Improving Efficiency and Accuracy of Causal Discovery using a Hierarchical Wrapper

Abstract

Causal discovery from observational data is an important tool in many branches of science. Under certain assumptions it allows scientists to explain phenomena, predict, and make decisions. In the large sample limit, sound and complete causal discovery algorithms have been previously introduced, where a directed acyclic graph (DAG), or its equivalence class, representing causal relations is searched. However, in real-world cases, only finite training data is available, which limits the power of statistical tests used by these algorithms, leading to errors in the inferred causal model. This is commonly addressed by devising a strategy for using as few as possible statistical tests. In this paper, we introduce such a strategy in the form of a recursive wrapper for existing constraint-based causal discovery algorithms, which preserves soundness and completeness. It recursively clusters the observed variables using the normalized min-cut criterion from the outset, and uses a baseline causal discovery algorithm during backtracking for learning local sub-graphs. It then combines them and ensures completeness. By an ablation study, using synthetic data, and by common real-world benchmarks, we demonstrate that our approach requires significantly fewer statistical tests, learns more accurate graphs, and requires shorter run-times than the baseline algorithm.

1 INTRODUCTION

[Glymour et al., 2019] A fundamental task in various disciplines of science is to discover causal relations among domain variables [Glymour et al., 2019, Shen et al., 2020]. In many cases, the causal relations can be properly represented by a DAG [Pearl, 2009]. Then, by interpreting this causal DAG as a statistical model, many of these causal relations can be discovered using observational data alone [Spirtes et al., 2000, Pearl and Verma, 1991, Peters et al., 2017], known as causal discovery. In constraint-based causal discovery algorithms, statistical independence is tested between pairs of variables conditioned on subsets of the remaining domain variables [Spirtes et al., 2000, Colombo et al., 2012, Claassen et al., 2013, Tsamardinos et al., 2006, Yehezkel and Lerner, 2009, Cheng et al., 2002]. As not all causal relations can be discovered purely from these statistical tests using observational data, these algorithms return an equivalence class of the true underlying DAG. Nevertheless, constraint-based algorithms are generally proven to be asymptotically correct. In this paper, we will consider this family of algorithms.

In most real-world cases, limited observational data is available and statistical tests are prone to errors. Moreover, statistical tests for conditional independence (CI) often suffer from the curse-of-dimensionality. Tests with large condition sets are more prone to errors than tests with smaller condition sets. Thus, a common principle in constraint-based algorithms is to derive the next CI tests to perform, from the result of previous CI tests of smaller condition sets [Spirtes et al., 2000]. Another challenge is that learning causal DAGs from observed data is NP-hard [Chickering et al., 2004]. The number of possible DAGs grows super-exponentially with the number of domain variables, posing a serious limitation on the expected computational complexity of algorithms for real-world applications. On one hand, it is assumed that enough data points are available such that the statistical test results will be reliable, but on the other hand, the computational complexity of these statistical tests increases with the number of data points. Thus, the number of statistical tests commonly serves as a measure of computational complexity.

The common approach to address these problems is to reduce the overall number of CI tests required by the algorithm, and favor those that have greater statistical power. In this paper, we propose a wrapper—hierarchical clustering

Submitted to the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021). **To be used for reviewing only**.

for causal discovery (HCCD)—for existing casual discovery algorithms, referred in the paper as baseline algorithms. This wrapper recursively clusters the domain variables, thus limiting the condition set size of CI tests within each cluster, which alleviates the curse-of-dimensionality. That is, the wrapper relies on the level of correlation between variables, as apposed to the Boolean result of the statistical CI tests. Using spectral clustering, sub-domains are derived with respect to the *relative* correlation between variables, recursively. Once a (sub-) domain cannot be divided into smaller sub-domains, a baseline causal discovery algorithm is called. Tracing back from the recursion, the causal graphs for each sub-domain are merged and the baseline algorithm is called again on the merged graph for the edges between sub-domains (inter-domain), retaining edges within each sub-domain (intra-domain). The proposed wrapper improves accuracy in common finite datasets while preserving the soundness and completeness of the baseline algorithm.

2 BACKGROUND

Constraint-based algorithms for causal discovery rely on the correctness of statistical tests for conditional independence. In practice, as only limited data is available, these tests are prone to errors, and often suffer from the curse-of dimensionality. The PC algorithm [Spirtes et al., 2000] iteratively refines an initial fully-connected graph. In each iteration, connected nodes are tested for independence conditioned on a subset of their neighbors, where this subset is restricted to a constant size. The edge is removed if an independence is found. The restriction on the condition set size is increased by one in the next iteration. Thus, this approach has an advantage when CI tests with smaller condition sets are more reliable than CI tests with larger condition sets. The RAI algorithm [Yehezkel and Lerner, 2009] follows this approach but relies heavily on information from CI tests with smaller condition sets. It orients the graph's edges and decomposes it into sub-graphs before additional CI testing. Thus, errors in earlier stages may cause errors in later stages. Other works [Cai et al., 2017, Aliferis et al., 2010, Xie and Geng, 2008] also leverage divide-and-conquer or local-search strategies in a hierarchical or recursive way, and report improved results by partitioning the nodes into subsets and learning local structure for each.

A line of work [Sondhi and Shojaie, 2019, Chickering and Meek, 2015] propose to leverages properties of the graph to improve running time. Previously, it was shown that relying on correlation level between pairs of variables, in addition to the Boolean result of the CI tests, can reduce the overall number of CI tests and improve accuracy. The TPDA algorithm [Cheng et al., 2002], having a complexity of $O(n^4)$ (n is the number of nodes), relies on the monotone-DAG-faithfulness assumption. It assumes that the mutual information between any pair of nodes cannot decrease by opening more dependency inducing paths. Nevertheless, although TPDA has lower complexity than algorithms that do not utilize the correlation level among variables, it performs more CI tests having large condition sets, rendering it unstable for limited data [Tsamardinos et al., 2006]. Recently, it was proposed to utilize inhomogeneity in the domain as a heuristic for improving accuracy and speed of existing causal discovery algorithms [Pashami et al., 2018, Zhang et al., 2018]. For example, TSCB [Zhang et al., 2018] is a 2-step wrapper algorithm that first clusters the domain variables invoking an existing causal discovery algorithm for each cluster, and then applies the same causal discovery algorithm to inter-cluster edges. However, it is not clear under which conditions clustering-based wrappers retain soundness and completeness of the baseline algorithm, and under which conditions they are faster and learn a more accurate graph than the baseline algorithm.

In this paper we discuss the implication of domain variables clustering on the soundness and completeness of causal discovery. We also discuss the properties of such clustering that may reduce or increase the probability of errors and the overall efficiency.

3 VARIABLES CLUSTERING FOR CAUSAL DISCOVERY

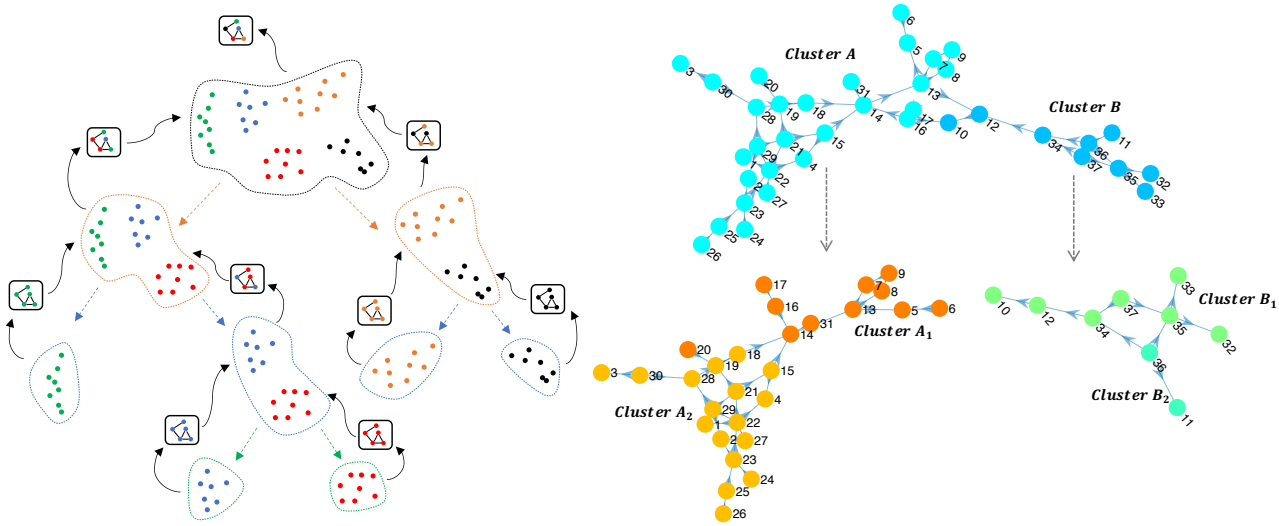
As discussed, dividing the set of variables into smaller subsets can be appealing for causal discovery algorithms. However, in the common scenario where the underlying causal DAG is connected, an optimal clustering, from which a causal discovery algorithm can benefit, is not clear.

