Causal Discovery
Finding Common Latent Causes

Master Thesis

Konstantina Xanthopoulou-Koukogia (s-2418312)

University of Twente (UT) Machine2learn (M2L)

Examination Committee
UT supervisor: Johannes Schmidt-Hieber
UT member: Hanyuan Hang
UT member: Felix Schweninnger
External (daily) supervisor: Bart Gips

June 2023

Abstract

In this project, we explore Causal Discovery with the presence of latent variables. We describe a method of testing covariance structures in order to distinguish causal from effect indicators in structural equation models. This is done by implementing a statistical hypothesis testing on simulated data. We analyse and compare different test statistics, including a bootstrap technique. We introduce the FOFC algorithm that checks quartets of measured variables in order to cluster them by their common cause, based on these statistical tests. Finally, we discuss how this could be used in more elaborate causal discovery algorithms such as the Copula PC.

Preface

This project is the final assignment of my masters program in Applied Mathematics (Mathematics of Data Science) in the University of Twente (UT). It was carried out in collaboration with Machine2learn, in the academic year 2022-2023.

I would like to thank my external supervisor, Dr. Bart Gips, for his patience, excellent guidance and support. I would also like to thank my UT supervisor, Dr. Johannes Schmidt-Hieber, for his insightful feedback and constructive criticism that helped me complete this project in the best way possible.
Finally, I want to thank my friends and family for their love and support.

Acknowledgement

This project is funded by the European Union under grant agreement No. 101057619. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Health and Digital Executive Agency (HADEA). Neither the European Union nor the granting authority can be held responsible for them. This work was also partly supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract No. 22.00115.

Contents

1 Introduction

Causality is a fundamental concept in the field of statistics and research, aiming to understand the relationships between variables and determine whether one variable influences another. It explores the idea of cause and effect in many scientific fields, such as medicine, neuroscience, psychology, social sciences and economics. The main goal is to obtain an answer to the question 'Does X cause Y?' or 'What are the effects of changing X on Y?'. This could lead to an estimation of the effect of smoking on lung cancer, of education on salaries, of carbon emissions on the climate. Understanding causality could lead to informed decisions by identifying the factors that have a direct impact on desired outcomes.

1.1 Background

In this project we focus on causal discovery within statistical models that contain latent variables. A latent variable in a statistical model is a random variable that is unmeasured (although not necessarily 'unmeasurable'). There are three main reasons for introducing latent variables into a statistical model. One reason is to include in the model features of interest that are not directly measurable, or were not measured. A second reason is that, in some circumstances, latent variables models can be used to construct estimators that are more efficient than those constructed from non-latent variable models. A third advantage is that, in some circumstances, latent variables models can be used to construct estimators of manipulation statistics that are unbiased, unlike those constructed from non-latent variable models [1].

In order to formulate a definition of *causality* we first need to explain what counfounding variables are.

Confounder: Confounding variables (also confounders or confounding factors) are a type of extraneous variable that influences both dependent and independent variables. If we undertake to estimate the effect of one variable X on another Y by examining the statistical association between the two, we ought to ensure that the association is not produced by factors other than the effect under study. The presence of spurious association – due, for example, to the influence of extraneous variables – is called confounding because it tends to confound our reading and to bias our estimate of the effect studied. Conceptually, therefore, we can say that X and Y are confounded when there is a third variable C that influences both X and Y; such a variable is then called a confounder of X and Y [2].
The existence of confounders is an important quantitative explanation why correlation does not imply causation (will be described below). In Figure 1 we use a graph to depict an example of a confound variable C, causing variables X and Y. Then, there might be a correlation between X and Y but it is not implied that X causes Y or vice versa.

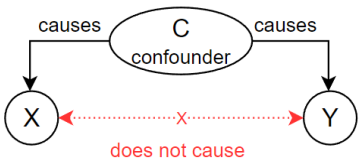


Figure 1: Example of confounder causing two variables, without forcing a causation among them

Now, the concept of *causality* can be introduced as follows:
For random variables X and Y, X directly causes Y if, with all confounders fixed in constant values, an intervention in X results in a change in Y, but an intervention in Y does not necessarily change X.

In addition, causality is not only about exploring whether two variables are causally related but also *how* they are. In order to describe causal relationships among variables, we use Structural Equation Models (SEMs).

Structural Equation Models (SEMs): SEMs are models that explain relationships between variables. SEMs can be viewed as general models of many commonly employed statistical models, such as analysis of variance, analysis of covariance, multiple regression, factor analysis, path analysis, econometric models of simultaneous equation and non-recursive modeling, multilevel modeling, and latent growth curve modeling.

Relationships among latent variables (or factors) and other variables in a SEM are structural relationships. Structural questions relate to the regression and correlational relationships among latent variables and among latent and observed variables. SEMs can include any combination of latent variables and observed variables. These variables can be measured or *latent*, that is, variables that cannot be measured directly but could be influential in other variables [3]. A simple example of such a SEM can be the following:

$$W := f_1(X) + \epsilon_1 \tag{1}$$
$$Z := f_2(X) + \epsilon_2 \tag{2}$$
$$Y := f_3(X, W) + \epsilon_3 \tag{3}$$

Here, variables W and Z are caused by variable X, and Y is caused by (X, W), and $\epsilon_1, \epsilon_2, \epsilon_3$ are noise terms.

To visualize the causal relations between variables we use Directed Acyclic Graphs (DAGs). Directed means that each edge has a defined direction, so that it represents a single directional flow from one vertex to another, and acyclic means that there are no loops in the graph, that is, starting from any vertex, following any edge that connects this vertex to another, there is no path in the graph to get back to that initial vertex. The DAG corresponding to the SEM we described above can be seen in Figure 2.

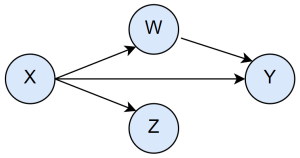


Figure 2: Example of Directed Acyclic Graph (DAG)

In this project, we consider only linear SEMs. That is, the functions that describe the relation among the variables will be linear, and therefore, these models are called *Linear Variable Models*. Taking the example used above, the functions in (1),(2),(3) will have the following form:

$$f_1(X) = a_1 X + b_1 + \epsilon_1$$
$$f_2(X) = a_2 X + b_2 + \epsilon_2$$
$$f_3(X, W) = a_3 X + a_4 W + b_3 + \epsilon_3$$

where $a_1, a_2, a_3, a_4, b_1, b_2, b_3$ are scalars and $\epsilon_1, \epsilon_2, \epsilon_3$ are independent noise variables that follow a Gaussian distribution.

Finally, the underlying goal is to draw conclusions about the causal relationships between variables in a system based on observed data. This process is called *Causal Inference* and it involves identifying the causes of an observed effect or predicting the effects of a potential cause, accounting for the influence of other variables that may also affect the relationship.

1.2 Goals

In this project we investigate causal structures that contain latent variables. The goal is to derive an efficient method for estimating the factor loadings in linear latent variables models. With the main objective being to obtain the loadings in a data-driven way, we apply a statistical hypothesis testing on the covariance structure of the observed variables of our model.

1.3 Structure of report

The rest of this paper is organized as follows. In Section 2, we start by describing existing methods to perform causal discovery using linear models and how to deal with latent variables. In Section 3, we will describe possible statistical tests that can be used for assessing the existence of common latent causes of measured variables. Section 4 will cover the implementation of these tests and analysis of their

application on simulated data. In Section 5, we will introduce an algorithm that systematically checks quartets of measured variables in order to group them by their common causes based on these statistical tests. Finally, we close with a general discussion and conclusion in Section 6.

2 Causal Discovery Methods

Causal discovery is the process of identifying causal relationships among variables in a system. It is a critical aspect of scientific research because understanding causal relationships allows us to make predictions and intervene in the system to achieve desired outcomes. Causal discovery can be challenging because causal relationships are often not directly observable, and there may be multiple possible causal structures that can explain the relationships among variables. There is a variety of methods used to obtain and analyse causal relationships from data, including statistical modeling techniques such as structural equation modeling, Bayesian networks, and causal inference algorithms.

2.1 PC Algorithm

Causal discovery aims to find an underlying directed acyclic graph (DAG), which represents direct causal relations between variables. It is a very popular approach for multivariate data analysis and therefore is widely studied in the past few years, resulting in various algorithms. The PC algorithm can be considered the reference causal discovery algorithm [4][5].

Before describing how the algorithm works, the definition of *d-separation* is needed.

Definition 1 (d-separated)**.** *For a graph G, if X and Y are vertices in G, $X \neq Y$, and W is a set of vertices in G not containing X or Y, then X and Y are d-separated given W in G if and only if there exists no undirected path U between X and Y, such that*
(i) every collider on U has a descendent in W and
(ii) no other vertex on U is in W [6].

The PC algorithm has two main steps. In the first step, it makes use of conditional independence tests using partial correlation based on Pearson correlations between variables (d-separation), to build the underlying DAG from observations. Starting from a complete undirected graph, it removes edges recursively according to the outcome of the conditional independence tests. This procedure yields an undirected graph, also called the skeleton. In the second step, after applying various edge orientation rules, the algorithm returns a partially directed graph to represent the underlying DAGs.

A simple example of these steps is depicted in Figure 3 [7]. In (A), we have the original true causal graph, that is, the desired output of the algorithm. By d-separation, this structure implies that X is independent of Y, written $X \perp\!\!\!\perp Y$, and that X and Y are each independent of W conditional on Z, written $(X, Y) \perp\!\!\!\perp W|Z$. Suppose when called, the statistical decision procedure finds these relations. PC assumes the Causal Markov and Faithfulness Conditions. These are:

Definition 2 (Causal Markov Condition)**.** *Let G be a causal graph with vertex set V and P be a probability distribution over the vertices in V generated by the causal structure represented by G. G and P satisfy the Causal Markov Assumption if and only if for every W in V, W is independent of $V \backslash \big(Descendants(W) \cup Parents(W)\big)$ given $Parents(W)$.*

Definition 3 (Faithfulness Condition)**.** *Let G be a causal graph and P a probability distribution generated by G. $< G, P >$ satisfies the Faithfulness Condition if and only if every conditional independence relation true in P is entailed by the Causal Markov Condition applied to G [6].*

Then, the algorithm is based on the fact that under these conditions, when there is no latent confounder, two variables are directly causally related (with an edge in between) if and only if there does not exist any subset of the remaining variables conditioning on which they are independent.

In (B), the algorithm starts by forming the complete undirected graph. In (C), after testing for conditional independence, the $X - Y$ edge is removed because $X \perp\!\!\!\perp Y$. In (D), similar to previous step, $X - W$ and $Y - W$ edges are removed because $X \perp\!\!\!\perp W|Z$ and $Y \perp\!\!\!\perp W|Z$. In (E), the graph results after finding the v-structures. Finally, (F) displays the final graph after applying orientation propagation, that is, for each triple of variables such that $A \to B - C$, and A and C are not adjacent, orient the edge $B - C$ as $B \to C$.

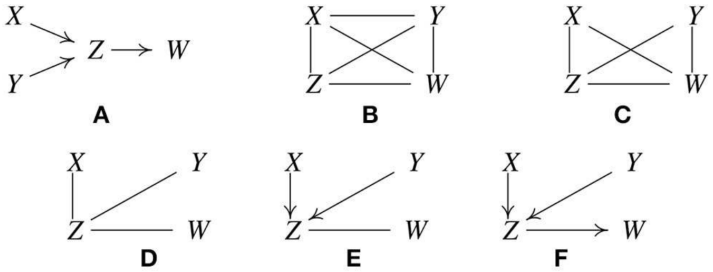


Figure 3: Illustration of how the (original) PC algorithm works [7]

2.2 Gaussian Copula Factor Model

In this section we describe the Gaussian copula factor model, that can be used for cases where a latent variable can be connected to either one or more observed variables.

