

823 A Proofs

824 **Definition 3.1.** (Low-order d-separation and d-connection FOL formulæ) Given a discrete directed
825 graph $\mathbf{A} \in \{0, 1\}^{d \times d}$, the 0th-order d-separation formula $S_{\mathbf{A}}^{(0)} : [d]^2 \rightarrow \{0, 1\}$ and the 1st-order
826 d-separation formula $S_{\mathbf{A}}^{(1)} : [d]^3 \rightarrow \{0, 1\}$ are defined as:

$$S_{\mathbf{A}}^{(0)}(x, y) := \forall a \in [d], \neg (R_{\mathbf{A}}(a, x) \wedge R_{\mathbf{A}}(a, y)), \quad (2)$$

$$S_{\mathbf{A}}^{(1)}(x, y \mid z) := S_{\mathbf{A}_{-z}}^{(0)}(x, y) \wedge \left(\left(\forall a \in [d] \setminus \{z\}, S_{\mathbf{A}_{-z}}^{(0)}(x, a) \vee \neg R_{\mathbf{A}}(a, z) \right) \right. \\ \left. \vee \left(\forall b \in [d] \setminus \{z\}, S_{\mathbf{A}_{-z}}^{(0)}(y, b) \vee \neg R_{\mathbf{A}}(b, z) \right) \right). \quad (3)$$

827 Equivalently, the 0-th and 1-st order d-connection statements $C_{\mathbf{A}}^{(0)}$ and $C_{\mathbf{A}}^{(1)}$ are:

$$C_{\mathbf{A}}^{(0)}(x, y) := \exists a \in [d], R_{\mathbf{A}}(a, x) \wedge R_{\mathbf{A}}(a, y), \quad (4)$$

$$C_{\mathbf{A}}^{(1)}(x, y \mid z) := C_{\mathbf{A}_{-z}}^{(0)}(x, y) \vee \left(\left(\exists a \in [d] \setminus \{z\}, C_{\mathbf{A}_{-z}}^{(0)}(x, a) \wedge R_{\mathbf{A}}(a, z) \right) \right. \\ \left. \wedge \left(\exists b \in [d] \setminus \{z\}, C_{\mathbf{A}_{-z}}^{(0)}(y, b) \wedge R_{\mathbf{A}}(b, z) \right) \right). \quad (5)$$

828 **Theorem 3.2.** For any DAG \mathbf{A} with d nodes and any three nodes $x, y, z \in [d]$ that are distinct,
829 $x \perp\!\!\!\perp_{\mathbf{A}} y$ if and only if $S_{\mathbf{A}}^{(0)}(x, y) = 1$, and $x \perp\!\!\!\perp_{\mathbf{A}} y \mid z$ if and only if $S_{\mathbf{A}}^{(1)}(x, y \mid z) = 1$. Similarly,
830 $x \not\perp\!\!\!\perp_{\mathbf{A}} y$ if and only if $C_{\mathbf{A}}^{(0)}(x, y) = 1$, and $x \not\perp\!\!\!\perp_{\mathbf{A}} y \mid z$ if and only if $C_{\mathbf{A}}^{(1)}(x, y \mid z) = 1$.³

831 *Proof.* First, we note that the d-separation formulæ $S_{\mathbf{A}}^{(0)}, S_{\mathbf{A}}^{(1)}$ and the d-connection formulæ
832 $C_{\mathbf{A}}^{(0)}, C_{\mathbf{A}}^{(1)}$ are the negation of each other and can be obtained from each other via the De Morgan's
833 law. Thus, the statement “ $x \perp\!\!\!\perp_{\mathbf{A}} y$ if and only if $S_{\mathbf{A}}^{(0)}(x, y) = 1$ ” is equivalent to “ $x \not\perp\!\!\!\perp_{\mathbf{A}} y$ if and
834 only if $C_{\mathbf{A}}^{(0)}(x, y) = 1$ ”, and vice versa, the statement “ $x \perp\!\!\!\perp_{\mathbf{A}} y \mid z$ if and only if $S_{\mathbf{A}}^{(1)}(x, y \mid z) = 1$ ”
835 is equivalent to “ $x \not\perp\!\!\!\perp_{\mathbf{A}} y \mid z$ if and only if $C_{\mathbf{A}}^{(1)}(x, y \mid z) = 1$.” Thus, we proceed to prove the
836 statements involving the d-connection formulæ $C_{\mathbf{A}}^{(0)}, C_{\mathbf{A}}^{(1)}$.

837 **Part 1:** $x \not\perp\!\!\!\perp_{\mathbf{A}} y$ if and only if $C_{\mathbf{A}}^{(0)}(x, y) = 1$.

838 • We show the direction $x \not\perp\!\!\!\perp_{\mathbf{A}} y \implies C_{\mathbf{A}}^{(0)}(x, y) = 1$.

839 If $x \not\perp\!\!\!\perp_{\mathbf{A}} y$, i.e., x is d-connected to y with an empty conditioning set, then there exists a path
840 between x and y that does not contain any colliders, because otherwise the path would be blocked.
841 Denote the sequence of nodes in this path $\mathbf{p} = (p_1, \dots, p_m)$ with $p_1 = x$ and $p_m = y$. In other
842 words, for any three consecutive nodes $a, b, c \in \mathbf{p}$, we have either the chain structure $a \rightarrow b \rightarrow c$
843 or $a \leftarrow b \leftarrow c$, or the fork structure $a \leftarrow b \rightarrow c$.

844 Because there is no collider structure in \mathbf{p} , we will show that there must be *at most one* fork
845 structure in \mathbf{p} . To see this by contradiction, consider if \mathbf{p} contains more than one fork, and let
846 $p_{l-2} \leftarrow p_{l-1} \rightarrow p_l$ and $p_r \leftarrow p_{r+1} \rightarrow p_{r+2}$ be two forks ($l < r$) in \mathbf{p} such that there is no other
847 fork between p_l and p_r . Then, either there are some colliders between p_l and p_r , or p_l and p_r
848 are connected by some directed paths, i.e., either $p_l \rightsquigarrow p_r$ or $p_r \rightsquigarrow p_l$. If $p_l \rightsquigarrow p_r$, then p_r is a collider,
849 and vice versa if $p_r \rightsquigarrow p_l$, then p_l is a collider. Hence, in all scenarios, \mathbf{p} will have at least one
850 collider, contradicting the fact that \mathbf{p} should not have any collider.

851 Now, we discuss case by case based on whether \mathbf{p} has no fork, or it has one fork. Figure 4a
852 illustrates the structure such a path \mathbf{p} .

853 **Case 1: Suppose there is no fork on \mathbf{p} :** If there is no fork at all, then \mathbf{p} is a directed path. Without
854 loss of generality, assume $\mathbf{p} = x \rightarrow \dots \rightarrow y$. Then, let $a = x$, we have $a \rightsquigarrow x$ (a node is always

³The proofs of Theorem 3.2 and all following theorems and lemmas can be found in Appendix A.

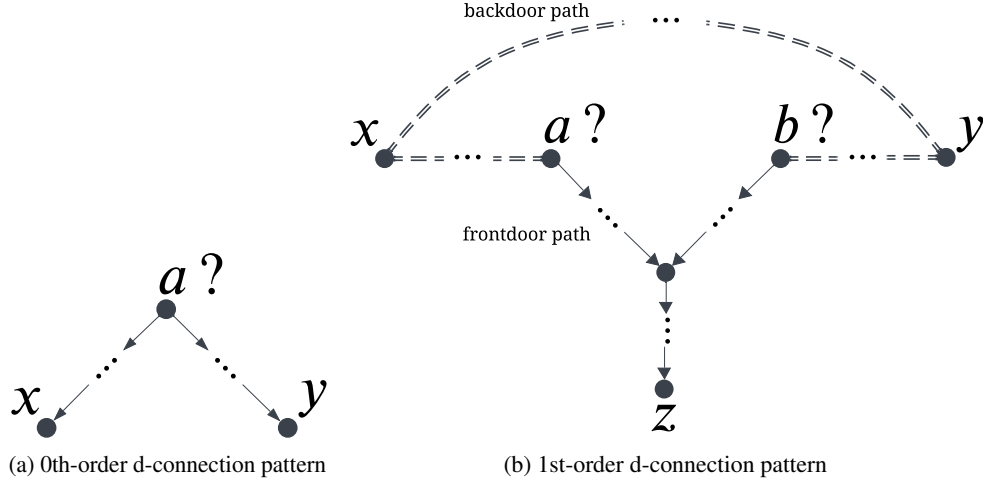


Figure 4: Illustration of the typical graph connectivity patterns checked by the low-order d-separation/d-connection FOL formulæ (Definition 3.1) to determine whether the given query $((x, y)$ or $(x, y \mid z))$ is d-separated or d-connected. The double-dashed line $== \cdots ==$ refers to a 0th-order d-connecting path, and the arrowed line $\rightarrow \cdots \rightarrow$ refers to a directed path. In either pattern, the variable nodes a and b could be the query node x or y themselves.

naively reachable from itself) and $a \rightsquigarrow y$. Hence, $(R_{\mathbf{A}}(a, x) \wedge R_{\mathbf{A}}(a, y)) = 1$, and therefore the formulæ $C_{\mathbf{A}}^{(0)}(x, y) = \exists a \in [d], R_{\mathbf{A}}(a, x) \wedge R_{\mathbf{A}}(a, y)$ is 1.

Case 2: Suppose there is one fork on p When there is one and only one fork, it means the center node of the fork is the common ancestor of both x and y . In other words, let a be this center node, then we have $a \rightsquigarrow x$ and $a \rightsquigarrow y$. Thus, similarly, we have $(R_{\mathbf{A}}(a, x) \wedge R_{\mathbf{A}}(a, y)) = 1$, and therefore the formulæ $C_{\mathbf{A}}^{(0)}(x, y) = \exists a \in [d], R_{\mathbf{A}}(a, x) \wedge R_{\mathbf{A}}(a, y)$ is 1.

Thus, we have shown that $x \not\perp_{\mathbf{A}} y \implies C_{\mathbf{A}}^{(0)}(x, y) = 1$

• Now we show the other direction $C_{\mathbf{A}}^{(0)}(x, y) = 1 \implies x \not\perp_{\mathbf{A}} y$.

Given $C_{\mathbf{A}}^{(0)}(x, y) = 1$, we have $\exists a \in [d]$ such that $a \rightsquigarrow x$ and $a \rightsquigarrow y$. Let a be such a node. Thus, a is a common ancestor of x and y . Let $\mathbf{p}_x = a \rightarrow \cdots \rightarrow x$ be the directed path that starts at a and ends at x , and similarly let $\mathbf{p}_y = a \rightarrow \cdots \rightarrow y$. Then, the path form by the sequence of nodes in \mathbf{p}_x and \mathbf{p}_y , i.e., $\mathbf{p} = (x, \dots, a, \dots, y)$, is clearly a path consisting of at most one fork structure with other wise chain structures. When $a = x$ or $a = y$, \mathbf{p} is a directed path without fork, and otherwise has one fork. In either case, \mathbf{p} is a d-connecting path between x and y , making $x \not\perp_{\mathbf{A}} y$.

Part 2: $x \not\perp_{\mathbf{A}} y \mid z$ if and only if $C_{\mathbf{A}}^{(1)}(x, y \mid z) = 1$.

• We show the direction $x \not\perp_{\mathbf{A}} y \mid z \implies C_{\mathbf{A}}^{(1)}(x, y \mid z) = 1$.

We are given $x \not\perp_{\mathbf{A}} y \mid z$, i.e., x and y are d-connecting when conditioning on z . Thus, there must exist a non-empty set of d-connecting paths \mathcal{P} between x and y that is not blocked given z . We proceed with a case-by-case discussion depending on whether all of these paths contain colliders.

1. First, we consider the case where there is some path in \mathcal{P} that does not contain any collider. Let $\mathbf{p} \in \mathcal{P}$ be such a path. Then, z must *not* be on \mathbf{p} , because otherwise, since z is not a collider and it is being conditioned, \mathbf{p} would be blocked and therefore $\mathbf{p} \notin \mathcal{P}$.

Since \mathbf{p} does not include z , it remains unchanged in the subgraph \mathbf{A}_{-z} , in which node z is removed from the graph \mathbf{A} , and thus it is still a d-connecting path in \mathbf{A}_{-z} . Hence, because of \mathbf{p} , we have x and y are d-connecting in the subgraph \mathbf{A}_{-z} . In other words, we have $C_{\mathbf{A}_{-z}}^{(0)}(x, y) =$

880 1. Since $C_{\mathbf{A}-z}^{(0)}(x, y)$ is the one of two terms from the outmost disjunction operator \vee in the
 881 formula for $C_{\mathbf{A}}^{(1)}(x, y \mid z)$, we have $C_{\mathbf{A}-z}^{(0)}(x, y) = 1$ renders $C_{\mathbf{A}}^{(1)}(x, y \mid z) = 1$.

882 In Figure 4b, this pattern is illustrated via the “backdoor path” structure.

883 2. Otherwise, all paths in \mathcal{P} contain colliders. We first show that this implies that we can find a
 884 path \mathbf{p}^* in \mathcal{P} that has only one collider.

885 To see this, pick any path $\mathbf{p} \in \mathcal{P}$. If \mathbf{p} indeed has only one collider, then we are done. Hence,
 886 we consider the non-trivial case where \mathbf{p} has more than one collider. We further divide the cases
 887 depending on whether node z is included in path \mathbf{p} .

888 (a) If z is included in \mathbf{p} , then z must itself be a collider, because otherwise \mathbf{p} would be blocked
 889 by z .

890 Now, let z' be another collider in \mathbf{p} . Then, z must be a descendant of z' , because otherwise
 891 \mathbf{p} would be blocked by z' even though z is in the conditioning set. In other words, z is
 892 reachable from z' with a directed path, i.e., $z' \rightsquigarrow z$.

893 Using this knowledge, we can construct the following alternative path $\mathbf{p}_{z'}$ that still
 894 has z as a collider but converts z' to a non-collider. Specifically, without loss of
 895 generality, we assume the path \mathbf{p} consists of the following sequence of nodes: $\mathbf{p} =$
 896 $(x, \dots, p_{l'}, z', p_{r'}, \dots, p_l, z, p_r, \dots, y)$ for some indices l, r, l', r' with collider structures
 897 $p_{l'} \rightarrow z' \leftarrow p_{r'}$ and $p_l \rightarrow z \leftarrow p_r$. We have shown that $z' \rightsquigarrow z$, and denote the cor-
 898 responding directed path $z' \rightarrow q_1 \rightarrow \dots \rightarrow q_m \rightarrow z$. We can obtain the new path as
 899 follows:

$$\mathbf{p}_{z'} = (x, \dots, p_{l'}, z', q_1, \dots, q_m, z, p_r, \dots, y),$$

900 where we replace the middle section sub-path $(z', p_{r'}, \dots, p_l, z)$ in \mathbf{p} to (z', q_1, \dots, q_m, z)
 901 to form $\mathbf{p}_{z'}$.

902 Now, note that in $\mathbf{p}_{z'}$, z' is no longer a collider but forms a chain structure, because
 903 $p_{l'} \rightarrow z' \rightarrow q_1$. Nevertheless, z is still a collider, because $q_m \rightarrow z \leftarrow p_r$. Furthermore, $\mathbf{p}_{z'}$
 904 is still a d-connecting path. This is because the sub-paths $(x, \dots, p_{l'}, z')$ and (z, p_r, \dots, y)
 905 are d-connecting conditioning on z because they are parts of the original \mathbf{p} which is a
 906 d-connecting path, and the “new” sub-path (z', q_1, \dots, q_m, z) by definition is d-connecting
 907 conditioning on z because it is a directed path. Hence, taken together, $\mathbf{p}_{z'}$ is d-connecting
 908 given z , so $\mathbf{p}_{z'} \in \mathcal{P}$.

909 Thus, by iteratively constructing these alternative paths $\mathbf{p}_{z'}$ for any colliders z' in \mathbf{p} that is
 910 not z , we can arrive at a path $\mathbf{p}^* \in \mathcal{P}$ that has z as the only collider.

911 (b) Now, suppose z is not included in \mathbf{p} and \mathbf{p} has more than one collider. Let z' and z'' be
 912 any two such colliders, and without loss of generality, let the path consists of the sequence
 913 of nodes $\mathbf{p} = (x, \dots, p_{l'}, z', p_{r'}, \dots, p_{l''}, z'', p_{r''}, \dots, y)$. Similarly, both z' and z'' must
 914 be ancestors of z , because otherwise one of them would block the path. Hence, we know
 915 that $z' \rightsquigarrow z$ and $z'' \rightsquigarrow z$. Let the directed paths be $z' \rightarrow q_1 \rightarrow \dots \rightarrow q_m \rightarrow z$ and
 916 $z'' \rightarrow o_1 \rightarrow \dots \rightarrow o_s \rightarrow z$ respectively. We can construct the following new path:

$$\mathbf{p}_{z', z''} = (x, \dots, p_{l'}, z', q_1, \dots, q_m, z, o_s, \dots, o_1, z'', p_{r''}, \dots, y).$$

917 Here, note that both z' and z'' are no longer a collider, because $p_{l'} \rightarrow z' \rightarrow q_1$ and
 918 $o_1 \leftarrow z'' \leftarrow p_{r''}$. Nevertheless, z is a collider since $q_m \rightarrow z \leftarrow o_s$. Furthermore, $\mathbf{p}_{z', z''}$ is
 919 still a d-connecting path conditioned on z . As we have established in the discussion of
 920 the previous case, that the sub-paths $(x, \dots, p_{l'}, z')$ and $(z'', p_{r''}, \dots, y)$ are d-connecting.
 921 Since (z', q_1, \dots, q_m, z) and $(z, o_s, \dots, o_1, z'')$ are directed paths, we have that the sub-
 922 paths (x, \dots, q_m, z) and (z, o_s, \dots, y) are also d-connecting. Taken together, the whole
 923 path is d-connecting given z , i.e., $\mathbf{p}_{z', z''} \in \mathcal{P}$.

924 In other words, we have converted both colliders z' and z'' in the original \mathbf{p} into non-
 925 colliders while introducing z as a new collider. This means that the resulting path $\mathbf{p}_{z', z''}$
 926 satisfies the initial condition of a path \mathbf{p} discussed in the previous case. Hence, by further
 927 following the procedure described in the previous case, we can eventually arrive at a path
 928 $\mathbf{p}^* \in \mathcal{P}$ that has z as the only collider.

929 Now that we know \mathcal{P} will necessarily include a path \mathbf{p}^* with only one collider, denote this
 930 collider z^* , and we know either $z^* = z$ or z^* is an ancestor of z . In other words, we have
 931 $z^* \rightsquigarrow z$. Hence, we can denote \mathbf{p}^* as a sequence of nodes: $\mathbf{p}^* = (x, \dots, p_l^*, z^*, p_r^*, \dots, y)$ for
 932 some indices l, r , where $p_l^* \rightarrow z^* \leftarrow p_r^*$.

933 Now, let $a = x$ if there is no fork structure in the sub-path (x, \dots, z^*) of \mathbf{p}^* , otherwise let a
 934 be the *rightmost* fork in (x, \dots, z^*) (i.e. there is no other fork in the sub-path (a, \dots, z^*) of

\mathbf{p}^*). Then, we have $a \rightsquigarrow z^*$ because there is neither any fork nor any collider in the sub-path (a, \dots, z^*) , and we know it has a constituent edge $p_l^* \rightarrow z^*$ with an edge direction pointing towards z^* . Moreover, because $z^* \rightsquigarrow z$, we have $a \rightsquigarrow z$, and equivalently $R_{\mathbf{A}}(a, z)$.

On the other hand, the sub-path (x, \dots, a) is a d-connecting path conditioning on z because its super-path $\mathbf{p}^* \in \mathcal{P}$ is a d-connecting path conditioning on z , does not have any collider because z^* is the only collider in \mathbf{p}^* , and does not include z . Hence, in the subgraph \mathbf{A}_{-z} where z is removed from the graph \mathbf{A} , the sub-path (x, \dots, a) remains unchanged and is still a d-connecting path, and in this case without conditioning on z . Therefore, x and a is d-connecting in \mathbf{A}_{-z} , i.e., $x \not\perp_{\mathbf{A}_{-z}} a$. Hence, we have found a node a such that $(x \not\perp_{\mathbf{A}_{-z}} a) \wedge (a \rightsquigarrow z)$, or equivalently, we have the following:

$$\left(\exists a \in [d] \setminus \{z\}, (x \not\perp_{\mathbf{A}_{-z}} a) \wedge R_{\mathbf{A}}(a, z) \right) = 1.$$

Similarly, let $b = y$ if there is no fork structure in the sub-path (z, \dots, y) of \mathbf{p}^* , otherwise let b^* be the *leftmost* fork in (z, \dots, y) . We then have also have the following:

$$\left(\exists b \in [d] \setminus \{z\}, (y \not\perp_{\mathbf{A}_{-z}} b) \wedge R_{\mathbf{A}}(b, z) \right) = 1.$$

Combining everything, we fulfill second term in the outermost disjunction operator in the FOL formula and thus have $C_{\mathbf{A}}^{(1)}(x, y | z) = 1$.