Constraint-based algorithms often rely on the causal Markov and faithfulness assumptions [Spirtes et al., 2000]. A probability distribution P and a DAG $\mathcal{G}$ are said to be faithful to one another if in P, variables A and B are independent conditioned on set $\boldsymbol{Z}$ if and only if A and B are d-separated by $\boldsymbol{Z}$ in $\mathcal{G}$, $A \perp\!\!\!\perp B|\boldsymbol{Z}$. It is then key in constraint-based algorithms to identify conditional independence relations for constructing the underlying graph. Let Alg be a causal discovery algorithm. Let $\mathrm{ClustCD}$ (Cluster Causal Discovery) be the following procedure. 1) Given observed data for domain variables $\boldsymbol{X} = \{A, B, \ldots\}$, partition $\boldsymbol{X}$ into k disjoint subsets $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$, i.e., $\cup_{i=1}^{k} \boldsymbol{X}_i = \boldsymbol{X}$ and $\boldsymbol{X}_i \cap \boldsymbol{X}_j = \emptyset, \forall i \neq j$. 2) Call Alg for each $\boldsymbol{X}_i$ (intra-cluster). 3) Call Alg for edges between any pair (A, B) such that $A \in \boldsymbol{X}_i$ and $B \in \boldsymbol{X}_j$, for all $i, j \in \{1, \ldots, k\}, i \neq j$ (inter-cluster).

Theorem 1. *If* Alg *is a sound and complete causal discovery algorithm, then procedure* $\mathrm{ClustCD}$ *is sound for* $\boldsymbol{X}$*, but not complete.*

Proof. The proof is given in Appendix A. □

Given a probability distribution P faithful to DAG $\mathcal{G}$, a



(a) (b)

Figure 1: (a) An illustration of a top-down 2-way clustering of feature set followed by and a bottom-up causal discovery. The domain variables are clustered hierarchically. Then, from the leaves upwards, causal discovery (shown as a colored graph) is applied disjointly to the variables in each cluster, and then the resultant graphs are unified at their parent cluster, conditioned on their white-lists as depicted by those graphs. This process is backtracked until the root of the cluster tree. Best viewed in color. (b) An example to the a top-down 2-way clustering of the ALARM dataset's domain variables. In the first level, HCCD creates 2 clusters $\{A, B\}$. Then, for each cluster a recursive call is evaluated, and clusters $\{A_1, A_2\}$ and $\{B_1, B_2\}$ are created, respectively. The nodes within each cluster are highlighted by a different color, and presented on the ground truth structure. Best viewed in color.

complete algorithm can identify from observed data of $\boldsymbol{X}_i$ the conditional independence relation between a pair $A, B \in \boldsymbol{X}_i$, not adjacent in $\mathcal{G}$, if there is at least one separating set, $Z \in \boldsymbol{X}_i$, i.e. $A \perp\!\!\!\perp B | Z$, where A, B, Z are disjoint sets. In general, it is not guaranteed that a partition of $\boldsymbol{X}$ into disjoint subsets (clustering) exists such that at least one separating set for every pair of conditionally independent variables are in the same cluster. Of course, there are cases where such a clustering does exists; for example, the clustering $\{A\}, \{B\}, \{C, D, E\}$ when the underlying graph is $A \to D \leftarrow C \to E \leftarrow B$. Now, consider the case of two clusters. In one extreme, the first cluster contains a single variable, and the second cluster contains the remaining variables. In such a case, the expected number of undetectable intra-cluster independence relations is minimized. However, the complexity of the number of independence tests is maximal. On the other extreme, the two clusters have equal number of variables. This minimizes the complexity of the number of independence tests performed by the algorithm. For example, the complexity of the PC algorithm is $O(n^m)$, (m is the maximal in-degree), so if one cluster has n_1 variables and the other $n - n_1$, then $O(n_1^m) + O((n - n_1)^m)$ is minimal for $n_1 = n/2$. However, the expected number of undetectable intra-domain independence relations is maximal. A clustering method used by procedure ClustCD should balance minimizing the number of undetectable independence relations and the complexity of CI tests. For reducing the number of CI tests in the typical case, we assume that unconnected pairs of variables in $\mathcal{G}$ are more correlated to nodes of the minimal separating set, relative to other nodes.

Assumption 1. *Let I be a pairwise symmetric correlation function. For every disjoint pairs of nodes (X, Y) in the true underlying graph, such that $X \perp\!\!\!\perp Y | \boldsymbol{Z}$, where $\boldsymbol{Z}$ is a minimal separating set, there exists $\boldsymbol{V} \subset \boldsymbol{X} \setminus (\{X, Y\} \cup \boldsymbol{Z})$, called a redundant set, such that*

$$\min_{Z \in \boldsymbol{Z}} \left[\max\left[I(X, Z), I(Y, Z)\right]\right] \geq I(X, Y) > \min_{V \in \boldsymbol{V}} \left[\max\left[I(X, V), I(Y, V)\right]\right].$$

This assumption is derived as follows. Let $X \perp\!\!\!\perp Y | \boldsymbol{Z}$, where $\boldsymbol{Z}$ is a minimal separating set. For a constraint-based causal discovery algorithm to identify this independence, it is essential that every $Z \in \boldsymbol{Z}$ is in the same cluster that includes X and Y. To ensure this, every $Z \in \boldsymbol{Z}$ should have a correlation level with X or Y, at least as the correlation level between X and Y. That is, $\forall Z \in \boldsymbol{Z}$, $I(Z, X) > I(X, Y)$ or $I(Z, Y) > I(X, Y)$. Thus, if X and Y are in the same cluster, $\boldsymbol{Z}$ is also in that cluster. This is formally expressed by the first relation of Assumption 1: $\min_{Z \in \boldsymbol{Z}}[\max[I(X, Z), I(Y, Z)]] \geq I(X, Y)$, where $\min_{Z \in \boldsymbol{Z}}$ essentially represents "$\forall Z \in \boldsymbol{Z}$". The second relation in Assumption 1 is $I(X, Y) > \min_{V \in \boldsymbol{V}}[\max[I(X, V), I(Y, V)]]$, where $\boldsymbol{V}$ is a set that does not include any $Z \in \boldsymbol{Z}$, X, and, Y. This relation

assumes that the variables can be clustered. If no such redundant set, $\boldsymbol{V}$, exists it means that every variable in $\boldsymbol{X} \setminus (\{X, Y\} \cup \mathbf{Z})$ will have a stronger correlation with X or Y than the correlation between X and Y. Thus, if X and Y are in the same cluster, then all other variables will be in the same cluster as well.

Assumption 1 is required for achieving efficiency[1] in the number of CI tests, and balances between: 1) allowing minimal separating sets to be discovered by Alg applied to a cluster, and 2) partitioning the variables into clusters.

