Causal discovery and inference of data from production flows

Jonas Bruun Hubrechts

Kongens Lyngby 2024

Technical University of Denmark
Department of Applied Mathematics and Computer Science
Richard Petersens Plads, building 324,
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Abstract

Casual relations between processes are a core issue for effective control of production systems. In particular, it is paramount to understand how variations, such as delays, propagate through the production system for feasible and robust scheduling of processes.

This thesis aims to uncover such causal relations from observations of production flow runs. To obtain the above, we shall present a method for network deconvolution, where direct and indirect effects are separated. The algorithm's robustness is explored in depth, focusing primarily on the influence of noise. Moreover, we shall extend the algorithm to work with mixed random variables, which is crucial for delay modeling, and discuss in detail how to estimate mutual information for such joint distributions. Furthermore, topological assumptions are considered and shown to improve the method's accuracy.

Applying the method to controlled examples with known causal structures, we observe that chainlike causal structures can be tough to recover. However, assuming a topological order, this problem is greatly alleviated. More complex causal structures are considered, where the method also performs well. In particular, with only a few hundred observations, the algorithm can recover the causal structure.

Finally, the method is applied to a dataset from a simulated pharmaceutical production system. We show how the framework functions in practice and demonstrate the robustness of the inferred causal structure by including more or fewer variables. In particular, we shall observe that almost all delays are unexplained by previous incidents.

Based on the results of this thesis, the algorithm combined with mutual information as a measure of similarity appears to constitute a promising and generally applicable framework for causal discovery. However, observations regarding possible optimizations of estimating mutual information from data are noted. Namely, for random variables that are almost perfectly descriptive of each other, the information is shown to be underestimated. This error results in a bias but is, in most situations, not a problem.

Preface

This thesis corresponds to 30 ECTS credits and was prepared at DTU Compute in fulfilment of the requirements for acquiring an M.Sc. in Engineering.

A special thanks to my supervisors Tobias Overgaard, Nicolai Siim Larsen and Bo Friis Nielsen for their help and guidance throughout this study.

Lyngby, 31-July-2024

Jonas Bruun Hubrechts

Contents

Chapter 1

Introduction

Nowadays, rising customer requirements regarding product quality and quantity are the key performance measures for manufacturers. Pharmaceutical production firms such as Novo Nordisk especially have seen an increase in customer interest [1]. It is thus of great interest how to efficiently schedule such production flows. Reducing the production time and increasing production in general is hence a major challenge for industry but also for academia. In general, complicated production systems, consisting of multiple disjoint processes, can involve many processes that may or may not influence each other and propagate through the production system. In pharmaceutical production systems, this encompasses processes such as filtration, reaction of chemical agents, centrifugation, and many others depending on the drug substance that is to be produced.

The duration of each process, and the quantity being processed of different substances among other factors all have sources of variation. For example, a human-operated part of a process is a known source of variation. How such variations propagate and affect other processes can be hard to detect without extensive exploration and knowledge of the undergoing processes of the system. Hence, it is of great interest from an industrial point of view to efficiently discover such relations and further use these to make informed decisions along the execution of such processes. In particular, sudden deviations and inaccuracies may occur but having a good understanding of the causal effects of the production system and how these deviations are expected to propagate can be crucial for keeping production throughput. More specifically, at some point in the process, a human may need to manually remove deposits from a reaction tank or adjust the pH by adding NaOH. This can influence how long the product stays in the reaction tank and further impact subsequent processes. In particular, it is of interest whether such variation influences a process further down the line directly or if it is an indirect effect i.e. the variation in a process influencing the variation in the next process and so on called transitive effects.

Due to the inherent graphical nature of the problem i.e. inferring the direct effects by filtering out transitive effects, it is natural to take a graphical approach to the problem. Graphical models have been proposed by [2]. Bayesian networks [3] [4] and the strongly related belief propagation algorithm for inference on graphical models [5] are also well-known methods, however, they are all computationally expensive on large scale systems and typically require prior assumptions or are limited to specific applications. We note that there exist many feature selection algorithms, but it is often not inherently clear how to extend these if one wants a detailed structure of how the effects propagate. Namely, feature selection is usually applied if one only wants to understand how a set of variables influences a specific measure. It is thus often not capable of identifying where errors originate and how they propagate.

Thus, the objective of this thesis is to quantify the impact of each process in a production system on a specific measure of performance here taken to be time through a systematic method by inferring the direct dependencies in the network representing true *interactions* thus removing transitive effects. In particular

Given;	historical observations of sojourn times, delays, concentration changes and more from a production system
Determine;	the causal structure and/or dependency relations between the attributes of the processes
Subject to;	limited observations and possible prior topological considerations

Based on existing material on the topic [6] [7] [8] we shall, in particular, investigate the robustness of the method through perfectly controlled simulations and based on theoretical results and extend the algorithm to infer causal direction subject to certain assumptions instead of only direct dependencies which initially are represented by undirected edges. The robustness is considered both in terms of the assumptions driving the algorithm and particularly also the accuracy of the estimator of mutual information which does not seem to be covered in the existing literature.

The rest of the thesis is structured as follows. In chapter 2, we introduce the data simulated from [9] which is subject to multiple error sources that need manual handling. In particular, we present some initial analysis of the simulated observations and conclude that a more advanced method for discovering the direct effects between the durations of each process. In chapter 3 we present the method proposed in [6] and methods for computing and estimating mutual information. We apply the method in chapter 4 and explore potential shortcomings of the method for removing indirect effects and estimating the information between pairs of variables on controlled simulations before finally applying everything to the data from chapter 2. Finally, in chapter 5 we summarize our findings and comment on the properties of the presented methodology.

Chapter 2

Data

In this chapter, we will introduce the pharmaceutical production data, that we shall later use to infer a causal structure pertaining different parts of the production system. In particular, as we are interested in the duration and amount of produced substance during the production flow, these are highly relevant attributes of the processes that make up the production. Hence, we will start this chapter with an overview of the production system, how the observations are structured and created to begin with. For the rest of the chapter, we will concern ourselves with analysis of the production system such as basic statistics, incomplete or wrongly labelled observations and initial observations about codependency (which is very relevant when studying causality).

The observations that we will ultimately use for the causal study is simulated by [9]. However, before diving into how these simulations were carried out, we present the overall structure of the simulated observations and the production system they are supposed to originate from. Namely, a set of 6 cycles, where each cycle consists of a set of batches executed one by one. Thus, as cycle is simply a notion for multiple batches that are executed in continuation of each other. In particular, different settings for the simulation of each cycle have been used to encompass multiple scenarios of how the production system can function. We note that although the cycles are generated from different settings, they are still representative of the same production system. Hence, we shall later combine observations from all cycles.

A batch refers to a collection of processes/unit $\mathcal{U}$ that need to be executed in some order to produce a product. In particular, for this simulation study, each batch is a collection of the processes depicted in Figure 2.1.

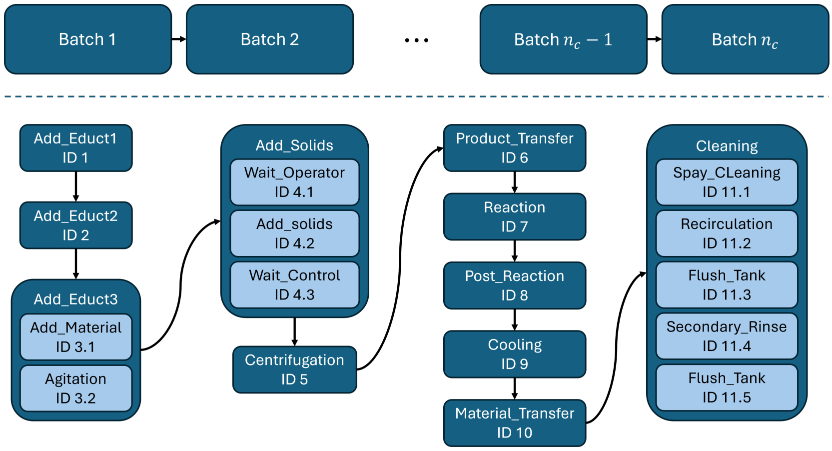


Figure 2.1: The structure of a single cycle. A cycle comprises n_c batches that are carried out one by one. Thus, a cycle is simply a collection of a complex task (a batch), that need to be repeated n_c times. The number of times is often based on time or amount of produced drug or substance, such that a cycle is terminated after these criterions are met. We shall later see that in the case of this simulation study, n_c is determined from the accumulated duration of the batches i.e. after a certain amount of time has been simulated, the simulation is terminated. The structure of each batch is observed in the lower part of the figure and consists of 11 main processes such as addition of solids and chemicals (processes labeled with ID 1 through 4). Each process can be made up of a number of *subprocesses* such as the subprocess with ID 4.3, where the batch waits for a control operator before proceeding.

For each cycle, we then have a time series, where the ID of the batch is given as well as sensory values. The sensors measure the level of the tank (percentage of mow much of the tank is filled), the height (equivalent to the level), the RPM of a motor that circulates the contents of the tank, the cooling water flow (specifically for the cooling process with ID 9) and the steam flow during the reaction process. We shall however restrict ourselves to only using the level sensor in this thesis but including the other variables could improve on our results later on. We have chosen only the level as it is assumed to be the most descriptive of how much product is eventually produced.

In Figure 2.2 an example of the temporal evolution of a process is shown (with the previous process as well). We define T_u^P to be the duration of the process/unit operation u and equivalently M_u^P to be the *change* in level during process u. Not that we have also defined random variables T_u^D and M_u^D. Why we need these will become apparent in Section 2.3 but for now we note that they correspond to delays after each of the processes. In particular, after a process is completed, the might be some downtime in the production system due to unforeseen reasons. We shall later see that for some processes, the delays will not influence the level of the tank whereas the reverse is true for other processes such as the reaction (labelled 7).

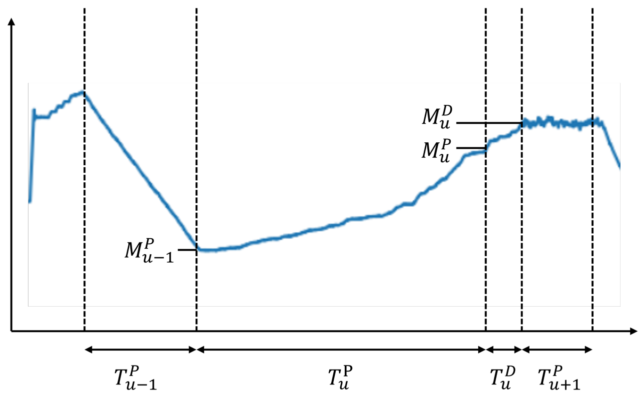


Figure 2.2: Exemplification of the evolution of the level in a tank during a process u and the previous process. The variables T_u^P, T_u^D, M_u^P and M_u^D related to the process are shown. Note that M_u^P and M_u^D are the changes in level from the previous process or delay of process such that they describe the accumulated evolution in the level of the tank during said process. In particular, changes in levels can occur when the production system is idle.

We note that simulations were carried out through a mixture of `Simulink` and `Stateflow` simulations. In particular, the continuous subsystems such as the reaction in process 7 where simulated through `Simulink` based mass-balance equations.

At this point, we have a rudimentary understanding of how the system is simulated and the meaning of the random variables that are related to each process. We thus continue with some basic statistics concerning the durations of batches. For the remaining of the chapter, we will primarily present results for the duration and delays of the processes as the analysis and results are identical to those of the change in level. Namely, we shall observe that the dimension of the random vector that describes each batch (i.e. durations, delays and level changes) large enough such that no meaningful conclusion on the causal relation between random variables can be drawn. In particular, we will need a framework such as the one presented in chapter 3, to discover such relationships.

2.1 Basic statistics

Before analyzing the time series in more depth and filter out (or correct) troublesome data points, we present some initial statistics on the duration of batches for each cycle. The statistics are summarized in Table 2.1 below. We note that some difference is observed from cycle to cycle, but we choose to assume that these differences are simply a feature of the production system, such that later on, we can combine all observations across all cycles into a single dataset to be used for causal discovery.

Cycle	number of batches	mean	variance	standard deviation	coefficient of variation
A	66	14.776	3.641	1.908	0.1291
B	64	15.644	3.915	1.979	0.1265
C	61	17.714	2.330	1.526	0.08617
D	60	18.069	6.922	2.631	0.1456
E	60	18.088	9.613	3.100	0.1714
F	63	17.227	7.766	2.787	0.1618
Combined cycles	374	16.876	7.218	2.687	0.1592

Table 2.1: Basic batch statistics for each cycle and by combining all cycles into a single data set. The average duration of batches across cycles appear similar when taking the variance of the durations into account. We note that later, we wish to estimate the dependency between pairs of random variables whence more observations is better, as always in data science. We do however note that there appears to be a difference between especially the first three cycles and the latter ones. In particular, the variance is larger for cycles named D, E and F. The source of this variation is at this point unknown however it could be seen as a feature of the dataset. Namely, if the observations are truly from the same production system, this variation could be an inherent feature of the production system which we should not remove.

In the following section, we discuss a problem with some of the batches. Namely, the trailing batches, which appear to be cut-off during simulation. In this way, we shall end up with a total of 368 batches, which after some correction (see Section 2.3) will be our final data set.

2.2 Incompleteness of trailing batches

In this section, we shall investigate the combined dataset of 374 batches in more detail. In particular, we shall observe some deviation from Figure 2.1 in terms of labels of each event in the time series and how we have handled these discrepancies. Namely, by looking through the time series for each cycle, we observe entries labeled with negative processes. These, we will investigate the next section and note that from paper introducing the simulations we present here [9], it is by design that some labels are incorrect. Their argument for this is in relation to training a robust machine learning algorithm but as this is none of our concern, we shall manually handle these in correct labels. In particular, the negative labels are initially negated to be positive instead. Hence, we observe events labeled 3, which is not originally a part of the production system description from Figure 2.1. With these negative labels transformed, we count for each of the (new) process labels, how many batches are observed. E.g. how many batches are at some point observed to be undergoing process 1 (the addition of a material). We do this to make sure that in fact every batch go through all processes from Figure 2.1. The result of this counting batches is presented in Table 2.2, where the description of the recognized processes has been copied from [9]. Note that `Educt1`, `Educt2` and `Educt3` are just some (unknown) materials that we do not care about. Note that we have not included

ID	Count	Description
1.0	374	Addition of liquid raw material `Educt1`
2.0	374	Addition of liquid raw material `Educt2`
3.0	181	
3.1	374	Addition of liquid raw material `Educt3`
3.2	374	Agitation
4.0	163	
4.1	374	Waiting for field operation
4.2	374	Addition of solids
4.3	374	Waiting for control operator
5.0	374	centrifugation
6.0	374	Product transfer
7.0	370	Reaction
8.0	369	Post reaction
9.0	369	Cooling
10.0	368	Material transfer

Table 2.2: The number of batches across all cycles that contains at least one observation for each different process label.

labels pertaining the cleaning operation as these will be handled separately in

Section 2.4 where we also argue why we will not use these observations in the later analysis.

Interestingly, the *unrecognized* process labels 3 and 4 only occur for processes with subprocesses. We shall later observe that these labels all originate from negative process labels and that they actually correspond to delays between processes as portrayed in Figure 2.2. For now, we however concentrate on the last four process labels 7, 8, 9 and 10. In particular, as all the other process labels (excluding 3 and 4) appear exactly 374 (the number of batches in total) times, we suspect that something weird is going on with these *missing* observations. As hinted to before, it turns out that the simulations have been cut off after 1100 hours. Therefore, the trailing batch of each cycle does not complete all processes. For example, in Figure 2.3, we have shown the first batch of cycle A as well as the trailing batch and how the over time switch between process labels (not that for this plot, we have not negated the negative process labels)

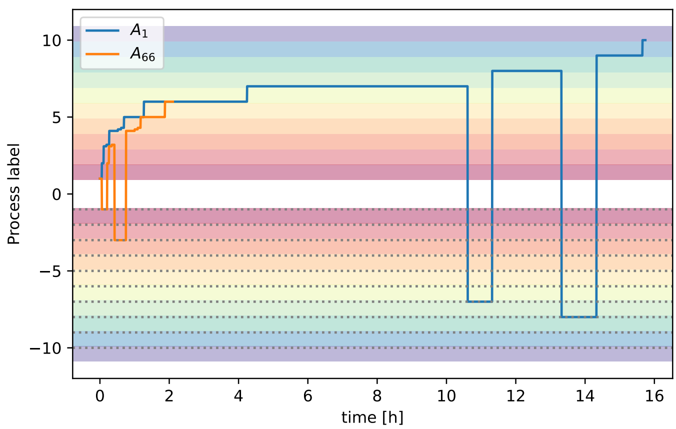


Figure 2.3: The first and last batch from cycle A. The horizontal colored bars corresponds to the different process labels such that time stamps labeled e.g. 3.1 and 3.2 fall in the same colored region. It is clear that the final batch is cut-off before finishing the last process. Furthermore, we observe that the negative process labels for these two batches only occur before the process label enters a new colored region. This hints to the negative process labels are actually delays between processes.

From the above, it is clear that we need to remove the final batches of each cycle. Thus, we now have a total of 368 batches. In the following section, we shall see in more detail when the negative process labels occur and make the assumption that they correspond to delays between processes. Note that the cleaning operation is not considered in the following section.

2.3 Production processes

We now focus on the processes labelled 1 through 10 from Figure 2.1. In particular, we shall denote these processes as *production* processes, as they are exactly the processes where a substance is produced or handled in some other way. Initially, we shall however focus on the first processes up to and including 4.3. Namely, from Table 2.2, we saw that it was these few initial processes where labels seemed to be weird.

In Figure 2.4, we have shown the 22^{nd} batch of cycle B. Once again, we observe the negative process label. We notice that it is only visited once, and only at the of the process which its label corresponds to.

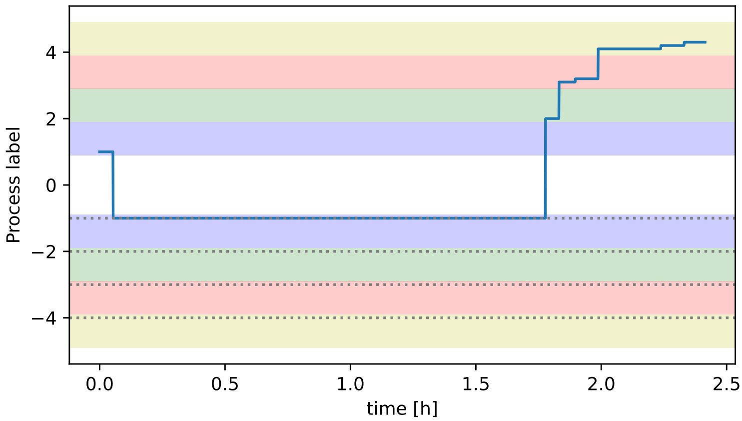


Figure 2.4: The temporal evolution of process labels for batch 22 from cycle B. Only the processes pertaining to the first boxes of Figure 2.1 are shown to easier tell what is happening.

Continuing the investigating, we see that negative process labels occur throughout all the six cycles. Furthermore, by saving what negative process labels have occurred for each cycle, we obtain Table 2.3 where we observe a clear tendency regarding the process labels -4.1, -4.2 and -4.3. Namely, they only occur in cycle F. From [9], we note that cycle F is the only phase containing wrongly labeled time points. In particular, we can conclude that the negative process labels apart from -4.1, -4.2 and -4.3 are not an error in the data set.

In Figure 2.5, we have shown some of the batches which contain the process labels -4.1 etc. We observe that if either of the three process labels are negative, then all of them are and no corresponding positive labels occur. We shall thus

Event \ Cycle	A	B	C	D	E	F
-1	■	■	■	■		
-2				■	■	■
-3	■		■	■	■	■
-4		■	■	■	■	
-4.1						■
-4.2						■
-4.3						■
-5	■	■	■	■	■	■
-6		■	■	■	■	■
-7	■		■	■	■	■
-8	■		■	■	■	■
-9				■	■	■
-10		■		■	■	■

Table 2.3: Occurrences of negative process labels. It is observed that the process labels -4.1, -4.2, -4.3 only occur in cycle F which is known to be the only cycle with wrongly labelled phases.

assume that whenever -4.1, -4.2 or -4.3 is observed, it is actually just the negated process label. Correcting the data set under this assumption, we then only have negative process labels that are integer which we have summarized in Table 2.4 below.

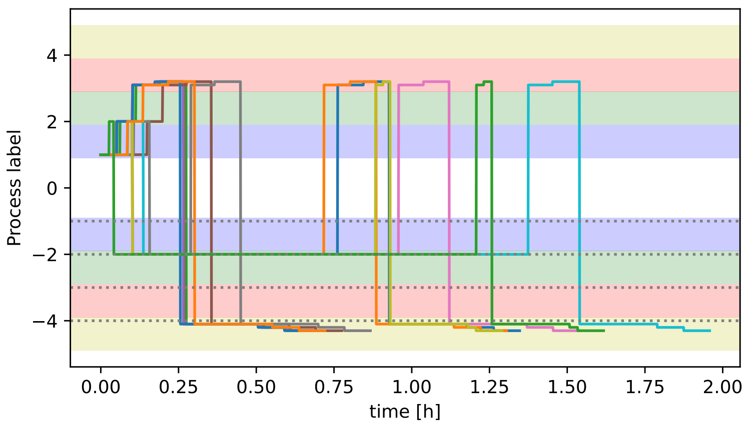


Figure 2.5: 13 out of the total 48 batches where at least one of the process labels -4.1, -4.2 or -4.3 were observed.

Event \ Cycle	A	B	C	D	E	F
-1	■	■	■	■		
-2				■	■	■
-3	■		■	■	■	■
-4		■	■	■	■	
-5	■	■	■	■	■	■
-6		■	■	■	■	■
-7	■		■	■	■	■
-8	■		■	■	■	■
-9				■	■	■
-10		■		■	■	■

Table 2.4: In the modified data set, where -4.1, -4.2 and -4.3 have been converted their absolute value. We observe that the occurrence of negative labels is not identical from cycle to cycle. Depending on the parameters of the simulation, this could either be per happenstance on different settings of the simulation. Either way, we assume that as the simulation is based on the same production system, this variation is observed naturally. Hence, we shall not do more with these observations in terms of filtering them out or correcting them.

From Table 2.4, we see that cycles D, E and F appear to contain more negative process labels. We will however assume that the cycles are simulated from the same production system such that no hyperparameters were changes. In particular, we shall assume that the observations of negative process labels occurs at random, independently of the cycle.

Furthermore, by plotting the different batches from different cycles, it is apparent that the negative process labels always occur after each process (including subprocesses) and before the next process. I.e. we only observe the label -1 after 1 and before 2. Likewise, -3 only occurs after both 3.1 and 3.2 but before 4.1. As hinted to before, we shall thus assume that these negative process labels corresponds to delays between processes. This does make sense from a production point, but they also note in [9] that delays between operations have been implemented.

At this point, we finally have a sufficient understanding of the simulation to define a few random variables. For each main process $u \in \{1, \dots, 10\}$, we define the *delay* after the process as T_u^D and the associated change in level M_u^D. Similarly, for the actual processes $u \in \{1, 2, 3.1, 3.2, 4.1, \dots, 10\}$ we have *process* duration T_u^P and likewise M_u^P. Converting the time series data to realizations of the random variables, we find that $T_{4.1}^P$, $T_{4.3}^P$ and T_8^P are always constant. Referring once again to [9], we see that indeed these processes are controlled such that the duration is 15 min, 5 min and 2 hours respectively. As these random variables are then not really random but constant, we exclude them from our analysis from this point onward. Note that the delays T_u^D have an atom at 0 since there is a possibility that there is no delay between processes.

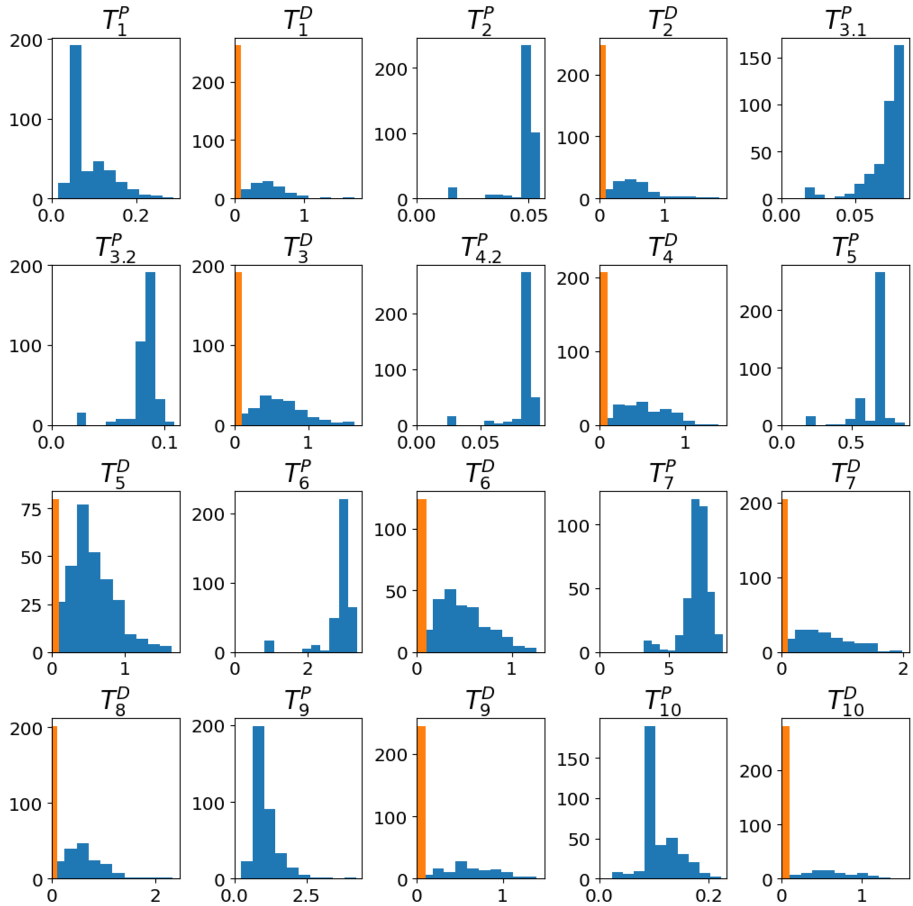


Figure 2.6: Histograms of all random variables T_u^D and T_u^P that are non-constant i.e. not controlled to be a fixed amount of time. The orange bars for T_u^D signify the occasions where no delay after process u took place. We observe that depending on the process, a delay is more or less common. In particular after process 5, there seem to a delay often whereas a delay of process 10 is very rare.

In Figure 2.6, we have shown histograms of frequencies for each of these random variables. We note that for the distribution of observations appear to be unimodal i.e. it does not appear as if they are a mixture of distributions. This is important, as it further strengthens our assumption that the hyperparameters of the simulation where the same for all cycles. In particular, the delays (when disregarding the atom at 0) do not appear as if they originated from different distributions as then we would likely have observed clusters for each of the cycles.

In the next section, we shall further examine the random variables in terms of the correlation structure of the observations. However, we first present some basic statistics of the random variables in Table 2.5 for each cycle. From the

Cycle	A	B	C	D	E	F
$\widehat{\mathbb{E}[\sum T_u^P]}$	13.993	13.898	15.343	14.471	14.589	14.418
$\widehat{\mathrm{Var}(\sum T_u^P)}$	0.95636	0.46587	0.76111	4.9589	4.2678	5.3545
$\sum \widehat{\mathrm{Var}(T_u^P)}$	0.50590	0.31182	0.36667	1.8322	1.5788	1.9696
$\widehat{\mathbb{E}[\sum T_u^D]}$	0.96398	1.9402	2.4503	3.6050	3.7390	3.0041
$\widehat{\mathrm{Var}(\sum T_u^D)}$	0.31843	0.39117	0.90187	1.2468	1.2787	1.0462
$\sum \widehat{\mathrm{Var}(T_u^D)}$	0.34921	0.53198	0.74914	1.4357	1.2454	1.3099
$\widehat{\mathbb{E}[\sum T_u^P + T_u^D]}$	14.957	15.838	17.793	18.076	18.328	17.422
$\widehat{\mathrm{Var}(\sum T_u^P + T_u^D)}$	1.1500	1.1352	1.7983	7.0000	5.7086	5.0103

Table 2.5: Mean and variance overview of each of the time related variables T_u^P and T_u^D. A clear difference of variance is observed between cycles $A-C$ and $D-F$. Furthermore, the durations T_u^D clearly show that the total variation is accounted for by the variance of the individual durations. The contrary holds true for the delays T_u^D.

table, we observe that the average total durations (excluding delays) of batches are approximately the same. However, the variance of the sum of durations is significantly larger for the cycles D, E and F. What causes this difference in variation is however unknown. Ideally, in a real world application, one should investigate this further. On could argue that cycles $A-C$ and $D-F$ should then be treated separately and indeed the variance for the accumulated delay during a batch exhibit the same behavior. Namely, the delays also seem to have a larger variance in the later three cycles. We shall however treat all the batches simultaneously in this thesis by arguing that although there is a clear difference in variation, the underlying causal structure of the processes is assumed to be the same. In particular, we could not infer from the simulation study [9] that the cycles should have been based on different hyperparameters resulting in the below difference of variances. Furthermore, we note that the difference between the variation in the accumulated duration and the sum of process durations i.e. $\mathrm{Var}\,\widehat{(\sum T_u^P)} - \sum \mathrm{Var}\,\widehat{(T_u^P)}$ indicate that the durations of the processes are not unrelated. Thus, in the next section we shall investigate the correlation structure of the variables. Finally, we note that the same difference for durations does not indicate a relationship. However, the missing covariances might just cancel each other out. Hence, we do not yet conclude anything regarding their causal structure.

2.3.1 Correlation structure of durations and delays

In this section, we proceed with investigating the correlation between pairs of the random variables. Based on the observations, we quickly compute a correlation matrix as observed in Figure 2.7.

