Learning and interpreting asymmetry-labeled DAGs: A case study on COVID-19 fear

Manuele Leonelli[1*] and Gherardo Varando[2]

[1*]School of Science and Technology, IE University, Madrid, Spain.
[2]Image Processing Laboratory, Universitat de València, València, Spain.

*Corresponding author(s). E-mail(s): manuele.leonelli@ie.edu;
Contributing authors: gherardo.varando@ie.edu;

Abstract

Bayesian networks are widely used to learn and reason about the dependence structure of discrete variables. However, they can only formally encode symmetric conditional independence, which is often too strict to hold in practice. Asymmetry-labeled DAGs have been recently proposed to extend the class of Bayesian networks by relaxing the symmetric assumption of independence and denoting the dependence between the variables of interest. Here, we introduce novel structural learning algorithms for this class of models, which, whilst efficient, allow for a straightforward interpretation of the underlying dependence structure. A comprehensive computational study highlights the efficiency of the algorithms. A real-world data application using data from the Fear of COVID-19 Scale collected in Italy showcases their use in practice.

Keywords: Bayesian networks, Conditional independence, Probabilistic graphical models, Staged trees, Structural learning

1 Introduction

Bayesian networks (BNs) are probabilistic graphical models that concisely represent the dependence structure between discrete variables through a directed acyclic graph (DAG) (e.g. Scutari & Denis, 2021; Sucar, 2021, for recent overviews). Any conditional independence between variables embedded in the model can be directly read from the underlying DAG through the so-called D-separation criterion (Pearl, 2009).

However, in practical applications, it has been found that the symmetric assumption of conditional independence is often too restrictive, and models graphically depicting asymmetric independence are needed. Various notions of asymmetric conditional independence have been since defined, including context-specific, partial, and local (Pensar, Nyman, Lintusaari, & Corander, 2016), and formal studies of their properties appeared (e.g. Corander, Hyttinen, Kontinen, Pensar, & Väänänen, 2019). Although extensions of BNs embedding and representing asymmetric conditional independence have been defined (e.g. Nicolussi & Cazzaro, 2021; Pensar, Nyman, & Corander, 2017; Pensar, Nyman, Koski, & Corander, 2015; Talvitie, Eggeling, & Koivisto, 2019), they often lose the intuitiveness associated to DAGs and no software is available for their use in practice.

Asymmetry-labeled DAGs (ALDAGs) have been recently introduced as an extension of DAGs where edges are labeled according to the type of dependence existing between any pairs of random variables (Varando, Carli, & Leonelli, 2024). They are constructed starting from a tree-based probabilistic graphical model called *staged tree* (Collazo, Görgen, & Smith, 2018; Smith & Anderson, 2008), to which a conversion algorithm is applied to construct the associated ALDAG. Standard tools for DAGs, e.g. the already-mentioned D-separation criterion, also hold for ALDAGs. However, they also carry additional information in the form of the associated edge labeling and the underlying staged tree used to construct them.

ALDAGs share features with labeled DAGs of Pensar et al. (2015), but they differ in two critical aspects: first, labeled DAGs can only embed context-specific independence while ALDAGs represent any asymmetric independence; second, labeled DAGs specifically report the contexts over which independences hold, while ALDAGs do not. There are two reasons behind this: on the one hand, the specific independences in ALDAGs can be read from the associated staged tree (see below); on the other, for applications with a larger number of variables, the required contexts are often too complex to be reported within the DAG.

While there are now dozens of structural learning algorithms for BNs, which are constantly upgraded to improve speed and accuracy (see e.g. X. Huang, Guo, Li, & Yu, 2023; Kuipers, Suter, & Moffa, 2022; Liu, Gao, Ru, Tan, & Wang, 2023; Luo, Zhao, & Du, 2019; Tsagris, 2021, for recent contributions), only two algorithms have been introduced for ALDAGs. Varando et al. (2024) defined the first routine for learning generic ALDAGs, but this needs to scale more efficiently with the number of variables. Leonelli and Varando (2022) proposed a structural learning algorithm for sparse ALDAGs with few edges consisting of three steps: (i) a DAG with an upper-bound on the number of parents is learned from data; (ii) the learned DAG is refined into a staged tree embedding asymmetric conditional independences; (iii) the staged tree is converted into its associated ALDAG representation. However, this routine suffers from various drawbacks:

- the ordering of the variables is chosen from one arbitrary topological order of the DAG in step (i), which may not be optimal for the ALDAG since it only considers symmetric forms of independence;
- the parents of a variable are also chosen from the DAG in step (i), and again, this choice may be highly affected by overlooking asymmetric dependences.

Here, we propose novel structural learning algorithms for ALDAGs that overcome these difficulties by replacing step (i) of learning a DAG from data with a procedure that chooses the ordering of the variables and the parents of each variable in the final ALDAG. To this end, we take advantage of the conditional mutual information already utilized in Carli, Leonelli, and Varando (2023) to quantify the strength of dependence between variables.

The algorithms we introduce here learn sparse ALDAGs with few connections between variables. The imposition of sparsity constraints is now the gold standard in learning traditional DAGs and related models (e.g. Aragam & Zhou, 2015; Li et al., 2020; Shajoonnezhad & Nikanjam, 2023; Wang & Allen, 2023). In our context, sparsity has two major advantages. First, the learning algorithms are more efficient since they search for an optimal model over a restricted number of available ALDAGs, speeding up computations. Second, they allow for straightforward and interpretable visualization of asymmetric dependence, which would not be feasible for generic ALDAGs. This aligns with the most recent trends in artificial intelligence and machine learning in explaining a model's outputs (Hagras, 2018; Linardatos, Papastefanopoulos, & Kotsiantis, 2020; Lundberg et al., 2020).

The paper is structured as follows. Section 2 reviews DAGs, ALDAGs, and notions of symmetric and asymmetric conditional independence. Section 3 first reviews staged trees, which are needed for constructing ALDAGs, and then introduces novel structural learning routines for ALDAGs that are both efficient and explainable. Section 4 presents a simulation study to investigate the quality of our algorithms. Section 5 presents a comprehensive analysis of the use of ALDAGs to study the relationship between factors influencing the fear of the COVID-19 virus. The paper is concluded with a discussion.

2 Asymmetric dependence and ALDAGs

2.1 DAGs and conditional independence

Let $G = ([p], F)$ be a directed acyclic graph (DAG) with vertex set $[p] = \{1, \dots, p\}$ and edge set F. Let $\boldsymbol{X} = (X_i)_{i\in[p]}$ be categorical random variables with joint mass function P and sample space $\mathbb{X} = \times_{i\in[p]}\mathbb{X}_i$. For $A \subset [p]$, we let $\boldsymbol{X}_A = (X_i)_{i\in A}$ and $\boldsymbol{x}_A = (x_i)_{i\in A}$ where $\boldsymbol{x}_A \in \mathbb{X}_A = \times_{i\in A}\mathbb{X}_i$.

Definition 1. *We say that a joint mass function P over categorical variables $\boldsymbol{X}$ is Markov to a DAG G if, for $\boldsymbol{x} \in \mathbb{X}$,*

$$P(\boldsymbol{x}) = \prod_{k\in[p]} P(x_k|\boldsymbol{x}_{\Pi_k}),$$

where Π_k is the parent set of k in G and $P(x_k|\boldsymbol{x}_{\Pi_k})$ is a shorthand for $P(X_k = x_k|\boldsymbol{X}_{\Pi_k} = \boldsymbol{x}_{\Pi_k})$.

The ordered Markov condition implies conditional independences of the form

$$X_i \perp\!\!\!\perp \boldsymbol{X}_{[i-1]} \,|\, \boldsymbol{X}_{\Pi_i}, \tag{1}$$

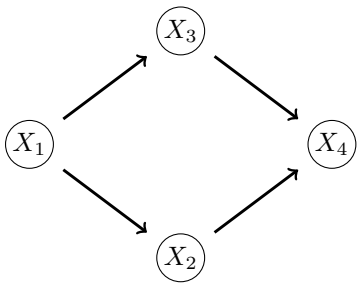

Fig. 1 An example of a DAG over four random variables.

which are equivalent to

$$P(x_i|\boldsymbol{x}_{[i-1]\setminus\Pi_i}, \boldsymbol{x}_{\Pi_i}) = P(x_i|\boldsymbol{x}_{\Pi_i}), \qquad \text{for all } \boldsymbol{x} \in \mathbb{X}. \tag{2}$$

Figure 1 shows a DAG over four random variables $X_1, \ldots, X_4$, implying the symmetric conditional independences $X_3 \perp\!\!\!\perp X_2|X_1$ and $X_4 \perp\!\!\!\perp X_1|X_2, X_3$. The associated Markov factorization is $P(x_4|x_3, x_2)P(x_3|x_2)P(x_2|x_1)P(x_1)$.

2.2 Asymmetric conditional independence

BNs have the capability of expressing only symmetric conditional independence of the form in (1) and (2). The most common asymmetric extension of conditional independence is the so-called *context-specific* independence, often represented by associating a tree to each vertex of a BN. Henceforth, let A, B and C be three disjoint subsets of $[p]$.

Definition 2. *We say that $\boldsymbol{X}_A$ is context-specific independent of $\boldsymbol{X}_B$ given context $\boldsymbol{x}_C \in \mathbb{X}_C$ if*

$$P(\boldsymbol{x}_A|\boldsymbol{x}_B, \boldsymbol{x}_C) = P(\boldsymbol{x}_A|\boldsymbol{x}_C) \tag{3}$$

holds for all $(\boldsymbol{x}_A, \boldsymbol{x}_B) \in \mathbb{X}_{A\cup B}$ and write $\boldsymbol{X}_A \perp\!\!\!\perp \boldsymbol{X}_B|\boldsymbol{x}_C$.

The condition in (3) reduces to standard conditional independence in (1) if it holds for all $\boldsymbol{x}_C \in \mathbb{X}_C$.

A more general definition of non-symmetric conditional independence called *partial conditional independence* was introduced in Pensar et al. (2016).