3.1 DOMAIN VARIABLE CLUSTERING

We now derive a clustering approach that complies with Assumption 1. Consider a fully connected undirected graph $\mathcal{U}$ over the domain variables $\boldsymbol{X}$. A symmetric similarity matrix $\boldsymbol{W}$ represents the weights of the edges in $\mathcal{U}$. The value of $\boldsymbol{W}_{i,j}$ is the weight of the edge between nodes $X_i, X_j \in \boldsymbol{X}$ and represents the correlation "strength" between these variables. The weight is the statistical measure of correlation, denoted I, and calculated by the statistical independence test that is used by the baseline causal discovery algorithm. For example, mutual information for discrete variables and correlation coefficient for continuous variables. Clustering can then be viewed as partitioning $\mathcal{U}$ into disjoint sub-graphs $\mathcal{U}_1, \ldots, \mathcal{U}_k$ by removing edges connecting the sub-graphs, where a cluster $\boldsymbol{X}_i$ consists of the nodes in sub-graph $\mathcal{U}_i$. Partitioning $\mathcal{U}$ by minimizing the sum of weights of removed edges violates Assumption 1, as discussed later. Moreover, as this sum increases with the number of removed edges, clustering algorithms based on this criterion favor creating small clusters of isolated nodes [Wu and Leahy, 1993]. As a solution, we follow Shi and Malik [2000] that proposed the k-way normalized cut (Ncut),

$$\mathrm{Ncut}(\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k\}) = \\ = \sum_{i=1}^{k-1} \sum_{j=k+1}^{k} \mathrm{cut}(\boldsymbol{X}_i, \boldsymbol{X}_j) / \mathrm{assoc}(\boldsymbol{X}_i, \boldsymbol{X}), \quad (1)$$

where $\mathrm{assoc}(\boldsymbol{X}_i, \boldsymbol{X})$ is the sum of weights of edges connecting each node in cluster i to every other node in $\boldsymbol{X}$, and $\mathrm{cut}(\boldsymbol{X}_i, \boldsymbol{X}_j)$ is the sum of weights of edges connecting each node in cluster i to every node in cluster j.

This criterion complies with Assumption 1. Let $X \perp\!\!\!\perp Y | \boldsymbol{Z}$ where $\boldsymbol{Z}$ is a minimal separating set. Now, consider an undesired clustering: $\boldsymbol{X_1} = \{X, Y\}$ and $\boldsymbol{X_2} = \boldsymbol{Z}$. Then, $\mathrm{Ncut}(\boldsymbol{X}_1, \boldsymbol{X}_2) = I(X,Z)+I(Y,Z) / I(X,Z)+I(Y,Z)+I(X,Y)$. To avoid such clustering, it is required to maximize the Ncut value $\forall Z \in \mathbf{Z}$. It is easy to see that this value is greater when $I(X, Z) > I(X, Y)$ than the value when $I(X, Z) < I(X, Y)$ and similarly for $I(Y, Z) > I(X, Y)$. Thus, this criterion complies with Assumption 1. It is important to note that a criterion equal to the numerator of Equation 1 does not support Assumption 1, as it ignores $I(X, Y)$. In addition, Ncut diminishes the creation of small clusters. In fact, in the extreme case of equal weights for all edges, Ncut is minimized for clusters with equal sizes.

Shi and Malik [2000] showed that minimizing 2-way Ncut is equivalent to

$$\min_{u} {}^{(u^{\mathrm{T}} \boldsymbol{L} u)}/_{(u^{\mathrm{T}} \boldsymbol{D} u)} \qquad \text{s. t.} \quad u^{\mathrm{T}} \boldsymbol{D} 1 = 0, \qquad (2)$$

where u is an indicator vector of length n, $\boldsymbol{D}$ is a diagonal matrix with elements $\boldsymbol{D}_{i,i} = \sum_{j=1}^{n} \boldsymbol{W}_{i,j}$, and $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is the Laplacian matrix. In our case, we can relax u to take on real values, and the criterion can be minimized by solving the generalized eigenvalue system, $(\boldsymbol{D} - \boldsymbol{W})u = \lambda \boldsymbol{D} u$. Taking the eigenvector corresponding to the smallest non-zero eigenvalue minimizes

$$(\textstyle\sum_{i,j} \boldsymbol{W}_{i,j} (u_i - u_j)^2) / (\sum_i \boldsymbol{D}_{i,i} u_i^2). \qquad (3)$$

A Laplacian eigenmap [Belkin and Niyogi, 2003] is formed by concatenating the m eigenvectors corresponding to the lowest non-zeros eigenvalues, $\tilde{\boldsymbol{u}} = [u^1, \ldots, u^m]$. Thus, each domain variable $X_i \in \boldsymbol{X}$ is represented by a point $\tilde{\boldsymbol{u}}_{(i,\cdot)}$ in $\mathbb{R}^m$. For our task, from Equation 3, variables that are strongly correlated, *relatively* to other pairs, will have a relatively small Euclidean distance $\mathbb{R}^m$. Finally, points $\tilde{\boldsymbol{u}}$, representing variables $\boldsymbol{X}$ in $\mathbb{R}^m$, are clustered using k-means++ [Arthur and Vassilvitskii, 2007]. This procedure is known as spectral clustering.

3.2 PROPOSED METHOD

We consider the problem of learning a causal model, given a dataset for n domain variables, $\{X_i\}_{i=1}^n$. Our method is composed of two main stages, commencing with a top-down hierarchical clustering stage followed by a bottom-up causal discovery in the backtracking stage.

In the first stage, hierarchical clustering aims to alleviate the curse-of-dimensionality by partitioning the variable set into clusters, each of which potentially contains variables that are statistically related to each other to a large extent, thereby avoiding spurious connectivity to weaker and undesirable variables (Assumption 1).

Our method starts off by clustering the entire variable set from the outset into a number of clusters (see Section 3.1), and thereafter successively clusters each of the resultant clusters furthermore, independently of the other clusters. This successive independent clustering process continues for each sub-cluster recursively, forming a tree of clusters, until a separability condition is met (explained later), at which point the entire variable set is clustered to subsets of variables. Figure 1(a) illustrates this process. We postulate

[1] Soundness and completeness of the method described in this paper does not rely on this assumption.

that each such variable set has a high probability to manifest some structural motif [Milo et al., 2002, Yang et al., 2018]. A separability condition is used to determine the termination of the hierarchical clustering, and for that the eigenvalues of the graph's Laplacian are used (Equation 2). Generally, those close to zero correspond to isolated groups in the graph, and therefore if more than one such eigenvalue exists then the variable set of the sub-cluster is likely to contain more (relatively) disjoint groups within it, hence the hierarchical clustering process continues. In this case, the number of clusters for the next recursion call is the number of Laplacian's eigenvalues that are close to zero (k' in Algorithm 1, line 9). Experimentally, it was observed that this criterion mostly terminates the clustering at optimal points.

In the second stage, a bottom-up causal discovery algorithm, denoted Alg, is applied to the sub-clusters, starting from the leaves of the cluster tree, and moving upwards towards the root of the tree. Alg is applied to the variable set of each sub-cluster independently to the other sub-clusters, assuming that being secluded and isolated by other irrelevant variables from other sub-clusters, is more probable to learn graphical models with reliable edges, i.e. with higher degree of certainty. In this paper we use the PC algorithm as Alg. Even though the PC algorithm was chosen as the baseline algorithm, other constraint-based causal discovery methods [Tsamardinos et al., 2006, Rohekar et al., 2018, Colombo et al., 2012] may be used and arguably improved, since this stage poses no assumptions or restrictions on the elements of any prospect method.

After learning a graph for each sub-cluster, it is thereafter represented as a sub-graph. Further on, adjacent sub-graphs (those belonging to the same parent cluster) are backtracked in tandem upwards to their parent cluster, at which point they are merged as a single unified variable set. For that, edges are added between every node in one sub-cluster to every node in the other sub-clusters, and a list, $\mathcal{E}$, is formed from these added edges. That is, $\mathcal{E}$ lists the edges of bipartite graphs between every pair of sub-clusters. Alg is applied to the unified variable set and only the edges listed in $\mathcal{E}$ are tested for removal. That is, Alg does not consider new connections or removal of edges between any pair of variables within each sub-cluster. Ultimately, each sub-graph keeps its intra-cluster connectivity, presumably stemming from a more reliable variable set, and appends new inter-cluster connectivity, which were not taken into consideration at the former stage. Then, for preserving the completeness of Alg, we apply Alg again to the unified variable set, this time considering all the remaining edges. The above process continues upwards the clusters tree and terminates after been applied at the root, at which point the final graph is formed from the entire variable set.

Note that Alg is required to learn about the edges in list $\mathcal{E}$. For the case of Alg =PC-algorithm, we set the conditional-independence test function to return a result only for edges in $\mathcal{E}$. For edges not in this list, the function simply returns the existence or absence of the edge in the current graph as "dependent" or "independent" respectfully.

The main purpose of our approach is to improve the accuracy and efficiency of a given baseline algorithm by reducing the number of (unique) statistical tests, while maintaining the soundness and completeness of the baseline algorithm. Although we run the baseline algorithm on all the clusters in the backtracking phase, this inclusion will not undo any advantages of the clustering in terms of efficiency and accuracy. The reason for this is that each of the clusters is effectively: (a) containing only part of the nodes, so many conditional tests are avoided, and (b) sparser, since some edges were already removed and would not be tested anymore, and (c) condition tests that were already applied in previous steps would not be reapplied. So in essence, effectively we only avoid applying unnecessary condition tests, and consequently improve both speed and accuracy.

