CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Sample-efficient machine learning with auxiliary information

Leveraging Privileged Information for Efficient Learning in Gaussian DAG Models

Master's thesis in Computer science and engineering

Xinxin Tan

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Sample-efficient machine learning with auxiliary information

Leveraging Privileged Information for Efficient Learning in Gaussian DAG Models

Xinxin Tan

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Sample-efficient machine learning with auxiliary information
Leveraging Privileged Information for Efficient Learning in Gaussian DAG Models
Xinxin Tan

Supervisor: Fredrik Johansson, Department
Examiner: Devdatt Dubhashi, Department

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Sample-efficient machine learning with auxiliary information
Leveraging Privileged Information for Efficient Learning in Gaussian DAG Models
Xinxin Tan
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Our thesis proposes a Learning using Privileged Mediating Information (LuPI) algorithm based on a directed Gaussian graphical model, and analyzes that LuPI outperforms the Ordinary Least Squares (OLS) model in terms of statistical properties under known causality by constructing a causal directed acyclic graph (DAG) containing mediating variables. Using the Rao-Blackwell theorem, it is shown theoretically that LuPI can efficiently decrease the mean square error (MSE) and the expected risk. In the experimental part, the improvement of LuPI over OLS is verified on a synthetic dataset under different noise levels and sample sizes, especially under high noise and small sample conditions. In addition, the experiments also investigate the impact of graph estimation bias on the performance of the algorithm, and the results show that appropriate removal of redundant edges in the causal graph can help reduce the variance, which in turn improves the overall performance of the model. Finally, the experiments based on real datasets further demonstrate the superiority of the LuPI algorithm under small sample sizes and validate its application value in complex causal data.

Keywords: Learning using Privileged Information, Directed Gaussian Graphical Model, Linear Regression, Causal Analysis.

Acknowledgements

I would like to express my appreciation to my supervisor, Fredrik, for all his support and encouragement throughout the entire process. His involvement has really made a significant difference to my work. I would also like to thank Rickard for the helpful advice and great discussions that have shaped my research. Thanks also to my examiner Devdatt and the rest of the Healthy AI lab group. I really appreciate all the feedback and insights you have shared with me. Last but not least, I'm incredibly grateful to my mum and friends for always supporting me during my time at Chalmers.

Xinxin Tan, Gothenburg, 2024-10-09

Contents

List of Figures

1

Introduction

Predicting outcomes based on known observations is a very common and classical type of tasks in the field of machine learning, especially supervised learning. Such predictions are usually made by constructing a regression model between the observed variables and the predicted outcomes[1]. However, on the basis of causal analysis, it is known that the relationship between observed variables and predicted outcomes may sometimes be an indirect causal effect occurring through one or more mediating variables rather than a direct causal effect[2].

To explain this, we first introduce some theories of causal analysis. In causal theory, the observed variable is usually defined as the variable that is directly measured or observed in an experiment or study, and the predicted outcome is the target variable that we want to predict or explain. For example, in a study of patients on a certain medication, the observed variable is whether or not the patient takes the medication, and the target variable is the patient's recovery. If the medication directly improves recovery, then the effect of the medication on recovery is called as a direct causal effect, in which the observed variable directly affects the predicted outcome. However, in the real world, the observed variable may indirectly affect the predicted outcome by influencing one or more mediating variables. Still in the patient example, the use of medication may also affect the patient's recovery by adjusting mediating variables such as blood pressure and pulse, rather than directly improving the outcome. In this case, the information in the mediating variable may be ignored if we rely only on data from the observed variable[2].

Therefore, training a model using only the observations of the dependent variable is not always the best choice, especially when there is insufficient data. Instead, sample-efficient machine learning can be achieved by using the observations of the mediator variable as auxiliary information in training, also known as $Learning with Privileged Information$.

1.1 Related work

The concept of learning using privileged information was first introduced by V. Vapnik in his paper *A new learning paradigm: Learning with privileged information* (2009) [3]. The paper discusses an advanced learning paradigm that improves learning models by providing privileged information in the training process that is not available in testing. This paradigm is inspired by elements of human teaching: the teacher

supplies additional information such as explanations, comments, comparisons, etc. to improve the student's ability, but in the actual test the student has to solve the problem independently without the help of additional information. The author developed this learning paradigm, abbreviated as LuPI, for support vector machine algorithms, called SVM+ algorithm, which introduces a correcting space based on auxiliary information in addition to the decision space of the classical SVM algorithm, where the coefficients in the decision function depend on the similarity measure of the two spaces. This new algorithm has been implemented on datasets such as biological and digital images, and also applied to time series data prediction, and in all these practices the superiority of the LuPI paradigm over the classical machine learning paradigm has been demonstrated.

Recently, an efficient learning method using time-series privileged information (LuPTS) has been proposed by Karlsson et al[4]. This new method is based on the assumption of Gaussian linear systems and Markovian time series, and uses samples of the time series observed between the baseline time and the future outcomes as privileged information to help train the linear regression model. Compared to the ordinary least squares(OLS) estimator, which is the estimator with the least mean squared error when predicting using only the baseline variables, LuPTS is theoretically proven to be unbiased like OLS when the model assumptions are met, and has smaller variance and mean squared error than OLS. Since LuPTS introduces possible bias if the assumptions are not satisfied, the authors use a generalized distillation method [5] that combines the LuPTS and OLS methods for the bias-variance trade-off. Experiments on time series data show that the generalized distillation method and the LuPTS method achieve a better performance than the classical OLS method.

Jung et al.[6] further developed time series privileged information learning and extended it to nonlinear systems. Their algorithm builds on the idea of kernel methods to estimate nonlinear relationships using a random feature map implemented as a neural network. Through the nonlinear map, a Markov chain time series is established that conforms to the assumption of a Gaussian linear system, which makes it possible to apply the theory of linear systems and to extend it to nonlinear systems. Similarly, the idea of generalized distillation has been adopted for bias-variance tradeoffs. While the current work focuses on modeling time-series data, in this project we aim to generalize the privileged information learning algorithm to a wider range of DAG models.

Given the existing focus on time series models, the practical approach and performance of learning with privileged information in more complex graphical models remain unclear. Therefore, our project aims to investigate the theory and experiments of Learning using Privileged Mediating Information (hereafter referred to as LuPMI) in the special case of directed Gaussian graphical models.

1.2 Purpose of research

The main goal of this project is to develop a generalized method based on directed Gaussian graphical models that uses privileged information to improve the perfor-

mance of classical supervised learning algorithms. This approach aims to improve the accuracy and reliability of the model compared to the classical method. Our specific goals in this project are as follows:

- **Algorithm**: Develop and implement an algorithm to improve the prediction of regression models by integrating privileged information based on directed Gaussian graphical models.
- **Theory of Effectiveness:** Theoretically analyze the proposed algorithm, focusing on its bias and variance properties, and determine its effectiveness over classical methods.
- **Evaluation:** Evaluate the performance of the algorithm on synthetic and real datasets, comparing its performance differences with classical regression methods, thus demonstrating its practical application and advantages.

1.3 Research questions

How does the performance of the proposed algorithm(LuPMI) compare to classical regression methods when applied to datasets?

Experiments on both synthetic and real data show that the LuPMI algorithm generally outperforms the OLS algorithm. On synthetic datasets with known real graphs, LuPMI consistently shows a lower relative mean square error (MSE) than OLS, especially in situations with smaller sample sizes and higher noise levels. In the real dataset, R-squared scores indicate that LuPMI provides a better model fit than OLS, confirming its effectiveness.

Furthermore, experiments on DAG estimation bias show that LuPMI's performance remains reliable even when edges are ignored or mis-estimated. While ignoring edges introduces bias, it also reduces variance and thus improves performance in high-noise conditions. In contrast, adding redundant edges increases variance but does not introduce bias, and LuPMI still matches or exceeds OLS performance. These results show the soundness and superior performance of LuPMI in a variety of experimental setups, demonstrating its potential benefits in real-world applications.

What modifications are required to extend the current LuPTS algorithm to a generalized DAG model?

The most critical change in the generalization from LuPTS to LuPMI is due to the modeling assumptions. The LuPTS algorithm is based on a Gaussian dynamical system in time series and thus has explicit causality of a single path. On the other hand, the LuPMI algorithm extends the model to an arbitrary directed acyclic graph between the prediction target and the independent variables, thus involving a more complex multi-path causal relationship. Thus, the result of the LuPMI algorithm is predicted based on the joint influence on multiple paths. In addition, the assumption of column linear independence of the dataset of a single variable in the original LuPTS algorithm is modified to the assumption of column linear independence of

the dataset of all parent nodes in the LuPMI algorithm. That's because the LuPMI algorithm performs multivariate linear regression.

What is the improvement of the LuPMI method over the current LuPTS method?

The LuPTS method is proposed in the setting of time series data that are assumed to come from a Gaussian linear system and follow the Markov chain assumption. The data at each time point depend only on the previous time point, forming a strict sequential dependency. However, the LuPMI method extends this assumption and is no longer limited to time series. Our model assumption is based on directed Gaussian graphical models, which means that the causality of the model is represented as a directed acyclic graph (DAG). In this model, a variable can depend on multiple parent nodes that are not necessarily in time order, and the state of any node is independent of other unconnected nodes. With this conditional independence assumption, the underlying causal dependencies of the data are conveniently summarized and visualized.

This extension makes LuPMI suitable not only for time series data, but also for other datasets with causal structures, such as causal inference problems in economics and medicine. It is possible to use information about observed and mediating variables in different directed graph structures to improve prediction. Therefore, the improvement of the LuPMI method over the LuPTS method is that it extends from a specific time series to a more general directed graph model that is able to deal with complex causal structure data.

Can the proposed algorithm generalize well across different types of datasets, including those with various levels of noise?

In our experiments in Chapter 4, the algorithm performs well on a variety of synthetic and real datasets and is able to effectively adapt to the noise in the data. By using privileged information, the algorithm significantly reduces the variance of the model on highly noisy datasets while keeping the bias low, thus improving the overall prediction performance. Besides, the experiment results also show that the algorithm retains robustness even in the presence of missing data, indicating that it has good generalization ability and is suitable for various application cases. In addition, the distillation method can avoid the effects of missing data to some extent because it combines the advantages of the LuPMI algorithm and the OLS algorithm, which are trained on non-missing data. This allows the algorithm to provide more reliable results than classical regression methods in practical applications, especially when dealing with complex and imperfect data. Therefore, it is reasonable to expect that the proposed algorithm can consistently provide excellent performance under various noise levels and missing data conditions.

1.4 Outline

This chapter offers a fundamental overview of our thesis project, detailing its purpose and motivations. In the next Chapter 2, we will specifically present the background needed in order to understand the theory and results of this project. Chapter 3 then contains our detailed model assumptions and mathematical formulation on which we develop our algorithm and effectiveness theory. Subsequently, the experimental results are presented in Chapter 4, encompassing both the synthetic and the real datasets. The final chapter, Chapter 5, summarizes the findings of the study and provides insights into potential avenues of research for future exploration.

2

Background

2.1 Notation

For the convenience of our illustration, there are some notations to clarify at the beginning of all theoretical foundations. First, the following notations are used to distinguish variables, observations, and data:

- Variables/nodes are denoted by the uppercase letters X, since they are corresponding in our problem.
- Observations/samples are denoted by the lowercase letters x.
- All observations for a variable X in the dataset are denoted by the bold uppercase letters $\mathbf{X}$, which can be either a matrix (for a vector variable X) or a vector (for a scalar variable X).

Second, to describe the structure of a directed acyclic graph $G = (\mathcal{V}, \mathcal{E})$, we set:

- $\mathcal{V}$ represents the set of all nodes, i.e. the set of all variables in the model, $\mathcal{V} = \{X_1, \ldots, X_n, Y\}$.
- $\mathcal{E}$ denotes the set of all edges, and $(X, Y) \in \mathcal{E}$ represents the edge from the node X to the node $Y (X \in \mathcal{V}$ and $Y \in \mathcal{V})$.
- $Pa(X)$ denotes the set of all parent nodes of the node X.
- $\mathcal{P}(X_i, X_j)$ denotes the set of all paths from node X_i to node X_j, which means every element (i.e., path) p in the set $\mathcal{P}(X_i, X_j)$ is a subset of $\mathcal{E}$, i.e., $p \in \mathcal{P}(X_i, X_j) \Leftrightarrow p \subseteq \mathcal{E}$.
- Based on the above, i_p denotes the index of the parent node of Y in the path p.

Moreover, in the context of vectors and matrices:

- X_i represents the i-th component of the vector X.
- A_{ij} denotes the j-th element of the i-th row of the matrix A.

2.2 Directed Gaussian graphical models

A graphical model is a tool for describing and dealing with complex dependencies between variables in a dataset using graphs, which can be categorized into directed and undirected graphical models. A classical case of directed graphical models is Bayesian networks, commonly used to visualize the causal relationships of variables. A simple example is shown in Figure 2.1.

The key in graphical models is to represent the joint distribution of a number of variables with dependencies based on the assumption of conditional independence[7]. Further, in Bayesian networks, it is assumed that each variable depends only on its parent node and is conditionally independent from all other nodes when its parent is observed. Therefore, the joint probability distribution of all variables in a Bayesian network can be discretized as a product of the conditional probability distributions(CPDs) of each variable.

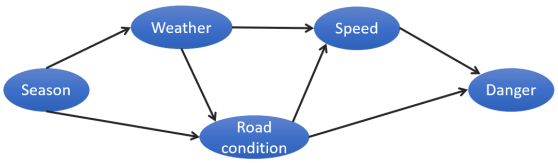


Figure 2.1: A simplified version of a Bayesian network to represent the impact of various influences on traffic, where each node represents an influence and the arrows indicate the causal relationship between them.

Directed Gaussian graphical models (directed GGMs), also referred to as Gaussian Bayesian networks, represent a specific type of model in which the relationship between random variables is linear and the noise is Gaussian. The model is based on a directed acyclic graph that represents the conditional dependency relationships. In detail, for a variable X in the directed GGM, its conditional probability distribution has the form[8]:

$$p(X \mid Pa(X)) = \mathcal{N}(X \mid \mu_X, \Sigma_X),$$

where Σ_X denotes the variance matrix. And the mean, μ_X, is related to the following:

$$\mu_X = \beta_X + \sum_{Z \in Pa(X)} \beta_Z^T Z.$$

In this context, the weight parameter corresponding to the parent variable Z of the variable X is represented by β_Z. Similarly, β_X represents the intercept. The above CPD relationship can also be expressed in noise form as follows[7]:

$$X = \beta_X + \sum_{Z \in Pa(X)} \beta_Z^T Z + \epsilon_X, \quad \epsilon_X \sim \mathcal{N}(0; \Sigma_X).$$

It implies that when all parents Z of a variable X in a Gaussian Bayesian network are observed, the expectation (i.e., the mean of the Gaussian distribution) of that variable X is linearly dependent on the observations of all parents. In addition, since the CPD of X is a multivariate Gaussian distribution, its variance matrix is explicitly

a diagonal matrix. Furthermore, when the Gaussian noise is isotropic, every value on the diagonal will be equal, i.e.,

$$\Sigma_X = I\sigma_X^2,$$

where I is the unit matrix, σ_X is the standard deviation of each component of the vector X. Overall, the Gaussian Bayesian network model with isotropic Gaussian noise is the main case considered in this thesis, and our algorithms and theories are based on this model assumption.

2.3 Learning using privileged information(LuPI)

Classical supervised learning tasks involve an input space $\mathcal{X}$ and an output space $\mathcal{Y}$. The goal of these tasks is to learn a mapping function

$$f \in \mathcal{F} \quad \text{and} \quad f : \mathcal{X} \to \mathcal{Y}$$

from the training data $\{\{x_i\}_{i=1}^n, \{y_i\}_{i=1}^n\}$, which predicts the corresponding output $Y \in \mathcal{Y}$ for any new input $X \in \mathcal{X}$. In order to measure the effectiveness of the mapping function f, the loss function $\mathcal{L}(Y, f(X))$ is introduced to compute the difference between the model's prediction $\hat{Y} = f(X)$ and the true label Y. There are many types of different loss functions, and it depends on the condition of the task to choose which function to use. In our project, the main loss function used is the :

$$\mathcal{L}(Y, \hat{Y}) = (Y - \hat{Y})^2$$

The loss function calculates the difference for each pair of $(Y, f(X))$, which can be seen as a kind of local evaluation. Furthermore, to achieve the best performance in the whole dataset, we need to synthesize the loss to all the predictions on the entire input space, which leads to the concept of expected risk(also known as expected loss[1]). It is related to the data's probability distribution $P(X, Y)$ and is defined as

$$R(f) = \mathbb{E}_{(X,Y)\sim P}[\mathcal{L}(Y, f(X))],$$

where $\mathbb{E}_{(X,Y)\sim P}$ represents the expectation with respect to the data distribution $P(X, Y)$. However, since the data distribution P is usually unknown, the expected risk is difficult to compute directly in most cases. Therefore, the empirical risk is used as an estimate of the expected risk. Assuming that the training dataset is $\{(x_i, y_i)\}_{i=1}^n$, the empirical risk is defined as

$$\begin{aligned} R_{emp}(f) &= \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}(y_i, f(x_i)) \\ &= \frac{1}{n}\mathcal{L}(\mathbf{X}, \mathbf{Y}, f), \end{aligned}$$

where $\mathbf{X} = \{x_i\}_{i=1}^n$ and $\mathbf{Y} = \{y_i\}_{i=1}^n$. So when the squared loss function is used, the empirical risk is actually equivalent to the mean square error (MSE). Given the

training dataset $D = \{\mathbf{X}, \mathbf{Y}\}$, the optimal mapping that minimizes the empirical risk can be estimated as

$$\hat{f} = \arg\min_{f \in \mathcal{F}} R_{emp}(f \mid \mathbf{X}, \mathbf{Y}).$$

However, instead of the conventional supervised learning paradigm, there is also a learning method that makes more efficient use of the auxiliary information that may be available. As these auxiliary data are employed only in the training set and are unavailable in the test set, they are also referred to as "privileged information." Accordingly, the learning paradigm that makes use of such auxiliary data is known as learning with privileged information(LuPI)[3], [9].