In Figure 4b, this pattern is illustrated as the “frontdoor path” structure.

- Now we show the other direction $C_{\mathbf{A}}^{(1)}(x, y | z) = 1 \implies x \not\perp_{\mathbf{A}} y | z$.

If $C_{\mathbf{A}}^{(1)}(x, y | z) = 1$, then we have

$$C_{\mathbf{A}_{-z}}^{(0)}(x, y) = 1,$$

or we have

$$\left(\left(\exists a \in [d] \setminus \{z\}, C_{\mathbf{A}_{-z}}^{(0)}(x, a) \wedge R_{\mathbf{A}}(a, z) \right) \wedge \left(\exists b \in [d] \setminus \{z\}, C_{\mathbf{A}_{-z}}^{(0)}(y, b) \wedge R_{\mathbf{A}}(b, z) \right) \right) = 1.$$

We discuss it case by case.

1. If it is the first case, $C_{\mathbf{A}_{-z}}^{(0)}(x, y) = 1$, then there exists a d-connecting path \mathbf{p} between x and y in the subgraph \mathbf{A}_{-z} . \mathbf{p} then does not include z , and must not include any colliders. Hence, back in the original graph \mathbf{A} , \mathbf{p} is a “backdoor” path that does not involve z that renders x and y d-connecting. Thus, we have $x \not\perp_{\mathbf{A}} y$ and also $x \not\perp_{\mathbf{A}} y | z$.
2. We now consider the other case. Since the left-hand-side term is a conjunction (\wedge), both term in the conjunction must be true, meaning there exists some nodes a and b satisfying either term respectively. Taking a step further, this means that all of the four terms – $C_{\mathbf{A}_{-z}}^{(0)}(x, a)$, $R_{\mathbf{A}}(a, z)$, $C_{\mathbf{A}_{-z}}^{(0)}(y, b)$, and $R_{\mathbf{A}}(b, z)$ – are true.

Using a similar reasoning as mentioned in the previous case, $C_{\mathbf{A}_{-z}}^{(0)}(x, a)$ implies that we have a d-connecting path between x and a and that $x \not\perp_{\mathbf{A}} a | z$. Similarly, $C_{\mathbf{A}_{-z}}^{(0)}(y, b)$ implies that $y \not\perp_{\mathbf{A}} b | z$. Furthermore, because $R_{\mathbf{A}}(a, z) = 1$ and $R_{\mathbf{A}}(b, z) = 1$, that is, $a \rightsquigarrow z$ and $b \rightsquigarrow z$, the path between a and b , formed by the corresponding directed paths that renders $a \rightsquigarrow z$ and $b \rightsquigarrow z$ respectively, is a d-connecting path conditioning on z , as z serves as the only collider. Hence, we have $a \not\perp_{\mathbf{A}} b | z$.

Hence, by transitivity of d-connection statements, we have that $x \not\perp_{\mathbf{A}} y | z$, finishing the entire proof.

□

Lemma 3.4. For any discrete graph $\mathbf{A} \in \{0, 1\}^{d \times d}$ with maximum directed path length l and for all pair of nodes $x, y \in [d]$, $x \rightsquigarrow_{\mathbf{A}} y$ if and only if $R_{\mathbf{A}}^{(l)}(x, y) = 1$.

Proof. We prove this lemma by induction on the maximum directed path length l .

974 **Basecase:** Suppose the maximum directed path length in \mathcal{A} is 0. This means there is no path and \mathcal{A}
 975 is an empty graph. Thus, for all $x \in [d]$, $x \rightsquigarrow_{\mathcal{A}} x$ and for all $x, y \in [d]$ with $x \neq y$, $x \not\rightsquigarrow_{\mathcal{A}} y$.

976 **Induction:** Suppose the statement in the lemma holds true with a maximum directed path length of
 977 $l - 1$. We now prove the statement for l .

978 • We prove the direction $x \rightsquigarrow_{\mathcal{A}} y \implies R_{\mathcal{A}}^{(l)}(x, y) = 1$.

979 Given that $x \rightsquigarrow_{\mathcal{A}} y$, let \mathbf{p} be such a directed path starting from x and ending at y . Let u be the
 980 second to last node in \mathbf{p} with an edge $u \rightarrow y$, or equivalently, $\mathbf{A}_{u,y} = 1$. Then, we also have
 981 $x \rightsquigarrow_{\mathcal{A}} u$.

982 Since the path from x to u will have a length smaller than or equal to $l - 1$, using the induction
 983 hypothesis, we have $R_{\mathcal{A}}^{(l-1)}(x, u) = 1$.

984 Thus, we have $R_{\mathcal{A}}^{(l-1)}(x, u) \wedge \mathbf{A}_{u,y} = 1$, and consequently $\bigvee_{u \in d} R_{\mathcal{A}}^{(l-1)}(x, u) \wedge \mathbf{A}_{u,y} = 1$,
 985 satisfying the first term in the outmost disjunction in the recursive formula for $R_{\mathcal{A}}^{(l)}$, rendering
 986 $R_{\mathcal{A}}^{(l)}(x, y) = 1$.

987 • Now we prove the other direction $R_{\mathcal{A}}^{(l)}(x, y) = 1 \implies x \rightsquigarrow_{\mathcal{A}} y$.

988 Given $R_{\mathcal{A}}^{(l)}(x, y) = 1$, there are possible scenarios: that $\bigvee_{u \in d} R_{\mathcal{A}}^{(l-1)}(x, u) \wedge \mathbf{A}_{u,y} = 1$ or that
 989 $R_{\mathcal{A}}^{(l-1)}(x, y) = 1$. we discuss case by case.

- 990 1. Suppose $\bigvee_{u \in d} R_{\mathcal{A}}^{(l-1)}(x, u) \wedge \mathbf{A}_{u,y} = 1$, then there exists at least one node u such that
 991 $R_{\mathcal{A}}^{(l-1)}(x, u) \wedge \mathbf{A}_{u,y} = 1$, meaning both $R_{\mathcal{A}}^{(l-1)}(x, u) = 1$ and $\mathbf{A}_{u,y} = 1$. This means that, by
 992 the induction hypothesis, $x \rightsquigarrow_{\mathcal{A}} u$ with maximum path length $l - 1$. Furthermore, we have
 993 $u \rightarrow y$ is a directed edge. Thus, there exists a directed path from x to y , i.e. $x \rightsquigarrow_{\mathcal{A}} y$ with
 994 maximum path length l .
- 995 2. Suppose $R_{\mathcal{A}}^{(l-1)}(x, y) = 1$, then by the inductive hypothesis, $x \rightsquigarrow_{\mathcal{A}} y$ with a maximum path
 996 length $l - 1$, which also makes the statement true under the setting of a maximum path length
 997 of l .

998 Thus, in either case, we have $x \rightsquigarrow_{\mathcal{A}} y$.

999 □

1000 The proof of Lemma 3.7 and subsequent Theorem 3.8 requires the following lemma showing the
 1001 lower-bound nature of LogLTN's [1] t-norm and t-conorm logical operators. We state this lemma and
 1002 its proof as follows.

1003 **Lemma A.1.** *Given the t-norm and t-conorm operators of LogLTN [1], which we restate as follows,*

$$\tilde{T}_m(\{x'_i\}_{i \in [m]}) := \sum_{i=1}^m x'_i, \tilde{O}_m(\{x'_i\}_{i \in [m]}) := \alpha \left(C + \log \left(\frac{\sum_{i=1}^n e^{x'_i/\alpha - C}}{m} \right) \right).$$

1004 Let $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m$ be m Bernoulli random variables that are mutually independent or positively
 1005 correlated. Denote their probabilities as p_1, p_2, \dots, p_m , respectively. Then, we have the following:

$$\begin{aligned} \tilde{T}_m(\{\log(p_1), \dots, \log(p_m)\}) &\leq \log \mathbb{P}(\mathbf{x}_1 \wedge \dots \wedge \mathbf{x}_m) \\ \tilde{O}_m(\{\log(p_1), \dots, \log(p_m)\}) &\leq \log \mathbb{P}(\mathbf{x}_1 \vee \dots \vee \mathbf{x}_m) \end{aligned}$$

1006 *Proof.* We first show the inequality involving the t-norm \tilde{T}_m . We first observe that \tilde{T}_m is the
 1007 logarithmic of the product t-norm, meaning

$$\exp(\tilde{T}_m(\{\log(p_1), \dots, \log(p_m)\})) = \prod_{i=1}^m p_i.$$

1008 Now, since \mathbf{x}_i 's are mutually independent or correlated, we naturally have $\prod_{i=1}^m p_i \leq$
 1009 $\mathbb{P}(\mathbf{x}_1 \wedge \dots \wedge \mathbf{x}_m)$. This is because for any two random variables \mathbf{x}_i and \mathbf{x}_j ,

$$\mathbb{P}(\mathbf{x}_i \wedge \mathbf{x}_j) = \mathbb{E}[\mathbf{x}_i \wedge \mathbf{x}_j] = \mathbb{E}[\mathbf{x}_i]\mathbb{E}[\mathbf{x}_j] + \text{Cov}(\mathbf{x}_i, \mathbf{x}_j) = p_i p_j + \text{Cov}(\mathbf{x}_i, \mathbf{x}_j) \geq p_i p_j,$$

1010 where the last inequality sign holds because $\text{Cov}(\mathbf{x}_i, \mathbf{x}_j) \geq 0$ for mutually positively correlated or
 1011 independent random variables.

1012 Thus, we have

$$\begin{aligned} \exp(\tilde{T}_m(\{\log(p_1), \dots, \log(p_m)\})) &\leq \prod_{i=1}^m p_i \leq \mathbb{P}(\mathbf{x}_1 \wedge \dots \wedge \mathbf{x}_m) \\ \tilde{T}_m(\{\log(p_1), \dots, \log(p_m)\}) &\leq \log \mathbb{P}(\mathbf{x}_1 \wedge \dots \wedge \mathbf{x}_m). \end{aligned}$$

1013 For the t-conorm \tilde{O}_m , the inequality holds because \tilde{O}_m is a lower bound of the logarithm of max,
 1014 and max is then also a lower bound of the union of random variables. Specifically, as shown in the
 1015 main text, \tilde{O}_m as the LogMeanExp operation has the following property:

$$\exp(\tilde{O}_m(\{\log(p_1), \dots, \log(p_m)\})) \leq \max\{p_1, \dots, p_m\}.$$

1016 On the other hand, max is in general the lower bound of the probability of the union of random
 1017 variables,

$$\max\{\log(p_1), \dots, \log(p_m)\} \leq \mathbb{P}(\mathbf{x}_1 \vee \dots \vee \mathbf{x}_m).$$

1018 Thus, taken together, we have

$$\begin{aligned} \exp(\tilde{O}_m(\{\log(p_1), \dots, \log(p_m)\})) &\leq \max\{p_1, \dots, p_m\} \leq \mathbb{P}(\mathbf{x}_1 \vee \dots \vee \mathbf{x}_m) \\ \tilde{O}_m(\{\log(p_1), \dots, \log(p_m)\}) &\leq \log \mathbb{P}(\mathbf{x}_1 \vee \dots \vee \mathbf{x}_m). \end{aligned}$$

1019 □

1020 **Lemma 3.7** (Reachability Percolation Lower Bound). *Given a weighted adjacency matrix $\mathbf{W} \in$*
 1021 *$[0, 1]^{d \times d}$, for any $0 \leq l < d$, and for any pair of nodes $x, y \in [d]$, we have*

$$\tilde{R}_{\mathbf{W}}^{(l)}(x, y) \leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[R_{\mathbf{A}}^{(l)}(x, y) \right] \text{ and } \tilde{U}_{\mathbf{W}}^{(l)}(x, y) \leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[U_{\mathbf{A}}^{(l)}(x, y) \right].$$

1022 *Proof.* We first show that $\tilde{R}_{\mathbf{W}}^{(l)}(x, y) \leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[R_{\mathbf{A}}^{(l)}(x, y) \right]$.

1023 Intuitively, this inequality holds because in a probabilistic graph \mathbf{W} , where each edge $x \rightarrow y$ is a
 1024 Bernoulli random variable parameterized by $\mathbf{W}_{x,y}$, the paths as random variables are either mutually
 1025 independent, when they consist of distinct edges, or mutually positively correlated, when they share
 1026 some common edges. Thus, using the results of Lemma A.1, the t-norm of the path probabilities
 1027 yields a lower bound to their joint probability, and the t-conorm yields a lower bound to their union
 1028 probability. Thus, the continuous reachability score $\tilde{R}_{\mathbf{W}}^{(l)}$, consisting of disjunctions and conjunctions
 1029 of the path and edge probabilities, also gives lower bounds.

1030 More specifically, we can observe this fact via induction. In the base case, we obviously have

$$\tilde{R}_{\mathbf{W}}^{(0)}(x, y) = \log(\mathbb{1}(x = y)) = \log \mathbb{P}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})}(R_{\mathbf{A}}^{(0)}(x, y)) = \mathbb{P}_{\mathbf{W}}(x \rightarrow y) = \log \mathbf{W}_{x,y}.$$

1031 In addition, we can obviously see that the random variable $R_{\mathbf{A}}^{(0)}(x, u)$ is independent of the random
 1032 variable $\mathbf{A}_{u,y}$ for any nodes $u \in [d]$.

1033 Then, assuming the inequality holds for a graph with a maximum path length of $l - 1$, and that
 1034 the random variable $R_{\mathbf{A}}^{(l-1)}(x, u)$ is either independent of or positively correlated with the random
 1035 variable $\mathbf{A}_{u,y}$ for any nodes $u \in [d]$, we now aim to show that these two statements also hold for l .

1036 First, we can see that, for any node $u \in [d]$,

$$\tilde{T}_2(\tilde{R}_{\mathbf{W}}^{(l-1)}(x, u), \log(\mathbf{W}_{u,y})) \leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[R_{\mathbf{A}}^{(l-1)}(x, u) \wedge \mathbf{A}_{u,y} \right],$$

1037 from result in Lemma A.1 regarding the t-norm \tilde{T}_2 and the induction hypothesis that $R_{\mathbf{A}}^{(l-1)}(x, u)$ is
 1038 either independent of or positively correlated with $\mathbf{A}_{u,y}$.

1039 Then, again using the result in Lemma A.1 regarding the t-conorm \tilde{O}_m , we have

$$\begin{aligned}\tilde{R}_{\mathbf{W}}^{(l)}(x, y) &= \tilde{O}_{d+1}\left(\{\tilde{T}_2(\tilde{R}_{\mathbf{W}}^{(l-1)}(x, u), \log(W_{uy}))\}_{u \in [d]} \cup \{\tilde{R}_{\mathbf{W}}^{(l-1)}(x, y)\}\right) \\ &\leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[\left(\bigvee_{u \in [d]} \left(R_{\mathbf{A}}^{(l-1)}(x, u) \wedge \mathbf{A}_{u,y} \right) \right) \vee R_{\mathbf{A}}^{(l-1)}(x, y) \right] \\ &= \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[R_{\mathbf{A}}^{(l)}(x, y) \right].\end{aligned}$$

1040 Regarding the relationship between $R_{\mathbf{A}}^{(l)}(x, u)$ and $\mathbf{A}_{u,y}$, we can see that the probability of $R_{\mathbf{A}}^{(l)}(x, u)$
 1041 increases if the probability of $\mathbf{A}_{u,y}$ also increases. This is because in the formula, there is no negation
 1042 and $\mathbf{A}_{u,y}$ makes a possible contribution to the value of $R_{\mathbf{A}}^{(l)}(x, u)$. Thus, $R_{\mathbf{A}}^{(l)}(x, u)$ and $\mathbf{A}_{u,y}$ is
 1043 positively correlated.

1044 Now, we show for the unreachability inequality, $\tilde{U}_{\mathbf{W}}^{(l)}(x, y) \leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[U_{\mathbf{A}}^{(l)}(x, y) \right]$.

1045 Similarly, we show by induction. In the base case, we have

$$\tilde{U}_{\mathbf{W}}^{(0)}(x, y) = \log(\mathbb{1}(x \neq y)) = \log \mathbb{P}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})}(\neg R_{\mathbf{A}}^{(0)}(x, y)) = \log(1 - \mathbf{W}_{x,y}).$$

1046 In addition, we can see that the random variable $U_{\mathbf{A}}^{(0)}(x, u)$ is independent of the random variable
 1047 $\neg R_{\mathbf{A}}^{(0)}(x, y)$.

1048 Then, assuming the inequality holds for a graph with a maximum path length of $l - 1$, and that
 1049 the random variable $U_{\mathbf{A}}^{(l-1)}(x, u)$ is either independent of or positively correlated with the random
 1050 variable $\neg R_{\mathbf{A}}^{(0)}(x, y)$. We aim to show that these two properties still hold for l .

1051 First, using the result in Lemma A.1 regarding the t-conorm \tilde{O}_m , we have, for all nodes $u \in [d]$,

$$\tilde{O}_2(\tilde{U}_{\mathbf{W}}^{(l-1)}(x, u), \log(1 - W_{uy})) \leq \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[U_{\mathbf{A}}^{(l-1)}(x, u) \vee \neg R_{\mathbf{A}}^{(0)}(u, y) \right].$$

1052 Then, we note that the random variable $(U_{\mathbf{A}}^{(l-1)}(x, u) \vee \neg R_{\mathbf{A}}^{(0)}(u, y))$ for different u are mutually
 1053 independent or positively correlated. They are also independent of or positively correlated with
 1054 $U_{\mathbf{A}}^{(l-1)}(x, y)$. This is similar to the mutual independence or positive correlation among paths, since
 1055 for these “non-paths”, if they consist of distinct “non-edges”, then they are independent, otherwise
 1056 they are positively correlated, since increasing the probability of the shared “non-edge” (or decreasing
 1057 the probability of the shared edge) would simultaneously increase the probabilities of said “non-paths.”
 1058 Thus, again using the result in Lemma A.1 regarding the t-norm \tilde{T}_{d+1} , we have

$$\begin{aligned}\tilde{U}_{\mathbf{W}}^{(l)}(x, y) &= \tilde{T}_{d+1}\left(\{\tilde{O}_2(\tilde{U}_{\mathbf{W}}^{(l-1)}(x, u), \log(1 - W_{uy}))\}_{u \in [d]} \cup \{\tilde{U}_{\mathbf{W}}^{(l-1)}(x, y)\}\right) \\ &\leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[\left(\bigwedge_{u \in [d]} \left(U_{\mathbf{A}}^{(l-1)}(x, u) \vee \neg \mathbf{A}_{u,y} \right) \right) \wedge U_{\mathbf{A}}^{(l-1)}(x, y) \right] \\ &= \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[U_{\mathbf{A}}^{(l)}(x, y) \right].\end{aligned}$$

1059 □

1060 **Theorem 3.8** (Lower Bound on Expected d -Separation Statements). *Given a weighted adjacency*
 1061 *matrix $\mathbf{W} \in [0, 1]^{d \times d}$, for any three nodes $x, y, z \in [d]$,*

$$\begin{aligned}\tilde{S}_{\mathbf{W}}^{(0)}(x, y) &\leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[S_{\mathbf{A}}^{(0)}(x, y) \right], & \tilde{S}_{\mathbf{W}}^{(1)}(x, y \mid z) &\leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[S_{\mathbf{A}}^{(1)}(x, y \mid z) \right], \\ \tilde{C}_{\mathbf{W}}^{(0)}(x, y) &\leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[C_{\mathbf{A}}^{(0)}(x, y) \right], & \tilde{C}_{\mathbf{W}}^{(1)}(x, y \mid z) &\leq \log \mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} \left[C_{\mathbf{A}}^{(1)}(x, y \mid z) \right].\end{aligned}$$

1062 *Proof.* First, we reiterate that, in the computation of differentiable d-separation $\tilde{S}_{\mathbf{W}}^{(0)}$ and $\tilde{S}_{\mathbf{W}}^{(1)}$, we use
 1063 the continuous unreachability score $\tilde{U}_{\mathbf{W}}^{(d)}$ directly, rather than taking the negation of the continuous
 1064 reachability score.