An improvement in performance of our method over a baseline is expected when Assumption 1 is complied, and this improvement is maximal when the sizes of the clusters are equal. As more clusters are revealed (while preserving causal sufficiency) the greater the improvement.

An additional virtue of our method is parallelism, that can be applied to successive independent clustering during the top-down stage, as well as to the causal discovery in independent sub-clusters during the bottom-up stage. The method is illustrated in Figure 1(a), and presented as Algorithm 1. Figure 1(b) exemplifies the top-down 2-way clustering of the ALARM dataset

Theorem 2. *Let* Alg *be a causal discovery algorithm that take as input an initial graph and a list of edges to be learned. If* Alg *is sound and complete, then Algorithm 1 is sound and complete.*

Proof. The proof does not rely on Assumption 1 (applies to arbitrary partitions). The proof is given in Appendix A. □

Completeness of our approach is achieved by calling the sound and complete algorithm $\mathrm{Alg}(\boldsymbol{X}, \mathrm{edges}(\mathcal{G}_{\boldsymbol{X}}))$ for refining the result $\mathcal{G}_{\boldsymbol{X}}$ of the merged cluster (Algorithm 1, line 18). In this call, all the graph edges are considered for learning, allowing undetected independence-relations withing the clusters to be detected.

4 EXPERIMENTS

First, we evaluate several aspects of the HCCD wrapper using synthetically generated data. The process we used for generating the data is detailed in Appendix B1. In addition, in appendix B2 we examine the gain achieved by the recursion, and the effect that the completeness requirement has on the accuracy. Then, we evaluate qualitative measures of graphs learned using publicly available datasets. In all

Algorithm 1: HCCD: Hierarchical clustering for causal discovery

Input: $\boldsymbol{X}$: set of nodes, $\boldsymbol{W}$: similarity matrix between pairs of nodes, k: number of clusters
Output: $\mathcal{G}_{\boldsymbol{X}}$: a graph representing an equivalence class of causal models

▷ **Exit condition**
if $k \leq 1$ *or* $|\boldsymbol{X}| < min_cluster_size$ ▷ exit condition
then
  $\mathcal{G}_{\text{init}} \leftarrow$ an uninformative graph over $\boldsymbol{X}$ (e.g., a complete graph if Alg is the PC algorithm)
  call causal discovery algorithm $\mathcal{G}_{\boldsymbol{X}} \leftarrow \text{Alg}(\boldsymbol{X}, \text{Edges}(\mathcal{G}_{\text{init}}), \mathcal{G}_{\text{init}})$ ▷ intra-cluster; $\text{Edges}(\mathcal{G})$ returns a list of edges from $\mathcal{G}$
  return graph $\mathcal{G}_{\boldsymbol{X}}$
end
▷ **Recursive calls: clustering**
call spectral clustering, using $\boldsymbol{W}$, to decompose $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k\} \leftarrow \boldsymbol{X}$
for i *in* $\{1, \ldots, k\}$ **do**
  $k' \leftarrow f(k, \boldsymbol{W}(\boldsymbol{X}_i))$ ▷ derive the number of clusters
  call: $\mathcal{G}_{\boldsymbol{X}_i} \leftarrow \text{HCCD}(\boldsymbol{X_i}, \boldsymbol{W}(\boldsymbol{X}_i), k')$ ▷ recursive HCCD call
end
▷ **Recursive back-tracking: causal discovery**
create a graph by merging the sub-graphs $\mathcal{G}_{\text{init}} \leftarrow \bigcup_{i=1}^{k} \mathcal{G}_{\boldsymbol{X}_i}$
$\mathcal{E} \leftarrow \emptyset$ ▷ initialize list of edges to be learned
for i *in* $\{1, \ldots, k\}$ **do**
  for each (A, B) such that $A \in \boldsymbol{X}_i$ and $B \in \boldsymbol{X} \setminus \boldsymbol{X}_i$, add (A, B) to $\mathcal{E}$, and add an uninformative edge to $\mathcal{G}_{\text{init}}$
end
call causal discovery algorithm $\mathcal{G}'_{\boldsymbol{X}} \leftarrow \text{Alg}(\boldsymbol{X}, \mathcal{E}, \mathcal{G}_{\text{init}})$ ▷ inter-cluster
call causal discovery algorithm $\mathcal{G}_{\boldsymbol{X}} \leftarrow \text{Alg}(\boldsymbol{X}, \text{Edges}(\mathcal{G}'_{\boldsymbol{X}}), \mathcal{G}'_{\boldsymbol{X}})$ ▷ for completeness
return graph $\mathcal{G}_{\boldsymbol{X}}$

our experiments, Alg is PC, a sound and complete algorithm [Spirtes et al., 2000]. Although it relies on the causal sufficiency assumption, it is often used as a first step of causal discovery in the presence of latent confounders and selection bias [Spirtes et al., 2000, Claassen et al., 2013, Colombo et al., 2012].

4.1 AN ANALYSIS USING SYNTHETIC DATA

In this section we evaluate the performance of the HCCD with respect to the number of training samples, and to the number of nodes in the graphs, compared to the baseline method (PC). For that, we measure the behaviour of 5 key aspects: 3 metrics of structural correctness - SID score [Peters and Bühlmann, 2015], structural Hamming distance (SHD), and causal accuracy. In addition, we measure the number of CI tests, and the run-time of the method, including the clustering.

Figure 2 shows the performance of the HCCD wrapper with respect to the number of training samples, for graphs with $n = 100$ nodes. The figures show mean $\pm$ std of 500 independent tests (DAGs), and values are normalized by the PC score in order to visualize the improvement over the baseline method. Additional experiments, for $n \in \{20, 50, 200, 1000\}$, are presented in Appendix C. It is evident that the HCCD wrapper is superior to the baseline for all 3 structural correctness metrics along the entire range

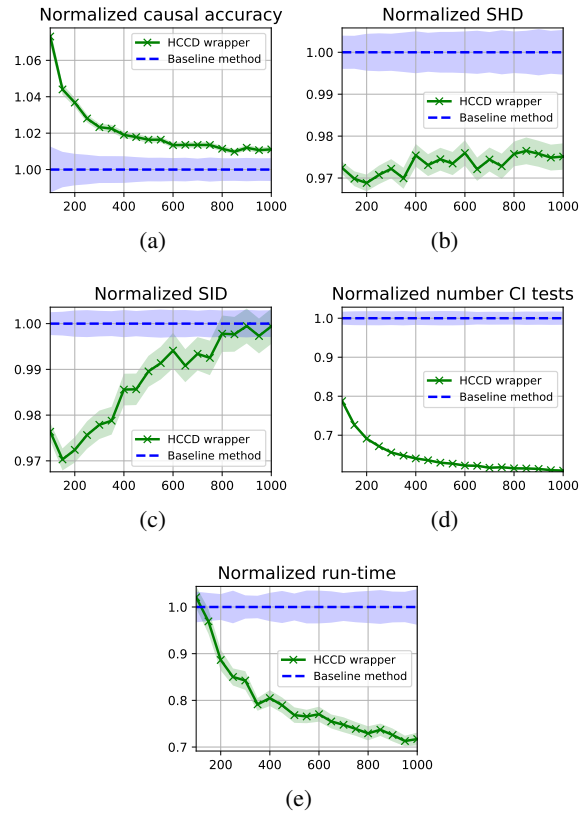


Figure 2: Performance of the HCCD wrapper, relatively to the baseline (PC), as a function of the number of training samples, for 100 graph nodes. Values are average over 500 DAGs, and normalized by the PC score. (a) Causal accuracy (higher is better); (b) SHD (lower is better); (c) SID (lower is better); (d) Number of CI tests (lower is better); (e) Run-time (lower is better). The HCCD wrapper achieves improvements in all the metrics.

of the training set size and for every n. In addition, there is an evident saving in the number of CI tests, and importantly in run-time (includes the clustering stage) along the entire range of the training set size. One exception is for the case of $n = 20$, for which the HCCD run-time is higher. This is expected since the run-time overhead of the clustering stage overtakes the saving in run-time gained by using fewer statistical tests in datasets with a small number of nodes. Nevertheless, for the common real-world cases, datasets having many variables, the HCCD wrapper achieves a significant reduction in run-time. Additionally, it is evident that the run-time reduction increases with the increase of the number of training samples, i.e. larger training sets benefit from a greater decrease in run-time.