In detail, additional data, denoted $\mathbf{X}^*$, and an auxiliary function, $g \in \mathcal{G}$, are introduced into the learning using privileged information. Thus, the training dataset is changed from the tuple $\{\mathbf{X}, \mathbf{Y}\}$ into the triple $\{\mathbf{X}, \mathbf{X}^*, \mathbf{Y}\}$[9], which results in the optimal solution problem as follows:

$$\arg\min_{f \in \mathcal{F}, g \in \mathcal{G}} R_{emp}(f, g \mid \mathbf{X}, \mathbf{X}^*, \mathbf{Y}).$$

The form of the auxiliary function g depends on the specific case, since there are a number of possible forms and sources of auxiliary information, $\mathbf{X}^*$. As a simple instance, consider a time series with the same interval between time points. In this case, the observations at time $t + 3\Delta t$ can be predicted based on the observations at time t. Then the observations at $t + \Delta t$, $t + 2\Delta t$ are also useful for training. Furthermore, we can even use the data at time $t + 4\Delta t$ and beyond. Although the future data is actually not visible at time $t + 3\Delta t$, it does not cause an information leakage problem because it is only used in the training set. In fact, the time series can be seen as a straightforward example of a Bayesian network (Markov chain). Besides, the construction of the auxiliary function depends on whether the relationship between the variables is linear or nonlinear, and on the machine learning method used. In our thesis, the main application of the LuPI paradigm was in the field of classical linear regression, and thus the algorithm was designed under similar assumptions to the linear regression models.

2.4 Linear regression

Linear models are widely used in machine learning, and linear regression is one of the main applications. The goal of linear regression is to predict the conditional expectation $\mathbb{E}[Y|X]$ from the input $X \in \mathbb{R}^d$, where $Y \in \mathbb{R}^q$ is the target variable. Assuming a linear relationship between the independent variable X and the target variable Y, the general form of linear regression is expressed as[10]:

$$\mathbb{E}[Y_j|X] = \beta_{0j} + \sum_{i=1}^{d} \beta_{ij} X_i,$$

where β_{ij} are the parameters to be estimated. If it is matrixed, i.e,

$$\beta = \begin{pmatrix} \beta_{01} & \beta_{02} & \cdots & \beta_{0q} \\ \beta_{11} & \beta_{12} & \cdots & \beta_{1q} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{d1} & \beta_{d2} & \cdots & \beta_{dq} \end{pmatrix}$$

and let $X = (1, X_1, \ldots, X_d)^T$, then the equivalent vector form expression of the above equation is

$$\mathbb{E}[Y|X] = \beta^T X.$$

A noise term is usually introduced to represent the uncertainty in the prediction, and so the model between Y and X is typically assumed to be a linear relationship with Gaussian noise. Let ϵ denote the noise term, which is assumed to be normally distributed with a mean of zero and a constant variance. Then the linear model is

$$Y = \beta^T X + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \Sigma).$$

Since each component of Y is assumed to be independent (i.e., the covariance between them is 0), Σ has to be a diagonal matrix. Whether or not each element on the diagonal of Σ is equal depends on if the noise in the system is isotropic or not. For isotropic noise one has

$$\Sigma = \sigma^2 I_q.$$

Instead, anisotropic Gaussian noise results in unequal diagonal elements. Based on the construction of all the above, it can be summarized that the linear regression models has some key assumptions as follows:

- The relationship between the target and independent variables is linear.
- The noise term ϵ is Gaussian and has zero mean.
- The variance of the noise term is constant, i.e. $\text{Var}(\epsilon) = \Sigma$.
- The noise term is uncorrelated with the independent variable X, and each of its components is independent of each other, i.e. $\text{Var}(\epsilon|X) = \Sigma$ and $\text{Cov}(\epsilon_i, \epsilon_j) = 0$.

2.5 Ordinary least squares

Since the true parameter matrix β in a linear regression model is usually not known, it is necessary to estimate the values of the parameters through observation samples. Typically, such estimation is based on a set of training data $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where each x_i is the feature vector for the i-th sample. One of the most common estimators is the least squares method. As the name implies, it uses the mean square loss as the criterion for fitting, with the goal of finding the coefficient $\hat{\beta}$ to minimize the residual sum of squares (SS_{res})[11]:

$$SS_{res}(\beta) = \sum_{i=1}^{N} (y_i - f(x_i))^2,$$

The residual sum of squares is a measure of the discrepancy between the predicted and observed values of a regression model. To matrix the above equations, let $\mathbf{X} = (\mathbf{1}, x_1, \ldots, x_m)^T$ and $\mathbf{Y} = (y_1, \ldots, y_m)^T$. The residual sum of squares can then be expressed as

$$SS_{res}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta).$$

It is a matrix function with derivative with respect to β:

$$\frac{\partial SS_{res}}{\partial \beta} = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta).$$

To find the minimum value, set the first order derivative to zero, i.e,

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0.$$

Assume that $\mathbf{X}$ has full column rank, which means that all features of the independent variable X are independent or approximately independent of each other (this is easy to achieve when the number of samples N is much larger than the number of features d+1). In this case $\mathbf{X}^T\mathbf{X}$ is invertible, resulting in a unique solution:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

The solution is commonly known as the least squares estimator, which can be shown to be unbiased. By replacing the model $\mathbf{y} = \mathbf{X}\beta + \epsilon$ into the expression for the estimator, one has

$$\begin{aligned}\hat{\beta} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\beta + \epsilon) \\ &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon \\ &= \beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon.\end{aligned}$$

Further, computing the expectation of $\hat{\beta}$ yields

$$\begin{aligned}\mathbb{E}[\hat{\beta}] &= \mathbb{E}\left[\beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\epsilon\right] \\ &= \beta + (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbb{E}[\epsilon].\end{aligned}$$

By the basic assumptions of the linear regression model, $\mathbb{E}[\epsilon] = 0$, so the above equation reduces to

$$\mathbb{E}[\hat{\beta}] = \beta,$$

which indicates the unbiasedness of $\hat{\beta}$. Given the positive relationship between the mean squared error (MSE) and the sum of squares of the residuals, as indicated by the equation $SS_{res} = n \cdot MSE$, it is evident that the least squares estimate is the optimal estimator in terms of MSE. In addition, one metric used to evaluate the fit of a regression model is the R^2 score (coefficient of determination). It takes a value between 0 and 1 and indicates the proportion of variation in the dependent variable that is explained by the independent variable, with higher values being better. If R^2 is close to 1, the model explains the variation in the data well; if R^2 is close to 0, the model explains the data poorly.The R^2 score is generally defined as

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}},$$

where SS_{tot} is the total sum of squares, i.e., the sum of squares of the differences between the real values and the means. The formulas are

$$SS_{tot} = \sum_{i=1}^{n}(y_i - \bar{y})^2,$$
$$SS_{res} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where y_i is the first i observation, $\bar{y}$ is the mean of the dependent variable y, and $\hat{y}_i$ is the predicted value of the first i observation.The relationship between R^2 and MSE can be expressed by the sum of squares of the residuals and the total sum of squares:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{n \cdot MSE}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Thus, both R^2 and MSE measure the prediction accuracy of the model. However, since R^2 is a normalized ratio, it is easier to interpret and can be used to compare the performance of different models. MSE, on the other hand, provides a measure of absolute error and is more commonly used to observe trends.

2.6 The Rao-Blackwell theorem

The Rao-Blackwell theorem[12], [13], an important statistical theorem, provides a way to improve an estimator by conditioning expectations on a sufficient statistical quantity. Specifically, if we have an estimator $\hat{\theta}$ for the parameter θ and a sufficient statistic T for θ, then the Rao-Blackwell improved estimator $\hat{\theta}^*$ is given by $\hat{\theta}^* = \mathbb{E}[\hat{\theta} \mid T]$. The theorem shows that the improved estimate $\hat{\theta}^*$ remains unbiased, and is better than or equal to the original estimate $\hat{\theta}$ in terms of the mean square error, i.e.

$$\mathbb{E}[(\hat{\theta}^* - \theta)^2] \leq \mathbb{E}[(\hat{\theta} - \theta)^2]. \tag{2.1}$$

The most important thing that has been needed in our theory is the argument idea of this theorem. As discussed in the statistics lecture notes of R. Weber[14], a short proof of the equation 2.1 is as follows:

$$\begin{aligned}
\mathbb{E}[(\hat{\theta}^* - \theta)^2] &= \mathbb{E}\left[(\mathbb{E}(\hat{\theta} \mid T) - \theta)^2\right] \\
&= \mathbb{E}\left[\mathbb{E}((\hat{\theta} - \theta) \mid T)^2\right] \\
&\leq \mathbb{E}\left[\mathbb{E}((\hat{\theta} - \theta)^2 \mid T)\right] \\
&= \mathbb{E}[(\hat{\theta} - \theta)^2].
\end{aligned}$$

In the above proof, the second line uses the conditional expectation property, the third line uses Jensen's inequality, and the last line is due to the law of total expectation. A similar proof will follow for Theorem 1 in this thesis. In addition, the mean square error can be decomposed into two parts, the variance and the square of the bias, by

the following steps:

$$\begin{aligned}\mathrm{MSE}(\hat{\theta}) &= \mathbb{E}[(\hat{\theta}-\theta)^2] \\ &= \mathbb{E}[(\hat{\theta}-\mathbb{E}[\hat{\theta}]+\mathrm{Bias}(\hat{\theta}))^2] \\ &= \mathbb{E}[(\hat{\theta}-\mathbb{E}[\hat{\theta}])^2+2(\hat{\theta}-\mathbb{E}[\hat{\theta}])\,\mathrm{Bias}(\hat{\theta})+\mathrm{Bias}(\hat{\theta})^2] \\ &= \mathrm{Var}(\hat{\theta})+2\,\mathrm{Bias}(\hat{\theta})\mathbb{E}[\hat{\theta}-\mathbb{E}[\hat{\theta}]]+\mathrm{Bias}(\hat{\theta})^2 \\ &= \mathrm{Var}(\hat{\theta})+\mathrm{Bias}(\hat{\theta})^2.\end{aligned}$$

Again, since both the improved estimates $\hat{\theta}^*$ and the original estimates $\hat{\theta}$ are unbiased, their variances will also satisfy the inequality

$$\mathrm{Var}(\hat{\theta}^*) \leq \mathrm{Var}(\hat{\theta}).$$

2.7 Distillation Method

The technique of knowledge distillation was proposed by Hinton et al. in 2015[15]. It transfers knowledge from a large, complex model (called a "teacher model") to a smaller, simpler model (called a "student model"), which allows the simplified model to run more efficiently with fewer resources, while maintaining relatively high performance. The principle of knowledge distillation is based on two core functions, the adjusted softmax function and the distillation loss function, which allow the student model to efficiently learn its prediction distribution from the teacher model. Softmax, a standard output layer for multi-class classification problems, transforms logits that are inputs to a model output layer into probability distributions. It is defined as

$$\mathrm{softmax}(z_i, T) = \frac{\exp(z_i/T)}{\sum_{j=1}^{C}\exp(z_j/T)}$$

where z_i is the logit of the first i category, C is the total number of categories, and T is the temperature parameter. In the distillation process, the sharpness of the output of the softmax function can be adjusted by introducing T. The higher the temperature T, the smoother the resulting probability distribution. In addition, to combine information from the teacher model and the observed data, the distillation loss function is usually a weighted sum of two parts. The first part is the cross-entropy loss between the student model output and the teacher model output, i.e., the soft target loss, and the other part is the cross-entropy loss between the student model output and the real label, i.e., the hard target loss:

$$L = \alpha \cdot H\left(\mathrm{softmax}(z^s, T), \mathrm{softmax}(z^t, T)\right) + (1-\alpha)\cdot H\left(\mathrm{softmax}(z^s, 1), y\right)$$

where z^s and z^t are the logits of the student and teacher models, respectively, y is the one-hot coding of the true labels, α is a hyperparameter that controls the importance of the two parts of the loss, and H denotes the cross-entropy function. However, to implement distillation methods in regression tasks, different loss and

distribution functions have to be used. In our directed Gaussian graphical model, the neural network is replaced by linear regression, while the cross-entropy loss is replaced by the squared loss. As a result, in Section 3.4, we will derive an expression for the estimator in this case and show that the mean square error of the distillation estimator will lie between the mean square error of the estimate of its teacher model and the mean square error of the ordinary least squares estimate.

3

Theory

This chapter describes the Learning using Privileged Mediating Information (LuPMI) algorithm based on Directed Gaussian graphical models. First, the problem formulation and model assumptions are given in the first section, followed by the corresponding algorithms using the model assumptions. Then, the effectiveness theory of the algorithms is presented and proved.

3.1 Problem formulation

The task of this thesis is to find a model, specifically a function $h \in \mathcal{H}$, where $\mathcal{H} \subseteq \{f : \mathbb{R}^d \to \mathbb{R}\}$, that predicts the outcome $Y \in \mathbb{R}$ from the independent variable $X_1 \in \mathbb{R}^d$. The system is primarily structured on the basis of a directed Gaussian graphical model(directed GGM), with a few additional assumptions. In detail, our problem is modeled as a causal DAG, where X_1 is the root node and Y is the leaf node. There are a number of mediating variables $X_2, \ldots, X_n \in \mathbb{R}^d$ on the path from X_1 to Y. As one of the basic assumptions of the directed GGM, it is supposed that each node depends only on its parent nodes and is conditionally independent of other nodes given its parents. A simple example of the specific system is shown in Fig.3.1, which involves 3 mediating variables and 3 paths from X_1 to Y.

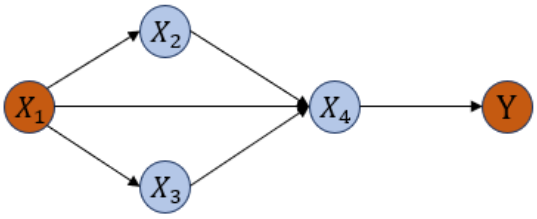


Figure 3.1: An example of the model, where the blue nodes represent the mediating variables X_2, X_3, X_4, and the yellow-brown nodes represent the independent variable X_1 and the outcome variable Y, respectively.

In the previous chapter, we introduced the mathematical formulation of directed GGM, expressed as a linear relationship with Gaussian noise. Similarly, the following assumption is proposed as the foundation of the theory as a rigorous and explicit expression of the model.

Assumption 1. *(Directed GGM) For any variable belonging to* $\mathcal{V} = \{X_1, \ldots, X_n\}$,

it is assumed that the variable is only linearly related to the variables represented by all its parent nodes in the DAG, with isotropic noise, and conditionally independent with respect to all other variables, that is,

$$X_k = \sum_{X_j \in Pa(X_k)} (A_{jk})^T X_j + \epsilon_k, \quad \textit{where } \epsilon_k \sim \mathcal{N}(0, \sigma_k^2)$$

$$Y = \sum_{X_{i_p} \in Pa(Y)} \beta_{i_p}^T X_{i_p} + \epsilon_Y, \quad \textit{where } \epsilon_Y \sim \mathcal{N}(0, \sigma_y^2)$$

The parameters $\{A_{jk} \mid (j,k) \in \mathcal{E}\}$ and $\{\beta_{i_p} \mid X_{i_p} \in Pa(Y)\}$ in Assumption 1 are also called "path coefficients" in the model. Where $\{A_{jk} \mid (j,k) \in \mathcal{E}\}$ denotes the linear relationship between the independent variable and the mediating variables and between the mediating variables and the target variable, and $\{\beta_{i_p} \mid X_{i_p} \in Pa(Y)\}$ denotes the relationship between the mediating variable and the target variable.

On the basis of Assumption 1 and the directed acyclic graph, it is clear that an analogous relationship exists between X_1 and Y. This can be formulated as $Y = \theta^T X_1 + \epsilon_0$, where the parameter θ and the noise ϵ_0 are derived from the following equations:

$$\theta = \sum_{p \in \mathcal{P}(X_1, Y)} \left[\left(\prod_{(X_i, X_j) \in p} A_{ij} \right) \beta_{i_p} \right], \tag{3.1a}$$

$$\epsilon_0 = \sum_{k=2}^{n} \theta_{ky}^T \epsilon_k + \epsilon_Y, \quad \text{where} \quad \theta_{ky} = \sum_{p \in \mathcal{P}(X_k, Y)} \left[\left(\prod_{(X_i, X_j) \in p} A_{ij} \right) \beta_{i_p} \right]. \tag{3.1b}$$

From the above equations, it is obvious that the effects of the independent variable X_1 on the target variable Y are accumulated along the paths, which eventually appear as the summation of the effects of multiple paths. Based on Eq.3.1, we can calculate the real coefficients and noise levels between X_1 and Y in the synthetic dataset, which is useful for estimating the characteristics of the synthetic system and provides theoretical support for the experiments in Section 4.2.1.