Definition 3. *We say that $\boldsymbol{X}_A$ is partially conditionally independent of $\boldsymbol{X}_B$ in the domain $\mathcal{D}_B \subseteq \mathbb{X}_B$ given context $\boldsymbol{X}_C = \boldsymbol{x}_C$ if*

$$P(\boldsymbol{x}_A|\boldsymbol{x}_B, \boldsymbol{x}_C) = P(\boldsymbol{x}_A|\tilde{\boldsymbol{x}}_B, \boldsymbol{x}_C) \tag{4}$$

holds for all $(\boldsymbol{x}_A, \boldsymbol{x}_B), (\boldsymbol{x}_A, \tilde{\boldsymbol{x}}_B) \in \mathbb{X}_A \times \mathcal{D}_B$ and write $\boldsymbol{X}_A \perp\!\!\!\perp \boldsymbol{X}_B|\mathcal{D}_B, \boldsymbol{x}_C$.

Clearly, (3) and (4) coincide if $\mathcal{D}_B = \mathbb{X}_B$. Furthermore, the sample space $\mathbb{X}_B$ must contain more than two elements for a non-trivial partial conditional independence to hold.

A final condition is the so-called *local conditional independence*. For $i \in [p]$ and an $A \subset [p]$ such that $A \cap \{i\} = \emptyset$, local conditional independence expresses equalities of probabilities of the form

$$P(x_i|\boldsymbol{x}_A) = P(x_i|\tilde{\boldsymbol{x}}_A) \tag{5}$$

for all $x_i \in \mathbb{X}_i$ and two $\boldsymbol{x}_A, \tilde{\boldsymbol{x}}_A \in \mathbb{X}_A$. Notice that in terms of generality, (3) $\preceq$ (4) $\preceq$ (5). Condition (5) simply states that some conditional probability distributions are identical, where no discernable patterns as in Equations (3) and (4) can be detected.

2.3 Classes of dependence

Suppose that a DAG model is available, but that we believe additional structure, which is not necessarily symmetric, exists between the variables. Such a structure might consist of equalities as those in Equations (3)-(5) associated with asymmetric independences. The following definition formalizes the types of dependence between two random variables joined by an edge in a DAG G.

Definition 4. *Let P be the joint mass function of $\mathbf{X}$ and P be Markov for a DAG $G = ([p], F)$. For each $(j, i) \in F$ we say that the dependence of X_i from X_j is of class*

- context*, if X_i and X_j are context-specific independent given some context $\mathbf{x}_C$ with $C = \Pi_i \setminus \{j\}$.*
- partial*, if X_i is partially conditionally independent of X_j in a domain $\mathcal{D}_j \subset \mathbb{X}_j$ given a context $\mathbf{x}_C$ with $C = \Pi_i \setminus \{j\}$; and X_i and X_j are not context-specific independent given the same context $\mathbf{x}_C$.*
- local*, if none of the above hold and a local independence of the form $P(x_i|\mathbf{x}_{\Pi_i}) = P(x_i|\tilde{\mathbf{x}}_{\Pi_i})$ is valid where $x_j \neq \tilde{x}_j$.*
- total*, if none of the above hold.*

In standard DAG models, all edges have the total label, but additional labels might be considered to denote more flexible types of dependence. Notice that if the class of dependence between X_i and X_j is context or partial then there may also be local independence statements as in Equation (5) involving these two variables. Similarly, the dependence between X_i and X_j can be both context and partial concerning two different contexts. On the other hand, if their class of dependence is local then, by definition, there are no context-specific or partial equalities. The lack of an edge is associated with conditional independence as formalized in Equation (1) and as in standard DAGs.

2.4 ALDAGs

Given the classes of dependence introduced in Definition 4, we can now embellish a DAG with additional information about asymmetric dependence structure. The resulting labeled graph is called ALDAG in Varando et al. (2024), where edges are colored depending on the label and the type of relationship between variables.

Formally, let G be a DAG, F its edge set, and

$$\mathcal{L}^A = \{\text{'context'}, \text{'partial'}, \text{'context/partial'}, \text{'local'}, \text{'total'}\}$$

be the set of edge labels marking the type of dependence.

Definition 5. *An ALDAG is a pair (G, ψ) where $G = ([p], F)$ is a DAG and ψ is a function from the edge set of G to $\mathcal{L}^A$, i.e. $\psi : F \to \mathcal{L}^A$. We say that a joint mass function P is compatible with an ALDAG (G, ψ) if P is Markov to G and additionally*

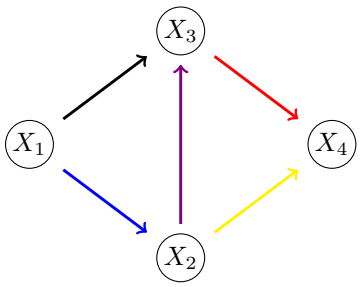

Fig. 2 An example of an ALDAG over four random variables. The edge coloring is: red - context; blue - partial; violet - context/partial; green - local; black - total.

P respects all the edge labels given by ψ; that is, for each $(j,i) \in F$, X_i is $\psi(i,j)$ dependent from X_j.

Henceforth, we represent the labeling via coloring the edges of the ALDAG. Standard BNs have an ALDAG representation where all edges have the label 'total'. Notice that standard features of BNs are also valid over ALDAGs: for instance, the already-mentioned d-separation criterion and fast probability propagation algorithms. Furthermore, they have been recently used for causal discovery by using the extra information given by the edge labels (Leonelli & Varando, 2023).

Figure 2 gives an example of an ALDAG over four random variables. The lack of the edge from X_1 to X_4 is associated with the symmetric conditional independence $X_4 \perp\!\!\!\perp X_1 | X_2, X_3$. The black edge with label total states that the relationship between X_1 and X_3 is symmetric. All other pairs of variables connected by an edge, which do not have a label total, highlight that there are some equalities in the model's conditional probabilities associated with asymmetric independences. Furthermore, in the case of the edge between X_2 and X_4, which has a label local, this asymmetric dependence is very unstructured and cannot be formalized either as context-specific or as partial.

3 Learning ALDAGs from data

Given observations from a vector of categorical variables, we want to use structural learning algorithms to learn the ALDAG that better describes the data dependence structure. Such algorithms will use the class of staged tree graphical models (Collazo et al., 2018; Smith & Anderson, 2008), which are henceforth reviewed next.

3.1 Staged trees

Unlike BNs, whose graphical representation is a DAG, staged trees visualize conditional independence using a colored tree. Let (V, E) be a directed, finite, rooted tree with vertex set V, root node v_0, and edge set E. For each $v \in V$, let $E(v) = \{(v, w) \in E\}$ be the set of edges emanating from v and $\mathcal{C}$ be a set of labels.

Definition 6. *An $\mathbf{X}$-compatible staged tree is a triple $T = (V, E, \theta)$, where (V, E) is a rooted directed tree and:*

1. $V = v_0 \cup \bigcup_{i\in[p]} \mathbb{X}_{[i]}$;
2. *For all* $v, w \in V$, $(v, w) \in E$ *if and only if* $w = \boldsymbol{x}_{[i]} \in \mathbb{X}_{[i]}$ *and* $v = \boldsymbol{x}_{[i-1]}$, *or* $v = v_0$ *and* $w = x_1$ *for some* $x_1 \in \mathbb{X}_1$;
3. $\theta : E \to \mathcal{L} = \mathcal{C} \times \cup_{i\in[p]}\mathbb{X}_i$ *is a labelling of the edges such that* $\theta(v, \boldsymbol{x}_{[i]}) = (\kappa(v), x_i)$ *for some function* $\kappa : V \to \mathcal{C}$. *The function* k *is called the colouring of the staged tree* T.

If $\theta(E(v)) = \theta(E(w))$ *then* v *and* w *are said to be in the same* stage. *The equivalence classes induced by* $\theta(E(v))$ *form a partition of the internal vertices of the tree in* stages.

Points 1 and 2 above construct a rooted tree where each root-to-leaf path, or equivalently each leaf, is associated with an element of the sample space $\mathbb{X}$. Then, a labeling of the edges of such a tree is defined where labels are pairs with one element from a set $\mathcal{C}$ and the other from the sample space $\mathbb{X}_i$ of the corresponding variable X_i in the tree. By construction, $\mathbf{X}$-compatible staged trees are such that two vertices can be in the same stage if and only if they correspond to the same sample space.

Figure 3 reports an example of an (X_1, X_2, X_3, X_4)-compatible staged tree model over two ternary variables (X_1 and X_2 with levels low/medium/high) and two binary ones (X_3 and X_4 with levels low/high). The coloring given by the function κ is shown in the vertices, and each edge $(\cdot, (x_1, \ldots, x_i))$ is labeled with x_i. The edge labeling θ can be read from the graph by combining the text label and the color of the emanating vertex. For example, $\theta(v_1, v_6) \neq \theta(v_2, v_7) = \theta(v_2, v_8) \neq \theta(v_2, v_9)$. This representation of the labeling θ over vertices is equivalent to that over edges while being more interpretable and is henceforth used. There are 31 internal vertices and the staging is $\{v_0\}$, $\{v_1, v_2\}$, $\{v_3\}$, $\{v_4, v_5, v_6\}$, $\{v_7, v_8\}$, $\{v_9\}$, $\{v_{10}\}$, $\{v_{11}\}$, $\{v_{12}\}$, $\{v_{13}, v_{14}, v_{19}, v_{20}, v_{25}, v_{26}\}$, $\{v_{15}, v_{18}, v_{21}, v_{24}, v_{27}, v_{30}\}$, $\{v_{16}, v_{22}, v_{28}\}$, and $\{v_{17}, v_{23}, v_{29}\}$.

Conditional independence is formally modeled and represented in staged trees by the labeling θ. As an illustration, consider the staged tree in Figure 3. The fact that v_4, v_5 and v_6 are in the same stage is associated with the context-specific independence $X_3 \perp\!\!\!\perp X_2 | X_1 = \text{low}$. The green stage $\{v_7, v_8\}$ represents the partial independence between X_2 and X_3 in the domain $X_2 \in \{\text{low,medium}\}$ and context $X_1 = \text{medium}$. The fact that, for instance, v_{27} and v_{30} are in the same stage is a generic local independence stating that the conditional distribution of X_4 given $X_1 = \text{high}, X_2 = \text{high}, X_3 = \text{high}$ is equal to that of X_4 given $X_1 = \text{high}, X_2 = \text{medium}, X_3 = \text{low}$. Lastly, the repeating staging pattern in $v_{13} - v_{18}$, $v_{19} - v_{24}$ and $v_{25} - v_{30}$ represents the symmetric conditional independence $X_4 \perp\!\!\!\perp X_1 | X_2, X_3$.