Figure 3 shows the performance of the HCCD wrapper with respect to n, the number of nodes, for 500 training samples. The figures show mean $\pm$ std of 500 independent tests (DAGs), and values are normalized by the PC score in order to analyze the improvement over the baseline method. It is

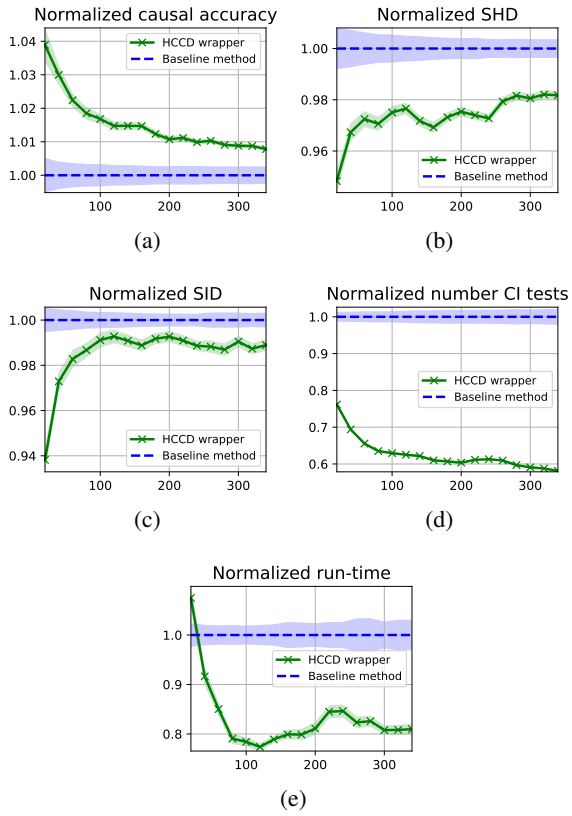


Figure 3: Performance of the HCCD wrapper, relatively to the baseline (PC), as a function of the number of graph nodes, for 500 training samples. Values are average over 500 DAGs, and normalized by the PC score. (a) Causal accuracy (higher is better); (b) SHD (lower is better); (c) SID (lower is better); (d) Number of CI tests (lower is better); (e) Run-time (lower is better). The HCCD wrapper achieves improvements in all the metrics.

evident that the HCCD wrapper is superior to the baseline for all 3 structural correctness metrics along the entire range of n. Moreover, saving in number of CI tests is evident, and importantly a reduction in run-time (which includes the clustering stage) for the entire range of n.

4.2 REAL-WORLD DATA

In this section we evaluate and compare the accuracy of our method over 10 publicly available datasets from the bnlearn package [Scutari, 2010], and 1 dataset from the Neuropathic Pain Diagnosis Simulator [Tu et al., 2019], all of which represent real decision support systems that cover a wide range of real life applications, such as medicine, agriculture, weather forecasting, financial modeling and animal breeding. Each of those datasets consists of 10 training sets, having 500 samples each, and corresponding 10 separate test sets, having 5000 samples each, for evaluating several qualitative measures of structural correctness. Thus, for each of the 11 datasets, the experiments were repeated 10 times. The number of domain variables across the datasets spans from tens to hundreds.

The first metric we measure is the BDeu score [Chickering et al., 1995], which under certain assumptions corresponds to the posterior probability of the learned graph. Tsamardinos et al. [2006] noted that this score does not rely on the true graph and may not be related to it, as it is not known in practice to what extent its underlying assumptions hold (e.g., Dirichlet distribution of the hyper-parameters). Nevertheless, since this score does not require knowing the true graph, it has a great value in practical situations. Moreover, this score is often used to tune the baseline parameters [Yehezkel and Lerner, 2009]. Figure 4 shows a scatter plot of normalized BDeu score, comparing HCCD, TSCB, and PC, evaluated on the 11 datasets, each consists of the 10 different training and test sets (total of 100 points). The BDeu scores are normalized by the PC BDeu score, and so a lower normalized score is better. In 97% of the cases, HCCD is better than PC. In 82% of the cases, TSCB is better than PC. Lastly, in 90% of the cases, HCCD is better than TSCB. As evident from the figure, HCCD is superior to the other methods. Additionally, for each complete dataset, the mean $\pm$ std BDeu score (unnormalized) is presented in Table 1, and better results for the HCCD are observed on all the datasets.

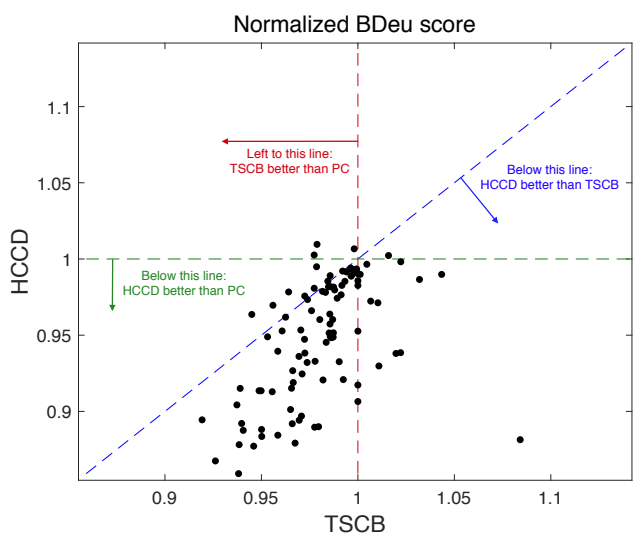


Figure 4: Scatter plot of normalized BDeu score, comparing HCCD, TSCB, and PC, evaluated on the 11 datasets from Table 1, each consists of 10 different training and test sets. The scores are normalized by the PC BDeu score, and so lower is better. Points below the green dashed line correspond to better results of HCCD compared to the PC, which are 97% of the cases. Points to the left of the red dashed line correspond to better results of TSCB compared to PC, which are 82% of the cases. Points below the blue dashed line correspond to better results of HCCD compared to TSCB, which are 90% of the cases.

Table 1: BDeu Scores (higher is better) of PC, TSCB, and HCCD on various datasets.

DATA SET	PC	TSCB	HCCD
ALARM	-60290 ± 2750	-57920 ± 1280	**-55852** ± 1703
CHILD	-67309 ± 1059	-66554 ± 765	**-64539** ± 290
INSURANCE	-74690 ± 1543	-73848 ± 1315	**-73469** ± 1038
MILDEW	-293679 ± 11072	-290456 ± 14157	**-267266** ± 3858
HAILFINDER	-301499 ± 2309	-292020 ± 3365	**-290200** ± 3503
BARLEY	-358461 ± 3592	-352608 ± 6365	**-350807** ± 3419
MUNIN	-451571 ± 2686	-434700 ± 4394	**-401007** ± 3543
WIN95PTS	-64439 ± 768	-62990 ± 947	**-60807** ± 876
PATHFINDER	-274163 ± 2334	-262620 ± 5004	**-248600** ± 2822
HEPAR2	-168822 ± 582	-168528 ± 549	**-167489** ± 533
NEUROPAIN	-185340 ± 678	-182572 ± 1265	**-181670** ± 538

We also calculate causal accuracy [Claassen and Heskes, 2012] as an evaluation metric to causal discovery. Figure 5 shows a scatter plot of causal accuracy, comparing HCCD, TSCB, and PC, evaluated on the 11 datasets, each consists of 10 different training sets (total of 100 points). The causal accuracies are normalized by the PC causal accuracy, and so higher is better. In 91% of the cases, HCCD is better than PC. In 51% of the cases, TSCB is better than PC. Lastly, in 93% of the cases, HCCD is better than TSCB. As evident from the figure, HCCD is superior to the other methods. Additionally, for each complete dataset, the mean ± std causal accuracy is presented in Table 2, and better results for the HCCD are observed on all the datasets, demonstrating improved ability to recover the ground-truth causal graph.