1065 Using a similar reasoning as in the proof of the previous lemma, we first observe that, when treating
 1066 $S_{\mathbf{A}}^{(0)}, S_{\mathbf{A}}^{(1)}, C_{\mathbf{A}}^{(0)}$, and $C_{\mathbf{A}}^{(1)}$ as random variables when $\mathbf{A} \sim \text{Bern}(\mathbf{W})$, all terms in the d-separation/d-
 1067 connection formulæ (Definition 3.1) are either mutually independent or positively correlated. In
 1068 particular, this holds true for $S_{\mathbf{A}-z}^{(0)}(x, a)$ and $\neg R_{\mathbf{A}}(a, z)$ for any $a \in [d]$, as well as for $C_{\mathbf{A}-z}^{(0)}(y, b)$
 1069 and $R_{\mathbf{A}}(b, z)$ for any $b \in [d]$. Thus, using the results from Lemma A.1, we have the lower bounds as
 1070 stated in the theorem. \square

1071 **Lemma 4.2** (Consistency of Multi-Task CI Losses). *Given a faithful data \mathcal{D} , as the sample size*
 1072 *$n \rightarrow \infty$ and the LogMeanExp temperature $\alpha \rightarrow 0^+$ (Equation (8)), the optimal DAG \mathbf{A}^* achieves*
 1073 *the minimum value on each of $\mathcal{L}_{TP-0}, \mathcal{L}_{TP-1}, \mathcal{L}_{TN-0}, \mathcal{L}_{TN-1}$, and \mathcal{L}_{DAG} .*

1074 *Proof.* Let $\theta^* \in \mathbb{R}^{d \times d}$ be the parameter that approximates the optimal binary DAG \mathbf{A}^* of the given
 1075 data \mathcal{D} via $\sigma(\theta^*) \rightarrow \mathbf{A}^*$.

1076 When the sample size $n \rightarrow \infty$ in a faithful data \mathcal{D} , we have $p_{\mathcal{D}}(x, y) \rightarrow 0$ and $p_{\mathcal{D}}(x, y | z) \rightarrow 0$
 1077 when $x \not\perp\!\!\!\perp_{\mathbf{A}^*} y$ and $x \not\perp\!\!\!\perp_{\mathbf{A}^*} y | z$, and conversely $p_{\mathcal{D}}(x, y) \rightarrow 1$ and $p_{\mathcal{D}}(x, y | z) \rightarrow 1$ when
 1078 $x \perp\!\!\!\perp_{\mathbf{A}^*} y$ and $x \perp\!\!\!\perp_{\mathbf{A}^*} y | z$ for any $x, y, z \in [d]$, where \mathbf{A}^* is the optimal causal graph of \mathcal{D} . When
 1079 the LogMeanExp temperature $\alpha \rightarrow 0^+$, we have $\log \max\{x_i\}_{i=1}^m - \tilde{O}_m(\{\log(x_i)\}_{i=1}^m) \rightarrow 0^+$.

1080 In addition, we note that when given a sequence of binary random variables $\mathbf{x}_1, \dots, \mathbf{x}_m$, whose
 1081 Bernoulli probabilities approach in the limit to either 0 or 1, the product t-norm T_m and max t-
 1082 conorm O_m over \mathbf{x}_i 's probabilities approach in the limit to the probability of $\mathbb{P}(\mathbf{x}_1 \wedge \dots \wedge \mathbf{x}_m)$ and
 1083 $\mathbb{P}(\mathbf{x}_1 \vee \dots \vee \mathbf{x}_m)$ respectively. This is because in the limit case, the random variables reduce to
 1084 constant 0's and constant 1's, in which case the product t-norm and t-conorm reduce to the vanilla
 1085 Boolean operations of conjunction and disjunction. Thus, combining the fact that LogLTN's [1]
 1086 t-norm \tilde{T}_m is equivalent to T_m in the logarithm, and its t-conorm \tilde{O}_m approaches in limit to O_m in
 1087 the logarithm, we have the following:

$$\begin{aligned} \log \mathbb{P}(\mathbf{x}_1 \wedge \dots \wedge \mathbf{x}_m) - \tilde{T}_m(\{\log \mathbb{P}(\mathbf{x}_i)\}_{i=1}^m) &\rightarrow 0^+ \\ \log \mathbb{P}(\mathbf{x}_1 \vee \dots \vee \mathbf{x}_m) - \tilde{O}_m(\{\log \mathbb{P}(\mathbf{x}_i)\}_{i=1}^m) &\rightarrow 0^+. \end{aligned}$$

1088 Using this property, we can see that for the continuous reachability and unreachability, for any given
 1089 maximum path length l and any nodes $x, y \in [d]$, they satisfy

$$\begin{aligned} R_{\mathbf{A}^*}^{(l)}(x, y) - R_{\sigma(\theta^*)}^{(l)}(x, y) &\rightarrow 0^+ \\ U_{\mathbf{A}^*}^{(l)}(x, y) - U_{\sigma(\theta^*)}^{(l)}(x, y) &\rightarrow 0^+. \end{aligned}$$

1090 Consequently, for the differentiable d-separation and d-connection scores, for any nodes $x, y, z \in [d]$,
 1091 they satisfy

$$\begin{aligned} S_{\mathbf{A}^*}^{(0)}(x, y) - \tilde{S}_{\sigma(\theta^*)}^{(0)}(x, y) &\rightarrow 0^+, & S_{\mathbf{A}^*}^{(1)}(x, y | z) - \tilde{S}_{\sigma(\theta^*)}^{(1)}(x, y | z) &\rightarrow 0^+, \\ C_{\mathbf{A}^*}^{(0)}(x, y) - \tilde{C}_{\sigma(\theta^*)}^{(0)}(x, y) &\rightarrow 0^+, & C_{\mathbf{A}^*}^{(1)}(x, y | z) - \tilde{C}_{\sigma(\theta^*)}^{(1)}(x, y | z) &\rightarrow 0^+. \end{aligned}$$

1092 Furthermore, recall from Theorem 3.2, since \mathbf{A}^* is a discrete DAG, $S_{\mathbf{A}^*}^{(0)}, S_{\mathbf{A}^*}^{(1)}, C_{\mathbf{A}^*}^{(0)}$, and $C_{\mathbf{A}^*}^{(1)}$
 1093 reduces exactly to the ground-truth d-separation and d-connection statement values in \mathbf{A}^* . Thus,
 1094 combining with the result above, we have that

$$\begin{aligned} \tilde{S}_{\sigma(\theta^*)}^{(0)}(x, y) &\rightarrow \begin{cases} 1 & \text{when } x \perp\!\!\!\perp_{\mathbf{A}^*} y \\ 0 & \text{otherwise} \end{cases}, & \tilde{S}_{\sigma(\theta^*)}^{(1)}(x, y | z) &\rightarrow \begin{cases} 1 & \text{when } x \perp\!\!\!\perp_{\mathbf{A}^*} y | z \\ 0 & \text{otherwise} \end{cases}, \\ \tilde{C}_{\sigma(\theta^*)}^{(0)}(x, y) &\rightarrow \begin{cases} 1 & \text{when } x \not\perp\!\!\!\perp_{\mathbf{A}^*} y \\ 0 & \text{otherwise} \end{cases}, & \tilde{C}_{\sigma(\theta^*)}^{(1)}(x, y | z) &\rightarrow \begin{cases} 1 & \text{when } x \not\perp\!\!\!\perp_{\mathbf{A}^*} y | z \\ 0 & \text{otherwise} \end{cases}, \end{aligned}$$

Therefore, combining with the values of the p-values $p_{\mathcal{D}}$, we have the following value matching between the model's predicted d-separation and d-connection score and the p-values. For all $(x, y) \in \mathbb{I}_0$ ($\mathbb{I}_0 = \{(x, y) \mid x, y \in [d], x > y\}$) and for all $(x, y, z) \in \mathbb{I}_1$ ($\mathbb{I}_1 = \{(x, y, z) \mid x, y, z \in [d], x > y, x \neq z, y \neq z\}$),

$$\begin{aligned}\tilde{S}_{\sigma(\theta^*)}^{(0)}(x, y) &\rightarrow \begin{cases} 1 & \text{when } p_{\mathcal{D}}(x, y) \rightarrow 1 \\ 0 & \text{when } p_{\mathcal{D}}(x, y) \rightarrow 0 \end{cases}, & \tilde{S}_{\sigma(\theta^*)}^{(1)}(x, y \mid z) &\rightarrow \begin{cases} 1 & \text{when } p_{\mathcal{D}}(x, y \mid z) \rightarrow 1 \\ 0 & \text{when } p_{\mathcal{D}}(x, y \mid z) \rightarrow 0 \end{cases}, \\ \tilde{C}_{\sigma(\theta^*)}^{(0)}(x, y) &\rightarrow \begin{cases} 1 & \text{when } M_0 - p_{\mathcal{D}}(x, y) \rightarrow 1 \\ 0 & \text{when } M_0 - p_{\mathcal{D}}(x, y) \rightarrow 0 \end{cases}, & \tilde{C}_{\sigma(\theta^*)}^{(1)}(x, y \mid z) &\rightarrow \begin{cases} 1 & \text{when } M_1 - p_{\mathcal{D}}(x, y \mid z) \rightarrow 1 \\ 0 & \text{when } M_1 - p_{\mathcal{D}}(x, y \mid z) \rightarrow 0 \end{cases},\end{aligned}$$

where $M_0 = \max_{\mathbb{I}_0} p_{\mathcal{D}}(x, y) \rightarrow 1$, and similarly $M_1 = \max_{\mathbb{I}_1} p_{\mathcal{D}}(x, y \mid z) \rightarrow 1$. In other words, the differentiable d-separation/d-connection scores over the optimal parameter θ^* and the p-values from the data have zero mismatch.

Thus, the TP and TN losses have the following values:

$$\begin{aligned}\mathcal{L}_{\text{TP-0}}(\theta, \mathcal{D}) &\rightarrow - \sum_{(x, y) \in \mathbb{I}_0} \mathbb{1}[x \perp\!\!\!\perp_{\mathbf{A}^*} y], & \mathcal{L}_{\text{TP-1}}(\theta, \mathcal{D}) &\rightarrow - \sum_{(x, y, z) \in \mathbb{I}_1} \mathbb{1}[x \perp\!\!\!\perp_{\mathbf{A}^*} y \mid z], \\ \mathcal{L}_{\text{TN-0}}(\theta, \mathcal{D}) &\rightarrow - \sum_{(x, y) \in \mathbb{I}_0} \mathbb{1}[x \not\perp\!\!\!\perp_{\mathbf{A}^*} y], & \mathcal{L}_{\text{TN-1}}(\theta, \mathcal{D}) &\rightarrow - \sum_{(x, y, z) \in \mathbb{I}_1} \mathbb{1}[x \not\perp\!\!\!\perp_{\mathbf{A}^*} y \mid z],\end{aligned}$$

which are the lowest possible values that these loss functions can achieve, because any other configuration of the values for the differentiable d-separation/d-connection scores would result in a larger loss value due to possible mismatches with the p-values. Thus, the TP and TN losses are consistent.

Finally, since \mathbf{A}^* is a DAG, the log-det acyclicity loss \mathcal{L}_{DAG} also achieves the minimum value, for any value of the hyperparameter s . This property is proved in Bello et al. [2]. Thus, all loss functions in Definition 4.1 are consistent. \square

Lemma 4.3 (Mutual Independence of Low-order CI Statements). *All 0th- and 1st-order CI statements over the variable sets $\mathbb{I}_0 = \{(x, y) \mid x, y \in [d], x > y\}$ and $\mathbb{I}_1 = \{(x, y, z) \mid x, y, z \in [d], x > y, x \neq z, y \neq z\}$ are mutually independent. In other words, none of these CI statements can be implied from any other of these CI statements via the graphoid axioms [19].*

Proof. Let $\mathcal{X}, \mathcal{Y} \subseteq [d]$ be any non-empty node sets, and $\mathcal{Z}, \mathcal{W} \subseteq [d]$ be node sets that could be empty. The graphoid axiom [19] states the following five rules describing the relationship and dependencies between different CI statements in a graphoid dependency model:

1. Symmetry: $\mathcal{X} \perp\!\!\!\perp \mathcal{Y} \mid \mathcal{Z} \iff \mathcal{Y} \perp\!\!\!\perp \mathcal{X} \mid \mathcal{Z}$
2. Decomposition: $\mathcal{X} \perp\!\!\!\perp \mathcal{Y} \cup \mathcal{W} \mid \mathcal{Z} \implies (\mathcal{X} \perp\!\!\!\perp \mathcal{Y} \mid \mathcal{Z}) \wedge (\mathcal{X} \perp\!\!\!\perp \mathcal{W} \mid \mathcal{Z})$
3. Weak Union: $\mathcal{X} \perp\!\!\!\perp \mathcal{Y} \cup \mathcal{W} \mid \mathcal{Z} \implies (\mathcal{X} \perp\!\!\!\perp \mathcal{Y} \mid \mathcal{Z} \cup \mathcal{W}) \wedge (\mathcal{X} \perp\!\!\!\perp \mathcal{W} \mid \mathcal{Z} \cup \mathcal{Y})$
4. Contraction: $(\mathcal{X} \perp\!\!\!\perp \mathcal{Y} \mid \mathcal{Z}) \wedge (\mathcal{X} \perp\!\!\!\perp \mathcal{W} \mid \mathcal{Z} \cup \mathcal{Y}) \implies \mathcal{X} \perp\!\!\!\perp \mathcal{Y} \cup \mathcal{W} \mid \mathcal{Z}$
5. Intersection: $(\mathcal{X} \perp\!\!\!\perp \mathcal{Y} \mid \mathcal{Z} \cup \mathcal{W}) \wedge (\mathcal{X} \perp\!\!\!\perp \mathcal{W} \mid \mathcal{Z} \cup \mathcal{Y}) \implies \mathcal{X} \perp\!\!\!\perp \mathcal{Y} \cup \mathcal{W} \mid \mathcal{Z}$

We check rule by rule whether any of the low-order CI statements considered in \mathbb{I}_0 and \mathbb{I}_1 can be implied by any others.

1. Symmetry: No CI statement in \mathbb{I}_0 and \mathbb{I}_1 can be inferred from others via the symmetry rule. This is because \mathbb{I}_0 and \mathbb{I}_1 consists of only asymmetric statements where always $x > y$.
2. Decomposition: In the non-trivial case when $|\mathcal{Y}| \geq 1$ and $|\mathcal{W}| \geq 1$, this rule does not apply because its left-hand side (LHS) statement involves nodes set $\mathcal{Y} \cup \mathcal{W}$ with cardinality of at least 2. This type of statement is not considered in \mathbb{I}_0 or \mathbb{I}_1 .
3. Weak Union: Similarly, in the non-trivial case when $|\mathcal{Y}| \geq 1$ and $|\mathcal{W}| \geq 1$, this rule does not apply because its LHS statement is not included in \mathbb{I}_0 or \mathbb{I}_1 .
4. In the non-trivial case when $|\mathcal{Y}| \geq 1$ and $|\mathcal{W}| \geq 1$, the LHS applies, but the right-hand-side (RHS) involve node set $\mathcal{Y} \cup \mathcal{W}$ with cardinality of at least 2, which is not included in \mathbb{I}_0 or \mathbb{I}_1 .

1132 5. Intersection: Similarly, in the non-trivial case when $|\mathcal{Y}| \geq 1$ and $|\mathcal{W}| \geq 1$, the RHS involve node
 1133 set $\mathcal{Y} \cup \mathcal{W}$ with cardinality of at least 2, which is not included in \mathbb{I}_0 or \mathbb{I}_1 .

1134 Thus, no CI statements in \mathbb{I}_0 and \mathbb{I}_1 can be inferred from other statements in \mathbb{I}_0 and \mathbb{I}_1 via the graphoid
 1135 axioms. Thus, the CI statements we considered in our approach are mutually independent from the
 1136 perspective of the graphoid dependency structure, justifying the use of the sum aggregation over these
 1137 CI statements in the TP and TN losses. \square

1138 B Further Discussions on Percolation versus Diffusion

1139 The distinction between graph percolation and graph diffusion is fundamental to understanding
 1140 the probabilistic interpretation of our differentiable d-separation framework. Percolation theory
 1141 studies graph properties (such as connectivity or reachability) when the graph structure itself is
 1142 randomized [8, 12]. In contrast, diffusion theory typically assumes a fixed graph structure, with
 1143 randomness arising from particles making probabilistic traversal decisions across this deterministic
 1144 environment.

1145 In the context of our weighted adjacency matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$, these perspectives offer different
 1146 interpretations. From a percolation standpoint, each entry $\mathbf{W}_{x,y}$ represents the probability that edge
 1147 $x \rightarrow y$ exists in the graph. The reachability question then becomes: what is the probability that a
 1148 directed path exists from one node to another, considering all possible graph configurations? From a
 1149 diffusion perspective, \mathbf{W} would instead represent transition probabilities on a complete graph, where
 1150 at node x , the weights $\{\mathbf{W}_{x,u}\}_{u \in [d]}$ (possibly normalized) determine the probability distribution of
 1151 a particle’s next position. The key distinction is that percolation randomizes the environment (the
 1152 graph structure), while diffusion randomizes the trajectory through a fixed environment.

1153 It is therefore most appropriate to frame our problem as a graph percolation problem. Given a
 1154 weighted adjacency matrix \mathbf{W} , we interpret it as parametrizing a distribution $\text{Bern}(\mathbf{W})$ from which
 1155 random discrete graphs are sampled, $\mathbf{A} \sim \text{Bern}(\mathbf{W})$. Our goal is to estimate the expected reachability
 1156 (with path lengths up to d), $\mathbb{E}_{\mathbf{A} \sim \text{Bern}(\mathbf{W})} [R_{\mathbf{A}}^{(d)}(x, y)]$, where the randomness comes from the graph
 1157 structure itself.

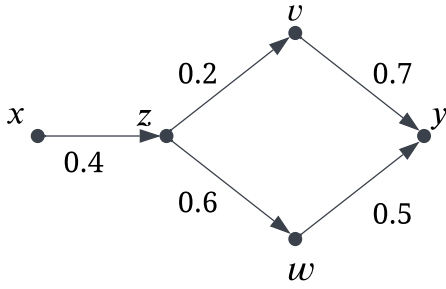
1158 This insight guides our estimation approach for the percolation-based reachability. We observe
 1159 that our generalized Bellman-Ford-based differentiable reachability score (Definition 3.5) provides
 1160 a lower bound on this expectation, while graph diffusion methods like Random Walk algorithms
 1161 typically yield upper bounds. This distinction becomes clear when considering two potential paths \mathbf{p}_1
 1162 and \mathbf{p}_2 from node x to node y that share some edges. In percolation theory, reachability requires only
 1163 that at least one path forms completely. Our Bellman-Ford approach with max operators captures this
 1164 intuition—when one path’s formation probability approaches 1, additional paths contribute minimally
 1165 to the overall reachability probability. The lower bound property stems from the approximation gap
 1166 between product-max t-norms/t-conorms and the true intersection/union of events, combined with
 1167 the approximation in LogLTN’s [1] LogMeanExp operator (Equation (8)). Formal proof of this lower
 1168 bound property can be found in the proof for Lemma 3.7 in Appendix A. Consequently, our method
 1169 will systematically *underestimate* true percolation-based reachability.

1170 Conversely, in diffusion models, particles make independent decisions at each node. When two
 1171 paths overlap, the probability of a particle reaching the target via either path is treated additively
 1172 rather than as a union probability. This independence assumption fails to account for the correlation
 1173 between overlapping paths in the percolation setting, causing diffusion methods to *overestimate* the
 1174 true percolation-based reachability.

1175 Figure 5 provides a concrete example illustrating the distinction between our max-product Bellman-
 1176 Ford reachability and graph diffusion scores. The example shows a weighted adjacency matrix \mathbf{W}
 1177 with 5 nodes, where edge weights are as indicated in the figure (with zeros elsewhere). For the
 1178 diffusion-based method, we choose the typical Random Walk algorithm, where we assume self-loops
 1179 at each node so that the sum of self-loop probability and outgoing edge probabilities equals 1 (e.g.,
 1180 node x has a self-loop probability of 0.6).

1181 Consider two possible paths from x to y : path $\mathbf{p}_1 = x \rightarrow z \rightarrow v \rightarrow y$ and path $\mathbf{p}_2 = x \rightarrow z \rightarrow$
 1182 $w \rightarrow y$. \mathbf{p}_1 and \mathbf{p}_2 are two random variables under the graph distribution $\text{Bern}(\mathbf{W})$. The true
 1183 percolation-based reachability probability, denoted by $\mathbb{P}(x \rightsquigarrow y)$, is computed as the union of the

Random Graph W :



True Reachability Probability (Percolation):

$$\begin{aligned}\mathbb{P}(x \rightsquigarrow y) &= 0.4 \cdot (0.2 \cdot 0.7 + 0.6 \cdot 0.5) \\ &\quad - 0.2 \cdot 0.7 \cdot 0.6 \cdot 0.5 \\ &= 0.1592\end{aligned}$$

Max-Product Bellman-Ford is Lower Bound:

$$\begin{aligned}R_{\mathbf{W}}^{(3)}(x, y) &= 0.4 \cdot \max\{0.2 \cdot 0.7, 0.6 \cdot 0.5\} \\ &= 0.12\end{aligned}$$

Random Walk (Diffusion) is Upper Bound:

$$\begin{aligned}\mathbb{P}^{(\text{RW})}(x \rightsquigarrow y) &= 0.4 \cdot (0.2 \cdot 0.7 + 0.6 \cdot 0.5) \\ &= 0.176\end{aligned}$$

Figure 5: Example random graph to illustrate that the Max-Product Bellman-Ford reachability computes a lower bound of the true percolation-based reachability probability, whereas diffusion-based random walk computes an upper bound. Marked numbers are the edge probabilities. For the random walk computation, we assume a model with self-loop probabilities, e.g. $P(x \rightarrow x) = 0.6$ so that the transition probabilities to other nodes and the self-loop probabilities sum to one.

