Expert Systems With Applications 238 (2024) 121831

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Local causal structure learning with missing data

Shaojing Sheng [a,b], Xianjie Guo [a,b], Kui Yu [a,b], Xindong Wu [a,c,*]

[a] Key Laboratory of Knowledge Engineering with Big Data (the Ministry of Education of China), Hefei University of Technology, Hefei, 230009, China
[b] School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, 230009, China
[c] Research Center for Knowledge Engineering, Zhejiang Lab, Hangzhou, 311121, China

ARTICLE INFO

Keywords:
Bayesian network
Local causal structure learning
Missing data

ABSTRACT

Local causal structure learning aims to discover and distinguish the direct causes and direct effects of a target variable. However, the state-of-the-art algorithms for local causal structure learning fail to perform well when dealing with missing data. The general approach is to fill in the missing data using imputation techniques before learning the local causal structure, but this method suffers from problems such as low accuracy, low efficiency, and instability. To address these issues, we propose a novel method for local causal structure learning with missing data, named misLCS. Firstly, we design an iterative data imputation method to obtain the complete and correct data from the missing data. Then, misLCS adopts a data subset strategy to get a data subset that variables are closely related to the target variable. Thirdly, within this data subset, misLCS constructs the local causal skeleton of the target variable using a mutual information-based feature selection method and orients the direction of edges using conditional independence tests and Meek rules. Finally, misLCS updates the missing data in preparation for the next iteration. This procedure continues until the direct causes and direct effects of the target variable have been identified. Our experiments on seven benchmark Bayesian networks and a real-world bioinformatics dataset, with a number of variables from 11 to 801, demonstrate that our algorithm achieves better accuracy than the existing local causal structure learning algorithms.

1. Introduction

Discovering causal relationships among a set of random variables is a significant objective in various scientific fields, including medicine (Sokolova et al., 2015; Yang et al., 2023), bioinformatics (Foraita et al., 2020; Triantafillou et al., 2017) and computer science (Khan et al., 2018; Nogueira et al., 2021). The identification of these relationships not only reveals the underlying data generation mechanism but also improves classification and prediction performance. As a result, extensive research (Cai et al., 2018; Yu et al., 2019; Zheng et al., 2018) has been devoted to learning causal relationships among variables. However, in real-life scenarios, many algorithms struggle to learn causal relationships when encountering missing data, or even fail to perform on incomplete data.

Learning a Bayesian network (BN) structure from observational data is a popular method to represent causal relationships. A BN structure often takes the form of a directed acyclic graph (DAG) in which nodes denote variables and edges denote dependence between variables. In a causal DAG (e.g. in Fig. 1), for example, the existence of a direct edge: $A \rightarrow T$ means that A is a direct cause of T, T is a direct effect of A. Therefore, learning the BN structure and determining the directions of edges are crucial components of causal structure learning.

Lots of causal structure learning methods (Cai et al., 2022) have been developed over the past few decades. It can be roughly categorized into two types based on their learning scales: global causal structure learning and local causal structure learning. The first type of methods, such as MMHC (Max–Min Hill-Climbing) (Tsamardinos et al., 2006), BCSL (Bootstrap sampling based Causal Structure Learning) (Guo et al., 2022), and ADL (Adaptive DAG Learning) (Guo et al., 2023) aims to learn the causal structure of all variables, and is an NP-hard problem. When users are only interested in causal relationships around a given variable and the number of variables is massive, learning global causal structure is costly and time-consuming, even unrealistic. Local causal structure learning algorithms, such as CMB (Causal Markov Blanket) (Gao & Ji, 2015), PCD-by-PCD (PCD means Parent, Children, Descendants) (Yin et al., 2008), ELCS (Efficient Local Causal Structure) (Yang et al., 2021), etc., are been proposed subsequently.

For example, in bioinformatics (Saeys et al., 2007), with a more than 10,000 gene expression dataset associated with a specified disease,

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: jssheng@mail.hfut.edu.cn (S. Sheng), xianjieguo@mail.hfut.edu.cn (X. Guo), yukui@hfut.edu.cn (K. Yu), xwu@hfut.edu.cn (X. Wu).

https://doi.org/10.1016/j.eswa.2023.121831
Received 31 May 2023; Received in revised form 23 September 2023; Accepted 24 September 2023
Available online 29 September 2023

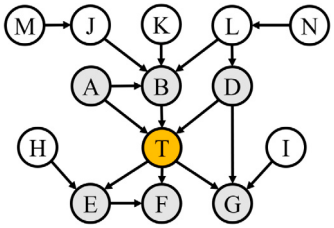


Fig. 1. A causal Bayesian network, the PC of the target variable T contains A, B, D, E, F, and G, which are marked in gray.

researchers are only interested in the genes that cause that disease, and not interested in the entire causal structure including all genes. Local causal structure learning methods allow us to identify the direct causes (parents) and direct effects (children) of any variables of our interest without learning an entire causal structure including all variables in an efficient way, especially with high dimensional datasets. In addition, directly learning the local causal relationships of a target variable provides an efficient way to estimate the causal effect of two variables by learning a local causal structure around the two variables without learning the entire causal structure (Geng et al., 2019).

However, all the pioneering algorithms, such as NOTEAR (Zheng et al., 2018), ELCS (Yang et al., 2021), etc. are merely applied to complete data, learning local causal structure with missing data has not been studied fully. To tackle these challenges, in this paper, we propose a novel algorithm – misLCS (Local Causal Structure learning with missing data) to learn local causal structures with missing data. Firstly, misLCS uses an iterative data imputation method to obtain complete and correct data. Then, it employs a data subset strategy to get a subset of variables that are closely related to the target variable. Finally, it calculates the conditional independence among the obtained set of variables to determine the direction of edges. The remaining undirected edge directions are oriented using Meek rules. The main contributions are summarized as follows.

- We propose the first algorithm to learn local causal structures with missing data, which can successfully discover and distinguish the direct causal and direct effect of the target variable.
- We have conducted extensive experiments on seven benchmark BNs, and have compared the proposed misLCS with four state-of-the-art local causal structure learning algorithms to validate the feasibility and effectiveness.

The remainder of this paper is organized as follows. Section 2 reviews the related work, Section 3 provides the notations and definitions, Section 4 detailedly presents the proposed misLCS algorithm, Section 5 reports and discusses the experimental results, and Section 6 summarizes the paper.

2. Related work

Our work focuses on local causal structure learning with missing data and is related to MB (Markov Blanket) discovering and causal structure learning.

Given a target variable T, the MB discovering algorithm aims to learn parents, children, and spouse of T simultaneously, and it is an essential part in the skeleton learning during BN structure learning. The classic methods, such as HITON-MB (Aliferis et al., 2003), IAMB (Tsamardinos & Aliferis, 2003), MMMB (Tsamardinos et al., 2003a), and their variants (Tsamardinos et al., 2003b; Yaramakala & Margaritis, 2005), mainly employ independence tests to find the MB of a given variable.

Although these methods have achieved excellent performance, MB discovering does not distinguish the spouse of T from its PC (Parents and Children). Variables in MB are not causally interpretable and still perform poorly when making predictions or classifications. Thus, algorithms for learning and distinguishing the causal relationships of variables are subsequently proposed (Cai et al., 2022).

Causal structure learning. Causal structure learning aims at learning and distinguishing causal relationships among a set of random variables. According to the scale of learning, causal structure learning can be categorized into two types: global causal structure learning and local causal structure learning. The first type of method aims to learn the causal structure of all variables. The representative algorithms include GSMB (Margaritis & Thrun, 1999), MMHC (Tsamardinos et al., 2006), NOTEAR (Zheng et al., 2018), DAG-GNN (Yu et al., 2019), BCSL (Guo et al., 2022), ADL (Guo et al., 2023) etc. MMHC, for example, adopts a local-to-global approach, it constructs a skeleton of a DAG using the learnt MBs or PCs, and then orients the edge of the learnt skeleton using score-based or constraint-based causal learning algorithms. However, global causal structure learning algorithms are time-consuming or even infeasible when the number of variables of a BN is large. In fact, in many practical scenarios, we are only interested in distinguishing parents from children of a variable of interest. To improve efficiency, a series of methods of local causal structure learning subsequently designed. The representative algorithms include PCD-by-PCD (Yin et al., 2008), CMB (Gao & Ji, 2015), ELCS (Yang et al., 2021), etc. PCD-by-PCD first learns the PCD of a target variable, then learns the PCD of the variables connected to the target variable. Then, PCD-by-PCD finds the V-structure to orient edges until all parents and children of the target variable are identified.

Causal structure learning with missing data. The ubiquitous missing data problem becomes an obstacle to causal structure learning. There are two common ways to deal with missing data: CC analysis (Complete-Case analysis, also called list deletion) and TD deletion (Test-wise Deletion, also called pairwise deletion or available case analysis). The former completely removes the missing cases from the data completely, whereas the latter ignores only the case where the variable required for the current conditional independence test is missing. When the missing rate is high, adopting the above deletion methods make the sample significantly reduced. This may cause important information loss and significant bias in the data analysis or mining. Besides, various statistical techniques and machine learning based techniques (Lin & Tsai, 2020) are employed to perform missing value imputation and improve the accuracy. Some global causal structure learning algorithms, such as MVPC (Missing Value PC) (Tu et al., 2019) and MICD (Foraita et al., 2020), are combined data imputation methods to solve domain-specific problems. MICD (Foraita et al., 2020), for example, uses multiple imputation and constraint-based algorithms to learn the global causal structure within incomplete gene data.

3. Notations and definitions

In this section, some basic definitions and notations frequently used in this paper will be introduced (see Table 1 for a summary of the notation).

Definition 1 (*Conditional Independence (Pearl, 2014)*)**.** Given a conditioning set $\boldsymbol{Z}$, variable X is conditionally independent of Y iff $P(X \mid Y, \boldsymbol{Z}) = P(X \mid \boldsymbol{Z})$.

Definition 2 (*Bayesian Network (Pearl, 2014)*)**.** A Bayesian Network (BN) is represented by the triplet – $\langle \boldsymbol{U}, \mathbb{G}, \mathbb{P} \rangle$ which satisfies the **Markov Condition**: each variable in $\boldsymbol{U}$ is conditionally independent of variables in its non-descendant given its parents in $\mathbb{G}$.

Definition 3 (*Faithfulness (Spirtes et al., 2000)*)**.** For a BN – $\langle \boldsymbol{U}, \mathbb{G}, \mathbb{P} \rangle$, $\mathbb{G}$ is faithful to $\mathbb{P}$ if all the conditional independence appear in $\mathbb{P}$ are entailed by $\mathbb{G}$. $\mathbb{P}$ is faithful iff there is a DAG $\mathbb{G}$ such that $\mathbb{G}$ is faithful to $\mathbb{P}$.

Table 1
Summary of notations.