Staged trees represent a wide array of asymmetric independences via the staging of the internal vertices. However, as the number of possible atomic events increases, it becomes more challenging to assess the underlying independences by visual inspection. For this reason, Varando et al. (2024) introduced ALDAGs as a compression of the information stored in a staged tree, which can be more intuitively read, as well as an algorithm to convert a staged tree into its ALDAG representation. Such a conversion algorithm requires two steps: (i) constructing a DAG G_T from the staged tree T so that if $\boldsymbol{X}_A \perp\!\!\!\perp \boldsymbol{X}_B | \boldsymbol{X}_C$ in T then $\boldsymbol{X}_A$ and $\boldsymbol{X}_B$ are d-separated by $\boldsymbol{X}_C$ in G_T; (ii) labeling each edge in G_T according to the type of dependence existing between the associated variables. As an illustration, the ALDAG associated with the staged tree in Figure 3 is

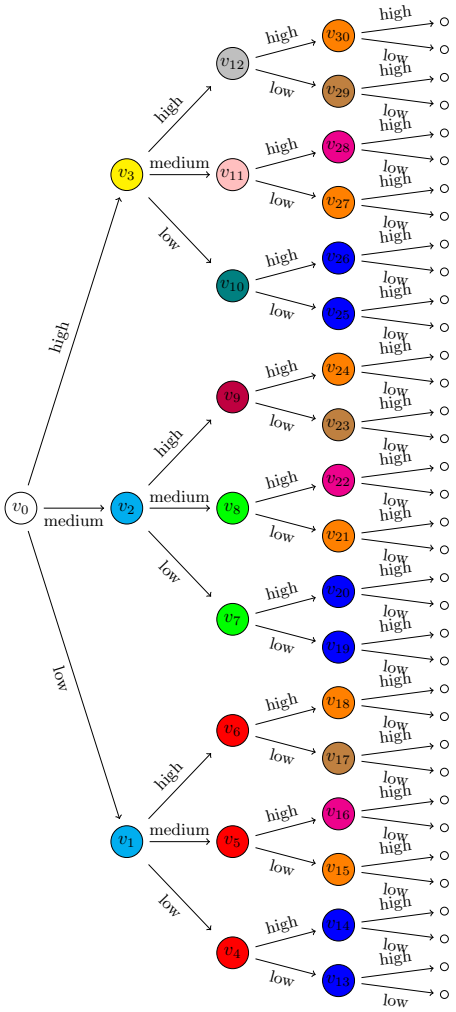

Fig. 3 A staged tree over two ternary variables and two binary variables, whose ALDAG is in Figure 2.

the one in Figure 2. The lack of the edge (X_1, X_4) follows from the repeating staging pattern in $v_{13} - v_{18}$, $v_{19} - v_{24}$ and $v_{25} - v_{30}$. The labeled edges summarize the very complex staging of the staged tree, which cannot be simply represented by a standard BN.

3.2 Learning ALDAGs with a fixed order

Given the conversion algorithm from staged tree to ALDAG introduced in Varando et al. (2024), learning an ALDAG from data comes down to learning the associated staged tree. There is now a wide array of algorithms to learn different types of staged trees (Barclay, Hutton, & Smith, 2013; Collazo & Smith, 2016; Freeman & Smith,

2011; Leonelli & Varando, 2022), which are also implemented in the freely-available `stagedtrees` R package (Carli, Leonelli, Riccomagno, & Varando, 2022). Because the model search space of staged trees is huge and much larger than that of DAGs, all these algorithms are based on greedy heuristic searches that, at each iteration, optimize a model score (most often the BIC as discussed in Görgen, Leonelli, & Marigliano, 2022). More formally, given a fixed ordering of p binary variables, the number of staged tree models is

$$\prod_{t=1}^{p-1} B_{2^t} \tag{6}$$

where B_t is the t-th *Bell number* counting the number of partitions of a set of t elements (Cowell & Smith, 2014). With $p = 6$, the number of staged trees is already in the order of 10^{41}.

Research has been pursued in simplifying the task of learning a staged tree by considering only specific sub-classes. In Carli et al. (2023), naive staged trees are defined with the same number of parameters of a naive BN over the same variables. In Varando et al. (2024), algorithms that learn trees whose ALDAG does not have local edges are studied. Lastly, in Leonelli and Varando (2022) k-parents staged trees are defined, whose associated ALDAGs have few parents. Since these are used in our novel algorithms for sparse ALDAGs, we recall here their formal definition.

Definition 7. *A staged tree T is in the class of k-parents staged trees if the maximum in-degree in G_T is less or equal to k.*

One of the first solutions to make structural learning of BNs scalable was to limit the number of parents each variable can have (e.g. Tsamardinos, Brown, & Aliferis, 2006). This was imposed not only to restrict the model space of possible DAGs but also made sense from an applied point of view since, most often, only a limited number of variables can be expected to directly influence one another. Setting a maximum number of parents is also available in the standard `bnlearn` software (Scutari, 2010).

Here we derive a new formal result enumerating the number of k-parents staged trees.

Proposition 1. *The number of k-parents staged trees given a fixed ordering of p binary variables is*

$$\prod_{t=1}^{k} B_{2^t} \prod_{t=k+1}^{p-1} \binom{t}{k} B_{2^k} \tag{7}$$

Table 1 gives an overview of the size of the model search space for DAGs, staged trees, and k-parents staged trees. The number of staged trees is much bigger than the one for DAGs, but by restricting the number of parents, there is a significant reduction in the model search space.

3.3 Learning sparse ALDAGs with a fixed order

The ALDAG associated with a k-parents staged tree is consequently sparse whenever k is set to a small integer. Notice that in Collazo and Smith (2016), staged trees whose ALDAGs would be sparse are learned using non-local priors, which have also been effective in learning sparse DAGs (e.g. Cao & Yang, 2021).

p	DAGs	STs	1-parent STs	2-parents STs	3-parents STs	4-parents STs
2	3	2	2	2	2	2
3	25	30	8	30	30	30
4	543	124200	48	1350	124200	124200
5	29281	$1.3 \cdot 10^{15}$	384	121500	$2.0 \cdot 10^{10}$	$1.3 \cdot 10^{15}$
6	$3.8 \cdot 10^{7}$	$1.7 \cdot 10^{41}$	3840	$1.8 \cdot 10^{8}$	$8.5 \cdot 10^{13}$	$6.8 \cdot 10^{25}$
7	$1.1 \cdot 10^{9}$	$2.9 \cdot 10^{106}$	46080	$4.1 \cdot 10^{10}$	$7.0 \cdot 10^{18}$	$1.1 \cdot 10^{37}$
8	$7.8 \cdot 10^{11}$	$3.2 \cdot 10^{264}$	645120	$1.3 \cdot 10^{12}$	$1.0 \cdot 10^{24}$	$3.9 \cdot 10^{48}$

Table 1 Number of DAGs, staged trees and k-parents staged trees (both with fixed variables' ordering) for $k = 1, 2, 3, 4$ and p ranging from 2 to 8.

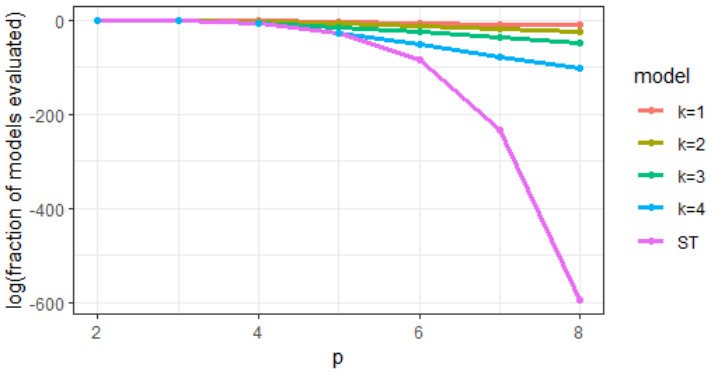

Fig. 4 Ratio between the maximum number of models evaluated and the model search space size for k-parents and generic staged trees.

Here we consider more flexible alternatives to the algorithms discussed in Leonelli and Varando (2022), which required utilizing a specific compatible order to a BN learned from data. Let's also start by assuming that we select a priori a possible ordering of the variables of interest X (henceforth we assume this is the ordering of the positive integers). To select which variables can be parents of a vertex in the learned ALDAG we use *conditional mutual information*. Recall that the conditional mutual information between two variables X_A and X_B given (a vector) $\boldsymbol{X}_C$, $I(X_A; X_B|\boldsymbol{X}_C)$, is defined as

$$I(X_A; X_B|\boldsymbol{X}_C) = \sum_{\boldsymbol{x}_c \in \mathbb{X}_C} P(\boldsymbol{x}_C) \sum_{x_A \in \mathbb{X}_A} \sum_{x_B \in \mathbb{X}_C} P(x_A, x_B|\boldsymbol{x}_C) \ln \left(\frac{P(x_A, x_B|\boldsymbol{x}_C)}{P(x_A|\boldsymbol{x}_C) P(x_B|\boldsymbol{x}_C)} \right) \tag{8}$$

In particular, the parents of a variable X_i are selected iteratively by choosing the variable X_j in $X_{[i-1]}$ maximizing the conditional mutual information with X_i given the already selected parent variables. Of course, if $i \leq k+1$, all preceding variables are the parents of X_i. Notice that the probability distributions in Equation (8) need to be computed empirically from data: for this task, we use the `infotheo` R package (Meyer & Meyer, 2009).

The previous routine constructs a DAG without estimating any of the associated probabilities. The structural learning algorithm for the ALDAG then takes this DAG as input and performs the following steps: (i) convert the DAG into an equivalent staged tree representation (embedding all the DAG conditional independences); (ii) run the backward-hill climbing algorithm of Carli et al. (2022), which at each iteration can only join stages of the underlying staged tree; (iii) transform the resulting staged tree into G_T and add the edge labels. By construction, the resulting ALDAG is such that any vertex has at most k parents.

The pseudocode to implement the above routine is reported in Algorithm 1 in the supplementary material. Notice that the algorithm does not evaluate all possible k-parents staged trees given a fixed ordering of the variables, but only a fraction of them. The following proposition enumerates the maximum number of models the algorithm might consider.