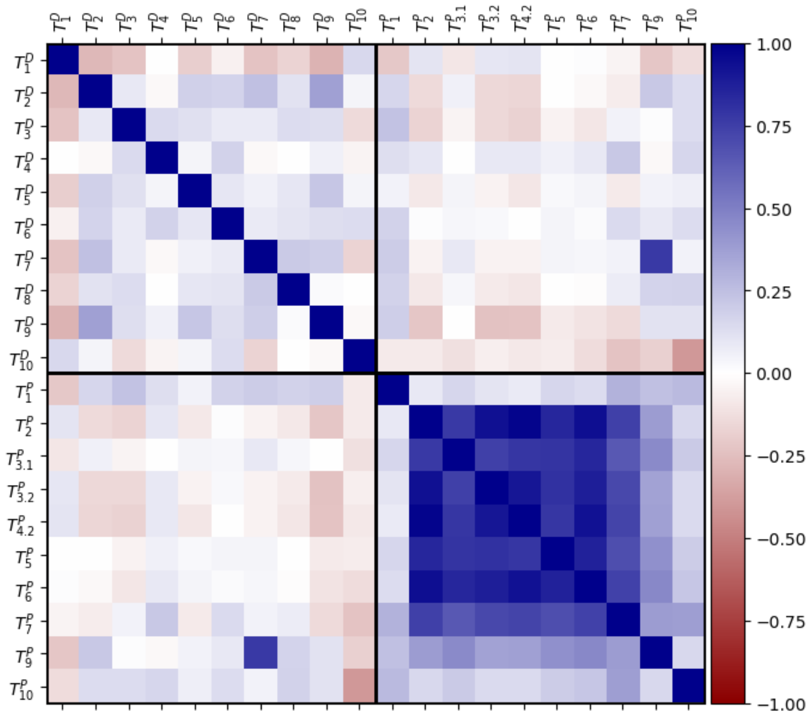


Figure 2.7: Estimated correlation matrix of the random variables T_u^P and T_u^D.

We immediately notice that the durations of the processes appear to be positively correlated. The fact that there exists strong correlations is not surprising when considering Table 2.5 where we observed that the variation of the sum of durations was much larger than the sum of individual variances. The only other immediately interesting observation that we make from the above figure is the large correlation between T_7^D and T_9^P. This means that there is a positive linear relationship between the delay after the reaction process 7 and the time it takes to cool the tank (process 9).

To assess the significance of these correlations, we performed a permutation test. Namely, by randomly permuting the observations of each random variable and recomputing the correlation matrix, we see if the new correlation coefficient is numerically larger than the one we computed from the original data. Repeating this multiple times (e.g. 10,000 times), we obtain an estimate of the probability of observing a correlation coefficient, numerically larger than the one we computed in Figure 2.7. In particular, the null hypothesis is that there is no cor-

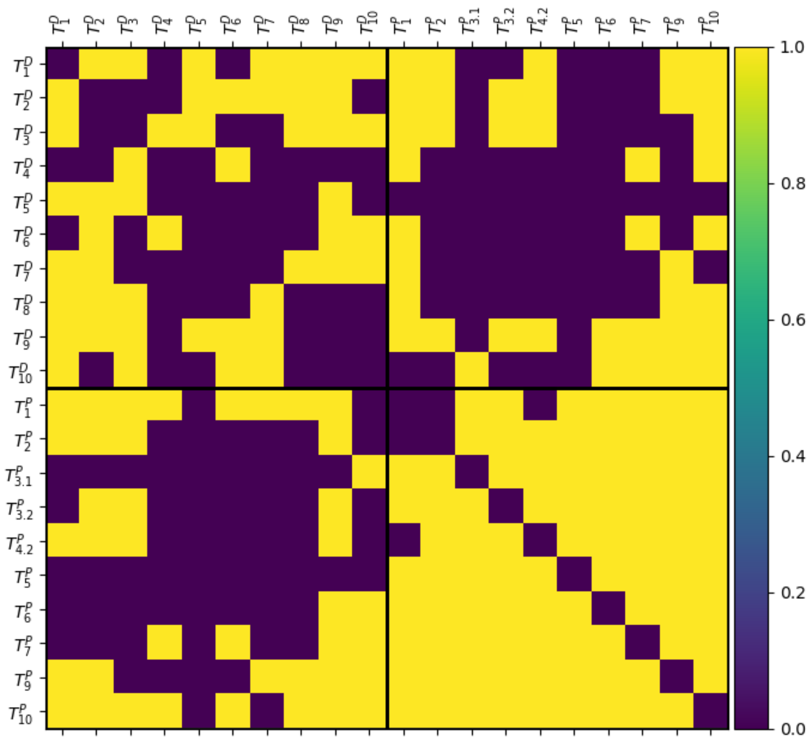


Figure 2.8: Permutation test with significance level $\alpha = 0.05$ based on 10,000 permutations. The diagonal elements have been set to 0 as it does not make sense to test the correlation between a random variable and itself.

relation between a pair of random variables. This is because when the samples are reshuffled independently of each other, the underlying assumption is that the random variables are independent of each other and hence has correlation 0. In Figure 2.8 we have shown a binary matrix for when the p-value was observed to be less than 0.05 i.e. significant results on a 5% significance level. We observe that many of the correlations appear to significant which is somewhat contradictory to what we would expect from such a production system. However, as we are performing multiple test, we really should correct for this in some way. Choosing a conservative approach through the Bonferroni correction, we get far fewer significant results as observed in Figure 2.9. Once again, we have removed the diagonal elements.

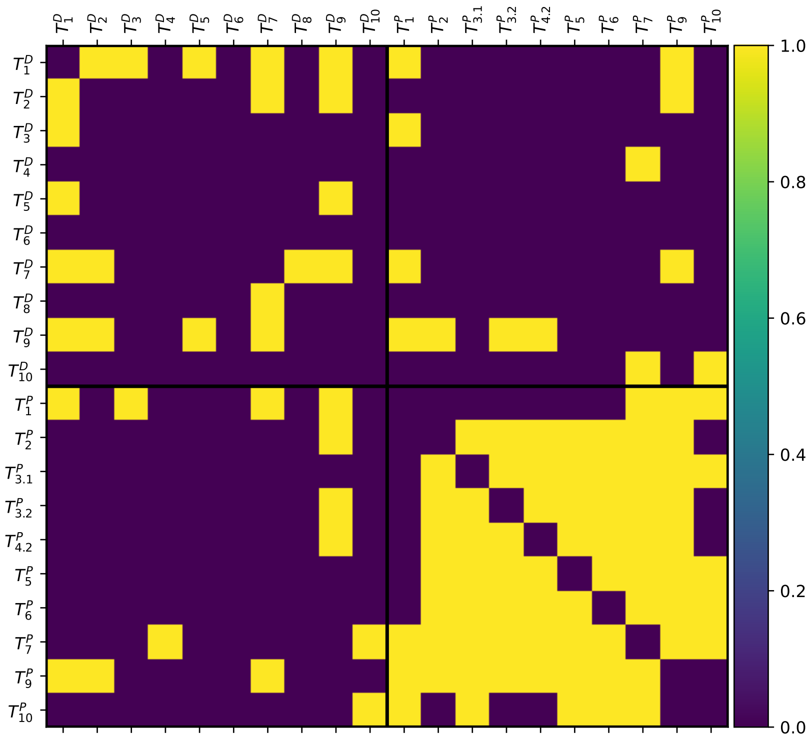


Figure 2.9: The permutation test with Bonferroni corrected significance level. Many of the significant correlations disappear however when plotting e.g. T_1^D vs T_2^D we see that the correlation does not stem from a linear relationship. In particular, from Figure 6.1, we observe that the joint distribution is more of an L-shape. We shall thus later investigate other measures of similarity than correlation to capture such non-linear tendencies

Suppose now that we instead did not know what part of a process what the duration and what was a delay. In particular, we define the total duration of a process as $T_u = T_u^P + T_u^D$. Note that for T_3 and T_4, we extend this definition slightly such that $T_3 = T_{3.1}^P + T_{3.2}^P + T_3^D$ and $T_4 = T_{4.1}^P + T_{4.2}^P + T_{4.3}^P + T_4^D$ (although $T_{4.1}^P$ and $T_{4.3}^P$ can be disregarded as they are constant and hence irrelevant when computing the correlation). Using only the cumulated random variables, we observe a much simpler correlation structure as shown in Figure 2.10. However, we also seem to lose much of the information that was otherwise visible in Figure 2.7.

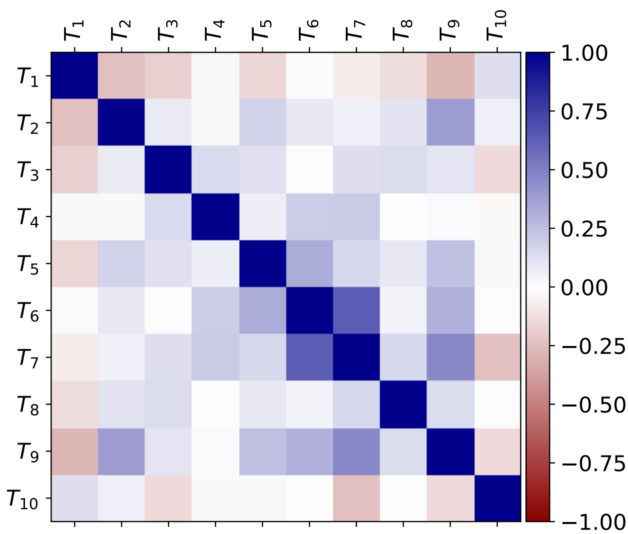


Figure 2.10: Correlation matrix for combined process durations T_u. The strong correlations that we observed previously appear to be lost. In particular, it is no longer clear what durations influence each other if any. Thus, we conclude that at least in the case of linear relations between durations, the extra information regarding when the system is idle due to a delay, is very important if we want to have any change of inferring anything useful about the causal structure of the production system.

As a final remark, we have shown in Figure 2.11 the duration of a process T_u and the process that follows immediately after T_{u+1}. Although the correlations in Figure 2.10 did not appear as informative as in Figure 2.7, there might still be some useful observations to be made from the joint raw observations. Note that Cycle 1 corresponds to Cycle A and so on. From the below figure, it is clear that the duration of a process is not well-described based solely on the previous process. However, we do observe an interesting behavior in T_2 vs T_1 and T_{10}

vs T_9. Namely, for T_2 and T_1, T_1 appear to be larger in the first 3 cycles while T_2 appears to be larger in the final three cycles. This results in the L-shape observed. A similar observation is made between T_{10} and T_9. From Table 2.4, we see that this coincides with the existence of delays in the cycles respectively. Furthermore, from Figure 2.6, the durations often last much longer than the actual duration of the process, such that the delay dominates the total duration of the process.

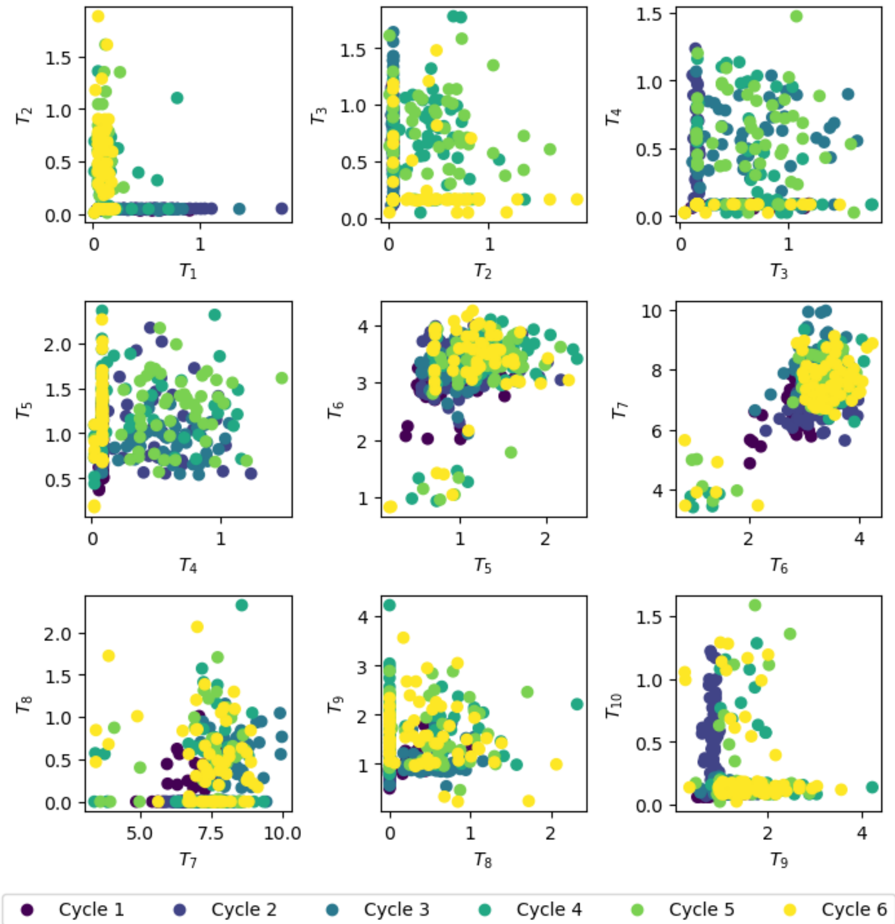


Figure 2.11: Total process durations T_u plotted against the next process T_{u+1}. In some cases, there seem to be a difference from cycle to cycle. Especially in T_2 vs T_1 and T_{10} vs T_9.

For the sake of completeness, we have in the appendix, Figure 6.1, Figure 6.2 and Figure 6.3 plotted all variables against each other. Although clear relations can be observed between some pairs of random variables it is unclear how e.g. durations and delays (and the related level changes) propagate through the system if they even do so. In chapter 3, we will discuss a method for discovering such relations, but first, we comment on the cleaning operations that until this point has been left out.

2.4 Cleaning operations

As per Figure 2.1, after each batch, the tank in which the process has taken place is to be cleaned. However, from inspection of the data, we observe that only for cycles A and B is this true. Namely, for cycles C through F, the tank is not cleaned after each batch. This, along with some basic statistics regarding the duration of the cleaning operation is summarized in Table 2.6.

Cycle	number of cleanses	min	max	sample mean	sample variance	sample standard deviation	CV
A	65	1.113	3.067	1.917	0.269	0.518	0.270
B	63	1.324	1.751	1.566	0.00883	0.0939	0.0600
C	9	1.544	3.306	2.153	0.277	0.526	0.245
D	10	1.474	2.009	1.581	0.0212	0.146	0.0922
E	10	0.827	1.584	1.465	0.0462	0.215	0.147
F	10	0.748	1.610	1.466	0.0595	0.244	0.166

Table 2.6: Per cycle cleansing statistics. We observe that cycles A and B have many more cleaning processes taking place than cycles $C - F$. Also, the variation in cycles A and C is much larger than for the other cycles. These observations indicate that the cleaning operations from the different cycles have not been simulated from the same hyperparameters. This apparent inequality between the cycles is the primary reason for why we have not chosen to use them in our further analysis of the production system.

The most notable differences from cycle to cycle are the number of cleanses as each cycle have approximately the same number of batches as seen in Table 2.1. For the first two cycles, the cleanses seem to be in between every batch, which is indeed the case. Furthermore, although the cleanses are between every batch for cycles A and B, the variances are extremely different. Also, we observe that cycles A and C have much larger variation than the remaining cycles. With these observations, we hypothesize that the cleaning operations have not been simulated based on the same settings whence we will not consider these processes in our further analysis.

Furthermore, histograms of the durations of cleaning processes for each cycle in Figure 2.12 show that although cycle B has many more cleanses, the distribution is somewhat comparable to those of cycles $D-F$ as the durations are approximately centered around 1.55 with similar interquartile range.

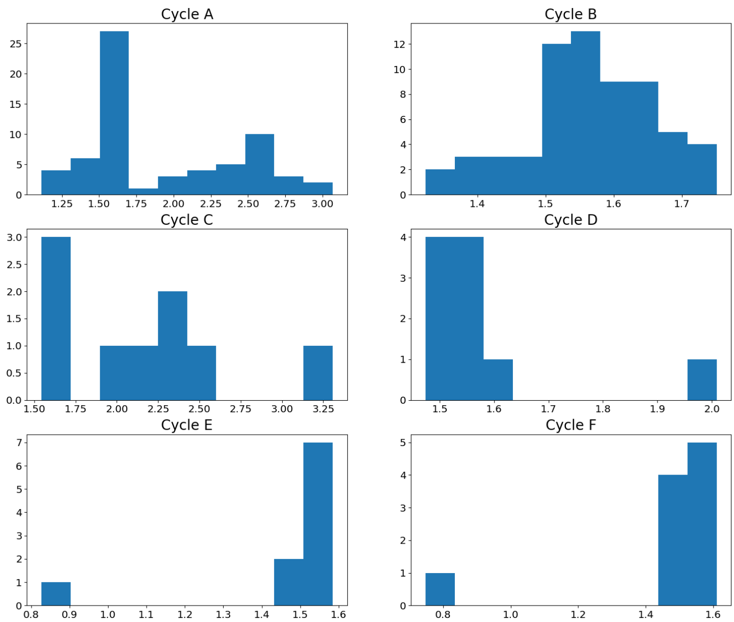


Figure 2.12: Each of the 6 cycles, cleaning operations histograms.

In the appendix, Figure 6.4 and Figure 6.5, we have shown how a cleaning operation goes through each of the subprocesses. Indeed, for cycles $C-F$ the cleaning process is only carried out every so often as summarized in Table 2.7. From Figure 6.5, the cleaning processes appear to be carried out between a random number of batches. In the remaining part of this section, we shall investigate this observation for the latter four cycles with some simple tests.

In particular, let C_i denote the indicator of whether the ith batch is followed by a cleaning of the tank or not. We shall then investigate whether the next batch is also cleaned or not. I.e. at a lag 1, if the variables C_i are associated. In particular, we shall use Fisher's exact test with the alternative hypothesis being two-sided. We use the `SciPy` implementation `stats.fisher_exact`. The results are summarized in Table 2.8. Indeed, we observe that the results are non-significant on a 5% significance level. Repeating the tests for lags up to

Cycle	Percentage cleaning processes after batches
A	100.00
B	100.00
C	15.00
D	16.95
E	16.95
F	16.13

Table 2.7: Per cycle, how often a batch is followed by a cleaning process.

and including 10 we observe no significant results either. More sophisticated tests could be carried out to test if using e.g. the previous 5 C_i is predictive of whether the next batch is followed by a cleaning process. However, as we shall not use this later, we end our discussion of the cleaning processes and note that in the remaining of the thesis, they have been excluded from the dataset.

C_i \ C_{i+1}	No	Yes
No	41	9
Yes	9	0

(a) Cycle C, $p = 0.3293$

C_i \ C_{i+1}	No	Yes
No	41	8
Yes	7	2

(b) Cycle D, $p = 0.6456$

C_i \ C_{i+1}	No	Yes
No	41	7
Yes	7	3

(c) Cycle E, $p = 0.3532$

C_i \ C_{i+1}	No	Yes
No	41	9
Yes	9	1

(d) Cycle F, $p = 1.0000$

Table 2.8: Contingency table for Cycle C-F at lag 1. No significant results are observed and repeating for lags larger than 1 we do not conclude otherwise.

Chapter 3

Method

The following chapter is structured as follows. Initially, we shall introduce the basic concept of causality and *structural causal models* (SCMs) based on [10]. From that, we shall discuss the method proposed by [6] to infer such SCMs. In particular, we shall state the underlying assumptions of the method and discuss the implication of these. Furthermore, we shall see how based on observed similarities between pairs of random variables, the proposed method deconvolves a matrix of similarities and result in a set of proposed structures for the underlying SCM. Namely, how we from data can make predictions of which variables influence each other directly or through mediators - also known as in-between variable.

From the basic assumptions, we shall then discuss correlation and mutual information as similarity measures and how they differ. In particular, we shall see that mutual information and copulas, a statistical tool for isolating the joint behavior of random variables, are very related topics. In particular, mutual information can be computed from only the copula. Furthermore, we shall extend on the original methodology by considering mixed random variables as well. This is important, such that the delays T_u^D from chapter 2 can be included later on in chapter 4, where we shall use the methods obtained here to infer possible causal structures.

In Section 3.3, we will discuss the algorithms to be used later in detail. In particular, we shall clarify a few results from the original paper [6] and extend on their results on robustness of the deconvolution algorithm. A major assumption regarding the *size* of the observed matrix of associations (G_{obs}) is removed, obtaining a more general and useful result. The Frobenius and maximum matrix norms are especially considered.

Finally, in Section 3.4, we shall discuss different methods for estimating mutual information as well as their drawbacks. Especially kernel based density estimators are considered discussed in terms of a simple dataset from [11] with observations related to suicide risk.

General introduction to method and causality

3.1 Causality and Causal Discovery

In this section, we shall discuss the method for network deconvolution, originally proposed by [6]. The underlying problem is inferring direct effects and dependencies. From this, using prior information on the production setup, we shall be able to infer causal dependencies by directing the resulting edges from the network deconvolution (ND) algorithm. Particularly, the framework and general algorithm proposed by Feizi et al. stems from a graph-theoretic approach to the problem of inferring direct dependencies. Namely, suppose that observations depend on quantities such as levels and sojourn times of in this case a chemical process. We shall represent these properties as vertices (nodes) V and dependencies between properties as edges. Initially, when observing the vertices, we observe both direct and indirect effects. Particularly, a vertex v_1 might influence some other vertex v_3 through another vertex v_2 if v_2 depends on v_1 and v_3 of v_2. In this case, we will observe that v_1 influences v_3, but actually it is v_2 that has a direct influence on v_3. In graph-theoretical terms, we thus observe the transitive closure of the information that flows between vertices but want to infer the underlying network structure.

An important note on the algorithm to come is that we only use vertices that we have observed. Namely, the underlying structure might be as in Figure 3.1(a) with an unobserved node/variable (named U in this case). However, without any more assumptions or modelling choices we would (ideally) infer the network structure depicted in Figure 3.1(b). With these initial comments, we proceed with the general setup and assumptions for network deconvolution based on observations.

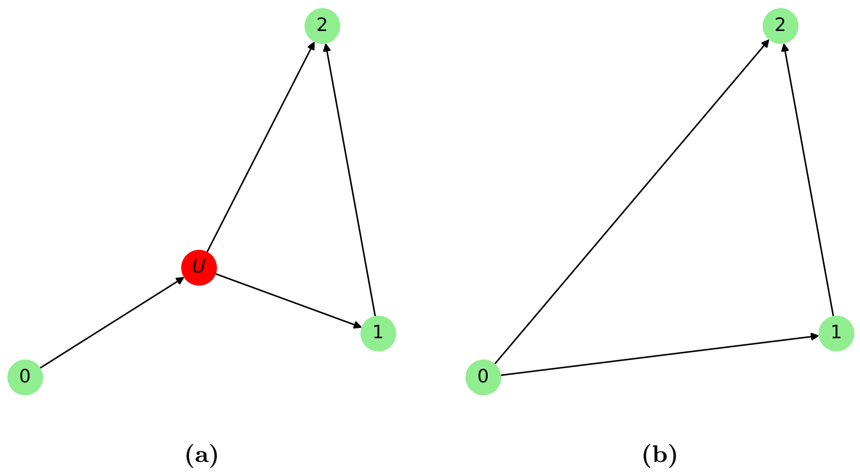


Figure 3.1: (a) An example of a causal structure depicted as a graph. When observing the network, only nodes 0, 1 and 2 are observed/recorded. (b) The resulting inferred graph from observational data. Although this is not a complete picture of the true underlying dynamics of the system, if only the observed variables are of interest, this will be an equally proper representation of the system. Furthermore, in practice this means no further assumptions are made which can and can not be of desire. Namely, if prior information is accessible one might introduce new nodes in the inferred network.

3.1.1 Setup and Assumptions

Suppose a set of d random variables (X_i) is given i.e. a d-dimensional random vector $\boldsymbol{X}$. The method presented in this section aims to discover direct relationships between pairs X_i and X_j for $i \neq j$. These relationships will be presented by a directed graph as in the previous section or an undirected graph in case the causal direction is either unknown or such an assumption on direction is not plausible. In particular, we shall let each random variable X_i be represented by a vertex in a graph. We will later discuss a way of directing edges such that a causal network may be discovered i.e. a directed acyclic graph that may be used for inference.

The method proposed by [6] then works as follows. Given an observed matrix $G_{obs} \in \mathbb{R}^{d \times d}$ of similarities between each pair of variables, we shall deduce a matrix $G_{dir} \in \mathbb{R}^{d \times d}$ of direct similarities between each pair of random variables X_i and X_j. In particular, we wish to filter out indirect effects which we will denote by G_{dir} defined as effects between pairs of variables that is the result of effects propagating through other variables. The measure of similarity, can in practice be any desired measure such as correlation or mutual information which we will focus on in this thesis. See Section 3.2 for a further discussion on these two measures and Section 3.3 for how to obtain such a matrix. Note that the algorithm presented will in theory work for non-symmetric measures as well such as *Interaction information*, *Directed information* and *Normalized information*.

Figure 3.2: On overview of the methodology that we shall develop in this thesis. In particular, the estimation of mutual information - a possible measure of similarity - and deconvolution will be discussed in this section.

The (direct) network is then presented by the discovered G_{dir} containing only the direct effects i.e. interaction between pairs of variables which can be viewed as weights on the edges of the complete graph with nodes representing the random variables. As we shall see in Subsection 3.3.3, the algorithm is somewhat robust to noise in the sense that we can ensure accuracy depending on the level of noise observed present in G_{obs} and on the norm chosen (from a certain, although rather general, set of norms). Namely, if G_{obs} is subject to noise, we find a bound on how different the inferred directed effects can be to the true direct effects using different matrix norms to measure this difference. This hints to that a threshold on the inferred weights on the edges of the network might be a good idea to remove small inferred effects. This is further supported by the facts that often only the most influential variables are of importance when trying to control the process.

The first assumption is that the observed matrix of co-dependence G_{obs} may be expressed as

$$G_{obs} = G_{dir} + G_{indir} \tag{3.1}$$

Namely, that the direct and indirect effects can be added together to get the total and thus observed interdependence between each pair of variables. Often, this is not the case as we shall see later on. However, the error made from this assumption and the ones to be presented seem to be small enough that the discovered network accurately resemble the true underlying network.

The second and final assumption is that the indirect effects G_{indir} can be computed in terms of G_{dir}. Namely, that

$$G_{indir} = G_{dir}^2 + G_{dir}^3 + \cdots = \sum_{k=2}^{\infty} G_{dir}^k \tag{3.2}$$

i.e. that the observed *information* exchanged on an edge e_{ij} between nodes X_i and X_j is the sum of the second, third etc. order effects, each given by the information on the n-path (where n is the order of the (diminishing) indirect effect) again assumed to be a sum of products. In other terms, the second order indirect effect between X_i and X_j (given as the (i,j) element of G^2_{dir}) is the sum of products on edges e_{ik} and e_{kj} for all k

$$\left[G^2_{dir}\right]_{ij} = \sum_{k=1}^{d} e_{ik}\, e_{kj}$$

where e_{ij} is the (i,j) element of G_{dir}. This is of course not true in general. However, through error analysis in Subsection 3.3.3 and controlled examples in chapter 4 we shall see that this assumption is either true under some additional assumptions or only results in small numerical errors. Immediately, we observe that e_{ii} is of interest in terms of its physical meaning. The co-dependence between a random variable and itself might be somewhat ambiguous or even undefined depending on the measure. Thus, the notion of (non-existing) edges e_{ii} will be of interest later on when using the method on controlled cases. We note that in G_{obs} we shall in general set these elements to 0.

Thus, from the above assumptions, it follows that we can express G_{obs} as

$$G_{obs} = G_{dir} + G^2_{dir} + G^3_{dir} + \cdots = G_{dir} + G_{dir}\, G_{obs} \tag{3.3}$$

Clearly, such a G_{dir} must have spectral radius at most 1 as otherwise, the above sum diverges and thus G_{obs} will not exist. I.e. $\rho\left(G_{dir}\right) < 1$, where $\rho\left(\cdot\right)$ denotes the spectral radius. Thus, assuming convergence we can rewrite the infinite series as

$$G_{obs} = G_{dir}\left(I - G_{dir}\right)^{-1} \tag{3.4}$$

Multiplying the above by $\left(I - G_{dir}\right)$ from the right and moving around terms, it immediately follows that

$$G_{obs} = G_{dir}\left(I + G_{obs}\right) \tag{3.5}$$

Thus, if we can show that $-1 \notin \sigma\left(G_{obs}\right)$ (where $\sigma\left(\cdot\right)$ denotes the spectrum of an operator), we can isolate G_{dir}. Namely, we need -1 to not be an eigenvalue of G_{obs}. This is indeed true under the assumption that $\rho\left(G_{dir}\right) < 1$. In particular, assume that (λ, v) is an eigenpair of G_{dir}. Then, by assumption $|\lambda| < 1$ and by Equation 3.3:

$$G_{obs}\, v = \sum_{k=1}^{\infty} G^k_{dir} v = \sum_{k=1}^{\infty} \lambda^k v = \frac{\lambda}{1-\lambda} v$$

where we used that v is an eigenvector of G_{dir} and the geometric series converges as $|\lambda| < 1$. In particular, $\left(\frac{\lambda}{1-\lambda},\, v\right)$ is an eigenpair of G_{obs}. In Section 6.1, we show that λ is an eigenvalue of G_{dir} if and only if $\frac{\lambda}{1-\lambda}$ is an eigenvalue of G_{obs} i.e. there is a bijection between the eigenvalues of G_{dir} and G_{obs}. Thus, as the spectral radius of G_{dir} is less than one such that $\lambda \in (-1, 1)$, we conclude that $\sigma(G_{obs}) \subset \left(-\frac{1}{2}, \infty\right)$. Hence, $-1 \notin \sigma(G_{obs})$ and G_{dir} can easily be isolated in Equation 3.5 as

$$G_{dir} = G_{obs}\left(I + G_{obs}\right)^{-1} \tag{3.6}$$

We note that from the above, we have that G_{obs} is a result of a G_{dir} only if the smallest eigenvalue of G_{obs} is larger than $-1/2$.

