EXMAG: LEARNING OF MAXIMALLY ANCESTRAL GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

In mixed graphs, there are both directed and bidirected edges. An extension of acyclicity to this mixed-graph setting is known as maximally ancestral graphs. This extension is of considerable interest in causal learning in the presence of confounders. There, directed edges represent a clear direction of causality, while bidirected edges represent confounding.

We propose a branch-and-cut algorithm for learning maximally ancestral graphs using a formulation as a mixed-integer quadratic program. Empirically, our method achieves comparable or improved reconstruction quality while requiring an order of magnitude fewer samples than state-of-the-art approaches.

1 INTRODUCTION

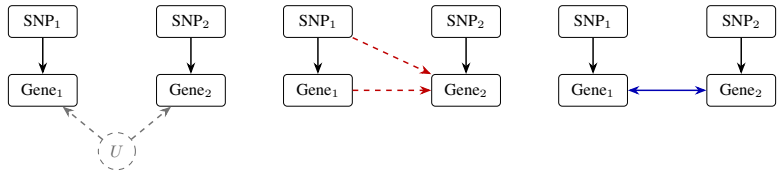
As one transitions from statistical to causal learning (Schölkopf & von Kügelgen, 2022), one is seeking the most appropriate causal model. Bayesian networks (BN) (Pearl, 1985) are a popular model, where a directed acyclic graph represents causal relationships. The vertices represent random variables, and the absence of oriented edges encodes conditional variable independence and, therefore, the edges suggest the causal relationships. The conditional probability distributions assigned to vertices quantify the strength of the causal relationships. While the learned graph needs to be acyclic, a major challenge in learning BNs is confounding.

Simpson's paradox shows that without considering confounding factors in statistical analysis (McElreath, 2018), the direction of causality can be misestimated completely. A textbook example (McElreath, 2018) comes from the Berkeley graduate admissions (Bickel et al., 1977). The data show that women find it harder to get admitted to Berkeley graduate schools. Nevertheless, this is because women tend to apply to departments that have lower admission rates. In this example, the choice of the graduate school is the confounder, impacting the probability of admission. Confounding is prevalent throughout high-dimensional statistics (Lin et al., 2014; Gilad & Mizrahi-Man, 2015).

Specifically, in biomedical sciences, confounders such as socio-economic status, age, or lifestyle factors can distort the true causal relationship between treatments and outcomes (Zhou et al., 2022). Techniques such as instrumental variables (Reiersøl, 1945; Imbens, 2014), propensity score matching (Rosenbaum & Rubin, 1983), and double machine learning (Chernozhukov et al., 2018) have been widely used to mitigate the effects of confounding in clinical trials and observational studies. To mitigate confounding biases, statistical models that explicitly account for hidden confounders, such as spectral methods and latent variable models, are often employed (Guo et al., 2022). Furthermore, meta-analysis and sensitivity analysis are often used to evaluate the robustness of findings in the presence of potential confounders, especially when combining results from multiple studies (Bühlmann, 2020; Mathur & VanderWeele, 2022).

Confounding arises when an unobserved or uncontrolled variable influences multiple observed variables simultaneously, thereby creating associations that need not correspond to direct causal effects (Bühlmann & Geer, 2011; Zhou et al., 2022). In causal structure learning, this is particularly challenging because hidden confounders can both obscure genuine causal directions and induce spurious dependencies among observed variables. Classical strategies such as instrumental variables, propensity score methods, double machine learning, and latent-variable adjustments can reduce the statistical impact of confounding (Reiersøl, 1945; Imbens, 2014; Rosenbaum & Rubin, 1983; Chernozhukov et al., 2018; Guo et al., 2022; Bühlmann & Ćevid, 2020), but they do not by themselves

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Ground truth with an unobserved confounder U (batch, cell type)

(b) Sparsest DAG consistent with the data: two spurious causal edges

(c) MAG: confounding encoded by a single bidirected edge

Figure 1: (a) Two genes share an unobserved confounder U. (b) Every DAG over the observed variables that is consistent with the data needs at least four edges, e.g. the spurious regulatory edge $\text{Gene}_1 \to \text{Gene}_2$ and the trans-effect $\text{SNP}_1 \to \text{Gene}_2$. (c) The MAG represents the same independences exactly, encoding confounding by the single bidirected edge $\text{Gene}_1 \leftrightarrow \text{Gene}_2$.

produce a graphical representation of the underlying causal system. This motivates learning graph classes that encode latent confounding explicitly, especially in partially observed settings where assumptions such as full observability are unrealistic (Wang & Zhou, 2021; Bhattacharya et al., 2021).

Maximal Ancestral Graphs (MAGs) (Richardson & Spirtes, 2002) provide a natural extension of Bayesian networks to settings with latent confounding. Rather than estimating a Directed Acyclic Graph (DAG) alone, one can estimate a MAG, which represents both direct causal effects and dependencies induced by hidden confounders through directed and bidirected edges. This richer representation allows MAGs to capture relationships that DAG-based models cannot express without explicitly introducing latent variables, making them better suited to partially observed causal systems.

There are only a few studies of MAG estimation (Chen et al., 2021; Rantanen et al., 2021; Claassen & Bucur, 2022; Hu, 2023; Hu & Evans, 2024b;a; Dash et al., 2025). Methods in (Richardson, 2014; van Ommen, 2024) are applicable to both discrete and nonparametric cases, which extend DAG to MAG or ADMG diagrams. Factorization in MAG is not directly decomposable into individual variables and their parent sets, as in DAGs, but must instead consider components connected by bidirected paths (termed *districts* or *c-components*), cf. (Richardson, 2014), although Claassen & Bucur (2022) proposed to use Markov equivalence classes (MEC) instead. In 2021, Chen et al. (2021) introduced a first mixed-integer programming (MIP) formulation, but the number of variables scales with the number of c-components, i. e., exponentially with the number of vertices in the worst case. Such formulations are also known as extended formulations (Conforti et al., 2010). The same year, Rantanen et al. (2021) explored a score-based approach for directed MAG discovery, leveraging a local score function optimized using pruning rules and dynamic programming. Additionally, Zhou et al. (2022) address exogenous covariates in causal formulation that help explain the heterogeneity in both sampling and causal mechanisms. Hu (2023)'s dissertation presented an extension of the imsets of Studeny (2006) from directed acyclic graphs (DAGs) to MAGs (Hu & Evans, 2024b), which allows for the use of the methods of Studený, and a score-based heuristic (Hu & Evans, 2024a). More recently, Dash et al. (2025) enhanced the scalability of methods of Chen et al. (2021) by utilizing linear programming (LP) relaxations instead of solving the MIP.

Our approach proposes a formulation of MAG estimation within Mixed-Integer Quadratic Programming (MIQP). From the exponential set of constraints enforcing acyclicity, only those violated by a solution are iteratively added to the program in a lazy manner. As a result, the initial program is polynomial in the number of vertices, while the worst-case scenario with exponentially many constraints is avoided in many instances of the program. In contrast, the so-called extended formulations of (Chen et al., 2021; Dash et al., 2025) have the dimension exponential in the number of vertices. While both the extended formulation of Chen et al. (2021) and ours ensure that confounding factors are properly accounted for and the true underlying data are better represented by the model, our implementation scales further, from 4-5 variables in the extended formulation of Chen et al. (2021) to 25 or more variables with the proposed compact formulation.

1.1 Motivating Example

In cells, gene activity is often affected by cofactors. As a result, expression levels of two correlated genes that do not regulate each other are affected by an unmeasured factor, such as batch effect or tissue type, which might not be captured in transcriptomics data. Genotyping, however, provides

single-nucleotide polymorphisms, i.e., small changes to the genetic code that influence the genes. No DAG over the four observed variables encodes exactly this behavior. The sparsest DAG consistent with the data needs four edges (Fig. 1b). Two of them are spurious. Adding the latent variable explicitly avoids this, but it requires knowing the confounders in advance. A MAG represents the same independence exactly with three edges (Fig. 1c). The confounding appears as a single bidirected edge. More generally, any method restricted to DAGs must trade hidden confounding for spurious causal edges, whereas a MAG can express it explicitly.

2 GRAPHS AND PROPERTIES

A Directed Acyclic Graph (DAG) is a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ such that there are no directed cycles. That is, there is no sequence of distinct vertices $v_1, v_2, \ldots, v_k \in \mathcal{V}$ such that $(v_i, v_{i+1}) \in \mathcal{E}$ for all $1 \leq i \leq k-1$ and $(v_k, v_1) \in \mathcal{E}$. Maximal ancestral graphs (MAGs), introduced by Richardson & Spirtes (2002), provide a framework for modeling distributions through conditional independence relations. Compared with DAGs, MAGs allow for latent confounders, accommodating data that arise from distributions with more complex independence structures and revealing hidden states in the graphs. While DAGs allow for the efficient computation of maximum likelihood estimates (MLEs) and scoring (e.g., via BIC), these properties are challenging to extend to MAGs due to their structural and computational complexity (Hu & Evans, 2020).

ADMG Mixed graphs feature two types of edges: directed ($\rightarrow$) and bidirected ($\leftrightarrow$). Mixed graph $\mathcal{G}$ thus consists of vertex set $\mathcal{V}$, a set of directed edges $\mathcal{E}$ and bidirected edges $\mathcal{B}$, where $\mathcal{E}$ are ordered pairs of vertices, while $\mathcal{B}$ are unordered 2-element subsets of distinct vertices. For vertex v in $\mathcal{V}$, we define the *parents*, *spouses*, and *ancestors* of v, respectively as:

$$\begin{aligned}
\text{pa}_{\mathcal{G}}(v) &= \{w : w \rightarrow v \text{ in } \mathcal{G}\},\\
\text{an}_{\mathcal{G}}(v) &= \{w : w \rightarrow \cdots \rightarrow v \text{ in } \mathcal{G} \text{ or } w = v\},\\
\text{sp}_{\mathcal{G}}(v) &= \{w : w \leftrightarrow v \text{ in } \mathcal{G}\}.
\end{aligned}$$

As in a DAG, a mixed graph is acyclic if it contains no directed cycles in $\mathcal{E}$, i.e., an acyclic directed mixed graph (ADMG) (Hu, 2023).