Proposition 2. *The maximum number of k-parents staged tree models evaluated by Algorithm 1 is*

$$\sum_{t=1}^{k} \binom{2^t+1}{3} + (p-k)\binom{2^k+1}{3}. \tag{9}$$

If there is no restriction on the number of parents, the maximum number of models evaluated is

$$\sum_{t=1}^{p-1} \binom{2^t+1}{3}. \tag{10}$$

Notice that this maximum is attained if and only if the best model is such that all vertices at the same depth are in the same stage, i.e. all variables are marginally independent. Figure 4 reports the logarithm of the ratio between the number of models that could be evaluated by Algorithm 1 and the total number of models within each class of either k-parents staged trees or generic staged trees. For generic staged trees, the fraction of explored models dramatically decreases as the number of variables p increases.

3.4 Learning sparse ALDAGs without a fixed order

The methods described in the previous section output an $\boldsymbol{X}_\pi$-compatible staged tree for a possible ordering π of the variables. For a small number of variables, it is possible to simply enumerate all $p!$ possible orders, and select the best one(s) according to a chosen criterion (e.g. BIC). We investigate below the feasibility of such an approach. Notice that equations (6), (7), (9), and (10) generalize to the case of a non-fixed order by simply multiplying them with the term $p!$.

3.5 Learning sparse ALDAGs with a partial order

An intermediate solution to avoid the factorial increase in complexity of considering all possible orders is to consider only those orders that are compatible with a learned BN from data. For each of the allowed orders, an ALDAG is learned from data using the algorithm of Section 3.3, and the best-scoring one according to a chosen criterion (e.g. BIC) is selected. There are three different possible ways to restrict the number of orders that we pursue here:

1. Learn a BN from data using the tabu algorithm and select all compatible orders to the associated DAG;
2. Learn a BN from data using the tabu algorithm and construct the mixed graph representing its equivalence class; we then select all compatible orders by considering the DAG consisting of only the directed edges of this mixed graph;
3. Learn a mixed graph using the PC stable algorithm (e.g. Vowels, Camgoz, & Bowden, 2022), which represents an equivalence class again; we then select all compatible orders by considering the DAG consisting of only the directed edges of this mixed graph.

The last two approaches consider possible orders where only causal relationships between variables (Pearl, 2009) are fixed. By construction, the number of possible orders considered by option 2 cannot be smaller than option 1.

Notice that the above algorithms require learning a BN, or a BN equivalence class, from data, but only to restrict the possible orders to be considered. The parents of the learned ALDAG do not depend on those of this initial BN and are chosen using conditional mutual information as in Section 3.3.

4 Experiments

4.1 Computational study

We start by considering nine datasets from the literature on probabilistic graphical models to assess the performance of sparse ALDAGs and different estimation procedures. If required, variables are discretized using the equal-frequency method. For each dataset, six different estimation procedures are considered and for each of these procedures, $k = 1, 2, 3$ number of parents is used. First, a BN model is learned using the tabu algorithm implemented in the bnlearn R package (Scutari, 2010) (labeled DAG). A topological order of the learned BN is then used to learn an ALDAG using the approach in Leonelli and Varando (2022) (labeled LV). The approaches proposed in this paper are then used: with a fixed order (labeled CMI) and using the three compatible order approaches of Section 3.5 (labeled ORD1, ORD2, and ORD3, respectively). Lastly, if computationally feasible, meaning when the number of variables is less than seven, we implemented an exhaustive model search for every possible variable ordering (labeled ALL). For each dataset, we implemented a 10-fold cross-validation and, for each fold, computed the BIC over the train data and the likelihood over the test data.

The average train BICs and predictive likelihoods are reported in Tables 2 and 3, respectively. We can draw the following conclusions from the tables:

- For every dataset and every algorithm, the BIC and the predictive likelihood decrease as the number of parents k increases.
- In all cases, there is at least one ALDAG algorithm that scores equal to or better than the standard DAG approach using tabu considering both the train BIC and the predictive likelihood. In the majority of the cases, all ALDAG algorithms outperform the DAG model.

data	k	DAG	LV	CMI	ORD1	ORD2	ORD3	ALL
asia	1	**20433.79**	**20433.79**	20636.15	20636.15	20636.15	20636.15	
	2	20004.46	19979.77	19973.42	19973.42	19973.42	**19973.96**	
	3	20004.46	19979.77	19971.45	**19968.93**	**19968.93**	**19968.53**	
cachexia	1	869.53	865.69	878.80	865.69	**862.71**	870.49	
	2	869.53	865.69	837.11	827.96	**813.08**	817.14	
	3	869.53	865.69	797.27	787.87	**765.67**	775.09	
chds	1	2552.58	2548.66	2547.07	2547.07	**2546.09**	2562.66	**2546.09**
	2	2552.58	2548.66	2546.91	2546.37	**2541.86**	2543.40	**2541.86**
	3	2552.58	2548.66	2545.65	2544.67	**2538.80**	2540.52	**2538.80**
coronary	1	**12161.90**	**12161.90**	12163.24	12182.16	12181.47	12204.53	**12161.90**
	2	12114.49	12091.51	12086.92	12123.63	12122.75	12150.08	**12080.24**
	3	12110.10	12061.47	12051.15	12055.81	12054.99	12122.67	**12039.67**
fall	1	125860.10	**124268.10**	128259.90	**124268.10**	**124268.10**	**124268.10**	**124268.10**
	2	124051.90	**123692.20**	123824.80	**123692.20**	**123692.20**	**123692.20**	**123692.20**
	3	124051.90	**123692.20**	123774.90	**123692.20**	**123692.20**	**123692.20**	**123692.20**
ksl	1	**10740.59**	**10740.59**	10982.63	10910.64	10910.64	10954.99	
	2	10632.64	10610.02	10772.5	**10600.28**	**10600.28**	10749.52	
	3	10632.64	10610.02	10739.52	**10583.86**	**10583.86**	10703.90	
mathmarks	1	865.67	851.71	851.79	850.92	**847.21**	848.56	**847.21**
	2	865.67	851.71	835.77	832.24	**823.13**	825.55	**823.12**
	3	865.67	851.71	805.01	797.91	**783.19**	785.03	**783.19**
phd	1	7543.68	7541.50	**7540.09**	7544.73	7544.03	7569.40	**7540.09**
	2	7543.68	7538.35	7529.64	7528.85	**7520.07**	7521.21	**7520.07**
	3	7543.68	7538.35	7518.26	7512.56	7504.91	7509.83	**7503.09**
titanic	1	9594.04	**9583.72**	**9583.72**	**9583.72**	**9583.72**	**9583.72**	**9583.72**
	2	9465.74	9414.08	9454.75	9414.08	**9412.60**	9435.73	**9412.60**
	3	9465.74	9414.08	9402.55	9397.20	**9395.80**	9397.18	**9395.80**

Table 2 Average train BIC for models learned over nine datasets: Bayesian networks (DAG), ALDAGs with the approach of Leonelli and Varando (2022) (LV), ALDAGs with a fixed order (CMI), ALDAGs using compatible orders (ORD1, ORD2 and ORD3), ALDAGs considering all orders (ALL).

- Except for the coronary data (with $k = 2, 3$) and the phd data (with $k = 3$), there is an ALDAG algorithm whose BIC scores equally well as the algorithm considering all possible orders.
- In terms of predictive likelihood, it is often the case that simpler routines (e.g. CMI or ORD1) outperform more complex ones (as ORD2 and ORD3).
- The learning algorithm starting from the equivalence class of the tabu DAG algorithm (ORD2) seems to outperform the one starting from the output of the PC algorithm (ORD3).

Table 4 reports the computational times for all models learned (using the complete dataset). As expected, time increases with k, the number of parents, since the model learned is more complex. The DAG algorithm is the fastest since it considers the smallest model class. Algorithms for ALDAGs with a fixed order (LV and CMI) are comparably fast, whilst those without a fixed order are the slowest, but still feasible for all datasets. In conclusion, it seems that the ORD1 algorithm (which considers all compatible orders to a BN learned with the tabu procedure) gives a fair compromise between fit, prediction, and computation speed.

Last, we report in Table 5 the number of edges having labels total or of some other type (again for models learned over the complete datasets). By default, all BN models have all total edge labels. ALDAGs with one parent per vertex only ($k = 1$) often have only total edge labels. Models with a better fit, corresponding to ALDAGs with $k =$

data	k	DAG	LV	CMI	ORD1	ORD2	ORD3	ALL
asia	1	-1128.68	**-1124.29**	-1136.00	-1136.00	-1136.00	-1136.00	
	2	-1103.51	-1099.60	**-1099.87**	**-1099.87**	**-1099.87**	-1099.99	
	3	-1103.51	-1099.60	-1099.30	**-1099.16**	**-1099.16**	-1099.26	
cachexia	1	-44.80	**-30.27**	-32.66	**-30.27**	-32.35	-32.51	
	2	-44.80	**-30.27**	-32.28	-30.53	-30.92	-31.23	
	3	-44.80	-30.27	-30.90	**-30.15**	-31.14	-31.14	
chds	1	-139.76	-135.96	**-135.46**	**-135.46**	-136.18	-137.74	-136.18
	2	-139.76	-135.96	**-135.59**	-136.04	-136.18	-136.13	-136.18
	3	-139.76	-135.96	**-135.39**	-136.04	-135.84	-136.25	-135.84
coronary	1	-673.28	-668.16	**-667.83**	-670.92	-670.98	-670.74	-668.16
	2	-668.20	-663.28	**-663.01**	-666.72	-667.01	-667.13	-663.88
	3	-667.53	-661.58	**-659.10**	-662.53	-663.04	-666.87	-660.87
fall	1	-6980.88	**-6891.13**	-7113.03	**-6891.13**	**-6891.13**	**-6891.13**	**-6891.13**
	2	-6877.52	**-6856.63**	-6861.01	**-6856.63**	**-6856.63**	**-6856.63**	**-6856.63**
	3	-6877.52	**-6856.63**	-6855.26	**-6856.63**	**-6856.63**	**-6856.63**	**-6856.63**
ksl	1	-595.16	**-583.44**	-599.01	-593.53	-593.53	-594.92	
	2	-585.28	-571.46	-582.32	**-570.43**	**-570.43**	-581.99	
	3	-585.28	-571.46	-581.60	**-570.92**	**-570.92**	-578.62	
mathmarks	1	-44.73	-36.25	-35.79	-36.05	-35.78	**-35.65**	-35.78
	2	-44.73	-36.25	-34.28	-33.84	**-33.01**	-33.20	**-33.01**
	3	-44.73	-36.25	**-30.86**	-31.68	-32.76	-32.11	-32.76
phd	1	-414.44	-408.35	**-408.12**	-409.24	-409.15	-410.20	**-408.12**
	2	-414.44	-408.32	-407.45	**-406.87**	-407.28	-407.78	-407.28
	3	-414.44	-408.32	-406.85	**-406.61**	-408.00	-407.75	-407.57
titanic	1	-528.43	**-525.34**	**-525.34**	**-525.34**	**-525.34**	**-525.34**	**-525.34**
	2	-517.96	-514.42	-517.52	-514.42	**-514.29**	-515.61	**-514.29**
	3	-517.96	-514.42	-515.07	-513.22	**-512.86**	-513.54	**-512.86**

Table 3 Average predictive likelihood for models learned over nine datasets: Bayesian networks (DAG), ALDAGs with the approach of Leonelli and Varando (2022) (LV), ALDAGs with a fixed order (CMI), ALDAGs using compatible orders (ORD1, ORD2 and ORD3), ALDAGs considering all orders (ALL).