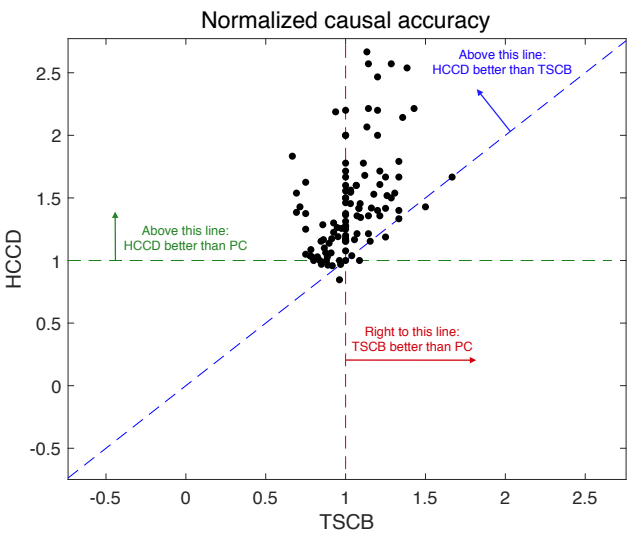


Figure 5: Scatter plot of normalized causal accuracy, comparing HCCD, TSCB, and PC, evaluated on the 11 datasets from Table 2, each consists of 10 different training sets. The values are normalized by the PC value, and higher is better. Points above the green dashed line correspond to better results of HCCD compared to the PC, which are 91% of the cases. Points to the right of the red dashed line correspond to better results of TSCB compared to PC, which are 51% of the cases. Lastly, points above the blue dashed line correspond to better results of HCCD compared to TSCB, which are 93% of the cases.

In addition, we measure the structural hamming distance (SHD) between the learned graph and the ground-truth graph. SHD calculates the number of edge insertions, deletions or flips in order to transform one graph to another graph. For each of the 11 dataset, the mean ± std SHD is presented in Table 2. For all the datasets except one, HCCD is better than the other methods.

Table 2: SHD (lower is better) and causal accuracy (higher is better) comparison for various datasets.

	STRUCTURAL HAMMING DISTANCE		
DATA SET	PC	TSCB	HCCD
ALARM	30.50 ± 3.14	39.6 ± 4.55	**28.80** ± 3.55
CHILD	18.50 ± 1.27	19.4 ± 1.35	**18.0** ± 1.25
INSURANCE	42.90 ± 2.62	46.40 ± 4.27	**42.20** ± 2.84
MILDEW	46.80 ± 1.62	46.80 ± 2.04	**45.70** ± 0.95
HAILFINDER	80.30 ± 2.63	86.20 ± 2.49	**80.10** ± 2.03
BARLEY	83.90 ± 0.74	83.90 ± 0.99	**81.60** ± 1.26
MUNIN	283 ± 1.06	286.10 ± 1.37	**279.60** ± 2.60
WIN95PTS	99.30 ± 4.30	103.20 ± 7.28	**97.80** ± 7.06
PATHFINDER	**193.10** ± 0.94	199.30 ± 2.21	195.20 ± 2.56
HEPAR2	115.70 ± 2.21	117.80 ± 0.92	**114.20** ± 2.53
NEUROPAIN	796.70 ± 13.71	804 ± 23.04	**791** ± 8.24

	CAUSAL ACCURACY		
DATA SET	PC	TSCB	HCCD
ALARM	0.700 ± 0.039	0.608 ± 0.044	**0.727** ± 0.036
CHILD	0.440 ± 0.067	0.448 ± 0.064	**0.597** ± 0.044
INSURANCE	0.476 ± 0.038	0.446 ± 0.044	**0.485** ± 0.038
MILDEW	0.143 ± 0.029	0.126 ± 0.022	**0.235** ± 0.019
HAILFINDER	0.056 ± 0.012	0.060 ± 0.010	**0.082** ± 0.010
BARLEY	0.180 ± 0.027	0.203 ± 0.022	**0.233** ± 0.017
MUNIN	0.051 ± 0.002	0.061 ± 0.005	**0.121** ± 0.011
WIN95PTS	0.320 ± 0.030	0.327 ± 0.026	**0.441** ± 0.015
PATHFINDER	0.066 ± 0.003	0.064 ± 0.011	**0.088** ± 0.008
HEPAR2	0.132 ± 0.024	0.139 ± 0.019	**0.181** ± 0.017
NEUROPAIN	0.037 ± 0.005	0.042 ± 0.003	**0.057** ± 0.004

5 CONCLUSIONS

We propose the HCCD wrapper for causal discovery algorithms (baseline algorithms). HCCD preserves soundness and completeness of the baseline algorithm, while reducing the number of statistical tests, increasing the accuracy of the resulting graph, and reducing the run-time. For constraint-based baseline algorithms, it is assumed that each pair of variables, not adjacent in the true underlying graph, is more strongly correlated to at least one of its separating sets than to other variables not in any of their separating set. Therefore, this property of relative strength of correlation is used by our method to hierarchically partition the domain variables, minimizing the number of independence relations that are not detectable from the cluster variables alone.

Using synthetically generated graphs and data, and selectively limiting certain aspect of HCCD, we demonstrated that recursion and completeness-requirement greatly improve efficiency of the learning procedure and accuracy of the resulting causal graph. Applying our method to real-world graphs, and common publicly available datasets, we demonstrated that HCCD learns significantly more accurate graphs, compared to the PC baseline algorithm.

Finally, we conjecture that scoring-based algorithms may benefit from the HCCD wrapper as well by defining corresponding "similarty" measures. Thus, the search strategy is applied to smaller search spaces, independently and in parallel. We suspect that this will lead to avoiding local maximum and finding higher maximum points.

References

Constantin F Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D Koutsoukos. Local causal and markov blanket induction for causal discovery and feature selection for classification part ii: analysis and extensions. *Journal of Machine Learning Research*, 11(1), 2010.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

Ruichu Cai, Zhenjie Zhang, and Zhifeng Hao. Sada: A general framework to support robust causation discovery with theoretical guarantee. *arXiv preprint arXiv:1707.01283*, 2017.

Jie Cheng, Russell Greiner, Jonathan Kelly, David Bell, and Weiru Liu. Learning bayesian networks from data: An information-theory based approach. *Artificial Intelligence*, 137:43–90, 2002.

David M. Chickering, Dan Geiger, and David Heckerman. Learning Bayesian networks: Search methods and experimental results. In *proceedings of fifth conference on artificial intelligence and statistics*, pages 112–128, 1995.

David Maxwell Chickering and Christopher Meek. Selective greedy equivalence search: Finding optimal bayesian networks using a polynomial number of score evaluations. *arXiv preprint arXiv:1506.02113*, 2015.

David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-sample learning of bayesian networks is np-hard. *Journal of Machine Learning Research*, 5 (Oct):1287–1330, 2004.

Tom Claassen and Tom Heskes. A bayesian approach to constraint based causal inference. *arXiv preprint arXiv:1210.4866*, 2012.

Tom Claassen, Joris M. Mooij, and Tom Heskes. Learning sparse causal models is not NP-hard. In *Uncertainty in Artificial Intelligence*, page 172. Citeseer, 2013.

Diego Colombo, Marloes H. Maathuis, Markus Kalisch, and Thomas S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, pages 294–321, 2012.

Clark Glymour, Kun Zhang, and Peter Spirtes. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10, 2019.

Patrik O Hoyer, Dominik Janzing, Joris M Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in neural information processing systems*, pages 689–696, 2009.

Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594): 824–827, 2002.

Sepideh Pashami, Anders Holst, Slawomir Nowaczyk, and Juhee Bae. Causal discovery using clusters from observational data. In *FAIM'18 Workshop on CausalML*, 2018.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge university press, second edition, 2009.

Judea Pearl and Thomas Verma. A theory of inferred causation. In *International Conference on Principles of Knowledge Representation and Reasoning*, pages 441–452, 1991.

Jonas Peters and Peter Bühlmann. Structural intervention distance for evaluating causal graphs. *Neural computation*, 27(3):771–799, 2015.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. MIT press, 2017.

Raanan Y. Rohekar, Yaniv Gurwicz, Shami Nisimov, Guy Koren, and Gal Novik. Bayesian structure learning by recursive bootstrap. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Marco Scutari. Learning bayesian networks with the bnlearn R package. *Journal of Statistical Software*, 35:1–22, 2010.

Xinpeng Shen, Sisi Ma, Prashanthi Vemuri, and Gyorgy Simon. Challenges and opportunities with causal discovery algorithms: Application to alzheimer's pathophysiology. *Scientific reports*, 10(1):1–12, 2020.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(Oct): 2003–2030, 2006.