A directed mixed graph $\mathcal{G}$ is an ancestral ADMG if it has no directed cycles and no vertex is an ancestor of any of its spouses. Formally,

$$\text{if } v \neq w \text{ and } v \in \text{an}_{\mathcal{G}}(w) \cup \text{sp}_{\mathcal{G}}(w), \text{ then } w \notin \text{an}_{\mathcal{G}}(v).$$

Equivalently, $\mathcal{G}$ is an ancestral ADMG if it contains no directed cycles or almost directed cycles (Chen et al., 2021; Hu, 2023). In an ADMG, an almost directed cycle is of the form $v \rightarrow u \rightarrow \ldots \rightarrow w \leftrightarrow v$; in other words, $\{w, v\} \in \mathcal{B}$ is a bidirected edge, and $v \in \text{an}_{\mathcal{G}}(w)$ (Chen et al., 2021).

A vertex u on a non-overlapping path is called a collider if it is contained in a non-overlapping subpath (w, u, v) with two arrowheads into u. In mathematical form, u is a collider on path $P = p_1 p_2 \cdots wuv \cdots p_k$ if and only if either $w \rightarrow u$ or $w \leftrightarrow u$, and at the same time $u \leftarrow v$ or $u \leftrightarrow v$.

Inducing Paths A path $P = (v_1, \ldots, v_k)$ in an ADMG is an inducing path (**?**Zhang) if every non-endpoint vertex on P is a collider and an ancestor of at least one of v_1 or v_k.

m-separation Graphs encode conditional independence via separation criteria. For acyclic directed mixed graphs (ADMGs), *m-separation* generalizes d-separation to handle bidirected edges. A path between vertices u and v is *m-connecting* given a conditioning set $C \subseteq \mathcal{V}$ if: (i) u and v are the endpoints; (ii) all non-colliders are not in C; and (iii) all colliders are in $\text{an}_{\mathcal{G}}(C)$. Vertices u and v are *m-separated* given C if no such path exists.

Maximal Ancestral Graph An *ancestral* ADMG $\mathcal{G}$ is called a maximal ancestral graph (MAG) if the *maximality* condition holds, i.e., for every pair of nonadjacent vertices u and v, there exists some set C such that u, v are *m-separated* given C in $\mathcal{G}$. We refer to Hu (2023) for multiple examples. The existence of a directed cycle or an almost directed cycle contradicts ancestrality. Moreover, in an ancestral ADMG, the existence of an inducing path between two nonadjacent vertices implies that the graph is not maximal.

3 STRUCTURAL EQUATION MODEL WITH LATENT CONFOUNDERS

Graphical causal models provide a principled framework for representing causal relationships and reasoning about conditional independence structures among observed variables in the presence of latent variables. Here, we adopt the Structural Equation Model (SEM) framework (Bollen, 1989; Pearl, 2009), which allows causal relationships among variables to be represented through structural equations while explicitly accommodating unobserved factors. Under this framework, the data are generated so that

$$X \leftarrow XW_D + LW_L + \epsilon,$$

where $X = (X_1, \ldots, X_d)$ is the vector of variables, $L = (L_1, \ldots, L_p)$ is the vector of latent variables, W_D is a $\mathbb{R}^{d\times d}$ weighted adjacency matrix of the DAG encoding the causal structure, $W_L \in \mathbb{R}^{p\times d}$ encodes structure of latent variables, and ϵ is the noise vector.

Further, we will assume that the latent confounding term LW_L has been absorbed into the error term ϵ.

$$X \leftarrow XW_D + \epsilon.$$

The covariance matrix $\beta = \mathbb{E}[\epsilon\epsilon^T]$ will represent the hidden confounders.

If each ϵ_i is Gaussian, then the resulting distribution $p(X)$ is multivariate normal with zero mean and covariance

$$\Sigma = (I - W_D)^{-T} W_B (I - W_D)^{-1}.$$

The cost function then becomes the error of the prediction, represented by the difference between the sampled data X and their prediction stemming from observed variables, and the influence of latent confounders, similarly to Bhattacharya et al. (2021). See derivation in the Supplementary materials.

$$J(W_D, W_B) = \|X - XW_D - (X - XW_D)\, W_B\|^2.$$

3.1 CONNECTION TO CAUSALITY

The causal parameters W_D and W_B are the solution of the following minimization problem:

$$\arg \min_{W_D, W_B} \max_{P\in\mathcal{P}} \mathbb{E}_P \left[(X - XW_D - (X - XW_D)\, W_B)^2 \right],$$

where $\mathcal{P}$ is a class of distributions containing perturbations of the original distribution, including confounding. Class $\mathcal{P}$, therefore, guides the overall structure of the MIQP formulation.

Now we denote by X the data matrix. Let $X \in \mathbb{R}^{n\times d}$ be an $n \times d$ matrix with data samples. Then, under the assumption of Gaussian noise, the problem can be reformulated as

$$\arg \min_{W_B, W_D} J(W_D, W_B). \tag{1}$$

4 FORMULATION OF THE MIXED INTEGER QUADRATIC PROGRAM

Here, we present the formulation of the Mixed Integer Quadratic Program (MIQP) used to infer the causal structure, with binary matrix $B = [b_{j,k}] \in \{0, 1\}^{d\times d}$ introduced to account for relationships explained by confounding factors, alongside binary adjacency matrix $E = [e_{j,k}] \in \{0, 1\}^{d\times d}$ adopted from the ExDAG model (Rytíř et al., 2024). Two weight matrices $W_D = [w_{j,k}] \in \mathbb{R}^{d\times d}$ and $W_B = [w_{j,k}] \in \mathbb{R}^{d\times d}$ encode the weights of directed and bidirected edges. Whenever entry $w_{Dj,k}$ ($w_{Bj,k}$, respectively) is non-zero, $e_{j,k}$ (directed edge) (or $b_{j,k}$ (bidirected edge), respectively) is nonzero. At the same time, we extend the existing formulation by introducing an additional optional binary input matrix $F = [f_{j,k}] \in \{0, 1\}^{d\times d}$, where $f_{j,k} = 1$, indicates that there is no direct causal relationship between variables j and k, but j and k might have a common latent confounder. This follows from the meaning of the edges in a MAG, a $\rightarrow$ edge implies a direct causal relationship, but does not rule out a possible latent confounding, $\leftrightarrow$ means no direct causal relationship. Such a matrix might be obtained by a statistical test or from background knowledge to improve the results if needed.

Formally, we define *Directed Edge Matrix* E as $e_{j,k} = 1$ if $j \to k$, 0 otherwise. Similarly, *Bidirected Edge Matrix* B is $b_{j,k} = 1$ if $j \leftrightarrow k$, and 0 otherwise. Lastly, the input matrix F with pairs of variables which are known not to be in a direct cause-effect relationship, is by definition $f_{j,k} = 1$ if $j \not\to k \wedge k \not\to j$ and 0 otherwise. Note that this matrix is by definition symmetric.

4.1 MIQP Formulation

The cost function for the Mixed Integer Quadratic Program of ExMAG is the l_q norm below. It has two components - the error of prediction of $X_{i,j}$, and the regularization term. Denote

$$R_{i,j} = X_{i,j} - \sum_{k=1;k\neq j}^{d} X_{i,k} w_{Dk,j}. \tag{2}$$

Then, the minimization problem in equation 1 extended by regularization terms corresponds to the minimization of the following cost function

$$\min_{W_D, W_B, E, B} \sum_{i=1}^{n} \sum_{j=1}^{d} \left| R_{i,j} - \sum_{k=1;k\neq j}^{d} R_{i,k} w_{Bk,j} \right|^q + \lambda \sum_{j=1}^{d} \sum_{k=1}^{d} (e_{j,k} + b_{j,k}), \tag{3}$$

In this formula, $X_{i,j}$ represents the value of the j-th variable for the i-th data point; $w_{Dk,j}$ represents the weight of the directed edge from variable k to variable j; $w_{Bk,j}$ represents the weight of the bidirected edge from variable k to variable j; $e_{j,k}$ is the binary decision variable indicating the presence of a directed edge from j to k; $b_{j,k}$ is the binary decision variable indicating a bidirected edge between j and k; $\lambda \in \mathbb{R}^+$ is a regularization parameter controlling the model fit and the edge penalty trade-off. The exponent $q \in \mathbb{N}$ can take values $q = 1$ or $q = 2$.

Optimization criterion in equation 3 implies that the dependencies between the variables are linear. The first part of the criterion encodes the actual cost as an error of the prediction, the second part encodes regularization, penalizing more edges with a larger λ.

As in the ExDAG (Rytíř et al., 2024) model, the weights are bounded by introducing large constant c, which is chosen large enough to exceed the maximum weight expected in the problem being solved. The bounding avoids bilinear terms in the cost function in equation 3 and takes the following form:

$$-c \cdot E \leq W_D \leq c \cdot E, \qquad \text{(Weight Constraint)}$$
$$-c \cdot B \leq W_B \leq c \cdot B, \tag{4}$$
$$E + B \leq \mathbf{1}. \qquad \text{(Edge Constraint)}$$

The Edge Constraint means that there cannot be a directed as well as a bidirected edge between the same two vertices. Additionally, we enforce that the bidirected matrix is symmetric by equation 5. If $f_{j,k} = 1$, then we know there is no direct causal relationship between j and k, and therefore, $e_{j,k} = 0$. This is formally enforced by equation 6. Conversely, $f_{j,k} = 0$ implies a directed edge rather than a bidirected edge between j and k in equation 7. Both equation 6 and equation 7 are enforced only if the respective fields in matrix F are defined, as they are optional.

$$B = B^T, \tag{5}$$
$$F + E \leq \mathbf{1}, \tag{6}$$
$$B \leq F. \tag{7}$$

Lastly, we must enforce conditions for directed or almost directed cycles and inducing paths. Those conditions are enforced lazily using a separation routine explained later. Directed cycles are enforced in a way adopted from Rytíř et al. (2024). Therefore, they are left out of this paper. An almost directed cycle formed by edges in set E' and a bidirected edge (u, v) is forbidden by the constraint

$$b_{u,v} + \sum_{\{j,k\} \in E'} e_{j,k} \leq |E'|. \qquad \text{(Acyclic Constraint)}$$

Similarly, if there is an inducing path formed by path P that contains bidirected edges, and set E' contains all directed edges that participate in the ancestor relationship (including multiple paths) between the inner points of the path and the terminals of P, this inducing path is forbidden by

$$\sum_{(j,k) \in P} b_{j,k} + \sum_{(j,k) \in E'} e_{j,k} \leq |E'| + |P| - 1. \qquad \text{(Inducing-Paths Constraint)}$$

Note that the second constraint does not necessarily eliminate the inducing path, as the optimizer might forbid one of the edges in E' without influencing the ancestor relationship. This results in path P being found in the next iteration, with a smaller set of directed edges, and the process is repeated. Proof of termination and more details on the lazy application of the constraints can be seen in the Supplementary Materials.