2, 3 learned without a fixed order, have more non-total edge labels, thus highlighting the need to learn more flexible models embedding asymmetric independences.

4.2 Simulation study

We next perform a simulation study to evaluate the proposed approach's feasibility and demonstrate its superiority to standard BN algorithms under the assumption that the true model is an ALDAG. We simulate data from randomly generated ALDAGs with different degrees of complexity: number of variables ($p \in \{4, 5, 6\}$); number of parents per variable ($k \in \{1, 2, 3\}$); number of stages per variable in the associated staged tree ($t \in \{2, 3, 4\}$); sample size ($N \in \{250, 500, 1000, 3000, 5000, 10000\}$). For each parameters' combination, we perform 20 repetitions of the experiment each time randomly shuffling the order of the variables to eliminate any possible bias of the search heuristics. The estimation procedures are the same as those of the previous section.

We first investigate whether the algorithms can retrieve the correct ordering of the variables. This is often of interest if the relationship between the variables is assumed to be causal (Pearl, 2009). Figure 5 reports the Kendall tau distance between the variable orderings as a function of the sample size N for each combination of the number of variables p and number of parents k. The Kendall tau distance is computed between the true order of the simulated ALDAG and the estimated order with the implementation in the `PerMallows` R package (Irurozki, Calvo, & Lozano, 2016). For

data	k	DAG	LV	CMI	ORD1	ORD2	ORD3	ALL
asia	1	0.02	0.07	0.16	34.72	169.97	1929.00	
	2	0.02	0.02	0.25	64.54	320.61	4530.72	
	3	0.02	0.03	0.36	102.28	526.64	7739.50	
cachexia	1	0.00	0.01	0.03	0.44	151.39	19.50	
	2	0.00	0.03	0.08	0.95	390.86	50.76	
	3	0.00	0.02	1.00	7.03	3237.02	376.25	
chds	1	0.01	0.00	0.00	0.02	0.14	0.06	0.12
	2	0.00	0.00	0.02	0.03	0.24	0.07	0.22
	3	0.00	0.01	0.02	0.06	0.50	0.18	0.47
coronary	1	0.00	0.02	0.01	0.03	0.50	0.58	14.58
	2	0.02	0.02	0.05	0.08	1.19	1.39	34.48
	3	0.00	0.01	0.07	0.08	1.72	2.00	52.14
fall	1	0.16	0.22	0.14	0.32	4.53	4.39	4.32
	2	0.14	0.20	0.17	0.35	5.82	5.65	5.70
	3	0.14	0.22	0.06	0.58	6.03	5.94	6.10
ksl	1	0.00	0.03	0.03	10.33	10.11	226.57	
	2	0.01	0.02	0.06	11.33	10.88	442.58	
	3	0.00	0.02	0.12	12.80	12.15	751.77	
mathmarks	1	0.00	0.01	0.00	0.02	1.32	0.42	1.25
	2	0.00	0.00	0.03	0.05	4.42	1.50	4.31
	3	0.00	0.02	0.49	0.44	54.95	18.20	54.46
phd	1	0.00	0.02	0.02	0.23	0.92	3.53	10.58
	2	0.01	0.02	0.04	0.47	1.78	7.00	23.11
	3	0.00	0.01	0.11	1.30	4.59	17.28	68.21
titanic	1	0.00	0.00	0.02	0.01	0.19	0.17	0.19
	2	0.00	0.02	0.02	0.03	0.45	0.41	0.43
	3	0.00	0.01	0.05	0.06	0.95	0.97	0.93

Table 4 Time (in seconds) to learn the models reported in Table 2. Computations carried out with an 11th Gen Intel(R) Core(TM) i7-1165G7 2.80GHz.

data	k	DAG	LV	CMI	ORD1	ORD2	ORD3	ALL
asia	1	(6, 0)	(6, 0)	(5, 0)	(5, 0)	(5, 0)	(6, 0)	
	2	(7, 0)	(3, 4)	(2, 6)	(2, 6)	(2, 6)	(1, 8)	
	3	(7, 0)	(3, 4)	(0, 12)	(0, 12)	(0, 12)	(0, 12)	
cachexia	1	(6, 0)	(4, 2)	(3, 3)	(4, 2)	(2, 4)	(4, 2)	
	2	(6, 0)	(4, 2)	(1, 10)	(1, 10)	(0, 11)	(0, 11)	
	3	(6, 0)	(4, 2)	(1, 14)	(1, 14)	(0, 15)	(0, 15)	
chds	1	(3, 0)	(2, 1)	(2, 1)	(2, 1)	(1, 2)	(1, 1)	(1, 2)
	2	(3, 0)	(2, 1)	(1, 3)	(2, 1)	(1, 2)	(1, 2)	(1, 2)
	3	(3, 0)	(2, 1)	(1, 5)	(2, 3)	(0, 4)	(0, 4)	(0, 4)
coronary	1	(5, 0)	(5, 0)	(5, 0)	(5, 0)	(5, 0)	(4, 0)	(5, 0)
	2	(8, 0)	(3, 5)	(2, 7)	(2, 7)	(2, 7)	(1, 7)	(1, 8)
	3	(8, 0)	(4, 4)	(2, 10)	(2, 9)	(2, 9)	(1, 10)	(2, 9)
fall	1	(3, 0)	(2, 1)	(2, 1)	(2, 1)	(2, 1)	(2, 1)	(2, 1)
	2	(5, 0)	(0, 5)	(1, 4)	(0, 5)	(0, 5)	(0, 5)	(0, 5)
	3	(5, 0)	(0, 5)	(1, 5)	(0, 6)	(0, 6)	(0, 6)	(0, 6)
ksl	1	(8, 0)	(8, 0)	(8, 0)	(7, 0)	(7, 0)	(7, 0)	
	2	(12, 0)	(3, 9)	(2, 12)	(2, 11)	(2, 11)	(1, 13)	
	3	(12, 0)	(3, 9)	(2, 15)	(0, 18)	(0, 18)	(0, 18)	
mathmarks	1	(4, 0)	(0, 4)	(0, 4)	(0, 4)	(0, 4)	(1, 3)	(0, 4)
	2	(4, 0)	(0, 4)	(0, 7)	(0, 7)	(0, 7)	(0, 7)	(0, 7)
	3	(4, 0)	(0, 4)	(0, 9)	(0, 9)	(0, 9)	(0, 9)	(0, 9)
phd	1	(5, 0)	(4, 1)	(3, 2)	(3, 2)	(3, 2)	(4, 1)	(3, 2)
	2	(6, 0)	(3, 3)	(3, 4)	(3, 6)	(3, 6)	(2, 6)	(3, 6)
	3	(6, 0)	(3, 3)	(2, 9)	(2, 9)	(2, 9)	(1, 10)	(2, 7)
titanic	1	(3, 0)	(1, 2)	(1, 2)	(1, 2)	(1, 2)	(1, 2)	(1, 2)
	2	(5, 0)	(0, 5)	(0, 5)	(0, 5)	(0, 5)	(0, 5)	(0, 5)
	3	(5, 0)	(0, 5)	(0, 6)	(0, 6)	(0, 6)	(0, 6)	(0, 6)

Table 5 Number of edges with label total (left number) and non-total label (right number) for the learned models in Table 2.

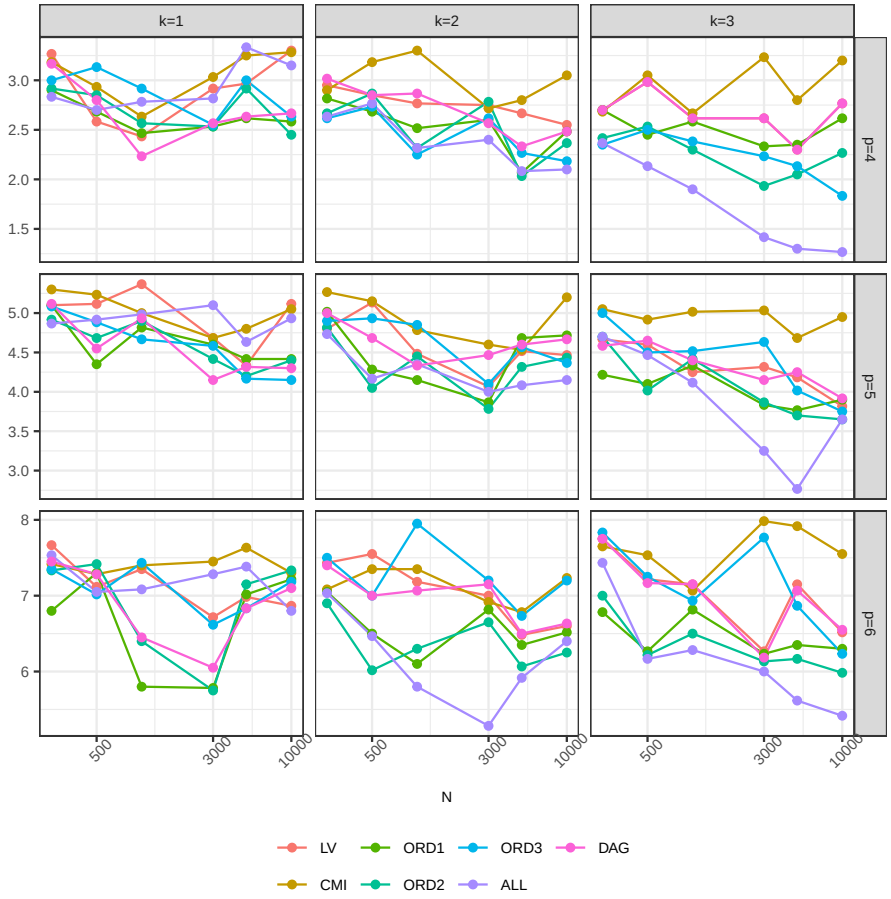

Fig. 5 Kendall tau distance between the estimated and true model in the simulation experiment.

the case $k = 1$, we can see almost no difference between the approaches in retrieving the correct ordering of the variables. For $k = 3$, the algorithm using every possible order outperforms the others as expected, but ORD2 and ORD3 give better results than the standard DAG approach. Not surprisingly, the CMI algorithm, which takes an arbitrary ordering of the variables, performs worse.