Furthermore, if the measure of dependence between pairs of variables is symmetric, then so is G_{obs} and hence diagonalizable by some orthogonal matrix U (such that $U^T = U^{-1}$) and diagonal matrix Λ_{obs} such that $G_{obs} = U\Lambda_{obs}U^T$ (with the columns of U being right eigenvectors of G_{obs}). This follows from the fact that any real symmetric matrix is diagonalizable. It follows that G_{dir} can be expressed in the following simple way (which is useful for computational efficiency)

$$\begin{aligned}
G_{dir} &= U\Lambda_{obs}U^T\left(I + U\Lambda_{obs}U^T\right)^{-1} \\
&= U\Lambda_{obs}U^T\left(UU^T + U\Lambda_{obs}U^T\right)^{-1} \\
&= U\Lambda_{obs}U^T\left(U\left(I + \Lambda_{obs}\right)U^T\right)^{-1} \\
&= U\Lambda_{obs}U^T U\left(I + \Lambda_{obs}\right)^{-1}U^T \\
&= U\Lambda_{obs}\left(I + \Lambda_{obs}\right)^{-1}U^T \\
&= U\Lambda_{dir}U^T
\end{aligned}$$

where $\Lambda_{dir} = \Lambda_{obs}\left(I + \Lambda_{obs}\right)^{-1}$ is also a diagonal matrix, with elements corresponding to the inverse of the mapping $\lambda \mapsto \frac{\lambda}{1-\lambda}$.

As we shall later use some assumptions regarding causality leading G_{obs} to be a triangular matrix, we shall investigate the properties of the resulting G_{dir}. Namely, in the following, we show that given the existence of G_{dir} (with necessary and sufficient conditions on G_{obs} as given above), G_{obs} is triangular if and only if G_{dir} is triangular. Thus, by directing the observed similarity (by removing half the edge weights in G_{obs}), we also infer a directed graph G_{dir}.

Clearly, if G_{dir} is triangular, so are the powers G_{dir}^i for all $i \in \mathbb{N}$ and hence if the infinite sum $\sum_{i=1}^{\infty} G_{dir}^i$ converges, G_{obs} is triangular as well.

To show the other way, assume that G_{obs} is triangular and is the result of a G_{dir} with spectral radius smaller than 1. By Equation 3.6, G_{dir} is triangular if the inverse of $I + G_{obs}$ is triangular (upper triangular if G_{obs} is also upper triangular and similarly for lower triangular). This is indeed the case as in general, the inverse of a triangular matrix is also triangular provided that the diagonal elements are non-zero. Note that $I + G_{obs}$ is never 0 in the diagonal, as $-1/2$ is the smallest possible eigenvalue of G_{obs} and hence smallest diagonal element. A simple proof is as follows. Assume without loss of generality, that a matrix T is upper triangular. Let D be the diagonal elements of T and T_u be the remaining strictly upper triangular part of T such that $T = D + T_u$. Then, assuming that D has non-zero diagonal elements, $T = D\left(I + D^{-1}T_u\right)$. Therefore,

$$\begin{aligned} T^{-1} &= \left(I + D^{-1}T_u\right)^{-1} D^{-1} \\ &= \sum_{i=0}^{\infty} \left(-D^{-1}T_u\right)^i D^{-1} \\ &= \sum_{i=0}^{d-1} \left(-D^{-1}T_u\right)^i D^{-1} \end{aligned}$$

which is clearly also upper triangular. The second and final equality follows from T_u being strictly upper triangular and thus nilpotent such that $\sigma\left(-D^{-1}T_u\right) = \{0\}$ and $T_u^d = 0$. We conclude that G_{obs} is triangular if and only if G_{dir} is (under the assumption G_{dir} exists and G_{obs}).

Finally, before discussing the implementation and analyzing the algorithm both analytically and through examples, we will take a closer look at the similarity measures that are to be used with this method and that in the end will make up the matrix G_{obs}. Namely, *mutual information* and *correlation*.

3.2 Information Measures and Computation

In this section we discuss two measures that can be used to construct the matrices of codependency from the previous section. Namely, we shall touch on correlation and discuss what one might choose to call Copula-based entropy. However, before discussing Copula entropy (CE) we first need to define what a copula is.

3.2.1 Copula

Given a set of d random variables $X_1, \ldots, X_d$, a copula is loosely speaking a distribution function with domain $[0, 1]^d$ incorporating the dependence structure between the random variables. Given a joint distribution function F for

$(X_1, \ldots, X_d)$ and (invertible) marginals $F_1, \ldots, F_d$ we define a copula C as

$$\begin{aligned} F(x_1, \ldots, x_d) &= \mathbb{P}\left(X_1 \leq x_1, \ldots, X_d \leq x_d\right) \\ &= \mathbb{P}\left(F_1\left(X_1\right) \leq F_1\left(x_1\right), \ldots, F_d\left(X_d\right) \leq F_d\left(x_d\right)\right) \\ &= C\left(F_1\left(x_1\right), \ldots, F_d\left(x_d\right)\right) \end{aligned}$$

Letting $u_i = F_i\left(x_i\right) \in [0,1]$ it is clear that C is a distribution function as described above [12]. Furthermore, it follows that the marginals of C are uniform as $F_i\left(X_i\right)$ is uniformly distributed. We thus define a copula in probabilistic terms as

Definition 3.1 (Copula)**.** *A function $C : [0,1]^d \to [0,1]$ is called a copula if it has uniform marginals and is a distribution function for a d-dimensional random vector $\mathbf{X}$.*

An important and fundamental theorem of copulas for especially continuous random variables where the marginals are also continuous functions is stated by Sklar:

Theorem 3.2 (Sklar's theorem)**.** *For a random vector $\boldsymbol{X}$ with CDF F and univariate marginal CDFs $F_1, \ldots, F_d$. There exists a copula C such that*

$$F(x_1, \ldots, x_d) = C(F_1(x_1), \ldots, F_d(x_d)) \tag{3.7}$$

If X is continuous, C is unique; otherwise C is uniquely determined on the Cartesian product of the ranges of distribution functions F_i, $\prod Ran\left(F_i\right)$.

Note that the last statement for non-continuous random variables can be made unique by instead using subcopulas, a generalization of copulas with domain I only a subdomain of the unit hypercube $\mathbb{I}^d = [0,1]^d$ containing all faces of the unit hyper cube. However, there are infinitely many ways of extending such a subcopula to a copula C[13]. In our case, this means that for discrete and/or mixed variables, we will later have to work around this non-uniqueness when calculating mutual information. The example made by Geenens[13] is a bivariate random vector of independent variables $X \sim \text{Bern}\left(\pi_X\right)$ and $Y \sim \text{Bern}\left(\pi_Y\right)$. The support of F_X and F_Y is then $\{0, 1-\pi_X\}$ and $\{0, 1-\pi_Y\}$ respectively. Due to the restriction on the boundary of the unit square, the only unique point of a copula C is then $(1-\pi_X, 1-\pi_Y)$, and by independence we must have

$$C\left(1-\pi_X, 1-\pi_Y\right) = (1-\pi_X)(1-\pi_Y)$$

Geenens then proceed to define an uncountable set of copulas that fulfill the above criterion which further illustrates that the basic concepts of copulas are not well suited for discrete random vectors. Note that in the article it is however argued how one can extend the concept to a more general concept that works for mixed variables.

From Equation 3.7 we see that a copula is thus simply just a function that *couples* the marginals of a random vector to the joint distribution. The following corollary follows immediately

Corollary 3.2.1 (Coordinate transformation)**.** *Under the assumptions of Theorem 3.2, given any set* $(T_1, \dots, T_d)$ *of strictly increasing functions, if* C *is a copula of* $(X_1, \dots, X_d)$ *then it is also a copula of* $(T_1(X_1), \dots, T_d(X_d))$.

Proof. Suppose $(X_1, \dots, X_d)$ admits a copula C and let T_i be given as stated. Consider coordinate wise the result of the transformation $Y_i = T_i(X_i)$ and consider the CDF $F_{Y_i}(y_i)$

$$
\begin{aligned}
F_{Y_i}(y_i) &= \mathbb{P}\left(Y_i \leq y_i\right) \\
&= \mathbb{P}\left(T_i^{-1}(Y_i) \leq T_i^{-1}(y_i)\right) \\
&= \mathbb{P}\left(X_i \leq T_i^{-1}\left(y_i\right)\right) \\
&= F_{X_i}\left(T_i^{-1}\left(y_i\right)\right)
\end{aligned}
$$

The above is easily generalized for a joint distribution as well. Thus, by the existence of a copula C for $\boldsymbol{X}$

$$
\begin{aligned}
F_{\boldsymbol{Y}}(y_1, \dots, y_d) &= F_{\boldsymbol{X}}\left(T_1^{-1}\left(y_1\right), \dots, T_d^{-1}\left(y_d\right)\right) \\
&= C\left(F_{X_1}(T_1^{-1}\left(y_1\right)), \dots, F_{X_d}(T_d^{-1}\left(y_d\right))\right) \\
&= C\left(F_{Y_1}(y_1), \dots, F_{Y_d}(y_d)\right)
\end{aligned}
$$

where Sklar's theorem have been used for the second equality. The above shows that C is indeed also a copula for $\boldsymbol{Y} = (T_1(X_1), \dots, T_d(X_d))$. □

The above corollary is actually equivalent with a seemingly stronger statement and follows easily

Proposition 3.3. *Since* T_i *is strictly increasing, the inverse* T_i^{-1} *exists and is also strictly increasing. Thus, the above implication is bidirectional and hence for strictly increasing functions* T_i, C *is a copula of* $(X_1, \dots, X_d)$ *if and only if it is a copula of* $(T_1(X_1), \dots, T_d(X_d))$.

3.2.2 Mutual Information and Copula Entropy

In this section we introduce copula entropy as done in [7] and see how it actually is equal to the well known mutual information (multiplied by -1) and hence as a corollary that mutual information is independent of marginals. Namely, under a coordinate transformation as in Corollary 3.2.1, the mutual information is constant. The name comes from the general definition of (differential) entropy as we shall see shortly. However, first we define mutual information between a set of random variables

Definition 3.4 (Mutual information)**.** *For a random vector $\boldsymbol{X} = \{X_i\}$, we define the mutual information as*

$$I(\boldsymbol{X}) = \mathbb{E}\left[\log_b \left(\frac{f(\boldsymbol{X})}{\prod_i f_i(X_i)}\right)\right]$$

where f is the joint density function with marginals f_i of the random vector $\boldsymbol{X}$. The base of the logarithm b is often chosen to be 2, e or 10 although the choice is unimportant as all logarithms are equivalent up to a scaling factor. We shall in general choose $b = e$ and drop the base b from this point on.

We note that later on, as the choice of b will result in a scaling of G_{obs}, but we will also introduce a scaling parameter α for G_{obs} to both ensure the convergence of the algorithm and to control higher order effects, we shall in general choose $b = e$.

An important property of mutual information is that the continuous version is the limit of the discrete mutual information for random (continuous) vector discretized as the mesh size goes to zero i.e. recovering the continuity of the random vector. This is discussed in Subsection 3.2.3. For now, we proceed with the definition of (joint) entropy for both discrete and continuous random vectors.

Definition 3.5 (Entropy)**.** *The (joint) entropy of a random vector $\boldsymbol{X}$ is defined as*

$$H(\boldsymbol{X}) = -\mathbb{E}\left[\log f(\boldsymbol{X})\right]$$

In case of a discrete random vector, this is called the Shannon entropy while for continuous random vectors, this is called differential entropy and is often denoted as $h(\boldsymbol{X})$ instead of $H(\boldsymbol{X})$.

We note the need for two separate notations of entropy as differential entropy is not the limit of Shannon entropy in the way mutual information is. Again, this is further discussed in Subsection 3.2.3.

Before discussing copula entropy (CE), we note a very useful relation between entropy and mutual information. Indeed, we shall later use this to show that mutual information in the continuous version is the limit of the discretization.

Lemma 3.6 (Mutual information and entropy relation). *For a continuous random vector $\boldsymbol{X}$, the (joint) mutual information $I(\boldsymbol{X})$ can be decomposed into a sum of differential entropies as*

$$I(\boldsymbol{X}) = \sum_{i=1}^{d} h_i(X_i) - h(\boldsymbol{X})$$

where d is the dimension of $\boldsymbol{X}$. The same is true for discrete variables but with entropy H instead of differential entropy h.

Proof. This follows immediately from the definition of mutual information and entropy:

$$\mathbb{E}\left[\log \frac{f(\boldsymbol{X})}{\prod_{i=1}^{d} f_i(X_i)}\right] = -\sum_{i=1}^{d} \mathbb{E}\left[\log f_i(X_i)\right] + \mathbb{E}\left[\log f(\boldsymbol{X})\right]$$

□

With the definitions of mutual information and entropy we are finally ready to introduce copula entropy.

Definition 3.7 (Copula entropy). *For a continuous random vector $\boldsymbol{X}$ with a uniquely defined copula C, and copula density c, we define the copula entropy CE of $\boldsymbol{X}$ as*

$$CE(\boldsymbol{X}) = h(\boldsymbol{U})$$

where $\boldsymbol{U}$ has density c. In particular,

$$CE(\boldsymbol{X}) = -\mathbb{E}\left[\log c(\boldsymbol{U})\right]$$

As stated above, copula entropy is actually equal to the negative mutual information which we state as a theorem

Theorem 3.8 (Equality of Copula entropy). *For a continuous random vector $\boldsymbol{X}$, the copula entropy CE is equal to the negative joint mutual information of $\boldsymbol{X}$*

$$CE(\boldsymbol{X}) = -I(\boldsymbol{X})$$

Proof. By Theorem 3.2, letting $x_i = F_i^{-1}(u_i)$, we can relate the copula density to the joint density of $\boldsymbol{X}$ and its marginals

$$\begin{aligned}
c(u_1, \dots, u_d) &= \frac{\partial}{\partial \mathbf{u}} C(u_1, \dots, u_d) \\
&= \frac{\partial}{\partial \mathbf{u}} F\left(F_1^{-1}(u_1), \dots, F_d^{-1}(u_d)\right) \\
&= \frac{\partial^d}{\prod_{i=1}^d \partial u_i} F\left(F_1^{-1}(u_1), \dots, F_d^{-1}(u_d)\right) \\
&= \frac{\partial^{d-1}}{\prod_{i=2}^d \partial u_i} \left(\frac{\partial}{\partial x_1} F\right)\left(F_1^{-1}(u_1), \dots, F_d^{-1}(u_d)\right) \cdot \frac{1}{f_1\left(F_1^{-1}(u_1)\right)} \\
&\;\vdots \\
&= \frac{\partial}{\partial \boldsymbol{x}} F\left(F_1^{-1}(u_1), \dots, F_d^{-1}(u_d)\right) \cdot \frac{1}{\prod_{i=1}^d f_i\left(F_i^{-1}(u_i)\right)} \\
&= f\left(F_1^{-1}(u_1), \dots, F_d^{-1}(u_d)\right) \frac{1}{\prod_{i=1}^d f_i\left(F_i^{-1}(u_i)\right)}
\end{aligned}$$

Where for the fourth equality, we have applied the chain rule and that $\frac{d}{dx} f^{-1}(x) = \frac{1}{f'(f^{-1}(x))}$ for any differentiable and invertible function f. The final equality follows by definition of the joint probability density function f in terms of F. We note that there is no need for the Jacobian as $x_i = F_i^{-1}(u_i)$ and hence no need for non-diagonal partial derivatives $\frac{\partial x_i}{\partial u_j}$. It follows directly that

$$\begin{aligned}
-CE(\boldsymbol{X}) &= \int_{[0,1]^d} c(\boldsymbol{u}) \log c(\boldsymbol{u})\ d\boldsymbol{u} \\
&= \int_{\mathcal{X}} \frac{f(\boldsymbol{x})}{\prod_{i=1}^d f_i(x_i)} \log\left(\frac{f(\boldsymbol{x})}{\prod_{i=1}^d f_i(x_i)}\right) \prod_{i=1}^d f_i(x_i)\ d\boldsymbol{x} \\
&= \int_{\mathcal{X}} f(\boldsymbol{x}) \log\left(\frac{f(\boldsymbol{x})}{\prod_{i=1}^d f_i(x_i)}\right) d\boldsymbol{x} \\
&= I(\boldsymbol{X})
\end{aligned}$$

where $\mathcal{X} = \prod_{i=1}^d \operatorname{dom} F_i$ is the domain of the random vector $\boldsymbol{X}$ and the third equality follows from a change of variables with the trivial substitution $u_i = F_i(x_i)$ such that $du_i = f_i(x_i)\, dx_i$ and $x_i = F_i^{-1}(u_i)$. This concludes the proof. □

Finally, before moving on to correlation as a measure of similarity, we discuss what happens in the limit of mutual information and entropy as we shall later need this as arguments for numerical stability.

3.2.3 Entropy and Mutual Information in the Limit

In this section, we shall discuss the differences between entropy and differential entropy and observe how this difference cancels when computing mutual information. In fact, we shall see that mutual information defined for continuous random vectors is the limit of the discrete version which will be useful later when implementing the algorithm.

First, although one may think differential entropy is the limit of (discrete) entropy, this is not the case. Namely, consider the support of $f(x)$ (here assumed to be the entire real line) binned into intervals i.e. a discretization of the continuous random variable X, which we shall denote X^Δ. To make notation simpler, we shall bin into equal-sized intervals of width Δ. Then, for each interval $[i\Delta, (i+1)\Delta)$ for $i \in \mathbb{Z}$, there exists an x_i such that the probability mass on this interval is represented by this x_i:

$$\mathbb{P}\left(X^\Delta = x_i\right) = f(x_i)\Delta = \int_{i\Delta}^{(i+1)\Delta} f(x)\,dx \tag{3.8}$$

Clearly, this discretization generates a valid distribution as

$$\sum_{i\in\mathbb{Z}} f(x_i)\Delta = \int_{\mathbb{R}} f(x)\,dx = 1$$

and in the limit, as $\Delta \to 0$ we recover the original distribution $f(x)$. However, if we try to calculate the entropy of this discretization, denoted by H^Δ, we get a diverging limit

$$\begin{aligned} H^\Delta &= -\sum_{i\in\mathbb{Z}} f(x_i)\Delta \log\left(f(x_i)\Delta\right) \\ &= -\sum_{i\in\mathbb{Z}} f(x_i)\Delta \log f(x_i) - \sum_{i\in\mathbb{Z}} f(x_i)\Delta \log \Delta \\ &= -\sum_{i\in\mathbb{Z}} f(x_i)\Delta \log f(x_i) - \log \Delta \end{aligned}$$

Clearly, the first term in the above expression converges to the differential entropy $h(X)$ as $\Delta \to 0$ whereas $\log \Delta \to -\infty$ i.e. the expression diverges altogether when differential entropy is well-defined.

A similar argument for the joint entropy between the discretization of X_1 and X_2 (and in principle to any number of dimensions), denoted by H_{12}^Δ, results in

$$H_{12}^\Delta = -\sum_{i,j\in\mathbb{Z}} f\left(x_1^{(i)}, x_2^{(j)}\right) \Delta_1\Delta_2 \log f\left(x_1^{(i)}, x_2^{(j)}\right) - \log\Delta_1 - \log\Delta_2$$

where $x_1^{(i)} \in [i\Delta_1, (i+1)\Delta_1)$ and $x_2^{(j)} \in [j\Delta_2, (j+1)\Delta_2)$ are defined such that

$$f\left(x_1^{(i)}, x_2^{(j)}\right)\Delta_1\Delta_2 = \int_{j\Delta_2}^{(j+1)\Delta_2}\int_{i\Delta_1}^{(i+1)\Delta_1} f(x_1,x_2)\,dx_1dx_2, \quad \forall i,j \in \mathbb{Z}$$

Note that clearly $\left(x_1^{(i)}, x_2^{(j)}\right)$ exists for all $i, j \in \mathbb{Z}$. Again, the joint entropy diverges however, when computing the mutual information, we see that the diverging terms cancel. Namely, from Lemma 3.6

$$\begin{aligned}
I_{12}^{\Delta} &= H_1^{\Delta} + H_2^{\Delta} - H_{12}^{\Delta} \\
&= -\sum_{i\in\mathbb{Z}} f_1\left(\tilde{x}_1^{(i)}\right)\Delta_1 \log f_1\left(\tilde{x}_1^{(i)}\right) - \log\Delta_1 \\
&\quad - \sum_{j\in\mathbb{Z}} f_2\left(\tilde{x}_2^{(j)}\right)\Delta_2 \log f_2\left(\tilde{x}_2^{(j)}\right) - \log\Delta_2 \\
&\quad + \sum_{i,j\in\mathbb{Z}} f\left(x_1^{(i)}, x_2^{(j)}\right)\Delta_1\Delta_2 \log f\left(x_1^{(i)}, x_2^{(j)}\right) + \log\Delta_1\Delta_2 \\
&= -\sum_{i\in\mathbb{Z}} f_1\left(\tilde{x}_1^{(i)}\right) \log f_1\left(\tilde{x}_1^{(i)}\right)\Delta_1 - \sum_{j\in\mathbb{Z}} f_2\left(\tilde{x}_2^{(j)}\right) \log f_2\left(\tilde{x}_2^{(j)}\right)\Delta_2 \\
&\quad + \sum_{i,j\in\mathbb{Z}} f\left(x_1^{(i)}, x_2^{(j)}\right) \log f\left(x_1^{(i)}, x_2^{(j)}\right)\Delta_1\Delta_2 \\
&\to h(X_1) + h(X_2) - h(X_1,X_2) \text{ as } \Delta_1, \Delta_2 \to 0
\end{aligned}$$

Thus, the limit of the mutual information for discrete random variables is indeed the mutual information defined for continuous random variables and can be computed either as the limit of discretizing the probability density function and then computing entropies or just using the initial definition for (discrete) mutual information in Definition 3.4. In particular, mutual information for discrete and random variables are comparable such that it makes sense define mutual information between mixed random variables. For a more rigorous treatment of this, we refer to [14] where they define mutual information between discrete and continuous random variables from a measure theoretical point of view.

Before continuing, we discuss the case where X_1 is equal to X_2. In this case, discretizing with a common Δ we have that

$$f\left(x_1^{(i)}, x_2^{(j)}\right)\Delta^2 = \int_{j\Delta}^{(j+1)\Delta}\int_{i\Delta}^{(i+1)\Delta} f(x_1,x_2)\,dx_1dx_2, \quad \forall i,j \in \mathbb{Z}$$

Clearly, the above integral is 0 for $i \neq j$. Although $f(x_1, x_2)$ is not well-defined in the usual functional sense, extending to distribution, we might write $f(x_1, x_2) = f(x_2|x_1)f(x_1)$. In terms of distributions, it works to put $f(x_2|x_1) = \delta(x_2 - x_1)$ where δ is the *Dirac delta* distribution, as then $\int_{\mathbb{R}} f(x_1, x_2)\,dx_2 = f(x_1)$ and

$f(x_1, x_2)$ is "0" when $x_1 \neq x_2$. I.e. the marginals and probability mass are correct. Then, when calculating the above integral, we get that

$$\begin{aligned} f\left(x_1^{(i)}, x_1^{(i)}\right)\Delta^2 &= \int_{i\Delta}^{(i+1)\Delta}\int_{i\Delta}^{(i+1)\Delta} f(x_1, x_2)\, dx_1 dx_2 \\ &= \int_{i\Delta}^{(i+1)\Delta} f\left(x_1\right)\, dx_1 \\ &= f\left(\tilde{x}_1^{(i)}\right)\Delta \end{aligned}$$

Thus, when calculating I^Δ for two identical variables, we obtain

$$\begin{aligned} I^\Delta = &-\sum_{i\in\mathbb{Z}} f\left(\tilde{x}_1^{(i)}\right)\log f\left(\tilde{x}_1^{(i)}\right)\Delta - \sum_{j\in\mathbb{Z}} f\left(\tilde{x}_2^{(j)}\right)\log f\left(\tilde{x}_2^{(j)}\right)\Delta \\ &+\sum_{i\in\mathbb{Z}} f\left(\tilde{x}_1^{(i)}\right)\log f\left(\tilde{x}_1^{(i)}\right)\Delta - \log\Delta \\ &\to \infty \text{ as } \Delta \to 0 \end{aligned}$$

Thus in practice, it would not make much sense to compare equal variables or even a random vector only defined on a lower dimensional manifold as we would get an infinite copula entropy.

3.2.4 Correlation

At this point, we have a good understanding of copula entropy/mutual information for calculations later on. However, another typical measure of similarity is correlation which is easily estimated from sample data. However, in this section we show that in general, we can not compute the correlation coefficient from a copula which we saw above is the case for mutual information. Namely, given a copula C for some set of random variables $\{X_i\}_{i\in I}$ indexed by finite I, one can not calculate ρ between any pair (X_i, X_j), $i \neq j$ from the copula. This is easily shown by the following argument.

First, note that from Corollary 3.2.1, C is also a copula for $Z_i := (X_i - \mu_i)/\sigma_i$ for $i \in I$ where $\mu_i = \mathbb{E}[X_i]$ and $\sigma_i = \sqrt{\operatorname{Var} X_i}$ (assuming that these exist). Clearly, the correlation coefficient for Z_i and Z_j is the same as between X_i and X_j. We thus proceed trying to calculate the correlation between any pair Z_i and Z_j.

$$\begin{aligned} \rho_{ij} &= \int\int_{\mathbb{R}^2} z_i z_j f_{ij}(z_i, z_j)\, dz_i\, dz_j \\ &= \int\int_{[0,1]^2} F_i^{-1}(u_i)\, F_j^{-1}(u_j)\, c_{ij}(u_i, u_j)\, du_i\, du_j \end{aligned} \tag{3.9}$$

where c_{ij} density version of the copula defined for X_i and X_j and F_i and F_j are the marginals of Z_i and Z_j with mean 0 and variance 1. From the above, it is then clear for a fixed, non-constant copula C, the correlation depends on the marginals of X_i and X_j. Also, we see that a constant copula density (only admissible if $c \equiv 1$ on $[0,1]^2$ and 0 elsewhere) always results in $\rho_{ij} = 0$ as

$$\int_0^1 F^{-1}(u)\, du = \int_{\mathbb{R}} z f(z)\, dz = 0$$

where the final equality follows from the construction of Z_i.

Thus, we conclude that indeed mutual information and correlation are very different measures of codependency. Namely, mutual information does not depend on the marginal distributions whereas from Equation 3.9 we see that correlation does. Thus, it does not make much sense to introduce copulas in the setting of correlation albeit at this point we do not favor one measure above the other. Only if marginals should be insignificant to the network, copula entropy is at this point preferred.

3.3 Copula Based Network Discovery

In this section, we will present the general algorithm and discuss some of its properties regarding uncertainty and convergence. We will focus on using mutual information i.e. copula entropy as the measure of similarity but other measures such as correlation can be interchanged at will in the general algorithm.

By Theorem 3.8 we can compute the mutual information from observed data from the copula. Namely, let CE_{ij} denote the (pairwise) copula entropy of variables X_i and X_j. We shall then set

$$G_{obs} = \begin{bmatrix} 0 & -CE_{12} & \dots & -CE_{1n} \\ -CE_{21} & 0 & \dots & -CE_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -CE_{n1} & -CE_{n2} & \dots & 0 \end{bmatrix} \tag{3.10}$$

where n is the number of nodes in the graph i.e. random variables that we have observed. Notice that we have chosen the diagonal elements as 0 since information between a random variable X and itself is not really well-defined and when trying to compute this numerically, we observe diverging results as also discussed in the previous section. Furthermore, only the information that propagates through the network is of interest and so setting 0 in the diagonal avoids a bias when deconvolving the information or any similarity in general.

Especially for mutual information where the information between a variable and itself diverges to ∞ thus in the limit, from Equation 3.6, we would get the identity matrix which does not tell us much about the direct dependencies.

Algorithm 1 then follows immediately from Equation 3.10

Algorithm 1 G_{obs} computation

Require: $n > 0$ ▷ Number of variables
  $G_{obs} \leftarrow \mathbf{0}$
  for $1 \leq i, j \leq n \mid i \neq j$ **do**
    Estimate F_i and F_j from $x_i^{\mathcal{D}}$ and $x_j^{\mathcal{D}}$
    $u_i^{\mathcal{D}} \leftarrow F_i(x_i^{\mathcal{D}})$
    $u_j^{\mathcal{D}} \leftarrow F_j(x_j^{\mathcal{D}})$
    Estimate c_{ij} from $u_i^{\mathcal{D}}$ and $u_j^{\mathcal{D}}$
    Compute NCE_{ij}
    $[G_{obs}]_{ij} \leftarrow -NCE_{ij}$
  end for
  return G_{obs}

Namely, for each entry in G_{obs}, except for the diagonal elements, first estimate the cumulative distributions of X_i and X_j based on samples $x_i^{\mathcal{D}}$. Then, transform the samples by the estimated distribution function to obtain corresponding uniform samples. This may be done outside the loop to increase computational efficiency. From the paired samples $(x_i^{\mathcal{D}}, x_j^{\mathcal{D}})$, estimate the copula density c_{ij} and finally use this to compute the mutual information/copula entropy. Methods for estimating the densities and in continuation hereof the distribution functions are presented in Section 3.4. The negative copula entropy is then recorded in (i, j) entry of G_{obs}. We note that the algorithm can be optimized for symmetric measures such as copula entropy itself, to only loop through $i < j$ and saving the computed entropy in the (j, i) entry as well. Also, as copula entropy diverges as X_i and X_j are jointly distributed closer to a one-dimensional manifold, ideally there should be a check for such or the user should check the paired observations to exclude such variable combinations.