Arjun Sondhi and Ali Shojaie. The reduced pc-algorithm: Improved causal structure learning in large random networks. *Journal of Machine Learning Research*, 20(164): 1–31, 2019.

Peter Spirtes, Glymour Clark, and Richard Scheines. *Causation, Prediction and Search*. MIT Press, 2nd edition, 2000.

Ioannis Tsamardinos, Laura E. Brown, and Constantin F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine learning*, 65(1): 31–78, 2006.

Ruibo Tu, Kun Zhang, Bo Bertilson, Hedvig Kjellstrom, and Cheng Zhang. Neuropathic pain diagnosis simulator for causal discovery algorithm evaluation. In *Advances in Neural Information Processing Systems (NeurIPS*, pages 12793–12804, 2019.

Zhenyu Wu and Richard Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 15(11):1101–1113, 1993.

Xianchao Xie and Zhi Geng. A recursive method for structural learning of directed acyclic graphs. *Journal of Machine Learning Research*, 9:459–483, 2008.

Liu Yang, Ye Qing, Wang Liwei, and Peng Jian. Learning structural motif representations for efficient protein structure search. *Bioinformatics*, 34:773–780, 2018.

Raanan Yehezkel and Boaz Lerner. Bayesian network structure learning by recursive autonomy identification. *Journal of Machine Learning Research (JMLR)*, 10:1527–1570, 2009.

Jiji Zhang. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence*, 172(16-17): 1873–1896, 2008.

Yikun Zhang, Yang Liu, and Jiming Liu. Learning bayesian network structure by self-generating prior information: The two-step clustering-based strategy. In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

A PROOFS

In the paper, we relate specifically to constraint-based causal discovery algorithms, relaying on the faithfulness and causal Markov assumption. Causal sufficiency assumption is not required. Let $\mathcal{G}$ be a causal DAG over a set of variables $\boldsymbol{V} = \boldsymbol{X} \cup \boldsymbol{H} \cup \boldsymbol{S}$, where $\boldsymbol{X}$, $\boldsymbol{H}$, and $\boldsymbol{S}$ are the observed, latent, and selection variable sets, respectively.

Let Alg be a causal discovery algorithm, and $\mathrm{ClustCD}$ be the following procedure:

1. Given observed data for domain variables $\boldsymbol{X} = \{A, B, \ldots\}$, partition $\boldsymbol{X}$ into k disjoint subsets $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k$.
2. Call Alg for each $\boldsymbol{X}_i$ (intra-cluster).
3. Call Alg for edges between any pair (A, B) such that $A \in \boldsymbol{X}_i$ and $B \in \boldsymbol{X}_j$, for all $i, j \in \{1, \ldots, k\}, i \neq j$ (inter-cluster).

Theorem 3. *If* Alg *is a sound and complete causal discovery algorithm, then for any partition of* $\boldsymbol{X}$ *procedure* $\mathrm{ClustCD}$ *is sound for* $\boldsymbol{X}$*, but not complete.*

That is, given some partition of the variable set, $\boldsymbol{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k\}$, a pair $A, B \in \boldsymbol{X}_i$, unconnected in the equivalence class of $\mathcal{G}$ (a CPDAG under causal efficiency, otherwise a PAG), might not be tested for independence in steps 2 and 3 of $\mathrm{ClustCD}$, conditioned on any of their separating sets. Once a skeleton is constructed and separating sets are identified, additional independence tests are not required for orienting the edges (e.g., orientation rules [Zhang, 2008], LiNGam [Shimizu et al., 2006], non-linear Gaussian models [Hoyer et al., 2009]). Nevertheless, they rely on the completeness of skeleton learning. Thus, if not all detectable independence relations are identified, the orientation of the edges will not be complete as well.

Proof. By contradiction, $\mathrm{ClustCD}$ is complete. Then, it follows that for every partition of $\boldsymbol{X}$, every pair of unconnected nodes in the equivalence class of $\mathcal{G}$ will be tested for independence in steps 2 and/or 3, conditioned on at least one of their separating sets. A contrary example can be easily constructed. For example, $\mathcal{G}$ is $A \to D \leftarrow C \to E \leftarrow B$ and a clustering $\boldsymbol{X}_1 = \{D, E\}, \boldsymbol{X}_2 = \{A, B, C\}$. Since C is the only separating set for D, E and $C \notin \boldsymbol{X}_1$ it will not be included in independence tests for $\boldsymbol{X}_1$ in step 2 of $\mathrm{ClustCD}$. In step 3, since only nodes not in the same cluster will be tested, and D, E are in the same cluster, they will not be tested for independence. Thus independence of D, E conditioned on C will not be tested. □

Note that Theorem 3 refers to an arbitrary partition of $\boldsymbol{X}$. Nevertheless, one may consider a specific partition such that every independence relation within each cluster is detectable in step 2 of $\mathrm{ClustCD}$. That is, for every pair in a cluster, $\boldsymbol{X}_i$, that are unconnected in the equivalence class of $\mathcal{G}$, at least one separating set exists in the same cluster, $\boldsymbol{X}_i$. However, some graphs do not have such partition (assuming each cluster consists of at least one independence relation needed to be recovered). One example of such graph is $A \to \{B, C, D, E\}, \{B, C, D, E\} \to F$, and $\{B, C, D, E\}$ are unconnected (A is their parent and F is a collider). Thus, Theorem 3 is permissible for these specific partitions as well (assuming each cluster contains at least one independence relation to be recovered).

Theorem 4. *Let* Alg *be a causal discovery algorithm that takes as input an initial graph and a connectivity to be retained. If* Alg *is sound and complete, then Algorithm 1 is sound and complete.*

Proof. The proof follows immediately from line 18 of the Algorithm. The sound and complete causal discovery algorithm Alg is called, and every pair of nodes connected by an edge in $\mathcal{G}'_{\boldsymbol{X}}$ can be tested for independence conditioned on any subset of $\boldsymbol{X}$.

Nevertheless, we also prove by mathematical induction for better clarity on how completeness is obtained gradually. We prove for recursion depth 1 (base case) and then for recursion depth $d+1$ (inductive step).

Base case: Recursion depth is 1. Input variable set $\boldsymbol{X}$ is partitioned into $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k\}$ clusters. As recursion depth of 1 is assumed, any further recursive calls (line 10) comply with the exit condition (line 1). As a result Alg is called for learning a causal graph for each of the clusters independently and returned. Since Alg is sound and complete, every independence relation between nodes in a cluster $\boldsymbol{X}_i$ that are detectable by conditioning on some subset $\boldsymbol{Z} \subset \boldsymbol{X}_i$ will be detected (due to soundness, no false independencies will be returned). Alg is called again, fixing (retaining) the sub-graph for each cluster, learning only inter cluster edges (line 17). However, the resulting graph $\mathcal{G}'_{\boldsymbol{X}}$ is not complete (Theorem 3). Calling Alg again, this time without fixing the sub-graphs, allows testing every pair, adjacent in $\mathcal{G}'_{\boldsymbol{X}}$ conditioned on any subset of $\boldsymbol{X}$. Thus, if Alg is complete, every independence relation in the equivalence class of $\mathcal{G}$, where at least one separating set is in $\boldsymbol{X}$, will be detected, and Algorithm 1 is complete.

Inductive step: Assume that the algorithm is sound and complete for recursion depth d and prove for recursion depth $d+1$. Input variable $\boldsymbol{X}$ is partitioned into $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k\}$ clusters. A recursive call (line 10) for cluster $\boldsymbol{X}_i$, and subsequent d recursive calls, ensures that all independence relations, with separating sets within $\boldsymbol{X}_i$ will be detected. All independence relations between clusters are identified in line 17. Finally, Alg is called in line 18 for $\boldsymbol{X}$ using an initial graph $\mathcal{G}'_{\boldsymbol{X}}$, allowing every adjacent pair to be tested for independence on any subset of $\boldsymbol{X}$. □

B SYNTHETIC DATA GENERATION AND ABLATION STUDY

In this section describe the process and present the results of Section 4 regarding the synthetic data generation and the ablation study.