3.2 Algorithm

Before introducing the algorithm and the efficiency theory, there are some notations and rules that need to be clarified. For complicated sets, $\{X \mid \text{condition } A\}$ denotes all elements X that satisfy condition A. For example, $\{\hat{A}_{jk} \mid j \in Pa(X_k)\}$ denotes a set in which each element $\hat{A}_{jk}$ has a subscript index jk satisfying $j \in Pa(X_k)$. Furthermore, since the index i of the nodes in the DAG is artificially determined, we can define the index rule as $Pa(X_{k+1}) \subseteq \{X_1, \ldots, X_k\}$. That is to say, all the parent nodes of a node X_{k+1} must have a smaller index ($\leq k$) than it. This index rule is only for the convenience of theoretical proofs and statements, and is not needed in the real implementation of the algorithm.

The whole training dataset $D = \{x_{i,1}, \ldots, x_{i,n}, y_i\}_{i=1}^m$ is a set of independent random samples observed from distribution $p(X_1, \ldots, X_n, Y)$, where $x_{i,n}$ represents the i-th observation of variable X_n. As an alternative form, it can also be written as

$(\{\mathbf{X}_j\}_{j=1}^n, \mathbf{Y})$, where $\mathbf{X}_j = (x_{1,j}, \ldots, x_{m,j})^T$ and $\mathbf{Y} = (y_1, \ldots, y_m)^T$. For simplicity of expression, $Pa(\mathbf{X}_i)$ represents the data of observations corresponding to the parent nodes of the node $\mathbf{X}_i$. As a metric for evaluating the model, our goal is to obtain the minimized expected risk. The expected risk is typically estimated by the empirical risk, and the estimator that minimizes the empirical risk is obtained from the ordinary least squares method. Based on the above, we apply ordinary least squares step by step to all causal relationships in the DAG, which leads to Algorithm 1.

Algorithm 1 Learning using Privileged Mediating Information(LuPMI)

Input: Data $D : \{x_{i,1}, \ldots, x_{i,n}, y_i\}_{i=1}^m$; graph $G = (\mathcal{V}, \mathcal{E})$
for every node X_k in G **do**

$$\{\hat{A}_{jk} \mid X_j \in Pa(X_k)\} = \underset{\{\hat{A}_{jk} \mid X_j \in Pa(X_k)\}}{\arg\min} \sum_{i=1}^{m} \frac{\| \sum_{X_j \in \mathrm{Pa}(X_k)} (A_{jk})^T x_{i,j} - x_{i,k}\|^2}{m}$$

end for

$$\{\hat{\beta}_{i_p} \mid X_{i_p} \in Pa(Y)\} = \underset{\{\beta_{i_p} \mid X_{i_p} \in Pa(Y)\}}{\arg\min} \sum_{i=1}^{m} \frac{\| \sum_{X_{i_p} \in \mathrm{Pa}(Y)} \beta_{i_p}^T x_{i,i_p} - y_i\|^2}{m}$$

for each path p in DAG from X_1 to Y **do**

$$\hat{A}_p = \prod_{(X_j, X_k) \in \text{path } p} \hat{A}_{jk}$$

$$\hat{A}'_p = \hat{A}_p \hat{\beta}_{i_p}$$

end for
return $\hat{\theta} = \sum_p \hat{A}'_p$

On Algorithm 1, estimates of all path coefficients, $\{\hat{A}_{jk} \mid (j,k) \in \mathcal{E}\}$ and $\{\hat{\beta}_{i_p} \mid X_{i_p} \in Pa(Y)\}$, are obtained by multivariate linear regression. This is because when a node in a Bayesian network has more than one parent, its conditional probability is a joint distribution with respect to all of its parents. Finally, based on the estimation of all path coefficients, the estimator of the parameter theta in the system is expressed as

$$\hat{\theta}_{\mathrm{LuPMI}} = \sum_{p \in \mathcal{P}(X_1, Y)} \left[\left(\prod_{(X_i, X_j)\ inp} \hat{A}_{ij} \right) \hat{\beta}_{i_p} \right].$$

3.3 Efficiency theory

In order to theoretically verify the efficiency of Algorithm 1, the discussion in this section is based on ideal conditions, i.e. it is assumed that the causality of the variables in the data set is known. In this case, the performance of the LuPMI estimator has an improvement in statistical properties compared to the classical OLS estimator. Using the Rao-Blackwell theorem and related lemmas, it can be shown that the LuPMI algorithm has advantages in terms of Mean Square Error (MSE) and Expected Risk. The mean square error of the parameter estimates is defined as

$$\mathrm{MSE}(\hat{\theta}) := \mathbb{E}_D[||\hat{\theta} - \theta||_2^2].$$

And the expected risk is an expectation of the trained model's performance across all potential datasets and in our model is expressed as

$$\bar{R}(\hat{\theta}) := \mathbb{E}_D \left[\mathbb{E}_{X_1,Y} \left[\ell(\hat{\theta}^\top X_1, Y) \right] \right].$$

The outer expectation is the expectation of the random traning dataset, while the inner expectation is the expectation of the joint distribution of the input-output pairs. Then the following lemmas and theorems prove that the estimation error and prediction risk of the LuPMI estimator are not inferior to the classical OLS method when the causal relationship is known, and in some cases show obvious advantages.

Theorem 1. *Let $D = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n, \mathbf{Y})$ be a random dataset from a known directed Gaussian graphical system as defined in Assumption 1, $\hat{\theta}_{LuPMI}$ be the output of Algorithm 1, and $\hat{\theta}_{OLS} = (X_1^T X_1)^{-1} X_1^T Y$ be the classical least squares estimator. With isotropic noise defined in Assumption 1, $\hat{\theta}_{LuPMI}$ is unbiased, and*

$$MSE(\hat{\theta}_{LuPMI}) = MSE(\hat{\theta}_{OLS}) - \mathbb{E}_D[Tr(Cov(\hat{\theta}_{OLS}|\hat{\theta}_{LuPMI}))],$$

Further, it holds for the expected risk that over any samples (X_1, Y),

$$\bar{R}(\hat{\theta}_{LuPMI}) = \bar{R}(\hat{\theta}_{OLS}) - \mathbb{E}_{D,X_1}[Var_{\hat{\theta}_{OLS}}((\hat{\theta}_{OLS}, X_1)|\hat{\theta}_{LuPMI})].$$

The key to the efficiency of the LuPMI algorithm is stated and proved in Theorem 1, which illustrates the advantages of the LuPMI algorithm over classical least squares estimation in terms of MSE and expected risk. The LuPMI estimator $\hat{\theta}_{\text{LuPMI}}$ has a smaller MSE than the OLS estimator $\hat{\theta}_{\text{OLS}}$ under Assumption 1. The proof of this theorem is based on Lemmas 2 and 3, as well as the standard Rao-Blackwell argument. Therefore, before giving a detailed proof, we first state the following lemmas.

3.3.1 Orthogonality of residuals

First, we show that there is still orthogonality between the residuals and the regressor matrix even for multivariate linear regression. Consider any node X_k $(k = 2, \cdots, n)$, then all possible parents of the node X_k are contained in $\{X_1, \ldots, X_{k-1}\}$ according to our indexing rule $Pa(X_k) \subseteq \{X_1, \ldots, X_{k-1}\}$. Besides, in Algorithm1, the estimators of the edge coefficients between each node and its parent node is determined by the following least squares estimation of the multivariate linear regression

$$\{\hat{A}_{jk} \,|\, X_j \in Pa(X_k)\} = \underset{\{\hat{A}_{jk} \,|\, X_j \in Pa(X_k)\}}{\arg\min} \mathcal{L}$$

where $\mathcal{L}$ is the squared loss function and

$$\begin{aligned} \mathcal{L} &= \sum_{i=1}^{m} \frac{\| \sum_{X_j \in \mathrm{Pa}(X_k)} (A_{jk})^T x_{i,j} - x_{i,k} \|^2}{m} \\ &= \| \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j A_{jk} - \mathbf{X}_k \|^2 \end{aligned}$$

For each i such that $X_i \in \mathrm{Pa}(X_k)$, compute the components of the gradient $\frac{\partial \mathcal{L}}{\partial A_{ik}}$ to obtain estimators that minimize the loss function,

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial A_{ik}} &= \frac{\partial}{\partial A_{ik}}\left[\left(\sum_{\substack{j\neq i\\ \mathbf{X}_j\in \mathrm{Pa}(\mathbf{X}_k)} A_{jk}\mathbf{X}_j^T + A_{ik}\mathbf{X}_i^T - \mathbf{X}_k^T\right)\left(\sum_{\substack{j\neq i\\ \mathbf{X}_j\in \mathrm{Pa}(\mathbf{X}_k)} \mathbf{X}_j A_{jk} + \mathbf{X}_i A_{ik} - \mathbf{X}_k\right)\right]\\ &= \frac{\partial}{\partial A_{ik}}\left(\sum_{\substack{j\neq i\\ \mathbf{X}_j\in \mathrm{Pa}(\mathbf{X}_k)} A_{jk}^T\mathbf{X}_j^T\mathbf{X}_i A_{ik} + \sum_{\mathbf{X}_j\in \mathrm{Pa}(\mathbf{X}_k)} A_{ik}^T\mathbf{X}_i^T\mathbf{X}_j A_{jk} - \mathbf{X}_k^T\mathbf{X}_i A_{ik} - A_{ik}^T\mathbf{X}_i\mathbf{X}_k\right)\\ &= 2\sum_{\mathbf{X}_j\in \mathrm{Pa}(\mathbf{X}_k)} \mathbf{X}_i^T\mathbf{X}_j A_{jk} - 2\mathbf{X}_i^T\mathbf{X}_k.\end{aligned}$$

Let the gradient be 0, we obtain that the minimizers should satisfy

$$\mathbf{X}_i^T\left(\mathbf{X}_k - \sum_{X_j\in \mathrm{Pa}(X_k)} \mathbf{X}_j\hat{A}_{jk}\right) = 0 \quad \text{for any } X_i \in \mathrm{Pa}(X_k),$$

where the bracketed parts are the residuals $\mathbf{R}_k = \mathbf{X}_k - \sum_{X_j\in \mathrm{Pa}(X_k)} \mathbf{X}_j\hat{A}_{jk}$. So the above equation is equivalent to

$$\mathbf{X}_i^T\mathbf{R}_k = 0 \quad \text{for any } X_i \in \mathrm{Pa}(X_k).$$

3.3.2 Symmetry of the conditional distribution

In a directed Gaussian graphical model, dependencies between nodes are represented by the edges of the DAG, and the mathematical representation of these dependencies is usually described by conditional distributions. It is worth noting that certain conditional distributions are shown to be symmetric, as follows:

Lemma 1. *Let $K = \{\hat{A}_{ij} \,|\, (X_i, X_j) \in \mathcal{E}\} \cup \{\hat{\beta}_{i_p} \,|\, X_{i_p} \in \mathrm{Pa}(Y)\}$ be the set of all the edge coefficient estimators obatined by Algorithm 1 and let $D = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n, \mathbf{Y})$ be a random dataset from a directed Gaussian graphical system with isotropic noises as defined in Assumption 1. Define*

$$\begin{aligned}\mathbf{X}_k &= \sum_{X_j\in Pa(X_k)} \mathbf{X}_j\hat{A}_{jk} + \mathbf{R}_k,\\ \mathbf{X}'_k &= \sum_{X_j\in Pa(X_k)} \mathbf{X}_j\hat{A}_{jk} - \mathbf{R}_k.\end{aligned}$$

Then for any $k = 2, \cdots, n$, it holds that

$$p(\mathbf{X}_k|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}) = p(\mathbf{X}'_k|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}) \tag{3.2}$$

and

$$p(\mathbf{Y}|\mathbf{X}_1, \ldots, \mathbf{X}_n) = p(\mathbf{Y}'|\mathbf{X}_1, \ldots, \mathbf{X}_n). \tag{3.3}$$

Proof. Let $\boldsymbol{\epsilon}'_k = \boldsymbol{\epsilon}_k - 2\mathbf{R}_k$, where $\boldsymbol{\epsilon}_k$ is the Gaussian noise for the variable X_k. Then we have from Assumption 1:

$$\mathbf{X}_k = \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk} + \boldsymbol{R}_k$$

and

$$\begin{aligned}
\mathbf{X}'_k &= \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk} - \mathbf{R}_k \\
&= \mathbf{X}_k - 2\mathbf{R}_k \\
&= \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j A_{jk} + \boldsymbol{\epsilon}_k - 2\mathbf{R}_k \\
&= \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j A_{jk} + \boldsymbol{\epsilon}'_k
\end{aligned}$$

Thus, Eq.3.2 is equivalent to $p(\boldsymbol{\epsilon}_k | \mathbf{X}_1, \ldots, \mathbf{X}_{k-1}) = p(\boldsymbol{\epsilon}'_k | \mathbf{X}_1, \ldots, \mathbf{X}_{k-1})$. Furthermore, the probability distribution function p of Gaussian noise depends on the inner product of the noise. By definition,

$$\boldsymbol{\epsilon}'^{T}_k \boldsymbol{\epsilon}'_k = \boldsymbol{\epsilon}^T_k \boldsymbol{\epsilon}_k - 4\mathbf{R}^T_k \boldsymbol{\epsilon}_k + 4\mathbf{R}^T_k \mathbf{R}_k = \boldsymbol{\epsilon}^T_k \boldsymbol{\epsilon}_k - 4\mathbf{R}^T_k (\boldsymbol{\epsilon}_k - \mathbf{R}_k).$$

With $\mathbf{R}_k = \mathbf{X}_k - \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk}$, it's obtained that

$$\begin{aligned}
\mathbf{R}^T_k (\boldsymbol{\epsilon}_k - \mathbf{R}_k) &= \mathbf{R}^T_k \left[(\mathbf{X}_k - \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk}) - (\mathbf{X}_k - \sum_{X_j \in \mathrm{Pa}(V_k)} \mathbf{X}_j A_{jk}) \right] \\
&= \mathbf{R}^T_k \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j (\hat{A}_{jk} - A_{jk}) \\
&= \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{R}^T_k \mathbf{X}_j (\hat{A}_{jk} - A_{jk}).
\end{aligned}$$

Since $\mathbf{R}^T_k \mathbf{X}_j = 0$ for any $X_j \in \mathrm{Pa}(X_k)$, we get $\mathbf{R}^T_k (\boldsymbol{\epsilon}_k - \mathbf{R}_k) = 0$, which implies

$$\boldsymbol{\epsilon}'^{T}_k \boldsymbol{\epsilon}'_k = \boldsymbol{\epsilon}^T_k \boldsymbol{\epsilon}_k.$$

This means that $\boldsymbol{\epsilon}'_k$ and $\boldsymbol{\epsilon}_k$ are either equivalent or differ by a negative sign because the noise is isotropic. And since the Gaussian distribution is positive-negative symmetric, the distributions of $\boldsymbol{\epsilon}'_k$ and $\boldsymbol{\epsilon}_k$ are the same in both cases, thus proving Eq.3.2. And for Eq.3.3, it can be proved by showing that $\boldsymbol{\epsilon}'^{T}_Y \boldsymbol{\epsilon}'_Y = \boldsymbol{\epsilon}^T_Y \boldsymbol{\epsilon}_Y$ using a similar argument. □

We also propose the following conjecture. Although we are not yet able to strictly prove it, there is a possible idea for a proof that could provide a preliminary demonstration.

Conjecture 1. *Let* $K = \{\hat{A}_{ij} \mid (X_i, X_j) \in \mathcal{E}\} \cup \{\hat{\beta}_{i_p} \mid X_{i_p} \in \mathrm{Pa}(Y)\}$ *be the set of all the edge coefficient estimators obatined by Algorithm 1 and let* $D = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n, \mathbf{Y})$

be a random dataset from a directed Gaussian graphical system with isotropic noises as defined in Assumption 1. Define

$$\mathbf{X}_k = \sum_{X_j \in Pa(X_k)} \mathbf{X}_j \hat{A}_{jk} + \mathbf{R}_k,$$
$$\mathbf{X}'_k = \sum_{X_j \in Pa(X_k)} \mathbf{X}_j \hat{A}_{jk} - \mathbf{R}_k.$$

Then for any $k = 2, \cdots, n$, it holds that

$$p(K|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}, \mathbf{X}_k) = p(K|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}, \mathbf{X}_k^{'})$$

and

$$p(K|\mathbf{X}_1, \ldots, \mathbf{X}_n, \mathbf{Y}) = p(K|\mathbf{X}_1, \ldots, \mathbf{X}_n, \mathbf{Y}^{'}).$$

Potential idea of proof:

Proof. To show $p(K \mid \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k) = p(K \mid \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X'}_k)$, we first define $\hat{\beta} = (\hat{\beta}_{i_1}, \hat{\beta}_{1_2}, \ldots, \hat{\beta}_{i_s})^\top$ (assume the node Y has s parent nodes), $\hat{\mathbf{A}}_j = (\hat{A}_{i_1}, \hat{A}_{i_2}, \ldots, \hat{A}_{i_t})^\top$ (assume the node X_j has t parent nodes) and $\tilde{\mathbf{X}}_j = (\mathbf{X}_{i_1}, \mathbf{X}_{i_2}, \ldots, \mathbf{X}_{i_t})^\top$.