1184 events of these two paths forming: $\mathbb{P}(\mathbf{p}_1 \vee \mathbf{p}_2)$. Since these paths share the edge $x \rightarrow z$ but diverge
 1185 afterward, we must carefully apply probability theory on union of events. For the divergent segments,
 1186 the probability is:

$$\mathbb{P}((z \rightarrow v \rightarrow y) \vee (z \rightarrow w \rightarrow y)) = \mathbb{P}(z \rightarrow v \rightarrow y) + \mathbb{P}(z \rightarrow w \rightarrow y) - \mathbb{P}((z \rightarrow v \rightarrow y) \wedge (z \rightarrow w \rightarrow y))$$

1187 Because these two sub-paths share no edges, they are independent, giving $\mathbb{P}((z \rightarrow v \rightarrow y) \wedge$
 1188 $(z \rightarrow w \rightarrow y)) = \mathbb{P}(z \rightarrow v \rightarrow y) \cdot \mathbb{P}(z \rightarrow w \rightarrow y)$. Calculating the full expression yields
 1189 $\mathbb{P}(x \rightsquigarrow y) = 0.1592$.

1190 Our max-product Bellman-Ford approach approximates this union using the max operator: for the
 1191 divergent sub-paths from z to y , we compute $R_{\mathbf{W}}^{(2)}(z, y) = \max\{\mathbb{P}(z \rightarrow v \rightarrow y), \mathbb{P}(z \rightarrow w \rightarrow y)\}$.
 1192 Since the max operator always provides a lower bound for the union of events, our approach yields
 1193 the final value $R_{\mathbf{W}}^{(3)}(x, y) = 0.12$, which underestimates the true reachability probability. In contrast,
 1194 the diffusion-based Random Walk method, denoted by $\mathbb{P}^{(\text{RW})}(x \rightsquigarrow y)$, adds probabilities additively
 1195 across possible paths: $\mathbb{P}^{(\text{RW})}(x \rightsquigarrow y) = \mathbb{P}(\mathbf{p}_1) + \mathbb{P}(\mathbf{p}_2)$. This leads to an overestimation of the true
 1196 reachability probability, giving $\mathbb{P}^{(\text{RW})}(x \rightsquigarrow y) = 0.176$. This example clearly demonstrates why our
 1197 max-product approach provides a principled lower bound on percolation-based reachability, while
 1198 diffusion-based methods typically yield upper bounds.

1199 Our max-product Bellman-Ford approach is thus particularly well-suited for our proposed model
 1200 instantiation, DAGPA, as it provides consistent lower bounds throughout computation, ensuring
 1201 a coherent probabilistic interpretation. While we focus on lower-bound estimation, an alternative
 1202 approach could theoretically estimate upper bounds on expected d-separation/d-connection statements.
 1203 For such a method, diffusion-based approaches like Random Walk would be more appropriate for
 1204 estimating graph reachability. However, this would require identifying a different pair of t-norm/t-
 1205 conorm differentiable logical operators that yield upper bounds instead of lower bounds. Based on
 1206 van Krieken et al. [26]’s comprehensive analysis of t-norm and t-conorm suitability for differentiable
 1207 learning, we have not identified a configuration that simultaneously provides upper bounds and
 1208 maintains gradient stability comparable to our max-product operators. This represents an interesting
 1209 direction for future research, potentially expanding the theoretical foundations of differentiable
 1210 d-separation frameworks, and we encourage future work to explore this avenue.

1211 C Full Algorithm and Implementation Details of DAGPA

1212 DAGPA, our proposed instantiation of the differentiable d-separation framework, employs multiple
 1213 techniques to address optimization challenges and result validation. This section provides comprehen-
 1214 sive implementation details for all techniques used in our approach. We first discuss the multi-task

1215 gradient conflict resolution technique (Section C.1) and the Bayesian sampling method for enhanced
 1216 weight space exploration (Section C.2). Since DAGPA uses Bayesian sampling, each optimization
 1217 step generates a new candidate causal graph. Consequently, we require a principled procedure to
 1218 select the best DAGs from those sampled, using only information available from the input data. We
 1219 present this DAG selection heuristic in Section C.3. Finally, we provide the complete algorithm
 1220 pseudocode in Section C.4.

1221 C.1 Multitask optimizer to address conflicting gradients

1222 The five distinct loss functions—0th and 1st-
 1223 order TP/TN losses plus the DAG acyclicity
 1224 constraint (Definition 4.1)—render our opti-
 1225 mization a multi-objective problem. Recent
 1226 research in multi-task learning has demon-
 1227 strated that naively summing these loss func-
 1228 tions leads to theoretically suboptimal con-
 1229 vergence, as contradictory gradient direc-
 1230 tions among tasks can impede optimization
 1231 progress [15, 17, 23, 27, 29]. This challenge
 1232 is particularly pronounced in our framework,
 1233 where the TP losses encouraging d-separation
 1234 naturally conflict with the TN losses that re-
 1235 quire d-connection. To address these inher-
 1236 ent gradient conflicts, we adopt PCGrad [29],
 1237 a gradient projection technique that resolves
 1238 conflicting directions by projecting one gradi-
 1239 ent onto the normal plane of the other when they conflict. We reproduce the PCGrad algorithm [29]
 1240 adopted to our setting in Algorithm 1.

Algorithm 1 PCGrad [29] Gradient Projection

Require: Model parameters θ , number of tasks K ,
 loss functions $\{\mathcal{L}_k\}_{k=1}^K$
 1: $\mathbf{g}_k \leftarrow \nabla_{\theta} \mathcal{L}_k(\theta) \quad \forall k \in [K]$
 2: $\mathbf{g}_k^{\text{PC}} \leftarrow \mathbf{g}_k \quad \forall k \in [K]$
 3: **for** $i = 1 : K$ **do**
 4: **for** $j \stackrel{\text{uniformly}}{\sim} [K] \setminus i$ **in random order do**
 5: **if** $\mathbf{g}_i^{\text{PC}} \cdot \mathbf{g}_j < 0$ **then**
 6: ▷ Subtract projection of \mathbf{g}_i^{PC} onto \mathbf{g}_j
 7: $\mathbf{g}_i^{\text{PC}} \leftarrow \mathbf{g}_i^{\text{PC}} - \frac{\mathbf{g}_i^{\text{PC}} \cdot \mathbf{g}_j}{\|\mathbf{g}_j\|^2} \mathbf{g}_j$
 8: **return** update $\Delta\theta^{(\text{PC})} = \mathbf{g}^{\text{PC}} = \sum_{i=1}^K \mathbf{g}_i^{\text{PC}}$

1241 C.2 Gradient-informed discrete Bayesian sampling of W

1242 Rather than using conventional stochastic gradient descent (SGD), we adopt Discrete Langevin
 1243 Proposal (DLP) [33], a gradient-informed discrete Bayesian sampling technique, to sample parameters
 1244 θ that yield low-loss configurations⁴. Our choice of DLP is motivated by several key factors. First,
 1245 our empirical observations show that SGD often becomes trapped in local minima, possibly due to
 1246 the highly non-convex loss landscape, in which case Bayesian sampling can naturally escape local
 1247 minima and explores the weight space more effectively. Second, we favor discrete over continuous
 1248 sampling because the most likely DAG \mathbf{A} from a weighted adjacency matrix \mathbf{W} is obtained via
 1249 thresholding at 0.5. This threshold divides the parameter space into discrete regions, and continuous
 1250 optimization or sampling approaches spend significant computational effort exploring within a single
 1251 region, repeatedly yielding the same causal graph structure. In contrast, DLP’s discrete support
 1252 enables efficient exploration across different regions while leveraging gradient information from the
 1253 continuous space to guide proposals.

1254 To adopt DLP for DAGPA, we first restrict the parameter space from the real space $\mathbb{R}^{d \times d}$ to a space
 1255 of discrete supports, $\mathbb{D}^{d \times d}$, where $\mathbb{D} = \{D_1, \dots, D_m \mid D_i < D_j, \forall i < j\}$ is a finite set of values.
 1256 Furthermore, we require $D_1 < 0$ and $D_m > 0$, so that for any pair of nodes (x, y) , if $\theta_{x,y} = D_1$,
 1257 then $W_{x,y} = \sigma(\theta_{x,y}) < 0.5$, which allows us to interpret x, y as more likely to not having an edge.
 1258 Similarly, if $\theta_{x,y} = D_m$, then $W_{x,y} = \sigma(\theta_{x,y}) > 0.5$, which allows us to interpret x, y as more
 1259 likely to have an edge $x \rightarrow y$.

1260 In reality through empirical experiments, we found that $\mathbb{D} = \{-2.0, 0.0, 2.0\}$ works quite well. We
 1261 hypothesize this could be due to -2.0 and 2.0 as sigmoid logits are not too extreme so that they enable
 1262 smooth and meaningful gradients, while also sufficiently far apart for the differentiable d-separation
 1263 formulæ. Furthermore, the “middle point” 0.0 allows DAGPA to model uncertain edges, whose final
 1264 values can only to be decided after other edges are settled.

⁴Note that this differs from the discrete DAG sampling $\mathbf{A} \sim \text{Bern}(\mathbf{W})$ in Section 3.2, which serves to theoretically establish the probabilistic interpretation of our continuous d-separation and d-connection values.

The next step to adopt DLP [33] is to define the proposal sampling distribution and the Metropolis-Hastings(MH) acceptance-rejection step [11, 16]. This is the discrete Metropolis-adjusted Langevin algorithm (DMALA) variant of DLP algorithm [33]. One of DLP’s novel innovations is the parallel sampling of the parameters θ when its proposal distribution can be factorized along the dimensions, meaning at every step, multiple θ_i for different i ’s may have their values updated. This technique significantly enhances DLP’s efficiency compared to other discrete sampling methods. DAGPA fits this requirement as we can formulate a factorizable proposal distribution. Specifically, we adopt Equation (2) from Zhang et al. [33], but change the gradients to those obtained via PCGrad [29] from Algorithm 1. That is, let $q(\theta' | \theta)$ be the proposal distribution. It can be factorized along the d dimensions, $q(\theta' | \theta) = \prod_{i=1}^d q_i(\theta'_i | \theta)$, where $q_i(\theta'_i | \theta)$ is a categorical distribution of the form:

$$q_i(\theta'_i | \theta) := \text{Categorical} \left(\text{Softmax} \left(\frac{1}{2} \Delta \theta^{(\text{PC})} (\theta_i - \theta'_i) - \frac{(\theta_i - \theta'_i)^2}{2\beta} \right) \right), \quad (9)$$

where $\Delta \theta^{(\text{PC})}$ is the projected gradients given by the PCGrad algorithm (Algorithm 1), and β is a hyperparameter controlling the DLP step size.

After a proposal θ' is sampled from θ according to Equation (9), we perform an MH acceptance-rejection step. Intuitively, this step is to ensure that we are not taking a step too far and landing into a “bad region” in the parameter space. Mathematically, this step is to ensure the Markov chain is reversible. Here, we adopt Equation (3) from Zhang et al. [33], but for the (negative) energy function, we simply adopt the heuristic of summing the multi-task losses (Definition 4.1). Thus, for DAGPA, we accept the proposal θ' with probability

$$\min \left(1, \exp(U(\theta) - U(\theta')) \frac{q(\theta | \theta')}{q(\theta' | \theta)} \right), \quad (10)$$

where $U(\theta) = \mathcal{L}_{\text{TP-0}}(\theta, \mathcal{D}) + \mathcal{L}_{\text{TP-1}}(\theta, \mathcal{D}) + \mathcal{L}_{\text{TN-0}}(\theta, \mathcal{D}) + \mathcal{L}_{\text{TN-1}}(\theta, \mathcal{D}) + \mathcal{L}_{\text{DAG}}(\theta, s)$ with \mathcal{D} the dataset.

We reproduce the DLP sampling algorithm adopted to DAGPA in Algorithm 2. We note, however, that the combination of PCGrad[29] projected gradients $\Delta \theta^{(\text{PC})}$ and the energy function $U(\theta)$ as the sum of multi-task losses may not form a mathematically well-defined and reversible Markov chain for the sampling, because the gradients are not directly derived from $U(\theta)$ but have been post-hoc modified via gradient projections. Nevertheless, we argue both theoretically and from empirical observations that the gradient modification step with either PCGrad or some alternative multi-task learning method is essential, as it addresses gradient conflicts and navigates the gradient landscape much more efficiently. Furthermore, we empirically found that the MH acceptance step using the energy function $U(\theta)$ obtained via summing the multi-task losses is sufficiently performative and, more importantly, provides the acceptance rate as an important indicator for adjusting the DLP step size β , which is one of the most important hyperparameters in DAGPA. We detail the hyperparameter choice in Appendix D.4. We invite future research to tackle this challenging of integrating multi-task gradients with MH acceptance step in a more principled way.

C.3 Training-time DAG Selection

Finally, DAGPA employs a heuristic score to evaluate the quality of causal graphs obtained from sampled model parameters. This evaluation step is essential because DAGPA adopts DLP [33], and each DLP sampling iteration generates new model parameters and corresponding causal graphs, making it necessary to identify the highest-quality DAGs for final output. Importantly, this score is not a traditional “validation score” since it requires neither a separate validation dataset nor knowledge of the ground-truth causal structure. Instead, it operates solely on the input dataset \mathcal{D} that is already used during optimization and sampling. This approach offers a practical advantage over model-based

Algorithm 2 DLP (DMALA) [33] Sampling Step

Require: Model parameters θ , step size β , gradient $\Delta \theta^{(\text{PC})}$, current energy $U(\theta)$

- 1: // Proposal Step
- 2: **for** $i = 1 : d$ **do** ▷ Can be done in parallel
- 3: **construct** $q_i(\cdot | \theta)$ as in Equation (9)
- 4: **sample** $\theta'_i \sim q_i(\cdot | \theta)$
- 5: // MH Acceptance Step
- 6: **compute** $U(\theta')$
- 7: **compute** $\Delta \theta'^{(\text{PC})}$ via PCGrad (Algorithm 1)
- 8: **compute** $q(\theta' | \theta) = \prod_i q_i(\theta'_i | \theta)$
- 9: **compute** $q(\theta | \theta') = \prod_i q_i(\theta_i | \theta')$
- 10: **set** $\theta \leftarrow \theta'$ with probability in Equation (10)
- 11: **return** new sample θ'

1313 methods like NOTEARS [34] and DAGMA [2], which typically require splitting the dataset to create
 1314 separate validation sets, thereby reducing the amount of data available for model training. Our score
 1315 makes full use of all available data while providing a principled mechanism for DAG selection.

1316 We name this score the ‘‘TPTN Ratio score’’, as it essentially checks the ratio of weighted true positive
 1317 (TP) and true negative (TN) d-separation / CI statements predicted by the model, while treating the
 1318 p-values from the dataset as the soft ground-truth labels. Specifically, given the current sampled
 1319 model parameter θ , we obtain the weighted adjacency matrix via $\mathbf{W} = \sigma(\theta)$, and then threshold it
 1320 to convert it to a binary graph $\hat{\mathbf{A}}$, $\hat{\mathbf{A}}_{x,y} = \mathbb{1}[\mathbf{W}_{x,y} > 0.5], \forall x, y \in [d]$. Additionally, we would like
 1321 to ensure that at every step of sampling, we are evaluating a DAG, as the graph $\hat{\mathbf{A}}$ converted from
 1322 the model parameters θ found by the DLP sampling algorithm may not fully satisfy the acyclicity
 1323 constraint. This requirement can be achieved by pruning $\hat{\mathbf{A}}$ to form an acyclic \mathbf{A} . In DAGPA, we
 1324 proceed by finding a feedback arc set (ARC) from $\hat{\mathbf{A}}$ and remove all the edges in it to construct \mathbf{A} .
 1325 We use the ARC method provided in the igraph package [5].

1326 Then, setting the LogMeanExp temperature α (Equation (8)) to a very small value (e.g. $\alpha = 1e - 5$),
 1327 we use the differentiable d-separation scores (Definition 3.6) on \mathbf{A} to obtain $\tilde{S}_{\mathbf{A}}^{(0)}$ and $\tilde{S}_{\mathbf{A}}^{(1)}$. Since α
 1328 is small but not exactly 0, $\exp(\tilde{S}_{\mathbf{A}}^{(0)})$ and $\exp(\tilde{S}_{\mathbf{A}}^{(1)})$ are close to but not exactly 0’s or 1’s. Thus, we
 1329 threshold again to obtain the binary d-separation statements, $S_{\mathbf{A}}^{(0)}(x, y) = \mathbb{1}[\exp(\tilde{S}_{\mathbf{A}}^{(0)})(x, y) > 0.5]$
 1330 and $S_{\mathbf{A}}^{(1)}(x, y \mid z) = \mathbb{1}[\exp(\tilde{S}_{\mathbf{A}}^{(1)})(x, y \mid z) > 0.5]$. We treat these as the binary d-separation
 1331 statements predicted by the model.

1332 Then, given the p-values $p_{\mathcal{D}}$ from the data, we compute the true positives (TP), true negatives (TN),
 1333 false positives (FP), and false negative (FN) scores as:

$$\begin{aligned} \text{TP}(\mathbf{A}, \mathcal{D}) &= \sum_{x,y \in [d]} S_{\mathbf{A}}^{(0)}(x, y \mid z) p_{\mathcal{D}}(x, y) + \sum_{x,y,z \in [d]} S_{\mathbf{A}}^{(1)}(x, y \mid z) p_{\mathcal{D}}(x, y \mid z) \\ \text{TN}(\mathbf{A}, \mathcal{D}) &= \sum_{x,y \in [d]} (1 - S_{\mathbf{A}}^{(0)}(x, y \mid z))(1 - p_{\mathcal{D}}(x, y)) + \sum_{x,y,z \in [d]} (1 - S_{\mathbf{A}}^{(1)}(x, y \mid z))(1 - p_{\mathcal{D}}(x, y \mid z)) \\ \text{FP}(\mathbf{A}, \mathcal{D}) &= \sum_{x,y \in [d]} S_{\mathbf{A}}^{(0)}(x, y \mid z)(1 - p_{\mathcal{D}}(x, y)) + \sum_{x,y,z \in [d]} S_{\mathbf{A}}^{(1)}(x, y \mid z)(1 - p_{\mathcal{D}}(x, y \mid z)) \\ \text{FN}(\mathbf{A}, \mathcal{D}) &= \sum_{x,y \in [d]} (1 - S_{\mathbf{A}}^{(0)}(x, y \mid z))p_{\mathcal{D}}(x, y) + \sum_{x,y,z \in [d]} (1 - S_{\mathbf{A}}^{(1)}(x, y \mid z))p_{\mathcal{D}}(x, y \mid z). \end{aligned}$$

1334 And then the TPTN Ratio score is computed as

$$\text{TPTN-Ratio}(\mathbf{A}, \mathcal{D}) = \frac{\text{TP}(\mathbf{A}, \mathcal{D}) + \text{TN}(\mathbf{A}, \mathcal{D})}{\text{TP}(\mathbf{A}, \mathcal{D}) + \text{TN}(\mathbf{A}, \mathcal{D}) + \text{FP}(\mathbf{A}, \mathcal{D}) + \text{FN}(\mathbf{A}, \mathcal{D})}, \quad (11)$$

1335 which ranges in $[0, 1]$ and the higher the score, the better the causal graph in matching the low-order
 1336 CI statements found in the data.

1337 C.4 Full algorithm of DAGPA

1338 Combining all techniques together, DAGPA’s full algorithm is given in Algorithm 3.

1339 D Experiment Details

1340 D.1 Code and dataset release

1341 We release our code and data in

1342 <https://anonymous.4open.science/r/Model-Free-Causal-Discovery-Submission-B021/>

1343 D.2 Dataset creation and conversion

1344 We generated synthetic binary dataset following the same way as introduced in the k -PC codebase
 1345 [13]. We simulated DAGs with Erdős–Rényi (ER) model and Scale-Free (SF) of 10 and 50 nodes,

Algorithm 3 DAGPA

Require: Data \mathcal{D} , initial parameter θ_0 , number of steps T , number of best DAGs K , step size β

- 1: **for** $t = 0 : T - 1$ **do**
- 2: $\mathbf{W}_t \leftarrow \sigma(\theta_t)$
- 3: **for** $(x, y) \in [d]^2$ **do** ▷ Can be done in parallel
- 4: **compute** $\tilde{S}_{\mathbf{W}_t}^{(0)}$ and $\tilde{C}_{\mathbf{W}_t}^{(0)}$ as in Definition 3.6
- 5: **for** $(x, y, z) \in [d]^3$ **do** ▷ Can be done in parallel
- 6: **compute** $\tilde{S}_{\mathbf{W}_t}^{(1)}$ and $\tilde{C}_{\mathbf{W}_t}^{(1)}$ as in Definition 3.6
- 7: **compute** $\mathcal{L}_{\text{TP-0}}, \mathcal{L}_{\text{TP-1}}, \mathcal{L}_{\text{TN-0}}, \mathcal{L}_{\text{TN-1}}, \mathcal{L}_{\text{DAG}}$ as in Definition 4.1
- 8: $U(\theta_t) \leftarrow \mathcal{L}_{\text{TP-0}} + \mathcal{L}_{\text{TP-1}} + \mathcal{L}_{\text{TN-0}} + \mathcal{L}_{\text{TN-1}} + \mathcal{L}_{\text{DAG}}$
- 9: **compute** $\Delta\theta_t^{(\text{PC})}$ via PCGrad [29] (Algorithm 1)
- 10: **compute** θ_{t+1} via DLP [33] (Algorithm 2)
- 11: **compute** \mathbf{A}_{t+1} by converting θ_{t+1} to a discrete DAG (Appendix C.3)
- 12: **compute** TPTN-Ratio($\mathbf{A}_{t+1}, \mathcal{D}$)
- 13: **return** K DAGs from $\{\mathbf{A}_t\}_{t=1}^T$ with the Top- K highest TPTN-Ratio($\mathbf{A}_t, \mathcal{D}$) score

1346 with arc ratio of 2 and 4. We then randomly simulated SCM with implementation in pyAgrum [7] and
1347 drew increasing number of samples to create our simulated dataset of size 10, 100, 1000 and 10000.
1348 We also generated continuous data with simulated ER and SF DAG following the same approach
1349 introduced in [2].