Notations	Meanings
$\boldsymbol{U}$	A set of random variables.
$\boldsymbol{V}$	A subset of $\boldsymbol{U}$.
$\mathbb{P}$	A joint probability distribution over $\boldsymbol{U}$.
$\mathbb{G}$	A direct acyclic graph over $\boldsymbol{U}$.
DAG	Direct acyclic graph.
$X \sim Z$	A single variable in $\boldsymbol{U}$.
$\boldsymbol{Z}$	A conditioning set within $\boldsymbol{U}$.
$X \perp\!\!\!\perp Y \mid \boldsymbol{Z}$	X and Y are independent given $\boldsymbol{Z}$.
$X \not\perp\!\!\!\perp Y \mid \boldsymbol{Z}$	X and Y are dependent given $\boldsymbol{Z}$.
$\boldsymbol{MB}_T$	Markov Blanket of T.
$\boldsymbol{PC}_T$	A set of parents and children of T.
$\boldsymbol{P}_T$	A set of parents of T.
$\boldsymbol{C}_T$	A set of children of T.
$\boldsymbol{UN}_T$	Undistinguished variables in $\boldsymbol{PC}_T$.
$\boldsymbol{SP}_T$	A set of spouses of T.
$\boldsymbol{SP}_T\{X\}$	A spouses of T with regard to T's child X.
$\boldsymbol{Sep}_T\{X\}$	A set that d-separates X from T.
$\boldsymbol{Sep}_T$	A set that contains the sets $\boldsymbol{Sep}_T\{\cdot\}$ of all variables.
$\boldsymbol{CSP}_T$	A set that contains the candidate spouse sets of all $\boldsymbol{PC}_T$ variables.
$subD\{T\}$	A complete data subset with regard to T.
$superPC$	$superPC$ contains the PC of the target variable and the PC of PC

From Definitions 2 and 3, we know that: **Markov Condition** enable us to recover $\mathbb{P}$ from known $\mathbb{G}$ (in the conditional independence), faithfulness enables us to recover $\mathbb{G}$ from $\mathbb{P}$ to fully describe $\mathbb{P}$.

Definition 4 (*Causal Bayesian Network (Pearl, 2009)*)**.** A BN is called Causal Bayesian Network (CBN) if a directed edge in $\mathbb{G}$ has causal interpretation, that is, $X \to Y$ indicates that X is a direct cause of Y.

Definition 5 (*Causal Structure Learning*)**.** Learning a directed acyclic graph (DAG) $\mathbb{G}$ of a set of variables $\boldsymbol{U}$ from observed data, with nodes denoting variables and edges denoting potential causal relationships between variables. If the variable X is the direct cause of Y, there exists a directed edge from node X to Y (Pearl, 2009). Global causal structure learning aims at learning a causal Bayesian network structure among variables. Local causal structure learning requires learning only the direct cause and direct effect of the given target variable.

Definition 6 (*V-structure (Pearl, 2014)*)**.** There are three variables X, Y, T forming a V-structure $X \to T \leftarrow Y$ if T has two incoming edges from X and Y, and X is not adjacent to Y. T is a collider.

Definition 7 (*Causal Sufficiency (Pearl, 2014; Spirtes et al., 2000)*)**.** Causal sufficiency assumes that any common cause of two or more variables in V is also in V.

In a BN, T is a collider when there are two directed edges from X to T and from Y to T, respectively, and they form a V-structure. A V-structure is an essential part of edge-orientating in causal structure learning.

Definition 8 (*D-separation (Pearl, 2014)*)**.** A path π between X and Y given a set $\boldsymbol{Z} \subseteq \boldsymbol{U} \backslash \{X, Y\}$ is blocked, if one of the following conditions is satisfied: (1) there is a non-collider variable with $\boldsymbol{Z}$ on π, or (2) there is a collider Z on π, while Z and its descendants are not in $\boldsymbol{Z}$. Otherwise, π between X and Y is unblocked. X and Y are d-separation given $\boldsymbol{Z}$ iff each path between X and Y is blocked by $\boldsymbol{Z}$.

In a faithful BN, X and Y are d-separation given $\boldsymbol{Z}$, $\boldsymbol{Z}$ is called the separation set, i.e., given a separation set $\boldsymbol{Z}$, X and Y are conditionally independent, i.e., $X \perp\!\!\!\perp Y \mid \boldsymbol{Z}$.

There is conditional independence implied by $X \perp\!\!\!\perp Y \mid T$ which is present in graphs like $X \to T \to Y$, $X \leftarrow T \leftarrow Y$ and $X \leftarrow T \to Y$. Although they have drastically different causal relations, the class of graphs that represents the same set of conditional independencies

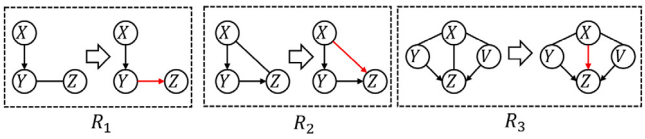


Fig. 2. Meek rules.

together constitutes the Markov equivalence class (MEC) (Vowels et al., 2022). Graphs belong to the same equivalence class when they have the same skeleton and the same immoralities (Verma & Pearl, 2022).

Definition 9 (*Partially Directed Acyclic Graph (Chickering, 2002)*)**.** A partially directed acyclic graph (PDAG) contains a directed edge for every edge participating in a V-structure and an undirected edge for every other uniquely identifying an equivalence class of a DAG.

An edge $X \to Y$ is compelled in $\mathbb{G}$ if that edge exists in every DAG that is equivalent to $\mathbb{G}$. If an edge $X \to Y$ is not compelled, we say that it is reversible (Chickering, 2002).

Definition 10 (*Completed Partially Directed Acyclic Graph (Chickering, 2002)*)**.** The completed PDAG (CPDAG) corresponding to an equivalence class is the PDAG consisting of a directed edge for every compelled edge in the equivalence class, and an undirected edge for every reversible edge in the equivalence class.

In causal structure learning, some algorithms cannot orient all edges' directions even with an exhaustive search. Therefore, some undirected edges will be retained in a PDAG and we take them as the CPDAG.

Definition 11 (*Markov Blanket (Pearl, 2014)*)**.** In a faithful BN, given a target variable T, the Markov blanket (MB) of T ($\boldsymbol{MB}_T$) is unique and consists of parents, children, and spouses of T.

All other variables in $\boldsymbol{U} \backslash \boldsymbol{MB}_T \cup \{T\}$ are conditionally independent of T given $\boldsymbol{MB}_T$, i.e., $\forall X \in \boldsymbol{U} \backslash \boldsymbol{MB}_T \cup \{T\}$, $X \perp\!\!\!\perp T \mid \boldsymbol{MB}_T$, where $X \perp\!\!\!\perp T \mid \boldsymbol{MB}_T$ denotes X and T are conditionally independent conditioning on $\boldsymbol{MB}_T$.

Theorem 1. *In a faithful BN (Spirtes et al., 2000), if any variable X and Y are adjacent, $X \not\perp\!\!\!\perp Y \mid \boldsymbol{Z}$, $X \in \boldsymbol{U}$, $Y \in \boldsymbol{U}$, $\boldsymbol{Z} \subseteq \boldsymbol{U} \backslash \{X, Y\}$.*

Theorem 2. *In a faithful BN, if any variables X, T, Y forming a V-structure ($X \to T \leftarrow Y$), then $X \perp\!\!\!\perp Y \mid \boldsymbol{Z}$, $X \not\perp\!\!\!\perp Y \mid \boldsymbol{Z} \cup \{T\}$, $\boldsymbol{Z} \subseteq \boldsymbol{U} \backslash \{X, Y, T\}$.*

Lemma 1. *The PC set of a given target variable T is denoted as $\boldsymbol{PC}_T$. Let $X \in \boldsymbol{PC}_T$, $Y \in \boldsymbol{PC}_T$. We can get the following two dependence relationships between X and Y:*

(1) $X \perp\!\!\!\perp Y \mid \emptyset$, $X \not\perp\!\!\!\perp Y \mid \{T\} \Rightarrow$ X and Y are both parents of T (X, T, Y forms a V-structure, $X \to T \leftarrow Y$, T is a collider).
(2) $X \not\perp\!\!\!\perp Y \mid \emptyset$, $X \perp\!\!\!\perp Y \mid \{T\} \Rightarrow$ at least one variable is the child of T (the possible structures are: $X \to T \to Y$, $X \leftarrow T \leftarrow Y$, $X \leftarrow T \to Y$). If X is the parent of T, Y must be the child of T.

Meek rules. Meek rules (Ling et al., 2021; Meek, 2013) orient undirected edges without destroying and introducing new V-structures. As shown in Fig. 2, specific rules are as follows:

(1) R_1: no new V-structures. Orient $Y - Z$ into $Y \to Z$ whenever there is a direct edge $X \to Y$ such that X and Z are not adjacent.
(2) R_2: preserve acyclicity. Orient $X - Z$ into $X \to Z$ whenever there is a chain $X \to Y \to Z$.
(3) R_3: enforce 3-fork V-structure. Orient $X - Z$ into $X \to Z$ whenever there is two chains $X - Y \to Z$ and $X - V \to Z$ such that Y and V are not adjacent.

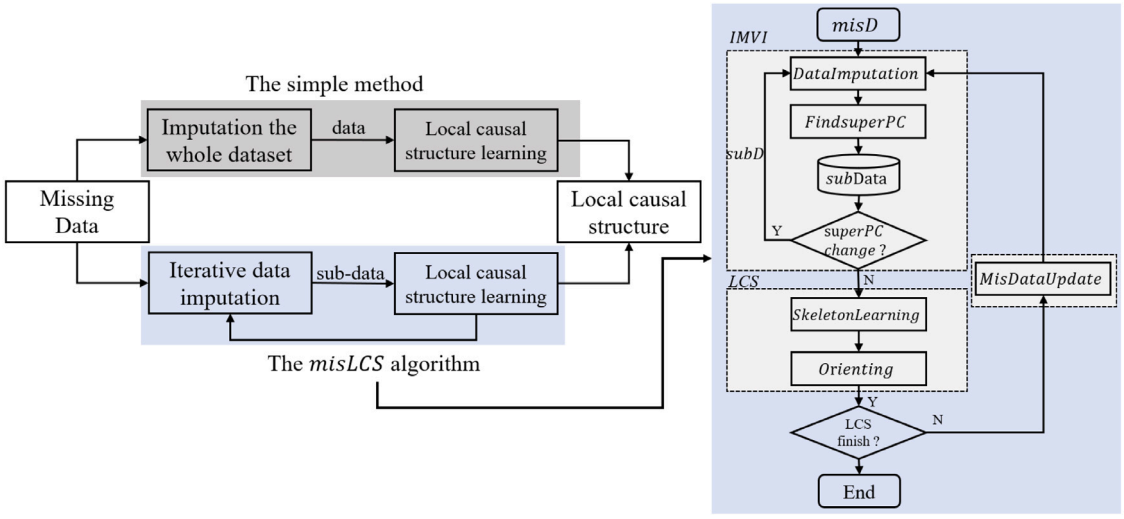


Fig. 3. Two methods of local causal structure learning with missing data.

Definition 12 (*Missing Value Completely at Random (MCAR) (Lin & Tsai, 2020)*). The probability of an instance (or case) having a missing value for an attribute (variable) does not depend on either the known values or the missing data. That is to say, in a dataset D_n^m (n is the number of instances, m is the number of attributes), the missing value in D_i^j (the ith instance and jth attribute) occurs does not depend on any conditions.