Figure 6 reports the average time required for estimating the models as a function of time. As expected, the DAG approach is the fastest but with a time comparable to the fixed-order approach. The algorithm investigating every possible order is the slowest: however, on average, it takes only ten seconds for six variables. Notice that the number of parents k does not significantly affect the computational times.

The average BIC for every combination of the number of variables (p), number of parents (k), and sample size (N) was also computed for every estimating procedure (not reported here for space). It showed that ALDAG routines without a fixed order provide a better fit than the standard DAG algorithm as the sample size increases.

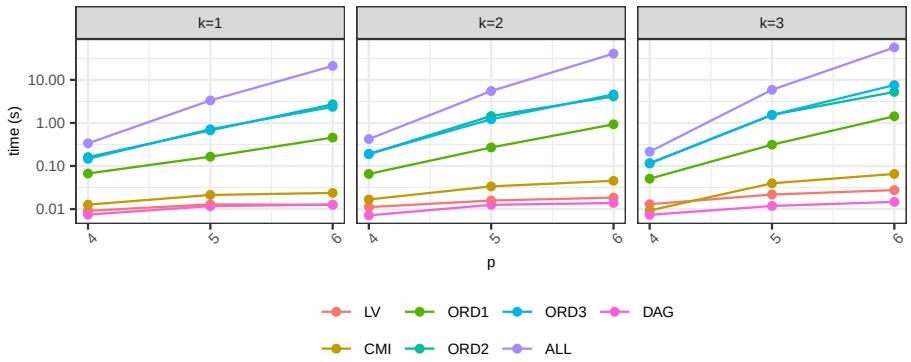

Fig. 6 Computational time (in seconds) as a function of the number of variables p for the simulation experiment.

The difference in fit also increases with p and k, as seen in the computational study reported above.

5 Data application: The fear of COVID-19 scale

The COVID-19 pandemic shook the world and changed the habits of most individuals due to months-long lockdowns of educational and nonessential business activities. An event of this scale and criticality was unprecedented in the lifespan of all citizens worldwide. Many aspects of the COVID-19 pandemic, such as uncertainty over patient outcomes, familiarity with infected people, and mandatory change of habits, have led many individuals across the globe to experience a generalized sense of fear (Y. Huang & Zhao, 2020). For previous epidemics, fear has been shown to be associated with other psychological disorders such as anxiety and depression (Ford et al., 2019), and these have also been shown to be strongly related to social isolation (Santini et al., 2020), which in the case of the COVID-19 pandemic has been imposed by social distancing governmental policies.

To quantify and measure the fear of individuals about the COVID-19 pandemic, the Fear of COVID-19 Scale was developed in Ahorsu et al. (2022), consisting of the following seven items:

- I am most afraid of COVID-19 (Fear).
- It makes me uncomfortable to think about COVID-19 (Think).
- My hands become clammy when I think about COVID-19 (Hands).
- I fear losing my life because of COVID-19 (Life).
- I become nervous or anxious when watching news and stories about COVID-19 on social media (Media).
- I cannot sleep because I am worried about getting COVID-19 (Sleep).
- My heart races or palpitates when I think about getting COVID-19 (Heart).

Participants indicate their level of agreement with the statements using a five-item Likert-type scale. Answers included “strongly disagree,” “disagree,” “neither agree

	Train BIC				Test Likelihood			
k	DAG	LV	CMI	ORD1	DAG	LV	CMI	ORD1
1	2961.27	2947.25	3013.99	2946.27	-156.77	-141.22	-142.73	-140.85
2	2961.27	2947.25	2938.32	2867.54	-156.77	-141.22	-135.69	-136.90
3	2961.27	2947.25	2913.63	**2810.43**	-156.77	-141.22	-135.72	**-134.24**

Table 6 BIC of the models learned using the Fear of COVID-19 Scale data. 2.

nor disagree," "agree," and "strongly agree". The minimum score possible for each question is 1, and the maximum is 5. A total score is calculated by adding up each item score (ranging from 7 to 35). The higher the score, the greater the fear of COVID-19. Because of the generalized sense of fear associated with the COVID-19 pandemic, the scale has been used and validated in many other countries, including China (Yang et al., 2022), Italy (Soraci et al., 2022), and Spain (Espejo & Checa, 2021).

Data from the Italian validation of the Fear of COVID-19 Scale is available from Soraci (2020) and consists of the answers of 149 individuals to the seven items of the scale and information about their age and gender. Age was dichotomized using the equal-frequency method, and the answer to the items of the scale were transformed into ternary variables: disagree (including strongly disagree and disagree), neither (neither agree nor disagree), and agree (including agree and strongly agree).

The analysis aims to understand the effect of the two demographic factors on the answers to the fear scale and the relationship between the scale items. To this end, DAG and ALDAG models are learned from the data using various methodologies: DAGs using the tabu algorithm; ALDAGs with the routine of Leonelli and Varando (2022); ALDAGs with a fixed order as in Section 3.3; ALDAGs considering all compatible orders to an initial BN. The other approaches are not considered here because with a total of nine variables, there would be around 362880 orders to consider, and thus, it would be computationally unfeasible. Notice, however, that the previous section's experiments showcased that the procedure of learning ALDAGs using all compatible orders to an initial BN is competitive in terms of fit to more nuanced approaches. Each of the considered algorithms is evaluated with $k = 1, 2, 3$ number of parents. Furthermore, for all algorithms, only outputs where age and gender have no items' parents are considered since our interest is in the effect of the demographic variables on how individuals responded.

Table 6 reports the average train BIC and predictive likelihood in a 10-fold cross-validation. It can be noticed that all DAGs are such that any vertex has at most one parent since all BICs are the same. Consequently, all ALDAGs learned with the approach of Leonelli and Varando (2022) are such that any vertex has at most one parent. The best scoring model is the one with three parents, and the order chosen from those compatible with a starting BN learned with the tabu algorithm. Figure 7 shows the learned BN. It can be seen that, indeed, every variable has at most one parent. Importantly, it shows that the two demographic variables do not affect how individuals responded to the items on the scale.

The best-scoring ALDAG reported in Figure 8 reveals that the strict independence assumption between the demographic variables and the items of the scale is

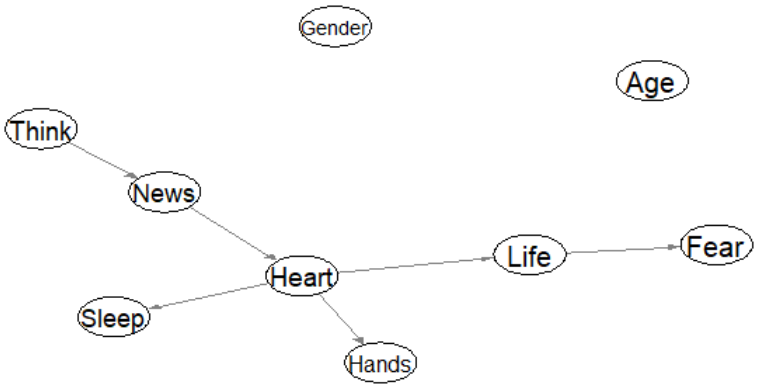

Fig. 7 BN learned with the tabu algorithm using the Fear of COVID-19 Scale data.

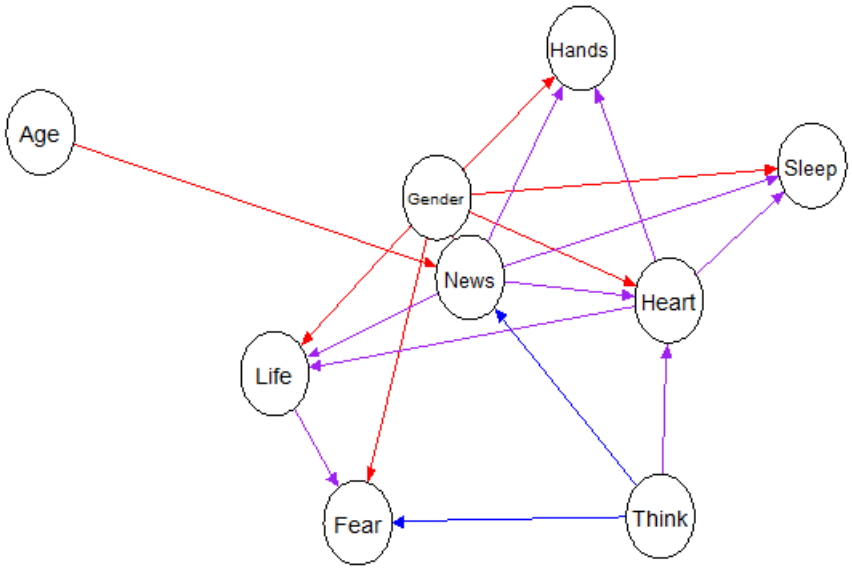

Fig. 8 Best scoring ALDAG for the Fear of COVID-19 Scale data. The edge coloring is: red - context; blue - partial; violet - context/partial; green - local; black - total.

not tenable. Age has a direct effect on the item News, whilst Gender has a direct effect on all items except for Think. Furthermore, the model shows a much wider dependence between the items of the scale since it includes all edges of the DAG in Figure 7 (although with a different label) as well as additional ones. Interestingly, no edges were given a label total; there are seven edges context, two partial, and eight context/partial.

Symmetric conditional independences can be read from the ALDAG as in standard BNs. For instance, the network implies that given Gender, Life and Think, Age and the remaining items do not affect how individuals answer the Fear item. In other words, the Fear item, which provides an overview of an individual's fear level, is only directly affected by gender, and by how the Life and Think items were answered.