By enforcing these constraints, we ensure that the MIQP correctly models the causal relationships between the variables while respecting the structure defined by $e_{j,k}$ and the potential confounding relationships captured by $b_{j,k}$.

5 Separation Routine for the Maximal Ancestral Graphs

The main contribution of this section is the separation routine that identifies whether a graph is an instance of a maximal ancestral graph. To do so, we need to identify directed cycles, almost directed cycles, and inducing paths. The presence of directed cycles can be detected in $\mathcal{O}(d^2)$ using depth-first-search (DFS); such an approach can be found in (Rytíř et al., 2024). For both inducing paths and almost directed cycles, we will use the distance matrix D constructed on the graph of directed edges E. This distance matrix can be obtained, for example, using the Floyd-Warshall algorithm (Floyd, 1962).

Having the distance matrix, to check for almost directed cycles, we can iterate over all bidirected edges and test whether the distance between the endpoints using E is finite. If so, we have a directed path connected by a bidirected edge. See Algorithm 1 for details.

Algorithm 1 Function that identifies almost directed cycles.

Input: directed edges E, bidirected edges B

function ALMOST-DIRECTED-CYCLES(E, B)
  $D \leftarrow$ DISTANCE-MATRIX(E)
  for all $(j, k) \in \{1, 2, \ldots, d\} \times \{1, 2, \ldots, d\}$ **do**
    if $j \neq k$ & $b_{j,k} == 1$ & $D_{j,k} < \infty$ **then**
      $E' =$ TRACE-DISTANCE-MATRIX(D,E,j,k)
        ▷ Finds all edges on any j to k path, see Suppl.
      Found cycle formed by edges E' and $j \leftrightarrow k$

Algorithm 2 Simplified function that identifies inducing paths.

Input: directed edges E, bidirected edges B

function INDUCING-PATHS(E, B)
  $D \leftarrow$ DISTANCE-MATRIX(E)
  for all $s \in 1, 2, \ldots, d$ **do**
    INDUCING-PATHS-DFS(D, E, B, s, s, $\{1, 2, \ldots d\}$, $[s]$)
function INDUCING-PATHS-DFS(D, E, B, s, u, possible endpoints, path)
  if possible endpoints are empty **then return**
  if LEN(path) > 2 & u in possible endpoints **then**
    FOUND-INDUCING-PATH(D,E, path)
      ▷ Edges participating in ancestor relationship are recovered, see Suppl. mat.
  for all $v \in 1, 2, \ldots, d$ such that $b_{u,v} = 1$ **do**
    v-endpoints $\leftarrow$ possible endpoints **if** $D_{v,s} < \infty$ **else** possible endpoints $\cap \{x \mid D_{v,x} < \infty\}$
    INDUCING-PATHS-DFS(D,E,B,s,v,v-endpoints,path+v)

In the case of inducing paths, we use a DFS starting from each vertex. Once started from vertex s, the DFS routine checks for all possible inducing paths that terminate in s. For efficiency, a set of all possible endpoints of the path is held. Once this set is empty, the DFS search is terminated, and no further exploration is performed. The set is updated using the distance matrix calculated on the directed edges. If we consider a vertex v, s must be either its ancestor (meaning that possible endpoints for v remain unchanged) or the second inducing path endpoint is among the points that are reachable from v (meaning that the possible endpoints for v are replaced with their intersection with

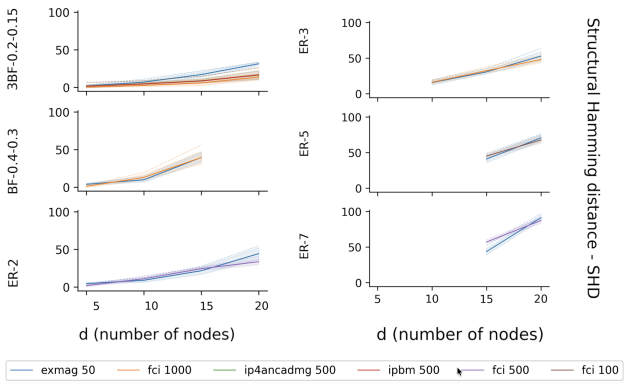


Figure 2: SHD values (in the vertical axis) for different settings of d (in the horizontal axis) and different choices of the graph. Standard deviations are depicted as the blurred regions, and dashed lines are the maximum values. For each method, the number provided is the number of samples n, and the plot shows primarily the best results over the choice of n, illustrating that ExMAG is able to provide competitive results with fewer samples than other methods. See supplementary materials for results on more datasets and error information.

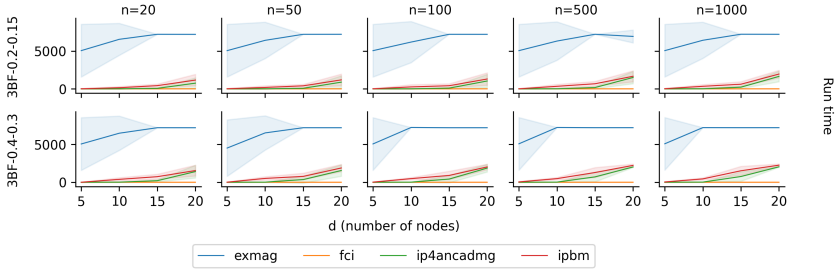


Figure 3: Run times of the compared algorithms in seconds.

the set of all points reachable from v). See Algorithm 2 for a simplified pseudocode that illustrates the idea, while leaving out some borderline cases and pruning conditions.

Once having the bidirected edges in the inducing paths and almost directed cycles, we need to trace back the Floyd-Warshall distance matrix to find all directed edges that form the cycle or the ancestor relationship. This is done using calls to the function TRACE-DISTANCE-MATRIX, which can be found in the Supplementary materials.

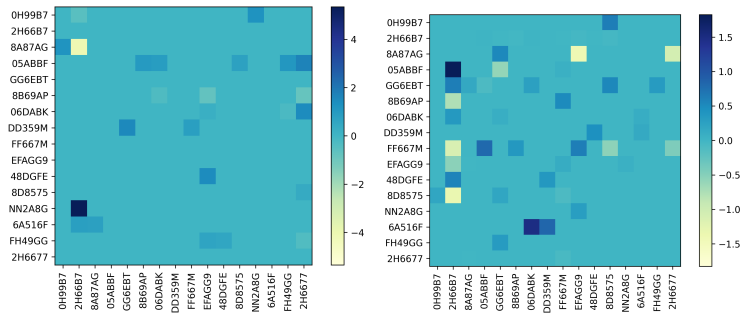


Figure 4: Heatmap of weight matrix W_D (left) and bidirected weight matrix W_B (right) on the financial dataset.

If directed cycles, almost directed cycles, or inducing paths are found, the algorithm applies lazy constraints in Acyclic Constraint and in Inducing-Paths Constraint. Note that removing one directed edge between two vertices where multiple paths exist is not a sufficient condition for the graph to become a MAG; however, this procedure can be repeated iteratively. If no inducing paths or almost directed cycles are found, we know that the program converged to the optimum, and we have a maximal ancestral graph, which minimizes equation 3.

6 Experimental Evaluation

Used Datasets We tested the ExMAG algorithm on both synthetic and real-world datasets. The first synthetic data set is based on the *Erdös-Rényi model* (ER) (Erdős & Rényi, 1959; Zheng et al., 2018), in which the ground truth graph is randomly selected from all graphs with d vertices and m edges (parameter of the experiment, for example, dataset ER-2 contains 2 edges per variable, that is, $m = 2 \cdot d$). The weights of the graph are randomly sampled from the set $(-2.0, -0.5) \cup (0.5, 2.0)$.

Once the ground truth model is created, the training data are generated using the structural model equation. Then, $20\,\%$ of variables are treated as latent variables and hidden from the training data. The respective columns and rows from the ground truth weight matrix W_D have also been removed. For fair comparison, F matrix is not provided to ExMAG.

The second dataset uses randomly generated *bow-free* (BF) graphs, based on (Bhattacharya et al., 2021). A bow-free graph is a graph such that for no pair of vertices i, j, $i \to j$ and at the same time $i \leftrightarrow j$. The BF graph generation process has two parameters: the probability of a directed edge and the probability of a bidirected edge. The generation process is as follows. First, a bow-free graph with the given edge probabilities is generated randomly. Then the weights of the sampled graphs are randomly sampled from the set $(-2.0, -0.5) \cup (0.5, 2.0)$. As this work focuses on MAGs and not all BF graphs are MAGs, we filter out all BF graphs that are not MAGs from the test data.

The third synthetic dataset, 3BF, is a modified version of BF. We generate the ground truth graph in the same way as in the BF dataset and then modify it. Specifically, we identify each vertex with a degree greater than three, and randomly remove edges until the vertex has a degree of at most three.

The adjacency matrix of the directed edges defines the weights of the structural equation model. Then the data samples are generated using the structural equation, where the noise is sampled from a multivariate Gaussian distribution with a covariance matrix equal to the adjacency matrix of bidirected edges generated in the previous step.

The fourth dataset uses real-world data from the *financial* sector. Ballester et al. (2023) work with systemic credit risk, one of the most important concerns within the financial system, using dynamic Bayesian networks. The data show that transport and manufacturing companies are likely to transfer risk to other sectors, while banks and the energy sector are likely to be influenced by the risks from other sectors. The data from (Ballester et al., 2023) contain a 10 time series capturing the spreads of 10 European credit default swaps (CDS), and further six time series are added from (Rytíř et al., 2024).

We set the matrix F to encode for no direct causal relationship between any two companies from different sectors. The banking sector includes 48DGFE, 05ABBF, 8B69AP, 06DABK, EFAGG9, 2H6677, FH49GG, and 8D8575. The insurance sector includes GG6EBT, DD359M, and FF667M. And lastly, the transportation sector and manufacturing include 0H99B7, 2H66B7, 8A87AG, NN2A8G, and 6A516F.