Assuming that the data missing rate is r, $rand(n)$ is a random integer randomly selected within the range $[0, n)$, and $NULL$ represents a missing value. A missing value completely at random in a D_n^m dataset is expressed as Eq. (1).

$$\begin{aligned} &\boldsymbol{for\ each}\ \ i = 1 : n * m * r \\ &\quad D_n^m(rand(n), rand(m)) = NULL \end{aligned} \tag{1}$$

Algorithm 1 misLCS

Input: missing data $misD$, target variable T, random variable set $\boldsymbol{U}$
Output: $\boldsymbol{P}_T$, $\boldsymbol{C}_T$, $\boldsymbol{UN}_T$

```
cloneD = misD
Que ← ∅, V ← ∅, subD ← ∅
Que.push(T)
repeat
    X = Que.pop()
    if X ∉ V then
        V = V ∪ X
        /*Step 1. iterative missing value imputing*/
        subD{X} = IMVI(misD, X)
        /*Step 2. LCS which includes local causal skeleton construction and
        causal direction identifying*/
        PC_X = SkeletonLearning(subD{X}, X)
        [P_X, C_X, UN_X] = Orienting(subD{X}, PC_X, X)
        /*Step 3. missing data updating*/
        misD = MisDataUpdate(UN_X, cloneD, misD, subD)
        Que.push(UN_X)
    end if
    Using Meek rules to orient the remaining undirected edges.
until (1) all parents and children of T can be determined, or (2) Que is
empty set, or (3) V is equal to U
```

4. The proposed method

Existing local causal structure learning algorithms, such as CMB (Gao & Ji, 2015), cannot learn causal relations from missing data. A common practice is to make the missing data completed and learn causal relations, as shown in Fig. 3. However, inaccurate data imputation results and shortcomings of existing local causal structure learning algorithms lead to more errors in the learnt PC set of a target variable. misLCS can improve performance in both missing data imputation and local causal structure learning strategies.

This paper presents the first algorithm to learn local causal structure with missing data (misLCS), which can effectively discover and distinguish parents and children of a given target variable. As shown in Algorithm 1 and the flow chart in Fig. 3, misLCS starts from a target variable and iteratively imputes missing values for getting a complete data subset (Step 1 of IMVI (Iterative Missing Value Imputation)), then constructs the local causal skeleton and orients edges direction based on the data subset (Step 2 of LCS (Local Causal Structure learning)), and finally updates the missing values (Step 3 of MisDataUpdate). This procedure continues until all the parents and children of the target variable have been distinguished or it is clear that they cannot be further distinguished. Note that all variables have missing values (the data missing mechanism is missing completely at random) other than the target variable. In the following section, we would give the details of misLCS.

4.1. IMVI subroutine

This section will explain why we design the data subset strategy and the iterative data imputation method and discuss the details of the IMVI subroutine.

There are two obstacles, in which the local causal structure learning cannot be performed in missing data. One is the current strategies of skeleton learning and causal direction identification not supporting missing data. The other is that irrelevant variables may hurt the performance of both data imputation and local causal structure learning. Kuang et al. (2023), non-causal features and spurious correlations are screened out by CI tests, which reduces the instability of prediction across unknown test data. We try to restrict the used variables for the CI test to correlated variables, rather than considering all variables in BN. $superPC$ is proposed to discover the true PC variables. The $superPC$ is composed of the PC_T and the PC of the PC_T, and $MB_T \subseteq superPC$ holds. Hence, we improve the performance of our method from two aspects: the removal of irrelevant variables and the improvement of data imputation strategy.

The data subset strategy is to obtain a data subset without irrelevant variables. From Section 3, we are aware that the relationship between any two variables can be identified by a series of conditional

Algorithm 2 IMVI

Input: missing data $misD$, target variable T
Output: complete data subset $subD$
1: $subD$ = ICkNNI($misD$)
2: **repeat**
3: $\boldsymbol{PC}$ = HITON-PC ($comD, T$)
4: $neigPC \leftarrow$ HITON-PC($comD, X$)$|X \in \boldsymbol{PC}$
5: $superPC \leftarrow \boldsymbol{PC} \cup neigPC$
6: $subD \leftarrow$ generate a new incomplete data only containing variables in $T \cup superPC$ and execute the missing value imputation – ICkNNI once again.
7: **until** the $superPC$ does not change or achieve stable

Algorithm 3 SkeletonLearning

Input: data subset $subD$, target variable T, threshold δ
Output: $\boldsymbol{PC}_T$
1: $\boldsymbol{PC}_T \leftarrow \emptyset, \boldsymbol{S} \leftarrow \emptyset$
2: **for each** variable $X \in subD.vars$ **do**
3: **if** $SU(T, X) > \delta$ **then**
4: $\boldsymbol{S} \leftarrow \boldsymbol{S} \cup X$
5: **end if**
6: **end for**
7: Order $\boldsymbol{S}$ in descending $SU(T; X)$
8: $len_S = |\boldsymbol{S}|, \boldsymbol{PC}_T = \boldsymbol{S}$
9: **for** $i =: len_S$ **do**
10: **for** $j = i + 1 : len_S$ **do**
11: $SU(\boldsymbol{S}(i), \boldsymbol{S}(j)) > SU(\boldsymbol{S}(j), T)$
12: $\boldsymbol{PC}_T \leftarrow \boldsymbol{PC}_T \backslash \boldsymbol{S}(j)$
13: **end for**
14: **end for**

independence judgments and the dependence relationship between two direct-connected variables will not change when discarding some variables in a dataset. Removing some variables could simplify the identified process. Kuang et al. (2023), non-causal features and spurious correlations are screened out by CI tests, which reduces the instability of prediction across unknown test data. We try to restrict the used variables for the CI test to correlated variables, rather than considering all variables in BN. $superPC$ is proposed to discover the true PC variables. The $superPC$ is composed of the PC_T and the PC of the PC_T, and $MB_T \subseteq superPC$ holds.

Take Fig. 1 as an instance, A is the direct cause of T, and the dependence between T and A remains whether all other variables are discarded or not. Variables B, T, D form a V-structure: $B \rightarrow T \leftarrow D$, when L is discarded, B and D are conditionally independent given an empty set as the condition set, while they are dependent conditioning on $\{T\}$.

In addition, some irrelevant variables would reduce the precision of missing value imputation. For example, some algorithms based on kNN (k Nearest Neighbors) calculate the distance between data cases to complete the missing values. The fewer irrelevant variables, the higher the missing value imputation precision. Therefore, the data subset strategy could enhance the precision of data imputation as well.

For we do not know which variables are closely related to T, we fill up the all missing values in a data set first, then compute the $superPC$ and extract a data subset that only contains $superPC$ from the raw data set. And then we re-impute this incomplete data subset and re-find the $superPC$. This iterative process ends when variables in $superPC$ do not change anymore. The $superPC$ successfully achieves the goal of accurately filling in missing data values and precisely identifying relevant variables.

As shown in Algorithm 2, we improve ICkNNI (Incomplete-Case k Nearest Neighbors Imputation) (Van Hulse & Khoshgoftaar, 2014) to fill in missing values and use HITON-PC (Aliferis et al., 2003) to find the PC of a target variable. Remarkably, ICkNNI is an excellent

Algorithm 4 Orienting

Input: data subset $subD$, parents and children $\boldsymbol{PC}_T$, target variable T
Output: $\boldsymbol{P}_T, \boldsymbol{C}_T, \boldsymbol{UN}_T$
1: $\boldsymbol{W} = D.vars$
2: /*step 1. find the parents of T*/
$[\boldsymbol{P}_T, \boldsymbol{C}_T, \boldsymbol{UN}_T]$ = FindParent($subD, \boldsymbol{W}, \boldsymbol{PC}_T, T$)
3: /*step 2. find the children of T*/
$\boldsymbol{C}_T$ = FindChildren($subD, \boldsymbol{W}, \boldsymbol{PC}_T, \boldsymbol{C}_T, \boldsymbol{UN}_T, T$)
4: $\boldsymbol{UN}_T \leftarrow \boldsymbol{UN}_T \backslash \boldsymbol{C}_T$

way that allows for the simultaneous use of complete and incomplete cases to fill in missing values. ICkNNI has extensive applicability and is not limited by the data missing rate and the data missing mechanism (i.e., missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR)). It can be effectively used in various scenarios to fill up missing values. HITON-PC computes the dependences between the target variable and the rest of the variables under different conditioning sets and adds the variables with the high dependences into the PC set of the target variable as much as possible. It may find some variables incorrectly, but not lose relevant variables. When the $superPC$ keeps the same as the last iteration, the algorithm stops and returns the last completed data subset.

In summary, IMVI not only removes the disturbance of irrelevant variables but also offers a complete data set for local causal structure learning in the next step.

4.2. LCS subroutine

The main goal of LCS is to discover and distinguish parents and children given a variable. It includes two steps: skeleton learning and edge orienting. Next, we will discuss the two steps in detail.

4.2.1. SkeletonLearning

This step discovers the PC of the target variable by FCBF (Fast Correlation-Based Filter) (Yu & Liu, 2004) and constructs local causal structures over the learnt PC set.

We had used HITON-PC to learn $superPC$ and LCS could get PC in the data subset directly. To improve time efficiency of learning the PC set from the $superPC$ set, we employ a lightweight PC learning algorithm, FCBF.

Feature selection (Yu et al., 2021) is to identify a subset of features (predictor variables) from the original features for model building or data understanding. It aims to find strongly relevant features of a given variable in a causal (Lee et al., 2020) or non-causal (Yu & Liu, 2004) way. FCBF is a classical mutual information-based and non-causal method for feature selection, which uses symmetric uncertainty (abbr. SU) to identify the PC set of the target variable. And FCBF does not need to specify the number of selected features in advance and can guarantee that all true PC would not be dropped.

The symmetric uncertainty formula is shown in Eq. (2). It utilizes mutual information to compute the correlation between two variables. $SU(X, Y) = 0$ indicates that X and Y are independent of each other, otherwise, they are dependent. Furthermore, the strength of their correlation depends on the value of SU. We need to set the threshold δ in advance to control the size of $\boldsymbol{S}$ (the number of potential PC of T). When $SU(T, X) > \delta$, X is considered as the PC variable of T.

$$SU(X, Y) = 2\left[\frac{IG(X, Y)}{H(X) + H(Y)}\right] \tag{2}$$

4.2.2. Orienting subroutine

This step distinguishes parents and children in the learnt PC set by multiple conditional independence (CI) tests.