Then, $p(K \mid \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_k)$ can be factorized as

$$p(\hat{\beta} \mid \hat{\mathbf{A}}_n, \hat{\mathbf{A}}_{n-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k) p(\hat{\mathbf{A}}_n \mid \hat{\mathbf{A}}_{n-1}, \ldots, \hat{\mathbf{A}}_1, \mathbf{X}_1, \ldots, \mathbf{X}_k) \ldots p(\hat{\mathbf{A}}_1 \mid \mathbf{X}_1, \ldots, \mathbf{X}_k)$$

let $C_m = \{\hat{\mathbf{A}}_2, \ldots, \hat{\mathbf{A}}_m\}$ for $m = 2, \cdots, n$, it's sufficient to prove:

$$p(\hat{\mathbf{A}}_m \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k) = p(\hat{\mathbf{A}}_m \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}'_k)$$

$$p(\hat{\beta} \mid C_n, \mathbf{X}_1, \ldots, \mathbf{X}_k) = p(\hat{\beta} \mid C_n, \mathbf{X}_1, \ldots, \mathbf{X}'_k)$$

for any m. With the distribution of OLS estimator, it's known that

$$\hat{\mathbf{A}}_m \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k \sim \mathcal{N}\left(A_m, \sigma_m^2 \mathbb{E}\left[(\tilde{\mathbf{X}}_m^\top \tilde{\mathbf{X}}_m)^{-1} \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k\right]\right)$$

$$\hat{\beta} \mid C_n, \mathbf{X}_1, \ldots, \mathbf{X}_k \sim \mathcal{N}\left(\beta, \sigma_Y^2 \mathbb{E}\left[(\tilde{\mathbf{X}}_Y^\top \tilde{\mathbf{X}}_Y)^{-1} \mid C_n, \mathbf{X}_1, \ldots, \mathbf{X}_k\right]\right)$$

Then it is sufficient to show

$$\mathbb{E}\left[(\tilde{\mathbf{X}}_m^\top \tilde{\mathbf{X}}_m)^{-1} \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k\right] = \mathbb{E}\left[(\tilde{\mathbf{X}}_m^\top \tilde{\mathbf{X}}_m)^{-1} \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}'_k\right], \quad (3.4a)$$

$$\mathbb{E}\left[(\tilde{\mathbf{X}}_Y^\top \tilde{\mathbf{X}}_Y)^{-1} \mid C_n, \mathbf{X}_1, \ldots, \mathbf{X}_k\right] = \mathbb{E}\left[(\tilde{\mathbf{X}}_Y^\top \tilde{\mathbf{X}}_Y)^{-1} \mid C_n, \mathbf{X}_1, \ldots, \mathbf{X}'_k\right] \quad (3.4b)$$

In partitioned matrix form, $(\tilde{\mathbf{X}}_m^\top \tilde{\mathbf{X}}_m)^{-1}$ can be written as

$$(\tilde{\mathbf{X}}_m^\top \tilde{\mathbf{X}}_m)^{-1} = \begin{pmatrix} \mathbf{X}_{i1}^\top \mathbf{X}_{i1} & \mathbf{X}_{i1}^\top \mathbf{X}_{i2} & \ldots & \mathbf{X}_{i1}^\top \mathbf{X}_{it} \\ \mathbf{X}_{i2}^\top \mathbf{X}_{i1} & \mathbf{X}_{i2}^\top \mathbf{X}_{i2} & \ldots & \mathbf{X}_{i2}^\top \mathbf{X}_{it} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{X}_{it}^\top \mathbf{X}_{i1} & \mathbf{X}_{it}^\top \mathbf{X}_{i2} & \ldots & \mathbf{X}_{it}^\top \mathbf{X}_{it} \end{pmatrix}^{-1}$$

Then it's sufficient to show $\mathbf{X}_i^\top \mathbf{X}_j$ for any $X_i, X_j \in Pa(X_m)$, given the conditions $\{C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k\}$ and $\{C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}'_k\}$ respectively. Also, since X_1 is the only

independent variable in our model, the data of the variable $\mathbf{X}_i$ $(i = 2, \cdots, n)$ in the DAG can be represented as follows according to Assumption 1:

$$\mathbf{X}_i = \mathbf{X}_1\hat{\mu}_{1i} + \bar{\mathbf{R}}_i, \quad \text{where} \quad \hat{\mu}_{1i} = \sum_{p\in\mathcal{P}(X_1,X_i)} \left[\left(\prod_{(X_r,X_q)\in p} \hat{A}_{rq} \right) \right]$$

where $\bar{\mathbf{R}}_i$ is the combined residual. Through the same steps as in Eq.3.1b, this combined residual is expressed as

$$\bar{\mathbf{R}}_i = \sum_{l=2}^{i} \mathbf{R}_l\hat{\mu}_{li} + \mathbf{R}_i, \quad \text{where} \quad \hat{\mu}_{li} = \sum_{p\in\mathcal{P}(X_l,X_i)} \left[\left(\prod_{(X_r,X_q)\in p} \hat{A}_{rq} \right) \right].$$

Therefore, applying the above decomposition to $\mathbf{X}_i^\top\mathbf{X}_j$ (for $X_i, X_j \in Pa(X_m)$) yields

$$\begin{aligned} \mathbf{X}_i^\top\mathbf{X}_j &= (\mathbf{X}_1\hat{\mu}_{1i} + \bar{\mathbf{R}}_i)^\top(\mathbf{X}_1\hat{\mu}_{1j} + \bar{\mathbf{R}}_j) \\ &= \hat{\mu}_{1i}^\top(\mathbf{X}_1^\top\mathbf{X}_1)\hat{\mu}_{1j} + \bar{\mathbf{R}}_i^\top\mathbf{X}_1\hat{\mu}_{1j} + \hat{\mu}_{1i}^\top(\mathbf{X}_1^\top\bar{\mathbf{R}}_j) + \bar{\mathbf{R}}_i^\top\bar{\mathbf{R}}_j \end{aligned} \tag{3.5}$$

Since X_i, X_j are assumed to be the parents of X_m, i, j are less than m according to our indexing rule $Pa(X_{k+1}) \subseteq \{X_1, \ldots, X_k\}$. Meanwhile, $\hat{\mu}_{li}$ relies only on path coefficient estimates $\hat{A}_{rq}$ for all paths from X_j to X_i (similarly, the indices r, q are also all less than m). So $\hat{\mu}_{li}$ is given in our conditions C_{m-1} for any $l = 1, 2, \cdots, i$ and $i < m$. Furthermore, since $\mathbf{X}_1$ is fixed, Eq. 3.5 demonstrates that $\mathbf{X}_i^\top\mathbf{X}_j$ depends only on $\bar{\mathbf{R}}_i^\top\mathbf{X}_1$ and $\bar{\mathbf{R}}_i^\top\bar{\mathbf{R}}_j$. To show that each block in the partition matrix $(\tilde{\mathbf{X}}_m^\top\tilde{\mathbf{X}}_m)^{-1}$ remains equivalent under two conditions, we need to prove that

$$\begin{aligned} \bar{\mathbf{R}}_i^\top\bar{\mathbf{R}}_j \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k &= \bar{\mathbf{R}}_i^\top\bar{\mathbf{R}}_j \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}'_k \\ \bar{\mathbf{R}}_i^\top\mathbf{X}_1 \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}_k &= \bar{\mathbf{R}}_i^\top\mathbf{X}_1 \mid C_{m-1}, \mathbf{X}_1, \ldots, \mathbf{X}'_k \end{aligned}$$

Some future work is necessary to complete a rigorous proof. □

Although we are unable to provide a complete proof of the lemma at the moment, preliminary results based on the above ideas suggest a high probability that it holds. At the same time, we have also observed results in our numerical experiments that are consistent with the expectations of the lemma, providing further support for our conjecture. Therefore, although the present lemma is based on conjectures, it can still provide important insights for subsequent theories.

3.3.3 Conditional Expectation

Lemma 1 states that for a system that conforms to the assumption of a directed Gaussian graphical model with isotropic noise, the conditional distribution of the observations of the data matrix at any node X_k has symmetry with respect to the predictions of the model, $\sum_{X_j\in\mathrm{Pa}(X_k)} \mathbf{X}_j\hat{A}_{jk}$. And we conjecture that the estimates of all edge coefficients of the system have similar properties. On this basis, using Bayes' theorem, we proceed to prove the following theory.

Lemma 2. *Assume that Conjecture 1 holds. Let* $K = \{\hat{A}_{ij} \,|\, (X_i, X_j) \in \mathcal{E}\} \cup \{\hat{\beta}_{i_p} \,|\, X_{i_p} \in \mathrm{Pa}(Y)\}$ *be the set of all the edge coefficient estimators obatined by Algorithm 1 and let* $D = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n, \mathbf{Y})$ *be a random dataset from a directed Gaussian graphical system with isotropic noises as defined in Assumption 1, then for any node* X_i *and* Y *in the DAG, it holds that*

$$\mathbb{E}[\mathbf{X}_k|\mathbf{X}_1, \cdots, \mathbf{X}_{k-1}, K] = \sum_{X_j \in Pa(X_k)} \mathbf{X}_j \hat{A}_{jk}$$
$$\mathbb{E}[\mathbf{Y}|\mathbf{X}_1, \cdots, \mathbf{X}_n, K] = \sum_{X_{i_p} \in Pa(Y)} \mathbf{X}_{i_p} \hat{\beta}_{i_p}$$

Proof. First, we prove that $\mathbb{E}[\mathbf{X}_k|\mathbf{X}_1, \cdots, \mathbf{X}_{k-1}, K] = \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk}$, and then explain that the same argument also applies to $\mathbb{E}[\mathbf{Y}|\mathbf{X}_1, \cdots, \mathbf{X}_n, K] = \sum_{X_{i_p} \in \mathrm{Pa}(Y)} \mathbf{X}_{i_p} \hat{\beta}_{i_p}$.

To prove the former, we only need to show that the conditional probability $p(\mathbf{X}_k|\mathbf{X}_1, \cdots, \mathbf{X}_{k-1}, K)$ is symmetric with respect to $\mathbf{X}_k = \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk}$. With Bayes' theorem, it's inferred that

$$p(\mathbf{X}_k|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}, K) = \frac{p(K|\mathbf{X}_1, \ldots, \mathbf{X}_k) p(\mathbf{X}_k|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1})}{p(K|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1})}.$$

Similar to Lemma 1 and Conjecture 1, define $\mathbf{X}_k = \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk} + \mathbf{R}_k$ and $\mathbf{X}'_k = \sum_{X_j \in \mathrm{Pa}(X_k)} \mathbf{X}_j \hat{A}_{jk} - \mathbf{R}_k$. Using the results of Lemma 1 and Conjecture 1,

$$p(\mathbf{X}_k|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}) = p(\mathbf{X}'_k|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}),$$
$$p(K|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}, \mathbf{X}_k) = p(K|\mathbf{X}_1, \ldots, \mathbf{X}_{k-1}, \mathbf{X}'_k),$$

then the Bayes' formula leads to

$$p\left(\mathbf{X}_k \mid \mathbf{X}_1, \cdots, \mathbf{X}_{k-1}, K\right) = p\left(\mathbf{X}'_k \mid \mathbf{X}_1, \cdots, \mathbf{X}_{k-1}, K\right)$$

The first result is thus proved. Second, to show that $\mathbb{E}[\mathbf{Y}|\mathbf{X}_1, \cdots, \mathbf{X}_n, K] = \sum_{X_{i_p} \in \mathrm{Pa}(Y)} \mathbf{X}_{i_p} \hat{\beta}_{i_p}$, we can use the same arguments as before to show

$$p\left(\mathbf{Y} = \sum_{X_{i_p} \in \mathrm{Pa}(Y)} \mathbf{X}_{i_p} \hat{\beta}_{i_p} + \mathbf{R}_Y \mid \mathbf{X}_1, \cdots, \mathbf{X}_n, K\right)$$
$$= p\left(\mathbf{Y}' = \sum_{X_{i_p} \in \mathrm{Pa}(Y)} \mathbf{X}_{i_p} \hat{\beta}_{i_p} - \mathbf{R}_Y \mid \mathbf{X}_1, \cdots, \mathbf{X}_n, K\right).$$

Similarly, we apply Bayes' formula again:

$$p(\mathbf{Y}|\mathbf{X}_1, \ldots, \mathbf{X}_n, K) = \frac{p(K|\mathbf{X}_1, \ldots, \mathbf{X}_n, \mathbf{Y}) p(\mathbf{Y}|\mathbf{X}_1, \ldots, \mathbf{X}_n)}{p(K|\mathbf{X}_1, \ldots, \mathbf{X}_n)}.$$

And according to Lemma 1 and Conjecture 1,

$$p(\mathbf{Y}|\mathbf{X}_1, \ldots, \mathbf{X}_n) = p(\mathbf{Y}'|\mathbf{X}_1, \ldots, \mathbf{X}_n),$$
$$p(K|\mathbf{X}_1, \ldots, \mathbf{X}_n, \mathbf{Y}) = p(K|\mathbf{X}_1, \ldots, \mathbf{X}_n, \mathbf{Y}').$$

Combining the above gives the second result. □

Lemma 3. *Assume that Conjecture 1 holds. Let* $D = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n, \mathbf{Y})$ *be a random dataset from a directed Gaussian graphical system as defined in Assumption 1,* $\hat{\theta}_{LuPMI} = \sum_{p \in \mathcal{P}(X_1, Y)} \left[\left(\prod_{(X_i, X_j) \in p} \hat{A}_{ij} \right) \hat{\beta}_{i_p} \right]$ *be the output of Algorithm 1, and* $\hat{\theta}_{OLS} = (X_1^T X_1)^{-1} X_1^T Y$ *be the classical least squares estimator. It holds that,*

$$\mathbb{E}_D[\hat{\theta}_{OLS} | \hat{\theta}_{LuPMI}] = \hat{\theta}_{LuPMI}$$

Proof. Let $K = \{\hat{A}_{ij} \,|\, (X_i, X_j) \in \mathcal{E}\} \cup \{\hat{\beta}_{i_p} \,|\, X_{i_p} \in \mathrm{Pa}(Y)\}$ be the set of all the edge coefficient estimators obatined by Algorithm 1 for the Gaussian linear system of DAG G, and the matrix $B = (\mathbf{X}_1^T \mathbf{X}_1)^{-1} \mathbf{X}_1^T$. Firstly we prove $\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}} | K] = \hat{\theta}_{\mathrm{LuPMI}}$.