1350 For experiments on real-world dataset, we benchmarked our method and baselines over the Sachs
1351 dataset [20] and the LUCAS (LUNG CAncer Simple set) dataset [10]. The Sachs dataset is a widely
1352 recognized benchmark in causal discovery research, consisting of protein signaling pathway data
1353 collected through flow cytometry experiments. It contains measurements of 11 phosphorylated
1354 proteins and phospholipids derived from thousands of individual primary immune system cells,
1355 gathered under various experimental conditions. What makes this dataset particularly valuable
1356 for benchmarking causal learning methods is that the pathways between these proteins are well-
1357 established in scientific literature, providing a reliable ground truth causal graph against which
1358 algorithms can be evaluated. The dataset has been extensively used in numerous studies to assess
1359 the performance of causal discovery algorithms, including recent work that demonstrates how less
1360 restrictive modeling approaches can capture complex causal relationships in this data that traditional
1361 methods assuming additive noise often fail to identify. We obtained and did benchmark on the subset
1362 of Sachs data containing approximately 800 samples with no perturbation. The LUCAS dataset is an
1363 artificially generated benchmark specifically designed to evaluate causal discovery algorithms under
1364 different conditions. It features binary variables in a causal Bayesian network structure and comes
1365 in several variants that present increasing levels of challenge. Among all experiments introduced in
1366 LUCAS, we used LUCAS0 (baseline unmanipulated data) of size 2000.

1367 D.3 Evaluation metrics

1368 The primary metric we adopt in this work is the Conditional Independence Matthews Correlation
1369 Coefficient (CI-MCC), which we describe in detail in Section 5 with a visual illustration in Figure 1.
1370 Additionally, we evaluate our method and baselines using the following standard graph structure-based
1371 metric:

- 1372 • **CPDAG Arrowhead F1:** We compare predicted causal graphs with the ground-truth CPDAG
1373 derived from the ground-truth DAG. Since different methods produce varying graph types, we
1374 standardize all outputs by converting them to CPDAGs. For methods like NOTEARS [34] and
1375 DAGMA [2] that return DAGs, we convert their outputs to CPDAGs using the utility method
1376 from the causal-learn package [36]. For k-PC [13], which returns k-essential graphs containing
1377 circle-marked edges (in addition to directed and undirected edges), we treat circle-marked edges as
1378 undirected edges in the CPDAG conversion. This treatment is appropriate because circle-marked
1379 edges in k-essential graphs denote edges whose directions cannot be determined from low-order
1380 CI statements alone. Once both predicted and ground-truth graphs are converted to CPDAGs, we
1381 compute the arrowhead F1 score exclusively over the directed edges in the CPDAGs.

- 1382 • **CPDAG Skeleton F1:** Following the same standardization process as CPDAG Arrowhead F1, we
1383 convert all predicted causal graphs to CPDAGs. We then further convert all directed edges in both
1384 predicted and ground-truth CPDAGs to undirected edges, creating graph skeletons. The F1 score is
1385 computed on these undirected skeleton graphs.
- 1386 • **CPDAG Structural Hamming Distance (SHD):** After standardizing predicted causal graphs
1387 by converting them to CPDAGs, we compute the structural Hamming distance (SHD) between
1388 predicted and ground-truth CPDAGs. Each unit of SHD corresponds to a single edge difference
1389 between the two CPDAGs, including: missing edges, extra edges, incorrect edge orientations
1390 (directed vs. undirected), or incorrect edge directions.
- 1391 • **DAG F1:** We also evaluate methods that directly output DAGs by comparing them with the
1392 ground-truth DAG. Since constraint-based methods like GES [4], PC [24], and k-PC [13] return
1393 CPDAGs or k-essential graphs rather than DAGs, this metric only applies to score-based methods:
1394 our approach, NOTEARS (linear and nonlinear) [34, 35], and DAGMA [2].

1395 We note that these graph structure-based metrics may pose an inherent disadvantage to our method
1396 compared to CI-MCC. Through empirical observation, we have identified cases where DAGPA
1397 produces DAGs with similarly high CI-MCC scores but vastly different performance on structure-
1398 based metrics. For instance, one sampled DAG may achieve both high CI-MCC and high structure-
1399 based scores, while another DAG with comparable CI-MCC performance may score poorly on
1400 structure-based metrics. This suggests that while DAGPA consistently samples DAGs that align
1401 well with conditional independence patterns (as measured by CI-MCC), this alignment does not
1402 necessarily translate to high performance on traditional graph structure metrics. We provide detailed
1403 analysis of this phenomenon in Appendix E.1.

1404 D.4 DAGPA model details and hyperparameters

1405 The most important and sensitive hyperparameter in DAGPA is the DLP sampling step size β
1406 (Equation (9)). To this end, we first find values for all other hyperparameters through preliminary
1407 experimentations then fix them, and only vary in the step size for the experiments on the synthetic
1408 binary dataset and the real-world datasets.

1409 Some of the other important hyperparameters and their values:

- 1410 • **DLP support logit set \mathbb{D} :** This hyperparameter controls the support logits that the model parameter
1411 θ can take during sampling. We use $\mathbb{D} = [-2.0, 0.0, 2.0]$.
- 1412 • **LogMeanExp temperature α :** This hyperparameter controls the approximation accuracy of the
1413 t-conorm operator (Equation (8)) and thus the accuracy of differentiable d-separation scores. Setting
1414 this value too large will lose approximation accuracy, while setting it too low will induce unstable
1415 and unsmooth gradients. Thus, during training or parameter update cycles, we use a $\alpha_{\text{train}} = 0.01$,
1416 while during evaluation when computing the DAG selection score, we use a $\alpha_{\text{eval}} = 1e - 5$.
- 1417 • **DAGMA’s [2] log-det acyclicity constraint hyperparameter s :** This hyperparameter controls
1418 the valid region of M-matrices in which the log-det acyclicity loss is well-defined. Setting this
1419 value too large will risk model parameters stepping out of this region and causing undefined
1420 gradients, whereas setting this value too large will cause the gradient to have very small norm. In
1421 our experiments, for small graphs (include $n = 10$ synthetic binary dataset and both Sachs [20]
1422 and Lucas [10]) we use $s = 3.0$, while for large graphs ($n = 50$ synthetic binary dataset) we use
1423 $s = 8.0$.

1424 Finally, for the DLP step size β , for each dataset we choose a different range to run hyperpa-
1425 rameter search and choose the best sampled DAGs therein according to the DAG selection score
1426 (Appendix C.3). The specific value range of β is chosen according to the acceptance rate in the
1427 DLP’s Metropolis-Hastings acceptance-rejection step. Specifically, we choose the lowest β value to
1428 be the one that can roughly achieve 0.8 acceptance rate, and the highest β value to be the one that can
1429 roughly achieve 0.2 acceptance rate.

- 1430 • **$n = 10$ graphs in synthetic binary dataset:** $\beta \in \{0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2\}$.
- 1431 • **$n = 50$ graphs in synthetic binary dataset:** $\beta \in \{0.31, 0.32, 0.33, 0.34, 0.35, 0.36, 0.37, 0.38\}$.
- 1432 • **Sachs and Lucas:** $\beta \in \{0.76, 0.78, 0.80, 0.82, 0.84, 0.86, 0.88, 0.90\}$

1433 D.5 Baseline model details and hyperparameters

1434 We tested our method against 5 methods from the table in the following section. For PC and
 1435 kPC, we used chi-square independence test and chose significance level threshold at 0.05, and we
 1436 tested with both first and second order conditional independence tests for kPC. We used the causal-
 1437 learn implementation[36] for PC and GES algorithms. For GES we used the local BIC score. For
 1438 NOTEARS and DAGMA linear mode, we used the l2 loss for non-linear mode, and we followed the
 1439 default dimensions for the MLP layer as used in the authors' original code. Rest of parameters all
 1440 followed default settings.

1441 D.6 Compute resources used

1442 For the baselines, we ran PC, kPC, GES and linear version of NOTEARS, DAGMA on a 64-core
 1443 AMD Epyc 7662 "Rome" processor with 16 CPU cores and 32 GB memory requested. The non-linear
 1444 version of NOTEARS and DAGMA were run on one A30 with same CPU and memory requirement.
 1445 Every experiment is completed in 4 hours.

1446 For DAGPA, we run all experiments on an AMD GPU cluster, equipped with 32GB MI108 and
 1447 64GB MI210 and EPYC 7V13 cpu with 64 cores. Experiments on small graphs ($n = 10$ synthetic
 1448 binary, Sachs, and Lucas) terminate within 1.5 hours, whereas experiments on large graphs ($n = 50$)
 1449 terminates within 12 hours.

1450 E Full Experiment Results

1451 E.1 Analysis of CI-MCC versus graph structure metrics

1452 In this work we primarily showcase the conditional independence Matthews Correlation Coefficient
 1453 (CI-MCC) metric, along with auxiliary graph-structure-based metrics like CPDAG Arrowhead F1,
 1454 CPDAG Skeleton F1, CPDAG SHD, and DAG F1. We notice, however, that the graph-structure-based
 1455 metrics may pose an unfair challenging to DAGPA. In particular, we found that the graph-structure
 1456 metrics may not always align with DAGPA's objective of matching the causal DAG's predicted
 1457 d-separation statements with the low-order CI statements found in the dataset. There are many cases
 1458 where the DAGs returned by DAGPA made few mistakes in aligning the CI statements, yet are still
 1459 scored badly by the graph-structure-based metrics. We provide concrete evidence in this section.

1460 Here, we pick one experiment run of DAGPA
 1461 on one of the synthetic binary dataset gener-
 1462 ated by an ER graph with $n = 10$ nodes with
 1463 in total $N = 10000$ samples. This particular
 1464 run performed 1000 sampling steps, thus giv-
 1465 ing us 1000 sampled DAGs. For each sampled
 1466 DAG, we compute the DAG selection score (Ap-
 1467 pendix C.3) and the CI-MCC metric score, as
 1468 well as the scores of all graph-structure-based
 1469 metrics. To recall, the DAG selection score is
 1470 used by DAGPA to select the best DAGs out
 1471 of all that were sampled during training and
 1472 requires only the input data to compute. It is
 1473 *not* a test metric. Figure 8 shows the scatter
 1474 plots of comparing CI-MCC to graph-structure-
 1475 based metrics, Figure 7 shows the scatter plots
 1476 of comparing the DAG selection score to graph-
 1477 structure-based method, and finally Figure 6 shows the relationship between the DAG selection score
 1478 with CI-MCC.

1479 From Figure 8, we can observe that, although the graph-structure-based metrics have generally
 1480 positive correlation with CI-MCC, there are many samples concentrated in the bottom-right - the
 1481 region where these sampled DAGs achieves high CI-MCC but low graph-structure-based metric
 1482 values. Moreover, especially on regions of high CI-MCC values, the samples vary a lot on their
 1483 graph-structure-based metric values. This implies that there is a misalignment between CI-MCC

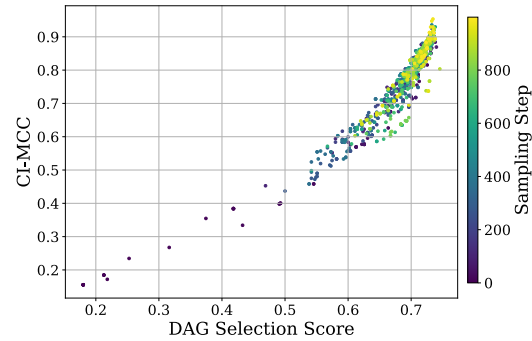


Figure 6: DAG selection score versus CI-MCC. There is a strong positive correlation.

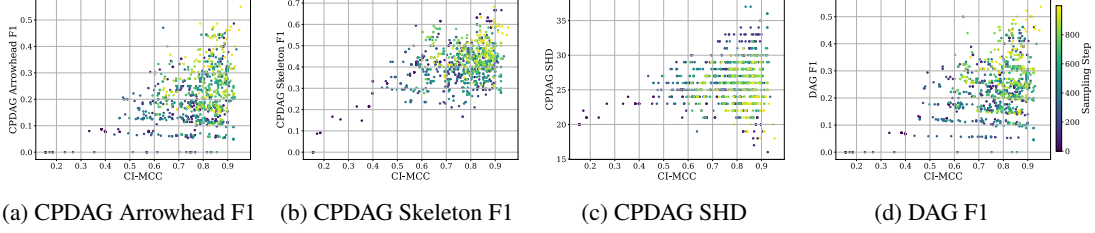


Figure 7: Scatter plots of CI-MCC metric versus standard graph-structure-based metrics of DAGs sampled by an exemplar DAGPA run on a $n = 10$ synthetic binary data. The graph-structure-based metrics are not strongly positively correlated with CI-MCC. Many DAGs have similarly good CI-MCC values, but vary significantly in their graph-structure-based metric values. Thus, the graph-structure-based metrics pose an unfair challenge to DAGPA that focuses on aligning low-order CI statements.

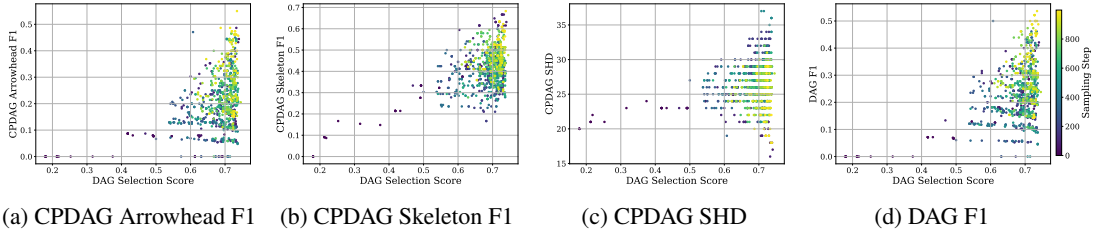


Figure 8: Scatter plots of DAGPA’s DAG selection score (Appendix C.3) versus standard graph-structure-based metrics of DAGs sampled by an exemplar DAGPA run on a $n = 10$ synthetic binary data. Similar to the case of CI-MCC, many DAGs with similarly good DAG selection score may have vastly different values on the graph-structure-based score.

and the graph-structure-based metric, where performing well on matching model’s d-separation with data’s low-order CI statements do not translate to similarity on the graph structure.

Figure 7 reveals a similar story. In this case, the x-axis is the score actually used by DAGPA during training. The similar pattern shows that, even if DAGPA discovers a high-quality DAG with high DAG selection score, which in turn translates to making minimal false positive and false negative mistakes on the low-order CI statements (Appendix C.3), it may still have a drastically different structure than the ground-truth DAG or CPDAG, yielding a very low graph-structure-based metric value such as a low CPDAG Arrowhead F1 value. We leave the problem of proposing an alternative DAG selection score that may yield much more positive correlation with graph-structure-based metric to future research.

As a sanity check, Figure 6 shows the relationship between DAGPA’s DAG selection score and the CI-MCC metric. Here, we can finally observe a strong correlation, demonstrating the validity of our proposed DAG selection score - it is correctly doing what it is designed to do, i.e., checking alignments of low-order CI statements between the model and input data.

Finally, throughout all three figures, we visualize the sampling step numbers in color scale, where the darker colors corresponds to early stages in the sampling, and lighter colors corresponds to later stages. We can observe that, in general, there exists a gradual transition from darker color to lighter color when going from bottom-left to top-right. This pattern suggests that DAGPA is indeed gradually approaching the regions in the weight space that have better and better performance, across all types of metrics.

E.2 Results on synthetic binary data

We present the full suite of results on the synthetic binary dataset in Tables 1 to 4.

Table 1: Experimental results on Synthetic binary ER, $r = 2$ (a) Synthetic binary ER, $r = 2$

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.2319 \pm 0.0351	0.2259 \pm 0.0370	0.2465 \pm 0.0502	0.2380 \pm 0.0305	0.0925 \pm 0.0198	0.0867 \pm 0.0144	0.0850 \pm 0.0264	0.0904 \pm 0.0150
DAGMA (Nonlinear) [2]:	0.5133 \pm 0.1653	0.3544 \pm 0.0682	0.5225 \pm 0.2650	0.4707 \pm 0.1433	0.2668 \pm 0.1312	0.1535 \pm 0.0466	0.1725 \pm 0.0652	0.1990 \pm 0.0713
NOTEARS (Linear) [34]:	0.2626 \pm 0.0517	0.2276 \pm 0.0333	0.2689 \pm 0.0483	0.2586 \pm 0.0363	0.1049 \pm 0.0215	0.0881 \pm 0.0211	0.0915 \pm 0.0273	0.0983 \pm 0.0178
NOTEARS (Nonlinear) [35]:	0.1949 \pm 0.0288	0.1810 \pm 0.0265	0.2023 \pm 0.0276	0.1901 \pm 0.0216	0.0758 \pm 0.0125	0.0727 \pm 0.0112	0.0718 \pm 0.0190	0.0734 \pm 0.0144
GES [4]:	0.2983 \pm 0.0899	0.6102 \pm 0.1428	0.8546 \pm 0.1076	0.7783 \pm 0.0864	-0.1131 \pm 0.0794	0.0178 \pm 0.1155	0.1844 \pm 0.1689	0.0387 \pm 0.0235
PC [24]:	0.2847 \pm 0.0541	0.6227 \pm 0.1281	0.8666 \pm 0.1004	0.7922 \pm 0.0988	-0.1931 \pm 0.0740	0.0081 \pm 0.1172	0.1073 \pm 0.1358	0.0325 \pm 0.0291
kPC (k=1) [13]:	0.3135 \pm 0.0706	0.7585 \pm 0.1540	0.8923 \pm 0.1001	0.8205 \pm 0.0906	-0.1049 \pm 0.1208	0.0512 \pm 0.0925	0.0837 \pm 0.1375	0.0258 \pm 0.0224
kPC (k=2) [13]:	0.2941 \pm 0.0694	0.7100 \pm 0.1484	0.8667 \pm 0.0954	0.8109 \pm 0.0900	-0.1679 \pm 0.0961	0.0590 \pm 0.0910	0.1139 \pm 0.1408	0.0292 \pm 0.0229
DAGPA (Ours)	0.4679 \pm 0.1285	0.6848 \pm 0.0817	0.8223 \pm 0.0970	0.7971 \pm 0.0849	0.0604 \pm 0.0163	0.2156 \pm 0.0555	0.0601 \pm 0.0199	0.3844 \pm 0.0974