From Subsection 3.2.3, to calculate the (joint) copula entropy of a continuous random vector, we simply discretize the domain of each random variable and use the estimated copula density evaluated at these points to estimate the total copula entropy. Furthermore, if one or more elements of the random vector are mixed random variables, we choose the discrete events to be their own bins and discretize the rest or in the context of Algorithm 1 only estimate the distribution functions for the continuous component of the random variable. This works due to the copula entropy for continuous random variables being the limit of the discretization and as such, the copula entropy is well-defined for mixed random variables as well.

We continue with an example of how this discretization of a mixed random variable would work. Notice that we only have a discrete event (an atom) at 0 as this resembles the observed behavior of the delays, although the example could be extended to more complex discrete distributions.

Example 3.1 (Discrization of mixed random variable)**.** *Let X be a mixture of an atom in e.g. 0 and an exponential with parameter λ with proportions p and $1-p$. Then, a discretization of X is 0 with probability mass p and the remaining support $(0,\infty)$ discretized in some way with total probability mass $1-p$ and each bin having probability according to Equation 3.8 scaled with $1-p$. If the bin size is a constant Δ, then for the discretized variable X^{Δ}, we have $\mathbb{P}\left(X^{\Delta}=0\right)=p$ and $\mathbb{P}\left(X^{\Delta}=x_i\right)=(1-p)\exp\left(-\lambda i\Delta\right)\left(1-\exp(-\lambda\Delta)\right)$, where x_i is given by*

$$x_i = i\Delta + \frac{1}{\lambda}\left(\log\left(\lambda\Delta\right)-\log\left(1-e^{-\lambda\Delta}\right)\right), \quad i\in\mathbb{N}_0$$

3.3.1 Network Deconvolution

At this point, we have obtained a convolved matrix of information G_{obs} and are ready to use Equation 3.6. We present the original algorithm from [6] in the case G_{obs} is symmetric and hence diagonalizable by an orthogonal matrix U. The original `Matlab` implementation was translated to `Python` and is summarized in the following pseudocode.

Algorithm 2 (ND) Network Deconvolution

Require: G_{obs}, α, β
  $[G_{obs}]_{ii} \leftarrow 0,\ \forall i\in\{1,\dots,d\}$ ▷ ensure zero-diagonal
  $[G_{obs}]_{ij} \leftarrow 0$, when $[G_{obs}]_{ij} < Q_\alpha\left(G_{obs}\right)$
  Compute eigendecomposition U,Λ of G_{obs}
  $\lambda^+ \leftarrow \max\left(\lambda^{\max},0\right)$
  $\lambda^- \leftarrow \min\left(\lambda^{\min},0\right)$
  $k^+ \leftarrow \frac{1-\beta}{\beta}\lambda^+$
  $k^- \leftarrow \frac{1+\beta}{\beta}\lambda^-$
  $c_s^{-1} \leftarrow \max\left(k^+,-k^-\right)$
  $\hat{\Lambda} \leftarrow \Lambda\left(c_s^{-1}I+\Lambda\right)^{-1}$
  return $U\hat{\Lambda}U^T$

where $Q_\alpha\left(G_{obs}\right)$ denotes the α quantile of the strictly upper (or lower due to symmetry) triangular part of G_{obs}. We note the two extra parameters *alpha* and β which we will discuss shortly. In particular, the paper contains conflicting information on how to find β from how it is defined. Furthermore, they include some analysis on the robustness of the above deconvolution algorithm but only in a somewhat particular case and with some confusion on matrix norms and spectral radius. This analysis on robustness, we will extend and clarify in the following Subsection 3.3.3.

From the definition of $Q_\alpha(G_{obs})$ it is clear that the α parameter is a filter on the observed edges and is useful if one wants to filter out insignificant observations. However, in practice, as we will see, it is often not very influential except for large α (corresponding to many edges set to 0) as small perturbations from e.g. imperfect calculations should not influence the results for fairly conditioned matrices as we shall observe in Section 4.3. Thus, setting $\alpha = 0$ retains all values in G_{obs} after setting the diagonal equal to 0. As a technical detail, we note that the `quantile` function from `NumPy` (v. 1.26.4) has been used to find this quantile as quantiles can be defined in many ways from a data set.

Finally, we note that the $\beta \in (0, 1)$ parameter corresponds to a scaling of G_{obs} such that the resulting spectral norm of G_{dir} is β. From Algorithm 2 it is seen that it serves as a regularization on the eigenvalues of G_{obs} and although this is discussed in [6], their results do not conform with their implementation, and we thus comment on this and what else could be done to ensure convergence of the algorithm in the following section. Also, in practice we choose a threshold t on the elements of G_{dir} returned from Algorithm 2 to further filter out insignificant direct dependencies.

3.3.2 Ensuring Convergence and the Effect of β

In this section we will further discuss the effect of β and how the steps for rescaling the observed similarity matrix G_{obs} are derived. In particular, we will reformulate the original derivation from [6] as there is a discrepancy between their code[1] and their proof of choosing a scaling parameter c_s of G_{obs}. Namely, denote $\tilde{G}_{obs}$ as the rescaled G_{obs} such that $\tilde{G}_{obs} = c_s G_{obs}$. Choosing c_s as in Algorithm 2 i.e. $c_s^{-1} = \max\left(\frac{1-\beta}{\beta}\lambda^+, -\frac{1+\beta}{\beta}\lambda^-\right)$ where λ^+ is the largest positive eigenvalue of G_{obs} (and 0 if no eigenvalue is positive) and λ^- is the most negative eigenvalue of G_{obs} (and 0 if no eigenvalue is negative) then implies $\tilde{G}_{dir}$ obtained from the new $\tilde{G}_{obs}$ has spectral radius $\beta < 1$ i.e. a proper G_{dir} with the largest numerical eigenvalue equal to β. This holds in general and not only for symmetric G_{obs} as we will see in the following. However, when G_{obs} is symmetric

[1] https://compbio.mit.edu/nd/

the resulting $\tilde{G}_{dir}$ can easily be expressed through the eigendecomposition of G_{obs}, U, Λ as

$$\begin{aligned}\tilde{G}_{dir} &= \tilde{G}_{obs}\left(I + \tilde{G}_{obs}\right)^{-1} \\ &= c_s G_{obs}\left(I + c_s G_{obs}\right)^{-1} \\ &= U c_s \Lambda U^T \left(UU^T + U c_s \Lambda U^T\right)^{-1} \\ &= U c_s \Lambda U^T U \left(I + c_s \Lambda\right)^{-1} U^T \\ &= U\Lambda \left(c_s^{-1} I + \Lambda\right)^{-1} U^T\end{aligned}$$

which can also be seen in Algorithm 2. Thus, with everything else explained about the algorithm, we show that the resulting $\tilde{G}_{dir}$ in general have spectral radius β.

Let (λ, v) be an eigenpair of G_{obs} with $\lambda \neq 0$, it then follows that $\left(\frac{\lambda}{c_s^{-1}+\lambda}, v\right)$ is an eigenpair of $\tilde{G}_{dir}$. Then, following the arguments in [6] (which we have redone to know why the original implementation and derivation differs), we obtain that for a λ in $[0, \infty)$, we must have that

$$c_s^{-1} \geq \frac{1-\beta}{\beta}\lambda^+$$

and similarly for $\lambda \in (-c_s^{-1}, 0)$

$$c_s^{-1} \geq -\frac{1+\beta}{\beta}\lambda^-$$

for $\lambda < -c_s^{-1}$ we obtain that the resulting eigenvalue is larger than 1 hence we must also have that $c_s^{-1} \geq -\lambda^-$ which is covered by the above constraint on c_S^{-1}. Thus, the smallest c_s^{-1} we can choose to ensure that $\rho\left(\tilde{G}_{dir}\right) \leq \beta$ is by $c_s^{-1} = \max\left(\frac{1-\beta}{\beta}\lambda^+, -\frac{1+\beta}{\beta}\lambda^-\right)$ which also implies that $\rho\left(\tilde{G}_{dir}\right) = \beta$ as either the most negative or most positive eigenvalue is mapped to β or $-\beta$ respectively. This coincides with the original implementation, noting that some error has been made in the original discussion of the parameter β in [6]. Furthermore, we note that if we just want the algorithm to converge, as we discussed before, this is equivalent to $\sigma\left(\tilde{G}_{obs}\right) \subseteq (-1/2, \infty)$, so really, we can just choose $c_s^{-1} = -(2+\delta)\lambda^-$ for some small δ if $\lambda^- < -1/2$ and otherwise not scale G_{obs} to preserve the structure. Finally, we note that as β tends to 0, higher order interactions become less significant as can clearly be seen from Equation 3.4. Thus, β also allows us to tune how much influence higher order interactions should have and one should try different β to see how influenced results are to higher order effects.

3.3.3 Robustness to Noise

Finally, before discussing how to compute and estimate the mutual information between two random variables based on observations, we turn our heads to error analysis of the deconvolution algorithm. It is important to understand how well the algorithm performs subject to noise and errors. Namely, in the case of mutual information, the assumption that higher order effects can be calculated as a sum of matrix powers of the direct effects does not hold. Thus, if we can quantify the error in G_{obs}, we can from the following analysis quantify the resulting error in G_{dir}. We shall first discuss the original result from [6], correcting some errors in terms of definitions and see how their result can also be expressed as an absolute upper bound on the error instead of only how this error behaves for small perturbations. Furthermore, we shall extend their result to not only hold when $\rho\left(G_{obs}\right) < 1$ and $\rho\left(G_{obs} + N\right) < 1$ where N is some noise e.g. from computation or assumptions that does not completely hold.

The original result states that $\left|\left|G_{dir} - \tilde{G}_{dir}\right|\right|_2 \leq \gamma + \mathcal{O}\left(\delta^2 + \gamma^2 + \delta\gamma\right)$ where $||\cdot||_2$ is the Euclidean norm also known as the spectral norm as this is equal to the largest singular value of the input matrix. However, they note that the Euclidean norm of a matrix M is equal to $\sqrt{\sum_{i,j} m_{ij}^2}$ which is incorrect. This is the Frobenius norm, and instead it should have been defined as

$$||M||_2 = \sup_{||x||_2=1} ||Mx||_2 = \sigma_{\max}\left(M\right)$$

They then proceed to let γ be the largest absolute eigenvalue of N and δ the largest absolute eigenvalue of $\tilde{G}_{obs} = G_{obs} + N$ however as the noise may be both positive and negative, it is easier to define δ as the largest absolute eigenvalue of G_{obs} instead which we will do in the following. We note that γ and δ are not the spectral/Euclidean norm of N and G_{obs} respectively as in general, we only have $\rho\left(M\right) \leq ||M||_2$. However, if G_{obs} and N are both (real) symmetric matrices, then the spectral norms are equal to the largest absolute eigenvalues of G_{obs} and N respectively. Thus, if instead one wanted to measure the difference in the direct dependency matrices in terms of e.g. the Frobenius norm, it is important to differentiate between the spectral radius and the norm that is actually being used. Finally, before constructing the actual upper bound on the error instead of quantizing the asymptotic behavior for small γ, we note that $||\cdot||_2$ is a sub-multiplicative matrix norm defined as bellow ([15]), and that we shall assume that $\rho\left(G_{obs}\right), \rho\left(\tilde{G}_{obs}\right) < 1$.

Definition 3.9 (Sub-multiplicative Matrix norm)**.** *A matrix norm $|||\cdot|||$ is said to be sub-multiplicative, if for every $A, B \in \mathbb{F}^{n\times n}$ where $\mathbb{F}$ is either the real or complex field:*

$$|||AB||| \leq |||A||| \cdot |||B|||$$

As we do not use any property of the spectral norm except that it is sub-multiplicative, we shall consider any norm $||\cdot||$ in general that is also sub-multiplicative. Thus, consider the norm of the difference $G_{dir} - \tilde{G}_{dir}$:

$$
\begin{aligned}
\left|\left|G_{dir} - \hat{G}_{dir}\right|\right| &= \left|\left|G_{obs}\left(I + G_{obs}\right)^{-1} - \hat{G}_{obs}\left(I + \hat{G}_{obs}\right)^{-1}\right|\right| \\
&= \left|\left|-\sum_{k\geq 1}\left(-G_{obs}\right)^{k} + \sum_{k\geq 1}\left(-\hat{G}_{obs}\right)^{k}\right|\right| \\
&\leq \sum_{k\geq 1}\left|\left|G_{obs}^{k} - \left(G_{obs} + N\right)^{k}\right|\right| \\
&\leq \sum_{k\geq 1}\sum_{i=1}^{k}\binom{k}{i}\left|\left|N\right|\right|^{i}\left|\left|G_{obs}\right|\right|^{k-i} \\
&= \sum_{k\geq 1}\sum_{i=1}^{k}\binom{k}{i}\gamma^{i}\delta^{k-i} \\
&= \sum_{k\geq 1}\left(\left(\gamma+\delta\right)^{k} - \delta^{k}\right) \\
&= \frac{\gamma+\delta}{1-\gamma-\delta} - \frac{\delta}{1-\delta} \\
&= \frac{\gamma}{(1-\gamma-\delta)(1-\delta)}
\end{aligned}
\tag{3.11}
$$

where in the second to last inequality, we assume that $\gamma + \delta < 1$ as then both $\sum(\gamma+\delta)^k$ and $\sum\delta^k$ converges as $\gamma + \delta \geq \delta \geq 0$ and hence also the difference of the sums converges. Also, the second equality uses that the spectral norm of G_{obs} and $\tilde{G}_{obs}$ is less than 1 in order to express the inverses as infinite series. Thus, the above bound on the difference $G_{dir} - \tilde{G}_{dir}$ does not hold in every case. Namely, for fixed γ, the bound tends to ∞ as $\delta \to 1$. Furthermore, we note that the final infinite sum diverges whenever $\gamma + \delta > 1$ through the following argument using the ratio test for infinite sums which is needed because we can not conclude on the convergence of a difference of diverging sums solely from the fact that the individual sums diverge:

$$\begin{aligned}\lim_{n\to\infty}\left|\frac{(\gamma+\delta)^{n+1}-\delta^{n+1}}{(\gamma+\delta)^{n}-\delta^{n}}\right| &= \lim_{n\to\infty}\left|\frac{(\gamma+\delta)\left(1+\frac{\gamma}{\delta}\right)^{n}-\delta}{\left(1+\frac{\gamma}{\delta}\right)^{n}-1}\right| \\ &= \lim_{n\to\infty}\left|\delta+\gamma\frac{\left(1+\frac{\gamma}{\delta}\right)^{n}}{\left(1+\frac{\gamma}{\delta}\right)^{n}-1}\right| \\ &= \lim_{n\to\infty}\left|\delta+\gamma\frac{1}{1-\left(1+\frac{\gamma}{\delta}\right)^{-n}}\right| \\ &= |\gamma+\delta| = \gamma+\delta\end{aligned}$$

assuming that $\gamma, \delta > 0$ corresponding to neither N nor G_{obs} is the zero matrix in which case the above analysis in nonsensical.

Before continuing with a more general bound on the error, we first note that examples of sub-multiplicative matrix norms are every induced norm such as the spectral norm and the Frobenius norm which is often useful when interpreting error. Also, the max norm is *not* sub-multiplicative, but a scaled version is (which is true for any matrix norm from the fact that all matrix norms are equivalent).

Now, consider the general case, where we do not restrict the spectral radius of either G_{obs} or N except such that G_{obs} and $\tilde{G}_{obs}$ admits direct similarity matrices G_{dir} and $\tilde{G}_{dir}$ (with spectral radius less than 1). To obtain a more general result, we shall use the following result from [16], which is very useful when doing matrix perturbation analysis.

Theorem 3.10 (Inverse of sum of matrices)**.** *Let $A, B \in \mathbb{R}^{n\times n}$ such that A and $A+B$ are invertible. Then the inverse of $A+B$ can be expressed as*

$$(A+B)^{-1} = A^{-1} - A^{-1}B\,(A+B)^{-1}$$

The proof of the above is simple through direct computation. Hence, we continue to once again consider the difference $G_{dir} - \tilde{G}_{dir}$

$$\begin{aligned}G_{dir}-\tilde{G}_{dir} &= G_{obs}\,(I+G_{obs})^{-1} - (G_{obs}+N)\,(I+G_{obs}+N)^{-1} \\ &= G_{obs}\left((I+G_{obs})^{-1}-(I+G_{obs}+N)^{-1}\right) - N\,(I+G_{obs}+N)^{-1} \\ &= G_{obs}\,(I+G_{obs})^{-1}N\,(I+G_{obs}+N)^{-1} - N\,(I+G_{obs}+N)^{-1} \\ &= -\,(I+G_{obs})^{-1}N\,(I+G_{obs}+N)^{-1}\end{aligned}$$

where the third equality follows from Theorem 3.10. This way, we have a simple exact expression for the difference without any further assumptions on G_{obs} and

N. Now, under a sub-multiplicative norm $||\cdot||$ we can bound the norm of the difference in the following way.

$$\left|\left|G_{dir} - \tilde{G}_{dir}\right|\right| \leq ||N|| \left|\left|\left(I + G_{obs}\right)^{-1}\right|\right| \left|\left|\left(I + G_{obs} + N\right)^{-1}\right|\right| \tag{3.12}$$

We note that if once again, we assume that the spectral radius of G_{obs} and $G_{obs} + N$ are smaller than 1, we rediscover Equation 3.11. Equation 3.12 also shows that in general, if N is small or G_{obs} is large we should observe small errors which is also what we would expect intuitively. The above result is also very useful when later on in Section 4.3 we discuss the error from using mutual information in the case of a multi-variate Gaussian.

From Equation 3.12, by another application of Theorem 3.10, we find the relative error in general to be bounded as follows

$$\frac{\left|\left|G_{dir} - \tilde{G}_{dir}\right|\right|}{||G_{dir}||} \leq ||N|| \left|1 - \frac{||I||}{\left|\left|G_{obs}\left(I + G_{obs}\right)^{-1}\right|\right|}\right| \left|\left|\left(I + G_{obs} + N\right)^{-1}\right|\right|$$

Finally, before discussing the methods for estimating the copula density, we comment on some frequently used matrix norms and show some explicit bounds on the error only using the difference of G_{obs} and $\tilde{G}_{obs}$, N. Namely, we shall consider the max norm and Frobenius norm of the difference $G_{dir} - \tilde{G}_{dir}$ and note that from [17], we can relate the Euclidean norm to the Frobenius and max norm in the following way. Namely, for any matrix $A \in \mathbb{R}^{n \times n}$ it holds that

$$||A||_2 \leq ||A||_F \leq \sqrt{n}\,||A||_2$$
$$||A||_{\max} \leq ||A||_2 \leq n\,||A||_{\max}$$

Finally, if G_{obs} and N are symmetric, the singular values are equal to the absolute eigenvalues for G_{obs} and $\tilde{G}_{obs}$ and because $\sigma\left(I + G_{obs}\right), \sigma\left(I + \tilde{G}_{obs}\right) \subseteq (1/2, \infty)$ implies $\sigma\left(\left(I + G_{obs}\right)^{-1}\right), \sigma\left(\left(I + \tilde{G}_{obs}\right)^{-1}\right) \subseteq (0, 2)$ we infer that $\left|\left|\left(I + G_{obs}\right)^{-1}\right|\right|_2, \left|\left|\left(I + \tilde{G}_{obs}\right)^{-1}\right|\right|_2 \leq 2$. Using this with the above equivalence on the Euclidean norm with Equation 3.12, we conclude that

$$\begin{aligned} \left|\left|G_{dir} - \tilde{G}_{dir}\right|\right|_F &\leq 4\sqrt{d}\,||N||_F \\ \left|\left|G_{dir} - \tilde{G}_{dir}\right|\right|_{\max} &\leq 4d\,||N||_{\max} \end{aligned} \tag{3.13}$$

This clearly shows us that for small networks (thus small d) we risk smaller errors in terms of the Frobenius and max norm (which is not surprising) which are clearly interpreted through the difference of individual element of G_{dir} and $\tilde{G}_{dir}$ and that the max norm scales linearly with the number of nodes while the Frobenius difference only scales with the square root of the number of nodes.

3.4 Estimating Mutual Information

For Algorithm 1 to work, we need a good and preferably fast estimator of mutual information. [6] proposes to use B-splines for this based on [18] which we shall describe in the following section. It is however quickly apparent that this estimator has some problems when computing mutual information based on the Copula representation. We extend the method to other splines but end up using kernel density estimators (KDE) as they can be regularized in a continuous manner and as a result of this in general show great performance.

3.4.1 B-splines

A spline is in general a piecewise polynomial [19]. We say that a spline is of order $p+1$ if the piecewise polynomials are of order p. A particular and widely used type of splines are B-splines which are a basis for all splines such that any spline can be expressed as a linear combination of B-splines. Five B-splines of degree 3 (order 4) can be seen in Figure 3.3(a) where we denote $B_{i,p}$ as a B-spline of degree p. The index i comes from the following. Namely, let $i \in \{1, \ldots, m+p+1\}$, we define knots t_i as where pieces of polynomials meet such that

$$B_{i,p}(t) = \begin{cases} \text{non-zero}, & t \in [t_i, t_{i+p+1}) \\ 0, & \text{otherwise} \end{cases}$$

uniquely defines m splines on $[t_{p+1}, t_m]$. If we furthermore constrain the splines such that

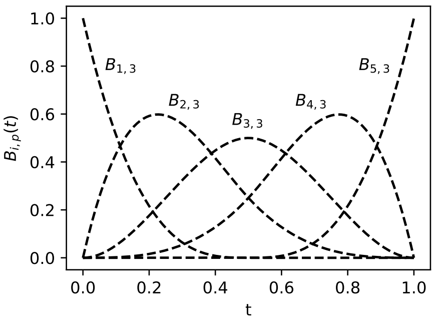
$$\sum_{i=1}^{m} B_{i,p}(t) = 1$$

The B-splines $B_{i,p}$ can then be evaluated at some t through recursion by the Cox-deBoor recursion formula:

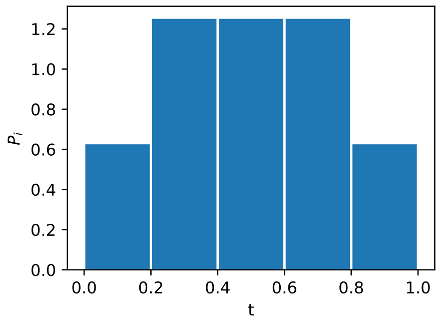
$$B_{i,0}(t) = 1, \quad t \in [t_i, T_{i+1}) \tag{3.14}$$

$$B_{i,k}(t) = \frac{t - t_i}{t_{i+k} - t_i} B_{i,k-1}(t) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} B_{i+1,k-1}(t) \tag{3.15}$$

The fact that the B-splines sum to 1 and that we have m splines of degree p



(a)



(b)

Figure 3.3

is what we will use to estimate the density and in return the mutual information between two random variables based on observations. Namely, suppose we want to compute the mutual information, this can then be done by discretizing the random variables by binning. We assign the probability mass $P_{i,j}$ to bin $(i, j) \in \{1, \ldots, m\} \times \{1, \ldots, m\}$ as the fraction of observations in the domain corresponding to that bin. In particular, using Copula entropy, we would divide the unit interval into m^2 equal bins. As we saw earlier, in theory, increasing the number of bins will result in a more and more exact estimate of the mutual information. However, with limited observations, this is not the case as in the limit, the bins would not represent the true underlying distribution due to the finite number of observations. Namely, we would observe a few bins with 1 observation and many with none. As an example, suppose we have n observations and m bins in both dimensions. Then, for m large enough, $P_{i,j} = \frac{1}{n}$ for a hundred distinct bins as well as the marginal probability masses $P_i = \frac{1}{n}$ and $P_j = \frac{1}{n}$. But then,

$$
\begin{aligned}
I\left(X_1^\Delta, X_2^\Delta\right) &= -\sum_{i=1}^{m} P_i \log P_i - \sum_{j=1}^{m} P_j \log P_j + \sum_{i,j=1}^{m} P_{i,j} \log P_{i,j} \\
&= -n\left(\frac{1}{n} \log \frac{1}{n}\right) \\
&= \log n
\end{aligned}
$$

which is clearly independent of the true underlying distribution. However, if for each observation x_j we assign it to bin i with probability mass $B_{i,p}(x_j)$ this problem is mitigated to some extent as long as m is not too large. It follows that in total each x_j is assigned to all m bins with a combined mass 1 as $\sum_i B_{i,p}(x_j) = 1$. Thus, let $\left(\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}\right)$ be a pair of random vectors each of d i.i.d variables representing observations drawn from a joint distribution, we then define the B-spline density estimator (i.e. a random variable) for $\boldsymbol{X}^{(1)}$ as

$$P_i^{(1)} = \frac{1}{d}\sum_{j=1}^{d} B_{i,p}\left(X_j^{(1)}\right), \quad i \in \{1, \ldots, m\}$$

and similarly for $\boldsymbol{X}^{(2)}$. Furthermore, the B-spline joint density estimator is given by

$$P_{i,j} = \frac{1}{d}\sum_{k=1}^{m} B_{i,p}\left(X_k^{(1)}\right) B_{j,p}\left(X_k^{(2)}\right)$$

i.e. a product of the probability masses such that $\sum_{i,j=1}^{d} P_{i,j} = 1$ and the marginal probability masses are given by $P_i^{(1)}$ and $P_j^{(2)}$ respectively.

However, there is still one problem. Namely, the marginals are not uniform when $p > 0$ when $\boldsymbol{X}^{(k)}$ is. This can also be seen from Figure 3.3(b). This is especially bad when computing Copula entropy as for e.g. a Gaussian random vector we would dramatically underestimate the mutual information as we shall also see in the next chapter. To see that this is the case, consider the expectation of $P_i^{(k)}$ for $k \in \{1, 2\}$:

$$\mathbb{E}\left[P_i^{(k)}\right] = \frac{1}{d}\sum_{j=1}^{d} \mathbb{E}\left[B_{i,p}\left(X_j^{(k)}\right)\right] = \int_0^1 B_{i,p}(x)\; dx$$

Indeed, we would want this to be $\frac{1}{m}$ but from the Cox-deBoor recursion formula, we see that this is not the case. Namely, by choosing the knots as in [18] we see that the bins close to the boundary have too little probability mass. Thus, we turn our head to another family of splines called M-splines.

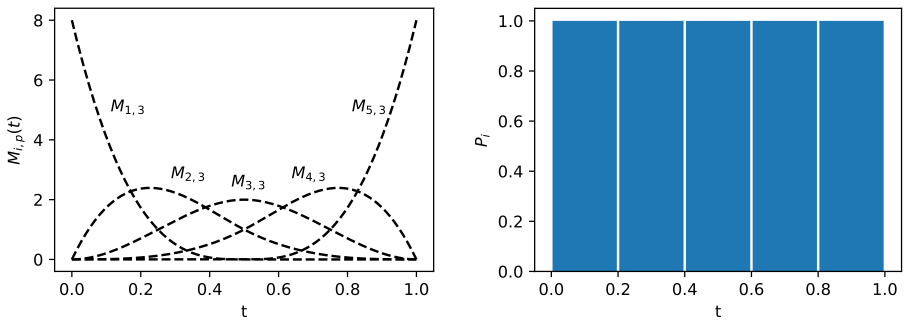
3.4.2 M-splines

Another known family of splines called M-splines have exactly the desired property of equal integrals. Namely, the M-splines $M_{i,p}$ all have unit integrals. Thus, rescaling with $\frac{1}{m}$ results in a family of splines $\tilde{M}_{i,p}$ such that on average we have that $P_i = \frac{1}{m}$. The M-splines equivalent to those of Figure 3.3 are shown in Figure 3.4. Indeed, we see that the marginals are uniform. M-splines can similarly to B-splines by computed recursively by the following [20]

$$M_{i,0}(t) = \frac{1}{t_{i+1} - t_i}, \quad t \in [t_i, t_{i+1}) \tag{3.16}$$

$$M_{i,k}(t) = \frac{k\left((t - t_i) M_{i,k-1}(t) + (t_{i+k} - t) M_{i+1,k-1}(t)\right)}{(k-1)(t_{i+k} - t_i)} \tag{3.17}$$

However, now the problem is that the splines no longer sum to 1 (after rescaling with $\frac{1}{n}$). Namely, we can no longer guarantee that the probability masses $P_{i,j}^M$ based on the M-spline sum to 1. Thus, a renormalization is needed to ensure a proper probability mass function. From Figure 3.5 we see that the effect of $\sum_{i=1}^m \tilde{M}_{i,p}(x) \neq 1$ in this case is that observations on the interior are smoothed more than those near the boundary.



(a) **(b)**

Figure 3.4: P_i is area of each rectangle i.e. 0.2.

This can however be a useful property especially for a bivariate Gaussian for which the Copula density, as we shall see, have peaks at $(0, 0)$ and $(1, 1)$ while being relatively smooth elsewhere.

We note that through construction, one can create a family of splines that both sum to 1 have integrals $\frac{1}{m}$. However, as these spline-based smoothing methods does not perform well in general perform very differently for different m with the lack of continuously varying this parameter that acts as a regularizing parameter, as we shall see in chapter 4, we only present the method for constructing such splines in Subsection 6.4.1 and instead consider a more general way of estimating the Copula density, namely kernel density estimators.