Consider a latent random vector $\eta = (\eta_1, ..., \eta_k)^T$, a response random vector $Z = (Z_1, ..., Z_p)^T$ (will be referred to as *indicators*), and an observed random vector $Y = (Y_1, ..., Y_p)^T$, satisfying

$$\eta \sim \mathcal{N}(0, C) \tag{4}$$

$$Z = \Lambda\eta + \epsilon \tag{5}$$

$$Y_j = F^{-1}(\Phi[Z_j/\sigma(Z_j)]), \quad \forall j = 1, ..., p \tag{6}$$

where

$\Lambda = (\lambda_{ij})$ a $p \times k$ matrix of factor loadings $(k \leq p)$

$\epsilon \sim \mathcal{N}(0, C)$ the Gaussian noise with $D = diag(\sigma_1^2, ..., \sigma_p^2)$

$\sigma(Z_j)$ the standard deviation of Z_j and

$F^{-1}(t) = \inf\{x : F(x) \geq t\}$ the quantile function of a random variable with cumulative distribution function F

Φ the CDF of the standard normal distribution.

This model is called a *Gaussian Copula Factor Model* with correlation matrix C, factor loadings Λ and univariate margins F_j. An example of such a model is shown in Figure 4.

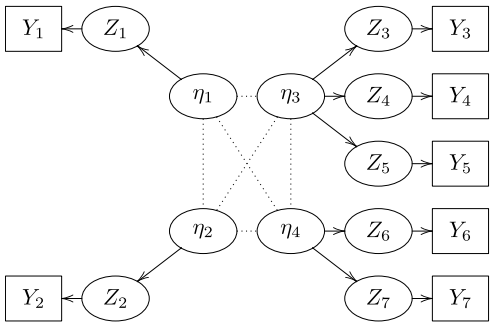


Figure 4: Example of Gaussian Copula Factor Model [8]

In this example, our model is a combination of a Gaussian factor model (from η to Z) and a Gaussian copula model (from Z to Y). In the special case of a factor having a single response (thus a single observed variable), e.g., $\eta_1 \to Z_1 \to Y_1$, it reduces to a Gaussian copula model where we set $\lambda_{11} = 1$ and $\epsilon_1 = 0$, thus $Y_1 = F_1^{-1}(\Phi[\eta_1])$.
This could be used, for instance, to infer the causal structure among measured variables and 4 latent concepts related to Attention Deficit Hyperactivity Disorder (ADHD): inattention, hyperactivity, impulsivity, forgetfulness. In the typical design for questionnaires, one tries to get a grip on a latent concept

through a particular set of well designed questions, which implies that a latent factor (η) in our model is connected to multiple indicators/questions (Z) while an indicator is only used to measure a single factor (Y). This model is called a pure measurement model.

Definition 4 (Pure Measurement Model)**.** *A pure measurement model is a measurement model in which each observed variable has only one latent parent, and no observed parent. That is, it is a tree beneath the latents.*

Copula PC Algorithm

Here, we introduce the *Copula PC* algorithm for causal discovery. It is based on a two-step approach. The first step applies Gibbs sampling on rank-based data to obtain samples of the posterior over the correlation matrices. From these posterior samples, we can extract the expected value and estimate the effective number of datapoints. In the second step, using these quantities we can apply the standard PC algorithm to obtain a Completed Partially Directed Acyclic Graph (CPDAG) [9].

Overall, even though the Copula PC Algorithm is flexible in the sense that it can deal with different data types and missing data, it still requires the user to give the factor loadings (Λ in equation (5)). Estimating these loadings will be described in Section 3.

2.3 Linear Latent Variable Models

As mentioned before, when trying to estimate causal relationships among variables, some of them can be unobserved, thus, latent. When these latent variables are believed to influence only one recorded variable directly, they are usually modeled as noise. In case they influence two or more measured variables directly, the intent is to identify them and their influence [10].

There are multiple ways to visualize the relations among variables. One example is the Factor Graph [11], that is, a bipartite graph representing the factorization of a function.
Let us consider a function that factorizes as follows:

$$g(X_1, X_2, X_3) = f_1(X_1)f_2(X_1, X_2)f_3(X_1, X_2)f_4(X_2, X_3)$$

then the corresponding Factor Graph is shown on Figure 5.

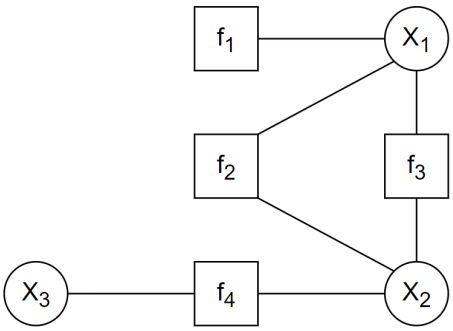


Figure 5: Example of Factor Graph

However, when trying to investigate the causal relation among variables, Directed Acyclic Graphs (DAGs) are most commonly used. In DAGs, the variables are represented by the vertices and the causal relations by the (directed) edges between them. This type of graph is not only used as a visualization of the causal relations but also as a tool for further causal inference. A simple example of such a graph can be seen in Figure 6, where there are three latent variables L_1, L_2, L_3 and nine measured variables $X_1, ..., X_9$.

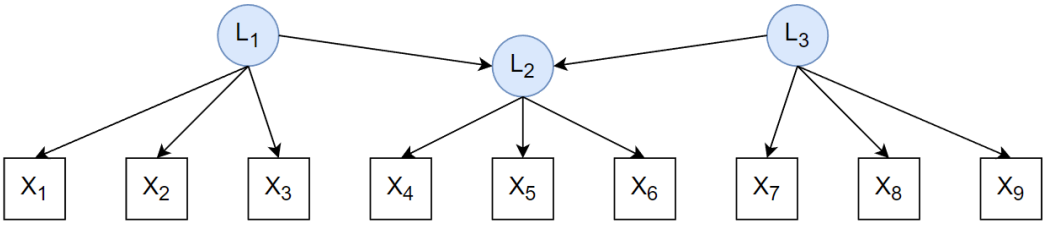


Figure 6: Example of DAG

The latent structure, the dependencies of measured variables on individual latent variables, and the linear dependency of the measured variables on their parents and (unrepresented) independent noises in Figure 6 imply a pattern of constraints on the covariance matrix among the X variables. In our example, X_1, X_2, X_3 have zero covariances with X_7, X_8, X_9, since the first three variables are caused by L_1, the latter three variables are caused by L_3, but L_1 and L_3 do not effect each other. However, when considering X_1, X_2, X_3 and any one of X_4, X_5, X_6, then three constraints are implied, since X_1, X_2, X_3 are directly caused by L_1, which also influences L_2, and L_2 is the cause of X_4, X_5, X_6.

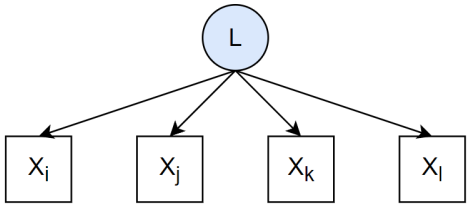
Introducing the following lemma will help us explain these constraints.

Lemma 2.1. *For every four variables X_i, X_j, X_k, X_l caused by the same latent variable L, following a linear model, the following equality holds:*

$$\sigma_{ij}\sigma_{kl} - \sigma_{ik}\sigma_{jl} = 0 \tag{7}$$

where σ_{ij} is the covariance of variables X_i and X_j.

Proof. Let us consider the following graph, where X_i, X_j, X_k, X_l are the observed variables and L is the common latent variable.



The linear equations that describe the causal relations in this graph are the following:

$$\begin{aligned} X_i &= b_i L + \epsilon_i \\ X_j &= b_j L + \epsilon_j \\ X_k &= b_k L + \epsilon_k \\ X_l &= b_l L + \epsilon_l \end{aligned}$$

where b_i, b_j, b_k, b_l are scalars and $\epsilon_i, \epsilon_j, \epsilon_k, \epsilon_l$ are noise terms.

Considering that $mean(X_i) = mean(X_j) = mean(L) = 0$ and $var(L) = 1$, we have:

$$\begin{aligned}
\sigma_{ij} &= E[X_iX_j] - \cancelto{0}{E[X_i]E[X_j]} \\
&= E[X_iX_j] \\
&= E[(b_iL + \epsilon_i)(b_jL + \epsilon_j)] \\
&= E[b_ib_jL^2 + \epsilon_jb_iL + \epsilon_ib_jL + \epsilon_i\epsilon_j] \\
&= E[b_ib_jL^2] + E[\epsilon_jb_iL] + E[\epsilon_ib_jL] + \cancelto{0}{E[\epsilon_i\epsilon_j]} \\
&= b_ib_jE[L^2] + b_i\cancelto{0}{E[\epsilon_jL]} + b_j\cancelto{0}{E[\epsilon_iL]} \\
&= b_ib_j\cancelto{1}{var(L)} \\
&= b_ib_j.
\end{aligned}$$

Similarly we have:

$$\begin{aligned}
\sigma_{kl} &= b_kb_l \\
\sigma_{il} &= b_ib_l \\
\sigma_{jk} &= b_jb_k \\
\sigma_{ik} &= b_ib_k \\
\sigma_{jl} &= b_jb_l
\end{aligned}$$

Therefore

$$\sigma_{ij}\sigma_{kl} - \sigma_{ik}\sigma_{jl} = b_ib_jb_kb_l - b_ib_kb_jb_l = 0$$

□

These constraints will be referred to as *tetrad constraints* and for example, for variables X_1, X_2, X_3, X_4, they will be defined as follows:

$$\rho_{12}\rho_{34} = \rho_{14}\rho_{23} \tag{8}$$
$$\rho_{14}\rho_{23} = \rho_{13}\rho_{24} \tag{9}$$
$$\rho_{13}\rho_{24} = \rho_{12}\rho_{34} \tag{10}$$

where ρ_{ij} is the Pearson correlation between X_i and X_j[1].

It is important to notice that any two of the three constraints above entail the third [10].

Also, assuming that the variables $X_1, ..., X_9$ are normalized, thus, $var(X_1) = ... = var(X_9) = 1$, we have that:

$$corr(X_i, X_j) = \frac{cov(X_i, X_j)}{\sigma_{X_i}\sigma_{X_j}} = \frac{cov(X_i, X_j)}{\sqrt{var(X_i)}\sqrt{var(X_j)}} = cov(X_i, X_j)$$

for every X_i, X_j in $X = \{X_1, ..., X_9\}$. Therefore, the constraints above hold as well if correlations are substituted by covariances. That is,

$$\sigma_{12}\sigma_{34} = \sigma_{14}\sigma_{23} \tag{11}$$
$$\sigma_{14}\sigma_{23} = \sigma_{13}\sigma_{24} \tag{12}$$
$$\sigma_{13}\sigma_{24} = \sigma_{12}\sigma_{34}. \tag{13}$$

In this project, we focus on *1-pure* variable models with 4 measured variables per latent variable. That is, models that are both *pure* and *1-Factor*. The definitions of *pure* and *1-Factor* variable models are given below:

[1] Pearson correlation: $\rho_{X_iX_j} = corr(X_i, X_j) = \frac{cov(X_i,X_j)}{\sigma_{X_i}\sigma_{X_j}} = \frac{E[(X_i-\mu_{X_i})(X_j-\mu_{X_j})]}{\sigma_{X_i}\sigma_{X_j}}$, if $\sigma_{X_i}\sigma_{X_j} > 0$

Definition 5 (1-Factor variable model)**.** *1-Factor variable model is a model where each (observed) variable has precisely 1 latent parent and might have observed parents, in addition to its "error" variable [12].*

Definition 6 (pure variable model)**.** *A pure variable model is a variable model in which each observed variable has only one latent parent. That is, it is a tree beneath the latent [10].*

Therefore, a *1-pure* variable model with 4 variables is a model where each observed variable has at most one latent parent, no observed parents and no correlated errors. Such a model can be seen in Figure 7, where L_1 and L_2 are the latent and $X_1, ..., X_8$ are the measured variables.

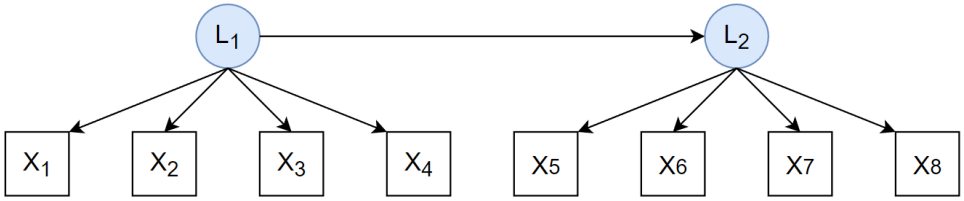


Figure 7: Example of 1-pure variable model with 4 measured variables per latent variable

Using lemma 2.1, we conclude that the tetrad constraints (11), (12) and (13) hold for such a model.