Algorithm 4 presents the orienting subroutine. Though the PC of T could be obtained after skeleton learning, the direct cause (parent)

Algorithm 5 FindParent

Input: data subset $subD$, random variable set $\boldsymbol{W}$, parents and children $\boldsymbol{PC}_T$, target variable T
Output: $\boldsymbol{P}_T, \boldsymbol{C}_T, \boldsymbol{UN}_T$
1: $\boldsymbol{P}_T \leftarrow \emptyset, \boldsymbol{C}_T \leftarrow \emptyset, \boldsymbol{UN}_T \leftarrow \emptyset$
2: $tmp \leftarrow \emptyset, count = 1$
3: **for** each $X \in \boldsymbol{PC}_T$ **do**
4: **for** each $Y \in \boldsymbol{PC}_T$ **do**
5: **if** $X \perp\!\!\!\perp Y \mid \emptyset$ and $X \not\perp\!\!\!\perp Y \mid \{T\}$ **then**
6: $\boldsymbol{P}_T \leftarrow \boldsymbol{P}_T \cup \{X, Y\}$
7: **else if** $X \not\perp\!\!\!\perp Y \mid \emptyset$ and $X \perp\!\!\!\perp Y \mid \{T\}$ **then**
8: $tmp\ [count] \leftarrow \{X, Y\}$
9: $count = count + 1$
10: **end if**
11: **end for**
12: **end for**
13: **for** $i = 1 : count$ **do**
14: **if** $tmp\ [count][1] \in \boldsymbol{P}_T$ **then**
15: $\boldsymbol{C}_T \leftarrow \boldsymbol{C}_T \cup \{tmp\ [count][2]\}$
16: **else if** $tmp[count][2] \in \boldsymbol{P}_T$ **then**
17: $\boldsymbol{C}_T \leftarrow \boldsymbol{C}_T \cup \{tmp\ [count][1]\}$
18: **end if**
19: **end for**
20: $\boldsymbol{UN}_T \leftarrow \boldsymbol{PC}_T \backslash \boldsymbol{P}_T \backslash \boldsymbol{C}_T$

Algorithm 6 FindChildren

Input: data subset $subD$, random variable set $\boldsymbol{W}$, parents and children $\boldsymbol{PC}_T$, children $\boldsymbol{C}_T$, undistiguished variables $\boldsymbol{UN}_T$, target variable T
Output: $\boldsymbol{C}_T$
1: $\boldsymbol{Sep}_T \leftarrow \emptyset, \boldsymbol{canSP}_T \leftarrow \emptyset$
2: **for** each $X \in \boldsymbol{UN}_T$ **do**
3: $\boldsymbol{PC}_X$ = SkeletonLearning (D, X)
4: $\boldsymbol{canSP}_T\{X\} = \boldsymbol{PC}_X \backslash \boldsymbol{PC}_T \cup \{T\}$
5: **for** each $Y \in \boldsymbol{canSP}_T\{X\}$ **do**
6: **if** $T \perp\!\!\!\perp Y \mid Z$ for each $Z \in \boldsymbol{PC}_T$ **then**
7: $\boldsymbol{Sep}_T\{Y\} \leftarrow \boldsymbol{Sep}_T\{Y\} \cup \{Z\}$
8: **end if**
9: **end for**
10: **end for**
11: $\boldsymbol{SP}_T \leftarrow \emptyset$
12: **for** each $X \in \boldsymbol{UN}_T$ **do**
13: **for** each $Y \in \boldsymbol{canSP}_T$ **do**
14: **if** $T \not\perp\!\!\!\perp Y \mid \{X\} \cup \boldsymbol{Sep}_T\{Y\}$ **then**
15: $\boldsymbol{SP}_T\{X\} \leftarrow \boldsymbol{SP}_T\{X\} \cup \{Y\}$
16: **end if**
17: **end for**
18: **end for**
19: **for** each $X \in \boldsymbol{UN}_T$ **do**
20: **if** $T \perp\!\!\!\perp X \mid \boldsymbol{SP}_T\{X\} \cup \boldsymbol{UN}_T \backslash \{X\}$ **then**
21: $\boldsymbol{UN}_T \leftarrow \boldsymbol{UN}_T \backslash \{X\}$, $\boldsymbol{SP}_T\{X\} \leftarrow \emptyset$
22: **end if**
23: **end for**
24: **for** each $X \in \boldsymbol{UN}_T$ **do**
25: **if** $\boldsymbol{SP}_T\{X\}$ is nonempty **then**
26: $\boldsymbol{C}_T \leftarrow \boldsymbol{C}_T \cup \{X\}$
27: **end if**
28: **end for**

and direct effect (children) have not been distinguished yet. The common practice is to use the CI test to judge causal-effect relationships between variables. We determine the edge-orientating order based on their difficulty degree. The parents can be identified by using the V-structure and Lemma 1. Additionally, it is possible to identify some of the children. Whereas finding the children involves the need to find their separation sets and spouses, which can introduce potential errors at each step and decrease its accuracy. In comparison to identifying parents, finding children is generally more challenging. Therefore, in the orienting subroutine, our algorithm finds the parent first and then the children.

The FindParent subroutine is shown in Algorithm 5 which aims at finding the parent of T. It makes good use of Lemma 1. That is to say, for each pair of variables in PC, both of them are stored as parents if they satisfied the rule of V structure: $X \perp\!\!\!\perp Y \mid \emptyset$ and $X \not\perp\!\!\!\perp Y \mid \{T\}$ (line 3∽6). The pairs of variables meet the one parent and one child condition: $X \not\perp\!\!\!\perp Y \mid \emptyset$ and $X \perp\!\!\!\perp Y \mid \{T\}$, and cannot be distinguished immediately. They are recorded to tmp (line 7∽9). Finally, traversing the set tmp, if one has been confirmed as the parent, the other should be the child (line 13∽19).

The detail of FindChildren is shown in Algorithm 6. It tries to identify the children by searching for the spouse of T. The FindChildren algorithm first computes the separation set Sep_T for each variable in the candidate children set $\boldsymbol{UN}_T$ (line 2∽10). Y is the calculated candidate spouse (line 3∽4) and Z belongs to $\boldsymbol{PC}_T$, if T and Y are conditionally independence given Z, Z is the member of $\boldsymbol{Sep}_T$ (Line 6). Therein, $\boldsymbol{Sep}_T\{Y\}$ denotes the separation set of T with respect to Y (similarly hereinafter). Then, the spouse of T can be identified preliminarily based on the candidate spouse $canSP_T$ and separation set Sep_T (line 12∽18). X is the candidate child and Y is the candidate spouse, if T is not conditionally independence given $\{X\} \cup \boldsymbol{Sep}_T\{Y\}$, Y is very likely the spouse of T. Thirdly, some wrong spouses could be filtered (line 19∽23). X is the candidate child, if T and X are not conditionally independence given $\boldsymbol{SP}_T\{X\} \cup \boldsymbol{UN}_T \backslash \{X\}$, X is not the child of T, and the $\boldsymbol{SP}_T\{X\}$ is set to the empty set. Finally, if T has a spouse about a certain child (line 25), we find the true children.

4.3. MisDataUpdate subroutine

As the PC set of T are not distinguishable at once, our algorithm takes a variable belonging to $\boldsymbol{UN}_T$ as a new target variable (line 12 in Algorithm 1). misLCS will learn their local causal structure, and then use three Meek rules to infer the edge direction between T and the variables in $\boldsymbol{UN}_T$ (line 14 in Algorithm 1). But the values in the newly set target variable are missing initially. The simple way is to take the last completed value as the true value. While a more reliable way is to employ the mode value of multiple-time fills as the true value. Therefore, the MisDataUpdate subroutine is developed to update the missing value before the next iteration.

The specific practice is as follows: the filled data subset will be stored in $subD$ in each iteration. When a missing variable is used as the new target variable, the mode value of multiple fills for this target variable is taken as the final value and labeled as the no-missing variable. That is, let X be the missing variable, and instance 1 has a missing value on X, there are four stored imputation values: {1, 0, 1, 1}, respectively. Then the mode value of 1 and is taken as the final value of X and instance 1. X is labeled as the non-missing variable.

Take Fig. 1 as an example, suppose that T and G are regarded as the target variable, and B will be imputed twice. When B is set as the target variable, the mode of the two imputed values is B's final values, and B is not the missing variable anymore. The purpose of this subroutine is to ensure that the variable used as the target variable has no missing values and obtains its more accurate values.

4.4. Tracing

In this subsection, we trace the execution of misLCS based on the example in Fig. 1. Suppose that we have a dataset for the variable set $\boldsymbol{U} = \{T, A, B, E, F, G, H, I, J, K, L, M, N\}$. The independent relationship between any two variables can be represented by the Bayesian Network structure in Fig. 1. In the following, we take T as the target variable and give the execution process of misLCS. Note that the target variable T has no missing values.

(1) Step 1. ICkNNI is used to impute all missing values in the missing dataset and get a complete dataset. HITON-PC then finds the $superPC$,

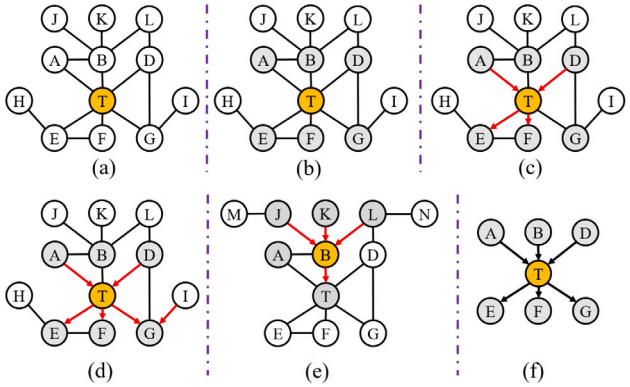


Fig. 4. An example of the execution process of misLCS.

i.e., $superPC = \{A, B, D, E, F, G, H, I, J, K, L, M\}$. M is wrongly added for the reason of incorrect data imputation and irrelevant variables distraction. IMVI extracts a data subset containing variables in $superPC \cup T$ and produces a new missing data subset–$subD$. It will re-impute $subD$ by ICkNNI and re-find the $superPC$ by HITON-PC. M is removed in the second iteration. This procedure will stop until $superPC$ does not change. Finally, as shown in Fig. 4(a), it will return the last imputed data subset that contains variables $\{A, B, D, E, F, G, H, I, J, K, L, T\}$.

(2) Step 2. FCBF is used to find the PC from $superPC$. According to Algorithm 3, A, B, D, E, F, G will be added to $\boldsymbol{PC}_T$. Note that, the $SU(I, T) = 0$, hence I will not be added to $\boldsymbol{S}$. M and N does not in this data subset, hence, they will not be calculated. As shown in Fig. 4(b), $\boldsymbol{PC}_T = \{A, B, D, E, F, G\}$ (marked in gray).

(3) Step 3. Edge orienting is used to distinguish parents and children. First, finding parents. $A \perp\!\!\!\perp D \mid \emptyset$ and $A \not\perp\!\!\!\perp D \mid \{T\}$, based on Lemma 1. (1), both A and D are parents of T. $A \not\perp\!\!\!\perp E \mid \emptyset$ and $A \perp\!\!\!\perp E \mid \{T\}$, based on Lemma 1. (2) and A is the parent of T, E is the child of T. Like E and A, F is also the child of T. Hence, as shown in Fig. 4(c), $\boldsymbol{P}_T = \{A, D\}$, $\boldsymbol{C}_T = \{E, F\}$, $\boldsymbol{UN}_T = \boldsymbol{PC}_T \backslash \boldsymbol{P}_T \backslash \boldsymbol{C}_T = \{B, G\}$.