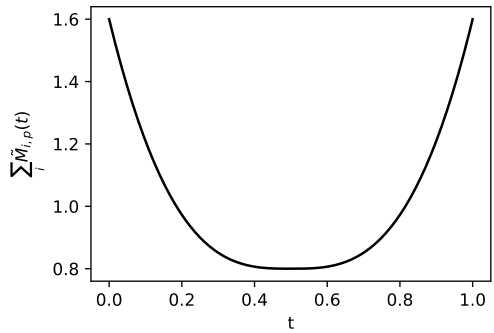


Figure 3.5

3.4.3 Naïve KDE

As we shall see in Subsection 4.3.1, if one can accurately determine the Copula density of X_1 and X_2, then using an approximation of the integral, one can calculate the mutual information to any precision wanted. This clearly follows from the above analysis regarding the behavior of the discretization of X_1 and X_2 in the limit as the mesh gets more fine. Thus, if we can estimate the joint Copula density well, we obtain a good estimate of the mutual information of X_1 and X_2. A widely used non-parametric method for density estimation is kernel density estimation. Namely, if $\{\boldsymbol{x}_i\}$ is a set of n, d-dimensional observations from a population, i.e. $\boldsymbol{x}_i$ can be both scalars and vectors in the case of a multi-dimensional distribution, the kernel density estimator (KDE) of the probability density function is in general given as

$$\hat{f}(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\prod_{j=1}^{d} \boldsymbol{h}_{i,j}} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{\boldsymbol{h}_i}\right) \tag{3.18}$$

where $\boldsymbol{h}_i$ is the bandwidth (vector) associated with observation $\boldsymbol{x}_i$ and K is the kernel (function), defined on the domain of $\boldsymbol{X}$ which is often $\mathbb{R}^d$. Often, the bandwidths $\boldsymbol{h}_i$ are taken to be equal and initially, we shall do so as well. Furthermore, the kernel K is a non-negative function, and is in itself a density function i.e. integrates to 1 as shown below. This ensures that $\hat{f}$ in Equation 3.18 integrates to 1 and is non-negative i.e. a proper distribution.

$$\int_{\mathbb{R}^d} K(\boldsymbol{x})\, d\boldsymbol{x} = 1$$

In one dimension, a particular useful kernel is the Gaussian kernel given by $K(x) = \phi(x)$ where ϕ is the density function for the standard Normal distribution. ϕ is chosen due to its simple behavior and mathematical properties.

In particular, we shall see in the following section, that the properties of the Gaussian kernel allows for simple expressions when correcting for a boundary such that computation is quick and efficient. For multiple dimensions, we often consider product kernels, which are kernels K of the form

$$K(\boldsymbol{x}) = \prod_{i=1}^{d} K_i(x_i) \tag{3.19}$$

I.e. just a product of kernels. In particular, we choose $K_i = \phi$ again due to the numerical properties. Thus, initially, we have a KDE $\hat{f}$ of the following form where we once again note that $\boldsymbol{h}_i = \boldsymbol{h}$ for all $i \in \{1, \ldots, n\}$ such that h_j denotes the bandwidth associated with the jth dimension.

$$\hat{f}(\boldsymbol{x}) = \frac{1}{n \prod_{j=1}^{d} h_j} \sum_{i=1}^{n} \prod_{j=1}^{d} \phi\left(\frac{x_j - x_{i,j}}{h_j}\right)$$

The choice of bandwidth $\boldsymbol{h}$ is important regarding a trade-off between the variance and bias of the KDE. In general, we want to choose h as small as possible resulting in the least bias but a too small $\boldsymbol{h}$ will result in large variance of the estimator. In particular, $\boldsymbol{h}$ acts as a smoothing parameter like the number of bins from the previous section, but here, we can choose any $h > 0$ making the KDE a much more versatile tool. Often the *Mean Integrated Square Error* (MISE) is used which is the expected L^2-norm of $\hat{f} - f$ i.e.

$$\text{MISE}\left(\hat{f}\right) = \mathbb{E}_f\left[\int_{\mathbb{R}^d} \left|\hat{f}(\boldsymbol{x}) - f(\boldsymbol{x})\right|^2 d\boldsymbol{x}\right]$$

which of course depends on $\boldsymbol{h}$. The expectation $\mathbb{E}_f$ denotes the expectation with respect to the samples $\{\boldsymbol{x}_i\}$ of $\boldsymbol{X}$ with (true) density distribution function f. Expanding the above expression, we obtain a simple expression relating MISE to the integrated squared bias and integrated variance as shown below

$$\text{MISE}\left(\hat{f}\right) = \int_{\mathbb{R}^d} \left|\mathbb{E}_f\left[\hat{f}(\boldsymbol{x})\right] - f(\boldsymbol{x})\right|^2 d\boldsymbol{x} + \int_{\mathbb{R}^d} \text{Var}\left[\hat{f}(\boldsymbol{x})\right] d\boldsymbol{x}$$

It is however quite complicated to optimize the above, and we shall thus often use a simple rule of thumb known as Scott's rule [21] for choosing $\boldsymbol{h}$. Namely, for product kernels, we let the bandwidths of each dimension j equal the following where $\hat{\sigma}_j$ is the standard deviation estimated from the observations of X_j

$$h_j^{Scott} = \hat{\sigma}_j n^{-1/(d+4)}, \quad j \in \{1, \ldots, d\}$$

In Figure 3.6, we have shown a basic example in one dimension with two different manual choices of the bandwidth h and h chosen by Scott's rule. We have used

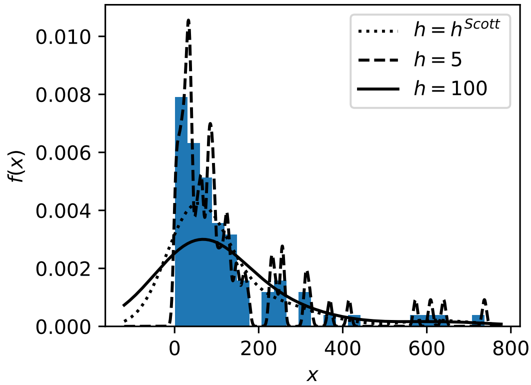


Figure 3.6: The suicide data from MHE. The 86 observations are shown as a histogram of densities along with Gaussian KDEs with bandwidth $h = 5$ and $h = 100$ and h chosen from Scott's rule $h = \hat{\sigma} n^{-1/5} \approx 59.86$.

data from [11], tabulated in [22] which has been used in [23] which propose a method for correcting the KDE near a boundary which we shall discuss in the following section. The data consists of 86 observations regarding suicide and is known to be non-negative. In consideration of the reader, we have included the observations in Table 6.1. It is clear that using h^{Scott} results in what we qualitatively would deem a good estimate for the probability density function as $h = 100$ seen to be overly smoothed whereas $h = 100$ too under-smoothed. In particular, from repeated samples we would expect the estimator using $h = 100$ would have large bias but small variance whereas for $h = 5$ would have much larger variance but smaller bias. However, a problem the estimators, $h = 100$ and $h = h^{Scott}$ especially, have is that they have probability mass below 0 which in this case is unwanted. I.e. when restricting $\hat{f}$ to $[0, \infty)$ they are no longer proper probability distribution functions as they do not integrate to 1. A Simple fix could be to simply rescale $\hat{f}$ such that this is the case, but as seen from the example in Figure 3.7 where this method is applied to the same example as above, this tends to underestimate the peaks especially near the boundary.

We note that using a non-constant h would improve on this behavior, but simpler methods exists, and we thus proceed in the next section with a method that shows great promise regarding this seemingly fundamental issue with KDE. In particular, we refer to a systematic way of letting the shape of each of the kernels depend on the associated observation x_i.

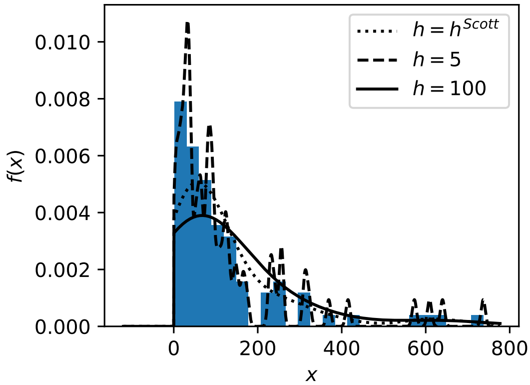


Figure 3.7: Using a rescaled version of $\hat{f}$ on the interval $[0, \infty)$ and disregarding any probability mass below $x = 0$ we obtain proper probability distributions once again. However, neither of the methods capture the peak near the boundary $x = 0$. In particular, although h^{Scott} still seem to be a good choice for h, the KDE does not capture the tendency observed in the data.

3.4.4 Boundary Corrected KDE

Before introducing the boundary corrected kernels presented by [23], we mention another simple method of boundary correction called reflection. Namely, suppose without loss of generality $x = 0$ is the lower boundary of the domain of $\boldsymbol{X}$ and let $\hat{f}$ be KDE as from the previous section. Then, the reflection boundary corrected KDE denoted $\hat{f}_R$ is defined as

$$\hat{f}_R(x) = \hat{f}(x) + \hat{f}(-x)$$

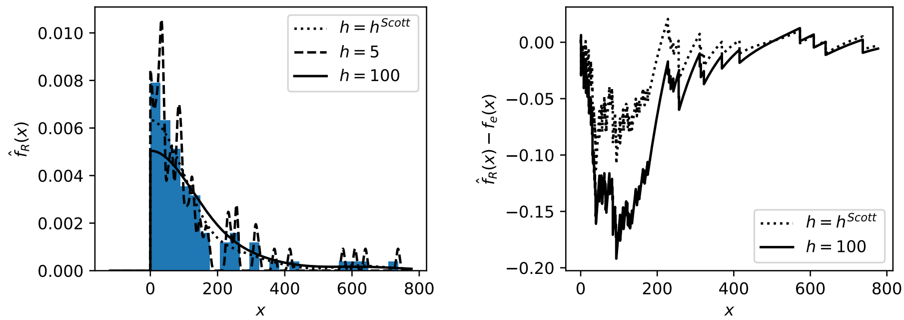
Clearly, $\hat{f}_R$ is non-negative, and it follows from the below that it is also a proper density function as the probability mass is 1

$$\int_0^\infty \hat{f}_R(x)\ dx = \int_0^\infty \hat{f}(x) + \hat{f}(-x)\ dx = \int_0^\infty \hat{f}(x) + \int_{-\infty}^0 \hat{f}(x)\ dx = 1$$

Also, the above is easily extended to two boundaries. Namely, if the domain is $[a, b]$, the reflection boundary corrected KDE is given by

$$\hat{f}_R(x) = \hat{f}(x) + \hat{f}(2a - x) + \hat{f}(2b - x), \quad x \in [a, b]$$

If we once again apply this to the suicide data, comparing to Figure 3.7 we see a big improvement near the boundary as shown in Figure 3.8(a). However, we still proceed with the method originally presented in [23]. The reason for this is that when testing for distribution type using the Kolmogorov Smirnov test (on a 5% significance level) we reject that the observations originate from $\hat{f}_R$ with $h = 100$. For $h = 5$ we do not but due to the above considerations



(a) **(b)**

Figure 3.8

regarding the integrated point wise variance of the estimator, this choice of h is undesired in either case. For $h = h^{Scott}$ we do not reject the distribution but as the shape of the error resembles the error for $h = 100$ we suspect that there is some systematic error which we also see from Figure 3.8(b). Furthermore, the largest deviation from the empirical distribution to $\hat{f}_R$ is close to the boundary (as expected). The test statistics (the largest absolute difference D between the distribution functions and the adjusted statistic) is shown in Table 3.1 where the adjusted test statistic should be compared to the critical value 1.358 on a 5% significance level. Furthermore, to really test the kernel density estimators, one should compute the MISE based on bootstrap and/or cross-validation. In particular, this way larger h i.e. more smoothing would be more favorable compared to the Kolmogorov Smirnov test results which shows small h best represent the empirical distribution.

h	D	Adjusted D
5	0.029042	0.27315
h^{Scott}	0.11262	1.0593
100	0.19194	1.8053

Table 3.1: $(\sqrt{n} + 0.12 + 0.11/\sqrt{n})\, D$

We now turn our attention to the boundary corrected KDE from [23]. They shortly described this in [7] where it was used to estimate the mutual information in terms of Copula entropy. However, issues arise when using this KDE in terms of non-negativity of the KDE and the facts that it does not integrate to unity. All of these issues can however be handled in a general way without effecting the results regarding bias of the estimator as we shall also see from the above example on suicide data. In particular, [23] show that the bias of their estimator $\hat{f}_L$ is of order $\mathcal{O}\left(h^2\right)$ whereas the reflection method discussed above has $\mathcal{O}\left(h\right)$ bias. The basic idea of $\hat{f}_L$ is that it is a linear combination of a symmetric kernel K and $x\,K$ i.e. a first order kernel. They do however only give explicit expressions when a lower bound at $x = 0$ is enforced but the math generalize nicely to 2 boundaries. In particular, we shall let x_u denote the upper bound of the domain and keep $x = 0$ as the lower bound. Before continuing with the derivation and results regarding implementation for the Gaussian kernel we note that in `R`[2], there is a library `evmix` which implements the boundary corrected KDE from [23] but only for 1 dimension and only with a lower bound at $x = 0$. We thus expand on this library (although in `Python`) to include both lower and upper bounds and furthermore, generalize to multiple dimensions using a product kernel as noted in Equation 3.19.

To expand on the boundary corrected KDE to include an upper bound on the domain, we define the functions $a_m\left(x\right)$. Note that in [23], they define a_m as a function of $p = x/h$ where h is the bandwidth, but to keep the expression later on easier to understand in terms of kernel centers etc. we instead define them as a function of $x \in [0, x_u]$. The reason for initially defining a_m in terms of h is to keep the bandwidth out of the expression, which we then define as follows, when there is no upper bound

$$a_m\left(x\right) = \int_{-\infty}^{\frac{x}{h}} u^m K\left(u\right)\, du, \quad x \in [0, \infty)$$

The above is actually equal to the part of the mth moment of the kernel centered at x (and width bandwidth h), that is inside the interval $[0, \infty)$ up to a difference in sign. In particular, using the change of variables $z = x - hu$ we have that

$$a_m\left(x\right) = (-1)^m \int_0^{\infty} \left(\frac{z-x}{h}\right)^m \frac{1}{h} K\left(\frac{z-x}{h}\right) dz$$

where we have used that K assumed to be symmetric. Indeed, the above is as described the part of the mth moment that comes from $[0, \infty)$ (up to a difference in sign) of the kernel that is centered at x with bandwidth h. From this, it is a natural extension to replace the upper bound of the integral with x_u which, when expanding on the initial definition in [23] as done in e.g. [24] turns out to be the correct adjustment. Thus, a_m can be understood as part of the moments which we shall then use as normalizing functions. Thus, for an upper bound x_u

[2] https://search.r-project.org/CRAN/refmans/evmix/html/bckden.html

we find that instead a_m is defined as

$$a_m(x) = \int_{\frac{x-x_u}{h}}^{\frac{x}{h}} u^m K(u)\, du, \quad x \in [0, x_u]$$

The boundary adjusted KDE which we shall denote K^L and index depending on the ith kernel center is then given by

$$K_i^L(x) = \frac{1}{h} \frac{a_2(x) - a_1(x)\frac{x-x_i}{h}}{a_0(x)\, a_2(x) - a_1^2(x)} K\left(\frac{x-x_i}{h}\right) \tag{3.20}$$

For the Gaussian kernel, $a_m(x)$ for $m \in \{0, 1, 2\}$ are easily computed and implemented in code through standard routines as they have simple closed forms in terms of ϕ and Φ, i.e. the standard Gaussian density and distribution functions. Namely, for a_0 we simply have that by definition of Φ

$$\begin{aligned} a_0(x) &= \int_{\frac{x-x_u}{h}}^{\frac{x}{h}} \phi(u)\, du \\ &= \Phi\left(\frac{x}{h}\right) - \Phi\left(\frac{x-x_u}{h}\right) \end{aligned}$$

And similarly, for a_1, using that $\int u\,\phi(u)\, du = -\phi(u) + C$

$$\begin{aligned} a_1(x) &= \int_{\frac{x-x_u}{h}}^{\frac{x}{h}} u\phi(u)\, du \\ &= \phi\left(\frac{x-x_u}{h}\right) - \phi\left(\frac{x}{h}\right) \end{aligned}$$

Finally, for a_2 we have, using integration by parts for the first step

$$\begin{aligned} a_2(x) &= \int_{\frac{x-x_u}{h}}^{\frac{x}{h}} u^2\phi(u)\, du \\ &= \left[-u\phi(u)\right]_{\frac{x-x_u}{h}}^{\frac{x}{h}} + \int_{\frac{x-x_u}{h}}^{\frac{x}{h}} \phi(u)\, du \\ &= \left(\frac{x-x_u}{h}\phi\left(\frac{x-x_u}{h}\right) - \frac{x}{h}\phi\left(\frac{x}{h}\right)\right) + \left(\Phi\left(\frac{x}{h}\right) - \Phi\left(\frac{x-x_u}{h}\right)\right) \\ &= \frac{x}{h}a_1(x) - \frac{x_u}{h}\phi\left(\frac{x-x_u}{h}\right) + a_0(x) \end{aligned}$$

From the improved bias of $\mathcal{O}\left(h^2\right)$ for K_i^L, we have then obtained an estimator that should perform better in terms of a smaller MISE as we have less bias and by choosing h optimally, we should expect small variance as well. Figure 3.9(a) shows the KDE $\hat{f}_L$ using K^L as the kernel (with $x_u = \infty$) and is compared to the reflected kernel from above using the same bandwidth. We note that $\hat{f}_L$ has be rescaled such that it integrates to unity on $[0,\infty)$. Also, from Figure 3.9(b) we see that the density is not statistically significant, and although we can not conclude from this alone, we observe that the deviation from the empirical distribution functions is less than that of the reflected KDE.

As noted above, we need to rescale $\hat{f}_L$ to unity as the kernels in Equation 3.20 does not have unit integrals. Furthermore, $\hat{f}_L$ is not even ensured to be non-negative. This can be handled in multiple ways. One way is to simply take the maximum of $\hat{f}_L$ and 0 effectively cutting off any negative density however in [25], a method is given for KDE based on K_i^L specifically, ensuring the $\mathcal{O}\left(h^2\right)$ bias of the estimator. Namely, let K_i^N be given as

$$K_i^N\left(x\right) = \frac{1}{h\, a_0\left(x\right)} K\left(\frac{x-x_i}{h}\right)$$

which is then the kernel, locally renormalized using a_0. We then define the related KDE as

$$\hat{f}_N\left(x\right) = \frac{1}{n}\sum_{i=1}^{n} K_i^N\left(x\right)$$

which is then non-negative everywhere as $a_0\left(x\right), K\left(x\right) > 0$. The non-negative boundary corrected KDE, denoted $\hat{f}_P\left(x\right)$ is then defined as

$$\hat{f}_P\left(x\right) = \hat{f}_N\left(x\right)\, e^{\frac{\hat{f}_L\left(x\right)}{\hat{f}_N\left(x\right)}-1}$$

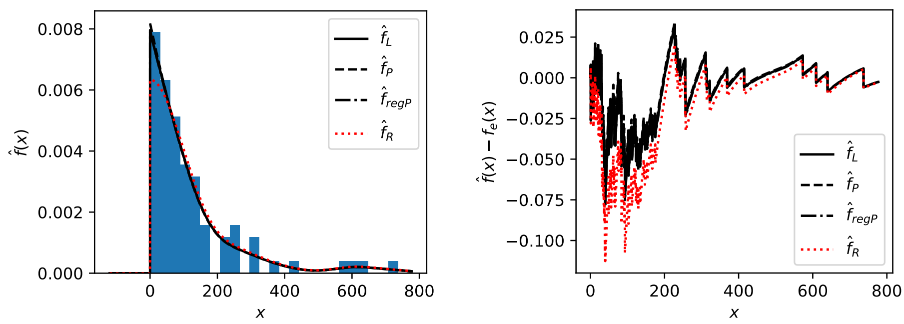
This $\hat{f}_P$ is also shown in Figure 3.9 resulting in identical density functions. We note that $\hat{f}_P$ works by multiplying the non-negative $\hat{f}_N$ by a (large positive) constant when $\hat{f}_L$ is larger than $\hat{f}_N$ and thus drives $\hat{f}_N$ towards $\hat{f}_L$. However, from implementation and trying on different distributions, sometimes we observe some odd behavior that can easily be derived from the definition of $\hat{f}_P$. Thus, we propose a modification to overcome these odd properties of $\hat{f}_P$. Namely, suppose both $\hat{f}_L$ and $\hat{f}_N$ are close to 0 at some point x. If $\hat{f}_L$ is a magnitude of 10 larger than $\hat{f}_N$, then $\hat{f}_P$ is approximately $8000 \cdot \hat{f}_N$ which even if $\hat{f}_N$ is small may be large. Thus, it is possible even if both are close to 0, that $\hat{f}_P >> 0$ which is in contrast to what we would want from the estimator. Thus, we propose a regularized KDE version of $\hat{f}_P$ denoted $\hat{f}_{regP}$ which is obtained from first rewriting $\hat{f}_P$ as follows

$$\hat{f}_P\left(x\right) = \bar{f}\left(x\right)\, e^{\frac{\tilde{f}\left(x\right)-\hat{f}\left(x\right)}{\hat{f}\left(x\right)}}$$

then, we introduce the regularizing parameters $\lambda \geq 0$ such that

$$\hat{f}_{regP}\left(x\right) = \bar{f}\left(x\right)\, e^{\frac{\tilde{f}\left(x\right)-\hat{f}\left(x\right)}{\hat{f}\left(x\right)+\lambda}}$$

In practice, we have found that $\lambda = 0.001$ is a sufficient regularization whilst preserving the shape. A small λ is preferred as then we are ensured $\mathcal{O}\left(h^2\right)$ behavior of the bias. In Figure 3.9 we have also shown this regularized version with $\lambda = 0.001$ and observe that it is basically identical to $\hat{f}_P$ on the domain while also behaving well numerically for large x, where the issue discussed above arise for $\hat{f}_P$. Before continuing with a few more interesting methods of density



(a) **(b)**

Figure 3.9

estimation and mutual information estimation in particular, we note that in the same R package there is a KDE based on the Gaussian Copula density which at first glance might seem a good choice especially if we are trying to estimate the density of a Copula that we know to be Gaussian. The KDE is based on [26], and trivially we get that the domain of the basis functions is already $[0, 1]$. In particular, the kernel for a given bandwidth h and kernel center x_i is given by

$$K_i^{GC}(x) = \frac{1}{h\sqrt{2-h^2}} e^{-\frac{\left(1-h^2\right)}{2h^2\left(2-h^2\right)}\left(\left(1-h^2\right)\left(\left(\Phi^{-1}(x)\right)^2+\left(\Phi^{-1}(x_i)\right)^2\right)-2\Phi^{-1}(x)\,\phi^{-1}(x_i)\right)}$$

which is obtained by letting $h^2 = 1-\rho$ in the original expression for the Gaussian Copula density. However, in practice, we shall see that this choice of kernel has some numerical instabilities especially for large correlations i.e. small h and does not perform as well as $\hat{f}_{regP}$ even when using their optimal bandwidth.

Furthermore, we note that as observations are initially transformed such that they are approximately uniform on $[0, 1]$, the variance of these uniform observations $u_i = \hat{F}(x_i)$ will also be approximately $1/12$ and hence the Scott rule of thumb will give a near constant bandwidth. However, as we shall see later, the performance is drastically improved by choosing smaller bandwidth for subdomains with high density. We thus note that a local bandwidth should be considered in the future. This could e.g. by using k-nearest neighbor and the distances of these which is the basic idea of [27]. Although we will not be using this method as it is out of scope for this paper, we note that their estimator seem to result in good estimates when $n \geq 300$. Using the core ideas of this paper, one could perhaps deduce a better algorithm for choosing h globally or locally.

Finally, we note that a recent method based on diffusion [28] [29] which shows great potential. Without going into too much detail, it works by considering the following PDE

$$\frac{\partial}{\partial t}\hat{f}\left(x;t\right) = \frac{1}{2}\frac{\partial^2}{\partial x^2}\hat{f}\left(x;t\right)$$

where $t = h^2$. The Gaussian kernel is then the unique solution when the domain is $\mathbb{R}$ with condition $\hat{f}\left[x;0\right] = \frac{1}{n}\sum_{i=1}^{n}\delta\left(x - x_i\right)$ i.e. the observations is the boundary condition at $h = 0$. If instead we add that the support should be e.g. $[0,1]$ the Neumann boundary conditions on the boundary results in an analytical solution. Namely, from the boundary conditions

$$\frac{\partial}{\partial x}\hat{f}\left(x;t\right)\bigg|_{x=0} = \frac{\partial}{\partial x}\hat{f}\left(x;t\right)\bigg|_{x=1} = 0$$

the analytical solution to $\hat{f}$ is

$$\hat{f}_D\left(x;t\right) = \frac{1}{n}\sum_{i=1}^{n}K_i^D\left(x\right)$$

where

$$K_i^D\left(x\right) = \frac{1}{h}\sum_{k=-\infty}^{\infty}\phi\left(\frac{x-(2k+x_i)}{h}\right) + \phi\left(\frac{x-(2k-x_i)}{h}\right)$$

which is similar to the reflection method from above. They even give an algorithm for calculating t which from the example below appear to work very well. Before presenting this example we note that although it is out of scope to use this method in the following chapter, it would be interesting to see how well it would perform as it is the only KDE here that does not use any transformation near the boundary while still being consistent at the boundary as they show in [29].

Example 3.2 (Diffusion based KDE)**.** *Let* $X \sim Beta(1,4)$ *such that* $f(x) = 4\left(1-x\right)^3$ *for* $x \in [0,1]$ *and thus* $F\left(x\right) = 1-(1-x)^4$*. Generating* 1000 *samples from the distribution and using the algorithm in [29] to estimate the bandwidth, we get a bandwidth* $h^D = 0.05461$ *whereas with Scott's rule we get* $h^{Scott} = 0.04145$*. From Figure 3.10 we see that diffusion and the regularized boundary corrected KDE* $\hat{f}_{regP}$ *from above agree almost everywhere. Only near the lower* *boundary at $x = 0$ there is a significant difference and here $\hat{f}_{regP}$ seem to fit the true distribution better. This is due to the Neumann boundary condition which enforce $\hat{f}_D$ to have horizontal derivative at both boundaries. Note that we have used both the h^D and h^{Scott} bandwidth for the regularized boundary corrected KDE resulting in basically no difference.*

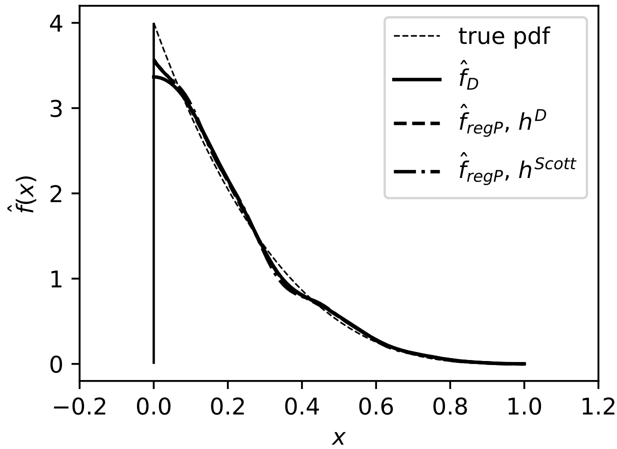


Figure 3.10: Using the diffusion based KDE on 1000 samples from a Beta $(1, 4)$ distribution, we see that in general it fits very well. Only at the boundary $x = 0$ there seems to be a problem which on the other hand, $\hat{f}_{regP}$ does not seem to suffer from to the same extend. In particular, repeating the experiment, $\hat{f}_{regP}$ is on average 4 at $x = 0$ while $\hat{f}_D$ constantly seem to undershoot.

At this point, we thus have a complete set of methods for both estimating the mutual information from observations and from these, algorithms to estimate the (causal) structure depending on assumptions regarding direction i.e. a topological ordering of the variables. We thus proceed with numerical results in the following chapter and apply the framework on both fully controlled systems and the observations from chapter 2.

Chapter 4

Results and Discussion

In this section, we will apply Algorithm 1 and Algorithm 2 on different examples to assess the capabilities and pain points of the framework. In particular, we shall initially consider a simple network whose causal structure is a simple chain and where only linear correlation exists between the random variables constituting the chain. We shall see that using correlations and an assumption of the topological structure, we can perfectly reconstruct the causal structure from G_{obs}. Furthermore, we shall see that using mutual information as the measure of association instead, we are still able to reconstruct the causal structure. We also argue why mutual information is preferable in a general setting and quantify the resulting errors.

After having considered a simple causal structure, we extend to a general network, with variables still only linearly correlated. Once again, using correlation and an assumption of the topological order of the random variables, we can perfectly reconstruct the causal structure. We shall however also use mutual information and observe that once again, we infer the true causal structure.

After the above results concerning Algorithm 2, we shall compare the different methods from Section 3.4 for estimating mutual information. In particular, we shall see how the estimator used in [6] performs on a few samples and compare this to the KDE-based method.

From this, we shall investigate how the algorithms work when combined. Specifically, we shall observe that preconditioning of the variables is important for accurate estimates of mutual information. Furthermore, we shall see that based on only a few hundred samples from the above network, the causal structure is recovered.

Finally, we shall use the combined learnings from the above on the pharmaceutical dataset, described in chapter 2. In particular, depending on the included variables and assumptions of topological structure, we shall infer a few possible causal structures and comment on these.

4.1 Gaussian chains

In this section, we discuss the errors made from the assumption that indirect effects can be computed as a sum of powers of the direct effects, i.e. $G_{indir} = \sum_{k\geq 1} G_{dir}^k$. In particular, on a theoretical level, we shall observe the error in G_{obs} based on the above assumption of how similarities are *convolved* which we equate with the noise N from Subsection 3.3.3, although it is a systematic error. To do this, we shall use a multivariate Gaussian to control the correlation and as an extension of this, the mutual information between pairs of random variables. As we already know, correlation and mutual information are independent of the mean and variance of each of the variables however for a bivariate Gaussian the mutual information can be computed directly from the correlation as stated in the following proposition.

Proposition 4.1. *Given a bivariate normal distribution* $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ *where*

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma^2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$

Then the mutual information $I(X_1, X_2) = -\frac{1}{2}\ln\left(1-\rho^2\right)$.

