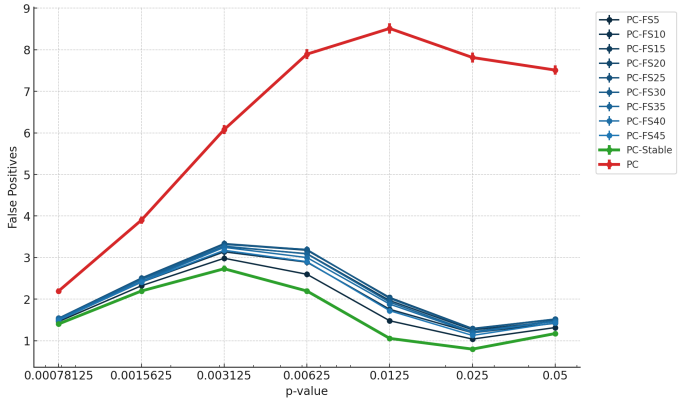


(b) Recall score against the p-value.

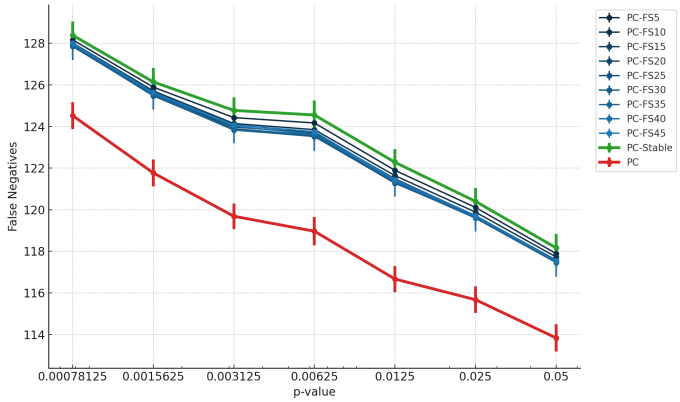


(c) F1 score against the p-value.

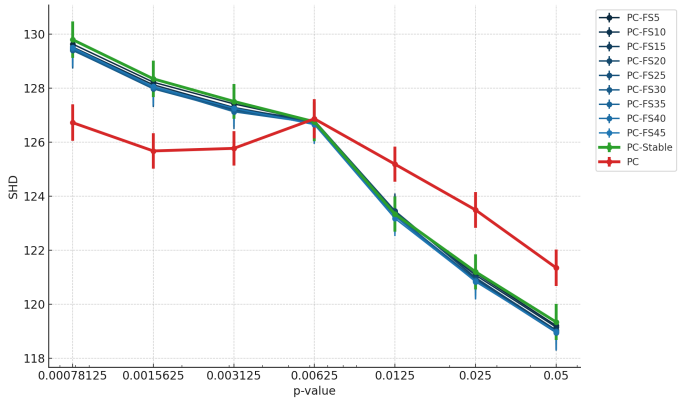
Figure 6.1: Precision (a), recall (b), and F1 (c) scores plotted against p-values. The 95% confidence intervals are represented by vertical error bars in each panel.



(a) FP against p-value.

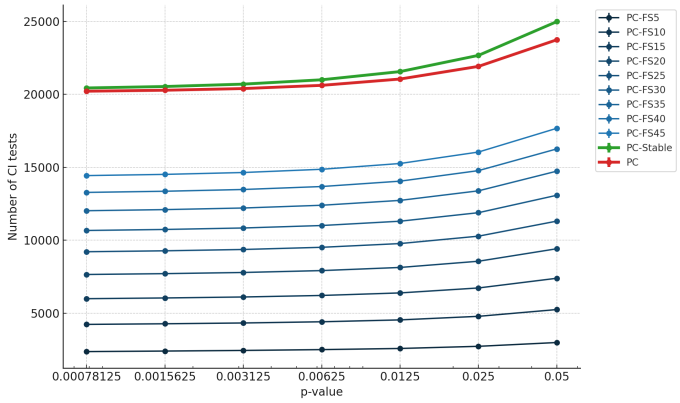


(b) FN against p-value.