(b) Synthetic binary ER, $r = 2$, CPDAG F1 Arrowhead

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.0273 \pm 0.0862	0.0000 \pm 0.0000	0.0211 \pm 0.0666	0.0000 \pm 0.0000	0.0085 \pm 0.0179	0.0214 \pm 0.0348	0.0143 \pm 0.0214	0.0146 \pm 0.0241
DAGMA (Nonlinear) [2]:	0.2143 \pm 0.1470	0.2244 \pm 0.0624	0.1921 \pm 0.1522	0.3070 \pm 0.2069	0.1914 \pm 0.0334	0.1791 \pm 0.0781	0.2194 \pm 0.0972	0.2749 \pm 0.0581
NOTEARS (Linear) [34]:	0.0296 \pm 0.0655	0.0000 \pm 0.0000	0.0000 \pm 0.0000	0.0000 \pm 0.0000	0.0696 \pm 0.0705	0.0273 \pm 0.0355	0.0239 \pm 0.0437	0.0231 \pm 0.0250
NOTEARS (Nonlinear) [35]:	0.0424 \pm 0.0912	0.0000 \pm 0.0000	0.0133 \pm 0.0422	0.0000 \pm 0.0000	0.0214 \pm 0.0212	0.0387 \pm 0.0407	0.0519 \pm 0.0455	0.0102 \pm 0.0141
GES [4]:	0.0915 \pm 0.1322	0.3860 \pm 0.1894	0.3900 \pm 0.1013	0.4866 \pm 0.1802	0.2006 \pm 0.0870	0.5226 \pm 0.0646	0.6921 \pm 0.0943	0.7022 \pm 0.0897
PC [24]:	0.1025 \pm 0.1618	0.3607 \pm 0.2217	0.5673 \pm 0.1444	0.7620 \pm 0.1541	0.1242 \pm 0.0499	0.4024 \pm 0.0553	0.5938 \pm 0.0590	0.7325 \pm 0.0687
kPC (k=1) [13]:	0.0824 \pm 0.1370	0.1991 \pm 0.1077	0.2435 \pm 0.1380	0.3359 \pm 0.1664	0.1039 \pm 0.0453	0.2255 \pm 0.0575	0.2595 \pm 0.0419	0.2797 \pm 0.0435
kPC (k=2) [13]:	0.0942 \pm 0.1650	0.2298 \pm 0.1473	0.4042 \pm 0.1623	0.5329 \pm 0.2219	0.0923 \pm 0.0551	0.2788 \pm 0.0520	0.4137 \pm 0.0559	0.5561 \pm 0.0658
DAGPA (Ours)	0.1057 \pm 0.0766	0.1882 \pm 0.0786	0.2041 \pm 0.0912	0.2684 \pm 0.1081	0.0005 \pm 0.0034	0.0347 \pm 0.0267	0.0015 \pm 0.0071	0.0541 \pm 0.0156

(c) Synthetic binary ER, $r = 2$, CPDAG F1 Skeleton

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.2921 \pm 0.1183	0.2629 \pm 0.1052	0.3034 \pm 0.1501	0.2682 \pm 0.1128	0.2953 \pm 0.0285	0.2417 \pm 0.0405	0.2431 \pm 0.0419	0.2428 \pm 0.0747
DAGMA (Nonlinear) [2]:	0.5418 \pm 0.0967	0.5362 \pm 0.0742	0.6114 \pm 0.1816	0.6341 \pm 0.1375	0.3667 \pm 0.0307	0.5542 \pm 0.0338	0.5701 \pm 0.0356	0.5989 \pm 0.0491
NOTEARS (Linear) [34]:	0.3535 \pm 0.1174	0.2791 \pm 0.0958	0.3692 \pm 0.1524	0.3252 \pm 0.1058	0.3574 \pm 0.0378	0.3135 \pm 0.0544	0.3057 \pm 0.0398	0.3036 \pm 0.0741
NOTEARS (Nonlinear) [35]:	0.1415 \pm 0.1327	0.0807 \pm 0.1433	0.1991 \pm 0.1278	0.0694 \pm 0.1124	0.1434 \pm 0.0518	0.1347 \pm 0.0482	0.1319 \pm 0.0812	0.1047 \pm 0.0501
GES [4]:	0.3845 \pm 0.1305	0.7013 \pm 0.0945	0.8038 \pm 0.0643	0.8271 \pm 0.0500	0.4303 \pm 0.0401	0.7213 \pm 0.0434	0.8588 \pm 0.0278	0.8627 \pm 0.0343
PC [24]:	0.4202 \pm 0.1054	0.8121 \pm 0.0899	0.9314 \pm 0.0420	0.9717 \pm 0.0322	0.4011 \pm 0.0272	0.7401 \pm 0.0414	0.8871 \pm 0.0184	0.9626 \pm 0.0161
kPC (k=1) [13]:	0.4270 \pm 0.0956	0.7831 \pm 0.0745	0.7793 \pm 0.0442	0.8004 \pm 0.0536	0.4278 \pm 0.0243	0.7488 \pm 0.0361	0.7860 \pm 0.0273	0.6889 \pm 0.0602
kPC (k=2) [13]:	0.4202 \pm 0.1054	0.8069 \pm 0.0806	0.8968 \pm 0.0332	0.8950 \pm 0.0670	0.4024 \pm 0.0276	0.7496 \pm 0.0411	0.8827 \pm 0.0154	0.9139 \pm 0.0292
DAGPA (Ours)	0.4062 \pm 0.0985	0.4273 \pm 0.0813	0.4442 \pm 0.0945	0.4452 \pm 0.0867	0.0062 \pm 0.0146	0.0826 \pm 0.0347	0.0079 \pm 0.0142	0.1043 \pm 0.0300

(d) Synthetic binary ER, $r = 2$, CPDAG SHD

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	17.8000 \pm 1.8135	17.5000 \pm 1.0801	17.1000 \pm 2.0248	16.1000 \pm 1.5239	97.5000 \pm 1.7159	96.2000 \pm 2.6162	97.7000 \pm 1.8886	96.7000 \pm 2.4060
DAGMA (Nonlinear) [2]:	17.4000 \pm 2.2706	15.6000 \pm 0.9661	16.4000 \pm 2.6750	13.4000 \pm 2.4585	152.9000 \pm 15.3511	88.1000 \pm 4.9989	85.4000 \pm 6.2752	81.7000 \pm 3.4010
NOTEARS (Linear) [34]:	17.7000 \pm 1.5670	17.5000 \pm 1.0801	17.3000 \pm 1.8288	16.1000 \pm 1.4491	96.6000 \pm 3.6878	95.8000 \pm 2.8983	96.6000 \pm 2.9136	95.9000 \pm 2.2828
NOTEARS (Nonlinear) [35]:	18.1000 \pm 2.0248	18.1000 \pm 1.2867	18.3000 \pm 1.7029	17.2000 \pm 1.6193	99.8000 \pm 1.1353	98.0000 \pm 2.3570	97.9000 \pm 2.5144	98.6000 \pm 1.5055
GES [4]:	17.5000 \pm 2.0683	13.7000 \pm 3.9172	16.1000 \pm 2.8848	13.9000 \pm 5.8963	93.1000 \pm 5.7436	61.4000 \pm 6.2752	43.2000 \pm 11.2921	48.8000 \pm 13.9507
PC [24]:	17.5000 \pm 2.5495	12.8000 \pm 4.3153	9.6000 \pm 1.8974	4.9000 \pm 2.8848	95.0000 \pm 3.3333	70.1000 \pm 4.7011	49.1000 \pm 6.3675	31.0000 \pm 8.8066
kPC (k=1) [13]:	17.8000 \pm 2.1499	14.4000 \pm 2.8363	19.5000 \pm 4.2230	16.9000 \pm 4.1486	94.9000 \pm 2.6854	77.4000 \pm 5.3166	88.0000 \pm 8.3999	134.8000 \pm 28.9858
kPC (k=2) [13]:	17.4000 \pm 2.6331	12.5000 \pm 3.2059	12.0000 \pm 2.3094	10.9000 \pm 4.7947	94.5000 \pm 3.6286	69.2000 \pm 3.9101	55.5000 \pm 5.4416	50.3000 \pm 11.3534
DAGPA (Ours)	21.4800 \pm 2.6972	23.8200 \pm 2.2740	25.5600 \pm 3.3022	24.1400 \pm 2.8357	103.6200 \pm 10.4664	180.8400 \pm 8.3626	106.2000 \pm 15.9796	220.7800 \pm 21.7596

(e) Synthetic binary ER, $r = 2$, DAG F1

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.1706 \pm 0.1091	0.1225 \pm 0.0616	0.2137 \pm 0.1476	0.1677 \pm 0.1313	0.2020 \pm 0.0328	0.1682 \pm 0.0327	0.1766 \pm 0.0490	0.1791 \pm 0.0485
DAGMA (Nonlinear) [2]:	0.3243 \pm 0.1575	0.2734 \pm 0.0819	0.3190 \pm 0.1602	0.4432 \pm 0.1806	0.2243 \pm 0.0269	0.3087 \pm 0.0649	0.3452 \pm 0.0670	0.3829 \pm 0.0546
NOTEARS (Linear) [34]:	0.2076 \pm 0.1263	0.1192 \pm 0.0676	0.2304 \pm 0.1460	0.1905 \pm 0.1282	0.2304 \pm 0.0583	0.1878 \pm 0.0441	0.2027 \pm 0.0489	0.2091 \pm 0.0354
NOTEARS (Nonlinear) [35]:	0.0688 \pm 0.0805	0.0179 \pm 0.0378	0.0564 \pm 0.0778	0.0327 \pm 0.0743	0.0583 \pm 0.0178	0.0665 \pm 0.0391	0.0725 \pm 0.0426	0.0507 \pm 0.0273
DAGPA (Ours)	0.1557 \pm 0.1036	0.2333 \pm 0.1026	0.2176 \pm 0.0971	0.2603 \pm 0.1145	0.0088 \pm 0.0117	0.0372 \pm 0.0230	0.0091 \pm 0.0143	0.0559 \pm 0.0163

Table 2: Experimental results on Synthetic binary ER, $r = 4$ (a) Synthetic binary ER, $r = 4$, CI-MC

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.2398 \pm 0.0631	0.2085 \pm 0.0347	0.2120 \pm 0.0400	0.2226 \pm 0.0584	0.0428 \pm 0.0064	0.0398 \pm 0.0039	0.0398 \pm 0.0039	0.0395 \pm 0.0051
DAGMA (Nonlinear) [2]:	<u>0.5724</u> \pm 0.1213	0.4450 \pm 0.1860	0.5007 \pm 0.1535	0.4624 \pm 0.1775	0.2658 \pm 0.0818	<u>0.0591</u> \pm 0.0169	<u>0.0735</u> \pm 0.0181	<u>0.0623</u> \pm 0.0132
NOTEARS (Linear) [34]:	0.2798 \pm 0.0848	0.2197 \pm 0.0391	0.2344 \pm 0.0665	0.2294 \pm 0.0273	0.0506 \pm 0.0081	0.0419 \pm 0.0049	0.0434 \pm 0.0046	0.0423 \pm 0.0057
NOTEARS (Nonlinear) [35]:	0.1882 \pm 0.0371	0.1804 \pm 0.0290	0.2091 \pm 0.0469	0.1894 \pm 0.0278	0.0369 \pm 0.0045	0.0350 \pm 0.0026	0.0342 \pm 0.0040	0.0361 \pm 0.0063
GES [4]:	0.3341 \pm 0.1212	0.5932 \pm 0.1362	0.9135 \pm 0.1097	0.9251 \pm 0.0568	-0.3803 \pm 0.1331	-0.1514 \pm 0.0513	-0.0267 \pm 0.0469	0.0000 \pm 0.0000
PC [24]:	0.3282 \pm 0.0842	0.7071 \pm 0.1704	0.8897 \pm 0.1385	0.9317 \pm 0.0444	-0.3514 \pm 0.0755	-0.0181 \pm 0.0608	-0.0049 \pm 0.0138	-0.0100 \pm 0.0155
kPC (k=1) [13]:	0.4053 \pm 0.1859	0.8818 \pm 0.1073	<u>0.9469</u> \pm 0.0775	0.9372 \pm 0.0443	-0.1670 \pm 0.0624	-0.0027 \pm 0.0514	0.0053 \pm 0.0147	0.0000 \pm 0.0000
kPC (k=2) [13]:	0.3307 \pm 0.0878	<u>0.8546</u> \pm 0.1099	0.9469 \pm 0.0775	<u>0.9378</u> \pm 0.0437	-0.2243 \pm 0.0834	-0.0036 \pm 0.0517	0.0054 \pm 0.0153	0.0000 \pm 0.0000
DAGPA (Ours)	0.6214 \pm 0.1562	0.8299 \pm 0.1328	0.9493 \pm 0.0537	0.9484 \pm 0.0357	<u>0.2341</u> \pm 0.0501	0.3292 \pm 0.0720	0.2669 \pm 0.1632	0.7209 \pm 0.1333

(b) Synthetic binary ER, $r = 4$, CPDAG F1 Arrowhead

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.0160 \pm 0.0506	0.0000 \pm 0.0000	0.0333 \pm 0.1054	0.0000 \pm 0.0000	0.0041 \pm 0.0087	0.0051 \pm 0.0086	0.0050 \pm 0.0108	0.0040 \pm 0.0085
DAGMA (Nonlinear) [2]:	0.1844 \pm 0.1478	0.1668 \pm 0.1707	0.0857 \pm 0.1173	0.1994 \pm 0.1346	0.1193 \pm 0.0338	0.0643 \pm 0.0368	0.1158 \pm 0.0564	0.0976 \pm 0.0409
NOTEARS (Linear) [34]:	0.0490 \pm 0.0871	0.0000 \pm 0.0000	0.0333 \pm 0.1054	0.0000 \pm 0.0000	0.0190 \pm 0.0232	0.0061 \pm 0.0109	0.0130 \pm 0.0182	0.0020 \pm 0.0063
NOTEARS (Nonlinear) [35]:	0.0000 \pm 0.0000	0.0000 \pm 0.0000	0.0847 \pm 0.1208	0.0314 \pm 0.0529	0.0109 \pm 0.0125	0.0010 \pm 0.0032	0.0101 \pm 0.0157	0.0070 \pm 0.0125
GES [4]:	0.0685 \pm 0.0793	<u>0.2782</u> \pm 0.1766	<u>0.3504</u> \pm 0.2241	0.4993 \pm 0.1762	0.0684 \pm 0.0327	0.2896 \pm 0.0465	0.5302 \pm 0.0599	<u>0.1253</u> \pm 0.2657
PC [24]:	0.1049 \pm 0.0729	0.3096 \pm 0.1615	0.4223 \pm 0.1233	<u>0.4759</u> \pm 0.1242	<u>0.0819</u> \pm 0.0301	<u>0.2572</u> \pm 0.0454	<u>0.3632</u> \pm 0.0439	0.4153 \pm 0.0472
kPC (k=1) [13]:	0.1014 \pm 0.0858	0.1965 \pm 0.1176	0.2402 \pm 0.1230	0.2588 \pm 0.1050	0.0377 \pm 0.0116	0.0264 \pm 0.0224	0.0262 \pm 0.0136	0.0200 \pm 0.0115
kPC (k=2) [13]:	0.1034 \pm 0.0874	0.1984 \pm 0.1402	0.3077 \pm 0.1590	0.3455 \pm 0.1079	0.0385 \pm 0.0132	0.0417 \pm 0.0287	0.0532 \pm 0.0236	0.0684 \pm 0.0296
DAGPA (Ours)	<u>0.1392</u> \pm 0.0984	0.1788 \pm 0.0976	0.2733 \pm 0.0832	0.2589 \pm 0.1219	0.0424 \pm 0.0204	0.0572 \pm 0.0228	0.0392 \pm 0.0232	0.0648 \pm 0.0131

(c) Synthetic binary ER, $r = 4$, CPDAG F1 Skeleton

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.2706 \pm 0.1085	0.2260 \pm 0.0935	0.2053 \pm 0.0892	0.2292 \pm 0.0902	0.0982 \pm 0.0344	0.0732 \pm 0.0204	0.0952 \pm 0.0135	0.0731 \pm 0.0159
DAGMA (Nonlinear) [2]:	0.5460 \pm 0.0904	0.5399 \pm 0.0802	0.5137 \pm 0.1327	0.5488 \pm 0.1238	<u>0.2296</u> \pm 0.0323	0.1997 \pm 0.0491	0.2653 \pm 0.0517	0.2288 \pm 0.0490
NOTEARS (Linear) [34]:	0.3598 \pm 0.1345	0.2533 \pm 0.0884	0.2489 \pm 0.1027	0.2722 \pm 0.0854	0.1388 \pm 0.0521	0.0915 \pm 0.0266	0.1193 \pm 0.0138	0.0960 \pm 0.0197
NOTEARS (Nonlinear) [35]:	0.1435 \pm 0.1318	0.0848 \pm 0.0995	0.1779 \pm 0.1279	0.1492 \pm 0.1253	0.0519 \pm 0.0353	0.0188 \pm 0.0109	0.0408 \pm 0.0281	0.0352 \pm 0.0166
GES [4]:	0.3903 \pm 0.0991	0.6616 \pm 0.0774	<u>0.8165</u> \pm 0.0861	<u>0.7991</u> \pm 0.0475	0.1820 \pm 0.0589	0.4372 \pm 0.0445	0.7155 \pm 0.0350	0.1574 \pm 0.3320
PC [24]:	0.4194 \pm 0.0577	<u>0.7178</u> \pm 0.1084	0.8209 \pm 0.0830	0.8080 \pm 0.0550	0.2193 \pm 0.0340	0.5469 \pm 0.0404	0.7493 \pm 0.0295	0.7816 \pm 0.0204
kPC (k=1) [13]:	<u>0.4421</u> \pm 0.0710	0.7346 \pm 0.1041	0.7659 \pm 0.0614	0.7724 \pm 0.0304	0.2317 \pm 0.0337	0.5565 \pm 0.0406	0.6894 \pm 0.0297	0.6425 \pm 0.0286
kPC (k=2) [13]:	0.4194 \pm 0.0577	0.7178 \pm 0.1084	0.8137 \pm 0.0830	0.7958 \pm 0.0502	0.2198 \pm 0.0333	<u>0.5530</u> \pm 0.0403	<u>0.7416</u> \pm 0.0310	<u>0.7458</u> \pm 0.0208
DAGPA (Ours)	0.4217 \pm 0.1182	0.4490 \pm 0.0858	0.5309 \pm 0.0943	0.5036 \pm 0.0678	0.1057 \pm 0.0277	0.1254 \pm 0.0348	0.0837 \pm 0.0329	0.1449 \pm 0.0270

(d) Synthetic binary ER, $r = 4$, CPDAG SHD

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	21.1000 \pm 2.5144	20.4000 \pm 3.9497	22.0000 \pm 3.3665	21.2000 \pm 3.3267	197.7000 \pm 2.4518	197.7000 \pm 1.6364	196.9000 \pm 2.1318	197.9000 \pm 1.2867
DAGMA (Nonlinear) [2]:	21.1000 \pm 2.9231	18.3000 \pm 3.8887	21.0000 \pm 3.8006	19.0000 \pm 3.2660	228.8000 \pm 6.9889	191.7000 \pm 5.8128	185.7000 \pm 5.3552	188.4000 \pm 4.0879
NOTEARS (Linear) [34]:	19.9000 \pm 3.2128	20.4000 \pm 3.9777	21.6000 \pm 3.7476	21.1000 \pm 3.3483	198.2000 \pm 3.0111	197.6000 \pm 2.1187	196.1000 \pm 1.5239	197.9000 \pm 1.2867
NOTEARS (Nonlinear) [35]:	21.9000 \pm 2.5781	21.3000 \pm 4.1379	22.1000 \pm 3.4140	21.6000 \pm 2.8752	198.5000 \pm 1.7795	199.1000 \pm 0.8756	198.6000 \pm 1.8974	198.9000 \pm 1.1972
GES [4]:	21.2000 \pm 2.6998	18.3000 \pm 5.0122	19.6000 \pm 6.3281	18.4000 \pm 7.2602	196.3000 \pm 4.3218	164.5000 \pm 7.7782	130.2000 \pm 12.6474	24.6000 \pm 52.7977
PC [24]:	20.7000 \pm 2.7909	17.9000 \pm 5.0870	17.3000 \pm 4.9227	17.4000 \pm 4.7656	205.0000 \pm 5.8119	177.3000 \pm 9.0437	164.4000 \pm 8.8343	171.1000 \pm 14.3717
kPC (k=1) [13]:	21.1000 \pm 2.6854	17.7000 \pm 5.1865	22.3000 \pm 4.4234	22.4000 \pm 5.1683	205.3000 \pm 4.2701	184.5000 \pm 7.7782	211.6000 \pm 9.7091	282.7000 \pm 26.8620
kPC (k=2) [13]:	20.5000 \pm 2.8382	17.6000 \pm 5.4610	18.0000 \pm 5.9815	18.3000 \pm 3.4657	204.4000 \pm 4.4771	177.3000 \pm 8.1384	173.2000 \pm 8.4564	198.9000 \pm 14.1142
DAGPA (Ours)	26.6400 \pm 3.6854	27.2800 \pm 4.2859	27.5600 \pm 3.1826	28.2400 \pm 3.6397	258.5600 \pm 4.5273	268.3000 \pm 10.5313	254.2000 \pm 18.9737	307.2400 \pm 24.4361