According to Assumption 1, there is a linear relationship between the root node X_1 and any mediating node X_i $(i \neq 1)$, which can be written similarly as $X_i = \mu_i^T X_1 + \epsilon$. Following the same analysis as Y, the least square estimator and the privileged information estimator are

$$\hat{\mu}_{i,OLS} = (\mathbf{X}_1^T \mathbf{X}_1)^{-1} \mathbf{X}_1^T \mathbf{X}_i = B \mathbf{X}_i$$
$$\hat{\mu}_{i,LuPMI} = \sum_{p \in \mathcal{P}(X_1, X_i)} \left(\prod_{(X_k, X_j) \in p} \hat{A}_{kj} \right)$$

Then we make an inductive proof: For the simplest case, that is, there are only two nodes with a single edge $X_1 \to X_2$, we have

$$\mathbb{E}_D[\hat{\mu}_{1,OLS} | K] = I = \hat{\mu}_{1,LuPMI}$$
$$\mathbb{E}_D[\hat{\mu}_{2,OLS} | K] = \hat{A}_{12} = \hat{\mu}_{2,LuPMI}$$

On this basis, we assume that $\mathbb{E}_D[\hat{\mu}_{i,OLS} | K] = \hat{\mu}_{i,LuPMI}$ holds for any X_i among the first k nodes $\{X_1, \ldots, X_k\}$ in the DAG. If a new node X_{k+1} is considered with all of its parent nodes in the set of the first k nodes (following the index rule $Pa(X_{k+1}) \subseteq \{X_1, \ldots, X_k\}$), we can deduce that

$$
\begin{aligned}
&\mathbb{E}_D[\hat{\mu}_{k+1,\mathrm{OLS}}|K] = \int p(\mathbf{X}_1,\ldots,\mathbf{X}_k,\mathbf{X}_{k+1}|K)\hat{\mu}_{k+1,\mathrm{OLS}}\, d\mathbf{X}_1\ldots d\mathbf{X}_{k+1}\\
&= \int p(\mathbf{X}_1,\ldots,\mathbf{X}_k|K)p(\mathbf{X}_{k+1}|\mathrm{Pa}(\mathbf{X}_{k+1}),K)B\mathbf{X}_{k+1}\, d\mathbf{X}_1\ldots d\mathbf{X}_{k+1}\\
&= \int p(\mathbf{X}_1,\mathbf{X}_2,\ldots,\mathbf{X}_k|K)B\\
&\times \underbrace{\left(\int p(\mathbf{X}_{k+1}|\mathrm{Pa}(\mathbf{X}_{k+1}),K)\mathbf{X}_{k+1}d\mathbf{X}_{k+1}\right)}_{=E(\mathbf{X}_{k+1}|\mathrm{Pa}(\mathbf{X}_{k+1}),K)} d\mathbf{X}_1\ldots d\mathbf{X}_k\\
&= \int p(\mathbf{X}_1,\ldots,\mathbf{X}_k|K)B\left[\sum_{X_i\in\mathrm{Pa}(X_{k+1})}\mathbf{X}_i\hat{A}_{i,k+1}\right] d\mathbf{X}_1\ldots d\mathbf{X}_{k+1}\\
&= \sum_{X_i\in\mathrm{Pa}(X_{k+1})}\int p(\mathbf{X}_1,\ldots,\mathbf{X}_k|K)B\mathbf{X}_i\hat{A}_{i,k+1}d\mathbf{X}_1\ldots d\mathbf{X}_k\\
&= \sum_{X_i\in\mathrm{Pa}(X_{k+1})}\int p(\mathbf{X}_1,\ldots,\mathbf{X}_i|K)\\
&\times p(\mathbf{X}_{i+1},\ldots\mathbf{X}_k|\mathbf{X}_1,\ldots,\mathbf{X}_i,K)B\mathbf{X}_i\hat{A}_{i,k+1}\, d\mathbf{X}_1\ldots d\mathbf{X}_k\\
&= \sum_{X_i\in\mathrm{Pa}(X_{k+1})}\int p(\mathbf{X}_1,\ldots,\mathbf{X}_i|K)\\
&\times \underbrace{\left(\int p(\mathbf{X}_{i+1},\ldots,\mathbf{X}_k|\mathbf{X}_1,\ldots,\mathbf{X}_i,K)d\mathbf{X}_{i+1}\ldots d\mathbf{X}_k\right)}_{=1} B\mathbf{X}_i\hat{A}_{i,k+1}\, d\mathbf{X}_1\ldots d\mathbf{X}_i\\
&= \sum_{X_i\in\mathrm{Pa}(X_{k+1})}\underbrace{\left(\int p(\mathbf{X}_1,\ldots,\mathbf{X}_i|K)B\mathbf{X}_i\, d\mathbf{X}_1\ldots d\mathbf{X}_k\right)}_{=\mathbb{E}_D[\hat{\mu}_{k,\mathrm{OLS}}|K]}\hat{A}_{i,k+1}\\
&= \sum_{X_i\in\mathrm{Pa}(X_{k+1})}\left[\sum_{p\in\mathcal{P}(X_1,X_i)}\left(\prod_{(X_k,X_j)\in p}\hat{A}_{kj}\right)\right]\hat{A}_{i,k+1}.
\end{aligned}
$$

Due to the property of DAG, any path from node X_1 to node X_{k+1} can be viewed as the combination of the path from node X_1 to a certain parent node of node X_{k+1} and the edge from the parent node to node X_{k+1}. Thus, the set of all paths from X_1 to X_{k+1}, $\mathcal{P}(X_1,X_{k+1})$, can be expressed as the union of the sets of all paths from X_1 through any parent node of X_{k+1} to X_{k+1}:

$$
\mathcal{P}(X_1,X_{k+1}) = \bigcup_{X_i\in\mathrm{Pa}(X_{k+1})}\left\{p\cup\{(X_i,X_{k+1})\}\mid p\in\mathcal{P}(X_1,X_i)\right\},
$$

which means that the last equation in the derivation of $\mathbb{E}_D[\hat{\mu}_{k+1,\mathrm{OLS}}|K]$ is actually equivalent to

$$
\mathbb{E}_D[\hat{\mu}_{k+1,\mathrm{OLS}}|K] = \sum_{p\in\mathcal{P}(X_1,X_{k+1})}\left(\prod_{(X_k,X_j)\in p}\hat{A}_{kj}\right).
$$

So far, we have completed induction and recursion and can prove that the above estimator applies to all nodes X_i. Therefore, for node Y, based on the same analysis

above, we get

$$\begin{aligned}\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}}|K] &= \sum_{X_{i_p}\in \mathrm{Pa}(Y)} \left[\sum_{p\in\mathcal{P}(X_1,X_{i_p})} \left(\prod_{(X_k,X_j)\in p} \hat{A}_{kj} \right) \right] \hat{\beta}_{i_p} \\ &= \sum_{p\in\mathcal{P}(X_1,Y)} \left[\left(\prod_{(X_i,X_j)\in p} \hat{A}_{ij} \right) \hat{\beta}_{i_p} \right] \\ &= \hat{\theta}_{\mathrm{LuPMI}} \ .\end{aligned}$$

Thus, with the law of total expectation,

$$\begin{aligned}\mathbb{E}_D[\hat{\theta}_{OLS}|\hat{\theta}_{LuPMI}] &= \mathbb{E}_{K|\hat{\theta}_{LuPMI}}[\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}}|K,\hat{\theta}_{LuPMI}]] \\ &= \mathbb{E}_{K|\hat{\theta}_{LuPMI}}[\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}}|K]] \\ &= \mathbb{E}_{K|\hat{\theta}_{LuPMI}}[\hat{\theta}_{LuPMI}] \\ &= \hat{\theta}_{LuPMI}.\end{aligned}$$

□

Lemma 2 presents the computation of the conditional expectation relation between arbitrary nodes in a linear Gaussian system using the edge coefficient estimators of Algorithm 1 (i.e., $\hat{A}_{ij}$ and $\hat{\beta}_{i_p}$). Lemma 3, on the other hand, shows that given a dataset D, the LuPMI estimator $\hat{\theta}_{\mathrm{LuPMI}}$ is able to provide a more accurate conditional expectation than the OLS estimator $\hat{\theta}_{\mathrm{OLS}}$, which takes the form $\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}}|\hat{\theta}_{\mathrm{LuPMI}}] = \hat{\theta}_{\mathrm{LuPMI}}$. These two lemmas provide the necessary theoretical support for the proof of Theorem 1 by standard Rao-Blackwell arguments. As a result, we give the proof of Theorem 1 as follows:

Theorem 1. *Assume that Conjecture 1 holds. Let $D = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n, \mathbf{Y})$ be a random dataset from a known directed Gaussian graphical system as defined in Assumption 1, $\hat{\theta}_{LuPMI}$ be the output of Algorithm 1, and $\hat{\theta}_{OLS} = (X_1^T X_1)^{-1} X_1^T Y$ be the classical least squares estimator. With isotropic noise defined in Assumption 1, $\hat{\theta}_{LuPMI}$ is unbiased, and*

$$MSE(\hat{\theta}_{LuPMI}) = MSE(\hat{\theta}_{OLS}) - \mathbb{E}_D[\,Tr(\,Cov(\hat{\theta}_{OLS}|\hat{\theta}_{LuPMI}))],$$

Further, it holds for the expected risk that over any samples (X_1, Y),

$$\bar{R}(\hat{\theta}_{LuPMI}) = \bar{R}(\hat{\theta}_{OLS}) - \mathbb{E}_{D,X_1}[\,Var_{\hat{\theta}_{OLS}}((\hat{\theta}_{OLS}, X_1)|\hat{\theta}_{LuPMI})].$$

Proof. First, according to the law of total expectation and Lemma 3, we can get

$$\begin{aligned}\mathbb{E}_D[\hat{\theta}_{OLS}] &= \mathbb{E}_D[\mathbb{E}_D(\hat{\theta}_{OLS}|\hat{\theta}_{LuPMI})] \\ &= \mathbb{E}_D[\hat{\theta}_{LuPMI}].\end{aligned}$$

Since $\hat{\theta}_{OLS}$ is an unbiased estimator with $\mathbb{E}_D[\hat{\theta}_{OLS}] = \theta$, it can be proved that $\hat{\theta}_{LuPMI}$ is also an unbiased estimator.

Second, following the same idea as the classical proof of the Rao-Blackwell theorem, we get

$$\begin{aligned}
\mathrm{MSE}(\hat{\theta}_{\mathrm{LuPMI}}) &= \mathbb{E}_D\left[||\hat{\theta}_{\mathrm{LuPMI}} - \theta||^2\right] \\
&= \mathbb{E}_D\left[||\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}}|\hat{\theta}_{\mathrm{LuPMI}}] - \theta||^2\right] \\
&= \mathbb{E}_D\left[||\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}} - \theta|\hat{\theta}_{\mathrm{LuPMI}}]||^2\right] \\
&= \mathbb{E}_D\left[\sum_{i=1}^{d}(\mathbb{E}_D[\hat{\theta}_{\mathrm{OLS}}^{(i)} - \theta^{(i)}|\hat{\theta}_{\mathrm{LuPMI}}])^2\right] \\
&= \mathbb{E}_D\left[\sum_{i=1}^{d}\left(\mathbb{E}_D[(\hat{\theta}_{\mathrm{OLS}}^{(i)} - \theta^{(i)})^2|\hat{\theta}_{\mathrm{LuPMI}}] - \mathrm{Var}[\hat{\theta}_{\mathrm{OLS}}^{(i)}|\hat{\theta}_{\mathrm{LuPMI}}]\right)\right] \\
&= \mathbb{E}_D\left[\mathbb{E}_D\left[||\hat{\theta}_{\mathrm{OLS}} - \theta||^2|\hat{\theta}_{\mathrm{LuPMI}}\right]\right] - \mathbb{E}_D\left[\sum_{i=1}^{d}\mathrm{Var}[\hat{\theta}_{\mathrm{OLS}}^{(i)}|\hat{\theta}_{\mathrm{LuPMI}}]\right] \\
&= \mathrm{MSE}(\hat{\theta}_{\mathrm{OLS}}) - \mathbb{E}_D\left[\mathrm{Tr}(\mathrm{Cov}[\hat{\theta}_{\mathrm{OLS}}|\hat{\theta}_{\mathrm{LuPMI}}])\right],
\end{aligned}$$

where $\hat{\theta}_{\mathrm{OLS}}^{(i)}$ represents the i-th component of the vector $\hat{\theta}_{\mathrm{OLS}}$ (the same as $\theta^{(i)}$). In order to obtain this last result, we consider that for any estimator $\hat{\theta}_{\mathrm{LuPMI}}$,

$$\begin{aligned}
\bar{R}(\hat{\theta}_{\mathrm{OLS}}) &= \mathbb{E}_D[R(\hat{\theta}_{\mathrm{LuPMI}})] \\
&= \mathbb{E}_D[\mathbb{E}_{X_1,Y}[(\hat{\theta}_{\mathrm{OLS}}^{\top}X_1 - Y)^2]] \\
&= \mathbb{E}_{X_1}[\mathbb{E}_D[\mathbb{E}_{Y|X_1}[(\hat{\theta}_{\mathrm{OLS}}^{\top}X_1 - Y)^2 \mid X_1]]].
\end{aligned} \tag{3.6}$$

In the last step of the above derivation, we use the fact that the distribution of the independent variable X_1 is separate from the dataset D. Then, due to the unbiased property of $\hat{\theta}_{\mathrm{OLS}}$, and applying our model $Y = \theta^{\top}X_1 + \epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma^2)$, we get

$$\begin{aligned}
\mathbb{E}_D[\mathbb{E}_{Y|X_1}[(\hat{\theta}_{\mathrm{OLS}}^{\top}X_1 - Y)^2]] &= \mathbb{E}_D\left[\mathbb{E}_{\hat{\theta}_{\mathrm{OLS}}}[(\hat{\theta}_{\mathrm{OLS}}^{\top}X_1 - \theta^{\top}X_1)^2]\right] + \sigma^2 \\
&= \mathbb{E}_D\left[\mathbb{E}_{\hat{\theta}_{\mathrm{OLS}}}[\langle\hat{\theta}_{\mathrm{OLS}} - \theta, X_1\rangle^2]\right] + \sigma^2,
\end{aligned} \tag{3.7}$$

where $\langle\cdot,\cdot\rangle$ represents the inner product of the two vectors in the triangular brackets. Then, since $\hat{\theta}_{\mathrm{LuPMI}}$ is also an unbiased estimator of θ,

$$\begin{aligned}
\mathbb{E}_D[R(\hat{\theta}_{\mathrm{LuPMI}})] &= \mathbb{E}_{X_1}\left[\mathbb{E}_D[\langle\hat{\theta}_{\mathrm{LuPMI}} - \theta, X_1\rangle^2]\right] + \sigma^2 \\
&= \mathbb{E}_{X_1}\left[\mathbb{E}_D\left[\langle\mathbb{E}_{\hat{\theta}_{\mathrm{OLS}}}[\hat{\theta}_{\mathrm{OLS}} \mid \hat{\theta}_{\mathrm{LuPMI}}] - \theta, X_1\rangle^2\right]\right] + \sigma^2 \quad \text{(by Lemma 2)} \\
&= \mathbb{E}_{X_1}\left[\mathbb{E}_D\left[\mathbb{E}_{\hat{\theta}_{\mathrm{OLS}}}[\langle\hat{\theta}_{\mathrm{OLS}} - \theta, X_1\rangle \mid \hat{\theta}_{\mathrm{LuPMI}}]^2\right]\right] + \sigma^2 \\
&= \mathbb{E}_{X_1}\left[\mathbb{E}_D\left[\mathbb{E}_{\hat{\theta}_{\mathrm{OLS}}}\left[\langle\hat{\theta}_{\mathrm{OLS}} - \theta, X_1\rangle^2|\hat{\theta}_{\mathrm{LuPMI}}\right]\right]\right] \\
&\quad - \mathbb{E}_{D,X_1}\left[\mathrm{Var}_{\hat{\theta}_{\mathrm{OLS}}}\left[(\hat{\theta}_{\mathrm{OLS}} - \theta, X_1)|\hat{\theta}_{\mathrm{LuPMI}}\right]\right] + \sigma^2
\end{aligned} \tag{3.8}$$

Eventually, combining the last step of Eq.3.8 with Eq.3.6 and Eq.3.7 yields the final result:

$$\mathbb{E}_D[R(\hat{\theta}_{\mathrm{LuPMI}})] = \mathbb{E}_D[R(\hat{\theta}_{\mathrm{OLS}})] - \mathbb{E}_{D,X_1}\left[\mathrm{Var}_{\hat{\theta}_{\mathrm{OLS}}}\left[(\hat{\theta}_{\mathrm{OLS}} - \theta, X_1)|\hat{\theta}_{\mathrm{LuPMI}}\right]\right]$$

which concludes the proof. □

Remark 1. *Since* $Tr(Cov[\hat{\theta}_{OLS}|\hat{\theta}_{LuPI}])$ *as the trace of a covariance matrix must be greater than or equal to 0, Theorem 1 actually states that* $MSE(\hat{\theta}_{LuPI}) \leq MSE(\hat{\theta}_{OLS})$. *As we have proved in Section 2.6,* $\mathrm{MSE}(\hat{\theta}) = \mathrm{Var}(\hat{\theta}) + \mathrm{Bias}(\hat{\theta})^2$, *and since* $\hat{\theta}_{LuPI}$ *and* $\hat{\theta}_{OLS}$ *are both unbiased estimators, Theorem 1 also proves that* $Var(\hat{\theta}_{LuPI}) \leq Var(\hat{\theta}_{OLS})$.

3.3.4 Special case

In general, our effectiveness theory suggests that LuPMI method should at least achieve no worse performance than OLS method with respect to the MSE and the expected risk, in an ideal case where Assumption 1 is satisfied. From Theorem 1, it is clear that

$$\mathrm{MSE}(\hat{\theta}_{\mathrm{OLS}}) - \mathrm{MSE}(\hat{\theta}_{\mathrm{LuPMI}}) = \mathbb{E}_D[\mathrm{Tr}(\mathrm{Cov}(\hat{\theta}_{\mathrm{OLS}} \mid \hat{\theta}_{\mathrm{LuPMI}}))] \geq 0.$$
$$\hat{R}(\hat{\theta}_{\mathrm{OLS}}) - \hat{R}(\hat{\theta}_{\mathrm{LuPMI}}) = \mathbb{E}_{D,X_1}[\mathrm{Var}_{\hat{\theta}_{\mathrm{OLS}}}((\hat{\theta}_{\mathrm{OLS}}, X_1) \mid \hat{\theta}_{\mathrm{LuPMI}})] \geq 0.$$

In most DAGs, the performance of the LuPMI algorithm will be better than that of the OLS algorithm, which means that the difference value of MSE in the above formula will be strictly greater than 0. However, in some special cases, we will get the result that $\hat{\theta}_{\mathrm{LuPMI}}$ and $\hat{\theta}_{\mathrm{OLS}}$ are exactly equal, for example, in a very dense DAG.