Evaluation Criteria Suppose that a tested algorithm produced weight matrix $\hat{W}$. Such a matrix can contain nearly zero weights. For such reasons, thresholding is done, keeping only edges with a weight greater than or equal to δ. In cases when the ground-truth weight matrix W is known, the best solution (in terms of structural Hamming distance, see below) is kept over those defined by different threshold δ values. In the evaluation, we use the *structural Hamming distance (SHD)*. This distance is the sum of contributions over all pairs of variables in the graph. For two variables i, j, let $GT \in \{\to, \leftarrow, \leftrightarrow, \emptyset\}$ be the edge type in the ground truth graph and $PR \in \{\to, \leftarrow, \leftrightarrow, \emptyset\}$ be edge type in the predicted graph. Then the contribution of i, j pair to SHD is $r_{ij} = 0$ if $GT = PR$, 0.5 if $GT \neq PR \wedge GT \neq \emptyset \wedge PR \neq \emptyset$, and 1 otherwise. Other measured criteria include *runtime* and *F1-score*, i.e., the harmonic mean of precision and recall.

Experiment Setting In the experiments, we show the results of ExMAG. In the case of synthetic datasets, we generated random graphs with the number of vertices $d \in \{5, 10, 15, 20, 25\}$. The number of samples was $n \in \{20, 50, 100, 500, 1000\}$, and for the ground-truth graph, the edge-to-vertex ratio was in $\{2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20\}$. All tested algorithms were run 10 times, each time on synthetic data generated using a random generator initialized with a different seed. The results were then averaged. We compared our method with the FCI algorithm (Spirtes et al., 1995), ip4ancadmg (Chen et al., 2021), and ipbm (Dash et al., 2025). We set regularization coefficient λ to 1.0. We ran experiments on a computing cluster with AMD EPYC 7543 CPUs, and each job had allocated two cores and 64GB RAM. Time limit was 7,200 seconds for ExMAG and 1,800 seconds for other methods. The competing methods typically finish well within this limit; it was set up arbitrarily to have a mechanism to terminate calls that were non-responsive. The total CPU time needed for experiments in this paper was around one month.

Experimental Results The SHD results are shown in Figure 2 and in the supplementary materials for additional datasets. The plots show a comparison of SHD values for ExMAG on the synthetic datasets. As can be seen, the structural Hamming distance grows with the number of variables.

In the case of ExMAG, a counterintuitive phenomenon happens on some of the datasets - the results get worse with the growing number of samples. We attribute this partly to the interaction between the solver and the resulting MIQP - with the growing number of samples, the MIQP has a larger objective function in equation 3, resulting in slower evaluation, and thus it is more difficult for the optimizer to reach the global optimum. A simple countermeasure lies in downsampling the input data. To provide a fair comparison, Figure 2 contains the results for the number of samples that gave the smallest SHD.

Since both ipbm and ip4ancadmg have a preprocessing step that depends exponentially on the maximum in-degree of the underlying ground truth graph, we tested these two algorithms only on the 3BF datasets, where the in-degree is bounded by three. Fig. 2 illustrates that to provide comparable results, ExMAG requires fewer samples than its competitors. The run times of the evaluated algorithms are shown in Figure 3. For additional results (incl. the F1-score), please see the supplementary materials.

The results on the real-world dataset can be seen in Figure 4. Contrary to the original expectations, the highest risk importer is company 2H66B7, which stands for Lufthansa. The second highest risk importer is 2H6677, i.e., the Deutsche Bank, which is an expected result.

7 Conclusion and Limitations

Learning of Bayesian networks has received considerable attention as a means of causal learning. With a few exceptions, the research has not considered confounding explicitly. Our method, ExMAG, estimates a maximally ancestral graph, capturing confounding and causal relationships using bidirected and directed edges in a mixed graph. The method provides state-of-the-art statistical performance.

As with many other methods for causal learning, the scalability of the method may leave space for improvement. Although the branch-and-cut algorithm runs in time that is exponential in the number of variables in the worst case, Figure 3 illustrates that our algorithm is able to produce results comparable to SOTA in the time limit. Note that the reported runtime of ExMAG may overestimate the time required to obtain high-quality solutions, as the MIQP solver is allowed to use the full time budget to certify optimality. In practice, near-optimal solutions are often found much earlier, with additional time spent on proving optimality. One could improve upon the run-time further by introducing additional cutting planes and more elaborate data structures for the separation of Acyclic Constraint and Inducing-Paths Constraint, perhaps drawing inspiration from solvers (Cook et al., 2011) for the travelling salesman problem.

In terms of future work, exploring the predictive power of forecasting using variants of dynamic Bayesian networks with confounding considerations seems prominent. Although it seems clear that marginalization is hard even in dynamic Bayesian networks, and thus the computational complexity may be high, statistical performance is likely to improve, when confounding is considered.

AI USE STATEMENT

During the writing of this paper, Large Language Models were used in the following ways:

- for fixing typos in the paper and to search for alternative formulations (e.g., Grammarly, Writeful, ChatGPT, and similar);
- to search for related works (e.g., ChatGPT, Gemini used in Google search, and similar);
- for generating straightforward parts of the code, especially data manipulation (IDE integrated AI-based code completion, ChatGPT, and similar), and simple experiments;
- full pseudocode writeup (see supplementary materials);
- generation of Figure 1.

LLMs contributed below the level of a contributing author; they helped with straightforward and repetitive tasks. They were not used to develop the method, to implement the program in GUROBI, nor to generate large portions of the text in the paper.

ETHICS STATEMENT

No ethics statement is required.

REPRODUCIBILITY STATEMENT

The attached supplementary materials contain the source code needed to reproduce the method. Should the paper be accepted, we will provide a link to the repository on GitHub.

REFERENCES

Ballester, L., López, J., and Pavía, J. M. European systemic credit risk transmission using bayesian networks. *Research in International Business and Finance*, 65:101914, 2023. ISSN 0275-5319. doi: https://doi.org/10.1016/j.ribaf.2023.101914. URL https://www.sciencedirect.com/science/article/pii/S0275531923000405.

Barocas, S., Hardt, M., and Narayanan, A. *Fairness and Machine Learning*. fairmlbook.org, 2019. Available at https://fairmlbook.org/.

Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., and Mahajan, A. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013. doi: 10.1017/S0962492913000032.

Bhattacharya, R., Nagarajan, T., Malinsky, D., and Shpitser, I. Differentiable causal discovery under unmeasured confounding. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130, pp. 2314–2322. PMLR, 2021.

Bickel, P. J., Hammel, E. A., and O'Connell, J. W. Sex bias in graduate admissions: Data from berkeley. *Statistics and public policy*, pp. 113–130, 1977.

Bollen, K. *Structural Equations With Latent Variables*, volume 210. John Wiley & Sons, 1989.

Bonami, P., Kilinç, M., and Linderoth, J. Algorithms and software for convex mixed integer nonlinear programs. In *Mixed integer nonlinear programming*, pp. 1–39. Springer, 2011.

Bühlmann, P. Invariance, causality and robustness. *Statistical Science*, 35(3):404–426, 2020.

Bühlmann, P. and Ćevid, D. Deconfounding and causal regularisation for stability and external validity. *International Statistical Review*, 88:S114–S134, 2020.

Bühlmann, P. and Geer, S. *Statistics for High-Dimensional Data: Method, Theory and Applications*. 01 2011. ISBN 978-3-642-20191-2. doi: 10.1007/978-3-642-20192-9.

Chen, R., Dash, S., and Gao, T. Integer programming for causal structure learning in the presence of latent variables. In *International Conference on Machine Learning*, pp. 1550–1560. PMLR, 2021.

Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., Newey, W., and Robins, J. *Double/debiased machine learning for treatment and structural parameters*. Oxford University Press Oxford, UK, 2018.

Claassen, T. and Bucur, I. G. Greedy equivalence search in the presence of latent confounders. In Cussens, J. and Zhang, K. (eds.), *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pp. 443–452. PMLR, 01–05 Aug 2022. URL https://proceedings.mlr.press/v180/claassen22a.html.

Conforti, M., Cornuéjols, G., and Zambelli, G. Extended formulations in combinatorial optimization. *4OR*, 8(1):1–48, 2010.

Cook, W. J., Applegate, D. L., Bixby, R. E., and Chvátal, V. *The traveling salesman problem: a computational study*. Princeton university press, 2011.

Dakin, R. J. A tree-search algorithm for mixed integer programming problems. *The Computer Journal*, 8(3):250–255, 01 1965. ISSN 0010-4620. doi: 10.1093/comjnl/8.3.250. URL https://doi.org/10.1093/comjnl/8.3.250.

Dash, S., Goncalves, J., and Gao, T. Integer programming based methods and heuristics for causal graph learning. In Li, Y., Mandt, S., Agrawal, S., and Khan, E. (eds.), *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 1360–1368. PMLR, 03–05 May 2025. URL https://proceedings.mlr.press/v258/dash25a.html.

Del Pia, A., Dey, S., and Molinaro, M. Mixed-integer quadratic programming is in np. *Mathematical Programming*, 162, 07 2014. doi: 10.1007/s10107-016-1036-0.

Erdős, P. and Rényi, A. On random graphs i. *Publ. math. debrecen*, 6(290-297):18, 1959.

Floyd, R. W. Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345, June 1962. ISSN 0001-0782. doi: 10.1145/367766.368168. URL https://doi.org/10.1145/367766.368168.

Gilad, Y. and Mizrahi-Man, O. A reanalysis of mouse encode comparative gene expression data. *F1000Research*, 4, 05 2015. doi: 10.12688/f1000research.6536.1.

Guo, Z., Ćevid, D., and Bühlmann, P. Doubly debiased lasso: High-dimensional inference under hidden confounding. *Annals of statistics*, 50(3):1320, 2022.

Hu, Z. *Causal Discovery with Ancestral Graphs*. DPhil dissertation, University of Oxford, 2023.

Hu, Z. and Evans, R. Faster algorithms for markov equivalence. In Peters, J. and Sontag, D. (eds.), *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pp. 739–748. PMLR, 03–06 Aug 2020. URL https://proceedings.mlr.press/v124/hu20a.html.

Hu, Z. and Evans, R. A fast score-based search algorithm for maximal ancestral graphs using entropy. *arXiv preprint arXiv:2402.04777*, 2024a.

Hu, Z. and Evans, R. J. Towards standard imsets for maximal ancestral graphs. *Bernoulli*, 30(3): 2026–2051, 2024b.

Imbens, G. W. Instrumental Variables: An Econometrician's Perspective. *Statistical Science*, 29(3):323 – 358, 2014. doi: 10.1214/14-STS480. URL https://doi.org/10.1214/14-STS480.

Kronqvist, J., Lundell, A., and Westerlund, T. The extended supporting hyperplane algorithm for convex mixed-integer nonlinear programming. *Journal of Global Optimization*, 64, 06 2015. doi: 10.1007/s10898-015-0322-3.