Proof. This follows by direct computation Using e.g. that $I(X_1, X_2) = h(X_1) + h(X_2) - h(X_1, X_2)$ □

Thus, if we know the correlation structure of a Gaussian random vector, we also know the mutual information between every pair of variables. We shall use this in the following made-up example. Namely, define a Gaussian chain in terms of a Gaussian random vector in the following way. Let $\boldsymbol{X}$ be a d-dimensional Gaussian random vector, the $\boldsymbol{X}$ is a standard Gaussian chain if it can be written in the following way in terms of d independent standard normal variables Z_i up to a permutation. I.e. there exists a permutation of the variables of the random vector $\boldsymbol{X}$ that permits the following structure.

$$\begin{aligned} X_1 &= Z_1 \\ X_2 &= \vec{\rho}_{1,2}X_1 + \sqrt{1-\vec{\rho}_{1,2}^2}Z_2 \\ X_3 &= \vec{\rho}_{2,3}X_2 + \sqrt{1-\vec{\rho}_{2,3}^2}Z_3 \\ &\vdots \\ X_d &= \vec{\rho}_{d-1,d}X_{d-1} + \sqrt{1-\vec{\rho}_{d-1,d}^2}Z_d \end{aligned} \tag{4.1}$$

It follows that the marginals have variance 1 as clearly $\text{Var}[X_1] = \text{Var}[Z_1] = 1$ and for $i > 1$, $\text{Var}[X_i] = \vec{\rho}^2_{i-1,i}\text{Var}[X_{i-1}] + \left(1 - \vec{\rho}^2_{i-1,i}\right)\text{Var}[Z_i] = 1$ by independence of X_{i-1} and Z_i. Thus, the above structure also implies the Cholesky factorization of the correlation matrix for $\boldsymbol{X}$, namely

$$L = \begin{bmatrix} 1 & & & & \\ \vec{\rho}_{1,2} & \sqrt{1-\vec{\rho}^2_{1,2}} & & & \\ \vec{\rho}_{2,3}\vec{\rho}_{1,2} & \vec{\rho}_{2,3}\sqrt{1-\vec{\rho}^2_{1,2}} & \sqrt{1-\vec{\rho}^2_{2,3}} & & \\ \vdots & & & \ddots & \\ \prod_{i=2}^{d}\vec{\rho}_{i-1,i} & \dots & \sqrt{1-\vec{\rho}^2_{j-1,j}}\prod_{i=j+1}^{d}\vec{\rho}_{i-1,i} & \dots & \sqrt{1-\vec{\rho}^2_{d-1,d}} \end{bmatrix}$$

This will allow us to both sample from such a chain and calculate G_{dir} and G_{obs} theoretically. However, in this example, it is easier to calculate the correlation between the variable X_i and X_j directly. As the variance of each variable is 1 we simply calculate the covariance. We assume without loss of generality that $i < j$ whence

$$\text{Cov}[X_i, X_j] = \text{Cov}\left[X_i, \vec{\rho}_{j-1,j}X_{j-1} + \sqrt{1-\vec{\rho}^2_{j-1,j}}Z_j\right] = \vec{\rho}_{j-1,j}\text{Cov}[X_i, X_{j-1}]$$

which by induction implies $\rho_{i,j} = \prod_{k=i+1}^{j}\vec{\rho}_{k-1,k} = \rho_{j,i}$. At this point, we are almost ready to use the algorithms from the previous chapter. First, we will only use Algorithm 2 to deconvolve the network based on theoretical correlations and later mutual information. However, before doing so, we note that from the definition in Equation 4.1 the random variable $\boldsymbol{X}$ exhibits a Markovian property. Namely, the X_i above can be understood discrete stochastic process as they are successively drawn based only on the previous variable X_{i-1} i.e. $f(x_i \mid X_{i-1}, X_{i-2}, \dots, X_1) = f(x_i \mid X_{i-1})$. Thus, if the algorithm works as intended, we should observe that the deconvolved network is a *chain* of variables as shown in the Figure 4.1 Thus, we now have the expected result, and we

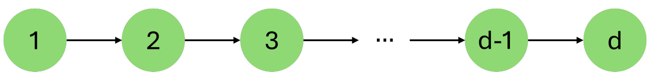


Figure 4.1: The graphical representation of a Gaussian chain. Arrows signify a possible causal structure. If furthermore, one assumes that X_1 is generated first, then X_2 and so on, this is the only causal structure that would make sense.

proceed with using correlation and mutual information to try and rediscover this structure in the following two sections.

4.1.1 Gaussian chain deconvolution using correlation

In this section, we will use the observed correlations as elements of G_{obs}. In particular, the (i,j) entry of G_{obs} is $\rho_{i,j} = \prod_{k=i+1}^{j} \vec{\rho}_{k-1,k}$ when $i < j$ and 0 otherwise. This makes G_{obs} strictly upper triangular. Note that although it makes sense to consider the correlation between a variable and itself, we shall as discussed before set the diagonal to 0. The reason for this becomes clear when we try to convolve G_{dir} based on the initial definition of a general Gaussian chain in Equation 4.1. We note that in Algorithm 1 we usually (without an assumption of the topology of the random variables) use a symmetrical G_{obs}. We shall however postpone this discussion a bit and first use an upper triangular G_{obs}. In particular, we shall observe that we perfectly recover the *directional* correlations $\vec{\rho}_{k-1,k}$ from Equation 4.1 through Equation 3.6.

As G_{obs} is in this case strictly upper triangular, the spectral radius is 0 and hence we have no problems with convergence of the infinite sum of powers of (the uniquely defined) G_{dir}. From the above, it is clear that G_{obs} is given as follows

$$G_{obs} = \begin{bmatrix} 0 & \vec{\rho}_{1,2} & \vec{\rho}_{1,2}\,\vec{\rho}_{2,3} & \cdots & \prod_{k=2}^{d} \vec{\rho}_{k-1,k} \\ & 0 & \vec{\rho}_{2,3} & \cdots & \prod_{k=3}^{d} \vec{\rho}_{k-1,k} \\ & & \ddots & & \vdots \\ & & & 0 & \vec{\rho}_{d-1,d} \\ & & & & 0 \end{bmatrix} \tag{4.2}$$

Now, let G_{dir} be given as follows

$$G_{dir} = \begin{bmatrix} 0 & \vec{\rho}_{1,2} & & & \\ & 0 & \vec{\rho}_{2,3} & & \\ & & \ddots & \ddots & \\ & & & 0 & \vec{\rho}_{d-1,d} \\ & & & & 0 \end{bmatrix}$$

then G_{dir}^2 is given by

$$G_{dir}^2 = \begin{bmatrix} 0 & 0 & \vec{\rho}_{1,2}\,\vec{\rho}_{2,3} & & & \\ & 0 & 0 & \vec{\rho}_{2,3}\,\vec{\rho}_{3,4} & & \\ & & \ddots & \ddots & \ddots & \\ & & & 0 & 0 & \vec{\rho}_{d-2,d-1}\,\vec{\rho}_{d-1,d} \\ & & & & 0 & 0 \end{bmatrix}$$

It is not hard to show that in fact $\sum_{k\geq 1} G_{dir}^k = \sum_{k=1}^{d} G_{dir}^k = G_{obs}$. Thus, if we know a graph topological ordering of the random variables corresponding to the structural causal model, we completely recover (without any error) the direct dependencies/correlation from the initial definition in Equation 4.1. This result holds for a general *chain* where Z_i can follow any distribution as long as they are uncorrelated. This follows from the above computation of $\text{Cov}\left[X_i, X_j\right]$, where no assumption of Z_j was needed except for correlation 0.

From the above, we might think that if we have a topological ordering of the random variables this is the preferred method, and it is as long as correlation is a good enough measure of similarity/codependency. Albeit this is only shown for the special case of a chain, in Section 4.2 we consider the more general case and conclude that this indeed holds. Regarding the comment on correlation being a good enough measure of similarity, a prototypical case is when the joint probability density function of two variables resembles a parabola. Namely, let $X_1 \sim \mathcal{U}(0,1)$ and $X_2 \mid X_1 \sim \mathcal{N}\left(1 - 4\left(x_1 - 1/2\right)^2, \sigma^2\right)$ i.e. $X_2 = 1 - 4\left(X_1 - 1/2\right)^2 + \epsilon$ where $\epsilon \sim \mathcal{N}\left(0, \sigma^2\right)$. In Figure 4.2, 1000 samples from this distribution is shown for $\sigma = 1/10$ along with the expectation $\mathbb{E}\left[X_2 | X_1\right]$. It is not hard to show that the covariance between X_1 and X_2 is 0 however we see a relationship between the two variables. Computing the mutual information results in $I\left(X_1, X_2\right) \approx 1.030$ implying $X_1 \not\perp X_2$ i.e. there exists a higher order (non-linear) dependency. Thus, if the algorithm permits, we would prefer mutual information to correlation as we can then use observed higher-order relationships to infer a causal structure. On a more technical note, mutual information is a

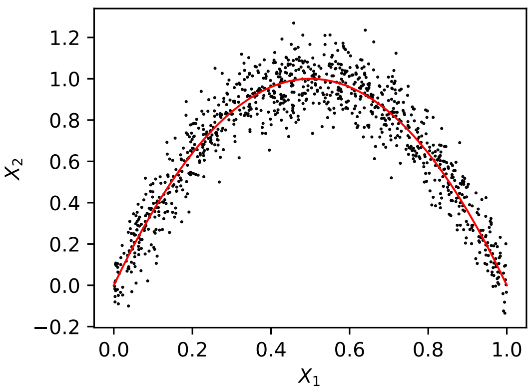
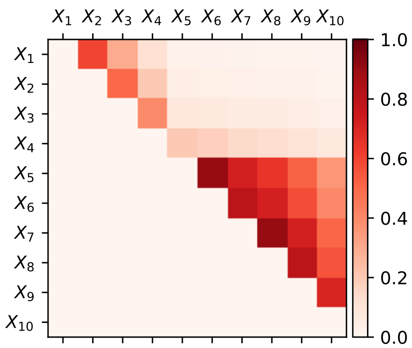


Figure 4.2: 1000 samples generated from $X_1 \sim \mathcal{U}(0,1)$ and $X_2 \mid X_1 \sim \mathcal{N}\left(1 - 4\left(x_1 - 1/2\right)^2, \sigma^2\right)$ with $\sigma = 1/10$. The mutual information is calculated theoretically to be $I\left(X_1, X_2\right) \approx 1.030$ and repeated simulations show that the empirical correlation is symmetric around 0 supporting the claim that the underlying correlation is in fact 0

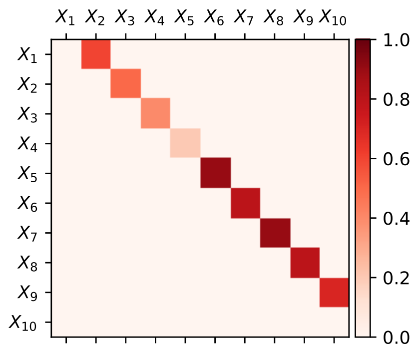
more general measure of correlation i.e. not just linear correlation. We proceed with a 10-Gaussian chain defined by the following correlations:

$$\begin{aligned} \rho_{1,2} &= 0.6, & \rho_{2,3} &= 0.5, & \rho_{3,4} &= 0.4 \\ \rho_{4,5} &= 0.2, & \rho_{5,6} &= 0.9, & \rho_{6,7} &= 0.8 \\ \rho_{7,8} &= 0.9, & \rho_{8,9} &= 0.8, & \rho_{9,10} &= 0.7 \end{aligned} \tag{4.3}$$

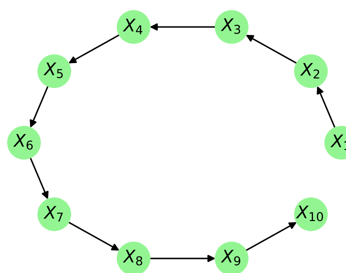
We have chosen correlations of different sizes to check if the deconvolution is robust in the presence of both strong and weak links. In particular, X_5 is only $\rho_{4,5}^2 = 4\%$ of X_4 and the remaining 96% is noise/indescribable variance i.e. a very weak link between the first part of the chain up to and including X_4 and the rest. However, as discussed above, if let G_{obs} be upper triangular, we should completely rediscover these direct relations which is indeed also the case.



(a) G_{obs}



(b) G_{dir}



(c) G_{dir} as a digraph

Figure 4.3: Results from using an upper triangular G_{obs} and correlation to infer the causal network structure. (a) shows the upper triangular G_{obs} with the correlation between every pair of variables. (b) shows the deconvolved G_{obs} and as we expect, the superdiagonal contains the original correlations given in Equation 4.3. (c) shows G_{dir} represented as a digraph and matches the expected result.

In particular, from Figure 4.3 we observe that the inferred network, represented by G_{dir}, is indeed a chain of variables and is exactly equal to the theoretical G_{dir} as we would expect (up to very small rounding errors of the size 10^{-16}). The *estimated* G_{dir} is also shown as a directed graph which the initial topological assumption implies, with edges wherever G_{dir} is non-zero.

We now proceed to investigate what happens when we remove the prior information of the topological ordering. Namely, if G_{obs} is no longer triangular but symmetric. In particular, let T_{dir} be given as G_{dir} above. We then have that G_{dir} in the symmetric case is $T_{dir}+T_{dir}^T$ and similarly for G_{obs}, $G_{obs} = T_{obs}+T_{obs}^T$. Clearly, $I + G_{obs}$ is positive definite as it is a proper correlation matrix. However, that also implies that we might have eigenvalues of G_{obs} less than or equal to $-1/2$ which we know from Subsection 3.1.1 is not the result of a G_{dir} such that Equation 3.4 holds as then the infinite sum diverges. However, as -1 is not an eigenvalue of G_{obs}, we will investigate what happens if one tries to use Equation 3.6 anyway.

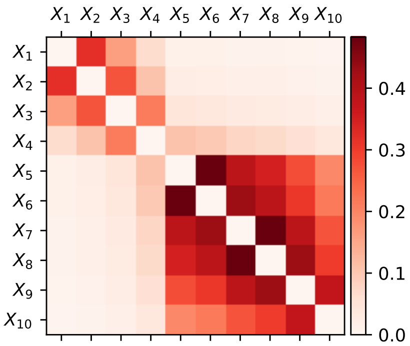
But first, we shall discuss the errors being made using the symmetric G_{obs} and G_{dir} instead of triangular. Namely, we investigate the powers of G_{dir}:

$$G_{dir}^2 = \left(T_{dir} + T_{dir}^T\right)^2 = T_{dir}^2 + \left(T_{dir}^T\right)^2 + T_{dir}T_{dir}^T + T_{dir}^T T_{dir}$$

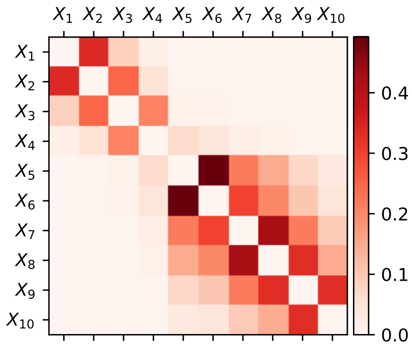
Higher power can be calculated similarly, but for the second power, we already observe an error. The first two terms correspond to a reflection of the second-order effects that we saw above and know to be true, whence the final two terms, that add to a diagonal matrix, is an error and will propagate with higher order powers of G_{dir}. Through simple calculation the resulting error is

$$T_{dir}T_{dir}^T + T_{dir}^T T_{dir} = \begin{bmatrix} \rho_{1,2}^2 + \rho_{2,3}^2 & & & \\ & \rho_{2,3}^2 + \rho_{3,4}^2 & & \\ & & \ddots & \\ & & & \rho_{d-2,d-1}^2 + \rho_{d-1,d}^2 \end{bmatrix}$$

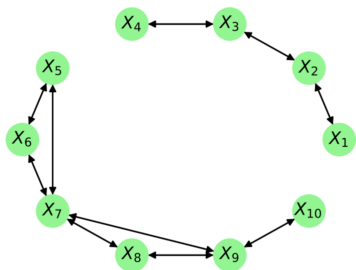
Thus, for chains, we expect larger errors for sub-chains with strong links i.e. a subgraph of a chain that is also a chain where the correlation from one variable to the next is large. Using $G_{obs} = T_{obs} + T_{obs}^T$ we have that the smallest eigenvalue is approximately $\lambda_{\min} \approx -0.92263$ thus, multiplying G_{obs} with a constant $c_s < 0.54192$ will make G_{dir} have spectral radius at most 1. The results vary with one or two edges for the choice of c_s and in the following we have chosen $c_s = 0.53651$ resulting in $\rho\left(G_{dir}\right) \approx 0.98020$ and $\tilde{G}_{obs}$ and $\tilde{G}_{dir}$ as seen in Figure 4.4.



(a) G_{obs}



(b) G_{dir}



(c) G_{dir} as a graph

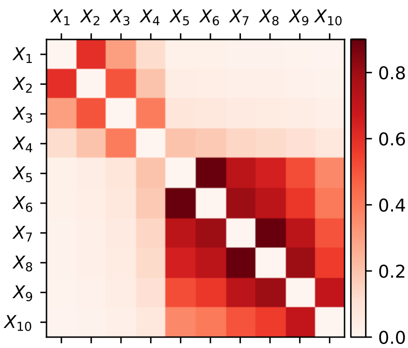
Figure 4.4: Using a symmetric G_{obs} as shown in (a), we observe that higher order effects start to emerge as can be seen in (b). The main response is still in the superdiagonal and subdiagonal as we expect, where some similarity seems to bleed to nearby nodes/variables thus making the threshold used important for the resulting graph. For (c), a threshold $t = 0.2$ was used to obtain a decent compromise between connectedness and denseness of the direct association.

From Figure 4.4(b) we see that some correlation/association seem to bleed to variables 2 or 3 edges away which we of course know is not true given the Markov property discussed above. However, it is also clear that the error here is that the original assumption does not hold since using a symmetric G_{obs} implies that the measure of similarity flows both ways where in this case it is very much unidirectional.

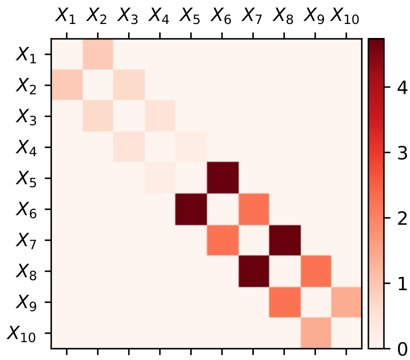
We conclude that we are somewhat able to rediscover the causal structure. Not surprisingly, we observe that the weak link between X_4 and X_5 is one of the first to break and that we observe some extra edges between the later more strongly linked sub-chain as by the above discussion. Finally, before presenting the results for the unscaled G_{obs} (where the smallest eigenvalue is smaller than $-1/2$) we note that changing the parameter α in Algorithm 2 did not have much of an effect indicating that the network is quite sparse (as we also know it to be) as even removing 65% of the smallest correlations from G_{obs} did not have any effect. The chosen threshold of $t = 0.2$ on G_{dir} seemed to be the best compromise of a connected graph and the density of the edges (although this is somewhat biased from prior knowledge of the true graphical structure).

Finally, we try using the unscaled G_{obs} in Equation 3.6. Interestingly, we find that the true structure emerges as can be seen from Figure 4.5. Although the *correlations* in Figure 4.5(b) are not really correlations they do resemble those discovered in Figure 4.3(b). On closer inspection, it is not apparent how they are related except that it is a non-linear relationship. Although in this case, it seemed to work without rescaling G_{obs} when discovering the causal structure. We will in general not apply this to real-world scenarios as the method is not well-defined in terms of assumptions and what the resulting G_{dir} should be interpreted as.

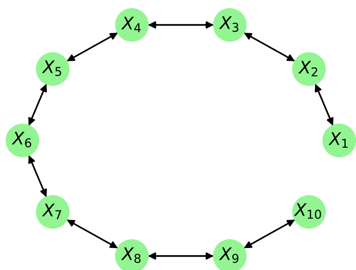
Thus, at this point, we have a rather good understanding of how the method works on Gaussian chains if one uses correlation as a measure of association. Furthermore, if one knows (a plausible) topological ordering of the variables, we can perfectly rediscover the network of direct dependencies. However, as noted above, correlation is not always a good measure of similarity. Thus, we proceed to experiment with mutual information on the same Gaussian chain.



(a) G_{obs}



(b) G_{dir}



(c) G_{dir} as a graph

Figure 4.5: Using an unscaled (symmetric) G_{obs} results in a good recovery of the causal structure as seen in (b) and (c). However, at this point it is not clear whether it holds only for chains and using correlation or if it is a more general phenomenon.

4.1.2 Gaussian chain deconvolution using mutual information

In this section, we continue the example from the previous section, but instead of using correlation as a measure of similarity, we will use mutual information. Immediately, we note that mutual information or Copula entropy does not propagate as assumed in Equation 3.4. As an example, from Proposition 4.1, we know that the mutual information in the case of a Gaussian chain between a variable X_i and the next variable X_{i+1} is $-1/2\log\left(1-\rho_{i,i+1}^2\right)$ and similarly, using Equation 4.2, we have that

$$I\left(X_i, X_{i+2}\right) = -\frac{1}{2}\log\left(1-\rho_{i,i+1}^2\rho_{i+1,i+2}^2\right)$$

Thus, if G_{dir} is triangular, using Equation 3.4 we should observe the following at the $(i, i+2)$ entry of G_{obs} instead

$$\frac{1}{4}\log\left(1-\rho_{i,i+1}^2\right)\log\left(1-\rho_{i+1,i+2}^2\right)$$

I.e. we make an error (which we could take to be the noise N from Subsection 3.3.3) for second order effects equal to

$$-\frac{1}{2}\log\left(1-\rho_{i,i+1}^2\rho_{i+1,i+2}^2\right) - \frac{1}{4}\log\left(1-\rho_{i,i+1}^2\right)\log\left(1-\rho_{i+1,i+2}^2\right)$$

In general, for a Gaussian chain, we have that

$$N_{i,j} = -\frac{1}{2}\log\left(1-\prod_{k=i+1}^{j}\rho_{k-1,k}^2\right) - \left(-\frac{1}{2}\right)^{j-i}\prod_{k=i+1}^{j}\log\left(1-\rho_{k-1,k}^2\right)$$

As we will see in Figure 4.6 and Figure 4.7, for Gaussian chains we can expect some of the same bleeding behavior as observed in Figure 4.4 where we did not use the topological ordering but based the deconvolution on correlation. In particular, from the figures below, we see that for 3-chains, the error is in many cases close to 0 and for most combinations of $\rho_{1,2}$ and $\rho_{2,3}$ less than 0.1. Furthermore, we note that the errors are the largest when it is a strongly connected 3-chain i.e. if both $\rho_{1,2}$ and $\rho_{2,3}$ are close to 1 which again resemble the behavior seen in the case of a symmetrical G_{obs} using correlation as the measure of association although in this case, the error does not propagate to the same extent which we shall also see shortly when applying the deconvolution algorithm. Notice that as only the absolute value of the correlation matters, we only show the error for $\rho_{1,2}, \rho_{2,3} \geq 0$.

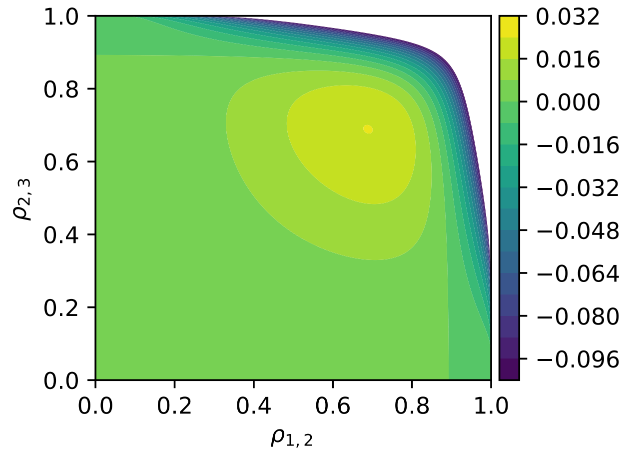
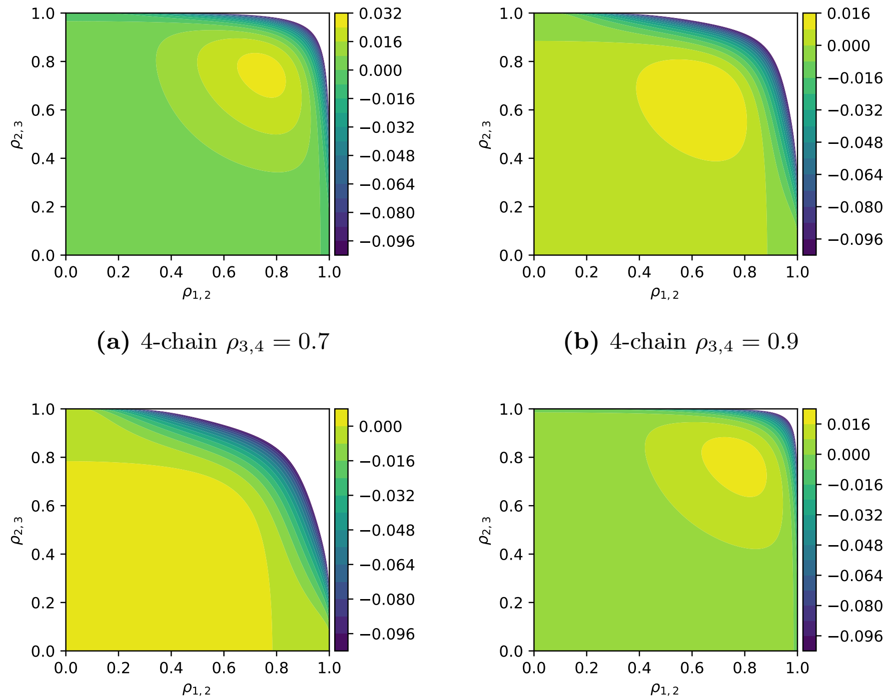


Figure 4.6: The error made by the assumption of G_{obs} and G_{dir} for second order observed effect. Although mutual information does not comply with the underlying assumptions, we observe that in the case of a Gaussian 2-chain, we can expect the error to be relatively small.

We extend the above discussion to 4- and 5-chains (i.e. $j = i + 3$ and $j = i + 4$ in the above expression for N_{ij}) to see how the error propagates in more detail. This is shown in Figure 4.7 for three different scenarios of a 4-chain and a single 5-chain. In particular, as the error $N_{i,j}$ is symmetric in $\rho_{1,2}$, $\rho_{2,3}$ and $\rho_{3,4}$ (and $\rho_{4,5}$ in the case of a 5-chain) and because it is hard to accurately show many-dimensional surfaces, we keep to a fixed set of $\rho_{3,4}$ and $\rho_{4,5}$ when investigating. For the 4-chain, choosing $\rho_{2,3} = 0.9$ (corresponding to mutual information about 0.8304) approximately results in the same error as in Figure 4.6 and if $\rho_{2,3}$ is above e.g. 0.95, we get a worse propagation of errors compared to the 3-chain. Finally, from Figure 4.7(d), we see the same picture i.e. that keeping the correlations and hence information between subsequent variable low results in smaller errors in G_{obs} and hence the inferred G_{dir}. Note that under the assumption of a topological ordering such that G_{obs} is strictly upper triangular results in $\rho\left(G_{obs}\right) = 0$ such that no rescaling is necessary (although different choices of the base of the logarithm would affect how much higher order associations influence G_{dir}).

Having obtained a good understanding of how shifting to mutual information instead of correlation in the case of Gaussian chains, we continue with the above example now using mutual information as the elements of G_{obs} based on the correlation matrix from the previous section. Using a triangular G_{obs} we observe similar behavior to that of the original example using a triangular G_{obs} but with correlation as can be seen from Figure 4.8. In particular, we do not observe the same magnitude of bleeding effects as in Figure 4.4.



(a) 4-chain $\rho_{3,4} = 0.7$

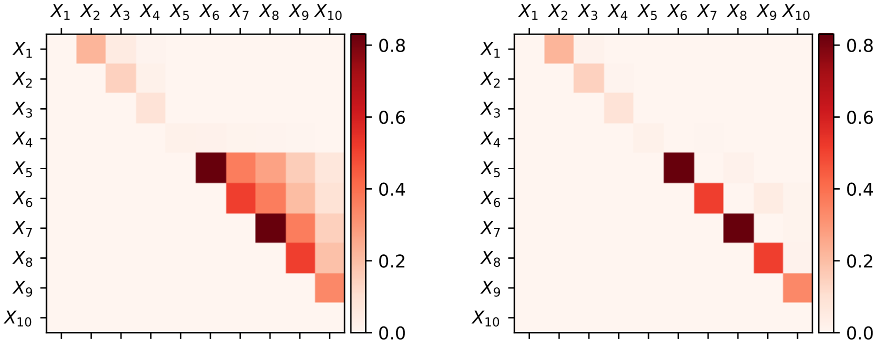
(b) 4-chain $\rho_{3,4} = 0.9$

(c) 4-chain $\rho_{3,4} = 0.95$

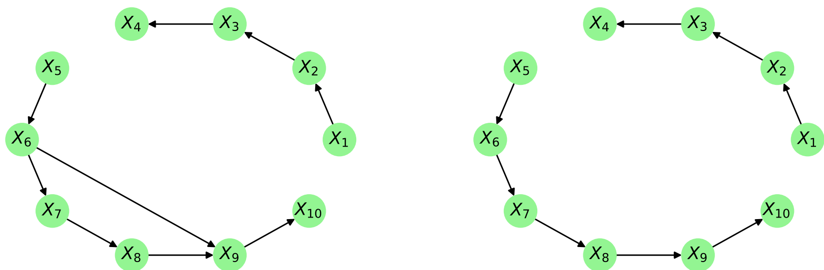
(d) 5-chain $\rho_{3,4} = 0.85$, $\rho_{4,5} = 0.6$

Figure 4.7: Errors of convolving mutual information along a 4-chain (a), (b), (c) and a 5-chain (d). Due to symmetry in the expression of the error, only the first 2 links i.e. $\rho_{1,2}$ and $\rho_{2,3}$ are varied on $[0, 1]$ respectively. Only positive correlations are shown as the sign of the correlation cancels in the expression for the error. We note that large correlations and hence large mutual information on each edge results in larger error. In particular, when not too many of the links are strong, we have almost 0 error.