The labeling of the edges provides further information about the relationship between a variable and its parents. In Varando et al. (2024), the so-called *dependence subtree* was introduced to retrieve the actual dependence structure between a variable and its parents from the ALDAG. This is shown in Figure 9 for the variable News, which is directly affected by the demographic variables and the Think item. The coloring of the vertices associated with the News variable ($v_9 - v_{20}$) represents asymmetric conditional independence as for staged trees. The coloring shows that for individuals who answered agree or disagree in the Think item, gender, and age do not affect how they answered News. It also holds that gender is independent of News for the adult group who answered Think = neither.

Next, we analyze the Life item, perhaps the most critical of the seven items. From standard conditional independence for DAGs, we can conclude that given gender, News, and Heart, Life is independent of Age, Think, Hands, and Sleep. The dependence subtree in Figure 10 sheds light on the dependence structure between Life and its parents. Compared to the one in Figure 9, we can see a slightly less regular structure embedding various asymmetric independences. For instance, for males who answered News = neither, Hearth and Life are independent. Similarly, Life is independent of gender, given Hearth and News = disagree.

Such complex patterns of dependence cannot be modeled using BNs, but they provide valuable information about the dependence structure of the data. The goodness of the ALDAG is evidenced by the BIC and predictive likelihood of the model, which are much lower than those of the BN model.

6 Discussion

ALDAGs are an expressive extension of standard DAGs that carry information about additional equalities in the model's conditional distributions. In this paper, we have introduced efficient learning algorithms for ALDAGs returning interpretable, sparse models, which can be embellished by small-dimensional staged trees. The computational study and the real-world data application showcase the power of ALDAGs and our learning algorithms in untangling complex, asymmetric dependence structures.

One of the main difficulties in learning ALDAGs from data is the size of the model search space as technically formalized in Propositions 1 and 2. For this reason, in order to explore the space of ALDAGs efficiently, modern optimization algorithms must be employed in the case of a large number of variables. In the case of DAGs, stochastic and swarm optimization, and genetic algorithms have been successfully used (Gheisari & Meybodi, 2016; Larrañaga, Karshenas, Bielza, & Santana, 2013; Ren, Wang, Li, Pang, & Wei, 2022). We have started investigating the use of such algorithms in the setup of ALDAGs.

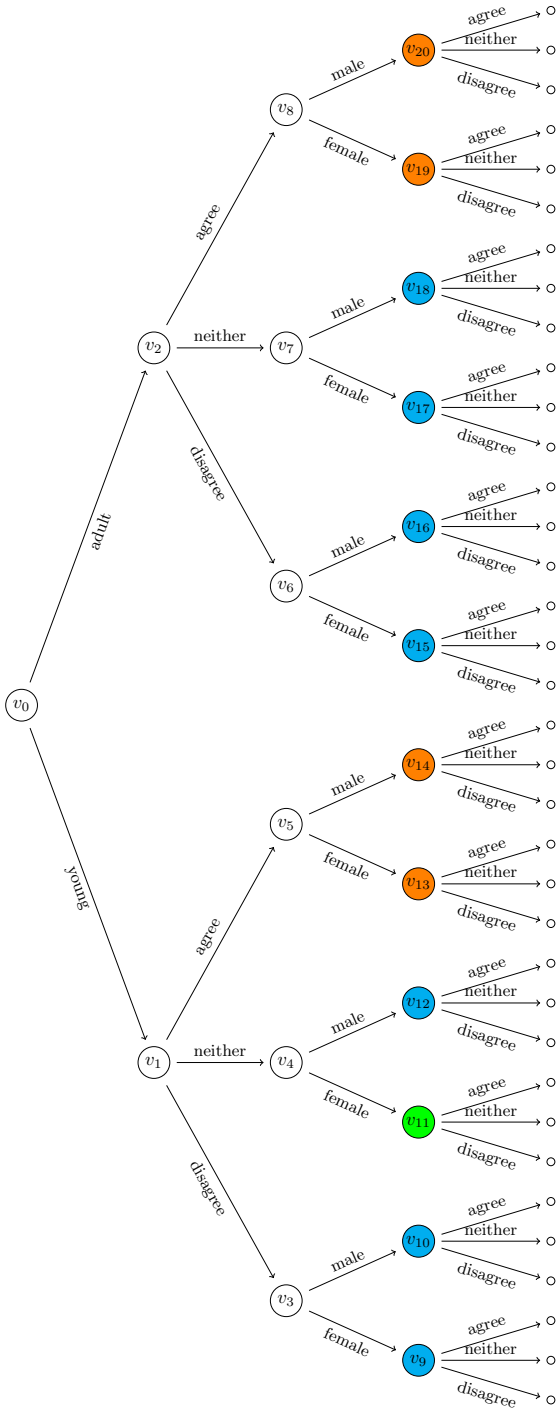

Fig. 9 Dependence subtree for the variable News with parents (Age, Think, Gender).

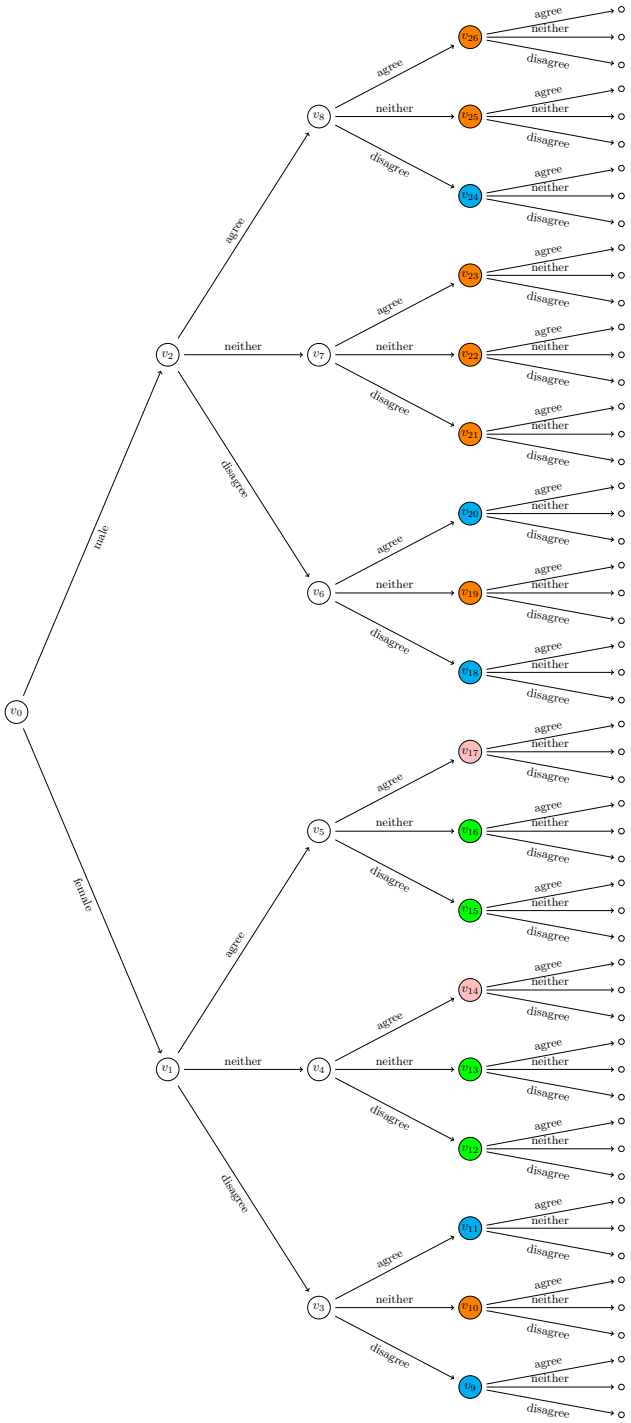

Fig. 10 Dependence subtree for the variable Life with parents (Gender, News, Hearth).

A promising line of research is extending ALDAGs to carry out casual information about the variables' relationships. Causal discovery using ALDAGs would provide more detailed information about causal effects than DAGs. We have started investigating this topic in Leonelli and Varando (2023), but a full characterization of causal ALDAGs is still missing.

A different avenue to impose sparsity in the learned model would be to take a Bayesian learning approach. Then, the prior distribution over the space of all possible ALDAGs could penalize more complex DAG structures in favor of simpler ones. We are currently investigating the implementation of such an approach and the use of different prior distributions, which are planned to be reported in the near future.

ALDAGs have now been applied in a variety of domain, including psychology (in this article), large national surveys (Varando et al., 2024), and medicine (Filigheddu et al., 2024). However, just as for DAGs, they can find applications in basically every area of research. One possible application we plan to explore in the future is medical image classification (Srivastava & Aggarwal, 2018; Srivastava, Singhal, & Aggarawal, 2017), where DAGs have already been used (Arias, Martinez-Gomez, Gamez, De Herrera, & Müller, 2016).

Competing interests

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Authors contribution statement

All authors contributed to the study conception and design. The first draft of the manuscript was written by M.L. and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Ethical and informed consent for data used

Not applicable

Data availability and access

The data used is available from Soraci (2020).

References

Ahorsu, D.K., Lin, C.-Y., Imani, V., Saffari, M., Griffiths, M.D., Pakpour, A.H. (2022). The fear of COVID-19 scale: Development and initial validation. *International Journal of Mental Health & Addiction*, *20*(3), 1537–1545,

Aragam, B., & Zhou, Q. (2015). Concave penalized estimation of sparse Gaussian Bayesian networks. *The Journal of Machine Learning Research*, *16*(1), 2273–2328,

Arias, J., Martinez-Gomez, J., Gamez, J.A., De Herrera, A.G.S., Müller, H. (2016). Medical image modality classification using discrete Bayesian networks. *Computer Vision and Image Understanding*, *151*, 61–71,

Barclay, L.M., Hutton, J.L., Smith, J.Q. (2013). Refining a Bayesian network using a chain event graph. *International Journal of Approximate Reasoning*, *54*(9), 1300–1309,

Cao, X., & Yang, F. (2021). On the non-local priors for sparsity selection in high-dimensional Gaussian DAG models. *Statistical Theory and Related Fields*, *5*(4), 332–345,

Carli, F., Leonelli, M., Riccomagno, E., Varando, G. (2022). The R package stagedtrees for structural learning of stratified staged trees. *Journal of Statistical Software*, *102*, 1–30,

Carli, F., Leonelli, M., Varando, G. (2023). A new class of generative classifiers based on staged tree models. *Knowledge-Based Systems*, *268*, 110488,

Collazo, R.A., Görgen, C., Smith, J.Q. (2018). *Chain event graphs*. CRC Press.