3 Estimating Lambda Factors

As mentioned in Subsection 2.2, in order to use the Copula PC algorithm, it is required from the user to give the lambda factor loadings. Since Machine2Learn has already developed an implementation of the Copula PC algorithm with extensions, it is highly interested in ways to find the factor loadings in a data-driven way. Hence the rest of this thesis will focus on this specific problem.

Here, we describe the technique we used, called Tetrad Test, and some variations of it, as well as a bootstrapped version of it. Finally, we compare and elaborate on the results.

Tetrad Test

In order to estimate the lambda factors we will use the statistical hypothesis testing method (see Appendix A) where the test statistic is based on tetrads. Tetrads are differences in the product of pairs of covariances among four random variables. Since the lambda factors are defined by the covariances among the variables, a tetrad test could give an estimation of suitable lambda factors, necessary for the Copula PC algorithm implementation.

However, the implementation of the test statistic can be complicated and computationally expensive. In the following sections, we describe the original tetrad test statistic, introduced by Bollen and Ting (1998) [13]. Moreover, we describe an extension based on a statistical technique, following closely the publication by Johnson and Bodner (2007) [14]. We compare their performance and visualize some of the results.

3.1 Background

Let $\mathbf{X}$ be an $n \times m$ matrix, consisting of n m-dimensional observations. These m-dimensional observations are assumed to be realizations from a distribution with covariance matrix $\mathbf{\Sigma}$ with elements σ_{ij}. The *terads* are:

$$\tau_{ijkl} = \sigma_{ij}\sigma_{kl} - \sigma_{ik}\sigma_{jl} \tag{14}$$

and can be estimated by

$$\hat{\tau}_{ijkl} = \hat{\sigma}_{ij}\hat{\sigma}_{kl} - \hat{\sigma}_{ik}\hat{\sigma}_{jl} \tag{15}$$

where $\hat{\sigma}_{ij}$ is the estimated covariance between the i-th and j-th variables from the observed covariance matrix computed from $\mathbf{X}$.

We consider the problem of modeling $\mathbf{\Sigma}$ and notice that some restrictions on the covariance structure may imply that some of the tetrads equal zero. These will be referred to as *vanishing tetrads*. As mentioned in Subsection 2.3, for every four variables, if at least two of the tetrad constraints (11), (12) and (13) hold, then these variables could have a common cause (common latent variable). Therefore, if at least two vanishing tetrads are present, it is implied that i, j, k and l could have a common latent variable.

Then, we can define a statistical test with:

- null hypothesis: H_0 = having vanishing tetrads
- alternative hypothesis: H_a = not having vanishing tetrads.

In order to reject the null hypothesis, we use a test statistic that is a function of those estimated tetrads, so that it captures whether they vanish, as well as a distribution which the test statistic follows under the null hypothesis [14].

3.2 Original Tetrad Test

We start by describing the steps of the original tetrad test and the computation of the tetrad test statistic.

Step 1. The null hypothesis assumes that the 4 variables are all caused by the same latent variable. Thus, the observations matrix will be $(n \times 4)$. This implies that there exist a set of vanishing tetrads,

that is, $\tau_{abcd} = 0$. The alternative hypothesis assumes that this is not the case, thus, $\tau_{abcd} \neq 0$. The test tries to verify which hypothesis is true, therefore, whether τ_{abcd} is vanishing or not.

Step 2. Compute the estimate covariance matrix of $\hat{\tau}$, the estimator of the vanishing tetrads identified in the previous step. This is

$$\mathbf{D}(\hat{\sigma})'\widehat{Cov}(\hat{\sigma})\mathbf{D}(\hat{\sigma}) \tag{16}$$

where $\mathbf{D}(\hat{\sigma})$ is a gradient matrix and $\widehat{Cov}(\hat{\sigma})$ is the estimated covariance matrix of sample covariances $\hat{\sigma}$. Note that $\widehat{Cov}(\hat{\sigma})$ must not be confused with the sample covariance matrix Σ.

The gradient matrix $\mathbf{D}(\hat{\sigma})$ is defined as $\mathbf{D}(\hat{\sigma}) = \frac{\partial \tau}{\partial \sigma}\Big|_{\sigma=\hat{\sigma}}$ but it can be easily computed by noting that the partial derivative of τ_{abcd} with respect to σ_{ij} is

$$\frac{\partial \tau_{abcd}}{\partial \sigma_{ij}} = \begin{cases} \sigma_{cd} & \text{if } i = a \text{ and } j = b \text{ or if } i = b \text{ and } j = a \\ \sigma_{ab} & \text{if } i = c \text{ and } j = d \text{ or if } i = d \text{ and } j = c \\ -\sigma_{bd} & \text{if } i = a \text{ and } j = c \text{ or if } i = c \text{ and } j = a \\ -\sigma_{ac} & \text{if } i = b \text{ and } j = d \text{ or if } i = d \text{ and } j = b \\ 0 & \text{otherwise} \end{cases}$$

The covariance matrix $Cov(\hat{\sigma})$ has elements $Cov(\hat{\sigma}_{ij}, \hat{\sigma}_{kl}) = E(Z_i Z_j Z_k Z_l) - \sigma_{ij}\sigma_{kl}$. When variables in $\mathbf{X}$ follow a multivariate normal distribution, this difference between the expected value of the product of 4 variables and the product of (co)variances simplifies to [15]:

$$Cov(\hat{\sigma}_{ij}, \hat{\sigma}_{kl}) = \sigma_{ik}\sigma_{jl} + \sigma_{il}\sigma_{jk}. \tag{17}$$

These (co)variances are estimated by replacing the true tetrads (see equation (14)) with the estimates based on the sample moments (see equation (15)) leading to the estimated covariance matrix $\widehat{Cov}(\hat{\sigma})$.

Step 3. Identify a set of non-redundant vanishing tetrads. As mentioned in subsection 2.3, for every three tetrad constraints, any two imply the third, therefore only two vanishing tetrads are non-redundant. To identify them, we apply a sweep operator [16], namely, a matrix A consisting of zeros and ones, to the estimated covariance matrix of $\hat{\tau}$ based on the fitted covariance matrix computed in the previous step (see formula (16)).
Randomization of the selection of the set of non-redundant vanishing tetrads can be done by randomly permuting the variables. Under the null hypothesis, if t and t^* are the number of all vanishing tetrads (this is three in our case) and the number of non-redundant vanishing tetrads (this is two in our case) respectively, with $t^* \leq t$, then a $t^* \times t$ matrix A of zeros and ones can effectively select a set of non-redundant vanishing tetrads $\tau^* = A\tau$.

Step 4. Compute the tetrad test statistic

$$\begin{aligned} T &= \hat{\tau}' A'[AD(\hat{\sigma})'\widehat{Cov}(\hat{\sigma})D(\hat{\sigma})A']^{-1}A\hat{\tau} \\ &= \hat{\tau}^{*\prime}[AD(\hat{\sigma})'\widehat{Cov}(\hat{\sigma})D(\hat{\sigma})A']^{-1}\hat{\tau}^* \\ &= \hat{\tau}^{*\prime}[Cov(\hat{\tau}^*)]^{-1}\hat{\tau}^* \end{aligned} \tag{18}$$

where the estimated covariance matrix of $\hat{\tau}$ is based on the gradient matrix $\mathbf{D}(\hat{\sigma})$ and the estimated covariance matrix of sample covariances $\hat{\sigma}$, $\widehat{Cov}(\hat{\sigma})$, following (16). This formula is the original tetrad test statistic and will be referred to as T_0. The statistical test that uses T_0 will be referred to as *OrigTetradTest*.

Convergence of T_0

According to Bollen & Ting (1993) [17], under the null hypothesis, T_0 converges in distribution to χ^2_k for large sample sizes and thus, is a suitable test statistic for our null hypothesis using a χ^2_k reference

distribution, where k is the number of degrees of freedom. In our case, k is equal to the number of non-redundant vanishing tetrads, that is, $k = 2$, as explained in subsection 2.3. We visualize this convergence in subsection 4.2 by implementing the test and comparing the distribution of T_0 to the χ^2_2-distribution.

Now, in order to decide whether to reject our null hypothesis, we need to calculate the p-value for the T_0.

Definition 7 (p-value)**.** *Consider an observed test statistic T_{obs} from unknown distribution. Then the p-value p is what the probability would be of observing a test statistic value T at least as 'extreme' as T_{obs}, if T follows the distribution given by null hypothesis H_0. That is:*

- $p = Pr(T \leq T_{obs}|H_0)$ *for a one-sided left-tail test,*
- $p = Pr(T \geq T_{obs}|H_0)$ *for a one-sided right-tail test,*
- $p = 2 \times \min\{Pr(T \leq T_{obs}|H_0), Pr(T \geq T_{obs}|H_0)\}$ *for a two-sided test.*
  If the distribution of T_{obs} is symmetric about zero, then $p = 2 \times Pr(T \geq |T_{obs}| \ |H_0)$[18].

In our case, we use the original tetrad test statistic $T = T_0$. Since our null hypothesis is that the tetrad difference is equal to 0, then the p-value is defined as the probability that the observed test statistic T_{obs} is larger than T_0. That is,

$$p = Pr(T_{obs} \geq T_0|H_0) = \int_0^{T_{obs}} p(T_0) \ \mathrm{d}T_0. \tag{19}$$

Moreover, there are two degrees of freedom ($k = 2$), since any two of the three vanishing tetrad differences (see (8), (9) and (10)) entail the third. Therefore, the p-value as defined in (19), can be calculated from the quantile function of the χ^2_2-distribution.

Alternative tetrad test statistics

Even though using test statistic T_0 has proven to be efficient and has the advantage of an asymptotic distribution (χ^2), there are some disadvantages that motivate us to experiment with alternative test statistics.

Starting from the formula of T_0, we notice that we need to invert the covariance matrix of $\hat{\tau}$ (see formula (16)). In some cases this matrix can be singular. To prevent this, we need to use matrix A, which makes computations more complicated and increases the chances of getting an error. Moreover, according to Bollen & Ting (1993) [17], decreasing the sample size, T_0 deviates from the asymptotic distribution.

To overcome the above complications, we will use some alternative tetrad test statistics, originally introduced by Johnson and Bodner (2007) [14]. These are the following:

– $T_1 = \hat{\tau}' \left[diag[D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})] \right]^{-1} \hat{\tau}$ (20)

by replacing $D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})$ with $diag[D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})]$[2]
and setting $A = I$, with I being the identity matrix of size t (number of all vanishing tetrads).

– $T_2 = \hat{\tau}' I \hat{\tau}$ (21)

by replacing $D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})$ with I and setting $A = I$.

– $T_3 = \hat{\tau}' A' \left[A \cdot diag[D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})] A' \right]^{-1} A \hat{\tau}$ (22)

by replacing $D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})$ with $diag[D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})]$.

However, altering the original test statistic T_0 to formulate T_1, T_2 and T_3 implies that we cannot use the χ^2-distribution as a reference, so we need to estimate the distribution of the test statistics under the null hypothesis H_0. To do so, we will use a bootstrapping technique that will be described in detail in the following subsection 3.3.

[2] Here, $diag[D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})]$ is a matrix with the same diagonal elements as of $[D(\hat{\sigma})' \widehat{Cov}(\hat{\sigma}) D(\hat{\sigma})]$ and zero as the non-diagonal elements.

3.3 Bootstrap Tetrad Test Statistic

As mentioned before, when using one of the alternative tetrad test statistics, we do not know the sampling distribution under the null hypothesis.Therefore, we need to estimate the distribution of the test statistics under the null hypothesis H_0 and calculate the p-value based on that. To do so, we resample our data in such a way that the null hypothesis holds, while keeping all other statistical properties as similar as possible.