(e) Synthetic binary ER, $r = 4$, DAG F1

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.1661 \pm 0.1206	0.1406 \pm 0.1301	0.1428 \pm 0.0775	0.1365 \pm 0.1200	0.0671 \pm 0.0271	0.0569 \pm 0.0181	0.0618 \pm 0.0236	0.0452 \pm 0.0191
DAGMA (Nonlinear) [2]:	0.2813 \pm 0.1782	0.3242 \pm 0.1300	0.2824 \pm 0.1015	0.3234 \pm 0.1329	0.1476 \pm 0.0287	0.1280 \pm 0.0440	0.1739 \pm 0.0548	0.1441 \pm 0.0466
NOTEARS (Linear) [34]:	0.2563 \pm 0.0874	0.1505 \pm 0.1279	0.1517 \pm 0.0926	0.1492 \pm 0.1050	0.0903 \pm 0.0424	0.0668 \pm 0.0261	0.0761 \pm 0.0162	0.0512 \pm 0.0224
NOTEARS (Nonlinear) [35]:	0.0772 \pm 0.0767	0.0238 \pm 0.0385	0.1082 \pm 0.1006	0.0730 \pm 0.0905	0.0279 \pm 0.0208	0.0079 \pm 0.0078	0.0195 \pm 0.0156	0.0196 \pm 0.0103
DAGPA (Ours)	0.2017 \pm 0.1142	0.2056 \pm 0.0526	0.2874 \pm 0.0891	0.3427 \pm 0.1043	0.0483 \pm 0.0176	0.0608 \pm 0.0245	0.0433 \pm 0.0207	0.0681 \pm 0.0153

Table 3: Experimental results on Synthetic binary SF, $r = 2$ (a) Synthetic binary SF, $r = 2$, CI-MC

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.2279 ± 0.0372	0.2247 ± 0.0472	0.2256 ± 0.0285	0.2167 ± 0.0328	0.1413 ± 0.0206	0.1483 ± 0.0215	0.1475 ± 0.0186	0.1560 ± 0.0178
DAGMA (Nonlinear) [2]:	0.4979 ± 0.1539	0.3630 ± 0.0893	0.4339 ± 0.1296	0.4195 ± 0.1210	0.1170 ± 0.0571	0.2208 ± 0.0521	0.2464 ± 0.0610	0.2867 ± 0.0398
NOTEARS (Linear) [34]:	0.2716 ± 0.0877	0.2422 ± 0.0590	0.2548 ± 0.0466	0.2556 ± 0.0563	0.1433 ± 0.0233	0.1551 ± 0.0168	0.1614 ± 0.0249	0.1659 ± 0.0188
NOTEARS (Nonlinear) [35]:	0.2018 ± 0.0362	0.1919 ± 0.0342	0.1960 ± 0.0298	0.1847 ± 0.0270	0.1263 ± 0.0190	0.1286 ± 0.0176	0.1263 ± 0.0161	0.1341 ± 0.0149
GES [4]:	0.3288 ± 0.1263	0.6257 ± 0.1922	0.7965 ± 0.1152	0.8394 ± 0.1061	0.0031 ± 0.0431	0.1955 ± 0.0926	0.2143 ± 0.1380	0.1430 ± 0.1201
PC [24]:	0.3359 ± 0.1207	0.7139 ± 0.1368	0.7956 ± 0.1502	0.8635 ± 0.0748	-0.0206 ± 0.0445	0.1634 ± 0.0986	0.0462 ± 0.0724	0.0537 ± 0.0733
kPC (k=1) [13]:	0.3899 ± 0.1479	0.8032 ± 0.1781	0.8170 ± 0.1356	0.8687 ± 0.0740	-0.0009 ± 0.0737	0.1036 ± 0.1189	0.0579 ± 0.0835	0.0392 ± 0.0693
kPC (k=2) [13]:	0.3435 ± 0.1270	0.7805 ± 0.1662	0.8199 ± 0.1342	0.8681 ± 0.0732	-0.0261 ± 0.0723	0.1487 ± 0.1267	0.0495 ± 0.0990	0.0356 ± 0.0518
DAGPA (Ours)	0.5491 ± 0.1461	0.7731 ± 0.1272	0.8684 ± 0.1128	0.8659 ± 0.0624	0.0728 ± 0.0278	0.0906 ± 0.0262	0.1057 ± 0.0361	0.1938 ± 0.0784

(b) Synthetic binary SF, $r = 2$, CPDAG F1 Arrowhead

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.0000 ± 0.0000	0.0125 ± 0.0395	0.0000 ± 0.0000	0.0426 ± 0.0904	0.0087 ± 0.0183	0.0000 ± 0.0000	0.0044 ± 0.0139	0.0084 ± 0.0178
DAGMA (Nonlinear) [2]:	0.1950 ± 0.1364	0.2072 ± 0.1209	0.2012 ± 0.1235	0.1725 ± 0.1522	0.1465 ± 0.0501	0.1891 ± 0.0760	0.2219 ± 0.0531	0.2210 ± 0.0975
NOTEARS (Linear) [34]:	0.0495 ± 0.1267	0.0211 ± 0.0666	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0273 ± 0.0312	0.0148 ± 0.0248	0.0132 ± 0.0212	0.0149 ± 0.0202
NOTEARS (Nonlinear) [35]:	0.0451 ± 0.0832	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0190 ± 0.0602	0.0284 ± 0.0367	0.0210 ± 0.0262	0.0106 ± 0.0150	0.0167 ± 0.0214
GES [4]:	0.0686 ± 0.0922	0.3097 ± 0.1616	0.5025 ± 0.1347	0.5590 ± 0.2118	0.0801 ± 0.0622	0.4695 ± 0.0836	0.7163 ± 0.0728	0.7741 ± 0.0545
PC [24]:	0.1330 ± 0.1174	0.3398 ± 0.0895	0.4687 ± 0.1326	0.4592 ± 0.1445	0.1093 ± 0.0673	0.3793 ± 0.0854	0.4883 ± 0.0582	0.5647 ± 0.1108
kPC (k=1) [13]:	0.1301 ± 0.0959	0.1593 ± 0.1036	0.2905 ± 0.1309	0.2411 ± 0.1627	0.0757 ± 0.0320	0.1542 ± 0.0634	0.2068 ± 0.0310	0.2871 ± 0.0657
kPC (k=2) [13]:	0.1131 ± 0.0897	0.1612 ± 0.1402	0.3378 ± 0.1372	0.2496 ± 0.1216	0.0729 ± 0.0341	0.1670 ± 0.0556	0.2671 ± 0.0434	0.4013 ± 0.0787
DAGPA (Ours)	0.1363 ± 0.0942	0.1919 ± 0.0815	0.2266 ± 0.1236	0.2374 ± 0.1035	0.0037 ± 0.0076	0.0186 ± 0.0185	0.0078 ± 0.0100	0.0387 ± 0.0178

(c) Synthetic binary SF, $r = 2$, CPDAG F1 Skeleton

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.2159 ± 0.1017	0.1933 ± 0.0984	0.2072 ± 0.1170	0.2296 ± 0.1284	0.2210 ± 0.0413	0.2187 ± 0.0249	0.2148 ± 0.0467	0.1912 ± 0.0305
DAGMA (Nonlinear) [2]:	0.5808 ± 0.0988	0.4768 ± 0.1131	0.5342 ± 0.1081	0.5834 ± 0.1504	0.3076 ± 0.0528	0.5308 ± 0.0508	0.5412 ± 0.0557	0.5433 ± 0.0681
NOTEARS (Linear) [34]:	0.2836 ± 0.1490	0.2358 ± 0.1330	0.2683 ± 0.1199	0.3141 ± 0.1574	0.2857 ± 0.0257	0.2748 ± 0.0333	0.2745 ± 0.0516	0.2630 ± 0.0485
NOTEARS (Nonlinear) [35]:	0.1153 ± 0.1130	0.0641 ± 0.0741	0.0815 ± 0.0977	0.0776 ± 0.1300	0.1159 ± 0.0571	0.1063 ± 0.0485	0.0860 ± 0.0648	0.0817 ± 0.0381
GES [4]:	0.3601 ± 0.1194	0.6713 ± 0.0849	0.8113 ± 0.0427	0.8196 ± 0.0875	0.2684 ± 0.1460	0.6875 ± 0.0449	0.8637 ± 0.0173	0.8839 ± 0.0322
PC [24]:	0.4474 ± 0.0788	0.7446 ± 0.0904	0.8168 ± 0.0510	0.8004 ± 0.0827	0.2942 ± 0.1584	0.7039 ± 0.0399	0.8277 ± 0.0306	0.8505 ± 0.0269
kPC (k=1) [13]:	0.4685 ± 0.0839	0.7454 ± 0.0918	0.7774 ± 0.0494	0.7737 ± 0.0593	0.3689 ± 0.0323	0.6872 ± 0.0304	0.7782 ± 0.0238	0.7501 ± 0.0268
kPC (k=2) [13]:	0.4474 ± 0.0788	0.7404 ± 0.0882	0.8120 ± 0.0508	0.7894 ± 0.0759	0.3612 ± 0.0344	0.7025 ± 0.0374	0.8231 ± 0.0281	0.8361 ± 0.0281
DAGPA (Ours)	0.4003 ± 0.0897	0.4746 ± 0.0788	0.4839 ± 0.1013	0.4846 ± 0.0975	0.0282 ± 0.0194	0.0389 ± 0.0183	0.0357 ± 0.0277	0.0851 ± 0.0264

(d) Synthetic binary SF, $r = 2$, CPDAG SHD

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	19.1000 ± 1.1972	19.2000 ± 0.6325	19.2000 ± 1.0328	18.7000 ± 1.4181	97.2000 ± 1.3166	96.5000 ± 1.2693	95.9000 ± 2.2336	96.7000 ± 1.1595
DAGMA (Nonlinear) [2]:	18.2000 ± 2.0976	17.2000 ± 1.1353	17.1000 ± 1.9692	16.9000 ± 2.1833	147.7000 ± 15.9865	86.1000 ± 4.8408	83.6000 ± 5.1251	85.2000 ± 6.9889
NOTEARS (Linear) [34]:	18.6000 ± 1.7764	19.2000 ± 0.6325	18.9000 ± 0.9944	18.7000 ± 0.8233	98.9000 ± 1.6633	95.2000 ± 1.9889	94.7000 ± 2.9458	95.9000 ± 1.6633
NOTEARS (Nonlinear) [35]:	19.3000 ± 0.8233	19.8000 ± 0.4216	19.8000 ± 0.4216	19.6000 ± 0.6992	98.8000 ± 2.1499	98.6000 ± 2.4129	99.0000 ± 0.8165	98.5000 ± 1.4337
GES [4]:	18.8000 ± 1.6193	15.9000 ± 2.7264	13.8000 ± 3.2249	14.0000 ± 6.0736	79.4000 ± 41.9952	66.3000 ± 7.2885	40.5000 ± 8.3832	36.7000 ± 8.9821
PC [24]:	18.3000 ± 1.9465	15.7000 ± 1.3375	14.2000 ± 3.1552	15.6000 ± 3.4383	84.7000 ± 44.7314	82.8000 ± 9.8070	71.8000 ± 10.0421	65.1000 ± 14.4795
kPC (k=1) [13]:	18.0000 ± 1.8257	16.6000 ± 1.7127	17.7000 ± 5.3759	20.1000 ± 3.0714	120.0000 ± 26.6041	88.6000 ± 9.2999	90.5000 ± 7.7496	106.1000 ± 10.4823
kPC (k=2) [13]:	18.1000 ± 1.7288	16.1000 ± 1.5951	15.3000 ± 4.2701	17.5000 ± 3.3082	118.0000 ± 26.5330	83.1000 ± 8.5434	75.7000 ± 7.2732	74.3000 ± 12.7632
DAGPA (Ours)	24.7800 ± 2.4436	25.7400 ± 2.4562	25.6400 ± 3.6576	25.8600 ± 3.3382	119.7500 ± 5.5366	124.3200 ± 6.9765	130.7000 ± 4.4366	192.3400 ± 31.3106

(e) Synthetic binary SF, $r = 2$, DAG F1

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.1383 ± 0.0889	0.0978 ± 0.0642	0.1394 ± 0.0736	0.1697 ± 0.1112	0.1275 ± 0.0364	0.1529 ± 0.0301	0.1451 ± 0.0566	0.1372 ± 0.0341
DAGMA (Nonlinear) [2]:	0.3603 ± 0.1203	0.2724 ± 0.0810	0.3587 ± 0.1040	0.3766 ± 0.1507	0.1845 ± 0.0362	0.3204 ± 0.0741	0.3446 ± 0.0670	0.3594 ± 0.0663
NOTEARS (Linear) [34]:	0.2089 ± 0.1474	0.1266 ± 0.1081	0.1872 ± 0.0802	0.2016 ± 0.1147	0.1648 ± 0.0436	0.1801 ± 0.0434	0.1849 ± 0.0516	0.1733 ± 0.0365
NOTEARS (Nonlinear) [35]:	0.0788 ± 0.1022	0.0281 ± 0.0453	0.0451 ± 0.0628	0.0265 ± 0.0605	0.0402 ± 0.0375	0.0450 ± 0.0233	0.0281 ± 0.0306	0.0438 ± 0.0268
DAGPA (Ours)	0.1689 ± 0.0987	0.2409 ± 0.0858	0.2662 ± 0.1244	0.2682 ± 0.0969	0.0061 ± 0.0080	0.0205 ± 0.0187	0.0163 ± 0.0195	0.0410 ± 0.0116

Table 4: Experimental results on Synthetic binary SF, $r = 4$ (a) Synthetic binary SF, $r = 4$, CI-MC

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.2029 ± 0.0485	0.1665 ± 0.0130	0.1764 ± 0.0243	0.1886 ± 0.0233	0.0527 ± 0.0119	0.0516 ± 0.0093	0.0454 ± 0.0075	0.0543 ± 0.0063
DAGMA (Nonlinear) [2]:	0.4897 ± 0.1854	0.3381 ± 0.1350	0.3735 ± 0.1415	0.3296 ± 0.0864	0.2058 ± 0.0892	<u>0.0811</u> ± 0.0204	<u>0.0708</u> ± 0.0374	<u>0.0781</u> ± 0.0315
NOTEARS (Linear) [34]:	0.2459 ± 0.0949	0.1940 ± 0.0335	0.1831 ± 0.0328	0.2024 ± 0.0238	0.0613 ± 0.0173	0.0548 ± 0.0068	0.0494 ± 0.0080	0.0577 ± 0.0100
NOTEARS (Nonlinear) [35]:	0.1762 ± 0.0327	0.1559 ± 0.0209	0.1734 ± 0.0326	0.1763 ± 0.0394	0.0494 ± 0.0096	0.0456 ± 0.0058	0.0417 ± 0.0069	0.0462 ± 0.0042
GES [4]:	0.2012 ± 0.1185	0.7465 ± 0.1809	<u>0.9963</u> ± 0.0078	0.9938 ± 0.0111	-0.3315 ± 0.0820	-0.1133 ± 0.0631	0.0046 ± 0.1675	-0.0089 ± 0.0175
PC [24]:	0.4374 ± 0.2841	0.8618 ± 0.1168	0.9601 ± 0.0637	<u>0.9950</u> ± 0.0065	-0.2749 ± 0.0535	-0.0404 ± 0.0358	-0.0052 ± 0.0131	-0.0013 ± 0.0023
kPC (k=1) [13]:	<u>0.4708</u> ± 0.2792	0.9846 ± 0.0166	0.9987 ± 0.0027	0.9975 ± 0.0033	-0.1443 ± 0.0624	-0.0175 ± 0.0295	-0.0014 ± 0.0031	0.0000 ± 0.0000
kPC (k=2) [13]:	0.4528 ± 0.2777	<u>0.9731</u> ± 0.0295	0.9987 ± 0.0027	0.9975 ± 0.0033	-0.1893 ± 0.0730	-0.0266 ± 0.0386	-0.0014 ± 0.0031	0.0000 ± 0.0000
DAGPA (Ours)	0.7190 ± 0.1381	0.9629 ± 0.0264	0.9899 ± 0.0079	0.9927 ± 0.0051	<u>0.0907</u> ± 0.0570	0.1031 ± 0.0610	0.1851 ± 0.0615	0.3492 ± 0.1561

(b) Synthetic binary SF, $r = 4$, CPDAG F1 Arrowhead

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0091 ± 0.0111	0.0020 ± 0.0063	0.0000 ± 0.0000	0.0010 ± 0.0032
DAGMA (Nonlinear) [2]:	<u>0.1100</u> ± 0.1109	0.1141 ± 0.0839	0.0759 ± 0.1125	0.0713 ± 0.0985	0.1244 ± 0.0284	0.0727 ± 0.0527	0.0943 ± 0.0360	0.0954 ± 0.0361
NOTEARS (Linear) [34]:	0.0125 ± 0.0395	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0000 ± 0.0000	0.0191 ± 0.0120	0.0040 ± 0.0084	0.0041 ± 0.0085	0.0051 ± 0.0087
NOTEARS (Nonlinear) [35]:	0.0105 ± 0.0333	0.0061 ± 0.0192	0.0000 ± 0.0000	0.0053 ± 0.0166	0.0100 ± 0.0114	0.0080 ± 0.0103	0.0051 ± 0.0086	0.0000 ± 0.0000
GES [4]:	0.0253 ± 0.0544	0.1724 ± 0.1392	0.2427 ± 0.0730	0.4408 ± 0.1339	0.0663 ± 0.0216	0.2716 ± 0.0383	0.5297 ± 0.0529	<u>0.3611</u> ± 0.3114
PC [24]:	0.0223 ± 0.0300	0.2349 ± 0.1248	0.3093 ± 0.1109	0.3147 ± 0.1203	<u>0.0730</u> ± 0.0231	<u>0.2456</u> ± 0.0320	<u>0.3968</u> ± 0.0569	0.4005 ± 0.0362
kPC (k=1) [13]:	0.0211 ± 0.0361	0.0528 ± 0.0543	0.1813 ± 0.1249	0.1299 ± 0.0942	0.0360 ± 0.0192	0.0333 ± 0.0136	0.0169 ± 0.0154	0.0211 ± 0.0137
kPC (k=2) [13]:	0.0159 ± 0.0349	0.0452 ± 0.0656	0.1128 ± 0.0828	0.1637 ± 0.1004	0.0400 ± 0.0157	0.0427 ± 0.0150	0.0430 ± 0.0229	0.0590 ± 0.0226
DAGPA (Ours)	0.1558 ± 0.0800	<u>0.2338</u> ± 0.0763	<u>0.2589</u> ± 0.1168	0.2731 ± 0.1252	0.0172 ± 0.0164	0.0276 ± 0.0155	0.0331 ± 0.0118	0.0476 ± 0.0197

(c) Synthetic binary SF, $r = 4$, CPDAG F1 Skeleton

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.1182 ± 0.0801	0.0625 ± 0.0319	0.0846 ± 0.0569	0.1074 ± 0.0528	0.0957 ± 0.0156	0.0722 ± 0.0154	0.0637 ± 0.0199	0.0785 ± 0.0248
DAGMA (Nonlinear) [2]:	<u>0.3830</u> ± 0.0892	0.2930 ± 0.1131	0.2995 ± 0.0944	0.3001 ± 0.0924	0.2290 ± 0.0240	0.2179 ± 0.0538	0.2149 ± 0.0313	0.2347 ± 0.0506
NOTEARS (Linear) [34]:	0.1631 ± 0.1014	0.1159 ± 0.0605	0.0980 ± 0.0618	0.1341 ± 0.0477	0.1305 ± 0.0223	0.1040 ± 0.0249	0.0922 ± 0.0249	0.1112 ± 0.0195
NOTEARS (Nonlinear) [35]:	0.0739 ± 0.0750	0.0284 ± 0.0504	0.0791 ± 0.0714	0.0817 ± 0.0957	0.0405 ± 0.0246	0.0293 ± 0.0205	0.0294 ± 0.0171	0.0245 ± 0.0206
GES [4]:	0.1416 ± 0.1132	0.4808 ± 0.0780	0.7679 ± 0.0314	<u>0.8424</u> ± 0.0302	0.1747 ± 0.0148	0.4210 ± 0.0358	0.7082 ± 0.0507	0.4682 ± 0.4031
PC [24]:	0.3141 ± 0.1136	0.5696 ± 0.0531	0.7348 ± 0.0409	0.7840 ± 0.0457	0.1984 ± 0.0180	0.5263 ± 0.0269	0.7387 ± 0.0342	0.7675 ± 0.0187
kPC (k=1) [13]:	0.3237 ± 0.1185	0.6298 ± 0.0675	0.8036 ± 0.0174	0.8551 ± 0.0264	<u>0.2114</u> ± 0.0244	0.5395 ± 0.0290	0.6921 ± 0.0280	0.6553 ± 0.0214
kPC (k=2) [13]:	0.3141 ± 0.1136	0.5836 ± 0.0662	0.7538 ± 0.0483	0.8036 ± 0.0361	0.2005 ± 0.0206	<u>0.5344</u> ± 0.0268	<u>0.7307</u> ± 0.0329	<u>0.7388</u> ± 0.0135
DAGPA (Ours)	0.5007 ± 0.0512	<u>0.5889</u> ± 0.0564	0.6445 ± 0.0586	0.6512 ± 0.0560	0.0435 ± 0.0249	0.0677 ± 0.0178	0.0726 ± 0.0265	0.0956 ± 0.0358