Lin, S., Lin, Y., Nery, J. R., Urich, M. A., Breschi, A., Davis, C. A., Dobin, A., Zaleski, C., Beer, M. A., Chapman, W. C., Gingeras, T. R., Ecker, J. R., and Snyder, M. P. Comparison of the transcriptional landscapes between human and mouse tissues. *Proceedings of the National Academy of Sciences*, 111(48):17224–17229, 2014. doi: 10.1073/pnas.1413624111. URL https://www.pnas.org/doi/abs/10.1073/pnas.1413624111.

Mathur, M. B. and VanderWeele, T. J. Methods to address confounding and other biases in meta-analyses: Review and recommendations. *Annual Review of Public Health*, 43(Volume 43, 2022): 19–35, 2022. ISSN 1545-2093. doi: https://doi.org/10.1146/annurev-publhealth-051920-114020.

McElreath, R. *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC, 2018.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35, 2021.

Pearl, J. Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 7, 1985.

Pearl, J. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, 2009.

Rantanen, K., Hyttinen, A., and Järvisalo, M. Maximal ancestral graph structure learning via exact search. In de Campos, C. and Maathuis, M. H. (eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 1237–1247. PMLR, 27–30 Jul 2021. URL https://proceedings.mlr.press/v161/rantanen21a.html.

Reiersøl, O. *Confluence analysis by means of instrumental sets of variables*. PhD thesis, Almqvist & Wiksell, 1945.

Richardson, T. and Spirtes, P. Ancestral graph markov models. *The Annals of Statistics*, 30(4): 962–1030, 2002.

Richardson, T. S. A factorization criterion for acyclic directed mixed graphs. *arXiv preprint arXiv:1406.6764*, 2014.

Rosenbaum, P. R. and Rubin, D. B. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Rytíř, P., Wodecki, A., and Mareček, J. Exdag: Exact learning of dags. *arXiv preprint arXiv:2406.15229*, 2024.

Schölkopf, B. and von Kügelgen, J. From statistical to causal learning. In *Proceedings of the International Congress of Mathematicians*, volume 1, 2022.

Spirtes, P., Meek, C., and Richardson, T. Causal inference in the presence of latent variables and selection bias. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, pp. 499–506, San Francisco, CA, USA, August 1995. Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-385-1.

Studeny, M. *Probabilistic Conditional Independence Structures*. Information Science and Statistics. Springer London, 2006. ISBN 9781846280832. URL https://books.google.cz/books?id=NJ4iwCMoznIC.

van Ommen, T. Efficiently deciding algebraic equivalence of bow-free acyclic path diagrams. *arXiv preprint arXiv:2406.09049*, 2024.

Wang, B. and Zhou, Q. Causal network learning with non-invertible functional relationships. *Computational Statistics & Data Analysis*, 156:107141, 2021. doi: 10.1016/j.csda.2020.107141. URL https://www.sciencedirect.com/science/article/pii/S0167947320302322.

Westerlund, T. and Pettersson, F. An extended cutting plane method for solving convex minlp problems. *Computers & Chemical Engineering*, 19:131–136, 1995. ISSN 0098-1354. doi: https://doi.org/10.1016/0098-1354(95)87027-X. URL https://www.sciencedirect.com/science/article/pii/009813549587027X. European Symposium on Computer Aided Process Engineering.

Zhang, J. Causal Reasoning with Ancestral Graphs.

Zheng, X., Aragam, B., Ravikumar, P. K., and Xing, E. P. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31, 2018.

Zhou, F., He, K., and Ni, Y. Causal discovery with heterogeneous observational data. In *Uncertainty in Artificial Intelligence*, pp. 2383–2393. PMLR, 2022.

SUPPLEMENTARY MATERIALS

A F1-SCORE RESULTS

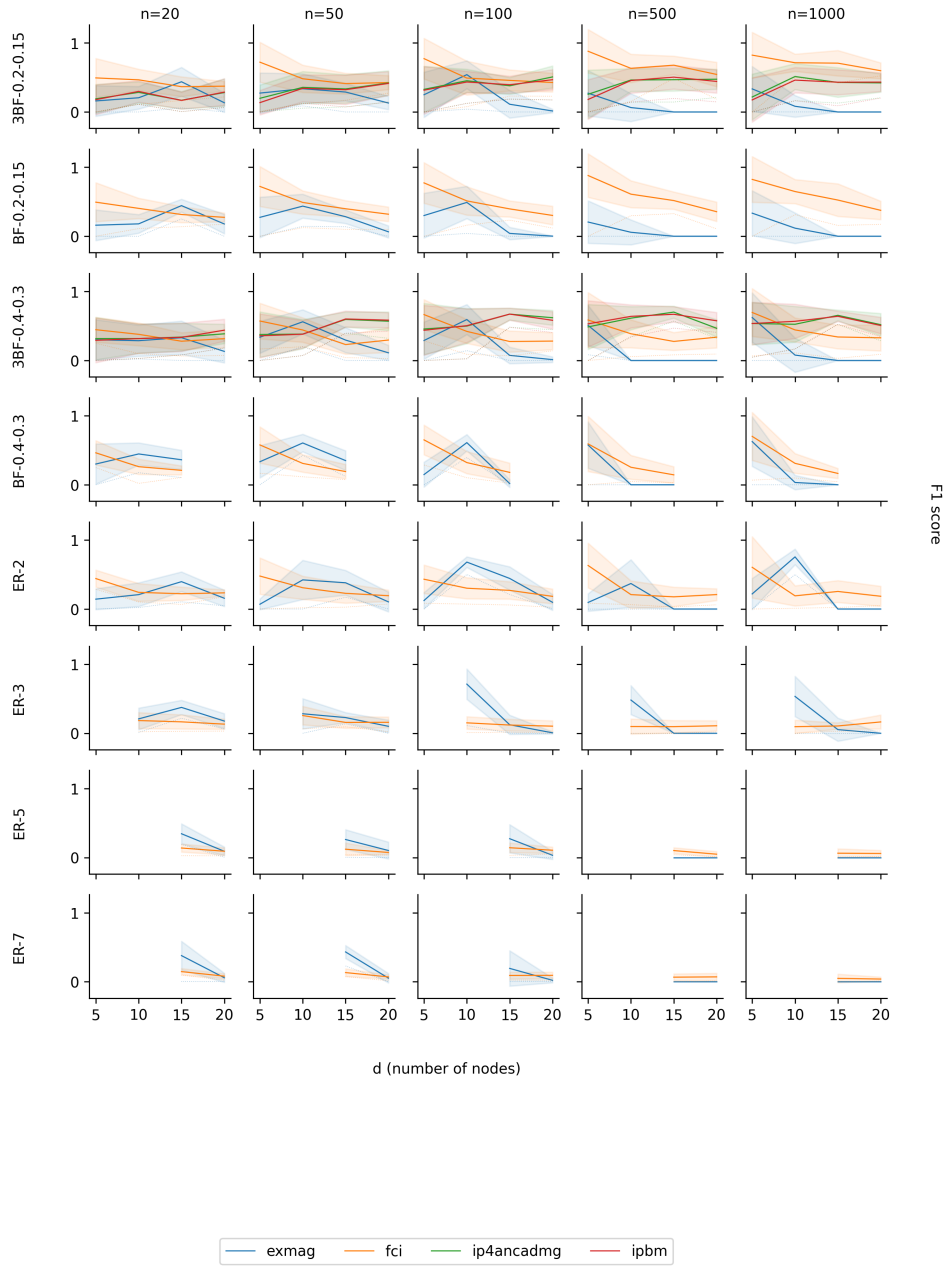


Figure 5: F1-score (in the vertical axis) for different settings of d (in the horizontal axis) and n (horizontal choice of the graph). The plots in the vertical dimension differ according to the dataset used. Standard deviations are depicted as the blurred regions, and dashed lines are the minimum values. Please note that for some of the ER plots, the graphs can be generated only for higher numbers of variables. For example, there exists no ER-5 with $d = 10$, as it would need to contain 50 edges, while the maximum is 45.