Bootstrapping is a sampling method that independently samples with replacement from an existing sample data with the same sample size n and performing inference among these resampled data. More details of this technique can be found in Appendix B. Here, we apply this technique to the tetrad test statistic.
The bootstrap tetrad test [13] is based on comparing the observed test statistic, $T = T(X)$, with B simulation realizations of the test statistic, $T(Z^{(1)}), T(Z^{(2)}), \ldots, T(Z^{(B)})$.

That is,

$$Z = X(Q^{-1})'R' \tag{23}$$

where

- Q is obtained from the Choleski decomposition $S = QQ'$, with S the sample covariance matrix
- R is obtained from the Choleski decomposition $\hat{\Sigma}_0 = RR'$, with $\hat{\Sigma}_0$ the estimated covariance matrix under the null hypothesis.

In order to obtain the Choleski decomposition for R, we need to estimate the covariance matrix under the null hypothesis, $\hat{\Sigma}_0$. This can be done by minimizing the Kullback–Leibler (KL) divergence between two multivariate normal distributions. The first normal distribution follows the empirical statistics, with mean $\mu = 0$ and covariance matrix S, and the other with covariance matrix Σ_0. Note that Σ_0 is constrained to follow the null hypothesis, by enforcing its elements to follow equality (7). A more detailed description of this approach is given in Appendix C. Resampling is done with respect to Z rather than X. For every simulation realization we generate an array of N elements chosen randomly with replacement from the range 1 to n. These elements are used as indexes on Z, creating a new array, that is, a new Z. Next, the new Z is used in the tetrad test. This way, the bootstrap sampling distribution is consistent with the null hypothesis in the sense that the sample covariance matrix computed from Z equals the covariance matrix under the null hypothesis. That is, if H_0 holds, then $\hat{\Sigma}_0 = S$ which implies that $Z = X$.

Now, the bootstrap p-value is defined as the number of all the simulation realizations of the test statistic that are larger or equal to the observed test statistic, divided by the number of simulations. That is,

$$p_{bs} = B^{-1} \sum_{b=1}^{B} I[T(Z^{(b)}) \geq T(X)]. \tag{24}$$

When applying the bootstrap in the tetrad test, we notice that we go through **Step 2**, **Step 3**, and **Step 4** $(B + 1)$ times. However, when choosing T_1 instead of the original test statistic T_0 the computations in **Step 2** and **Step 4** are significantly reduced, and **Step 3** is omitted.

4 Implementation of Tetrad Test Statistic

In this section, we apply the tetrad test introduced in the previous chapter to simulated data, using different tetrad test statistics. The `Python` script for every implementation that is mentioned below, can be found here: https://github.com/KonstantinaXK/CDFCLC.git

4.1 Implementation Settings

We start by creating simulated data. More specifically, we generate the observation matrix $\mathbf{X}$ we need as input in the tetrad test, with size $(N \times D)$, where D is the number of variables in the test, thus $D = 4$, and N is the number of observations per variable.
We experimented with two types of observation matrices: one according to a linear latent variable model and one sampled from a multivariate normal distribution.

Observation matrix sampled from model: This observation matrix is sampled from a linear latent variable model, with 4 variables, whose covariance structure satisfies equality (7), giving that:

$$\mathbf{X} = \begin{pmatrix} X_{11} = \lambda_1 L_1 + \epsilon_1 & X_{12} = \lambda_2 L_1 + \epsilon_2 & X_{13} = \lambda_3 L_1 + \epsilon_3 & X_{14} = \lambda_4 L_1 + \epsilon_4 \\ X_{21} = \lambda_1 L_2 + \epsilon_1 & X_{22} = \lambda_2 L_2 + \epsilon_2 & X_{23} = \lambda_3 L_2 + \epsilon_3 & X_{24} = \lambda_4 L_2 + \epsilon_4 \\ \vdots & \vdots & \vdots & \vdots \\ X_{N1} = \lambda_1 L_N + \epsilon_1 & X_{N2} = \lambda_2 L_N + \epsilon_2 & X_{N3} = \lambda_3 L_N + \epsilon_3 & X_{N4} = \lambda_4 L_N + \epsilon_4 \end{pmatrix}$$

where
$\lambda_1, ..., \lambda_4$ the factor loadings
$L_1, ..., L_N$ the latent variable L sampled from a standard normal distribution
$\epsilon_1, ..., \epsilon_4$ the noise terms.

Observation matrix sampled at random: This observation matrix is sampled from a multivariate normal distribution, with mean $\mu = 0$ and a randomly generated covariance matrix. We do so, simply by using a `NumPy` function that gives the desired output.

Moreover, in our implementation, we use the following notation:
– N: the number of observations per variable, as mentioned above
– M: the number of observation matrices
– B: the number of simulation realizations in bootstrap

4.2 Implementing tetrad test using T_0

As mentioned in subsection 3.2, T_0 converges in distribution to χ^2_k, with $k = 2$ (see Figure 8). This is visualized in Figure 9, where the test statistic has been calculated for $M = 200$ observation matrices sampled according to the model, with $N = 2000$ observations per matrix.

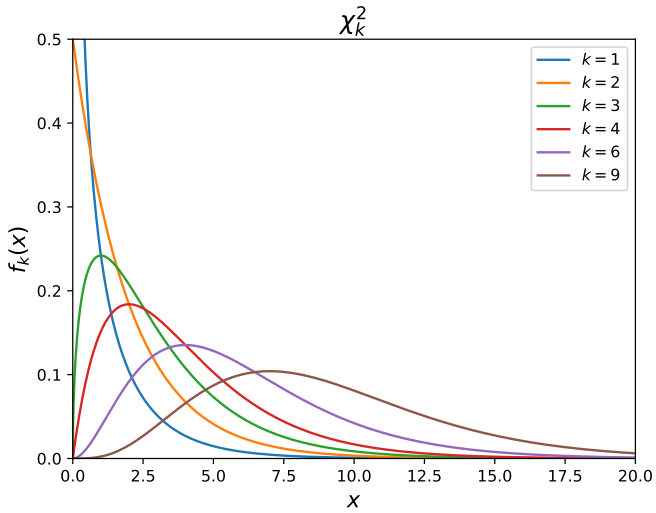


Figure 8: PDF of χ^2_k for different degrees of freedom k

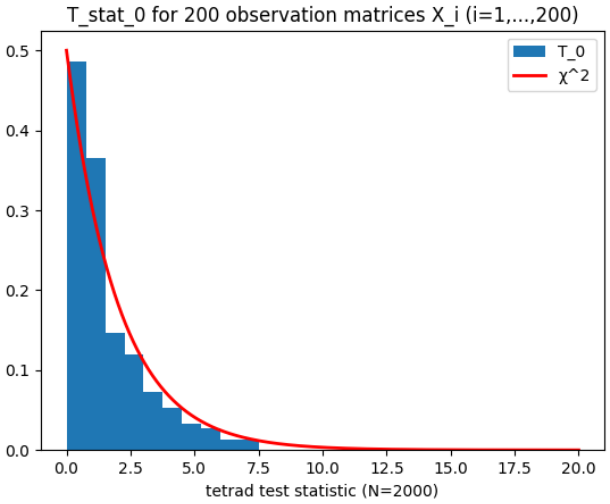


Figure 9: Test statistic T_0 and χ^2 distribution with 2 degrees of freedom

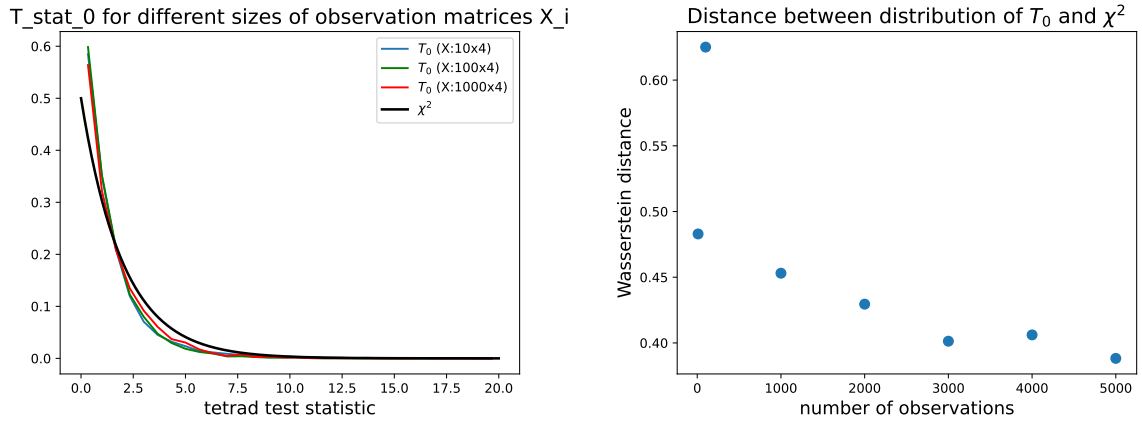
Moreover, even though Bollen & Ting (1993) [17] mention that decreasing the sample size, makes T_0 deviate from the reference distribution χ^2, we notice that in our case, where we consider only $m = 4$ variables, this deviation is quit small. In Figure 10a, we can see an example of three distributions with different sized observation matrices, compared to the χ^2-distribution. Here, the distribution of the test with the largest observation matrix is, indeed, closer to the χ^2-distribution, but the distance from the other two distributions is insignificantly small. In Figure 10b, we implement the test, starting from a very small observation matrix ($N = 10$) and gradually going to a large observation matrix ($N = 5000$), comparing the Wasserstein distance from the reference distribution. That is:

Definition 8 (Wasserstein distance [19])**.** *For $p \in [1, \infty)$ and probability measures P, Q on $\mathbb{R}^d$ with finite p-moments, their p-Wasserstein distance is*

$$W_p(P, Q) = \left(\inf_{\pi \in \Gamma(P,Q)} \int_{\mathbb{R}^d \times \mathbb{R}^d} ||X - Y||^p d\pi \right)^{1/p}$$

where $\Gamma(P,Q)$ is the set of all joint probability measures on $\mathbb{R}^d \times \mathbb{R}^d$ whose marginals are P, Q, i.e. such that for all subsets $A \subset \mathbb{R}^d$ we have $\pi(A \times \mathbb{R}^d) = P(A)$ and $\pi(\mathbb{R}^d \times A) = Q(A)$.

Again, the conclusion that decreasing the sample makes T_0 deviate from the χ^2-distribution is confirmed, but this deviation is insignificant.



(a) (b)

Figure 10: Distribution of test statistic compared to distribution of χ^2 (for different sizes of X_i)

Additionally, implementing the tetrad test using T_0 for $M = 2000$ observation matrices X_i, $(i = 1, ..., M)$, sampled from the model, with $N = 1000$ observations per matrix, we get the p-values that can be seen in Figure 11. As we can observe, the majority of the derived p-values are larger than the threshold $\alpha = 0.05$, thus, the tetrads are vanishing, as expected for an observation matrix sampled from the model.

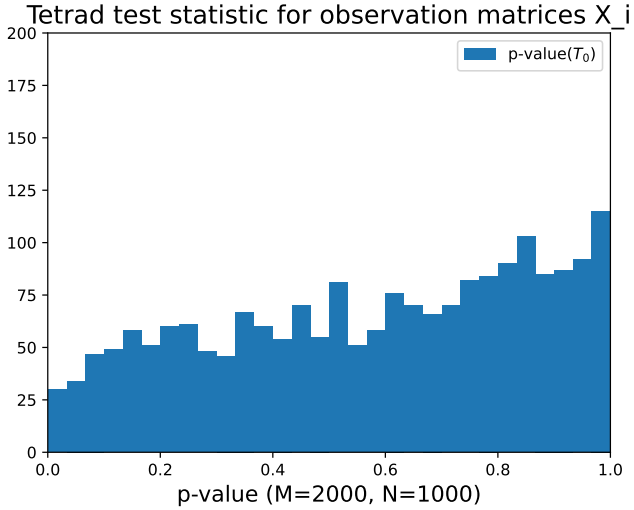


Figure 11: Observed p-values of the tetrad test using T_0 with observation matrices sampled from the model

Finally, we implement the test for $M = 2000$ observation matrices X_i, $(i = 1, ..., M)$, sampled randomly, with $N = 1000$ observations per matrix. As expected, the vast majority of the p-values equal 0, since the probability of achieving a vanishing tetrad when sampling the observation matrix at random, is quite small. This can be seen in Figure 12.