Second, finding children. For each undistinguished variables in $\boldsymbol{UN}_T$, $\boldsymbol{canSP}_T\{B\} = \boldsymbol{PC}_B \backslash \boldsymbol{PC}_T \backslash \{T\} = \{J, K, L\}$, and $\boldsymbol{Sep}_T\{J\} = \{B\}$, $\boldsymbol{Sep}_T\{K\} = \{B\}$, $\boldsymbol{Sep}_T\{L\} = \{B, D\}$. $\boldsymbol{canSP}_T\{G\} = \boldsymbol{PC}_G \backslash \boldsymbol{PC}_T \backslash \{T\} = \{I\}$ and $\boldsymbol{Sep}_T\{I\} = \emptyset$. $T \perp\!\!\!\perp J \mid \{B\} \cup \boldsymbol{Sep}_T\{J\}$, J is not the spouse of T. Similarly, K and L are also not spouse of T, and hence $\boldsymbol{SP}_T\{B\} = \emptyset$. While $T \not\perp\!\!\!\perp I \mid \{B\} \cup \boldsymbol{Sep}_T\{I\}$, J is the spouse of T, and $\boldsymbol{SP}_T\{G\} = \{I\}$. Thus, as shown in Fig. 4(d), G is the child of T. $\boldsymbol{C}_T = \boldsymbol{C}_T \cup \{G\} = \{E, F, G\}$. The final unidentified variable is B, and we store it into the Que as the next target variable in the last.

(4) Step 4. In this step, the missing values in variables $superPC$ have been stored. When B is the new target variable, MisDataUpdate takes the imputed values for B as its fixed value and other variables are still missing. The algorithm will repeat steps 1∽3 to get $\boldsymbol{P}_B = \{J, K, L\}$, as shown in Fig. 4(e). Based on Meek rules, B is the parent of T. The parent and children of T are all known, as shown in Fig. 4(f), and the program ends.

4.5. The differences between misLCS and existing local causal structure learning algorithms

In this section, the main differences between misLCS and existing local causal structure learning algorithms will be explained from the following three aspects.

Firstly, during the skeleton learning phase of misLCS, there is an interleaving execution between missing data imputation and skeleton learning. This strategy is designed to enhance the precision of missing value imputation while simultaneously improving the accuracy of skeleton learning, that is, skeleton learning and data imputation promote each other.

Table 2
Summary of benchmark BNs.

Network	Num. Vars	Num. Edges	Max In/out Degree	Min/Max |PCset|	Domain Range
Child	20	25	2/7	1/8	2–6
Alarm1	37	46	4/5	1/6	2–4
Mildew	35	46	3/3	1/5	3–100
Alarm3	111	149	4/5	1/6	0–3
HailFinder5	280	458	5/18	1/19	0–10
Pigs	441	592	2/39	1/41	3–3
Gene	801	972	4/10	0/11	2–5

Secondly, during the edge orientation phase of misLCS, misLCS employs a data subset (containing the potential variables belonging to the skeleton of the target variable) obtained from the earlier skeleton learning stage to distinguish between parents and children instead of using all of the data. This approach aims to mitigate the potential adverse impact of irrelevant variables.

Thirdly, during the edge orientation phase, the sequential order we employ for edge orientation is to identify parents first and then children. Specifically, when the target variable is a collider, we determine its adjacent variables in the PC as the parents. Following this, we proceed to identify children by assessing condition independence based on a conditional set containing the target variable or by searching for the spouse of the target variable.

In conclusion, the interleaving execution between missing data imputation and skeleton learning, the use of data subset during the edge orientation phase, and the order of edge orientation are the three main differences between misLCS and existing algorithms.

5. Experiments

In this section, we compare performance of the misLCS algorithm against its rivals. This section is organized as follows: Section 5.1 gives the experimental setting, and Sections 5.2 and 5.3 summarizes and discusses the experimental results on seven benchmark BNs and a real-world dataset, respectively.

5.1. Experimental setting

5.1.1. Comparision algorihm

We compare our approach misLCS with four state-of-the-are local causal structure learning algorithms, including PCD-by-PCD (Yin et al., 2008), MB-by-MB (Wang et al., 2014), CMB (Gao & Ji, 2015), and ELCS (Yang et al., 2021). As shown in Fig. 3, due to those four algorithms cannot learn the local causal structure with missing data, we fill the missing data into complete data using the method of missing value imputation initially.

5.1.2. Evaluation metric

In the experiments, we evaluate the performance of local causal structure learning using the following three general metrics.

- SHD: SHD is the number of total error edges, which includes undirected edges, reverse edges, missing edges, and extra edges. The smaller value of SHD is better.
- $Precision$: the number of true directed edges in the output (i.e., the variables in the output belonging to the true parents and children of a target variable in a test DAG) divided by the number of edges in the output of an algorithm
- $Distance = \sqrt{(1 - Precision)^2 + (1 - Recall)^2}$.
  $Precision$ is same as above and $Recall$ is the number of true directed edges in the output divided by the number of true directed edges (i.e., the number of parents and children of a target variable in a test DAG). It means that $Distance = 0$ is the best case (perfect precision and recall) while $Distance = \sqrt{2}$ is the worst case. Thus, the lower $Distance$ is better.

Table 3
SHD on seven BNs using different data sizes and missing rates. SHD denotes the total number of error edges, which contain undirected edges, reverse edges, missing edges, and extra edges. Lower value is better.

Network	Algorithm	Size = 500						Size = 1000					
		10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
Child	PCD-by-PCD	3.80	4.33	4.73	4.87	5.93	8.00	3.53	3.93	3.80	5.13	6.73	7.73
	MB-by-MB	3.40	3.73	4.07	4.60	4.53	6.00	3.47	3.80	4.20	4.20	5.60	5.13
	CMB	3.20	3.87	3.60	4.33	5.33	7.53	4.20	3.33	3.40	4.27	5.80	8.20
	ELCS	**2.80**	3.00	3.13	4.20	4.80	7.13	**2.13**	**2.47**	3.53	4.07	5.47	7.13
	misLCS	2.87	**2.87**	**2.67**	**2.40**	**3.07**	**3.80**	2.80	2.80	**2.40**	**2.40**	**2.27**	**3.27**
Alarm1	PCD-by-PCD	1.67	2.40	2.73	2.67	3.33	4.07	**1.13**	1.40	2.27	2.47	3.20	4.00
	MB-by-MB	1.60	2.27	2.27	3.07	3.13	4.13	1.93	2.13	2.73	2.87	3.60	4.53
	CMB	1.80	2.33	2.80	3.07	3.93	4.53	1.80	1.73	2.07	2.60	3.67	4.53
	ELCS	**1.53**	**1.80**	2.33	2.27	3.47	4.20	1.27	**1.20**	1.87	2.53	3.33	4.93
	misLCS	1.67	1.93	**1.87**	**1.93**	**2.00**	**3.00**	1.53	1.47	**1.07**	**1.67**	**2.27**	**2.53**
Mildew	PCD-by-PCD	23.73	24.13	24.53	25.00	25.67	26.93	20.00	20.73	21.47	23.00	24.53	26.27
	MB-by-MB	4.20	4.20	**4.20**	**4.20**	**4.20**	**4.20**	4.13	4.13	4.33	4.80	4.13	5.20
	CMB	4.33	4.60	5.00	5.40	5.93	7.13	6.87	7.73	8.27	9.73	11.27	13.00
	ELCS	4.93	5.20	5.60	6.00	6.47	7.60	7.87	8.73	9.33	10.87	12.33	14.07
	misLCS	**3.33**	**4.07**	5.00	4.93	5.33	4.40	**2.53**	**3.20**	**3.87**	**4.73**	**5.47**	**5.80**
Alarm3	PCD-by-PCD	1.53	2.27	2.20	2.47	2.93	2.73	1.60	1.67	2.27	2.47	2.53	2.87
	MB-by-MB	2.73	2.27	3.73	3.27	3.20	3.53	3.53	3.80	3.53	3.13	3.13	4.07
	CMB	2.53	3.13	2.87	3.33	3.73	3.60	2.40	2.60	2.20	2.93	3.00	3.73
	ELCS	2.40	2.67	2.67	3.33	3.60	3.67	2.20	2.07	2.47	2.87	3.27	3.60
	misLCS	**1.87**	**1.53**	**1.87**	**2.33**	**2.33**	**2.73**	**1.13**	**1.40**	**1.20**	**1.87**	**2.07**	**2.40**
HailFinder5	PCD-by-PCD	5.80	5.60	5.33	4.80	5.07	6.00	4.80	4.87	4.87	4.67	4.93	–
	MB-by-MB	4.13	4.33	4.13	4.07	4.07	3.53	3.73	3.67	4.00	4.00	4.13	–
	CMB	–	–	–	–	–	–	4.07	5.00	4.67	5.27	–	–
	ELCS	6.67	6.33	6.47	6.00	6.27	–	4.33	5.07	5.27	5.73	5.73	–
	misLCS	**2.53**	**2.87**	**2.47**	**2.53**	**2.40**	**2.80**	**2.53**	**2.73**	**2.40**	**2.40**	**2.47**	–
Pigs	PCD-by-PCD	0.87	0.93	1.53	1.33	2.47	2.87	0.93	1.93	1.27	1.60	2.53	2.93
	MB-by-MB	2.47	2.53	2.80	3.07	3.53	**3.47**	2.40	1.73	2.07	2.47	2.80	2.93
	CMB	1.27	1.47	1.67	2.40	2.40	3.67	1.27	1.53	1.80	2.27	2.73	3.67
	ELCS	**0.67**	**0.53**	**0.67**	1.53	2.40	3.73	0.87	1.40	1.47	**1.47**	3.07	3.87
	misLCS	0.93	0.80	1.07	**1.13**	**1.93**	3.67	**0.73**	**0.93**	**1.13**	**1.47**	**1.80**	**2.20**
Gene	PCD-by-PCD	2.33	1.80	3.13	3.40	3.27	3.87	1.93	2.80	3.33	4.07	4.33	4.13
	MB-by-MB	4.13	3.73	4.47	4.20	5.07	4.67	4.00	4.27	4.00	3.80	3.80	4.73
	CMB	1.47	1.73	2.73	2.40	3.60	4.20	1.60	2.20	3.07	3.40	3.60	3.67
	ELCS	**0.67**	1.40	2.27	2.40	2.60	3.60	1.47	1.67	2.20	2.80	3.40	3.80
	misLCS	**0.67**	**0.87**	**0.80**	**1.80**	**2.33**	**2.93**	**0.53**	**0.60**	**1.40**	**1.60**	**1.87**	**2.53**

5.1.3. Implementation detail

PCD-by-PCD, MB-by-MB, CMB, and ELCS algorithms are implemented by ourselves in MATLAB (https://github.com/kuiy/CausalLearner). ICkNNI is also implemented by ourselves in MATLAB. In the experiments, G^2-test with the significance level of 0.01 is utilized to measure the conditional independence between variables. The threshold δ is preset to 0.05 in FCBF for selecting potential PC of T. All experimental results are conducted on Windows 10 with Intel(R) i9-10900F, 2.80 GHz CPU, and 16 GB memory. These experimental settings ensure that our results are reliable and can be easily reproduced.

5.2. Benchmark BNs dataset

This paper first uses seven benchmark BNs with low to high dimensionality to evaluate misLCS against its rivals. The number of variables of these BNs ranges from 20 to 801. Each BN has two groups of data: one group contains three data sets with 500 data samples, and the other one contains three data sets with 1000 data samples. A brief description of the seven benchmark BNs are listed in Table 2.