Before discussing the impact of graph density on the performance of the LuPMI algorithm, we first give the definition of a dense graph in this study. For a DAG with n nodes, the maximum number of edges is $n(n-1)/2$. Therefore, a DAG is said to be dense if the number of edges is close to this maximum. The densest DAG can be constructed in the following way: first, initialize a graph with n nodes without any edges; second, add edges from node X_1 to all other nodes; then, each time, select a node with out-degree 0 (except Y), add edges from it to all other nodes with out-degree 0, repeat this step until all nodes except Y have out-degree greater than 0. Finally, the total number of edges added is

$$(n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}.$$

Such densest DAGs are also referred to as full DAGs in the context of our work. As the simplest full DAG, an example of three variables is shown in Fig.3.2. In the

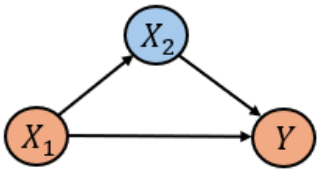


Figure 3.2: A simple dense DAG, where X_1 is the independent variable, X_2 is the mediating variable, and Y is the target variable

following, we will show that in the simplest dense DAG example, the theoretical MSE difference between the OLS estimator and the LuPI estimator is zero, i.e.

$\mathbb{E}_D[Tr(\mathrm{Cov}(\hat{\theta}_{\mathrm{OLS}} \mid \hat{\theta}_{\mathrm{LuPMI}}))] = 0$. According to Assumption 1, the relationship between variables in the system can be expressed as

$$\begin{aligned} X_2 &= A_{12}^T X_1 + \epsilon_1, \quad \text{where } \epsilon_1 \sim \mathcal{N}(0, \sigma_1^2) \\ Y &= \beta_1^T X_1 + \beta_2^T X_2 + \epsilon_Y, \quad \text{where } \epsilon_Y \sim \mathcal{N}(0, \sigma_y^2) \end{aligned}$$

For the convenience in deriving the relevant results, it is assumed that the variables X_1, X_2, Y are scalars. This implies that their data matrices, $\mathbf{X}_1, \mathbf{X}_2, \mathbf{Y}$, are vectors, and that the inner products between these data matrices will be scalar. Consequently, the product of the inner products is commutative. Furthermore, the least squares method allows us to calculate the estimators of all coefficients as follows:

$$\begin{aligned} \hat{A}_{12} &= (X_1^T X_1)^{-1} X_1^T X_2, \\ \begin{pmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \end{pmatrix} &= \begin{pmatrix} X_1^T X_1 & X_1^T X_2 \\ X_2^T X_1 & X_2^T X_2 \end{pmatrix}^{-1} \begin{pmatrix} X_1^T Y \\ X_2^T Y \end{pmatrix} \\ &= \frac{1}{C} \begin{pmatrix} X_2^T X_2 & -X_1^T X_2 \\ -X_2^T X_1 & X_1^T X_1 \end{pmatrix} \begin{pmatrix} X_1^T Y \\ X_2^T Y \end{pmatrix}, \end{aligned}$$

where $C = X_1^T X_1 \cdot X_2^T X_2 - (X_1^T X_2)^2$ is a constant scalar value. Equivalently,

$$\begin{aligned} \hat{\beta}_1 &= \frac{1}{C}(X_2^T X_2 X_1^T Y - X_1^T X_2 X_2^T Y) \\ \hat{\beta}_2 &= \frac{1}{C}(X_1^T X_1 X_2^T Y - X_2^T X_1 X_1^T Y) \end{aligned}$$

Therefore, it can be demonstrated that the LuPMI estimator and the OLS estimator are identical in this case by applying the commutativity of scalar multiplication. This is achieved through the following steps:

$$\begin{aligned} \hat{\theta}_{\mathrm{LuPMI}} &= \hat{\beta}_1 + \hat{A}_{12}\hat{\beta}_2 \\ &= \frac{1}{C}\left[X_2^T X_2 X_1^T Y - X_1^T X_2 X_2^T Y + X_1^T X_2 X_1^T Y - (X_1^T X_1)^{-1} X_1^T X_2 X_2^T X_1 X_1^T Y\right] \\ &= \frac{1}{C}(X_1^T X_1)^{-1}\left[(X_1^T X_1) X_2^T X_2 X_1^T Y - X_1^T X_2 X_2^T X_1 X_1^T Y\right] \\ &= \frac{1}{C}(X_1^T X_1)^{-1} C X_1^T Y \\ &= (X_1^T X_1)^{-1} X_1^T Y \\ &= \hat{\theta}_{\mathrm{OLS}}. \end{aligned}$$

Thus, we have shown that in the simplest full DAGs, which is also called as "triangular structures", the results of the two estimators are identical when the variables are scalars. Although the above theoretical proof is difficult to generalize to vector systems of more than 1 dimension and more complex DAGs, it illustrates the possibility that privileged information loses its usefulness. Related experimental results are presented in Sections 4.2.1 and 4.2.2, where we find that privileged information does not lead to any performance improvement in the directed GGM model with a full causal DAG.

3.4 Distillation

The original distillation method is applied to the classification task performed by a neural network, and thus has a loss function for discrete variables, as shown in Section 2.7. Lopez-Paz et al. proposed a learning method that unifies distillation and privileged information, which is also used for classification[5]. Their approach uses privileged information from training to guide the distillation process, which improves the performance of the student model. However, the idea of weighting can also be applied to regression tasks, with a different form of loss function. Our theory is grounded in the concept of distillation for regression tasks introduced by Karlsson et al. in their work with privileged time series information[4]. For the directed GGM model, an appropriate loss function is the squared loss function. Thus, the distillation loss function has the following form:

$$\mathcal{L}(\hat{\theta}) = \lambda \|Y_{soft} - X_1\hat{\theta}\|_2^2 + (1-\lambda)\|Y - X_1\hat{\theta}\|_2^2, \tag{3.9}$$

where $\lambda \in [0,1]$ and Y is the soft label, i.e., the predictions made by the teacher model. Accordingly, the goal of regression is to find an optimal solution

$$\hat{\theta} = \arg\min_{\hat{\theta}} \mathcal{L}(\hat{\theta}) \tag{3.10}$$

In our case, the learning method using the privileged information is adopted as the teacher model in knowledge distillation, i.e. $Y_{soft} = X_1\hat{\theta}_{\text{LuPMI}}$. Based on the above assumptions and consideration, we propose the following theorem:

Theorem 2. *Let $\hat{\theta}_{LuPMI}$ be the output of Algorithm 1 and $\hat{\theta}_{OLS} = \left(\mathbf{X}_1^\top \mathbf{X}_1\right)^{-1} \mathbf{X}_1^\top \mathbf{Y}$. With $\hat{\mathbf{Y}}_{soft} = \mathbf{X}_1\hat{\theta}_{LuPMI}$, the solution $\hat{\theta}_{Dist}$ to* (3.10) *holds that*

$$\hat{\theta}_{Dist} = \lambda\hat{\theta}_{LuPMI} + (1-\lambda)\hat{\theta}_{OLS}, \tag{3.11}$$

where $\lambda \in [0,1]$. And under the Assumption 1, it holds that

$$MSE(\hat{\theta}_{LuPMI}) \leq MSE(\hat{\theta}_{Dist}) \leq MSE(\hat{\theta}_{OLS}) \tag{3.12}$$

Proof. To obtain the optimal solution in Eq.(3.10), first calculate the derivative of the loss function $\mathcal{L}(\hat{\theta})$ and set it to zero by following the steps

$$\begin{aligned}
\frac{\partial \mathcal{L}(\hat{\theta})}{\partial \hat{\theta}} &= 2\lambda X_1^\top (X_1\hat{\theta} - Y_{soft}) + 2(1-\lambda)X_1^\top (X_1\hat{\theta} - Y) \\
&= 2X_1^\top X_1\hat{\theta} - \left[2\lambda X_1^\top Y_{soft} + 2(1-\lambda)X_1^\top Y\right] \\
&= 0.
\end{aligned}$$

Solving the above equation by algebraic transformation results in

$$\begin{aligned}
\hat{\theta}_{\text{Dist}} &= (X_1^\top X_1)^{-1} X_1^\top \left[\lambda Y_{soft} + (1-\lambda)Y\right] \\
&= \lambda(X_1^\top X_1)^{-1} X_1^\top Y_{soft} + (1-\lambda)(X_1^\top X_1)^{-1} X_1^\top Y \\
&= \lambda\hat{\theta}_{\text{LuPMI}} + (1-\lambda)\hat{\theta}_{\text{OLS}}.
\end{aligned}$$

This proves Eq.(3.11). To show Eq.(3.12), the MSE of each component of $\hat{\theta}_{\text{Dist}}$ can be decomposed into bias and variance according to its statistical property, i.e.

$$\text{MSE}(\hat{\theta}^{(i)}_{\text{Dist}}) = \text{Bias}(\hat{\theta}^{(i)}_{\text{Dist}})^2 + \text{Var}(\hat{\theta}^{(i)}_{\text{Dist}}) = \text{Var}(\hat{\theta}^{(i)}_{\text{Dist}}),$$

where $\hat{\theta}^{(i)}_{\text{LuPI}}$ represents the i-th component of the vector $\hat{\theta}_{\text{LuPMI}}$. Then we use Eq.(3.11) again to decompose the variance:

$$\begin{aligned}\text{Var}(\hat{\theta}^{(i)}_{\text{Dist}}) &= \text{Var}\left(\lambda\hat{\theta}^{(i)}_{\text{OLS}} + (1-\lambda)\hat{\theta}^{(i)}_{\text{LuPMI}}\right) \\ &= \lambda^2\text{Var}(\hat{\theta}^{(i)}_{\text{OLS}}) + (1-\lambda)^2\text{Var}(\hat{\theta}^{(i)}_{\text{LuPMI}}) + 2\lambda(1-\lambda)\text{Cov}(\hat{\theta}^{(i)}_{\text{OLS}}, \hat{\theta}^{(i)}_{\text{LuPMI}})\end{aligned} \tag{3.13}$$

For the last part, consider the complete covariance matrix $\text{Cov}(\hat{\theta}_{\text{OLS}}, \hat{\theta}_{\text{LuPMI}})$. Applying the law of total covariance, we get

$$\begin{aligned}\text{Cov}(\hat{\theta}_{\text{OLS}}, \hat{\theta}_{\text{LuPMI}}) = \mathbb{E}_{\hat{\theta}_{\text{LuPMI}}}&\left[\text{Cov}(\hat{\theta}_{\text{OLS}}, \hat{\theta}_{\text{LuPMI}} \mid \hat{\theta}_{\text{LuPMI}})\right] \\ &+ \text{Cov}_{\hat{\theta}_{\text{LuPMI}}}\left[\mathbb{E}(\hat{\theta}_{\text{OLS}} \mid \hat{\theta}_{\text{LuPMI}}), \mathbb{E}(\hat{\theta}_{\text{LuPMI}} \mid \hat{\theta}_{\text{LuPMI}})\right]\end{aligned}$$

Further, according to the definition of covariance, the first term can be written as

$$\mathbb{E}_{\hat{\theta}_{\text{LuPMI}}}\left[\mathbb{E}\left[(\hat{\theta}_{\text{OLS}} - \mathbb{E}[\hat{\theta}_{\text{OLS}} \mid \hat{\theta}_{\text{LuPMI}}])(\underbrace{\hat{\theta}_{\text{LuPMI}} - \mathbb{E}[\hat{\theta}_{\text{LuPMI}} \mid \hat{\theta}_{\text{LuPMI}}]}_{0})^\top\right]\right] = \mathbf{0}_{d\times d},$$

where $\mathbf{0}_{d\times d}$ is the zero matrix of dimension $d \times d$. Therefore,

$$\begin{aligned}\text{Cov}(\hat{\theta}_{\text{OLS}}, \hat{\theta}_{\text{LuPMI}}) &= \text{Cov}_{\hat{\theta}_{\text{LuPMI}}}\left[\mathbb{E}(\hat{\theta}_{\text{OLS}} \mid \hat{\theta}_{\text{LuPI}}), \mathbb{E}(\hat{\theta}_{\text{LuPMI}} \mid \hat{\theta}_{\text{LuPMI}})\right] \\ &= \text{Cov}(\hat{\theta}_{\text{LuPIM}}) \quad \text{(by Lemma 2)} \\ &= \text{Var}(\hat{\theta}_{\text{LuPMI}})\end{aligned}$$

It follows that the diagonal elements of $\text{Cov}(\hat{\theta}_{\text{OLS}}, \hat{\theta}_{\text{LuPMI}})$ are also the same as those of $\text{Var}(\hat{\theta}_{\text{LuPMI}})$. Substituting this result into Eq.3.13 gives us

$$\begin{aligned}\text{Var}(\hat{\theta}^{(i)}_{\text{Dist}}) &= \lambda^2\text{Var}(\hat{\theta}^{(i)}_{\text{OLS}}) + (1-\lambda)^2\text{Var}(\hat{\theta}^{(i)}_{\text{LuPMI}}) + 2\lambda(1-\lambda)\text{Var}(\hat{\theta}^{(i)}_{\text{LuPMI}}) \\ &= \lambda^2\text{Var}(\hat{\theta}^{(i)}_{\text{OLS}}) + (1-\lambda^2)\text{Var}(\hat{\theta}^{(i)}_{\text{LuPMI}})\end{aligned}$$

According to the above reasoning process, $\text{Var}(\hat{\theta}^{(i)}_{\text{Dist}})$ can actually be regarded as the weighted average of $\text{Var}(\hat{\theta}^{(i)}_{\text{OLS}})$ and $\text{Var}(\hat{\theta}^{(i)}_{\text{LuPMI}})$ due to the fact that $\lambda \in [0, 1]$. And because it is known from Remark 1 that $\text{Var}(\hat{\theta}^{(i)}_{\text{LuPMI}}) \leq \text{Var}(\hat{\theta}^{(i)}_{\text{OLS}})$, we can conclude that

$$\text{Var}(\hat{\theta}^{(i)}_{\text{LuPMI}}) \leq \text{Var}(\hat{\theta}^{(i)}_{\text{Dist}}) \leq \text{Var}(\hat{\theta}^{(i)}_{\text{OLS}}),$$

which, since all three estimators are unbiased, can be expressed as

$$\text{MSE}(\hat{\theta}^{(i)}_{\text{LuPMI}}) \leq \text{MSE}(\hat{\theta}^{(i)}_{\text{Dist}}) \leq \text{MSE}(\hat{\theta}^{(i)}_{\text{OLS}}).$$

Finally, according to the definition $\text{MSE}(\hat{\theta}_{\text{LuPMI}}) = \sum_i \text{MSE}(\hat{\theta}^{(i)}_{\text{LuPMI}})$, the inequality is proved:

$$\text{MSE}(\hat{\theta}_{\text{LuPMI}}) \leq \text{MSE}(\hat{\theta}_{\text{Dist}}) \leq \text{MSE}(\hat{\theta}_{\text{OLS}})$$

□

4

Experiments

In order to validate the efficiency of the algorithm and to explore the limitations, we conducted separate experiments on both synthetic and real datasets. This chapter provides the setup of the experiment and the results obtained. Section 4.1 outlines the procedures for synthesizing datasets, the preprocessing of real datasets, and the libraries and functions utilized in the process. The results on the synthetic dataset and the real dataset are presented in Sections 4.2 and 4.3, respectively.

4.1 Experimental setup

All experiments were performed in Python, where the functions used for regression and model evaluation were sourced from the classic Scikit-learn machine learning library[16]. Since our algorithm involves estimating the coefficients of each edge in the DAG, every single estimate is computed by the *LinearRegression* function in Scikit-learn. Three main metrics have been used to evaluate the performance of the OLS algorithm and the LuPMI algorithm, including MSE (*mean squared error*), R^2 score (*the coefficient of determination*), and relative MSE. In this context, the relative MSE is used only in the synthetic dataset, which scales the MSE of the parameter estimator $\hat{\theta}$ relative to the real parameter θ, expressed as $\|\theta - \hat{\theta}\|_2^2 / \|\theta\|_2^2$.

Synthetic data

In accordance with the Assumption 1, we construct a system of vectors conforming to a directed Gaussian graph model for synthetic datasets based on a given DAG. First, for each edge, a random matrix of coefficients is generated which represents the linear relationship between parent and child nodes in the DAG. In order to ensure the numerical stability of the generated matrix, the spectral radius (i.e., the mode of the largest eigenvalue) of the matrix is adjusted to a pre-defined value to control the growth of dependencies in the system. For the root node X_1, which has no parent node, the data is sampled from the standard normal distribution. For the other nodes, values are generated from a linear combination of the parent nodes along with a Gaussian noise.

Coefficient scaling and noise scaling are also introduced in the experiments and are calculated based on Equation 3.1. The actual linear coefficients and noise of the input variable X_1 on the scalar target variable Y are obtained by computing

linear transformations on all paths from X_1 to Y. If coefficient scaling is enabled, all coefficients are scaled to a given range of standard deviations; noise scaling is adjusted according to the cumulative effect of noise on different paths. This mechanism has been used in exploring the impact of the complexity of the real causal graph on the performance of the LuPMI algorithm.

Communities and Crime

The Communities and Crime dataset is a well-known dataset widely used in social science and machine learning, and is downloaded from the UCI Machine Learning Repository[17]. The dataset contains economic and social data from different communities in the United States, covering a wide range of characteristics such as income level, education, employment status, family structure, etc. The full dataset contains 1994 instances, each representing a U.S. community, with a total of 127 features. The most common task on this dataset is to explore the potential relationship between community characteristics and crime rates, and thus we use crime rates as the target variable.