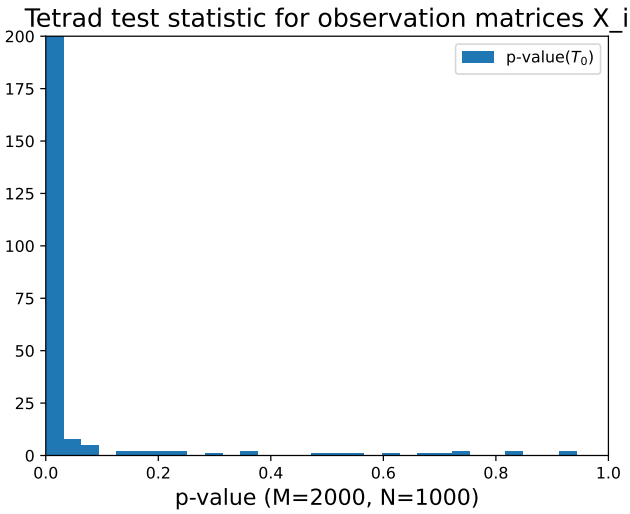


Figure 12: Observed p-values of the tetrad test using T_0 with randomly sampled observation matrices

4.3 Implementing bootstrapped tetrad test

Implementing the bootstrapped tetrad test using T_1, we first calculate the simulation realizations $T(Z^{(b)})$ and then we compare them with the observed test statistic $T(X)$. Figure 13 is the result of an example where the observation matrix X is sampled from the model with $N = 1000$ observations and we compare the observed test statistic T_1 with $B = 300$ simulation realizations. As we can observe, the derived p-value is larger than the threshold $\alpha = 0.05$, thus, the tetrads are vanishing, as expected for an observation matrix sampled from the model.

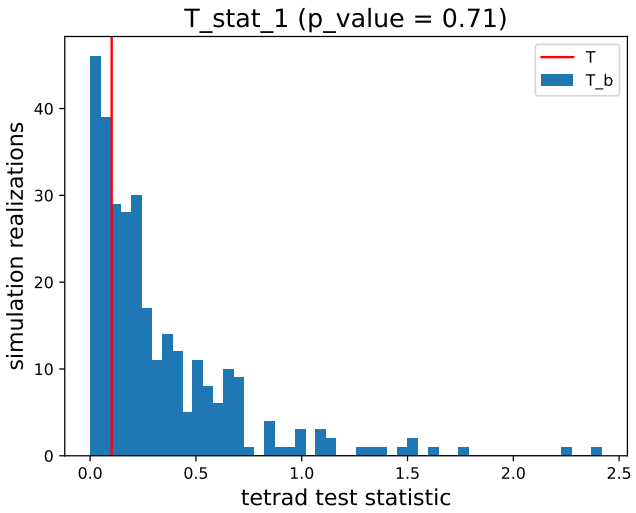


Figure 13: Observed test statistic T_1 (here T) and 300 simulation realizations $T(Z^{(b)})$ (here T_b)

Now, when sampling the observation matrix from a multivariate distribution with a given covariance matrix (generated at random), we observe that the p-value becomes zero. This is expected from a randomly sampled observation matrix, since there is a really low chance of entailing a vanishing tetrad. The histogram of the test statistic and the simulation realizations can be seen in Figure 14. We observe that the simulation realizations T_b are smaller than the observed test statistic T, and thus, according to formula (24), the p-value is zero.

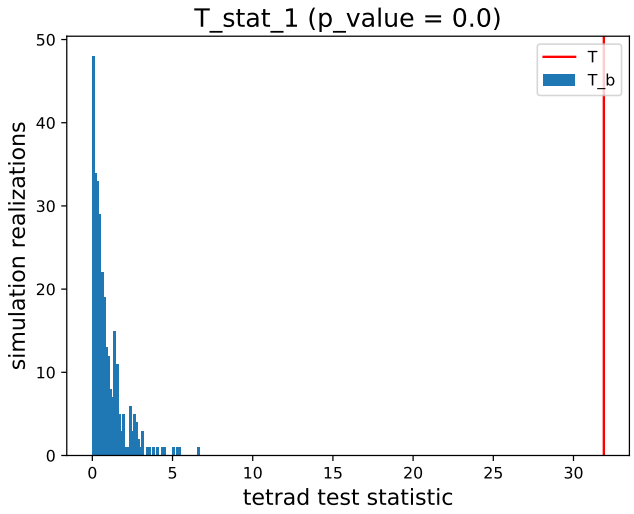


Figure 14: Test statistic T_1 for random observation matrix

Comparing alternative tetrad test statistics

In order to conclude which of the test statistics is the most appropriate to use in our case, we need to investigate the alternatives. Thus, we compare the performance of T_1, T_2 and T_3.
We start by sampling an observation matrix X from the model, with $N = 1000$ observations. Then we repeatedly calculate the test statistics and the corresponding simulation realizations with $B = 100$, leading to the p-value of every test statistic. This step is repeated 1000 times. The result can be seen in Figure 15. Here, we observe that the histogram of T_1 is centered closer to 1 than that of T_2 and T_3, indicating that the majority of the tetrads are vanishing, as expected when sampling from the model. We notice that even though we use the same observation matrix as input, there is a small variance on the p-values, within each test statistic repetition. This is expected, since, in the bootstrap method generates additional randomness. However, the difference is really small and thus, has no influence on the outcome of the statistical test. Moreover, we observe that the test statistics give different sets of p-values. Therefore, we will investigate further the differences between them.

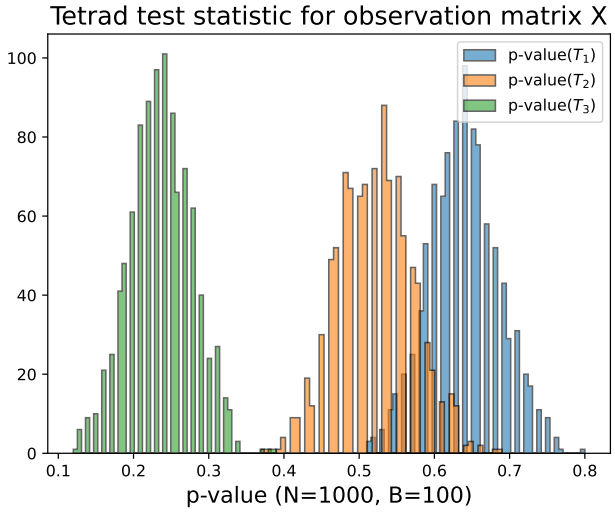


Figure 15: p-values of different tetrad test statistics for the same simulated observation matrix

Now, we sample M observation matrices $X_1, X_2, ..., X_M$ the same way as in Figure 15, that is, according

to our model, with $N = 1000$ observations per matrix. For every matrix X_i $(i = 1, ..., M)$, we calculate the test statistics and the corresponding simulation realizations. Finally, using formula (24), we derive the p-values for the different test statistics, for all the observation matrices. The p-value histogram for $M = 2000$ can be seen in Figure 16. We observe that all the histograms approximate a uniform distribution, as expected, without, however, giving any insight on which statistic is most suitable for our tetrad test.

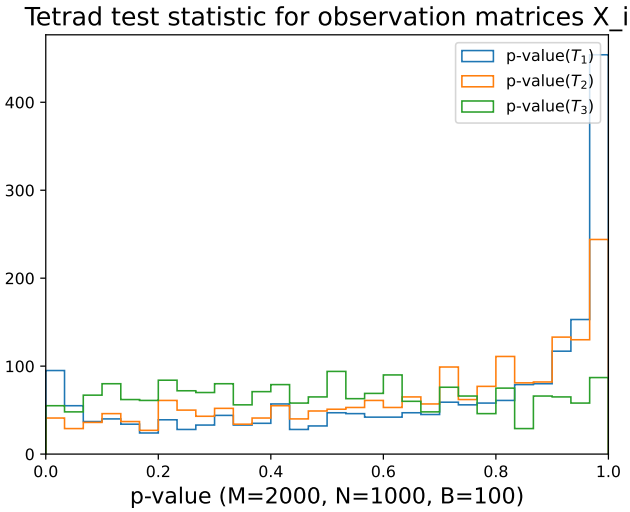


Figure 16: p-values of different tetrad test statistics for multiple observation matrices

Moreover, we want to explore the p-values of every test statistic in relation to all the other ones. Therefore we consider the following pairs, (T_1, T_2), (T_2, T_3) and (T_1, T_3), and we compare the p-values within each pair. To do so, we draw the scatter plots with the identity line, for an example of $M = 500$ observation matrices sampled from the model, as can be seen in Figure 17. We observe that most dots of the pair (T_1, T_2) fall above the identity line, suggesting that the test T_2 has more power than T_1 (see 17a). Similar is the case of (T_1, T_3), where the deviation from the identity line is larger, thus, T_3 leads to much larger p-values than T_1 (see 17c). Finally, for pair (T_2, T_3), we cannot safely conclude whether the majority of the dots fall under or above the line (see 17b).

In conclusion, after comparing the three alternative test statistics T_1, T_2, T_3 and taking into account that we deal with a small number of observed variables ($D = 4$), we cannot claim any statistic to be more efficient for our case, since their performance is quite similar. Thus, we will follow the suggestion by Bollen & Ting (1993) [17] and choose T_1 for our implementation.

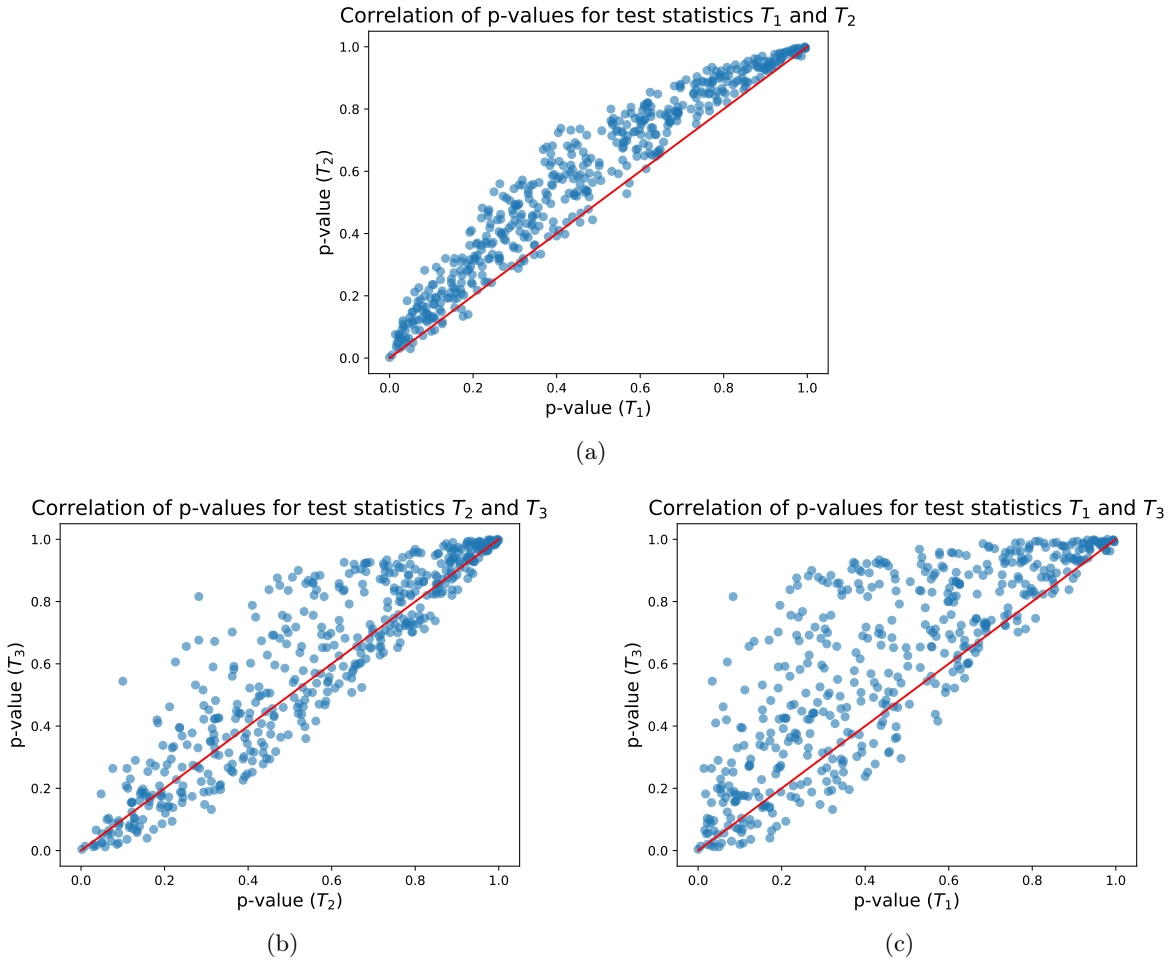
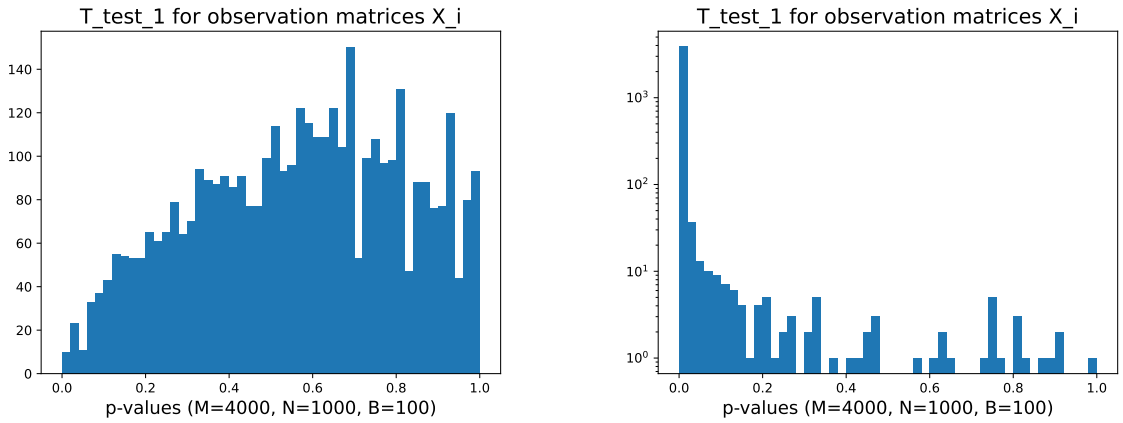