Collazo, R.A., & Smith, J.Q. (2016). A new family of non-local priors for chain event graph model selection. *Bayesian Analysis*, *11*(4), 1165–1201,

Corander, J., Hyttinen, A., Kontinen, J., Pensar, J., Väänänen, J. (2019). A logical approach to context-specific independence. *Annals of Pure and Applied Logic*, *170*(9), 975–992,

Cowell, R., & Smith, J. (2014). Causal discovery through MAP selection of stratified chain event graphs. *Electronic Journal of Statistics*, *8*(1), 965–997,

Espejo, B., & Checa, I. (2021). The fear of COVID-19 scale (FCV-19S) in Spain: Adaptation and confirmatory evidence of construct and concurrent validity. *Mathematics*, *9*(19), 2512,

Filigheddu, M.T., Leonelli, M., Varando, G., Gómez-Bermejo, M.Á., Ventura-Díaz, S., Gorospe, L., Fortún, J. (2024). Using staged tree models for health data: Investigating invasive fungal infections by Aspergillus and other filamentous fungi.

Computational and Structural Biotechnology Journal, *24*, 12–22,

Ford, B.N., Yolken, R.H., Dickerson, F.B., Teague, T.K., Irwin, M.R., Paulus, M.P., Savitz, J. (2019). Reduced immunity to measles in adults with major depressive disorder. *Psychological Medicine*, *49*(2), 243–249,

Freeman, G., & Smith, J.Q. (2011). Bayesian MAP model selection of chain event graphs. *Journal of Multivariate Analysis*, *102*(7), 1152–1165,

Gheisari, S., & Meybodi, M.R. (2016). Bnc-pso: Structure learning of Bayesian networks by particle swarm optimization. *Information Sciences*, *348*, 272–289,

Görgen, C., Leonelli, M., Marigliano, O. (2022). The curved exponential family of a staged tree. *Electronic Journal of Statistics*, *16*(1), 2607–2620,

Hagras, H. (2018). Toward human-understandable, explainable AI. *Computer*, *51*(9), 28–36,

Huang, X., Guo, X., Li, Y., Yu, K. (2023). A novel data enhancement approach to DAG learning with small data samples. *Applied Intelligence*, 1–19,

Huang, Y., & Zhao, N. (2020). Generalized anxiety disorder, depressive symptoms and sleep quality during COVID-19 outbreak in China: A web-based cross-sectional survey. *Psychiatry Research*, *288*, 112954,

Irurozki, E., Calvo, B., Lozano, J.A. (2016). PerMallows: An R package for Mallows and generalized Mallows models. *Journal of Statistical Software*, *71*, 1–30,

Kuipers, J., Suter, P., Moffa, G. (2022). Efficient sampling and structure learning of Bayesian networks. *Journal of Computational and Graphical Statistics*, *31*(3), 639–650,

Larrañaga, P., Karshenas, H., Bielza, C., Santana, R. (2013). A review on evolutionary algorithms in bayesian network learning and inference tasks. *Information Sciences*, *233*, 109–125,

Leonelli, M., & Varando, G. (2022). Highly efficient structural learning of sparse staged trees. *International Conference on Probabilistic Graphical Models* (pp. 193–204).

Leonelli, M., & Varando, G. (2023). Context-specific causal discovery for categorical data using staged trees. *International Conference on Artificial Intelligence and Statistics* (pp. 8871–8888).

Li, Y., Liu, D., Chu, J., Zhu, Y., Liu, J., Cheng, X. (2020). A sparse Bayesian learning method for structural equation model-based gene regulatory network inference. *IEEE Access*, *8*, 40067–40080,

Linardatos, P., Papastefanopoulos, V., Kotsiantis, S. (2020). Explainable AI: A review of machine learning interpretability methods. *Entropy*, *23*(1), 18,

Liu, X., Gao, X., Ru, X., Tan, X., Wang, Z. (2023). Improving greedy local search methods by switching the search space. *Applied Intelligence*, 1–18,

Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., ... Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, *2*(1), 56–67,

Luo, G., Zhao, B., Du, S. (2019). Causal inference and Bayesian network structure learning from nominal data. *Applied Intelligence*, *49*, 253–264,

Meyer, P.E., & Meyer, M.P.E. (2009). Package 'infotheo'. *R Package, Version 1*, ,

Nicolussi, F., & Cazzaro, M. (2021). Context-specific independencies in stratified chain regression graphical models. *Bernoulli*, *27*(3), 2091 – 2116,

Pearl, J. (2009). *Causality.* Cambridge University Press.

Pensar, J., Nyman, H., Corander, J. (2017). Structure learning of contextual Markov networks using marginal pseudo-likelihood. *Scandinavian Journal of Statistics*, *44*(2), 455–479,

Pensar, J., Nyman, H., Koski, T., Corander, J. (2015). Labeled directed acyclic graphs: A generalization of context-specific independence in directed graphical models. *Data Mining and Knowledge Discovery*, *29*, 503–533,

Pensar, J., Nyman, H., Lintusaari, J., Corander, J. (2016). The role of local partial independence in learning of Bayesian networks. *International Journal of Approximate Reasoning*, *69*, 91–105,

Ren, Y., Wang, L., Li, X., Pang, M., Wei, J. (2022). Stochastic optimization for Bayesian network classifiers. *Applied Intelligence*, *52*(13), 15496–15516,

Santini, Z.I., Jose, P.E., Cornwell, E.Y., Koyanagi, A., Nielsen, L., Hinrichsen, C., ... Koushede, V. (2020). Social disconnectedness, perceived isolation, and symptoms of depression and anxiety among older americans (NSHAP): A longitudinal mediation analysis. *The Lancet Public Health*, *5*(1), e62–e70,

Scutari, M. (2010). Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, *35*, 1–22,

Scutari, M., & Denis, J.-B. (2021). *Bayesian networks: With examples in R*. CRC press.

Shajoonnezhad, N., & Nikanjam, A. (2023). A stochastic variance-reduced coordinate descent algorithm for learning sparse Bayesian network from discrete high-dimensional data. *International Journal of Machine Learning and Cybernetics*, *14*(3), 947–958,

Smith, J.Q., & Anderson, P.E. (2008). Conditional independence and chain event graphs. *Artificial Intelligence*, *172*(1), 42–68,

Soraci, P. (2020). Validation and psychometric evaluation of the Italian version of the fear of COVID-19 scale. *UniData - Bicocca Data Archive, Milan. Study Number SN225, Data file version 1.0*, ,

Soraci, P., Ferrari, A., Abbiati, F.A., Del Fante, E., De Pace, R., Urso, A., Griffiths, M.D. (2022). Validation and psychometric evaluation of the Italian version of the fear of COVID-19 scale. *International Journal of Mental Health and Addiction*, *20*, 1913—1922,

Srivastava, A., & Aggarwal, A.K. (2018). Medical image fusion in spatial and transform domain: A comparative analysis. *Handbook of Research on Advanced*

Concepts in Real-Time Image and Video Processing (pp. 281–300). IGI global.

Srivastava, A., Singhal, E., Aggarawal, A. (2017). Comparative analysis of multimodal medical image fusion using PCA and wavelet transforms. *International Journal of Latest Technology in Engineering, Management & Applied Science*, *VI*(VI), 115–118,

Sucar, L. (2021). *Probabilistic graphical models: Principles and applications*. Springer.

Talvitie, T., Eggeling, R., Koivisto, M. (2019). Learning Bayesian networks with local structure, mixed variables, and exact algorithms. *International Journal of Approximate Reasoning*, *115*, 69–95,

Tsagris, M. (2021). A new scalable Bayesian network learning algorithm with applications to economics. *Computational Economics*, *57*(1), 341–367,

Tsamardinos, I., Brown, L.E., Aliferis, C.F. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, *65*, 31–78,

Varando, G., Carli, F., Leonelli, M. (2024). Staged trees and asymmetry-labeled DAGs. *Metrika (to appear)*, ,

Vowels, M.J., Camgoz, N.C., Bowden, R. (2022). D'ya like dags? A survey on structure learning and causal discovery. *ACM Computing Surveys*, *55*(4), 1–36,

Wang, M., & Allen, G.I. (2023). Thresholded graphical lasso adjusts for latent variables. *Biometrika*, *110*(3), 681–697,

Yang, W., Li, P., Huang, Y., Yang, X., Mu, W., Jing, W., ... Zhang, X. (2022). Cross-cultural adaptation and validation of the fear of COVID-19 scale for Chinese university students: A cross-sectional study. *International Journal of Environmental Research and Public Health*, *19*(14), 8624,

A Algorithm to learn k-parents staged trees

Algorithm 1: Learning algorithm for k-parent staged trees using CMI

Input : A dataset $\mathcal{D}$ over categorical variables $X_1, \ldots, X_p$ and $k \in \mathbb{Z}_+$
Output: A staged tree T

for $i \leftarrow 1$ **to** p **do**
  $\Pi_i \leftarrow \emptyset$;
  if $i \leq k+1$ **then**
    $\Pi_i \leftarrow [i-1]$;
  else
    for $j \leftarrow 1$ **to** k **do**
      $max \leftarrow -\infty$;
      for $s \in [i-1] \setminus \Pi_i$ **do**
        if $I(X_i, X_j | X_{\Pi_i}) > max$ **then**
          $new \leftarrow s$;
        end
      end
      $\Pi_i \leftarrow \Pi_i \cup \{s\}$;
    end
  end
end
Construct G using $[p]$ and $\Pi_1, \ldots, \Pi_p$;
Transform G to its equivalent staged tree T with staging $U_1, \ldots, U_{p-1}$;
$score \leftarrow BIC(T)$; $T^* \leftarrow T$;
for $i \leftarrow 1$ **to** $p-1$ **do**
  $indicator \leftarrow 1$;
  while $indicator \neq 0$ **do**
    for *every pair of stages* $u_j, u_s \in U_i$ **do**
      construct $T^{'}$ by merging u_i and u_j;
      if $BIC(T^{'}) < BIC(T^*)$ **then**
        $score \leftarrow BIC(T^{'})$; $T^* \leftarrow T^{'}$;
      end
    end
    if $T = T^*$ **then**
      $indicator \leftarrow 0$
    else
      $T \leftarrow T^*$
    end
  end
end
return T