Notably, many of the features in this dataset have a high linear correlation with each other. To analyze the causal relationships between these features, a clustering method based on their correlation matrices is used. In particular, we defined a rule that creates links between features when the correlation between them exceeds a certain threshold (set to 0.79 in this experiment). This method clusters all features based on the correlations, and the final clusters have relatively strong correlations within them and weak correlations with other clusters. The effectiveness of this clustering method can be visualized by a heat map of the correlation matrix. In Fig.4.1, dark squares appearing along the diagonal indicate clusters of features with a high level of internal correlation. Each dark square represents a different cluster, indicating a set of features that are more closely correlated with each other than with features in other clusters. Cluster analysis is performed on the 60 features most highly correlated with crime rates. A total of 8 disjoint clusters are finally obtained after removing the clusters that are not very relevant to the predicted target (with the highest correlation coefficient is less than 0.4). In our model, the underlying causality of the data can be inferred through the internal connections of these clusters.

4.2 Synthetic dataset

The experiments on the synthetic dataset are divided into two main sections. The purpose of the first section was to validate the effectiveness theory developed in Chapter 3, in particular the performance gap between the OLS algorithm and the LuPMI algorithm as demonstrated by Theorem 1. In addition, we explored how this gap varies with experimental settings. The second section deals with possible biases in DAG estimation, which are categorized into two cases: ignoring some real edges and mis-estimating redundant edges.

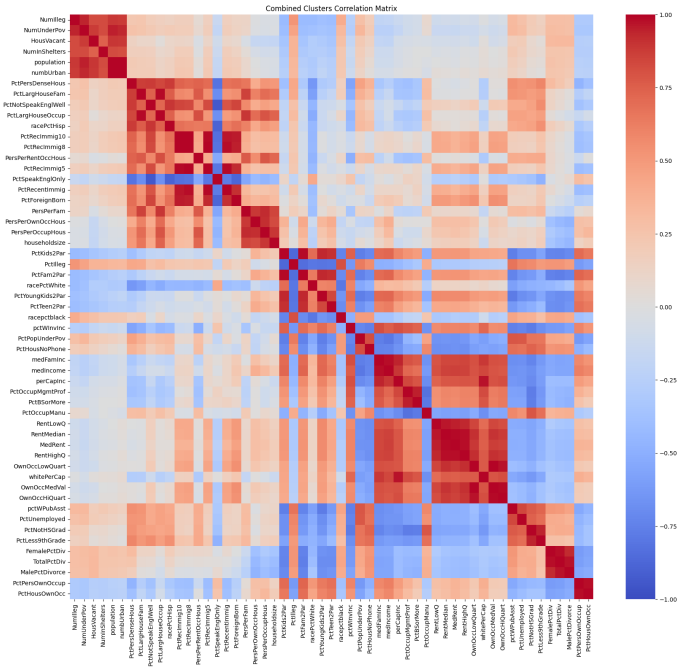


Figure 4.1: Correlation matrix of clusters of filtered features with strong target correlation. The heatmap shows the correlations between a subset of features that are most highly correlated with the target variable, crime rate. The color scale from blue (negative correlation) to red (positive correlation) indicates the level of correlation between the features.

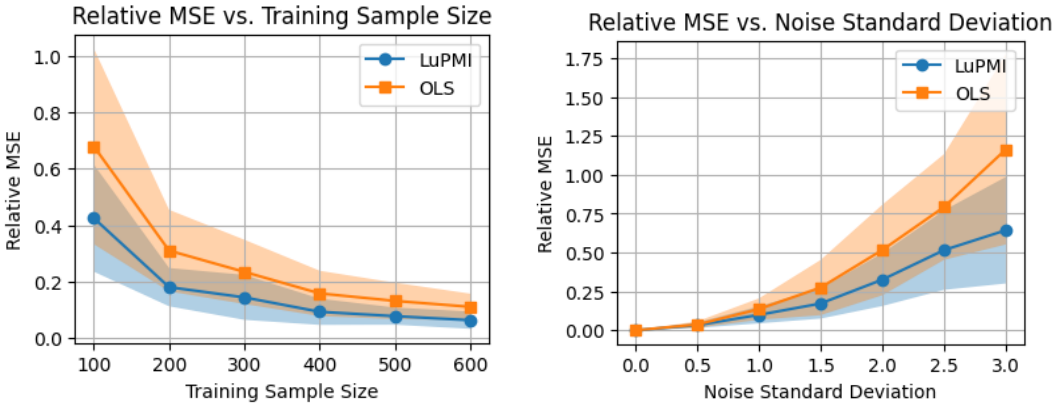
4.2.1 Validation of theory

This section shows the relative MSE differences between OLS and LuPMI under different experimental settings. The following parameters are adjusted: the number of samples in the training set, the noise level of the system, and the density of the graph.

Sample size and noise level

This part of the experiment was carried out by varying the experimental parameters, only one parameter was adjusted in each experiment, leaving the other parameters unchanged. In order to achieve stable results, each experiment was repeated 50 times, with each iteration generating a new dataset based on the same system. In both experiments, all variables have dimension d of 10 and the input variable X_1 is distributed with a standard multivariate normal distribution, i.e. $X_1 \sim \mathcal{N}(0, I_d)$.

Figures 4.2a and 4.2b present how the number of training samples and the standard deviation of the noise affect the relative MSEs of the OLS and LuPMI estimators, respectively. It can be observed that the LuPMI estimator consistently exhibits a relative MSE that is not higher than that of the OLS estimator, as predicted by Theorem 1. Furthermore, the differences in the relative MSE are found to vary significantly with the tuning of the experimental parameters that are shown. In Fig.4.2a, the performance of the two methods improves as the number of samples



(a) Varying number of training samples *(b)* Varying standard deviation of system noise

Figure 4.2: Experimental results for the effect of adjusting one experimental parameter at a time. The relative MSE is used as metric, and color-filled areas indicate intervals from minus to plus one standard deviation calculated over 50 replicate experiments.

grows, but the gap between the two methods diminishes, which suggests that the LuPI algorithm brings greater improvement than the OLS algorithm when the number of samples is relatively small. Fig.4.2b shows that both methods perform worse as system noise increases, which is to be expected. Notably, there is no performance gap between the two methods when the standard deviation of the system noise is 0. This can be interpreted to mean that when the system is noiseless, the OLS estimate and the LuPMI estimate have 0 variance and are themselves unbiased, so that their relative MSEs are both 0.

Graph density

In Section 3.3.4, we analyze a special case of the LuPMI algorithm when the DAG in the model is very dense, i.e., the number of edges is close to or equal to the maximum $\frac{n(n-1)}{2}$. The objective of this experiment is to investigate the impact of graph density on the LuPMI algorithm. Fig.4.3 depicts the experimental results in a system with 5 variables, including the independent variable X_1, the mediating variables X_2, X_3, X_4, and the target variable Y. The minimum number of edges is 4, corresponding to a sequence, and the maximum number of edges is 10, corresponding to a full graph. In this experiment, the dimension d of each variable is 50, the number of training samples is 350, and the standard deviation of the combined noise ϵ_0 of the system is 6.5. The experiment is repeated 30 times for each edge number. In each iteration, a DAG with five nodes and the given number of edges is randomly created, and a Gaussian dataset is generated based on this DAG.

The superiority of the LuMPI method over the OLS method is also evident in this experiment. Furthermore, this discrepancy in performance is more pronounced in systems with a smaller number of edges for a given number of nodes, parameter size, and noise level. Given that a smaller number of edges for a given number of nodes indicates a sparser graph, this also implies that the LuPMI algorithm will result in a greater performance improvement relative to the OLS algorithm when the true

causal DAG of the system is sparser.

An important observation in Fig.4.3 is that when the number of edges in the DAG reaches the maximum number of edges, the relative MSEs of the LuPMI method and the OLS method are exactly equal. This result corresponds to our discussion in Section 3.3.4. It is due to the fact that the coefficients computed by LuPMI estimates and OLS estimates are always identical in full graphs with a maximum number of edges, regardless of how random the dataset is.

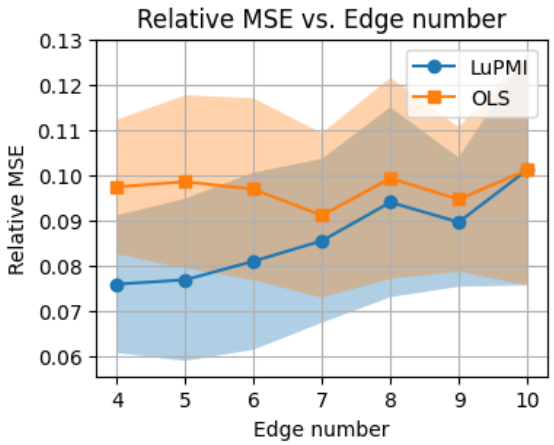
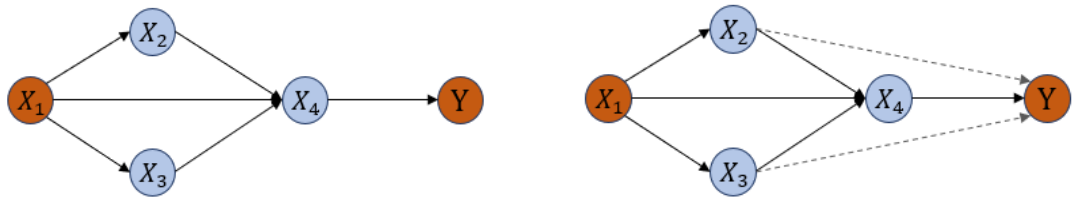


Figure 4.3: The performance of two algorithms in a 5-node system when the number of edges is varied. The relative MSE is used as metric, and color-filled areas indicate intervals from minus to plus one standard deviation calculated over 50 replicate experiments.

4.2.2 Graph estimation bias

For the ideal situation where the estimated graph and the actual graph are completely consistent, the effectiveness of the algorithm has been proven in Theorem 1. In implementation, however, it is challenging to learn the exact graph of the relationship between variables. As a result, there is often a discrepancy between the predicted graphs on which the LuPMI algorithm is based and the actual graphs of the relationships between the variables, which may introduce bias and variance changes to the algorithm. Bias in graph estimation comes from two cases, the first is when some edges that really exist are ignored, and the second is when some edges that don't really exist are mis-estimated. So theoretically there are two main instances to consider: the estimated graph G' is a subset of the real graph G, i.e. $G' \subseteq G$, or the real graph G is a subset of the estimated graph G', i.e. $G \subseteq G'$. The second case of $G \subseteq G'$ can be interpreted as the estimated graph having more edges than the real graph, as shown in Fig.4.6.

The experiments in this section are concerned with the estimation bias of the causality graph. Therefore, they are divided into the two main cases where the estimated graph G' is a subset of the real graph G, $G' \subseteq G$, or the real graph G is a subset of the estimated graph G', $G \subseteq G'$. The first case means that some real edges are ignored, and the second case means that some non-existent edges are mis-estimated.



(a) Actual graph *(b)* Estimated graph

Figure 4.4: An example of $G \subseteq G'$, where the solid black lines represent real relationships between variables and the dashed gray lines represent relationships between variables that are artificially estimated but do not actually exist.

In the following results, the LuPMI method was implemented using graphs with estimation bias and its performance was compared with that of the LuPMI method using real graphs and the OLS method. The symbol "LuPMI_estimated" denotes the LuPMI method with an estimated graph, while "LuPMI_real" denotes the LuPMI method with a real graph.

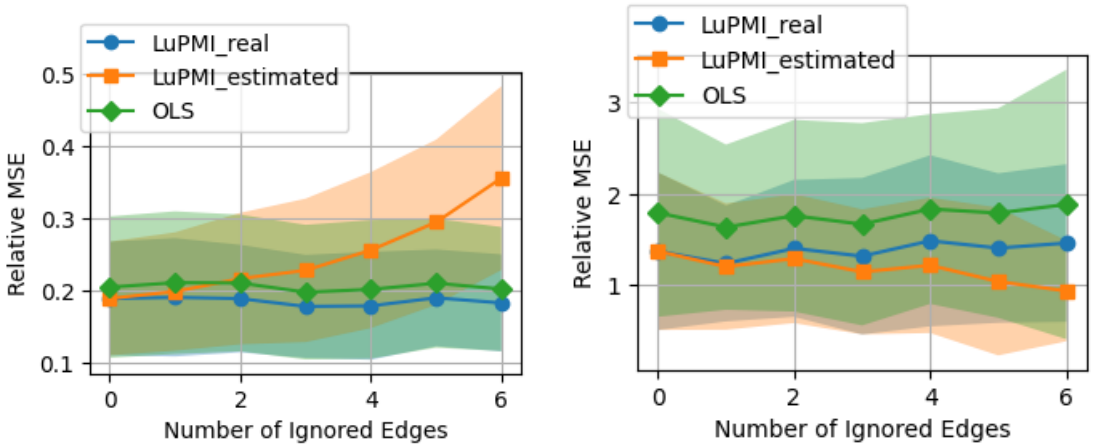
Ignoring edges

In this part, we designed and implemented two experiments to investigate the performance under different noise levels. In the first experiment, a system with a high noise level is set up. Specifically, the dimension of the system is 10, the noise standard deviation is 4.0, and the number of training samples is 100. The real DAG model has 7 nodes and contains 14 edges. During the experiment, 20 iterations of the experiment are conducted, and in each iteration, 6 edges are iteratively removed from the real graph. In each iteration, LuPMI estimation is performed on the real and estimated graphs with the edges removed, and the performance of the two LuPMI estimators as well as the OLS estimator are plotted separately. In the second experiment, we evaluated the performance of the three estimators at low noise levels. The dimension of this system is set to 50, the noise level is set to a standard deviation of 1.0, and the number of samples is 350. The rest of the settings and experimental steps are the same as in the first experiment.

Ignoring edges may introduce bias due to lack of causality. However, as the number of edges considered decreases, there may be a corresponding decrease in variance. Thus, there is a trade-off between bias and variance. Experiments conducted on systems with different levels of noise are designed to explore this trade-off relationship. In particular, the experiments visualize the dynamics between the bias introduced by ignoring edges and the variance reduced by reducing edges by observing the change in relative MSE as the number of edges in the graph is iteratively ignored.

As shown in Fig. 4.5a, the relative MSE of the estimated graph-based LuPMI increases significantly with the number of edges removed at low noise levels. This phenomenon can be attributed to the fact that the ignored edges introduce a significant bias that outweighs the variance reduction as the number of ignored edges increases, causing the overall relative MSE to increase. However, in the system with high noise level

(as shown in Fig. 4.5b), the estimated graph-based LuPMI method shows superior performance compared to the other two methods. This result may be due to the fact that at a high noise level, the variance dominates the MSE, so reducing the number of edges effectively decreases the variance and still improves the overall performance of the model, although some bias is introduced as a result. This suggests that the variance reduction effect of reducing the number of edges may be greater than the negative effect of increased bias in a high noise system.



(a) Low noise level

(b) High noise level

Figure 4.5: Relative MSE vs. number of ignored edges for different estimators. The plot compares the performance of three estimators (LuPMI_real, LuPMI_estimated, and OLS) in terms of relative MSE as the number of ignored edges increases in two separate systems with distinct noise levels. The shaded regions represent the interval of one standard deviation around the mean.

Mis-estimating edges

In the following experiment, we use a dataset synthesized by the same directed Gaussian graphical model as in the previous section. Similarly, the experiment is repeated 20 times. Its feature dimension is set to 10, there is noise in the system with a standard deviation of 2.5. The number of training samples is always 100. In each repetition of the experiment, a real graph with 7 nodes and 9 edges is randomly generated, and then the dataset is generated based on the real graph. To observe the effect of mis-estimating edges on the LuPMI method, some edges are iteratively added from the real graph. In each iteration, one more edge is added on the basis of the modified graph from the previous iteration, for a total of 6 edges added from the DAG. The performance of the three estimators is evaluated in the same way as in the experiment with the ignored edges. And the results are shown in Fig.4.6.

In this case, the estimated coefficients of the LuPMI method are expected to remain unbiased, meaning that the expected values of the estimated coefficients should be consistent with the real parameters. However, it is important to recognize that although no systematic bias is introduced by this method, there is a corresponding increase in variance. Thus, the increased variance shows up as an increase in the

relative mean square error (MSE) as the number of mis-estimated edges in the estimation graph increases. Since the real underlying graph is known to contain 9 edges in the 7-node system(with a maximum edge number of 21), the estimation graph is actually full when the number of mis-estimated edges reaches 12. In this particular case, it can be observed that the estimation results of the LuPMI and OLS methods are exactly the same. Thus, it is shown that LuPMI estimation does not provide an advantage at this point, while still making use of auxiliary information.

Nevertheless, an important argument can be made that the LuPMI method consistently shows at least equivalent or even better performance than the OLS method, even when some edges are mis-estimated. This claim holds despite the observed increase in variance. Thus, in this case, the LuPMI method ensures that the estimation quality does not fall below the baseline established by the OLS method.