Figure 17: Correlation of p-values for every pair of test statistics

Here, using statistic T_1, we implement the test for multiple observation matrices and we obtain the histogram in Figure 18a. This is an example of the p-values obtained from $M = 4000$ observation matrices, sampled from the model, with $N = 1000$ observations per matrix. It can be observed that the majority of p-values are larger than threshold $\alpha = 0.05$, which is expected, since the observation matrices are sampled from the model.

However, when we sample the observation matrices from a multivariate distribution with a given covariance matrix (generated at random), we observe that the p-value is zero for almost all the observation matrices, with very few exceptions. This makes sense, since a randomly sampled observation matrix is expected to have a really low chance of entailing a vanishing tetrad. The histogram of the p-values in such an example, for $M = 1000$ observation matrices, can be seen in Figure 18b (notice that y-axis is in logarithmic scale).



(a) (b)

Figure 18: p-values of bootstrapped test T_1 with observation matrices sampled (a) from model and (b) randomly (in (b): y-axis in logarithmic scale)

4.4 Comparing original and bootstrapped tetrad test

Finally, we compare the original tetrad test using test statistic T_0 with the bootstrapped tetrad test using test statistic T_1.

There are some definite advantages of T_1 when it comes to singularity, matrix A (mask), computational savings and bootstrapping. Starting off with singularity, in the T_1 formula (see (20)), we notice that instead of calculating the inverse of a matrix, we consider a diagonal matrix[2], thus a non singular matrix (unless there are zero variances, which is highly unlikely), thus invertible. Moreover, it becomes clear that there is no need for matrix A as stated in Step 3 of the Original Tetrad Test, since we use the inverse of the diagonal covariance matrix and thus, the properties and outcome of the test are independent of the selected set of non-redundant vanishing tetrads. Also, T_1 can result in significant computational savings, especially in the bootstrap test, where the test statistic is repeatedly computed. Only the diagonal elements of the covariance matrix of $\hat{\tau}$ need to be computed, and the inverse matrix is obtained by simply computing the reciprocals of the variances. Finally, since there are fewer elements in the covariance matrix of $\hat{\tau}$ to be estimated, this may result in a reduction in the variance in T_1 and, therefore, more power in smaller samples [14].

In Figure 19, we depict 3 examples of $M = 4000$ observation matrices sampled from the model, with $N = 1000$, 2000 and 4000 observations per matrix and $B = 500$ simulations in bootstrapping test T_1. T_0 appears to perform better, since its p-value histogram follows a uniform distribution, as expected, under the null hypothesis, while T_1 seems skewed towards 1, decreasing the power of the tetrad test. Thus, T_1 appears to be a more conservative test towards the null hypothesis.
Considering that the tetrad test performs slightly better when using T_0 and in this case, the bootstrap step is not required, making the test easier and quicker to implement, we conclude that T_0 is the most appropriate test statistic for our problem. To conclude, we will use the *OrigTetradTest*.

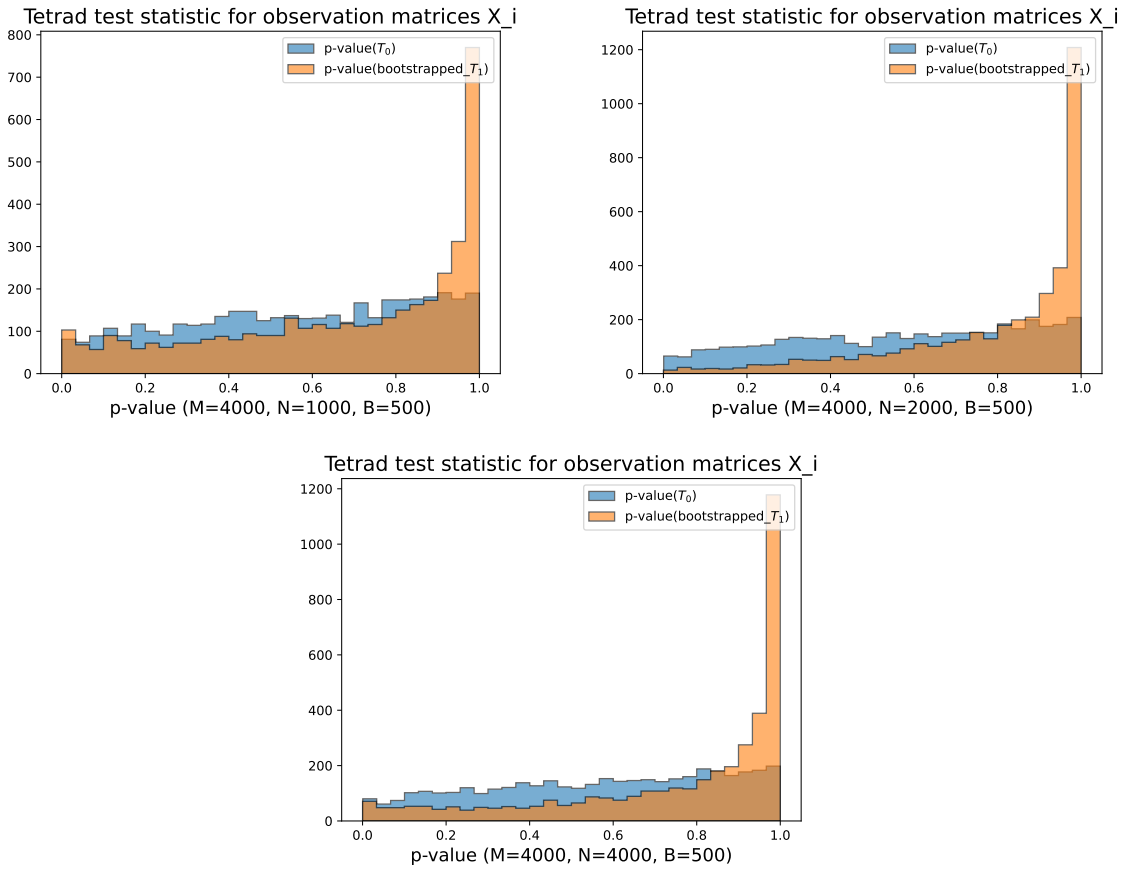


Figure 19: Examples of test statistic T_0 and bootstrapped T_1 for different sized observation matrices

5 FOFC Algorithm

In the previous section, we explained how the confirmatory tetrad test could be used to confirm whether a quartet of observed variables coming from a certain SEM, have a common latent cause. However, we want to do some data-driven analysis instead. To this end, we would like to have a stable method to search through the large space of all possible models with the measured variables and latent causes. To do so, we describe and implement the FindOneFactorClusters (FOFC) algorithm, originally introduced by Kummerfeld and Ramsey (2016) [12]. Using FOFC, one can reliably infer measurement models from measured indicators (variables), without prior knowledge of the causal relations or the number of latent variables.
The FOFC receives as input an observation matrix and gives a list of non-intersecting clusters of variables, where it is implied that the variables within the same cluster have a common latent variable. In order to obtain these clusters, we need to test whether a given quartet of variables is vanishing, examining its covariance structure. For that, we use the tetrad test described in the previous section.

While describing the FOFC, we use the following notation:

- K: the number of observed variables
- $\mathbf{X} = \{X_1, X_2, ..., X_K\}$: the set of observed variables
- α: the threshold of hypothesis testing (critical value)

The FOFC algorithm works as follows. It calls three main functions: *FindPureClusters*, *GrowClusters* and *SelectClusters*. It first calls *FindPureClusters*, which tests each triple of variables to see if it has the property that adding any other member of $\mathbf{X}$ creates a vanishing quartet; if it does have the property it is added to the list *PureList* of pure triples. *FindPureClusters* tests whether a given quartet of variables is a vanishing quartet by calling the *OrigTetradTest*, described in the previous section.
Then, *GrowClusters* initializes *CList* to *PureList*. If any two pure sets of variables overlap, their union is also pure. FOFC calls *GrowClusters* to see if any of the pure triples in *PureClusters* can be combined into a larger pure set. In order to determine whether a given variable $\mathbf{u}$ can be added to a cluster $\mathbf{C}$ in *CList*, *GrowClusters* checks whether a given fraction (determined by the parameter **Cpar**) of the sub-clusters of size 3 containing 2 members of $\mathbf{C}$ and $\mathbf{u}$ are on *PureList*. If they are not, then *GrowClusters* tries another possible expansion of clusters on *CList*; if they are, then *GrowCluster* adds $\mathbf{u}$ to $\mathbf{C}$ in *CList* and deletes all subsets of the expanded cluster of size 3 from *PureList*. *GrowClusters* continues until it exhausts all possible expansions.
Finally, when *GrowClusters* is done, *SelectClusters* goes through *CList*, iteratively outputting the largest remaining cluster $\mathbf{C}$ still in *CList*, and deleting any other clusters in *CList* that intersect $\mathbf{C}$ (including $\mathbf{C}$ itself).

Even though the algorithm follows closely the steps described by Kummerfeld and Ramsey (2016) [12], we made some adjustments in *GrowClusters* function.

In Algorithm 1, the steps are described in more detail.