However, we observe the same tendency to miss weak connections as was also observed in Figure 4.4. In total, we get excellent results using a triangular G_{obs} even though mutual information does not have the same properties as correlation. In particular, this is what we expected as we have only used $\rho_{i,i+1} \leq 0.9$, from the above investigation of the error.



(a) G_{obs} **(b)** G_{dir}

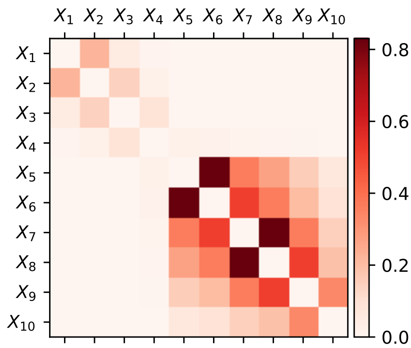


(c) G_{dir} from $t = 2.1 \cdot 10^{-2}$ **(d)** G_{dir} from $t = 4.6 \cdot 10^{-2}$

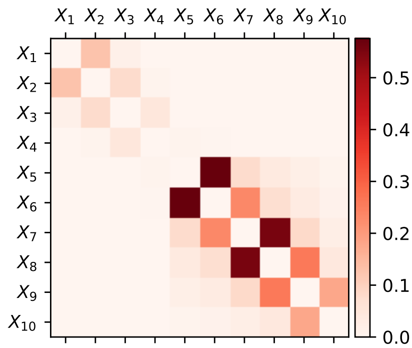
Figure 4.8: Using mutual information as the measure of similarity as well as assuming a topological order i.e. making G_{obs} strictly triangular as seen in (a) we almost perfectly infer G_{dir} as seen in (b) except for $[G_{dir}]_{6,9}$. Choosing cutoffs $t = 2.1 \cdot 10^{-2}$ (c) and $t = 4.6 \cdot 10^{-2}$ (d) it is clear that adjusting the threshold we can get a better result than using a symmetric G_{obs} with correlation.

Finally, we use the corresponding symmetric G_{obs} (rescaled such that the largest absolute eigenvalue of G_{dir} is 0.99) which results in G_{dir} and the graph using a threshold $t = 4.88 \cdot 10^{-2}$ shown in Figure 4.9. Again, we observe some bleeding on the more strongly connected sub-chain as with the symmetric G_{obs} using correlation in Figure 4.4. Again, we observe comparable results and note that increasing the threshold would disconnect X_3 and X_4 before removing the higher order effects.

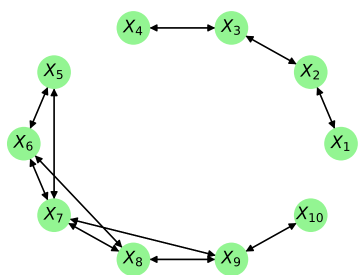
In conclusion, we have seen what errors can arise in the discovered network using both correlation and mutual information as the measure of association. Namely, long strongly connected chains seem to be a problem if one does not know a topological ordering of the variables, in which case these are heavily reduced as seen in Figure 4.3 and Figure 4.8. Thus, we proceed in the next section by considering a more complicated underlying (Gaussian) network to observe if other unwanted effects can occur and if a topological ordering is necessary if the network is not simply a path.



(a) G_{obs}



(b) G_{dir}



(c) G_{dir} as a graph

Figure 4.9: Using a symmetric G_{obs} containing the observed mutual information (a) we infer a G_{dir} (b) comparable to that if we had used correlation instead. Choosing the threshold $t = 4.88 \cdot 10^{-2}$ seem a good compromise between connectedness and density resulting in an almost identical discovered network structure to that of using a symmetric correlation G_{obs}.

4.2 Directed acyclic Gaussian graphs

In this section, we will expand on the results from the previous section by considering a more general structure. In particular, let $\mathcal{G}$ be a directed acyclic graph with nodes corresponding to variables from a random vector $\boldsymbol{X}$ with directed edges indicating direct dependencies. Such a DAG has a topological ordering. We shall index the variables 1 through d such that if the index of a variable is i, and j is the index of another element of the random vector $\boldsymbol{X}$, then $i < j$ implies there is no (directed) path from j to i. Note that since a topological ordering is not necessarily unique, we can not infer that there is a (directed) path from i to j or even if k is reachable from j (i.e. there exists a path from j to k) it does not follow that k is reachable from i. In Figure 4.10 a subset of such a DAG is shown with a possible labeling where $i < j$ and $k_m < k_n$ when $m < n$. It is then the weights along these directed edges which we will once again call G_{dir} that we wish to infer based on the transitive closure. As an example, from Figure 4.10, the transitive closure would result in an observed similarity between i and j although no 1 path i.e. single direct edge connects the two variables.

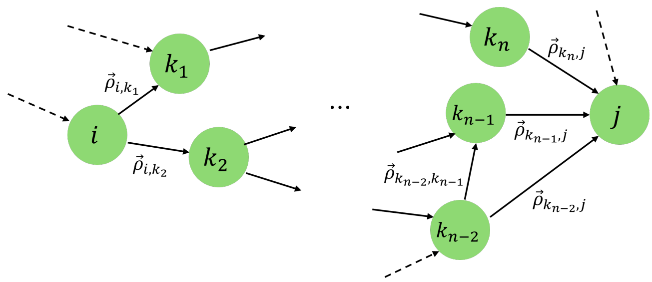


Figure 4.10: A general (linear) network represented as a DAG. The directed edge weights $\vec{\rho}_{k,l}$ specify how much the variable index k make up of variable l. Although i and j are not directly connected, multiple paths may exist between the two nodes, making the propagation of similarity more complex from that of a chain.

From the definition of the labels, it is clear that G_{dir} is once again strictly upper triangular as entries below the diagonal correspond to edges going from a random variable with an index i to another random variable with index j such that $i > j$ which is clearly a contradiction. Also, the diagonal elements are 0 as there can not be any loops in DAGs.

Similarly to the definition of (Gaussian) chains, based on d independent (or even just pairwise uncorrelated) random variables Z_i, we can define a general network of random variables X_i based on $\boldsymbol{Z}$ in the following way

$$\begin{aligned}
X_1 &= Z_1 \\
X_2 &= \vec{\rho}_{1,2} X_1 + \sqrt{1-\vec{\rho}_{1,2}^2} Z_2 \\
X_3 &= \vec{\rho}_{1,3} X_1 + \vec{\rho}_{2,3} X_2 + c_3 Z_3 \\
&\vdots \\
X_d &= \sum_{k<d} \vec{\rho}_{k,d} X_k + c_d Z_d
\end{aligned} \tag{4.4}$$

where c_i is chosen such that $\text{Var}\left(X_i\right) = 1$. This is done to make the analysis later on simpler as then covariance and correlation are equal and $\vec{\rho}_{i,j}$ becomes the *direct* correlation between the variables indexed i and j as shown in Figure 4.10. Of course, for the variance of each random variable to be 1 there must be some constraints on the chosen $\vec{\rho}_{i,j}$ such that neither one of them can exceed 1 in absolute value. Furthermore, consider the following bound on the variance of X_i assuming c_k for $k < i$ has been chosen such that $\text{Var}\left(X_k\right) = 1$.

$$\begin{aligned}
\text{Var}\left[X_i\right] &= \sum_{k<i} \vec{\rho}_{k,i}^2 + 2 \sum_{k<l<i} \vec{\rho}_{k,i} \vec{\rho}_{l,i} \text{Cov}\left[X_k, X_l\right] + c_i^2 \\
&\leq \sum_{k<i} \vec{\rho}_{k,i}^2 + 2 \sum_{k<l<i} \vec{\rho}_{k,i} \vec{\rho}_{l,i} + c_i^2 \\
&= \left(\sum_{k<i} \rho_{k,i}\right)^2 + c_i^2
\end{aligned} \tag{4.5}$$

where we have used that Z_i is uncorrelated with X_k for $k < i$ and that the covariance between variables with variance 1 is at most 1 to obtain the inequality. Hence, choosing the sum of the ingoing edges to be at most 1 for every node ensures that the constants c_i for $i \in \{2, \ldots, d\}$ exist to make the variance of each X_i 1. This, we will use in the following example to easily build a network such that $\vec{\rho}_{i,j}$ is the pure correlation.

However, before constructing an example and using bot correlation and mutual information we must determine the theoretical G_{obs} for both cases. To do this, we shall consider the (i, j) element of G_{obs} when using correlation as a measure of similarity and later use mutual information based on these correlations and Proposition 4.1 in the case of $\boldsymbol{Z}$ being a Gaussian random vector. To calculate $\left[G_{obs}\right]_{i,j}$ we shall consider the immediate predecessors to node j in the graph $\mathcal{G}$ corresponding to Equation 4.4. The immediate predecessors or *in-neighbors* of a node j is denoted $N^-\left(X_j\right)$ or in shorthand notation N_j^-. An example of this is shown in Figure 4.11 where the in-neighbors of j have been marked in red. With this notation, we proceed with the computation of the (i, j) entry of G_{obs} which is the covariance between X_i and X_j when $i < j$ and 0 elsewhere.

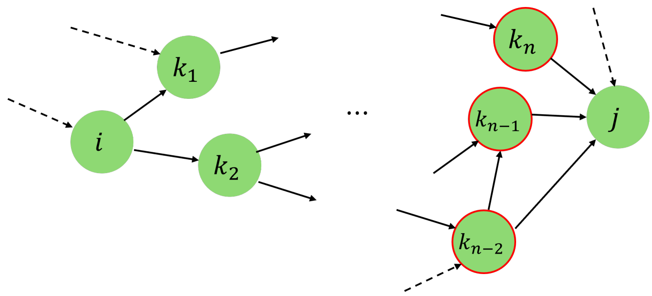


Figure 4.11: For node j, the set N_j^- is illustrated with red borders. It is exactly the set of nodes going directly into j. We note that an in-neighbor l of in-neighbor k of node j can also be an in-neighbor of j i.e. l can influence both k and j whilst k also directly influenced j. It is in particular these direct dependencies we want to be sure of as their existence makes the network more complex but failing to discover these can lead to a significant reduction in prediction accuracy.

$$\begin{aligned}
[G_{obs}]_{i,j} &= \mathrm{Cov}\left[X_i, \sum_{k\in N_j^-} \vec{\rho}_{k,j}\, X_k + c_j Z_j\right] \\
&= \mathrm{Cov}\left[X_i, \sum_{k\in N_j^-} \vec{\rho}_{k,j}\, X_k\right] \\
&= \sum_{k\in N_j^-} \vec{\rho}_{k,j} \mathrm{Cov}\left[X_i, X_k\right] \\
&= \sum_{k\in 1}^{j-1} \vec{\rho}_{k,j} \mathrm{Cov}\left[X_i, X_k\right] \\
&= \vec{\rho}_{i,j} + \sum_{k\in 1}^{d} \vec{\rho}_{k,j}\, [G_{obs}]_{i,k}
\end{aligned} \tag{4.6}$$

For the fourth equality, we have used that $\vec{\rho}_{k,j} = 0$ whenever $k \notin N_j^-$ which again for the fifth equality holds for any $k \geq j$. Furthermore, since $[G_{obs}]_{i,i} = 0$ we need to add $\vec{\rho}_{i,j}$ to make the final equality hold. The above can also be expressed as a matrix equation, namely

$$G_{obs} = G_{obs} G_{dir} + G_{dir}$$

Hence, as G_{dir} is strictly upper triangular thus making $I - G_{dir}$ invertible, we can directly express G_{obs} in terms of G_{dir}. We find that

$$G_{obs} = G_{dir}\left(I - G_{dir}\right)^{-1}$$

which we recognize as Equation 3.4 hence also for a general network (and not just a chain), using correlation and knowing/assuming a topological order of the random variables we can perfectly rediscover G_{dir} from G_{obs}.

With the above, we then define an example Gaussian network with the following weights and shown in Figure 4.12 to get a better understanding of this example hopefully should reappear after deconvolution using both correlation and mutual information respectively.

$$\begin{aligned} \vec{\rho}_{1,2} &= 0.7, & \vec{\rho}_{5,6} &= 0.5, & \vec{\rho}_{2,7} &= 0.3 \\ \vec{\rho}_{6,7} &= 0.3, & \vec{\rho}_{6,8} &= 0.7, & \vec{\rho}_{4,9} &= 0.3 \\ \vec{\rho}_{8,9} &= 0.3, & \vec{\rho}_{7,10} &= 0.4, & \vec{\rho}_{9,10} &= 0.2 \end{aligned} \tag{4.7}$$

In particular, from Equation 4.5 and Figure 4.6 and Figure 4.7, we suspect that the bleeding effects observed for the Gaussian chain won't appear to the same extent in this case.

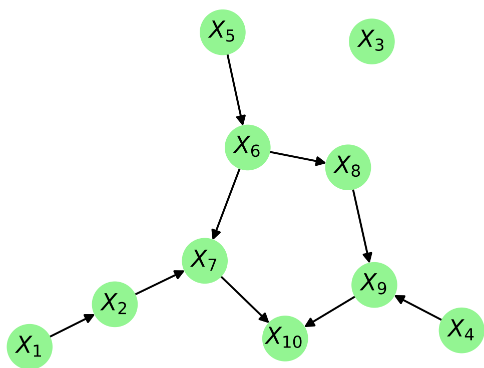
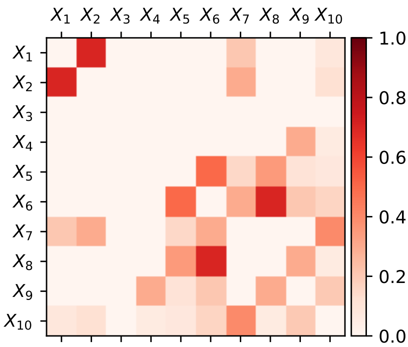


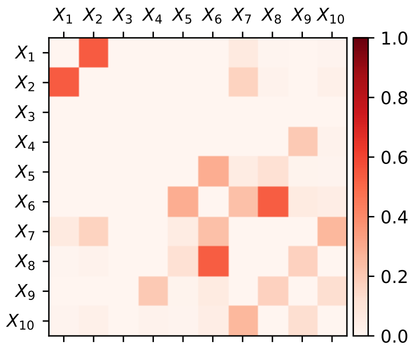
Figure 4.12: The graph defined in Equation 4.7. Note that X_3 is neither influenced nor influences any other variable. It is of course in our interest to accurately tell if X_3 should be considered if we try to infer a probability distribution on e.g. X_{10} given observations of the other variables.

Applying the deconvolution algorithm, we obtain the results in Figure 6.14 which trivially, from the above analysis on G_{obs}, results in a perfect reconstruction of the network. If instead, we do not assume a topological structure, we can also recover the structure, although we need to tune the threshold as can be seen from Figure 4.13. Tuning the α and β did not have much of an effect.

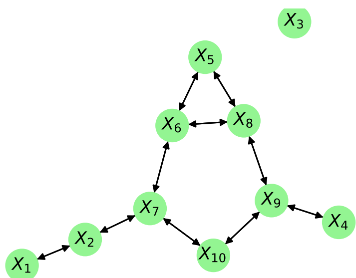
Actually, decreasing β seemed to worsen the results which is also in line with our expectations as choosing smaller β skews the effects of higher-order interactions. Thus, it is primarily the threshold that we want to tune in this case, and choosing $t = 1.18 \cdot 10^{-1}$ we accurately infer the network structure contrary to the results from the Gaussian chain. However, we still observe second-order effects i.e. the edge between X_5 and X_8 which was also the case in Figure 4.4



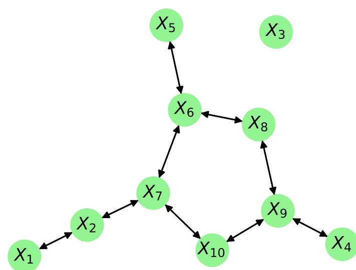
(a) G_{obs}



(b) G_{dir}



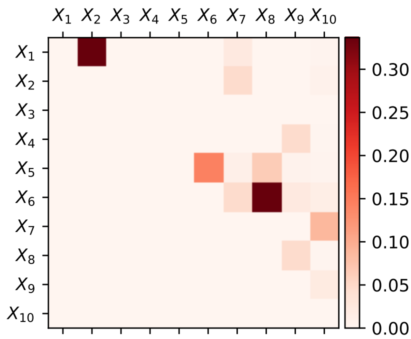
(c) G_{dir} from $t = 6.4 \cdot 10^{-2}$



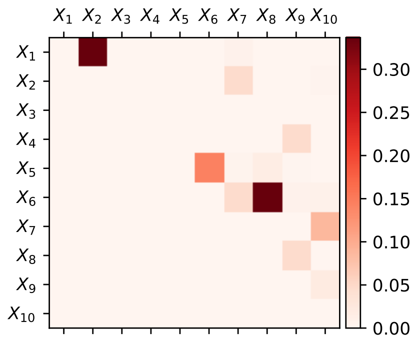
(d) G_{dir} from $t = 1.18 \cdot 10^{-1}$

Figure 4.13: Not knowing the topological structure and thus using a symmetric G_{obs} (a) we obtain the G_{dir} in (b). Clearly, there is some bleeding, but choosing the threshold $t = 1.18 \cdot 10^{-1}$ we can accurately rediscover the network structure up to a direction on the edges. As with the previous example of Gaussian chains, we observe some tendency to inaccurately filter out second order effects as can be seen in (c) where X_5 and X_8 is connected.

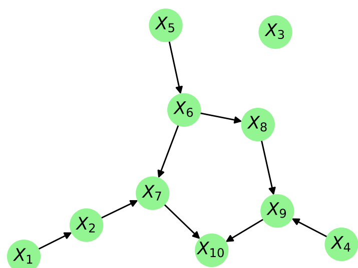
Finally, before continuing with results regarding the different methods for estimating mutual information, we present the results from above using mutual information instead of correlation as the measure of similarity. Namely, once again assuming the topological order such that G_{obs} is strictly upper triangular and hence no need for rescaling we get the results shown in Figure 4.14. As expected, we observe on-par performance compared to using correlation. Only the edge from 5 to 8 being almost as strong as 9 to 10 could be a problem i.e. choosing a threshold a little larger than $t = 1.7 \cdot 10^{-2}$ (which is quite small and has been used for Figure 4.14(c)) would have resulted in an edge from X_5 to X_8. Hence, in a real-world example, we might have chosen to either leave out both edges which depending on the scenario may or may not be an acceptable error, or include both of them.



(a) G_{obs}



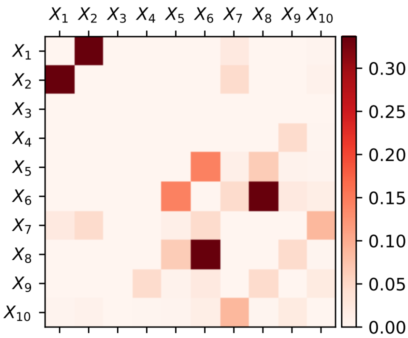
(b) G_{dir}



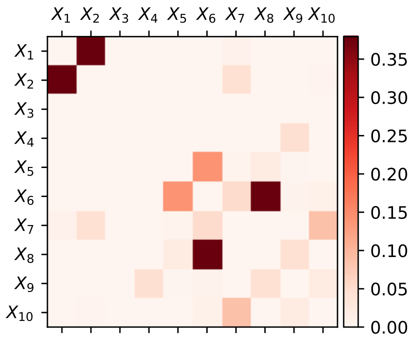
(c) G_{dir} as a graph

Figure 4.14: Using mutual information instead of correlation results in G_{obs} shown in (a). The non-linear map from correlation to mutual information only effects the resulting G_{dir} a little as shown in (b) when comparing to the $\vec{\rho}_{i,j}$ from Equation 4.7. Choosing the relatively small threshold $t = 1.7 \cdot 10^{-2}$ results in a perfect reconstruction of the graph structure.

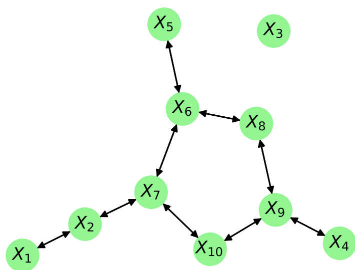
Furthermore, using a symmetric G_{obs} instead i.e. no assumption on topology does not seem to have much of an effect as seen from Figure 4.15. Although there still is a small weight on the edge from X_5 to X_8, by choosing the threshold $t = 1.96 \cdot 10^{-2}$ we can accurately construct the true network structure.



(a) G_{obs}



(b) G_{dir}



(c) G_{dir} as a graph

Figure 4.15: Using a symmetric G_{obs} instead of an upper triangular G_{obs} result in near identical G_{dir} in terms of relative weights on the edges. Namely, the G_{dir} shown in (b) seem to be almost a scaled version of the (reflected) G_{dir} derived from a triangular G_{obs}. Thus, as (c) also shows, we can accurately infer the structure of the network using a threshold $t = 1.96 \cdot 10^{-2}$.

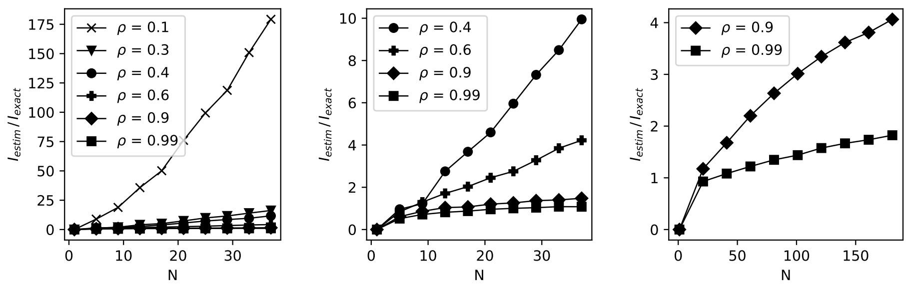
In conclusion, we observe a useful property of more general networks that for both mutual information and correlation, the additional assumption of the topological order does not have much of an effect in these cases contrary to what we observed for Gaussian chains and linear chain models in general, when using correlation.

4.3 CE computation

Having discussed the strengths and weaknesses of Algorithm 2, we now turn our attention to Algorithm 1. Namely, in this section, we shall discuss the different methods from Section 3.4 and how they perform on two examples. Once again, we shall base our results on two examples. The first is a simple case, where we shall see what to be aware of when initially the observations are transformed through estimated distribution functions as well as how accurate the different methods for estimating the Copula entropy i.e. mutual are. Continuing from the first example, we shall once again consider the network from Section 4.2 specified by Equation 4.7. In particular, we will see how well the combined framework performs on an example we have already seen to be quite solvable if one uses accurate estimates of the mutual information that we previously calculated theoretically.

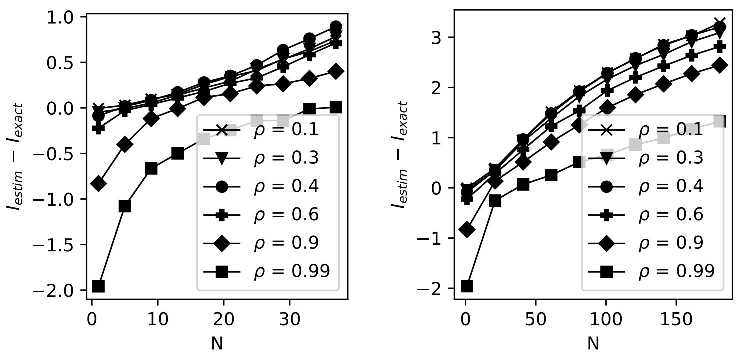
4.3.1 Spline and KDE based CE estimation

Before the first example, we shall discuss the problem with the spline-based method and using histograms in general. Namely, we shall first see that if one were to just simply use a raw binning approach, the number of bins N influences the estimate a lot, and no number of bins seems to perform well in all cases. Namely, let $\boldsymbol{X}$ be a bivariate Gaussian with correlation ρ, then the Copula density looks as in Figure 4.21(c). In particular, we notice the peaks at $(0, 0)$ and $(1, 1)$ from which most of the mutual information originates. Now, simulating $n = 400$ observations from the joint distribution and transforming to the unit square through the marginal distribution function for varying correlations ρ, we can compare the estimated mutual information I_{estim} using the results from Subsection 3.2.3 to the true mutual information given by $I_{exact} = -\frac{1}{2}\log\left(1 - \rho^2\right)$. The results are shown in Figure 4.16 where we report the relative size of the estimate and the exact mutual information and in Figure 4.17 where the difference is reported. From Figure 4.16, we might choose $N \approx 10$ as in [6] however for large correlations, we drastically underestimate mutual information. Increasing the number of bins to e.g. $N = 25$ corrects this error for large correlations, while small mutual information for small correlations remains relatively small. However, when deconvolving we would preferably want a more precise way of estimating the mutual information.



(a) (b) (c)

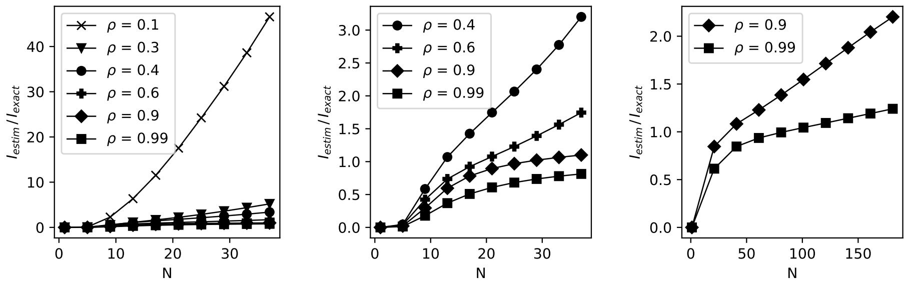
Figure 4.16: Relative error when estimating mutual information for different bivariate Gaussian distributions with varying bin counts N. Information originating from small correlations are vastly overestimated.



(a) (b)

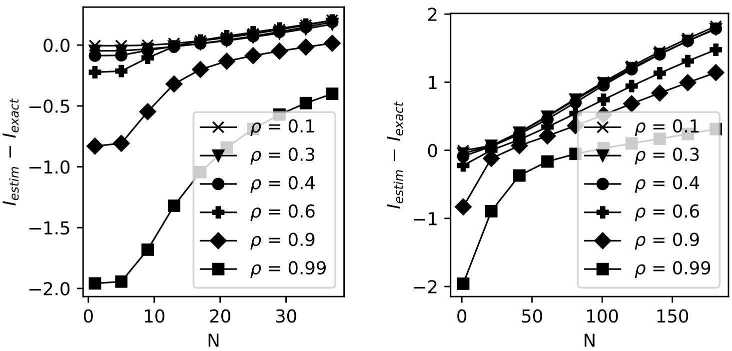
Figure 4.17: Error for mutual information estimation for varying correlations. Contrary to the relative error, we see that it is the mutual information from large corrlations that is the most error-prone for small bin counts.

We thus proceed with using the B-spline approach. Similar to the above results, in Figure 4.18 and Figure 4.19, we observe that the B-spline approach is prone to the same errors as the raw binning approach. However, from Figure 4.19 we see that the error are smaller for the B-spline-based approach for large N but also that a better choice for the number of bins is around $N = 50$ contrary to the results of [6].



(a) (b) (c)

Figure 4.18: Relative error of mutual information estimates using B-splines. Comparing to the relative error of the raw histogram based approach, we obtain relative error much smaller.



(a) (b)

Figure 4.19: The actual error for mutual information estimation using the B-splines approach. For $n = 400$ observations, performance is comparable to that of the raw histogram based approach although a larger bin count is preferred.

The results for M-splines are shown in Figure 6.11 and Figure 6.11 we observe comparable performance except perhaps for a better estimation of mutual information for large correlations which is what we would expect from our discussion in Subsection 3.4.2.

The problem we observe with the above methods is that when mutual information is large, most of the mutual information comes from small domains at $(0, 0)$ and $(1, 1)$. Hence, to calculate the mutual information to a high precision, we need many bins. However, with many bins, the estimate becomes more noisy as the support of each spline shrinks with $\frac{1}{N}$. However, as seen from Figure 4.20, if we can accurately estimate the Copula density function from observations, we can compute the mutual information perfectly by increasing the fineness of the integral approximation. In particular, the results below were obtained through the theoretical Copula density function evaluated at the bin centers $\left(\frac{2i-1}{2N}, \frac{2j-1}{2N}\right)$ for $i, j \in \{1, \dots, N\}$. We see that this simple approximation with $N = 1000$ is good for correlations up to $\rho = 0.99$. However, due to numerical limitations, we shall use $N = 500$ and only $n = 400$ observations to evaluate the performance of the KDE from the previous chapter. We note that a more memory-efficient implementation is possible by splitting up the computation into multiple parts as the problem with many observations and bins is that the computation is $\mathcal{O}\left(N^2 n\right)$ time and memory if done all at once.