B.1 SYNTHETIC DATA GENERATION

We generate random DAGs and sampled data using the following procedure. A DAG, having n variables and connectivity factor ρ is sampled in the following way. First, an adjacency matrix $\boldsymbol{A}$ of a DAG $\mathcal{G}$ is created by independent realization of $\mathrm{Bernoulli}(\rho/(n-1))$ in the upper triangle. Importantly, if the resulting DAG is unconnected, we repeat until a connected DAG is sampled. Then, a weight matrix $\boldsymbol{W}$ for the graph edges is created by sampling from $\mathrm{Uniform}([0.1, 1])$ for each non-zero element in $\boldsymbol{A}$. Finally, graph $\mathcal{G}$ is treated as a statistical model by setting conditional probabilities $p(X_i|\mathrm{Pa}_{\boldsymbol{A}}(X_i)) = \boldsymbol{W}_{(\cdot,i)}\boldsymbol{A}^{\mathrm{T}}_{(\cdot,i)} + \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0,1)$.

B.2 ABLATION STUDY

We evaluate how the density of the underlying graph affects the number of independence tests performed. In addition, we examine the gain achieved by the recursion, and the effect that the completeness requirement has on the accuracy.

Conditional independence is determined if partial correlation, after applying Fisher's z-transform, is equal to zero, with significance level $\alpha = 0.01$. In each experiment 100 DAGs are created. For each DAG, the number of nodes is $n = 100$, connectivity factor is $\rho \in \{3, 4, 5, 6, 7\}$, and the number of data samples $\ell = 1000$. For HCCD, in Algorithm 1-line 9, $k' = k = 2$. For this experiment, we limit to only one recursive call. Thus, 4 clusters are formed before backtracking. We compare this to case where no recursive calls are performed, achieved by setting $k = 4$ and $k' = 1$ in Algorithm 1-line 9, denoted "*HCCD-flat*" (this is equivalent to the TSCB wrapper). We consider a modified version of HCCD where the second call, line 18 in Algorithm 1, is removed from the recursion and called only after the algorithm concludes (this modification still preserves completeness). We denote this modified version, "*HCCD-not-c*". From Table 3, it is evident that the number of required independence tests is significantly reduced by HCCD wrapper for the PC algorithm. Moreover, when comparing to HCCD-flat (same number of clusters without recursion) it is evident that HCCD benefits from recursion. Finally, we can see that removing the completeness requirement for each cluster (HCCD-not-c) has constant improvement over PC but is inferior to HCCD. We attribute this to the many undetectable independence relations that are represented by retained edges ($\mathcal{L}$). This causes PC to consider more condition sets resulting in more independence tests. As in this experiment we limited HCCD to one recursive call and to minimal number of clusters, gain in accuracy, compared to PC or modified versions of HCCD, is statistically insignificant. Nevertheless, this same setting is adequate for evaluating the effect of the completeness requirement on the accuracy. We calculate the ratio of errors, dividing average number of HCCD errors by average number of errors of its non-complete modified version (without line 18, Algorithm 1). For DAG connectivity factors (ρ) $[3, 4, 5, 6, 7]$, extra-edges errors ratios are, $[0.11, 0.08, 0.1, 0.16, 0.21]$, and missing edges ratios are $[1.11, 1.12, 1.12, 1.08]$, respectively. The ratios of errors are reported in Table 4. Although the completeness requirement results in some increase in missing

edges, it also results in a significant decrease in extra edges.

Table 3: Normalized number of independence tests required for recovering DAGs having various densities (parametrized by the connectivity factor ρ).

DENSITY	PC	HCCD-FLAT	HCCD-NOT-C	HCCD
$\rho = 3$	1.00	0.89	0.86	**0.80**
$\rho = 4$	1.00	0.82	0.85	**0.74**
$\rho = 5$	1.00	0.78	0.86	**0.66**
$\rho = 6$	1.00	0.77	0.81	**0.61**
$\rho = 7$	1.00	0.76	0.74	**0.58**

Table 4: The effect of the completeness requirement (Algorithm 1, line 18) on the structural correctness. The numbers of extra and missing edges in the graph resulting from HCCD, are divided by the corresponding values after relaxing the completeness requirement. Values represent ratios, e.g., a value of 0.1 indicates that only 10% of errors in the non-complete algorithm are present in HCCD.

DENSITY	$\rho = 3$	$\rho = 4$	$\rho = 5$	$\rho = 6$	$\rho = 7$
EXTRA EDGES RATIO	0.11	0.08	0.10	0.16	0.21
MISSING EDGES RATIO	1.11	1.12	1.12	1.10	1.08

C AN ANALYSIS USING SYNTHETIC DATA

Section 4.2 evaluates the performance of the HCCD wrapper with respect to the number of training set size, and to the number of nodes in the graphs, compared to the baseline method (PC) over 5 measures, for graphs with $n = 100$ nodes. This evaluation is extended here, in Figure 6–7, to other (low and high) number of nodes in a graph, i.e. $n \in \{200, 1000\}$. Each figure shows mean $\pm$ std of 500 independent tests (DAGs), and values are normalized by the PC score in order to analyze the improvement over the baseline method. In each figure, the evaluated metrics are: (a) Causal accuracy (higher is better); (b) SHD (lower is better); (c) SID (lower is better); (d) Number of CI tests (lower is better). (e) Run-time (lower is better). It is evident that the HCCD performs well for all 3 structural correctness metrics along the entire range of the training set size and for every n, and it is superior to the baseline method. In addition, we can see the saving in the number of CI tests, and importantly the reduction in the run-time (which includes the clustering stage) for the entire range of the training set size. One exception is for the case of $n = 20$ (figure not presented here due to lack of space), for which the HCCD run time is higher, which is expected since the complexity overhead of the clustering stage overtakes the saving in run-time gained by using less statistical tests in datasets with a small number of nodes. Nevertheless, for the common real-world cases of datasets with many domain variables, the HCCD achieves a significant reduction in run-time. Additionally, it is evident that the run-time reduction increases with the increase of the number of training samples, i.e. bigger training sets gain more in real-time reduction.

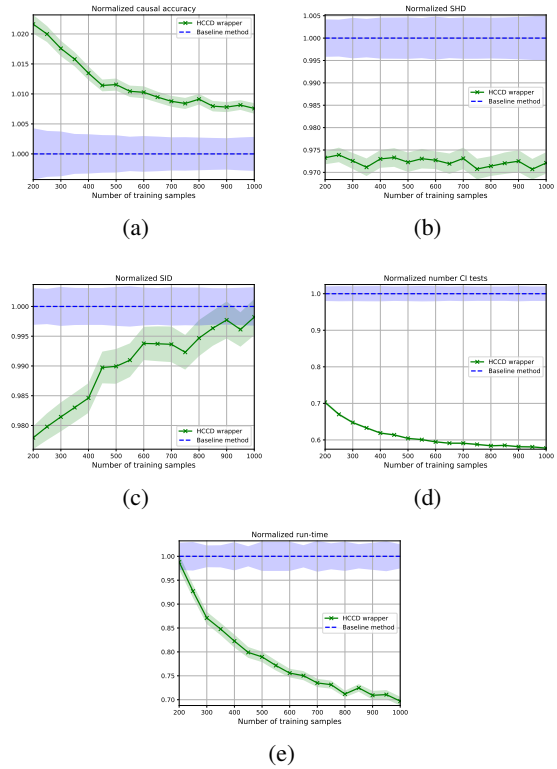


Figure 6: Performance of the HCCD wrapper, relatively to the baseline (PC), as a function of the number of training samples, for 200 graph nodes.

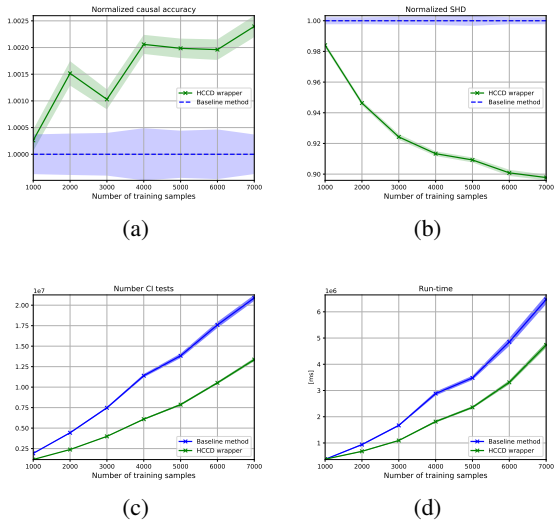


Figure 7: Performance of the HCCD wrapper, relatively to the baseline (PC), as a function of the number of training samples, for 1000 graph nodes. In (c) and (d) we present the absolute values in order to demonstrate the actual savings.