B SHD RESULTS

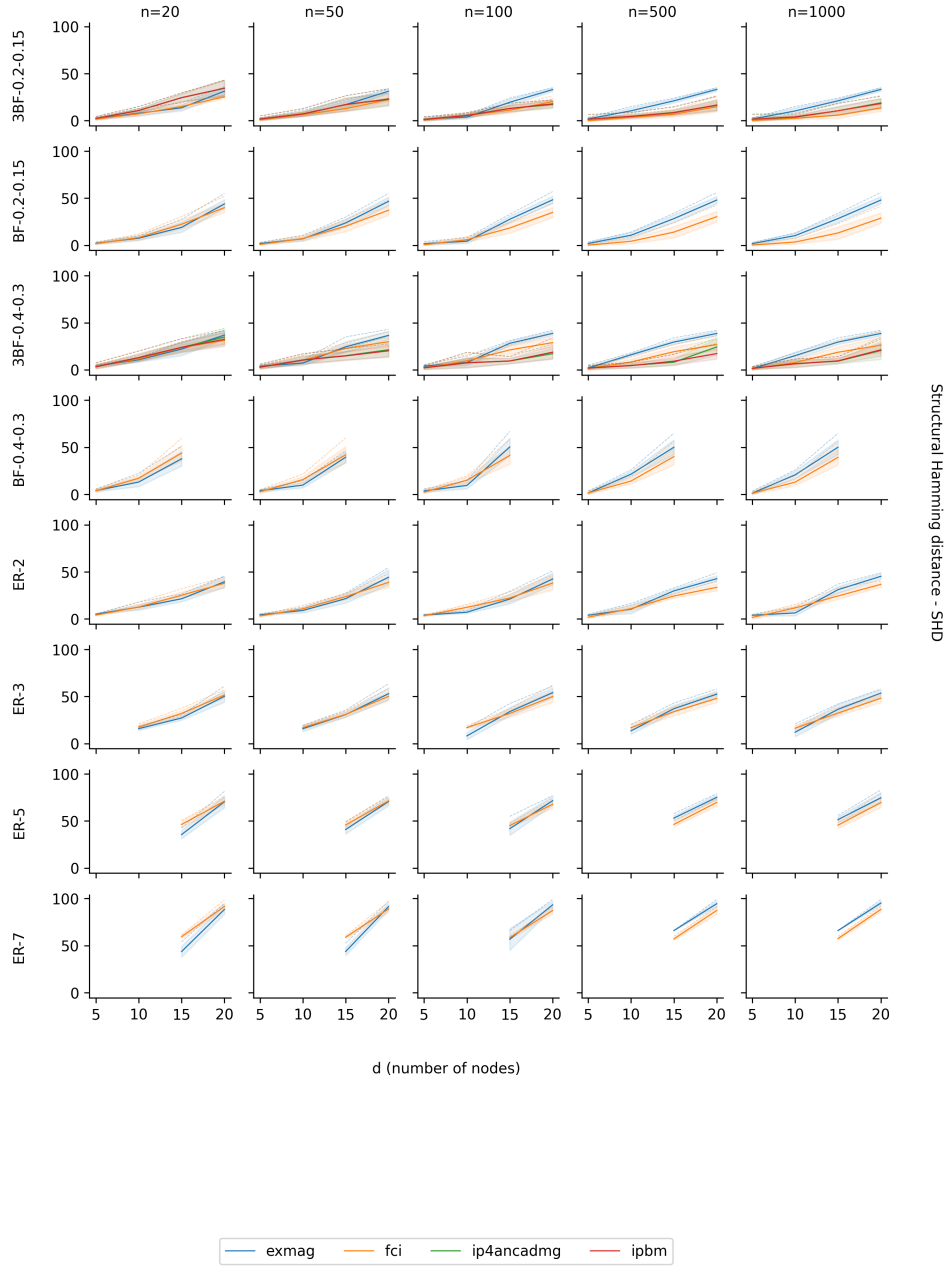


Figure 6: SHD values (in the vertical axis) for different settings of d (in the horizontal axis) and n (horizontal choice of the graph). The plots in the vertical dimension differ according to the dataset used. Standard deviations are depicted as the blurred regions, and dashed lines are the maximum values.

C USAGE OF TIME LIMIT

To illustrate how the available time budget is used, we conducted additional experiments examining the relationship between the time limit assigned to the MIQP solver in ExMAG and the resulting SHD. The results are shown in the following table:

problem	60	120	300	600	1200	1800	2700	3600	5400	7200
n50-d15-er2	27.56	27.94	28.12	28.38	26.25	26.50	23.00	22.94	23.38	22.62
n50-d15-er5	50.38	50.00	49.56	49.56	45.50	42.06	41.62	41.94	40.44	40.06

The value of MIPGAP was nearly constant, slightly under 1.0 all the time.

D COMPARISON OF THE TESTED METHODS

Method	Paradigm	Score	Output	Vars.
FCI	constraint-based (CI tests)	no	PAG	
ip4ancadmg	score-based (MIP, extended formulation)	yes	MAG	18
ipbm	score-based (LP-relaxed MIP)	yes	MAG	20
ExMAG (ours)	score-based (MIQP, branch-and-cut)	yes	weighted MAG	25

E TABLE OF NOTATION

Table 1: A Table of Notation.

Symbol	Representation
$\mathcal{G}$	Graph
$\mathcal{V}$	Vertex set of the graph
$\mathcal{E}$	Set of directed edges
$\mathcal{B}$	Set of bidirected edges
$\text{pa}_{\mathcal{G}}(v)$	Parents of vertex v in graph $\mathcal{G}$
$\text{sp}_{\mathcal{G}}(v)$	Spouses of vertex v in graph $\mathcal{G}$
$\text{an}_{\mathcal{G}}(v)$	Ancestors of vertex v in graph $\mathcal{G}$
W_D	Directed weight matrix
W_B	Bidirected weight matrix
E	Directed edge matrix
B	Bidirected edge matrix
F	Indicates that two variables have no direct causal relationship
$X_{i,j}$	Value of the j-th variable for the i-th data point
$w_{Dk,j}$	Weight of the directed edge from variable k to variable j
$w_{Bk,j}$	Weight of the bidirected edge from variable k to variable j
$e_{j,k}$	Binary variable indicating a directed edge from j to k
$b_{j,k}$	Binary variable indicating a bidirected edge between j and k
$f_{j,k}$	Optional binary variable indicating no direct causation between j and k
$d_{j,k}$	Binary variable indicating a directed edge between j and k
r_{ij}	Contribution of pair (i, j) to SHD
λ	Regularization parameter
c	Large constant for weight bounding
d	Number of variables
n	Number of data points
GT	Edge type in the ground truth graph
PR	Edge type in the predicted graph
δ	Threshold for edge weights
X	Causal variable, later data matrix
ϵ	Noise term
$\mathcal{P}$	Class of distributions
C	Conditioning set

F PSEUDOCODE

Algorithm 3 Functions that help in the separation routine.

Input: distances D defined by directed edges E, start point j, and endpoint k
Output: edges on any path from j to k

function TRACE-DISTANCE-MATRIX(D, E, j, k)
  if $D_{j,k} == \infty$ **then return**
  visited = $\{(j, k)\}$
  stack = stack with (j, k)
  edges = $\{\}$
  while stack is not empty **do**
    $u, v \leftarrow$ POP(stack)
    for all $w \in 1, 2, \ldots, d$ s. t. $D_{u,w} + D_{w,v} < \infty$ **do**
      visited $\leftarrow$ visited $\cup\{(u, w), (w, v)\}$
      if $E_{u,w}$ **then** edges $\leftarrow$ edges $\cup\{(u, w)\}$
      else if $E_{w,v}$ **then** edges $\leftarrow$ edges $\cup\{(w, v)\}$
      add $\{(u, w), (w, v)\}$ to stack
  return edges

function FOUND-INDUCING-PATH(D,E, P)
  E' = $\{\}$
  for all vertices $j \in P$ and $j \notin \{P_0, P_{|P|}\}$ **do**
    $E' = E' \cup$ TRACE-DISTANCE-MATRIX(D,E,j,P_0) $\cup$ TRACE-DISTANCE-MATRIX(D,E,j,$P_{|P|}$)
    ▷ Finds all edges on any j to P_0 ($P_{|P|}$) path
  Found inducing path formed by path P and directed edges E'

G FULL PSEUDOCODE OF EXMAG

For brevity, the main paper body leaves out many details in the pseudocode. Here, we present the full pseudocode of the method. The differences that are omitted in the main paper body are:

1. We presented the pseudocode so that it looks as if the inducing path can be only in the form of $X_1 \leftrightarrow X_2 \leftrightarrow X_3 \leftrightarrow \cdots \leftrightarrow X_k$ (plus the ancestrality relationship). This is a simplified case that follows from the definition; however, the definition assumes that each inner point is a collider, i.e., there are two edges on the path pointing toward it. Therefore, valid options are also $X_1 \rightarrow X_2$, and $X_{k-1} \leftarrow X_k$ (or any combination of the latter). The change is trivial. Firstly, we start with the first directed edge and then call the procedure as usual (see line 96 in `magseparation.py`). Secondly, we also detect the final directed edge (see line 74 in `magseparation.py`).
2. The presented pseudocode finds each path twice, firstly as $X_1 \cdots X_k$, secondly as $X_k \cdots X_1$. This is unnecessary, only one case is used (see `only_one_dir` variable parameter in `magseparation.py:inducing_path_dfs()`).
3. Some variables cannot be endpoints of an inducing path by definition - WLOG assume path from X_k to X_1 formed by directed edges, and an inducing path $X_1 \cdots X_k$. X_2 has to be an ancestor of X_k (cannot be an ancestor of X_1, as this would create an almost directed cycle), and therefore X_2 is an ancestor of X_1, which creates either an almost directed cycle or a cycle (there is either $X_1 \rightarrow X_2$ or $X1 \leftrightarrow X_2$, and at the same time a path from X_2 to X_1). Therefore, the endpoints of the inducing path cannot have a directed path between them. See filtering on line 83 of `magseparation.py`.
4. Lastly, there is a set named visited, filtering out redundant DFS calls. If there are multiple inducing paths between two vertices, only one path is enough to explore, similar to standard array marking visited vertices in normal DFS. See line 56 of `magseparation.py`. This filtering is a bit more complicated - we need to revisit some vertices, but the pruning filters state - searching for a path between two vertices, not only a vertex.

Point 1. has influence on the correctness; however, it is a marginal case that would make the pseudocode much harder to understand. The previous two points have no effect on correctness; they slightly improve runtime by a constant factor. They do not remove any inducing path, 2. means avoiding duplicates, and 3. means pruning of non-necessary candidates. Point 4. has no effect on correctness, but has the very same effect on runtime as re-visiting vertices in normal DFS for graph search, which might cause infinite looping.

Algorithm 4 ExMAG-2 (`solve`)

Input: data X, λ, loss ℓ, regulariser ρ, weight threshold τ, tabu pairs T (may be empty), cycle mode $\in \{\text{SHORTEST}, \text{ALL}\}$, time limit $t_{\max}$, weight bound M
Output: directed weights W, bidirected weights W_B, MIP gap, number L of lazy cuts

procedure ExMAG2($X, \lambda, \ell, \rho, \tau, T, mode, t_{\max}, M$)
  build model (see Sect. 4.1 equations) for the chosen ℓ and ρ
  Gurobi parameters: `LazyConstraints` $= 1$, `MIPGap` $= 0.1$, `TimeLimit` $= t_{\max}$, `NonConvex` $= 2$
  $L \leftarrow 0$
  run branch and cut; for every new incumbent candidate $(\bar{e}, \bar{b})$ call MagSeparation($\bar{e}, \bar{b}$)
  if no feasible solution was found **then**
    return $\perp$
  $W_{uv} \leftarrow w^*_{uv}$, $W_{Buv} \leftarrow \beta^*_{uv}$ for all $(u, v) \in \mathcal{P}$; zero diagonals
  set every entry of W and W_B with absolute value below τ to 0
  assert the support of W is acyclic
  return W_D, W_B, final MIP gap, L

Algorithm 5 MAG separation: Gurobi callback at `MIPSOL` (`check_for_mag`)

procedure MagSeparation($\bar{e}, \bar{b}$)
  $E \leftarrow \{(u, v) \in \mathcal{P} : \bar{e}_{uv} > 0.5\}$
  $\mathcal{C} \leftarrow$ FindCycles($E, mode$)
  for all $C \in \mathcal{C}$ **do** *// C: edge list of a directed cycle*
    add lazy cut $\sum_{(u,v)\in C} e_{uv} \leq |C| - 1$; $L \leftarrow L + 1$
  if $\mathcal{C} = \emptyset$ **then** *// acyclic: check ancestrality and maximality*
    $D \leftarrow$ FloydWarshall($\bar{e}$)
    for all $(Dir, Bi) \in$ AlmostDirectedCycles($\bar{e}, \bar{b}, D$) $\|$ InducingPathCuts($\bar{e}, \bar{b}, D$) **do**
      add lazy cut $\sum_{(u,v)\in Dir} e_{uv} + \sum_{(u,v)\in Bi} b_{uv} \leq |Dir| + |Bi| - 1$; $L \leftarrow L + 1$
  // no cut added: Gurobi accepts the candidate