We carefully use seven BNs to generate synthetic datasets with different missing rates (ranging from 10% to 60%). And the data missing setting is: the target variable is not missing, and the missing mechanism of the remaining variables is missing completely at random (MCAR, see Definition 12). Especially, misLCS randomly select five variables in a BN as target variables. Then each algorithm is run on those synthetic data sets to learn the local causal structure of five variables. And we compute the average performance of five target variables as the final performance of each algorithm.

5.2.1. Experiments results of misLCS and its rivals

In Tables 3–5, values in those table represent the average result, and we use '–' to denote that a method does not generate results with the corresponding BN due to the total running times of five variables on three data sets exceeding more than two-days, and the best results are highlighted in boldface type. Based on our experimental results, we make the following observation.

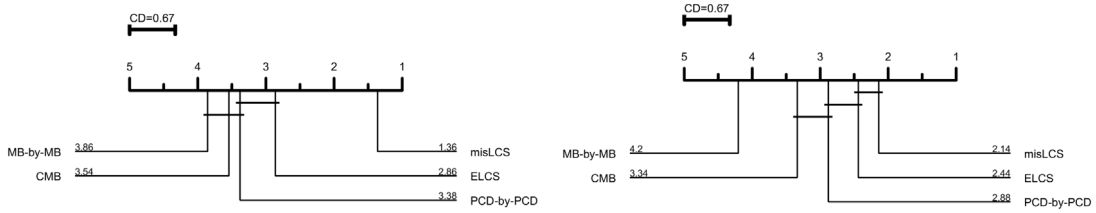
Table 3 presents the total number of error edges for seven BNs. When the data sample is 500 and the missing rate is higher than 30%, misLCS consistently outperforms the other four algorithms in terms of SHD, except for the Mildew dataset which requires a large number of data samples due to its large domain ranges. In cases where the data sample is 1000 and the missing rate is higher than 20%, misLCS also outperforms the other four algorithms. Specifically, for datasets such as Alarm3, HailFinder5, and Gene, misLCS consistently achieves the best SHD performance. It is worth noting that MB-by-MB performs poorly with only 500 data samples due to the lack of sufficient samples for learning, especially for the Mildew dataset. Overall, misLCS exhibits superior performance compared to the other four algorithms, as it achieves the lowest total number of learning error edges.

Table 4 displays the precision of causal structure learning on seven BNs. It shows that in datasets such as Mildew, HailFinder5, and Gene, the *Precision* of misLCS is consistently higher than the other four algorithms. For the remaining datasets, there are only a few cases where the missing rate is slightly below the optimal value. In the Pigs dataset, misLCS is only outperformed by PCD-by-PCD when the missing rate is 60% and data samples are 500. Therefore, misLCS exhibits the highest precision among the five algorithms.

Table 5 gives the calculated distance which considers both precision and recall together. We figure out the percentage of the best values for all algorithms. There are 42.86%, 27.38%, 19.05%, 3.57%, 2.38% for

Table 4
Precision on seven BNs using different data sizes and missing rates. *Precision* denotes the accuracy of learning edges. Higher value is better.

Network	Algorithm	Size = 500						Size = 1000					
		10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
Child	PCD-by-PCD	0.49	0.43	0.39	0.43	0.36	0.25	0.50	0.48	0.54	0.39	0.27	0.28
	MB-by-MB	**0.67**	**0.68**	0.51	0.51	0.38	0.16	0.62	0.61	0.53	0.54	0.32	0.37
	CMB	0.64	0.53	0.60	0.61	0.43	0.31	0.32	0.61	0.66	0.50	0.38	0.26
	ELCS	0.62	0.57	0.61	0.48	0.44	0.34	**0.71**	**0.67**	0.55	0.54	0.41	0.34
	misLCS	0.60	0.53	**0.63**	**0.73**	**0.61**	**0.58**	0.61	0.59	**0.67**	**0.67**	**0.65**	**0.54**
Alarm1	PCD-by-PCD	**0.89**	0.71	0.62	0.69	0.64	0.57	**0.97**	**0.90**	0.73	0.71	0.60	0.54
	MB-by-MB	0.84	0.74	0.72	0.60	0.55	0.47	0.78	0.70	0.67	0.59	0.52	0.42
	CMB	0.73	0.70	0.59	0.56	0.54	0.40	0.73	0.73	0.69	0.61	0.50	0.43
	ELCS	0.85	0.82	0.73	0.78	0.53	0.48	0.89	0.83	0.79	0.62	0.53	0.40
	misLCS	0.88	**0.82**	**0.77**	**0.80**	**0.68**	**0.58**	0.81	0.84	**0.94**	**0.78**	**0.68**	**0.64**
Mildew	PCD-by-PCD	0.15	0.14	0.14	0.13	0.12	0.11	0.17	0.16	0.15	0.14	0.13	0.12
	MB-by-MB	0.00	0.00	0.00	0.00	0.00	0.00	0.07	0.07	0.20	0.06	0.30	0.02
	CMB	0.30	0.28	0.23	0.21	0.19	0.15	0.26	0.23	0.22	0.20	0.16	0.14
	ELCS	0.18	0.17	0.14	0.13	0.13	0.11	0.14	0.13	0.11	0.10	0.09	0.08
	misLCS	**0.63**	**0.54**	**0.47**	**0.42**	**0.45**	**0.59**	**0.70**	**0.62**	**0.52**	**0.48**	**0.40**	**0.46**
Alarm3	PCD-by-PCD	0.45	0.33	0.44	0.39	0.33	0.40	0.54	0.55	0.40	0.42	0.38	0.36
	MB-by-MB	**0.49**	**0.58**	0.41	0.41	0.45	0.37	0.42	0.38	0.47	**0.53**	0.47	0.33
	CMB	0.44	0.36	0.40	0.37	0.34	0.32	0.51	0.46	0.56	0.48	0.44	0.35
	ELCS	0.49	0.49	**0.52**	0.41	0.40	0.30	0.51	0.56	0.54	0.54	**0.48**	0.34
	misLCS	0.45	0.56	0.46	**0.41**	**0.47**	**0.45**	**0.62**	**0.58**	**0.60**	0.52	0.46	**0.49**
HailFinder5	PCD-by-PCD	0.48	0.39	0.36	0.37	0.54	0.47	0.55	0.58	0.42	0.42	0.31	–
	MB-by-MB	0.27	0.26	0.32	0.31	0.27	0.39	0.43	0.41	0.43	0.43	0.43	–
	CMB	–	–	–	–	–	–	0.55	0.41	0.46	0.40	–	–
	ELCS	0.45	0.32	0.31	0.40	0.41	–	0.54	0.52	0.40	0.30	0.26	–
	misLCS	**0.66**	**0.62**	**0.73**	**0.67**	**0.59**	**0.57**	**0.72**	**0.70**	**0.76**	**0.74**	**0.71**	–
Pigs	PCD-by-PCD	0.84	0.81	0.73	0.77	0.55	**0.52**	0.82	0.67	0.77	0.74	0.55	0.53
	MB-by-MB	0.63	0.63	0.60	0.53	0.48	0.45	0.66	0.74	0.68	0.63	0.58	0.51
	CMB	0.74	0.69	0.69	0.53	0.50	0.49	0.73	0.71	0.68	0.60	0.53	0.45
	ELCS	0.85	0.88	0.85	0.72	0.50	0.46	0.85	0.73	0.76	0.75	0.48	0.45
	misLCS	**0.89**	**0.91**	**0.82**	**0.82**	**0.63**	0.46	**0.92**	**0.89**	**0.88**	**0.76**	**0.63**	**0.65**
Gene	PCD-by-PCD	0.51	0.63	0.47	0.38	0.39	0.36	0.50	0.52	0.49	0.39	0.33	0.41
	MB-by-MB	0.37	0.44	0.37	0.36	0.36	0.44	0.40	0.35	0.41	0.43	0.43	0.35
	CMB	0.71	0.68	0.58	0.63	0.44	0.38	0.67	0.66	0.56	0.50	0.48	0.49
	ELCS	0.87	0.76	0.69	0.58	0.60	0.46	0.67	0.73	0.65	0.59	0.53	0.55
	misLCS	**0.87**	**0.86**	**0.82**	**0.62**	**0.64**	**0.54**	**0.88**	**0.88**	**0.68**	**0.63**	**0.60**	**0.55**



(a) *Precision* (b) *Distance*

Fig. 5. Crucial difference diagram of the Nemenyi test for *Precision* / *Distance* of local causal structure learning algorithms.

misLCS, ELCS, PCD-by-PCD, CMB, MB-by-MB, respectively. It signifies that the misLCS is more likely to get the best results.

To further analyze the significant difference between misLCS and its rivals on seven BNs, we perform the Nemenyi test, which states that the performance of two algorithms is significantly different if the corresponding average ranks differ by at least one critical difference (CD). Fig. 5 provides the CD diagrams of *Precision* and *Distance*, where the average rank of each algorithm is marked along the axis (lower ranks to the right). We observe that misLCS is the only method that achieves the lowest rank in all experimental results. This result provides strong evidence of the superior performance of misLCS in local causal structure learning with missing data.

In summary, misLCS not only improves the accuracy of missing data imputation, but also outperforms the four state-of-art local causal structure learning algorithms from above the analysis. These results validate that misLCS is a promising approach for local causal structure learning with missing data.

5.2.2. Why misLCS is effective?

In this section, we analyze the reason why misLCS is effective from the following two aspects. First, we evaluate the effectiveness of the iterative method missing value imputation. Second, we evaluate the accuracy of skeleton learning.

(1) Comparison of iterative and non-iterative missing value imputation.

For evaluating the performance of missing value imputation, we use the PCP (the Percentage of Correct Prediction) that is shown in Eq. (3). PCP is an accurate metric to evaluate the difference between real values and predicted values in missing value imputation.

$$PCP = 100 \times \frac{number\ of\ correct\ predictions}{total\ number\ of\ predicitons} \tag{3}$$

We compare the iterative missing value imputation with the non-iterative way on seven BNs within different missing rates (10%, 30%, 50%) and samples (500, 1000). The results of PCP are shown in

Table 5
Distance on seven BNs using different data sizes and missing rates.