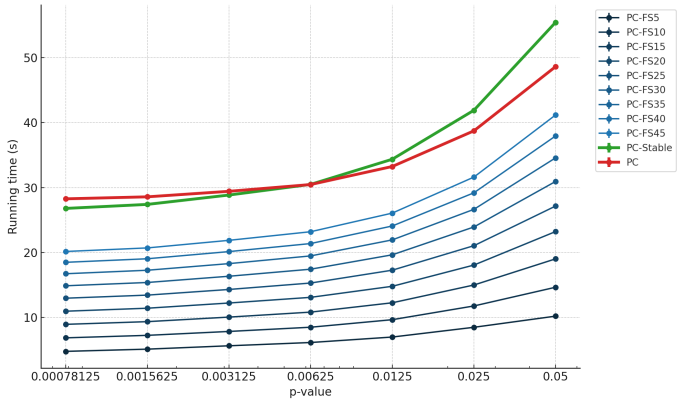


(c) SHD against p-value.

Figure 6.2: FP (a), FN (b), and SHD (c) scores plotted against p-values. The 95% confidence intervals are represented by vertical error bars in each panel.

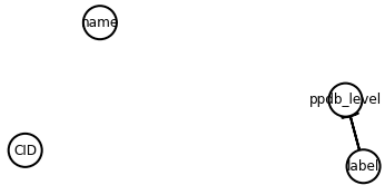


(a) Number of CI tests of each algorithm, given different pruning parameters.

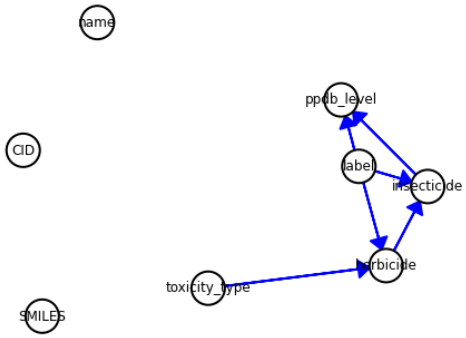


(b) Running time of each algorithm, given different pruning parameters.

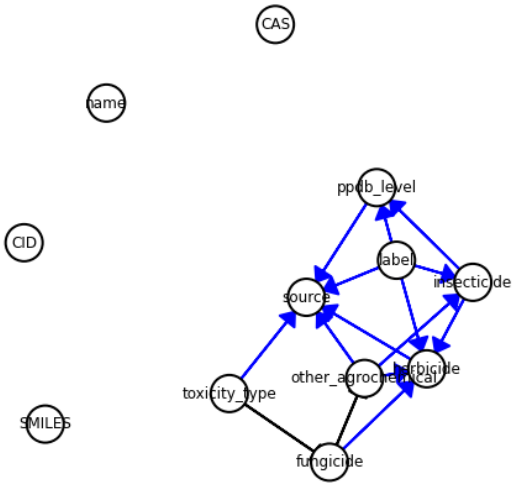
Figure 6.3: Number of CI tests (a) and running time (b) for each algorithm, plotted as a function of the pruning parameter. Each panel shows how varying the pruning parameter affects computational complexity for the different algorithms evaluated.



(a) Causal structure learned with the PC-Stable with the features *CID, label, name* and *ppdb level*



(b) Causal structure learned with PC-FS on top of the network in Figure 6.4a after the arrival of features *herbicide*, *insecticide*, *SMILES*, and *toxicityType*.



(c) Causal structure learned with PC-FS on top of the network in Figure 6.4b after the arrival of features name, *CAS*, *source*, *fungicide*, and *otherAgrochemical*.

Figure 6.4: Graphs learned from the real-world dataset in [38], were new features arrive incrementally.

Chapter 7

Impact Analysis

The development of the FCI-FS algorithm for incremental causal discovery with feature streams has broad implications across various domains. Causal discovery methods in general provide insights beyond what correlation-based analyses can offer, finding true cause-effect relationships that underpin complex systems. By extending the Fast Causal Inference approach to handle feature streams (i.e., scenarios where new variables arrive over time), FCI-FS opens up these capabilities to dynamic, evolving datasets. In this chapter, we assess the potential impact of the results from an environmental perspective and how FCI-FS aligns with the *United Nations 2030 Agenda and its Sustainable Development Goals* (SDGs). We highlight expected benefits in each context as well as possible adverse effects. Furthermore, we discuss how certain development decisions were guided by impact considerations.