Algorithm 1 FindOneFactorClusters

Input: Set of variables V, observation matrix X, alpha
Output: Set of non-intersecting clusters SelClusters

PureList = FindPureClusters(V, X, alpha=0.05)
CList = GrowClusters(PureList, Cpar=0.7)
SelClusters = SelectedClusters(CList)

FindPureClusters(V**,** X**,** $alpha$**)**

PureList = $\emptyset$
for $S \subseteq V$, $|S| = 3$ **do**
    pure=True
    for $v \in V \notin S$ **do**
        if $\big($OrigTetradTest($(S \cup v)$, X, $alpha$)$\big)$=True **then**
            pure=False
            BREAK
        end if
        if pure **then**
            if $S \notin$ PureList **then**
                PureList = PureList $+S$
            end if
        end if
    end for
end for
RETURN PureList

GrowClusters($PureList, Cpar$**)**

union=$\bigcup_{i \in PureList} i$
CList = PureList
for cluster $\in$CList **do**
    for $u \in$union $\notin$cluster **do**
        for sub$\subset$cluster, |sub|=2 **do**
            testcluster=sub$\cup\{u\}$
            if testcluster$\in$PureList **then**
                acc+ = 1
            else
                rej+ = 1
            end if
        end for
        if acc/(rej + acc) >= Cpar **then**
            CList = CList $+cluster \cup \{u\}$
            for $s \in$cluster$\cup\{u\}$, $s \in$CList **do**
                PureList = PureList $-s$
            end for
        end if
    end for
end for
RETURN CList

SelectedClusters($CList$**)**

CListNew = sort(CList) ▷ Sort CList from largest clusters to smallest
for cluster1, cluster2 $\in$CListNew, cluster1$\neq$cluster2 **do**
    if cluster1$\cap$cluster2$\neq \emptyset$ **then**
        if cluster1$\subseteq$cluster2 **then**
            CListNew = CListNew - cluster1
        else
            CListNew = CListNew - cluster2
        end if
    end if
end for
RETURN CListNew

Implementation of FOFC

In order to test the efficiency of the FOFC algorithm, using the *OrigTetradTest*, we implement it for a simple example of generated data. We consider 10 variables and their observation matrix, generated according to the following linear latent variable model:

$$\begin{aligned} L_1 &= \lambda_L L_0 + \epsilon \\ X_i &= \lambda_i L_0 + \epsilon \quad \text{for } i = 0, 1, 2, 3, 4 \\ X_i &= \lambda_i L_1 + \epsilon \quad \text{for } i = 5, 6, 7, 8, 9 \end{aligned} \tag{25}$$

with
L_0, L_1 the latent variables,
$X_0, ..., X_9$ the observed variables,
$\lambda_L, \lambda_0, ..., \lambda_9$ the factor loadings,
ϵ the noise.

This example is visualized in Figure 20.

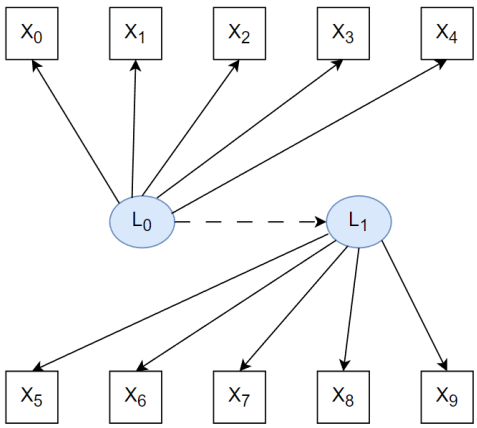


Figure 20: True DAG of Model (25)

Now, using this generated data as input in the FOFC algorithm, we get a list of two non-intersecting clusters, indicating that the variables coming from the same cluster, also caused by the same latent variable, and the variables coming from different clusters, are caused by different latent variables. Taking, for example, an observation matrix with $N = 10.000$ observations, **Cpar** $= 0.7$ and **alpha** $= 0.05$, we get the following output:

$$\texttt{Non-intersecting Clusters} = [(0, 1, 2, 3, 4), (5, 6, 7, 8, 9)]$$

This output translates to Figure 21, which is identical to the true DAG in Figure 20, except from the causal relation between the latent variables, that the FOFC is unable to identify. This is where the Copula PC algorithm comes in use, that investigates the relations among latent variables.

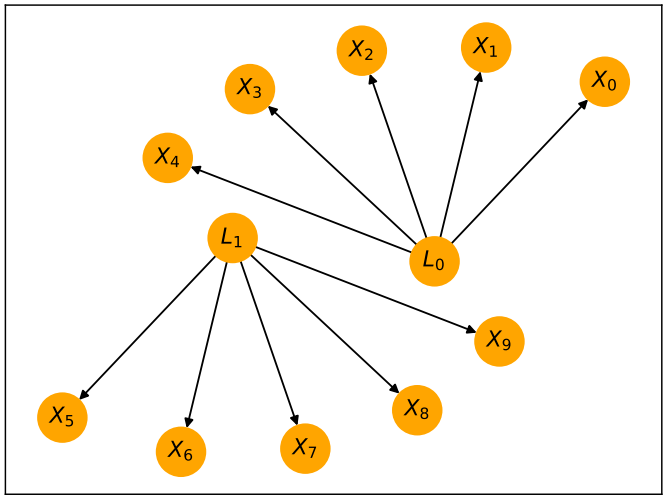


Figure 21: DAG resulted from FOFC

However, when running the FOFC 500 times, for different observation matrices ($N = 10.000$, **Cpar** $= 0.7$, **alpha** $= 0.05$) that are generated according to Model (25), we observe that the output can be inaccurate. More specifically, 120 out of 500 outputs look like the DAGs in Figure 22, where we notice that an observed variable can be absent in either one or both of the clusters. This inaccuracy could occur due to underperformance of the tetrad test.

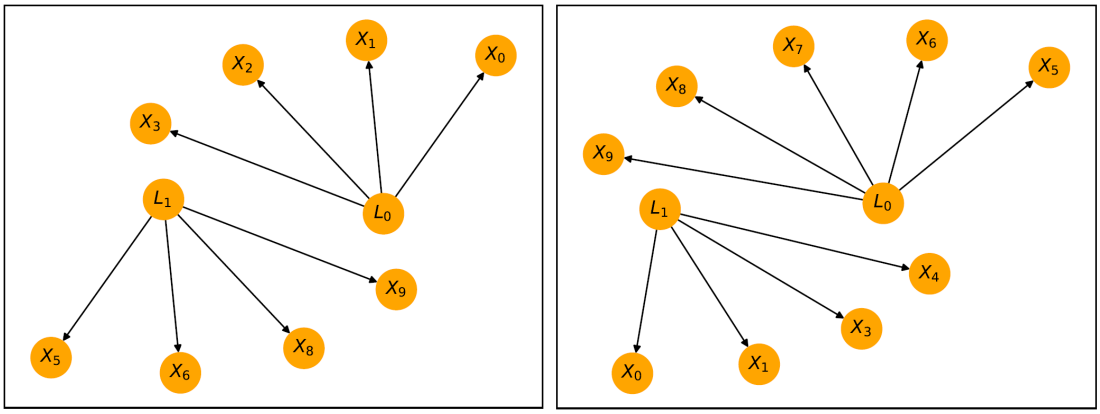


Figure 22: Examples of inaccurate outputs of FOFC

Therefore, we can conclude that the algorithm is able to identify the clusters that indicate the causal relationships among observed variables, but further analysis is needed, in order to improve its performance. For instance, experimenting with the parameters **Cpar** (*GrowClusters* step of Algorithm 1) and **alpha** (*OrigTetradTest* in *FindPureClusters* step of Algorithm 1) could potentially change the number of inaccurate outputs.

6 Conclusions

In this project, we implemented a confirmatory tetrad test, that is able to verify whether a quartet of observed variables come from the same latent cause. We experimented with different test statistics and compared their performance on simulated data. We proved that, using this test together with a search algorithm such as the FOFC, one is ablr to infer measurement models from measured variables and discover their common latent causes, without prior knowledge of the causal relations. However, further analysis is necessary, in order to improve its performance. After this, the result could be used in more complex causal discovery algorithms, such as the Copula PC.

7 Future Research

Future research could investigate how the FOFC algorithm could be used into the Copula PC algorithm. The Copula PC can deal with discrete variables and missing values. However, FOFC uses continuous variables and thus, it is required to investigate how to 'convert' the data in such a way that would work for both algorithms. One method could be to develop an algorithm similar to Copula PC, using Gibbs sampling, to estimate the covariance matrix of Z instead of Y (refer to Gaussian Copula Factor Model) and then use this matrix on a tetrad test. This way, a discrete matrix Y can be used in the software, with possibly missing values.
Furthermore, since we only experimented with linear variable models, it would be quite useful to extend the use of these methods to other types of causal relations, such as, non-linear or non-Gaussian.

References

[1] P. Spirtes, *International Encyclopedia of the Social & Behavioral Sciences*. Latent Structure and Causal Variables, Oxford: Elsevier, 2nd ed., 2015.

[2] J. Pearl, *Causality*. Models, Reasoning, and Inference, Cambridge: Cambridge University Press, 2nd ed., 2009.

[3] N. Bowen and S. Guo, "Structural equation modeling," *Structural Equation Modeling: A Multidisciplinary Journal*, pp. 1–240, 2012.

[4] T. D. Le, T. Hoang, J. Li, L. Liu, H. Liu, and S. Hu, "A fast pc algorithm for high dimensional causal discovery with multi-core pcs," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 16, no. 5, pp. 1483–1495, 2016.

[5] P. Spirtes and C. Glymour, "An algorithm for fast recovery of sparse causal graphs," *Social science computer review*, vol. 9, no. 1, pp. 62–72, 1991.

[6] P. Spirtes, C. N. Glymour, R. Scheines, and D. Heckerman, *Causation, prediction, and search*. Cambridge, Massachusetts: MIT press, 2nd ed., 2000.

[7] C. Glymour, K. Zhang, and P. Spirtes, "Review of causal discovery methods based on graphical models," *Frontiers in genetics*, vol. 10, p. 524, 2019.

[8] R. Cui, P. Groot, M. Schauer, and T. Heskes, "Learning the causal structure of copula models with latent variables," *Uncertainty in Artificial Intelligence: Proceedings of the Thirty-Fourth Conference*, pp. 188–197, 2018.

[9] R. Cui, P. Groot, and T. Heskes, "Copula pc algorithm for causal discovery from mixed data," *Joint European conference on machine learning and knowledge discovery in databases*, pp. 377–392, 2016.

[10] R. Silva, R. Scheines, C. Glymour, P. Spirtes, and D. M. Chickering, "Learning the structure of linear latent variable models.," *Journal of Machine Learning Research*, vol. 7, no. 2, 2006.

[11] H. A. Loeliger, "An introduction to factor graphs," *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 28–41, 2004.

[12] E. Kummerfeld and J. Ramsey, "Causal clustering for 1-factor measurement models," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1655–1664, 2016.

[13] K. A. Bollen and K.-F. Ting, "Bootstrapping a test statistic for vanishing tetrads," *Sociological Methods & Research*, vol. 27, no. 1, pp. 77–102, 1998.

[14] T. R. Johnson and T. E. Bodner, "A note on the use of bootstrap tetrad tests for covariance structures," *Structural Equation Modeling: A Multidisciplinary Journal*, vol. 14, no. 1, pp. 113–124, 2007.

[15] K. A. Bollen and K. F. Ting, "A tetrad test for causal indicators.," *Psychological methods*, vol. 5, no. 1, p. 3, 2000.

[16] J. H. Goodnight, "A tutorial on the sweep operator," *The American Statistician*, vol. 33, no. 3, pp. 149–158, 1979.

[17] K. A. Bollen and K. F. Ting, "Confirmatory tetrad analysis," *Sociological Methodology*, vol. 23, pp. 147–175, 1993.

[18] F. Emmert-Streib and M. Dehmer, "Understanding statistical hypothesis testing: The logic of statistical inference," *Machine Learning and Knowledge Extraction*, vol. 1, no. 3, pp. 945–962, 2019.

[19] A. Ramdas, N. Trillos, and M. Cuturi, "On wasserstein two-sample testing and related families of nonparametric tests," *Entropy*, vol. 19, no. 2, p. 47, 2017.

[20] B. Efron and R. Tibshirani, *An Introduction to the bootstrap*. Monographs on statistics and applied probability 57, New York: Chapman & Hall, 1994.

[21] D. Boos, "Introduction to the bootstrap world," *Statistical Science*, vol. 18, no. 2, pp. 168–174, 2003.

[22] I. Csiszar, "I-Divergence Geometry of Probability Distributions and Minimization Problems," *The Annals of Probability*, vol. 3, no. 1, pp. 146 – 158, 1975.