Network	Algorithm	Size = 500						Size = 1000					
		10%	20%	30%	40%	50%	60%	10%	20%	30%	40%	50%	60%
Child	PCD-by-PCD	0.86	0.96	1.01	0.92	0.96	1.09	0.85	0.93	0.78	0.97	1.05	0.95
	MB-by-MB	0.82	0.81	0.94	0.96	1.08	1.27	0.70	0.66	0.85	0.82	1.09	1.03
	CMB	**0.68**	0.74	0.66	**0.66**	0.85	0.92	1.01	0.62	**0.53**	0.72	0.86	0.97
	ELCS	**0.68**	**0.68**	**0.65**	0.78	0.82	0.90	**0.51**	**0.58**	0.73	**0.68**	0.79	0.86
	misLCS	0.86	0.84	0.79	0.69	**0.79**	**0.83**	0.85	0.84	0.73	0.70	**0.73**	**0.86**
Alarm1	PCD-by-PCD	**0.28**	0.53	0.60	0.52	0.60	0.75	**0.23**	0.23	0.45	**0.43**	**0.60**	**0.61**
	MB-by-MB	0.30	0.48	0.51	0.68	0.74	0.88	0.36	0.43	0.49	0.55	0.68	0.84
	CMB	0.44	0.49	0.62	0.66	0.70	0.89	0.46	0.39	0.45	0.54	0.72	0.72
	ELCS	0.31	**0.33**	**0.44**	**0.38**	0.68	**0.76**	0.24	**0.22**	0.30	0.52	0.64	0.85
	misLCS	0.44	0.50	0.50	0.51	**0.67**	0.86	0.44	0.38	**0.25**	0.46	0.67	0.78
Mildew	PCD-by-PCD	0.92	0.94	0.94	**0.95**	**0.99**	1.00	0.90	0.89	0.92	0.92	**0.94**	**0.95**
	MB-by-MB	1.41	1.41	1.41	1.41	1.41	1.41	1.37	1.37	1.29	1.38	1.22	1.40
	CMB	1.09	1.10	1.13	1.14	1.17	1.19	1.00	1.01	1.02	1.03	1.07	1.09
	ELCS	1.27	1.27	1.29	1.30	1.29	1.30	1.26	1.25	1.27	1.28	1.28	1.30
	misLCS	**0.62**	**0.78**	**0.90**	0.98	1.04	**0.94**	**0.54**	**0.66**	**0.79**	**0.89**	1.08	1.05
Alarm3	PCD-by-PCD	**0.84**	1.03	0.91	**0.89**	1.04	**0.93**	0.80	0.75	0.92	0.92	0.95	1.00
	MB-by-MB	0.87	0.80	1.02	0.92	0.92	1.02	0.91	0.94	0.89	**0.77**	**0.85**	1.02
	CMB	0.95	1.06	0.92	0.98	1.06	1.04	0.86	0.89	0.76	0.82	0.91	1.02
	ELCS	0.85	0.85	**0.78**	0.92	1.00	1.09	0.78	**0.73**	0.75	0.75	**0.85**	1.00
	misLCS	0.94	**0.79**	0.96	1.00	**0.99**	0.99	**0.72**	0.77	**0.68**	0.87	1.00	**0.95**
HailFinder5	PCD-by-PCD	**0.77**	**0.88**	0.91	0.92	**0.78**	**0.95**	0.81	0.79	0.90	0.87	1.04	–
	MB-by-MB	1.15	1.18	1.13	1.12	1.09	1.06	0.94	0.98	0.92	0.93	0.91	–
	CMB	–	–	–	–	–	–	**0.63**	0.89	0.76	0.83	–	–
	ELCS	0.89	1.04	1.11	0.97	0.94	–	0.72	0.80	0.90	1.07	1.10	–
	misLCS	0.80	0.90	**0.69**	**0.75**	0.82	0.97	0.73	**0.74**	**0.70**	**0.70**	**0.71**	–
Pigs	PCD-by-PCD	0.23	0.30	0.39	0.33	0.71	0.81	0.23	0.45	0.29	0.34	0.61	0.65
	MB-by-MB	0.53	0.53	0.61	0.75	0.79	0.95	0.53	0.41	0.46	0.56	0.64	0.80
	CMB	0.33	0.44	0.46	0.70	0.72	**0.77**	0.38	0.33	0.39	0.54	0.64	0.75
	ELCS	**0.15**	**0.12**	**0.19**	0.32	0.72	0.81	**0.16**	0.30	0.26	0.26	0.72	0.77
	misLCS	0.22	0.17	0.28	**0.29**	**0.57**	0.88	0.19	**0.26**	**0.19**	**0.39**	**0.60**	**0.56**
Gene	PCD-by-PCD	0.89	0.69	0.85	1.03	0.92	1.07	0.78	0.81	0.76	1.01	0.98	0.88
	MB-by-MB	1.02	0.97	0.97	1.01	0.96	0.86	0.99	1.02	0.97	0.92	0.95	1.03
	CMB	0.47	0.49	0.58	**0.49**	0.84	1.01	0.51	0.43	0.56	0.67	0.67	0.69
	ELCS	**0.23**	0.38	0.44	0.60	**0.58**	**0.79**	0.53	0.31	**0.46**	**0.48**	**0.61**	**0.62**
	misLCS	0.29	**0.29**	**0.35**	0.72	0.78	0.92	**0.21**	**0.20**	0.58	0.69	0.76	0.85

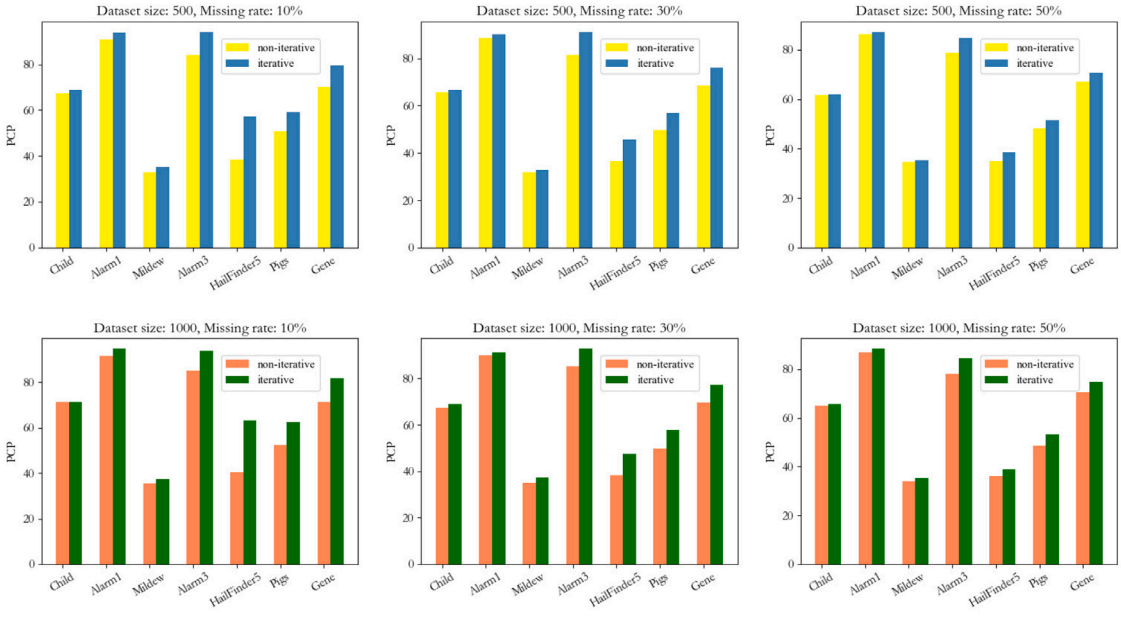
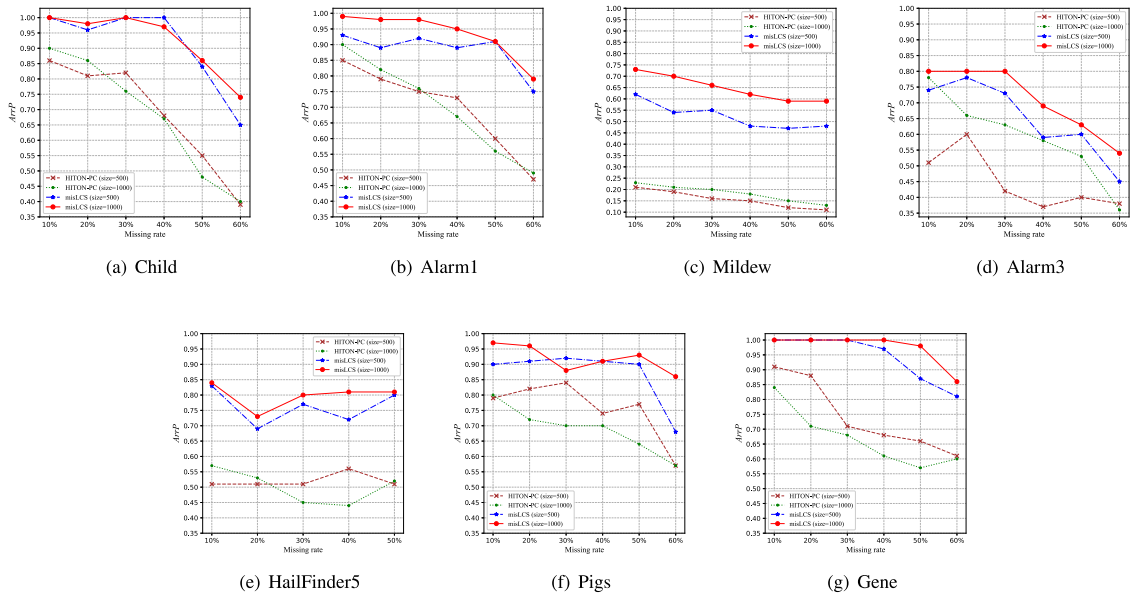


Fig. 6. The performance of missing value imputation of the iterative method and the non-iterative method.

Fig. 6. We observed that the iterative method consistently outperformed the non-iterative method in terms of PCP values. The iterative method's superior performance shows that it is more accurate than the non-iterative method in missing value imputation. It means that the iterative method can provide more accurate data for the local causal structure learning.

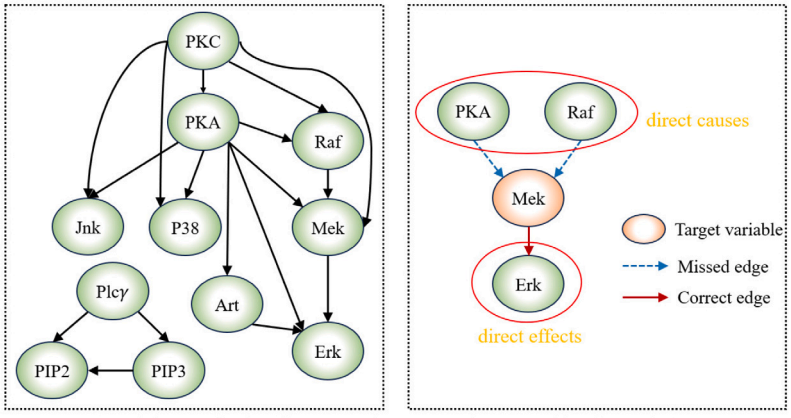
(2) Comparison of local skeleton learning between misLCS and HTION-PC.



(a) Child (b) Alarm1 (c) Mildew (d) Alarm3

(e) HailFinder5 (f) Pigs (g) Gene

Fig. 7. Comparison of the local skeleton learning between misLCS and HITON-PC.



(a) True DAG of the protein signaling network (b) Learned local causal structure

Fig. 8. True DAG of the protein signaling network and the learned local causal structure by misLCS (with the data missing rate of 10%).

During the skeleton learning process, misLCS aims to find the PC of the target variable. It has been proved that under certain assumptions (such as Causal Faithfulness Assumption and Causal Sufficiency Assumption), HITON-PC is capable of correctly learning the PC of any target variable (Yu et al., 2021). To quantitatively measure the distinction between misLCS and HITON-PC in skeleton learning, we use the $ArrP$ as our evaluation metric.