7.1. Environmental Impact

SDG 13 – Climate Action: Take urgent action to combat climate change and its impacts. The FCI-FS algorithm carries implications for the environment, both in how it can be applied to environmental challenges and how it affects the sustainability of computing practices. Expected environmental benefits and contributions to improving energy efficiency in computing. A core advantage of FCI-FS is its ability to reuse previous computation (earlier CI test results and partially learned graphs) when new features arrive, rather than recomputing a model from scratch. This design leads to a reduction in the number of costly CI tests and overall runtime. In practical terms, that means significantly less CPU/GPU usage for continuous learning tasks. Given that data centers and IT infrastructure account for a share of global electricity usage on the order of 1%-2% of global electricity consumption [46], any improvement in algorithmic efficiency contributes to environmental sustainability. By lowering computational energy requirements, FCI-FS helps reduce the carbon footprint associated with large-scale data analysis. Especially as big data and streaming analytics grow, using more energy-efficient algorithms will be key in curbing the IT sector's en-

ergy demand. FCI-FS illustrates how algorithmic innovation can support green computing: it achieves the same analytical goals with fewer resources, aligning with efforts to make technology more eco-friendly. On the other hand, FCI-FS can be a tool for climate scientists and environmental researchers to discover causal drivers in climate systems – for example, identifying which climate indicators cause increased frequency of extreme events. By yielding more robust insights that hold under future scenarios [47], it can inform climate adaptation strategies and policies.

SDG 7 – Affordable and Clean Energy: Ensure access to affordable, reliable, sustainable, and modern energy for all. While FCI-FS is a digital technology, its efficiency touches on the targets of SDG 7, which include improving energy efficiency and expanding access to sustainable energy. By cutting down on unnecessary computations, the algorithm exemplifies energy efficiency in IT. This not only reduces costs (making computing more affordable) but also lessens the load on power systems. In large-scale deployments, using energy-efficient algorithms contributes to sustainable digital infrastructure, which is increasingly part of modern energy considerations. Furthermore, FCI-FS could be applied in energy systems management – for instance, incremental causal discovery on streaming data from smart grids could identify causes of energy loss or failure conditions, leading to a more reliable energy supply.

7.2. Possible Adverse Effects

The main possible adverse effect is the **global and cultural bias in data:** If FCI-FS is primarily developed and tested on data from certain cultures or contexts, its usage globally might face challenges. Cultures differ in how data is generated and what causal relationships exist (for example, causal factors in one country's public health might differ from another's due to cultural behaviors). Culturally, some may distrust the tool as not accounting for local nuances. To mitigate this, it will be important to apply and validate FCI-FS in diverse contexts and possibly incorporate cultural domain knowledge into the analysis (for instance, including culturally relevant features in the data streams).

7.3. Development Decisions Influenced by Impact Considerations

Emphasis on Computational Efficiency: The primary design goal was to reduce redundant computations (especially expensive CI tests) when updating the causal model with new features. This decision was driven not only by performance needs but also by an understanding of its impact on resource usage. By making the algorithm more efficient, we aimed to minimize energy consumption and hardware strain for continuous incremental tasks. The decision to reuse prior information directly contributes to lower computational load, aligning with greener AI practices.

Chapter 8

Conclusion

This Bachelor's thesis addressed the problem of constraint-based causal discovery in scenarios with incremental streaming features, providing the first incremental algorithm, FCI-FS, that saves computational time by skipping redundant CI tests while maintaining soundness and completeness. The thesis highlighted the importance of latent-variable models in incremental settings and provided a formal proof of the correctness of the proposed algorithm. Furthermore, an initial version, the PC-FS algorithm, was implemented in Python and evaluated using synthetic datasets, where its performance was compared against standard (non-incremental) PC and PC-Stable algorithms; experiments with real-world datasets were also conducted. Overall, the results demonstrated that FCI-FS is currently the only reliable, theoretically justified, and computationally efficient incremental causal discovery algorithm for streaming features.

For future work, the Python package can be updated to include the FCI-FS version and not just the PC-FS. Furthermore, an application we are currently studying is leveraging the independence preservation property (Proposition 5.1.1) to parallelize the FCI algorithm while guaranteeing correctness.

Bibliography

[1] J. Pearl, *Causality*. Cambridge University Press, 2009.

[2] H. Reichenbach, *The Direction of Time*, M. Reichenbach, Ed. Dover Publications, 1956.

[3] C. Villa-Blanco, C. Bielza, and P. Larrañaga, "Feature subset selection for data and feature streams: A review," *Artificial Intelligence Review*, vol. 56, no. Suppl 1, pp. 1011–1062, 2023.