Algorithm 6 Directed cycles of the candidate (`find_cycles`)

function FINDCYCLES(E, $mode$)
  $Vis \leftarrow \emptyset,\ On \leftarrow \emptyset,\ \mathcal{C} \leftarrow [\,],\ \pi \leftarrow$ empty map *// π: DFS parent*
  for all $r \in V(E)$ with $r \notin Vis$ **do**
    $S \leftarrow [r]$ *// explicit DFS stack*
    while $S \neq [\,]$ **do**
      $v \leftarrow \mathrm{top}(S)$
      if $v \notin Vis$ **then**
        $Vis \leftarrow Vis \cup \{v\},\ On \leftarrow On \cup \{v\}$
      else
        $On \leftarrow On \setminus \{v\}$; pop S
      for all $(v, x) \in E$ **do**
        if $x \notin Vis$ **then**
          push x on S; $\pi(x) \leftarrow v$
        else if $x \in On$ **then** *// back edge $v \to x$ closes a cycle*
          $c \leftarrow (x, v);\ p \leftarrow \pi(v)$
          while $p \neq x$ **do**
            append p to c; $p \leftarrow \pi(p)$
          append to $\mathcal{C}$ the edge list $[(c_{k+1}, c_k)]_{k=1}^{|c|-1} \,\|\, [(c_1, c_{|c|})]$
  if $mode =$ SHORTEST **then**
    return $[\,C^*\,]$ with C^* a shortest element of $\mathcal{C}$ (or $[\,]$ if $\mathcal{C} = [\,]$)
  return $\mathcal{C}$ *// $mode =$ ALL*

Algorithm 7 Reachability and edges on directed walks (`floyd_warshall`, `trace_f_w`)

function FLOYDWARSHALL(A)
  $D_{uv} \leftarrow 0$ if $u = v,\ 1$ if $A_{uv} > 0.5,\ \infty$ otherwise
  for $k \in V, i \in V, j \in V$ **do** *// k outermost*
    $D_{ij} \leftarrow \min(D_{ij},\ D_{ik} + D_{kj})$
  return D

function TRACE-DISTANCE-MATRIX(D, A, u, v) *// directed edges lying on directed walks $u \rightsquigarrow v$*
  if $D_{uv} = \infty$ **then**
    return $[\,]$
  $F \leftarrow [(u, v)]$ if $A_{uv} = 1$, else $F \leftarrow [\,]$
  $Mark \leftarrow \{(u, v)\},\ Q \leftarrow [(u, v)]$
  while $Q \neq [\,]$ **do**
    pop (i, j) from Q
    for all $k \in V$ with $D_{ik} + D_{kj} < \infty$ **do**
      for all $(s, t) \in \big((i, k), (k, j)\big)$ with $(s, t) \notin Mark$ **do**
        $Mark \leftarrow Mark \cup \{(s, t)\}$; append (s, t) to F if $A_{st} = 1$; push (s, t) on Q
  return F *// $F = \{(s, t) : A_{st} = 1,\ u \rightsquigarrow s,\ t \rightsquigarrow v\}$*

Algorithm 8 Almost directed cycles (`check_for_almost_directed_cycles`)

function ALMOSTDIRECTEDCYCLES($\bar{e}, \bar{b}, D$)
  $\mathcal{R} \leftarrow [\,]$
  for all $(u, v) \in \mathcal{P}$ with $\bar{b}_{uv} \neq 0$ and $D_{uv} < \infty$ **do** *// $u \leftrightarrow v$ and $u \rightsquigarrow v$*
    append $\big($TRACE-DISTANCE-MATRIX$(D, \bar{e}, u, v),\ [(u, v)]\big)$ to $\mathcal{R}$
  return $\mathcal{R}$

Algorithm 9 Inducing paths of the candidate (`inducing_paths`, `inducing_path_dfs`)

function INDUCINGPATHS($D, \bar{b}$) *// default mode: stop once $|\mathcal{P}| > 1$*
  $\mathcal{T} \leftarrow \{t \in V : D_{st} < \infty \text{ for some } s \neq t\}$ *// vertices with a proper ancestor*
  $\mathcal{P} \leftarrow [\,]$
  for all $v \in \mathcal{T}$ **do**
    $vis(x) \leftarrow \emptyset$ for all $x \in V$ *// endpoints already searched from x*
    $\mathcal{E} \leftarrow \{t \in \mathcal{T} : t < v\}$ *// each path in one direction only*
    $\mathcal{P} \leftarrow \mathcal{P} \,\|\, \text{IPSEARCH}(v, v, \mathcal{E}, (v))$
    for all $u \in V$ with $D_{vu} = 1$ **do** *// paths starting with the directed edge $v \to u$*
      $\mathcal{P} \leftarrow \mathcal{P} \,\|\, \text{IPSEARCH}(v, u, \mathcal{E}, (v, u))$
    if $|\mathcal{P}| > 1$ **then**
      return $\mathcal{P}$
  return $\mathcal{P}$

function IPSEARCH($s, u, \mathcal{E}, P$) *// P: current path from s to u; $\mathcal{E}$: admissible endpoints*
  if $\mathcal{E} = \emptyset$ **then**
    return $[\,]$
  $\mathcal{P} \leftarrow [\,]$
  if $|P| > 2$ **then**
    $vis(u) \leftarrow vis(u) \cup \mathcal{E}$
    if $u \in \mathcal{E}$ and $u < s$ **then**
      append P to $\mathcal{P}$ *// u is an endpoint*
  for all $x \in V$ **do**
    if $\bar{b}_{ux} > 0.5$ and $x \notin P$ **then** *// extend along $u \leftrightarrow x$*
      if $D_{xs} < \infty$ **then**
        $\mathcal{E}_x \leftarrow \mathcal{E}$ *// x is an ancestor of s*
      else
        $\mathcal{E}_x \leftarrow \mathcal{E} \cap \{t : D_{xt} < \infty\}$ *// x must be an ancestor of the endpoint*
      $\mathcal{E}_x \leftarrow \mathcal{E}_x \setminus vis(x)$
      $\mathcal{P} \leftarrow \mathcal{P} \,\|\, \text{IPSEARCH}(s, x, \mathcal{E}_x, P \,\|\, (x))$
      if $|\mathcal{P}| > 1$ **then**
        return $\mathcal{P}$
    if $|P| > 2$ and $D_{xu} = 1$ and $x \in \mathcal{E}$ and $x < s$ **then** *// endpoint x with last edge $x \to u$*
      append P to $\mathcal{P}$
  return $\mathcal{P}$

Algorithm 10 Inducing-path cuts (`check_for_inducing_path`)

function INDUCINGPATHCUTS($\bar{e}, \bar{b}, D$)
  $\mathcal{R} \leftarrow [\,]$
  for all $P = (p_0, \ldots, p_m) \in \text{INDUCINGPATHS}(D, \bar{b})$ **do**
    $s \leftarrow p_0,\ t \leftarrow p_m$
    $Bi \leftarrow [(p_k, p_{k+1})]_{k=1}^{m-2},\ Dir \leftarrow [\,]$ *// inner edges are bidirected*
    append (p_0, p_1) to Dir if $\bar{e}_{p_0 p_1} = 1$, otherwise to Bi
    append (p_m, p_{m-1}) to Dir if $\bar{e}_{p_m p_{m-1}} = 1$, otherwise to Bi
    for all $k = 1, \ldots, m-1$ **do** *// inner vertices are ancestors of s or t*
      $Dir \leftarrow Dir \,\|\, \text{TRACE-DISTANCE-MATRIX}(D, \bar{e}, p_k, s) \,\|\, \text{TRACE-DISTANCE-MATRIX}(D, \bar{e}, p_k, t)$
    append (Dir, Bi) to $\mathcal{R}$
  return $\mathcal{R}$

H DERIVATION OF THE OPTIMIZATION CRITERION

We base our cost function on work (Bhattacharya et al., 2021). The paper assumes structural equation model

$$X \leftarrow XW_D + \varepsilon. \tag{8}$$

where the noise term ε contains correlations between variables, represented by covariance matrix $\beta = \mathbb{E}(\varepsilon\varepsilon^T)$. Matrix β encodes the confounding. With the goal of optimizing the ABIC criterion, Bhattacharya et al. (2021) then minimizes (up to regularization) term

$$LS = \sum_{j=1}^{d} \left\| X_{\cdot,j} - XW_{D\cdot,j} - Z^{(j)}\beta_{\cdot,j} \right\|^2, \tag{9}$$

where $Z^{(j)}_{\cdot,j} = 0$ and for $k \neq j$

$$Z^{(j)}_{i,k} = \sum_{l=1,l\neq j}^{d} \varepsilon_{i,l}\beta^{-T}_{l,k}. \tag{10}$$

The error term is calculated as

$$\varepsilon_{i,j} = X_{i,j} - X_{i,\cdot}W_{D\cdot,j}. \tag{11}$$

The notation is adjusted, and the terms are rearranged to match the main paper body. For any matrix, for example X, $X_{i,j}$ denotes entry of X on i-th row, and j-th column, $X_{i,\cdot}$ denotes i-th row, and $X_{\cdot,j}$ denotes j-th column. Please note that Bhattacharya et al. (2021) has a typo in Algorithm 2, where X and $W_{D\cdot,j}$ are reversed (resulting in undefined matrix product).