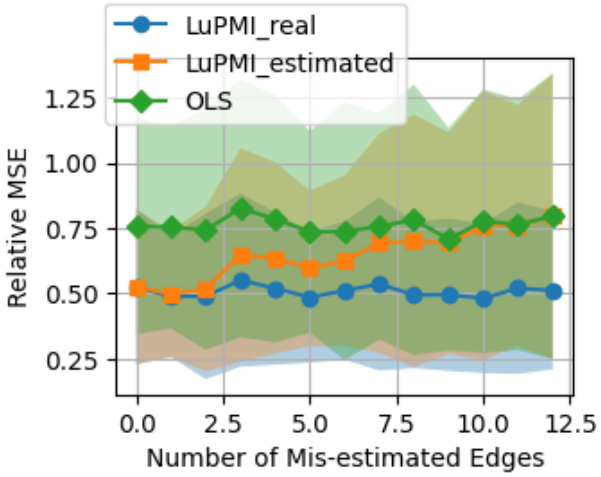


Figure 4.6: Relative MSE vs. number of mis-estimated edges for different estimators. The plot compares the performance of three estimators (LuPMI_real, LuPMI_estimated, and OLS) in terms of relative MSE as the number of mis-estimated edges increases. The shaded regions represent the interval of one standard deviation around the mean.

4.3 Real dataset

In this section, we conduct experiments on real datasets to explore the performance of the LuPMI method on data with complex causal relationships. The main focus is on the Communities and Crime dataset[17]. Figure 4.7 illustrates the graphical structure of causality in the model, showing the direct and indirect influence relationships between variables. There are 29 causal variables, 26 mediating variables, and 1 target variable in this model, with the mediating variables divided into 4 groups. The names of the features involved in particular are shown in Table 4.1.

The final results of the experiment are shown in Figure 4.8, following the above model. The LuPMI method performs best in terms of R^2 scores at all sample sizes, and its advantage is particularly evident at smaller sample sizes, which is consistent with the results of the experiments on the synthetic dataset. This suggests that LuPMI makes

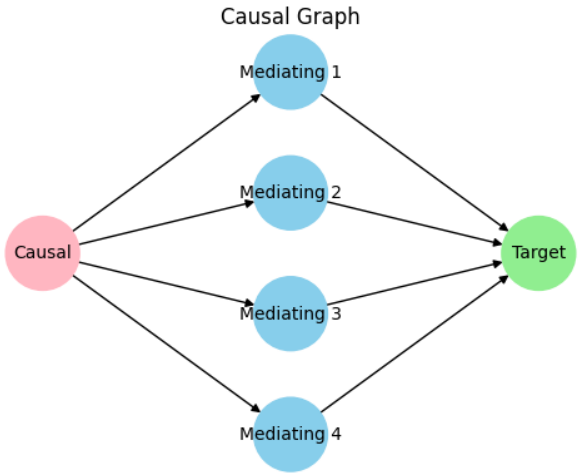


Figure 4.7: Causality graph that describes the relationship between the independent variables, mediating variables, and the target in the model. The pink node represent the independent variable, the green node represent the target, and the blue nodes represent the mediating variables.

better use of the extra privileged information to improve the model's generalization ability under small dataset conditions. The OLS method performs less well than LuPMI overall, although it improves with increasing training sample size. The shaded area shows the variance of the different algorithms at different sample sizes. It can be seen that the variance of LuPMI is smaller, which means that it has better stability at different sample sizes. In addition, since the LuPMI estimate in the distillation method used here has a weight factor of 0.5, the distillation estimate is actually an average of LuPMI and OLS. It is observed that in this case the performance of the distillation method is close to that of the LuPMI method, but slightly inferior to the LuPMI and superior to the OLS method. This result is consistent with the theoretical analysis presented in Section3.4. Thus, it also partially suggests that our causal model is more appropriate to the underlying structure between the variables in the data than a linear regression model without privileged information.

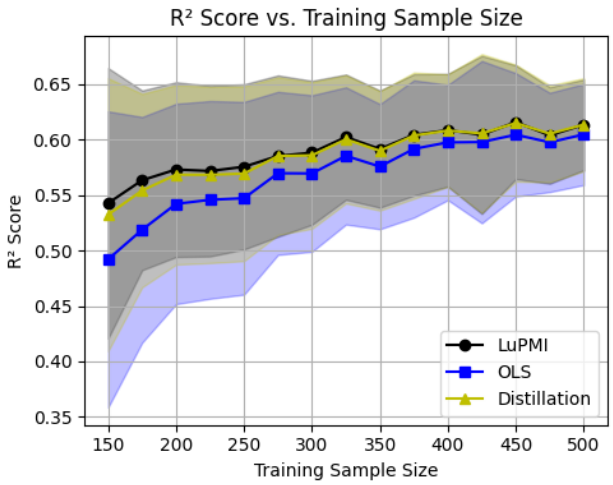


Figure 4.8: R^2 scores of different estimation methods versus training sample size. This line graph compares the R^2 scores of the three estimation methods (LuPMI, OLS and distillation) for different training sample sizes. The shaded regions represent the interval of one standard deviation around the mean.

Table 4.1: Features included in the groups of the model.

Causal Variables	Mediating Variables
	Group 1:
	population
numbUrban	PersPerFam
NumInShelters	racePctHisp
NumUnderPov	racePctWhite
householdsize	MedRent
PersPerOwnOccHous	medFamInc
PctForeignBorn	TotalPctDiv
PctSpeakEnglOnly	
PersPerRentOccHous	**Group 2:**
PctRecImmig10	HousVacant
PctLargHouseOccup	PctRecentImmig
PctLargHouseFam	PctNotSpeakEnglWell
racepctblack	PctIlleg
PctYoungKids2Par	RentLowQ
PctFam2Par	PctPopUnderPov
PctKids2Par	PctPersOwnOccup
OwnOccHiQuart	
whitePerCap	**Group 3:**
RentHighQ	NumIlleg
RentMedian	PctRecImmig5
PctOccupManu	PctPersDenseHous
PctOccupMgmtProf	OwnOccMedVal
medIncome	PctBSorMore
PctHousNoPhone	pctWPubAsst
pctWInvInc	
PctUnemployed	**Group 4:**
PctLess9thGrade	PersPerOccupHous
MalePctDivorce	PctRecImmig8
FemalePctDiv	PctTeen2Par
PctHousOwnOcc	OwnOccLowQuart
	perCapInc
	PctNotHSGrad

5

Discussion

The purpose of this chapter is to explore the theoretical and implementation limitations of this research and to suggest possible directions for extension and future work. First, we will analyze in detail the limitations of the current theoretical models and algorithm implementations, including limitations in terms of assumptions and application. These limitations provide us with opportunities for further improvement and optimization. Then, we will discuss how to overcome these limitations and increase the generality of the algorithms by introducing nonlinear models. Finally, we will propose specific directions for future work to improve the usefulness of our research results.

5.1 Limitations of theory and implementation

While our study relies on several key assumptions, especially the directed Gaussian graphical model (i.e., Gaussian Bayesian networks), experiments on real-world data show that even when these assumptions are not fully satisfied, the LuPMI algorithm still brings improvements over the OLS algorithm. However, it is still important to further explore these limitations in order to improve the performance and adaptability of the algorithm. By exploring the limitations of these assumptions and identifying situations where the model may have trouble, future work could focus on extending the algorithm to better handle situations where the underlying data structure is not in these idealized conditions.

On the one hand, although the Bayesian network model is a classical model for describing causality, its conditional independence assumptions are sometimes too strong to be satisfied. Section 4.2.2 discusses the problems caused by this limitation, exploring the effect of bias between the estimated causality graphs used in the algorithms and the true causality on the performance of the algorithms. The results of the experiment show that ignoring some real causal relationships in the estimation graph leads to biased parameter estimates, while the variance decreases. In contrast, if some non-existent causal relationships are mis-estimated, no bias is expected, but the variance will increase. Thus, in general, a bias in the estimation of true causality will always lead to an increase in the mean square error. However, since it is often difficult to obtain a true representation of the relationships in a dataset, some statistical methods are needed to make the estimates as accurate as possible. For example, in Section 4.3, causal relationships between variables are estimated

using correlation clustering and correlation matrices.

On the other hand, our model works under the assumption of linear and Gaussian noise, which introduces some restrictions. Assumption 1 expects a linear relationship between the independent variables and the outcome, and that the noise of the system follows a Gaussian distribution with isotropic and constant variance. The linear assumption may oversimplify the complex dependencies in the data, and the Gaussian noise assumption may not be able to fit the irregular noise structure. Despite these limitations, the properties of linear models make them interpretable and robust in many applications, especially when the underlying relationships are approximately linear or when interpretability is critical. While this is a rather idealized situation, these assumptions have significant practical value when dealing with a variety of real-world datasets, as evidenced by the success of our experiments in capturing meaningful causal relationships and achieving reliable predictions. However, real-world systems sometimes exhibit more complex nonlinear relationships, so extending our model to nonlinear mappings is both a hopeful and feasible direction for future work.

5.2 Potential extensions

For nonlinear systems, a possible extension would be to replace the linear relationship $\theta^T X_1$ in our model with $\theta^T \Phi(X_1)$, and then have the following relationship between the independent variable X_1 and target variable Y:

$$Y = \theta^T \Phi(X_1) + \epsilon_Y, \quad \text{where } \epsilon_Y \sim \mathcal{N}(0, \sigma_y^2).$$

In order to still be able to apply the theory of linear systems, the main discussion here is on the case of isotropic Gaussian noise. Here $\Phi(X_1)$ denotes any nonlinear relationship that may exist between the independent and dependent variables. For this model, neural networks provide an efficient means of approximating and representing complex nonlinear mappings. It is able to fit arbitrary nonlinear functions through a combination of layer structure and nonlinear activation functions. Hence, the mapping $\theta^T \hat{\Phi}(X_1)$ can be expressed as

$$\theta^T \hat{\Phi}(X_1) = \sum_{i=1}^{N} \theta_i \sigma(w_i^T X_1 + b_i),$$

Where N is the number of neurons in the hidden layer, α_i is the weight of the neuron in the output layer, and $w_i \in \mathbb{R}^n$ and $b_i \in \mathbb{R}$ are the weights and intercepts for the i neuron in the first hidden layer. $\sigma(\cdot)$ is the activation function, which is usually nonlinear. Each neuron $\sigma(w_i^T X_1 + b_i)$ generates a basic component function, and the output of the network is a weighted sum of these component functions. By adjusting the neuron weights α_i, w_i, and intercepts b_i, the network can be made to approximate the mapping $\theta^T \Phi(X_1)$ with any accuracy. Thus, as the number of neurons N increases, the network can flexibly fit more complex mappings. As an important theorem in the theory of neural networks, the Universal Approximation

Theorem[18], [19] shows that for any given $\epsilon > 0$, there exists a single hidden-layer neural network such that the output function $\theta^T \hat{\Phi}(X)$ of the network satisfies

$$\sup_{X_1 \in K} \left| \Phi(X_1) - \hat{\Phi}(X_1) \right| < \epsilon,$$

where $K \subset \mathbb{R}^n$ is a compact set and $\hat{\Phi}(X_1)$ is a function represented by a neural network. Therefore, according to the theorem, even if based on only a single hidden-layer neural network, we can approximate any continuous nonlinear function $\Phi(X_1)$ which is defined on the compact set by an appropriate loss function and gradient descent method.

5.3 Future work

In future work, we expect to extend the LuPMI algorithm to more types of datasets, especially time series datasets. Although the current experiments are mainly based on static datasets, in the real world many data have time-dependent correlations, which makes them at least partially explicit causal relationships. Therefore, future studies can design experiments to compare the performance differences between the LuPMI algorithm and the existing time-series algorithms, such as the LuPTS algorithm, when dealing with time-series data. By analyzing the performance of these algorithms on time series data, the adaptability of the LuPMI algorithm in different cases of data structures can be further verified.

In addition, there is still space to improve the performance of the model when dealing with nonlinear systems. Although the current study assumes that the data obeys linear relationships and Gaussian noise, many datasets do not fully satisfy these assumptions in practical applications. For this reason, it is useful to explore how the LuPMI algorithm can be extended to nonlinear systems, e.g. by introducing neural networks to capture more complex nonlinear relationships between features. Moreover, the combination of nonlinear activation functions and neural network models may provide the algorithm with greater expressiveness, thus improving its prediction accuracy in nonlinear scenarios.

6

Conclusion

In this thesis, we propose a Learning using Privileged Mediating Information (LuPMI) algorithm based on the directed Gaussian graphical model (directed GGM). By constructing a causal DAG model with X_1 as the independent variable and Y as the target variable, with several mediating variables $X_2, \ldots, X_n$ in between, this study analyzes in detail how the LuPMI algorithm outperforms the ordinary least squares (OLS) model in terms of statistical properties, given the known causal relationships. In theory, the Rao-Blackwell theorem and related lemmas are used to show that the LuPMI algorithm can effectively reduce the mean square error (MSE) and expected risk, and in most cases, the LuPMI algorithm has a smaller MSE and expected risk than the OLS model. Furthermore, it is discussed that in some specific cases, such as very dense DAGs, the performance of LuPMI and OLS may be identical, but overall, the LuPMI algorithm provides a solid improvement for causal linear Gaussian systems. Theoretical analysis shows that the use of the LuPMI algorithm can improve prediction performance in regression tasks, especially when the data contains a relatively sparse causal graph structure.

Then, in the experimental part, we verify the efficiency of the proposed algorithm and explore its limitations through experiments on synthetic and real datasets. First, in the synthetic dataset, the theoretical advantages of the LuPMI algorithm over the OLS algorithm are validated through the experiments of adjusting the parameters of the training set, such as sample size, noise level, and graph density. The experimental results show that the LuPMI algorithm consistently exhibits better or at least not inferior relative MSE than the OLS algorithm under different conditions, especially when the noise level is higher or the graph is sparser, the advantage of the LuPMI algorithm over the OLS algorithm is more obvious. In addition, for the error introduced by the graph estimation bias, we investigate the effect of ignoring the real edges and mis-estimating the redundant edges in two separate instances, and the results show that reducing the number of edges helps to reduce the variance and thus improve the overall performance of the algorithms in high-noise systems, while ignoring the real edges may significantly increase the bias in low-noise conditions. For the experiments on real datasets, we compare the LuPMI algorithm with the baseline OLS algorithm based on the Communities and Crime datasets. The results show that the LuPMI algorithm outperforms the OLS algorithm in terms of R^2 scores at all sample sizes, and performs particularly well at small sample sizes, which is consistent with the experimental results on synthetic data. Overall, the LuPMI algorithm not only performs well in using privileged information to improve model

generalization ability, but also shows better stability at small sample sizes, confirming its efficiency in dealing with complex causal data.

Bibliography

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] J. Pearl, *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2000.

[3] V. Vapnik and A. Vashist, "A new learning paradigm: Learning using privileged information," *Neural Networks*, vol. 22, no. 5, pp. 544–557, 2009, Advances in Neural Networks Research: IJCNN2009, ISSN: 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2009.06.042. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608009001130.

[4] R. K. A. Karlsson, M. Willbo, Z. Hussain, R. G. Krishnan, D. Sontag, and F. D. Johansson, *Using time-series privileged information for provably efficient learning of prediction models*, 2022. arXiv: 2110.14993 [cs.LG].

[5] D. Lopez-Paz, L. Bottou, B. Schölkopf, and V. Vapnik, *Unifying distillation and privileged information*, 2016. arXiv: 1511.03643 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1511.03643.

[6] B. Jung and F. D. Johansson, *Efficient learning of nonlinear prediction models with time-series privileged information*, 2023. arXiv: 2209.07067 [cs.LG].

[7] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[8] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques* (Adaptive computation and machine learning). MIT Press, 2009, ISBN: 9780262013192. [Online]. Available: https://books.google.co.in/books?id=7dzpHCHzNQ4C.

[9] V. Vapnik and R. Izmailov, "Learning using privileged information: Similarity control and knowledge transfer," *Journal of Machine Learning Research*, vol. 16, no. 61, pp. 2023–2049, 2015. [Online]. Available: http://jmlr.org/papers/v16/vapnik15b.html.

[10] R. Johnson and D. Wichern, *Applied multivariate statistical analysis*, 5. ed. Upper Saddle River, NJ: Prentice Hall, 2002, XVIII, 767, ISBN: 0130925535. [Online]. Available: http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+330798693&sourceid=fbw_bibsonomy.

[11] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*, 2nd ed. Springer, 2009. [Online]. Available: http://www-stat.stanford.edu/~tibs/ElemStatLearn/.

[12] D. Blackwell, "Conditional expectation and unbiased sequential estimation," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 105–110, 1947.

[13] C. R. Rao, "Information and the accuracy attainable in the estimation of statistical parameters," in *Breakthroughs in Statistics: Foundations and Basic*

Theory, S. Kotz and N. L. Johnson, Eds. New York, NY: Springer New York, 1992, pp. 235–247, ISBN: 978-1-4612-0919-5. DOI: 10.1007/978-1-4612-0919-5_16. [Online]. Available: https://doi.org/10.1007/978-1-4612-0919-5_16.

[14] R. Weber, *Statistics lecture notes: Lecture 03*, https://www.statslab.cam.ac.uk/~rrw1/lectures/statistics/lecture03.pdf, University of Cambridge, Statistical Laboratory, 2024.

[15] G. Hinton, O. Vinyals, and J. Dean, *Distilling the knowledge in a neural network*, 2015. arXiv: 1503.02531 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1503.02531.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[17] M. Redmond, *Communities and Crime*, UCI Machine Learning Repository, DOI: https://doi.org/10.24432/C53W3X, 2009.

[18] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989.

[19] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.