[4] J. Li, X. Hu, J. Tang, and H. Liu, "Unsupervised streaming feature selection in social media," in *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*. Association for Computing Machinery, 2015, pp. 1041–1050.

[5] R. AlTalhi, M. N. Saqib, U. Saeed, and A. Alghamdi, "Malicious URL detection using streaming feature selection," in *Proceedings of the 5th International Conference on Future Networks and Distributed Systems*. Association for Computing Machinery, 2021, pp. 100–104.

[6] D. Wang, W. Ding, K. Yu, X. Wu, P. Chen, D. L. Small, and S. Islam, "Towards long-lead forecasting of extreme flood events: A data mining framework for precipitation cluster precursors identification," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2013, pp. 1285–1293.

[7] P. Spirtes and C. Glymour, "An algorithm for fast recovery of sparse causal graphs," *Social Science Computer Review*, vol. 9, no. 1, p. 62–72, Apr. 1991.

[8] T. Verma and J. Pearl, "Equivalence and synthesis of causal models," in *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, ser. UAI '90. USA: Elsevier Science Inc., Jul. 1990, p. 255–270.

[9] P. Spirtes, C. N. Glymour, and R. Scheines, *Causation, Prediction, and Search*. The MIT Press, 2000.

[10] J. Ramsey, J. Zhang, and P. Spirtes, "Adjacency-faithfulness and conservative causal inference," in *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2006, pp. 401–408.

[11] N. Harris and M. Drton, "PC algorithm for nonparanormal graphical models," *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 3365–3383, 2013.

[12] D. Colombo, M. H. Maathuis *et al.*, "Order-independent constraint-based causal structure learning." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3741–3782, 2014.

[13] R. Cui, P. Groot, and T. Heskes, "Copula PC algorithm for causal discovery from mixed data," in *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2016), Part II 2016*. Springer, 2016, pp. 377–392.

[14] T. D. Le, T. Hoang, J. Li, L. Liu, H. Liu, and S. Hu, "A fast PC algorithm for high dimensional causal discovery with multi-core PCs," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 16, no. 5, pp. 1483–1495, 2016.

[15] A. Sondhi and A. Shojaie, "The reduced PC-algorithm: Improved causal structure learning in large random networks," *Journal of Machine Learning Research*, vol. 20, no. 164, pp. 1–31, 2019.

[16] B. Zarebavani, F. Jafarinejad, M. Hashemi, and S. Salehkaleybar, "cuPC: CUDA-based parallel PC algorithm for causal structure learning on GPU," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 530–542, 2019.

[17] E. Giudice, J. Kuipers, and G. Moffa, "The dual PC algorithm for structure learning," in *International Conference on Probabilistic Graphical Models*. PMLR, 2022, pp. 301–312.

[18] M. Kalisch and P. Bühlman, "Estimating high-dimensional directed acyclic graphs with the PC-algorithm." *Journal of Machine Learning Research*, vol. 8, no. 3, 2007.

[19] P. Spirtes, C. Meek, and T. Richardson, "Causal inference in the presence of latent variables and selection bias," in *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc, 1995, pp. 499–506.

[20] J. Zhang, "On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias," *Artificial Intelligence*, vol. 172, no. 16-17, pp. 1873–1896, 2008.

[21] I. Tsamardinos, C. F. Aliferis, and A. Statnikov, "Time and sample efficient discovery of Markov blankets and direct causal relations," in *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, 2003, pp. 673–678.

[22] D. Margaritis and S. Thrun, "Bayesian network induction via local neighborhoods," in *Advances in Neural Information Processing Systems*, vol. 12. The MIT Press, 1999, pp. 505 – 511.

[23] C. F. Aliferis, I. Tsamardinos, and A. Statnikov, "HITON: A novel Markov blanket algorithm for optimal variable selection," in *AMIA Annual Symposium Proceedings*, vol. 2003, 2003, p. 21.

[24] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos, "Local causal and Markov blanket induction for causal discovery and feature selection for classification. Part I: Algorithms and empirical evaluation," *Journal of Machine Learning Research*, vol. 11, no. 1, pp. 171–234, 2010.

[25] K. Yu, X. Wu, H. Wang, and W. Ding, "Causal discovery from streaming features," in *2010 IEEE International Conference on Data Mining*. IEEE, 2010, pp. 1163–1168.

[26] K. Yu, X. Wu, W. Ding, and H. Wang, "Exploring causal relationships with streaming features," *The Computer Journal*, vol. 55, no. 9, pp. 1103–1117, 2012.