(d) Synthetic binary SF, $r = 4$, CPDAG SHD

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	38.6000 ± 0.8433	39.1000 ± 0.7379	38.6000 ± 1.1738	38.3000 ± 1.0593	197.9000 ± 1.6633	198.0000 ± 1.2472	198.6000 ± 1.7764	197.8000 ± 1.4757
DAGMA (Nonlinear) [2]:	36.4000 ± 2.0656	37.1000 ± 1.8529	36.8000 ± 3.0478	36.1000 ± 2.7669	221.4000 ± 10.6999	190.5000 ± 4.3525	189.6000 ± 3.6878	188.4000 ± 3.9497
NOTEARS (Linear) [34]:	38.5000 ± 1.5092	38.4000 ± 1.4298	38.3000 ± 1.3375	38.1000 ± 1.3703	198.9000 ± 2.4244	197.4000 ± 1.3499	198.0000 ± 2.1602	197.1000 ± 1.5239
NOTEARS (Nonlinear) [35]:	39.5000 ± 1.0801	39.8000 ± 0.4216	39.5000 ± 1.2693	39.5000 ± 0.7071	199.8000 ± 1.8135	199.1000 ± 1.3703	199.4000 ± 0.8433	199.4000 ± 1.2649
GES [4]:	31.2000 ± 16.4776	35.9000 ± 2.8067	35.4000 ± 2.9136	28.1000 ± 5.0870	197.6000 ± 3.4705	167.4000 ± 5.8916	127.0000 ± 12.9013	75.6000 ± 65.6374
PC [24]:	39.6000 ± 1.3499	34.7000 ± 2.8304	33.0000 ± 4.1366	32.0000 ± 4.7376	206.9000 ± 2.3310	181.4000 ± 8.2892	158.6000 ± 15.1526	177.7000 ± 11.6433
kPC (k=1) [13]:	39.8000 ± 1.3166	33.9000 ± 3.6040	32.7000 ± 4.7152	34.1000 ± 4.3063	207.8000 ± 2.1499	188.9000 ± 11.5609	196.5000 ± 13.1085	270.3000 ± 20.3309
kPC (k=2) [13]:	39.9000 ± 1.2867	34.2000 ± 2.5734	33.5000 ± 3.4721	32.2000 ± 4.8944	205.7000 ± 2.5841	181.6000 ± 9.2280	167.9000 ± 14.4564	203.5000 ± 7.9757
DAGPA (Ours)	37.1250 ± 2.8482	36.0000 ± 2.2315	35.1600 ± 3.6609	34.7400 ± 3.7022	233.0000 ± 19.1066	244.4800 ± 10.0249	250.1000 ± 8.9972	272.6444 ± 21.6671

(e) Synthetic binary SF, $r = 4$, DAG F1

Method	Dataset Settings							
	$d=10, n=100$	$d=10, n=1k$	$d=10, n=10k$	$d=10, n=100k$	$d=50, n=100$	$d=50, n=1k$	$d=50, n=10k$	$d=50, n=100k$
DAGMA (Linear) [2]:	0.0682 ± 0.0555	0.0433 ± 0.0271	0.0517 ± 0.0409	0.0651 ± 0.0493	0.0673 ± 0.0142	0.0530 ± 0.0128	0.0406 ± 0.0086	0.0603 ± 0.0218
DAGMA (Nonlinear) [2]:	0.2486 ± 0.0855	0.1928 ± 0.1048	0.1950 ± 0.0860	0.1987 ± 0.0994	0.1565 ± 0.0358	0.1362 ± 0.0558	0.1259 ± 0.0300	0.1475 ± 0.0555
NOTEARS (Linear) [34]:	0.1008 ± 0.0867	0.0882 ± 0.0547	0.0608 ± 0.0435	0.0879 ± 0.0382	0.0845 ± 0.0229	0.0738 ± 0.0156	0.0628 ± 0.0174	0.0763 ± 0.0203
NOTEARS (Nonlinear) [35]:	0.0369 ± 0.0560	0.0096 ± 0.0203	0.0276 ± 0.0385	0.0497 ± 0.0714	0.0144 ± 0.0149	0.0127 ± 0.0131	0.0118 ± 0.0077	0.0098 ± 0.0080
DAGPA (Ours)	0.2219 ± 0.0624	0.2752 ± 0.0778	0.3358 ± 0.1050	0.3581 ± 0.0739	0.0235 ± 0.0144	0.0334 ± 0.0174	0.0416 ± 0.0110	0.0532 ± 0.0150

1506 E.3 Results on real-world data

1507 We show the results with additional metrics on Sachs [20] dataset and on an additional real-world
1508 dataset Lucas [10] in Figure 9.

1509 F Related Work

1510 The methodological evolution of causal discovery has progressed through two primary
1511 paradigms—constraint-based and score-based approaches—with recent advances bridging their
1512 complementary strengths.

1513 **Constraint-Based Methods** Pioneered by [25], constraint-based methods leverage conditional
1514 independence (CI) tests to reconstruct causal graphs. The PC algorithm [24] established core
1515 principles: (1) infer conditional independencies via statistical tests, (2) eliminate edges violating
1516 d-separation rules, and (3) orient edges using collider detection. While effective in principle, finite-
1517 sample reliability suffered from error propagation in high-order CI tests [6]. Extensions like FCI [31]
1518 addressed latent confounding through more sophisticated separation criteria, and Kernel CI Test
1519 (KCIT) [32] enabled nonparametric testing via Hilbert space embeddings. Modern variants like
1520 LOCI [28] demonstrated that low-order CI statements suffice for structure recovery under specific
1521 faithfulness conditions, while k-PC [13] provided theoretical guarantees for bounded-order testing.

1522 **Score-Based Methods** Score-based methods reformulated causal discovery as a combinatorial
1523 optimization problem, maximizing score functions (*e.g.*, BIC [21]) over DAG spaces. GES [4]
1524 advanced this paradigm through greedy equivalence class search, but scalability remained constrained
1525 by discrete search spaces. The field transformed with [34]’s NOTEARS framework, which redefined
1526 acyclicity through a differentiable constraint:

$$\text{tr}(e^{W \circ W}) - d = 0, \quad (12)$$

1527 where W is the weighted adjacency matrix. This enabled continuous gradient-based optimization,
1528 spawning derivatives like DAGMA [2] with improved stability via spectral radius constraints:

$$\det(I - \alpha W \circ W) = 1, \quad (13)$$

1529 and GOLEM [18] using likelihood-based objectives. Nonlinear extensions [35] incorporated neural
1530 networks to model complex functional relationships while preserving acyclicity.

1531 **Paradigm Convergence.** Modern methods synthesize constraint- and score-based approaches.
1532 CCD [28] combines logical reasoning with low-order CI constraints. Differentiable architectures now
1533 integrate constraint satisfaction through novel loss functions [35] enforces additive noise models via
1534 neural mechanisms, and [30]’s DAG-GNN combines variational inference with graph networks. Our
1535 approach extends this synthesis by transforming discrete CI decisions into continuous differentiable
1536 objectives, expressed through probabilistic constraint satisfaction:

$$\mathcal{LCI} = \sum (i, j | S) |\mathbb{P}(X_i \perp X_j | X_S) - \hat{\mathbb{P}}_\theta(X_i \perp X_j | X_S)|^2, \quad (14)$$

1537 where θ parameterizes the learned causal graph. This hybrid framework preserves the interpretability
1538 of constraint-based methods while leveraging the scalability of continuous optimization.

1539 **Frontier Challenges.** Current research addresses persistent limitations: handling latent confounders
1540 through instrumental variable methods [14], incorporating interventional data via differentiable
1541 experimental design [3], and scaling to high-dimensional settings with graph sparsity priors [9]. Our
1542 methodology’s constraint-driven architecture naturally extends to these scenarios through modular
1543 constraint incorporation, offering a unified framework for advancing causal discovery in complex
1544 real-world settings.

1545 G Licenses

1546 In this work, we evaluated our method on two publicly available causal discovery benchmark datasets:

1547 **Sachs Dataset:** The Sachs dataset contains simultaneous measurements of 11 phosphorylated proteins
1548 and phospholipids derived from thousands of individual primary immune system cells, subjected to

1549 both general and specific molecular interventions. This dataset was originally published by Sachs et
1550 al. (2005) and is widely used as a benchmark in causal discovery research. The dataset is publicly
1551 available through multiple repositories including bnlearn [22], other causal discovery toolboxes.

1552 **Availability:** The dataset is publicly accessible for research purposes through various causal discovery
1553 software packages and repositories.

1554 **LUCAS Dataset:** The LUCAS (LUng CAncer Simple set) dataset [10] is a synthetic benchmark
1555 dataset consisting of 12 binary variables and 2000 instances, representing 12 different causal rela-
1556 tionships in a medical diagnosis problem for identifying patients with lung cancer. This dataset was
1557 created as part of the Causality Workbench project to provide standardized benchmarks for testing
1558 causal discovery algorithms.

1559 **Availability:** The dataset is publicly available for research and educational purposes through the
1560 Causality Workbench repository.

1561 **Usage Declaration:** Both datasets were used in accordance with their respective terms of use for
1562 academic research purposes. No additional permissions were required for their use in this study.

1563 H Societal Impact Statement

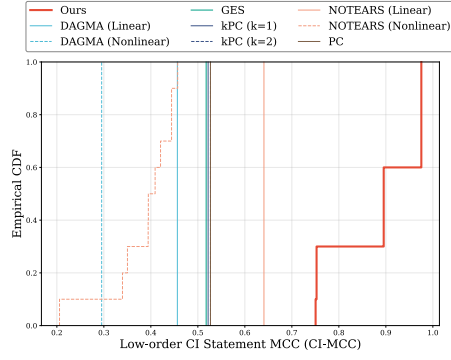
1564 Our differentiable d-separation framework for causal discovery has potential positive impacts in
1565 healthcare (identifying causal factors in disease), public policy (evaluating intervention effective-
1566 ness), scientific discovery (understanding complex systems), and algorithmic fairness (distinguishing
1567 causation from correlation in decision systems). However, several risks must be acknowledged: (1)
1568 incorrect causal discoveries could lead to harmful interventions if implemented without domain
1569 expert validation; (2) causal discovery in sensitive domains may raise privacy concerns, necessi-
1570 tating differential privacy techniques; (3) when applied to historically biased datasets, discovered
1571 relationships might reflect and perpetuate these biases rather than ground truth; and (4) computational
1572 requirements might limit accessibility to well-resourced institutions. We recommend multiple mitiga-
1573 tions: returning diverse candidate structures rather than single models, requiring expert validation
1574 before implementation, implementing privacy-preserving techniques with sensitive data, examining
1575 discovered relationships for bias, and developing more efficient implementations to improve acces-
1576 sibility. Causal discovery tools require responsible application and domain expertise, especially in
1577 high-stakes domains where incorrect inferences could lead to harmful consequences.

1578 Additional References

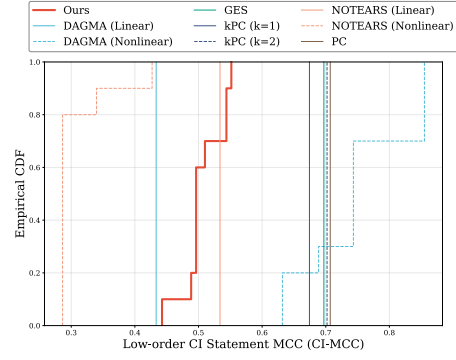
- 1579 [1] Samy Badreddine, Luciano Serafini, and Michael Spranger. logltn: Differentiable fuzzy logic
1580 in the logarithm space. *ArXiv*, abs/2306.14546, 2023.
- 1581 [2] Kevin Bello, Bryon Aragam, and Pradeep Ravikumar. Dagma: Learning dags via m-matrices
1582 and a log-determinant acyclicity characterization. In S. Koyejo, S. Mohamed, A. Agarwal,
1583 D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*,
1584 volume 35, pages 8226–8239. Curran Associates, Inc., 2022.
- 1585 [3] Philippe Brouillard, Sébastien Lachapelle, Alexandre Lacoste, Simon Lacoste-Julien, and
1586 Alexandre Drouin. Differentiable causal discovery from interventional data. In *Advances in*
1587 *Neural Information Processing Systems*, volume 33, pages 21865–21877, 2020.
- 1588 [4] David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of*
1589 *machine learning research*, 3(Nov):507–554, 2002.
- 1590 [5] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research.
1591 *InterJournal*, Complex Systems:1695, 11 2005.
- 1592 [6] Iddo Drori, Anant Kharkar, William R Sickinger, Brandon Kates, Qiang Ma, Suwen Ge,
1593 Eden Dolev, Brenda Dietrich, David P Williamson, and Madeleine Udell. Learning to solve
1594 combinatorial optimization problems on real-world graphs in linear time. In *2020 19th IEEE*
1595 *International Conference on Machine Learning and Applications (ICMLA)*, pages 19–24. IEEE,
1596 2020.

- [7] Gaspard Ducamp, Christophe Gonzales, and Pierre-Henri Wuillemin. aGrUM/pyAgrum : a Toolbox to Build Models and Algorithms for Probabilistic Graphical Models in Python. In *10th International Conference on Probabilistic Graphical Models*, volume 138 of *Proceedings of Machine Learning Research*, pages 609–612, Skørping, Denmark, September 2020.
- [8] HL Frisch and JM Hammersley. Percolation processes and related topics. *Journal of the society for industrial and applied mathematics*, 11(4):894–918, 1963.
- [9] Ruichu Guo, Wenming Cheng, Jundong Li, Matthew Hahn, and Huan Liu. A graphical model approach to structural causal modeling. *arXiv preprint arXiv:1909.09189*, 2019.
- [10] Isabelle Guyon, Constantin Aliferis, Gregory Cooper, André Elisseeff, Jean Philippe Pellet, Peter Spirtes, and Alexander Statnikov. Causality workbench. In *Causality in the sciences*. Oxford University Press, 2011.
- [11] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.
- [12] Barry D Hughes. *Random walks and random environments*. Oxford University Press, 1996.
- [13] Murat Kocaoglu. Characterization and learning of causal graphs with small conditioning sets. *Advances in Neural Information Processing Systems*, 36, 2023.
- [14] Jack Kuipers, Giusi Moffa, and David Heckerman. Generalized instrumental variables: A robust approach to causal inference with weak instruments. *Journal of Machine Learning Research*, 24(13):1–45, 2023.
- [15] Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021.
- [16] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.
- [17] Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. In *International Conference on Machine Learning*, pages 16428–16446. PMLR, 2022.
- [18] Ignavier Ng, Kun Zhang, Bryon Aragam, and Pradeep Ravikumar. Learning dags with continuous optimization and optimal transport. *Advances in Neural Information Processing Systems*, 33:17416–17427, 2020.
- [19] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 1558604790.
- [20] Karen Sachs, Omar Perez, Dana Pe’er, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721):523–529, 2005.
- [21] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.
- [22] Marco Scutari and Jean-Baptiste Denis. *Bayesian Networks with Examples in R*. Chapman and Hall, Boca Raton, 2nd edition, 2021. ISBN 978-0367366513.
- [23] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. In *Neural Information Processing Systems*, 2018.
- [24] Peter Spirtes and Clark Glymour. An algorithm for fast recovery of sparse causal graphs. *Social science computer review*, 9(1):62–72, 1991.
- [25] Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

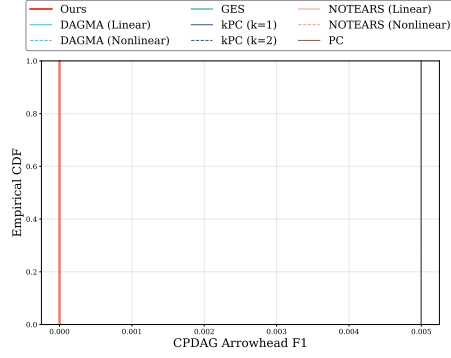
- 1643 [26] Emile van Krieken, Erman Acar, and Frank van Harmelen. Analyzing differentiable fuzzy logic
1644 operators. *Artificial Intelligence*, 302:103602, 2022.
- 1645 [27] Zirui Wang and Yulia Tsvetkov. Gradient vaccine: Investigating and improving multi-task
1646 optimization in massively multilingual models. In *Proceedings of the International Conference*
1647 *on Learning Representations (ICLR)*, 2021.
- 1648 [28] Marcel Wienöbst and Maciej Liskiewicz. Recovering causal structures from low-order con-
1649 ditional independencies. In *Proceedings of the AAAI Conference on Artificial Intelligence*,
1650 volume 34, pages 10302–10309, 2020.
- 1651 [29] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn.
1652 Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*,
1653 33:5824–5836, 2020.
- 1654 [30] Yue Yu, Jie Chen, Tian Gao, and Mo Yu. Dag-gnn: Dag structure learning with graph neural
1655 networks. In *International conference on machine learning*, pages 7154–7163. PMLR, 2019.
- 1656 [31] Jiji Zhang. Causal inference and reasoning in causally insufficient systems. *Philosophy of*
1657 *Science*, 75(5):960–970, 2008.
- 1658 [32] Kun Zhang, Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. Kernel-based conditional
1659 independence test and application in causal discovery. In *Proceedings of the 27th Conference*
1660 *on Uncertainty in Artificial Intelligence (UAI)*, pages 804–813, 2012.
- 1661 [33] Ruqi Zhang, Xingchao Liu, and Qiang Liu. A langevin-like sampler for discrete distributions.
1662 In *International Conference on Machine Learning*, pages 26375–26396. PMLR, 2022.
- 1663 [34] Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears:
1664 Continuous optimization for structure learning. *Advances in neural information processing*
1665 *systems*, 31, 2018.
- 1666 [35] Xun Zheng, Chen Dan, Bryon Aragam, Pradeep Ravikumar, and Eric Xing. Learning sparse
1667 nonparametric dags. In *International Conference on Artificial Intelligence and Statistics*, pages
1668 3414–3425. PMLR, 2020.
- 1669 [36] Yujia Zheng, Biwei Huang, Wei Chen, Joseph Ramsey, Mingming Gong, Ruichu Cai, Shohei
1670 Shimizu, Peter Spirtes, and Kun Zhang. Causal-learn: Causal discovery in python. *Journal of*
1671 *Machine Learning Research*, 25(60):1–8, 2024.



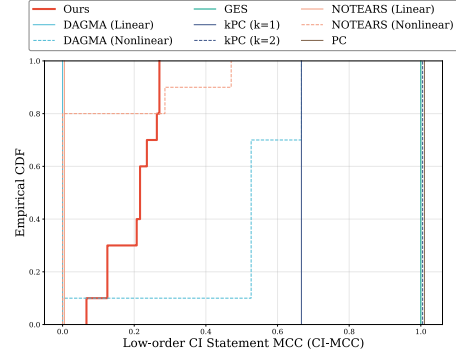
(a) Sachs CI-MCC



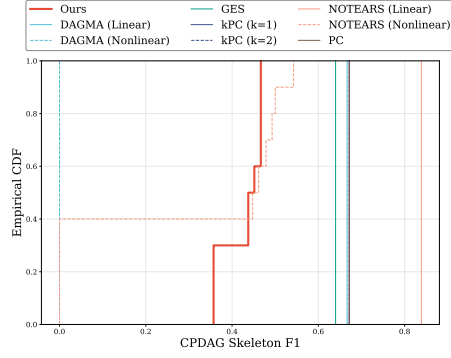
(b) Lucas CI-MCC



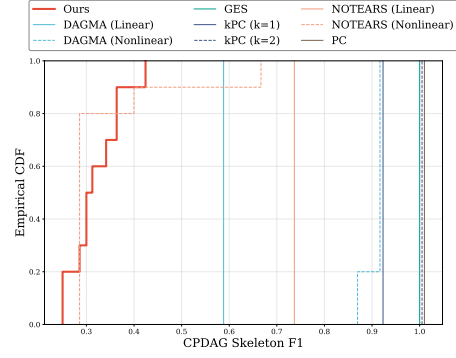
(c) Sachs CPDAG F1 Arrowhead



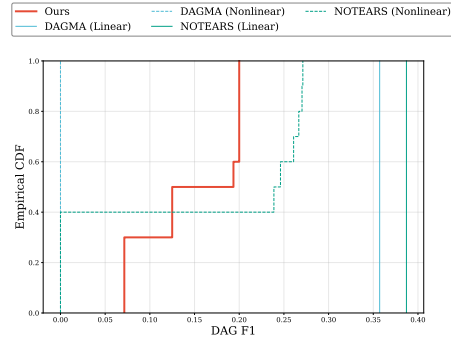
(d) Lucas CPDAG F1 Arrowhead



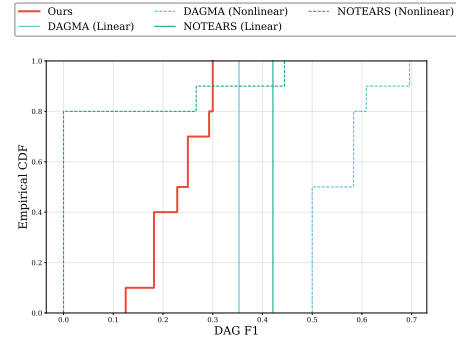
(e) Sachs CPDAG F1 Skeleton



(f) Lucas CPDAG F1 Skeleton



(g) Sachs DAG F1



(h) Lucas DAG F1

Figure 9: Full results on Sachs [20] and Lucas [10] real-world dataset.