- $ArrP$: the number of true positives in the output (i.e., the variables in the output belonging to the true PC of a target variable in a test DAG) divided by the number of variables in the output of an algorithm.

Fig. 7 presents a precision comparison of skeleton learning between misLCS and HITON-PC. On the seven BNs with varying missing rates (ranging from 10% to 60%) and samples (500, 1000), the $ArrP$ value of misLCS consistently exceeds that of HITON-PC. This indicates that misLCS attains a notably higher accuracy in skeleton learning.

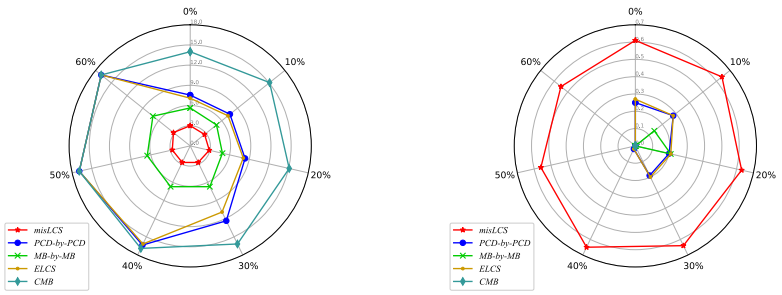
5.3. Real-world dataset

In addition to benchmark BN dataset, the performance on real-world datasets is also essential. We adopt a widely used bioinformatics dataset (Sachs et al., 2005) for the discovery of a protein signaling network based on the expression level of proteins and phospholipids. In this paper, a total of 7466 samples, 11 cell types, and 20 directed edges are utilized. And the true DAG of the protein signaling network is shown in Fig. 8(a).

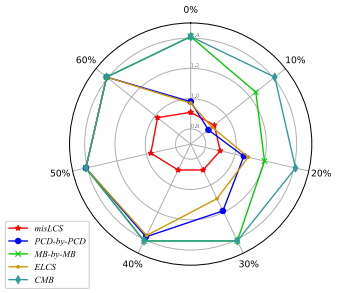
We consider the "Mek" variable as the target variable to learn its local causal structure. From Fig. 8(b), we observe that misLCS is able to correctly learn the direct effects of "Mek" under the data missing rate of 10%. We further show in Fig. 9 a comparison of misLCS with the other four local causal structure learning algorithms under MCAR settings. We observe that misLCS achieves a better performance on SHD, $Precision$, and $Distance$ within different missing rates (ranging from 0% to 60%).

6. Conclusion and future work

A novel local causal structure learning algorithm with missing data (misLCS) has been proposed in this paper, which improves the performance in distinguishing parents from children of a target variable of interest. misLCS designs the iterative data imputation method and data subset strategy to get a complete and correct data subset. And it constructs and identifies the causal direction of the target variable by



(a) SHD (b) *Precision*



(c) *Distance*

Fig. 9. Results on a real-world dataset: comparison of local causal structure learning with missing rates ranging from 0% to 60%.

applying feature selection and conditional independence tests. Extensive experimental results on seven benchmark BNs and a real-world bioinformatics dataset demonstrate that misLCS not only solves the problems in local causal structure learning with missing data but also achieves better performance in accuracy. In future, we would focus on: (1) designing a new missing data imputation method to enhance its accuracy, and (2) extending the idea of misLCS to global causal structure learning.

CRediT authorship contribution statement

Shaojing Sheng: Conceptualization, Methodology, Software, Writing – original draft. **Xianjie Guo:** Validation, Writing – review & editing, Formal analysis. **Kui Yu:** Resource, Supervision, Writing – review & editing, Formal analysis. **Xindong Wu:** Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the International Cooperation and Exchange of the National Natural Science Foundation of China (under grant 62120106008) and the National Natural Science Foundation of China (under grant 62376087).

References

Aliferis, C. F., Tsamardinos, I., & Statnikov, A. (2003). HITON: A novel Markov blanket algorithm for optimal variable selection. In *AMIA annual symposium proceedings*: *Vol. 2003*, (p. 21). American Medical Informatics Association.

Cai, R., Qiao, J., Zhang, Z., & Hao, Z. (2018). Self: Structural equational likelihood framework for causal discovery. In *Proceedings of the AAAI conference on artificial intelligence*: *Vol. 32*.

Cai, R., Wu, S., Qiao, J., Hao, Z., Zhang, K., & Zhang, X. (2022). THPs: Topological Hawkes processes for learning causal structure on event sequences. *IEEE Transactions on Neural Networks and Learning Systems*.

Chickering, D. M. (2002). Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 507–554.

Foraita, R., Friemel, J., Günther, K., Behrens, T., Bullerdiek, J., Nimzyk, R., Ahrens, W., & Didelez, V. (2020). Causal discovery of gene regulation with incomplete data. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *183*, 1747–1775.

Gao, T., & Ji, Q. (2015). Local causal discovery of direct causes and effects. *Advances in Neural Information Processing Systems*, *28*.

Geng, Z., Liu, Y., Liu, C., & Miao, W. (2019). Evaluation of causal effects and local structure learning of causal networks. *Annual Review of Statistics and its Application*, *6*, 103–124.

Guo, X., Wang, Y., Huang, X., Yang, S., & Yu, K. (2022). Bootstrap-based causal structure learning. In *Proceedings of the 31st ACM international conference on information & knowledge management* (pp. 656–665).

Guo, X., Yu, K., Liu, L., Li, P., & Li, J. (2023). Adaptive skeleton construction for accurate DAG learning. *IEEE Transactions on Knowledge and Data Engineering*, 1–14. http://dx.doi.org/10.1109/TKDE.2023.3265015.

Khan, W., Ansell, D., Kuru, K., & Bilal, M. (2018). Flight guardian: Autonomous flight safety improvement by monitoring aircraft cockpit instruments. *Journal of Aerospace Information Systems*, *15*, 203–214.

Kuang, K., Wang, H., Liu, Y., Xiong, R., Wu, R., Lu, W., Zhuang, Y. T., Wu, F., Cui, P., & Li, B. (2023). Stable prediction with leveraging seed variable. *IEEE Transactions on Knowledge and Data Engineering*, *35*, 6392–6404.

Lee, J., Jeong, J., & Jun, C. (2020). Markov blanket-based universal feature selection for classification and regression of mixed-type data. *Expert Systems with Applications*, *158*, Article 113398.

Lin, W., & Tsai, C. (2020). Missing value imputation: A review and analysis of the literature (2006–2017). *Artificial Intelligence Review*, *53*, 1487–1509.

Ling, Z., Yu, K., Wang, H., Liu, L., & Li, J. (2021). Any part of Bayesian network structure learning. arXiv preprint arXiv:2103.13810.

Margaritis, D., & Thrun, S. (1999). Bayesian network induction via local neighborhoods. *Advances in Neural Information Processing Systems*, *12*.

Meek, C. (2013). Causal inference and causal explanation with background knowledge. arXiv preprint arXiv:1302.4972.

Nogueira, A. R., Gama, J., & Ferreira, C. A. (2021). Causal discovery in machine learning: Theories and applications. *Journal of Dynamics & Games*, *8*, 203.
Pearl, J. (2009). *Causality*. Cambridge University Press.
Pearl, J. (2014). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan kaufmann.
Sachs, K., Perez, O., Pe'er, D., Lauffenburger, D. A., & Nolan, G. P. (2005). Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, *308*, 523–529.
Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, *23*, 2507–2517.
Sokolova, E., Groot, P., Claassen, T., Rhein, D. v., Buitelaar, J., & Heskes, T. (2015). Causal discovery from medical data: Dealing with missing values and a mixture of discrete and continuous data. In *Conference on artificial intelligence in medicine in Europe* (pp. 177–181). Springer.
Spirtes, P., Glymour, C. N., Scheines, R., & Heckerman, D. (2000). *Causation, prediction, and search*. MIT Press.
Triantafillou, S., Lagani, V., Heinze-Deml, C., Schmidt, A., Tegner, J., & Tsamardinos, I. (2017). Predicting causal relationships from biological data: Applying automated causal discovery on mass cytometry data of human immune cells. *Scientific Reports*, *7*, 1–11.
Tsamardinos, I., & Aliferis, C. F. (2003). Towards principled feature selection: Relevancy, filters and wrappers. In *International workshop on artificial intelligence and statistics* (pp. 300–307). PMLR.
Tsamardinos, I., Aliferis, C. F., & Statnikov, A. (2003). Time and sample efficient discovery of Markov blankets and direct causal relations. In *Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 673–678).
Tsamardinos, I., Aliferis, C. F., Statnikov, A. R., & Statnikov, E. (2003). Algorithms for large scale Markov blanket discovery. In *FLAIRS conference*: *Vol. 2*, (pp. 376–380). St. Augustine, FL.
Tsamardinos, I., Brown, L. E., & Aliferis, C. F. (2006). The max–min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, *65*, 31–78.
Tu, R., Zhang, C., Ackermann, P., Mohan, K., Kjellström, H., & Zhang, K. (2019). Causal discovery in the presence of missing data. In *The 22nd international conference on artificial intelligence and statistics* (pp. 1762–1770). PMLR.
Van Hulse, J., & Khoshgoftaar, T. M. (2014). Incomplete-case nearest neighbor imputation in software measurement data. *Information Sciences*, *259*, 596–610.
Verma, T. S., & Pearl, J. (2022). Equivalence and synthesis of causal models. In *Probabilistic and causal inference: The works of Judea Pearl* (1st ed.). (pp. 221–236). New York, NY, USA: Association for Computing Machinery.
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*, 1–36.
Wang, C., Zhou, Y., Zhao, Q., & Geng, Z. (2014). Discovering and orienting the edges connected to a target variable in a DAG via a sequential local learning approach. *Computational Statistics & Data Analysis*, *77*, 252–266.
Yang, J., Jiang, L., Xie, K., Chen, Q., & Wang, A. (2023). Lung nodule detection algorithm based on rank correlation causal structure learning. *Expert Systems with Applications*, *216*, Article 119381.
Yang, S., Wang, H., Yu, K., Cao, F., & Wu, X. (2021). Towards efficient local causal structure learning. *IEEE Transactions on Big Data*.
Yaramakala, S., & Margaritis, D. (2005). Speculative Markov blanket discovery for optimal feature selection. In *Fifth IEEE international conference on data mining* (p. 4). IEEE.
Yin, J., Zhou, Y., Wang, C., He, P., Zheng, C., & Geng, Z. (2008). Partial orientation and local structural learning of causal networks for prediction. In *Causation and prediction challenge* (pp. 93–105). PMLR.
Yu, Y., Chen, J., Gao, T., & Yu, M. (2019). DAG-GNN: DAG structure learning with graph neural networks. In *International conference on machine learning* (pp. 7154–7163). PMLR.
Yu, L., & Liu, H. (2004). Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, *5*, 1205–1224.
Yu, K., Liu, L., & Li, J. (2021). A unified view of causal and non-causal feature selection. *ACM Transactions on Knowledge Discovery from Data*, *15*, 1–46.
Zheng, X., Aragam, B., Ravikumar, P. K., & Xing, E. P. (2018). Dags with no tears: Continuous optimization for structure learning. *Advances in Neural Information Processing Systems*, *31*.