[27] X. Guo and J. Yang, "Causal structure learning algorithm based on streaming features," in *2017 IEEE International Conference on Big Knowledge*. IEEE, 2017, pp. 192–197.

[28] J. Yang, X. Guo, N. An, A. Wang, and K. Yu, "Streaming feature-based causal structure learning algorithm with symmetrical uncertainty," *Information Sciences*, vol. 467, pp. 708–724, 2018.

[29] J. Yang, L. Jiang, A. Shen, and A. Wang, "Online streaming features causal discovery algorithm based on partial rank correlation," *IEEE Transactions on Artificial Intelligence*, vol. 4, no. 1, pp. 197–208, 2022.

[30] D. You, S. Dong, S. Niu, H. Yan, Z. Chen, S. Jin, D. Wu, and X. Wu, "Local causal structure learning for streaming features," *Information Sciences*, vol. 647, p. 119502, 2023.

[31] C. Meek, "Causal inference and causal explanation with background knowledge," in *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann, 1995, pp. 403–410.

[32] J. Wang, M. Wang, P. Li, L. Liu, Z. Zhao, X. Hu, and X. Wu, "Online feature selection with group structure analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 11, pp. 3029–3041, 2015.

[33] T. Richardson and P. Spirtes, "Ancestral graph Markov models," *The Annals of Statistics*, vol. 30, no. 4, pp. 962–1030, 2002.

[34] J. Zhang, "Causal Inference and Reasoning in Causally Insufficient Systems," Ph.D. dissertation, Carnegie Mellon University, 2006.

[35] P. Spirtes, "An anytime algorithm for causal inference," in *International Workshop on Artificial Intelligence and Statistics*. PMLR, 2001, pp. 278–285.

[36] D. Colombo, M. H. Maathuis, M. Kalisch, and T. S. Richardson, "Learning high-dimensional directed acyclic graphs with latent and selection variables," *The Annals of Statistics*, vol. 40, pp. 294–321, 2012.

[37] A. Ankan and A. Panda, "pgmpy: Probabilistic graphical models using python," in *Proceedings of the 14th Python in Science Conference*. SciPy, pp. 6–11.

[38] J. Adamczyk, J. Poziemski, and P. Siedlecki, "ApisTox: A new benchmark dataset for the classification of small molecules toxicity on honey bees," *Scientific Data*, vol. 12, no. 1, p. 5, 2025.

[39] D. M. Hausman and J. Woodward, "Independence, invariance and the causal Markov condition," *The British Journal for the Philosophy of Science*, vol. 50, no. 4, pp. 521–583, 1999.

[40] S. L. Lauritzen, A. P. Dawid, B. N. Larsen, and H.-G. Leimer, "Independence properties of directed Markov fields," *Networks*, vol. 20, no. 5, pp. 491–505, 1990.

[41] T. Verma, "Graphical aspects of causal models," *Technical Report R-191, UCLA*, 1993.

[42] P. Spirtes and T. S. Richardson, "A polynomial time algorithm for determining DAG equivalence in the presence of latent variables and selection bias," in *Proceedings of the 6th International Workshop on Artificial Intelligence and Statistics*, vol. 12. PMLR, 1997, pp. 489–500.

[43] D. Colombo, M. H. Maathuis, M. Kalisch, and T. S. Richardson, "Supplement to "Learning high-dimensional directed acyclic graphs with latent and selection variables."," *The Annals of Statistics*, vol. 40, no. 1 *Supplement*, pp. 1–33, 2012.

[44] I. Tsamardinos, L. Brown, and C. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," *Machine Learning*, vol. 65, pp. 31–78.

[45] N. K. Kitson, A. C. Constantinou, Z. Guo, Y. Liu, and K. Chobtham, "A survey of Bayesian network structure learning," *Artificial Intelligence Review*, vol. 56, no. 8, pp. 8721–8814, 2023.

[46] E. Masanet, A. Shehabi, N. Lei, S. Smith, and J. Koomey, "Recalibrating global data center energy-use estimates," *Science*, vol. 367, no. 6481, pp. 984–986, 2020.

[47] J. Runge, S. Bathiany, E. Bollt, G. Camps-Valls, D. Coumou, E. Deyle, C. Glymour, M. Kretschmer, M. D. Mahecha, J. Muñoz-Marí, E. H. van Nes, J. Peters, R. Quax, M. Reichstein, M. Scheffer, B. Schölkopf, P. Spirtes, G. Sugihara, J. Sun, and J. Zscheischler, "Inferring causation from time series in earth system sciences," *Nature Communications*, vol. 10, no. 1, p. 2553, 2019.