Here, we show the derivation of equation 3 from the least squares error equation 9. As a by-product, we show interpretation of matrix W_B. First, we expand the L2-norm in equation 9, to obtain

$$LS = \sum_{j=1}^{d}\sum_{i=1}^{n} \left\| X_{i,j} - X_{i,\cdot}W_{D\cdot,j} - Z^{(j)}_{i,\cdot}\beta_{\cdot,j} \right\|^2. \tag{12}$$

As both sums above are independent, we can swap their order. We expand matrix multiplication of $Z^{(j)}_{i,\cdot}\beta_{\cdot,j}$, keeping in mind, that $Z^{(j)}_{\cdot,j} = 0$.

$$LS = \sum_{i=1}^{n}\sum_{j=1}^{d} \left\| X_{i,j} - X_{i,\cdot}W_{D\cdot,j} - \sum_{k=1,k\neq j}^{d} Z^{(j)}_{i,k}\beta_{k,j} \right\|^2. \tag{13}$$

Further, we plug equation 10 into the formula

$$LS = \sum_{i=1}^{n}\sum_{j=1}^{d} \left\| X_{i,j} - X_{i,\cdot}W_{D\cdot,j} - \sum_{k=1,k\neq j}^{d} \left[\sum_{l=1,l\neq j}^{d} \varepsilon_{i,l}\beta^{-T}_{l,k} \right] \beta_{k,j} \right\|^2. \tag{14}$$

As indices k and l are independent, we can rearrange the terms into

$$LS = \sum_{i=1}^{n}\sum_{j=1}^{d} \left\| X_{i,j} - X_{i,\cdot}W_{D\cdot,j} - \sum_{k=1,k\neq j}^{d}\sum_{l=1,l\neq j}^{d} \left[\varepsilon_{i,l}\beta^{-T}_{l,k}\beta_{k,j} \right] \right\|^2. \tag{15}$$

Further, we can swap the order of summation, and factor out $\varepsilon_{i,l}$ to obtain

$$LS = \sum_{i=1}^{n}\sum_{j=1}^{d} \left\| X_{i,j} - X_{i,\cdot}W_{D\cdot,j} - \sum_{l=1,l\neq j}^{d} \varepsilon_{i,l} \left[\sum_{k=1,k\neq j}^{d} \beta^{-T}_{l,k}\beta_{k,j} \right] \right\|^2. \tag{16}$$

Let us denote W_B matrix obtained by the multiplication of matrices $\beta^{-T}\beta$ which uses the summation above, i.e., let

$$W_{Bl,j} = \sum_{k=1,k\neq j}^{d} \beta^{-T}_{l,k}\beta_{k,j}. \tag{17}$$

Then, using notation from equation 2, we obtain

$$LS = \sum_{i=1}^{n} \sum_{j=1}^{d} \left\| R_{i,j} - \sum_{l=1, l\neq j}^{d} R_{i,l} W_{Bl,j} \right\|^2, \tag{18}$$

which is equivalent to the MIQP formulation in equation 3.

Now, we provide interpretation of the weight matrix W_B. Denote $\mathrm{diag}(\beta)$ the matrix containing diagonal entries of β, i.e., the diagonal matrix of marginal variances. As summation in equation 17 ignores fields, where $k = j$, i.e., we can replace $\beta_{k,j}$ with 0 for $k = j$. As a result W_B can be written in matrix form as

$$W_B = \beta^{-T}(\beta - \mathrm{diag}(\beta)). \tag{19}$$

This can be rearranged into (I stands for identity matrix)

$$W_B = I - \beta^{-T} \mathrm{diag}(\beta). \tag{20}$$

For non-diagonal entry of W_B, its value is zero only if the respective field in the precision matrix is zero. In other words, $W_{Bl,j} = 0$ if and only if l and j are conditionally independent given all variables but l and j.

I Residual Assumptions And Statistical Significance

There exists a fundamental premise in structural equation modeling that the residuals—representing unexplained variation—are asymptotically unbiased, meaning they are independent of both observed and latent variables, and follow a zero-mean distribution. This assumption plays a critical role in ensuring that learned causal relationships are not distorted by hidden confounders or systematic error. The ExMAG framework embraces this principle by design, explicitly modeling residual independence as a safeguard against spurious causal edges. Just as fairness-aware systems aim to isolate structural patterns from social bias (Barocas et al., 2019), ExMAG works to separate signal from statistical noise. The result is a model capable of learning causal mechanisms that are not only mathematically sound but also resilient across different subpopulations, forming a foundation of causal inference.

Causal discovery systems, like decision-making algorithms in high-stakes domains, must operate effectively across structurally diverse populations. This paper uses real-world financial data spanning multiple sectors—banking, insurance, manufacturing, and transportation—each exhibiting distinct systemic exposures. These domains can be viewed as a *privileged setting* where data availability and quality are high, yet subgroup heterogeneity remains significant. In such contexts, ExMAG successfully identifies dominant risk propagation patterns, even when feature distributions vary across industries. This mirrors broader challenges in fairness: the need to perform robustly across populations with unequal baseline conditions Mehrabi et al. (2021). The model's consistent recovery of risk links—illustrated in Figure 4—not only affirms its structural fidelity but also its capacity to generalize without group-specific tuning.

Understanding the statistical reliability of a model's output requires more than average performance—it demands insight into variance. To that end, the authors conduct 10 independent trials for each configuration, reporting both mean and standard deviation for key metrics such as SHD and F1-score. The inclusion of error bars in Figures 2 and 5 provides a visual representation of variability, revealing not just how well the model performs, but how consistently. In contrast to baseline methods with large fluctuations, ExMAG demonstrates narrow error margins, underscoring its stability in the face of stochastic elements like data partitioning and initialization.

J Proof of Termination

Theorem 1 (Finite termination)**.** *ExMAG halts after finitely many operations. The number of separation-generated cuts is at most*

$$N(d) \;\le\; \underbrace{2^{d(d-1)}}_{\mathcal{C}_{\mathrm{dc}}} + \underbrace{\tbinom{d}{2}\, 2^{d(d-1)}}_{\mathcal{C}_{\mathrm{adc}}} + \underbrace{2^{\binom{d}{2}}\, 2^{d(d-1)}}_{\mathcal{C}_{\mathrm{ip}}} \;=\; O\big(2^{2d(d-1)}\big),$$

exponential in d.

Proof:

Step 1 — the cut universe is finite. Each cut can be indexed by a finite index vector on the $\leq d(d-1)$ directed edges and $\binom{d}{2}$ bidirected edges: a directed cycle is an edge subset; an almost-directed-cycle cut is a (bidirected edge, directed-edge subset) pair; an inducing-path cut is a (bidirected-edge subset, directed-edge subset) pair. Counting these gives the bound above, so the number of cuts is $N(d) < \infty$.

Step 2 — Every separation-generated cut is new. Suppose that the separation routine is invoked on a solution (E, B) and returns a cut γ. Because (E, B) is feasible for the *current* model, it satisfies every previously added cut. But γ is violated by (E, B). Hence, γ could not have been found before. So each call to the separation routine adds at least one new cut to the existing program.

Step 3 — finitely many separation events. By Steps 1–2, at most $N(d)$ cuts are ever added, so at most $N(d)$ calls to the separation routine are made. Also, an inducing path P reappearing after the solver merely drops one directed edge of E', now supported by a different set $E'' \neq E'$ can recur only finitely often, as each recurrence is a distinct pair and the cardinality of the set of edges E', E'' strictly decreases. Thus, the incompleteness of a *single* inducing-path cut does not threaten termination.

Step 4 — finite work between cuts. With the constraint set held fixed between two consecutive cut additions, the algorithm solves a quadratically constrained MIQP with $O(d^2)$ binary variables, which is finite time. The separation routine itself costs $O(d^3)$.

The run, therefore, consists of at most $N(d) + 1$ finite branch-and-bound phases, hence is finite overall. □

While the algorithm is guaranteed to terminate, the overall runtime can be exponential. For example, the inducing path definition includes ancestral relationships defined by one or more edges. Imagine a chain of variables sorted in topological order that defines this ancestral relationship (as part of some inducing path which includes X_1 and X_k), such as $X_1 \to X_2 \to \cdots \to X_k$. Imagine that this graph is a complete DAG (with all edges going right), which is not against the other constraints. In each iteration, only one edge connecting X_1 to X_k is forbidden. Depending on the weights, in the worst case, we have to forbid at least $k(k-1)/2 - (k-1)$ edges to break the connection between X_1 and X_k, which is quadratic with respect to one single inducing path. The main advantage of the current formulation is the avoidance of the exponential preprocessing step of our competitors (ipbm and ip4ancadmg) by allowing us to explore only the constraints in a lazy manner - if no inducing path is found, this exponential step is avoided entirely

K Brief Introduction To Mixed Integer Quadratic Programming

Let us also provide a short introduction to mixed-integer quadratic programming. An optimization problem is called a mixed-integer quadratically constrained quadratic program (MIQCQP) if it is of the form

$$\min_{x \in \mathbb{R}^n} \quad x^T Q x + q^T x, \tag{21}$$

$$\text{s.t. } x^T Q_i x + q_i^T x \leq a_i, \tag{22}$$

$$Ax \leq b, \tag{23}$$

$$x \in \mathbb{R}^{n-r} \times \mathbb{Z}^r \tag{24}$$

where $Q, Q_i \in \mathbb{R}^{n \times n}$, $q, q_i \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $a \in \mathbb{R}^k$, $b \in \mathbb{R}^m$ and $m, n, k, r \in \mathbb{N}$. Equation 21 is called the cost or loss function, equation 22 represents the quadratic constraints, equation 23 are the linear constraints, and equation 24 enforces the integrality constraints for the last r components of the vector of decision variables x.

Mixed-integer quadratic programs have been shown to be NP-hard (Del Pia et al., 2014), which often leads to an exhaustive demand for computational resources. The algorithms used to solve MIQP

are typically branch-and-bound or cutting plane Dakin (1965); Bonami et al. (2011); Westerlund & Pettersson (1995); Kronqvist et al. (2015). Both of these algorithmic treatments are often employed together, often with the addition of a presolving step, the use of heuristics, and parallelism. The aforementioned allows many modern solvers to solve even large problems despite the NP-hardness. Some of these solvers are open source (like SCIP and GLPK), and others are commercial (GUROBI and CPLEX). The powerful infrastructure present in these solvers can be made use of together with additional problem-specific modifications to deliver high-quality solutions.

Due to the exhaustive nature of the algorithms mentioned in the previous paragraph, global convergence is guaranteed Belotti et al. (2013). Furthermore, convergence to the global solution may be tracked and the error estimated by computing the dual problem of (21–24). The dual of the problem is then used to compute the so-called MIPGAP as follows

$$\text{MIPGAP} = \frac{|J(x^*) - J_{\text{dual}}(y^*)|}{|J(x^*)|}, \tag{25}$$

where x^* and y^* are the current best solutions of the primal and dual problems, respectively, and J and J^* are the cost functions of the primal and dual problems, respectively. The MIPGAP ensures that we can assess the quality of the minimization during solution time and terminate the computation when the result is good enough (small enough MIPGAP). Furthermore, if the gap reaches 0 at any point, we are sure that the current solution is a global optimum.