Appendix

A Statistical Hypothesis Testing

The principle idea of a statistical hypothesis test is to decide if a data sample is typical or atypical compared to a population assuming a hypothesis we formulated about the population is true. In general, regardless of the specific hypothesis test one is conducting, there are seven steps underlying all of hypothesis tests. These components are summarized in Figure 23.

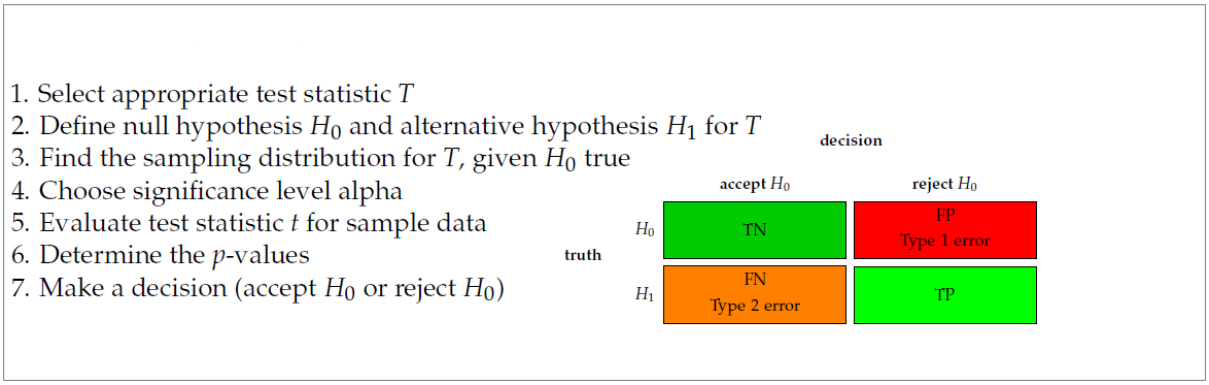


Figure 23: Overview for hypothesis tests

– **Step 1**: Select Test Statistic
Put simply, a test statistic quantifies a data sample. In statistics the term 'statistic' refers to any mapping (or function) between a data sample and a numerical value. Popular examples are the mean value or the variance. Formally, the test statistic can be written as

$$t_n = T(D(n)) \tag{26}$$

whereas $D(n) = \{x_1, ..., x_n\}$ is a data sample with sample size n. Here we denoted the mapping by T and the value we obtain by t_n. Typically the test statistic can attain real values but restrictions are possible.

The test statistic plays the central role in a hypothesis test because by deciding which test statistic to use one determines a hypothesis test to a large extend. For this reason one needs to carefully select a test statistic that is of interest and importance for the conducted study.

We would like to emphasize that in this step, we select the test statistic but we neither evaluate it nor we use it yet. This is done in step 5.

– **Step 2**: Null Hypothesis H_0 and Alternative Hypothesis H_1
At this step, we define two hypotheses which are called the null hypothesis H_0 and the alternative hypothesis H_1. Both hypotheses make statements about the population value of the test statistic and are mutually exclusive. For the test statistic $t = T(D)$ we selected in step 1, we call the population value of t as θ. Based on this we can formulate the following hypotheses:

$$\text{null hypothesis: } H_0 : \theta = \theta_0$$
$$\text{alternative hypothesis: } H_1 : \theta > \theta_0$$

As one can see, the way the two hypotheses are formulated, the value of the population parameter θ can only be true for one statement but not for both. For instance, either $\theta = \theta_0$ is true but then the alternative hypothesis H_1 is false or $\theta > \theta_0$ is true but then the null hypothesis H_0 is false.
Figure 23 shows the four possible outcomes of a hypothesis test. Each of these outcomes has a specific name that is commonly used. If the null hypothesis is false and we reject H_0 this is called a 'true positive' (TP) decision. The reason for calling it 'positive' is related to the asymmetric meaning of a hypothesis test, because rejecting H_0 when H_0 is false is more informative than accepting H_0 when H_0 is true. In this case one can consider the outcome of a hypothesis test a positive result.

The alternative hypothesis formulated above is an examples for a one-side hypothesis. Specifically, we formulated a right-sided hypothesis because the alternative assumes values larger than θ_0. In addition, we can formulate a left-sided alternative hypothesis stating

$$\text{alternative hypothesis: } H_1 : \theta < \theta_0$$

Furthermore, we can formulate a two-side alternative hypothesis that is indifferent regarding the side by

$$\text{alternative hypothesis: } H_1 : \theta \neq \theta_0.$$

Despite the fact that there are hundreds of different hypothesis tests, the above description principally holds for all of them.
In order to connect the test statistic t, which is a sample value, with its population value θ one needs to know the probability distribution of the test statistic. Because of this connection, this probability distribution received a special name and is called the sampling distribution of the test statistic. It is important to emphasize that the sampling distribution represents the values of the test statistic assuming the null hypothesis is true. This means that in this case the population value of θ is θ_0.

– **Step 3**: In our general discussion about the principle idea of a hypothesis test above, we mentioned that the connection between a test statistic and its sampling distribution is crucial for any hypothesis test.

Definition 1. Let $X(n) = \{X_1, ..., X_n\}$ be a random sample from a population with $X_i \sim P_{pop} \quad \forall i$ and $T(X(n))$ be a test statistic. Then the probability distribution $f_n(x|H_0 \text{ is true})$ of $T(X(n))$, assuming H_0 is true, is called the sampling distribution of the null hypothesis or the null distribution.

Similarly, one defines the sampling distribution of the alternative hypothesis by $f_n(x|H_1 \text{ is true})$. Since there are only two different hypotheses, H_0 and H_1, there are only two different sampling distributions in this context.

– **Step 4**: Significance Level α
The significance level α is a number between zero and one, that is, $\alpha \in [0, 1]$. It has the meaning

$$\alpha = P(\text{Type 1 error}) = P(\text{reject } H_0|H_0 \text{ is true}) \tag{27}$$

giving the probability to reject H_0 provided H_0 is true. This is the probability of making a Type 1 error resulting in a false positive decision.
When conducting a hypothesis test, we have the freedom to choose this value. The most frequent choice of α is 0.05.
Finally, we want to remark that formally we obtain the value of the right-hand side of equation (27) from the quantiles of the sampling distribution, as given by equation (31) (discussed below).

– **Step 5**: Evaluate Test Statistic from Data
This step is our connection to the real world, as represented by the data, because everything until here has been theoretical. For $D(n) = X(n) = \{x_1, ..., x_n\}$ we estimate the numerical value of the test statistic selected in Step 1 giving

$$t_n = T(D(n)). \tag{28}$$

Here t_n represents a particular numerical value obtained from the observed data $D(n)$. Due to the fact that our data set depends on the number of samples n, also this numerical value will be dependent on n. This is explicitly indicated by the subscript.

– **Step 6**: Determine the p-Values
For determining the p-values of a hypothesis test, we need to use the sampling distribution (Step 3) and the estimated test statistic tn (Step 5). That means the p-values results from a comparison of theoretical assumptions (sampling distribution) with real observations (data sample) assuming H_0 is true. This situation is visualized in Figure 24 for a right-sided alternative hypothesis. The p-values is the probability for observing more extreme values than the test statistic t_n assuming H_0 is true

$$p = P(\text{observe} \times \text{at least as extreme as } |t||H_0 \text{ is true}) = P(x \geq |t||H_0 \text{ is true}) \tag{29}$$

Formally it is obtained by an integral over the sampling distribution

$$p = \int_{t_n}^{\infty} f_n(x'|H_0 \text{ is true})dx' \tag{30}$$

The final decision if we reject or accept the null hypothesis will be based on the numerical value of p. Furthermore, we can use the following integral

$$\alpha = \int_{\theta_c}^{\infty} f_n(x'|H_0 \text{ is true})dx' \tag{31}$$

to solve for θ_c. That means, the significance level α implies a threshold θ_c. This threshold can also be used to make a decision about H_0.

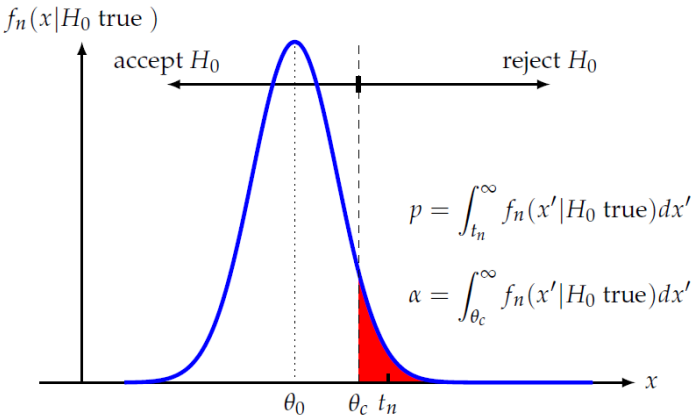


Figure 24: Determining the p-values from the sampling distribution of the test statistic

– **Step 7**: Make a Decision about the Null Hypothesis
In the final step we are making a decision about the null hypothesis. In order to do this there are two alternative ways. First, we can make a decision based on the p-values or, second, we make a decision based on the value of the test statistic t_n.

1. Decision based on the p-values:

$$\text{if } p < \alpha \text{ reject } H_0$$

2. Decision based on the threshold θ_c:

$$\text{if } t_n > \theta_c \text{ reject } H_0$$

In case we cannot reject the null hypothesis we accept it [18].

B Bootstrapping

The bootstrap method is a resampling technique that uses random sampling with replacement, mimicking the sampling process, and it is used in statistics and data analysis. Its purpose it to estimate the sampling distribution of a test statistic or to obtain robust estimates of parameters when the underlying assumptions are unknown or violated. The method works by generating multiple bootstrap samples from the original data set. A bootstrap sample is created by randomly sampling observations from the original data set with replacement, meaning that each observation has an equal chance of being selected in each sample.

Let $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ be a dataset with n observations. The goal is to estimate a population parameter or statistic, standard errors for estimators or the p-values for test statistics under a null hypothesis. The method consists of the following steps:

– **Resampling**: The method starts by generating B bootstrap samples, denoted as $T^{(1)}, T^{(2)}, ..., T^{(B)}$. Each bootstrap sample is obtained by randomly sampling n observations from the original dataset with replacement. This means that each observation has an equal chance of being selected in each bootstrap sample, and some observations may be selected multiple times while others may not be selected at all.

– **Statistic Calculation**: For each bootstrap sample T, the statistic of interest is calculated. Let us denote the statistic of the i-th bootstrap sample as θ_i, where i ranges from 1 to B. These bootstrap statistics represent estimates based on the resampled data.

– **Statistical Inference**: Using the distribution of the bootstrap statistics θ_i $(i = 1, ..., B)$, we can make inferences about the population parameter or estimate the sampling variability. The most common approach is to construct a bootstrap confidence interval, which provides a range of plausible values for the parameter. Alternatively, hypothesis tests can be performed by comparing the observed statistic from the original dataset to the distribution of bootstrap statistics.

In conclusion, by generating multiple bootstrap samples, calculating the desired statistic for each sample, and analyzing the distribution of the bootstrap statistics, we can obtain robust estimates, confidence intervals, and perform hypothesis tests without strong distributional assumptions [20][21].

C Estimating Covariance Matrix $\hat{\Sigma}_0$

The Kullback–Leibler divergence is a type of statistical distance, that is, a measure of how one probability distribution P is different from a second, reference probability distribution Q [22]. More specifically, for two discrete probability distributions P and Q defined on the same sample space $\mathcal{X}$, the relative entropy from Q to P is defined as:

$$D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log \Big(\frac{P(x)}{Q(x)} \Big)$$

which is equivalent to

$$D_{KL}(P||Q) = - \sum_{x \in \mathcal{X}} P(x) \log \Big(\frac{Q(x)}{P(x)} \Big).$$

As mentioned in subsection 3.3, in order to estimate the covariance matrix $\hat{\Sigma}_0$, we minimize the Kullback–Leibler (KL) divergence between two multivariate normal distributions, with mean $\mu = 0$, where one has covariance matrix S and the other has covariance matrix Σ_0. We do that by implementing an optimization algorithm from a `PyTorch` package, and minimizing the KL-divergence, using stochastic gradient descent.