Anexos

Appendix A

Informe de Originalidad de Turnitin

Se añade el informe de originalidad de Turnitin en las siguientes páginas del trabajo.

Recibo digital

Este recibo confirma quesu trabajo ha sido recibido por Turnitin. A continuación podrá ver la información del recibo con respecto a su entrega.

La primera página de tus entregas se muestra abajo.

Autor de la entrega:	JOSE MARIA DE MIGUEL CONTRERAS
Título del ejercicio:	Turnitin Memoria Final
Título de la entrega:	TFG-21m018.pdf
Nombre del archivo:	11101_JOSE_MARIA_DE_MIGUEL_CONTRERAS_TFG-21m018_83...
Tamaño del archivo:	6.35M
Total páginas:	60
Total de palabras:	15,908
Total de caracteres:	82,671
Fecha de entrega:	04-jun.-2025 09:11p. m. (UTC+0200)
Identificador de la entrega:	2692206253

Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Grado en Matemáticas e Informática

Trabajo Fin de Grado

Bachelor's Thesis

Descubrimiento Causal con Redes Bayesianas

Causal Discovery with Bayesian Networks

Autor: José María De Miguel Contreras
Tutores: Concha Bielza, Pedro Larrañaga

Madrid, Junio 2025

JOSE MARIA DE MIGUEL CONTRERAS

TFG-21m018.pdf

Turnitin Memoria Final

TFG ETSIINF (Moodle PP)

Universidad Politecnica de Madrid

Detalles del documento

Identificador de la entrega
trn:oid:::1:3268710767

Fecha de entrega
4 jun 2025, 9:11 p.m. GMT+2

Fecha de descarga
4 jun 2025, 9:32 p.m. GMT+2

Nombre de archivo
11101_JOSE_MARIA_DE_MIGUEL_CONTRERAS_TFG-21m018_83714_175543256.pdf

Tamaño de archivo
6.3 MB

60 Páginas

15.908 Palabras

82.671 Caracteres

3% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Exclusions

- 1 Excluded Source

Match Groups

29 Not Cited or Quoted 3%
Matches with neither in-text citation nor quotation marks

6 Missing Quotations 1%
Matches that are still very similar to source material

0 Missing Citation 0%
Matches that have quotation marks, but no in-text citation

0 Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
3% Submitted works (Student Papers)

Match Groups

- **29** Not Cited or Quoted 3%
  Matches with neither in-text citation nor quotation marks
- **6** Missing Quotations 1%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

0% Internet sources
0% Publications
3% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Student papers	
Universidad Politécnica de Madrid		<1%
2	Student papers	
University College London		<1%
3	Student papers	
University of South Australia		<1%
4	Student papers	
The University of Manchester		<1%
5	Student papers	
University College Dublin (UCD)		<1%
6	Student papers	
Erasmus University of Rotterdam		<1%
7	Student papers	
University of Edinburgh		<1%
8	Student papers	
Queen Mary and Westfield College		<1%
9	Student papers	
University of Sheffield		<1%
10	Student papers	
King's College		<1%

11 Student papers
Ahmedabad University <1%

12 Student papers
University of York <1%

13 Student papers
Carnegie Mellon University <1%

14 Student papers
Imperial College of Science, Technology and Medicine <1%

15 Student papers
University of Oxford <1%

16 Student papers
Corporación Universitaria Minuto de Dios,UNIMINUTO <1%

17 Student papers
Higher Education Commission Pakistan <1%

18 Student papers
University of Florida <1%

19 Student papers
University of Sydney <1%

20 Student papers
University of Leeds <1%

Este documento esta firmado por

	**Firmante**	CN=tfgm.fi.upm.es, OU=CCFI, O=ETS Ingenieros Informaticos - UPM, C=ES
	Fecha/Hora	Wed Jun 04 23:56:55 CEST 2025
	Emisor del Certificado	EMAILADDRESS=camanager@etsiinf.upm.es, CN=CA ETS Ingenieros Informaticos, O=ETS Ingenieros Informaticos - UPM, C=ES
	Numero de Serie	561
	Metodo	urn:adobe.com:Adobe.PPKLite:adbe.pkcs7.sha1 (Adobe Signature)