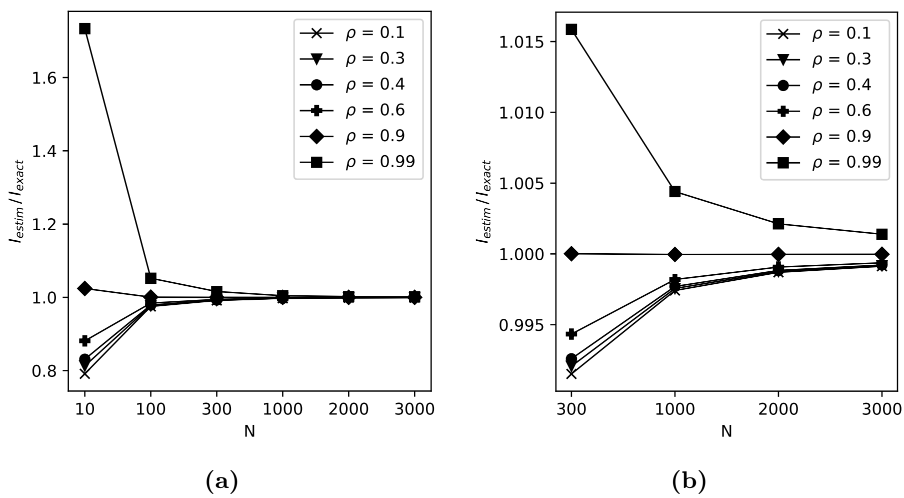


Figure 4.20: Relative error of mutual information based on the true Copula density. We observe that for $N \geq 300$ the relative error is almost negligible.

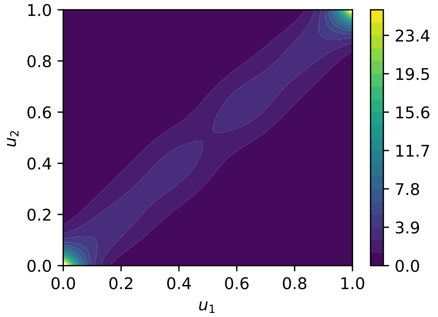
In Table 4.1, we have used the boundary-corrected KDE from Subsection 3.4.4 to first estimate the Copula density function and then estimate the mutual information from this. We note that by default, the bandwidth is chosen to be $h = h^{Scott} = 0.085$ as the marginals are approximately uniform and hence the variance is constant.

From Table 4.1, we observe relatively low variance, and in general, we compute the mutual information to a higher accuracy than both B-splines and M-splines. Thus, in the following example, we will only consider this method for estimating

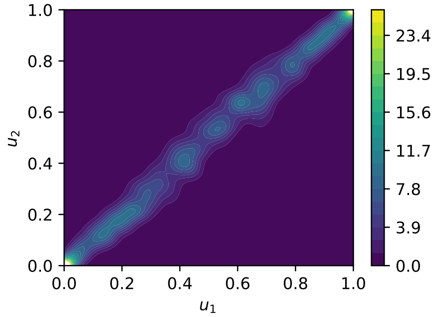
ρ	h	mean error	variance
0.1	h^{Scott}	0.01583	$5.3446 \cdot 10^{-5}$
0.3	h^{Scott}	0.02302	$3.7957 \cdot 10^{-4}$
0.4	h^{Scott}	0.006898	$3.2655 \cdot 10^{-4}$
0.6	h^{Scott}	0.007803	$1.0027 \cdot 10^{-3}$
0.9	h^{Scott}	-0.1844	$4.4478 \cdot 10^{-4}$
0.99	h^{Scott}	-1.007	$1.6328 \cdot 10^{-4}$
0.99	$0.3\,h^{Scott}$	-0.3468	$1.0616 \cdot 10^{-3}$

Table 4.1: Estimating mutual information based on $n = 400$ samples from different bivariate Gaussians using the boundary corrected KDE. Repeating the simulations 10 times, we obtain average errors and variance of the estimate. In particular, the estimates are very certain for $n = 400$. However, as for the spline based methods, large correlations result in underestimating the mutual information. This can however be corrected by tuning the bandwidth as is also observed in the above.

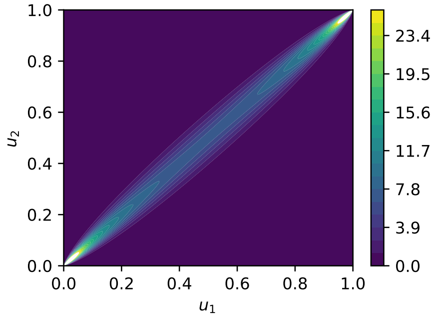
the mutual information between pairs of variables. In Figure 4.21 we have shown the estimated density and the theoretical copula. We observe that indeed the method accurately estimates the Copula density, although we note that the concept of a local bandwidth as discussed in Subsection 3.4.4 is likely to improve on the results as the peaks at $(0,0)$ and $(1,1)$ does not quite resemble those of the theoretical Copula density. In particular, we observe that reducing the bandwidth improves the estimate, observed by the improved resemblance with the theoretical Copula density. Although this is at the cost of undersmoothing on the interior. Indeed, a K-means-based estimator of the bandwidth h (as discussed in Subsection 3.4.4) could work well as the mean distance near $(0,0)$ and $(1,1)$ is very small compared to the interior. However, we note that as long as observations are not close to perfectly correlated, the KDE-based method performs quite well. This, we shall also see in the following section where we couple the above discussions on mutual information estimation with the deconvolution algorithm.



(a)



(b)



(c)

Figure 4.21: Estimated Copula densities (a) and (b) compared to the theoretical Copula density (c) for $\rho = 0.99$. Using a smaller bandwidth we are able to capture the peaks at $(0, 0)$ and $(1, 1)$ more accurately but at the cost of undersmoothing the interior.

4.3.2 Exponentiated multivariate Gaussian

In this section, we will consider a small example, testing the combined algorithm. In particular, we shall discover what to be aware of when data is not easily transformed by the inverse distribution function. Let us consider a simple case with $\mathbf{Y} = e^{\mathbf{X}}$ (element-wise exponentiation) where $X \sim \mathcal{N}(\mathbf{0}, \Sigma)$ where

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0.9\sigma_1\sigma_2 & 0 \\ 0.9\sigma_1\sigma_2 & \sigma_2^2 & 0 \\ 0 & 0 & \sigma_3^2 \end{bmatrix} = \operatorname{diag}(\boldsymbol{\sigma}) \begin{bmatrix} 1 & 0.9 & 0 \\ 0.9 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \operatorname{diag}(\boldsymbol{\sigma}) \tag{4.8}$$

In particular, in terms of Equation 4.4, we have that for $\boldsymbol{X}$, $\vec{\rho}_{1,2} = 0.9$. From Corollary 3.2.1, it is clear that the (symmetric) mutual information matrix G_{obs} is the same as of *boldsymbolX* and hence by Proposition 4.1 is as follows

$$G_{obs} = \begin{bmatrix} 0 & -\frac{1}{2}\log\left(1 - \vec{\rho}_{1,2}^2\right) & 0 \\ \log\left(1 - \vec{\rho}_{1,2}^2\right) & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \approx \begin{bmatrix} 0 & 0.83037 & 0 \\ 0.83037 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

In particular, independent of the choice of $\boldsymbol{\sigma}$, we should observe an estimated $\hat{G}_{obs}$ close to this. We choose the following three cases for the choice of $\boldsymbol{\sigma}$.

$$\boldsymbol{\sigma} = (0.07, 0.3, 0.9), \quad \boldsymbol{\sigma} = (1, 1, 1), \quad \boldsymbol{\sigma} = (1, 2, 3)$$

To draw from this distribution, one can use built-in functions or the Cholesky factorization of the correlation matrix to generate correlated variables from 3 independent standard normal distributions and then scale with the chosen standard deviation to generate samples from all three cases based on the same seed. Once again, we shall only use 400 samples as this resembles the number of observations in the pharmaceutical dataset, which we shall treat in Section 4.4. We note that a KDE has been used to approximate the distribution function. We will see shortly why this is not always a good idea.

For $\boldsymbol{\sigma} = (0.07, 0.3, 0.9)$, Algorithm 1 returns the following

$$\hat{G}_{obs} = \begin{bmatrix} 0 & 0.8618 & 0.07889 \\ 0.8618 & 0 & 0.07880 \\ 0.07889 & 0.07880 & 0 \end{bmatrix} \tag{4.9}$$

Similarly, for $\boldsymbol{\sigma} = (1, 1, 1)$:

$$\hat{G}_{obs} = \begin{bmatrix} 0 & 1.066 & 0.1667 \\ 1.066 & 0 & 0.1825 \\ 0.1667 & 0.1825 & 0 \end{bmatrix} \tag{4.10}$$

Finally, for $\boldsymbol{\sigma} = (1, 2, 3)$:

$$\hat{G}_{obs} = \begin{bmatrix} 0 & 1.797 & 1.549 \\ 1.797 & 0 & 2.145 \\ 1.5492 & 2.145 & 0 \end{bmatrix} \tag{4.11}$$

Note that for this example, we have chosen $h = \frac{1}{3}h^{Scott}$ to correct for the behavior observed in Table 4.1 when computing the mutual information between X_1 and X_2.

For $\boldsymbol{\sigma} = (0.07, 0.3, 0.9)$ we observe the most resemblance to the theoretical G_{obs}. For the latter two examples, $\boldsymbol{\sigma} = (1, 1, 1)$ and $\boldsymbol{\sigma} = (1, 2, 3)$, we see a completely different result and immediately suspect that there must be some errors either in the algorithm or the underlying assumptions of the algorithm. Investigating the partial results of Algorithm 1 we immediately see a flaw in the supposedly uniform variables U_i as shown in figure Figure 4.22 for $\boldsymbol{\sigma} = (1, 2, 3)$, which through a Kolmogorov Smirnov test are indeed significant (Table 4.2).

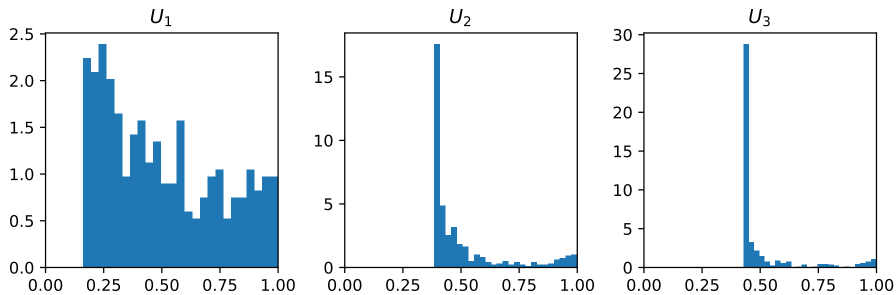


Figure 4.22: The samples transformed using $U_i = F_i(Y_i)$ for $\boldsymbol{\sigma} = (1, 2, 3)$. These should be uniformly distributed, but clearly this is not the case for neither of them, as we have demonstrated in Table 4.2.

	U_1	U_2	U_3
D_n	0.16512	0.38354	0.42764
p-value	0	0	0

Table 4.2: Kolmogorov Smirnov test result based on 400 samples for $\boldsymbol{\sigma} = (1, 2, 3)$. It is clear that all samples are statistically significant.

Before handling this, the non-uniformity of U_1, U_2 and U_3 in Figure 4.22 is likely also present in the case when $\boldsymbol{\sigma} = (1, 1, 1)$. Indeed, Figure 4.23 shows that this is indeed the case which is further shown statistically significant in Table 4.3.

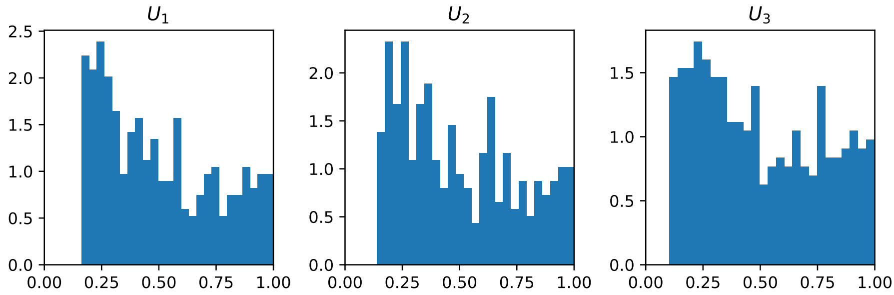


Figure 4.23: The samples transformed using $U_i = F_i(Y_i)$ for $\boldsymbol{\sigma} = (1, 1, 1)$.

	U_1	U_2	U_3
D_n	0.16511	0.14672	0.10561
p-value	0	0	$2.382 \cdot 10^{-4}$

Table 4.3: Kolmogorov Smirnov test result based on 400 samples for $\boldsymbol{\sigma} = (1, 1, 1)$. Once again, all the transformed samples are shown statistically significant.

Finally, for the sake of completeness, $\boldsymbol{\sigma} = (0.07, 0.3, 0.9)$ is also shown in Figure 4.24 and seems very reasonable, except for U_3 which again, is shown through the Kolmogorov Smirnov test in Table 4.4 to be significant.

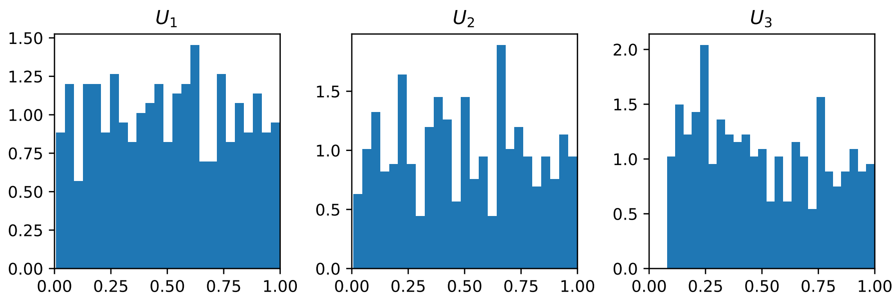


Figure 4.24: The samples transformed using $U_i = F_i(Y_i)$ for $\boldsymbol{\sigma} = (0.07, 0.3, 0.9)$.

From the above examples, it seems that the larger the variance, the worse the uniforms turn out. From Figure 4.25, we see that this is primarily due to a poor fit of the KDE, where we have zoomed in on the interval $[-200, 200]$ which contains 96.2% of observations. In particular, the peak neat $y_3 = 0$ is not

	U_1	U_2	U_3
D_n	0.029036	0.029026	0.085611
p-value	0.88427	0.88454	0.0052791

Table 4.4: Kolmogorov Smirnov test result based on 400 samples for $\boldsymbol{\sigma} = (0.07, 0.3, 0.9)$.

captured by the KDE. The poor fit is primarily due to the use of Scott's Rule which in this case overshoots the optimal bandwidth by a lot. The poor fit also explains the high concentration of U_3 around 0.5 in Figure 4.22 as only 54.5% of the probability mass for the KDE lies above 0.

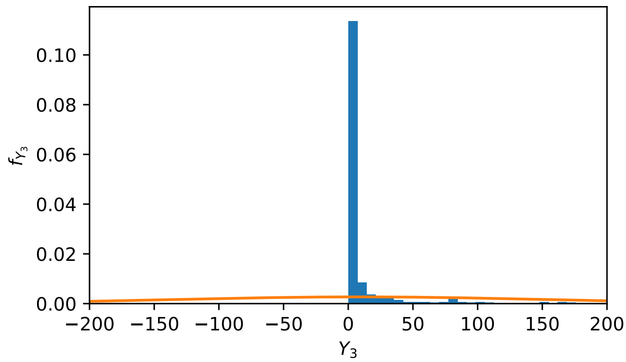


Figure 4.25: Inspecting the fit of the KDE on Y_3, we indeed observe that it is quite poor. The peak near 0 is not captured. Thus, either Y_3 has to be transformed through some strictly positive function or another estimate of the distribution function (such as the empirical distribution function) need to be applied.

By Corollary 3.2.1, we can get rid of these numerical issues by transforming Y_i using e.g. $\log(\cdot)$ or $(\cdot)^p$ for $p > 0$ to even out the observations more. As the first simply inverts the initial transformation of $\boldsymbol{X}$, we choose the latter as a more interesting case. In particular, choosing $p < 1$ will result in a more even distribution. In the following, $p = 1/10$ has been used to transform $\boldsymbol{Y}$, resulting in $\boldsymbol{Y}^p$, prior to running Algorithm 1. The resulting samples $u_i^{(j)}$ are shown in Figure 4.26 along with statistical test for uniform distribution in Table 4.5

The resulting $u_i^{(j)}$ are now no longer significant and indeed the KDE fits much better as seen in Figure 4.27. Furthermore, the estimated $\hat{G}_{obs}$ in Equation 4.12 is similar to the one obtained from $\boldsymbol{\sigma} = (0.07,\ 0.3,\ 0.9)$ with the notable difference $I(Y_1, Y_3)$ and $I(Y_2, Y_3)$ are closer to the true value, 0.

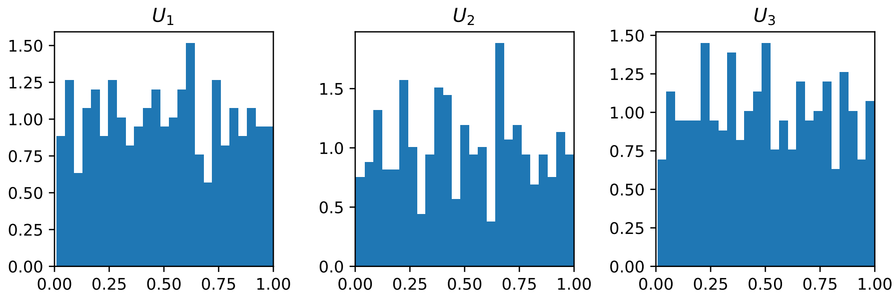


Figure 4.26: The resulting observations of $\boldsymbol{U}$ after a power transformation with $p = 1/10$ has been applied to $\boldsymbol{Y}$.

	U_1	U_2	U_3
D_n	0.0061099	0.0061435	0.0073148
p-value	0.84838	0.84368	0.65690

Table 4.5: Based on 400 samples of $\boldsymbol{Y}^p$ with $\boldsymbol{\sigma} = (1, 2, 3)$ neither are statistically significant.

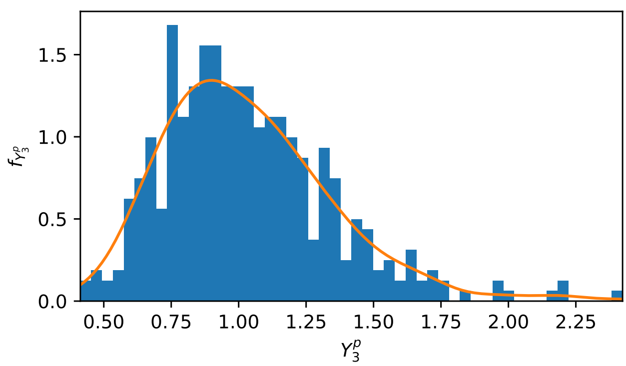
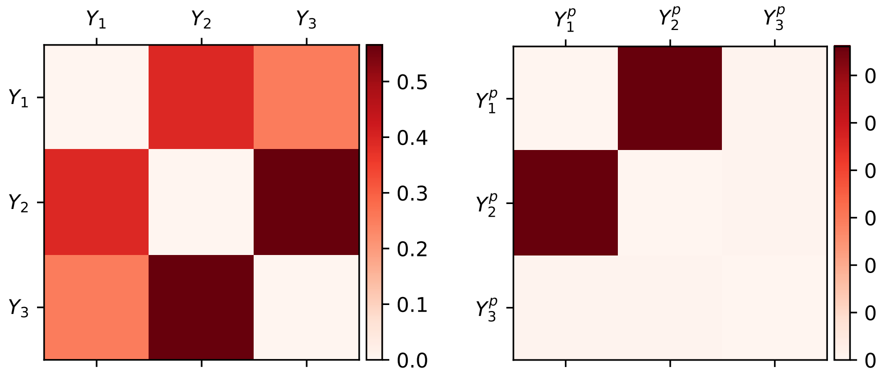


Figure 4.27: The KDE fit on Y_3^p with $p = 1/10$. Now, the KDE fits the samples much better, leading to non-significant results regarding tests of uniformity.

$$\hat{G}_{obs} = \begin{bmatrix} 0 & 0.8629 & 0.01799 \\ 0.8629 & 0 & 0.01886 \\ 0.01799 & 0.01886 & 0 \end{bmatrix} \tag{4.12}$$

To illustrate the importance of the above discussion, although it should already be quite clear from the differences in $\hat{G}_{obs}$, we have in Figure 4.28, shown the resulting $\hat{G}_{dir}$ after using Algorithm 2. Namely, we infer a relation between Y_2 and Y_3 especially, which we know to be independent of each other from Equation 4.8.



(a) **(b)**

Figure 4.28: G_{dir} resulting from 400 samples from multi variate Gaussian with $\boldsymbol{\sigma} = (1, 2, 3)$ in **(a)** with raw samples from $\boldsymbol{Y}$ and in **(b)** the transformed data corresponding to $\boldsymbol{Y}^p$.

We note that from Section 6.6, we can compute a CI of the absolute value of the correlation (under the assumption the mutual information was computed from a bivariate Gaussian or a strictly increasing transform of such variables) based on [30]. In particular, computing confidence intervals for the absolute correlation from each mutual information from $\hat{G}_{0bs}$ in Equation 4.12, we see that the estimated $I\,\widehat{(Y_1, Y_2)}$ is not significantly different from the theoretical mutual information while the remaining are (i.e. between (Y_1, Y_3) and (Y_2, Y_3) respectively).

Thus, in the above, we have shown that the assumption of uniform marginals does not always hold when the data has heavy tails. Namely, the key assumption for Algorithm 1 is that we are doing computations on a Copula density such that the entropy of the marginals $h\,(Y_i)$ and $h\,(Y_j)$ is 0. Thus, when this assumption does not hold, the algorithm does estimate the mutual information correctly. However, we can fix this by adding the marginal entropies to the algorithm using Lemma 3.6. However, for the KDE to estimate the marginal entropies well, we still would not want a tail too heavy. What a too-heavy tail is, is a bit ambiguous, but using the modified algorithm on $\boldsymbol{\sigma} = (1,\, 2,\, 3)$ without the power transformation we obtain the following $\hat{G}_{obs}$, which is of course not as good as the above $\hat{G}_{obs}$, but still closer than the original estimate, in Equation 4.11

$$\hat{G}_{obs} = \begin{bmatrix} 0 & 0.6263 & 0.02288 \\ 0.6263 & 0 & 0.01688 \\ 0.02288 & 0.01688 & 0 \end{bmatrix}$$

In the following section, we shall apply our learning from this section to the more complicated yet fully controlled example, introduced in Section 4.2. In particular, we shall observe that without any manual tuning of the bandwidths h, we are able to accurately infer the causal structure.

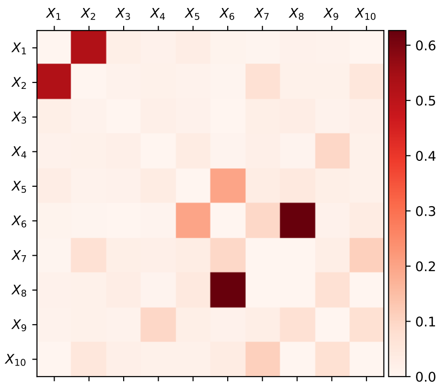
4.3.3 Gaussian network revisited - Application of complete framework

In this section, we will redo the example from Section 4.2 where the underlying causal structure was defined in Equation 4.7. However, this time, we shall first simulate $n = 400$ observations from the joint distribution and then estimate the Copula density based on these observations instead of using the theoretical Copula densities (i.e. the exact mutual information) along with the learnings from the previous sections. In particular, we shall observe that the algorithm for estimating the mutual information between pairs of random variables performs well enough for us to recover the structure perfectly. As the largest (direct) correlation is 0.7, we expect from Table 4.1 that the mutual information will be accurately estimated.

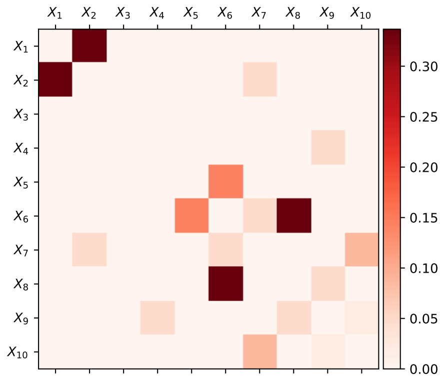
In Figure 4.29, we have summarized the results using a symmetric G_{obs}. In particular, we observe that the estimated $\hat{G}_{dir}$ is nearly identical to the true G_{dir} (from Equation 4.7). Furthermore, choosing the relatively small threshold $t = 0.06$ we recover the true causal structure although the entries of $\hat{G}_{dir}$ appear to be a factor 2 as large as the true G_{dir}. If we instead assume a topological structure, resulting in a triangular $\hat{G}_{obs}$, this deviation is alleviated as seen in Figure 4.30.

From this short example of combining Algorithm 1 and Algorithm 2, we observe that indeed the methodology proposed can at least for networks as defined in Section 4.2 recover the causal structure. Furthermore, from Corollary 3.2.1, we note that the same structure would be inferred if instead the variables had been transformed by a strictly increasing function. This is particularly important in the following section where we shall use the framework on the data introduced in chapter 2.

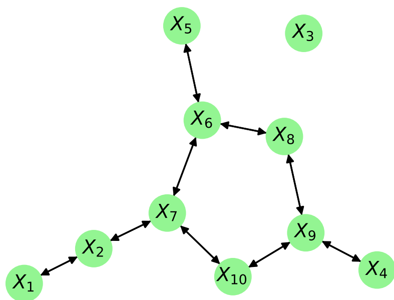
As an example, suppose that a network is defined as in Section 4.2, but we only observe $\boldsymbol{Y}$ where $Y_i = f(X_i)$ with f being strictly positive. Then, the mutual information between Y_i and Y_j is the same as between X_i and X_j which we from this example is accurately deconvolved.



(a) $\hat{G}_{dir}$

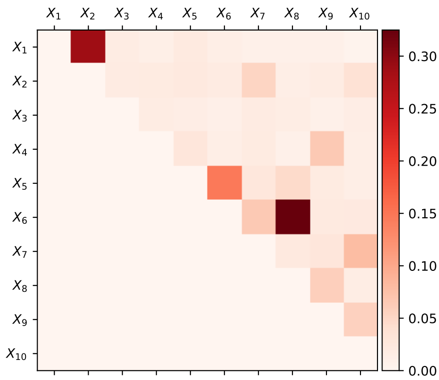


(b) G_{dir}

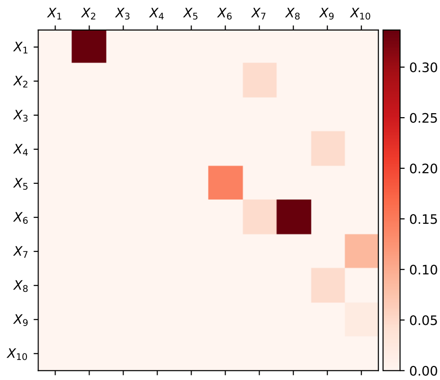


(c) $\hat{G}_{dir}$ as a graph

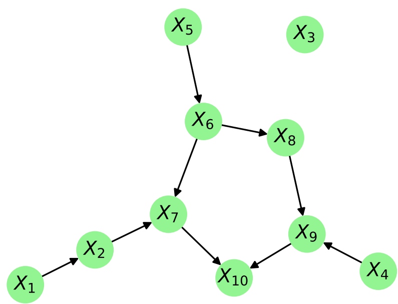
Figure 4.29: Combining the method for estimating the mutual information and algorithm for deconvolving the network (using a symmetric G_{obs}) we observe near-optimal results. In particular, the structure of $\hat{G}_{dir}$ (a) is very alike to $\hat{G}_{dir}$ (b). This is also shown in (c), where the $\hat{G}_{dir}$ is represented as a graph using $t = 0.06$, and we recover the original graphical structure. We observe that the entries of $\hat{G}_{dir}$ are almost twice the size of G_{dir}.



(a) $\hat{G}_{dir}$



(b) G_{dir}



(c) $\hat{G}_{dir}$ as a digraph

Figure 4.30: When assuming a topological order of the random variables, we observe that the inferred $\hat{G}_{dir}$ (a) is much more alike the true G_{dir} (b). Furthermore, using a threshold $t = 0.05$ we recover the true causal structure, removing the noise entries in $\hat{G}_{dir}$.

4.4 Pharmaceutical data deconvolution

Finally, we turn our attention to the pharmaceutical production data introduced in chapter 2. Namely, we have at this point used the methodology discussed in chapter 4 both to estimate mutual information, to deconvolve Gaussian networks and chains based on analytical expressions for the correlation between variables, and finally in Subsection 4.3.3 where we have combined Algorithm 1 and Algorithm 2 to infer the causal structure of a 10-dimensional Gaussian network.

In particular, we saw that the methodology combined produced very accurate results when the mutual information between pairs of random variables was not too large as these proved difficult to estimate to high accuracy without turning to a manual tuning of the bandwidths. We especially want to avoid the manual tuning of the bandwidth in this section, as we shall compute mutual information between 1653 pairs of random variables.

The random variables are the duration and delays of each process as well as the change in the level of the tank during these operations. Recall that e.g. $T_{4.1}^P$ is the duration of process 4.1 while T_4^D is the delay after all subprocesses of process 4 are completed. Also, for each *temporal* variable T_i^P and T_i^D, we have a corresponding change of level M_i^P and M_i^D. Initially, we also add the accumulated random variables. Namely, for process 1, we add the random variable $T_1 = T_1^P + T_1^D$ and likewise for the level changes for all processes. Also, the total duration $T = \sum_{i=1}^{10} T_i$ and change in level $M = \sum_{i=1}^{10} M_i$ are added as random variables. The accumulated random variables are initially added to see if our method can *rediscover* these simple causal relations. We will later remove these from our discussion as we try to gain a deeper understanding of the production system.

Hence, from the above, we initially have 68 variables. However, some of them are constant and thus out of interest. The constant *random* variables are

$$M_1^D, \quad M_2^D, \quad T_{4.1}^P, \quad M_{4.1}^P, \quad T_{4.3}^P, \quad M_4^D, \quad M_6^D, \quad T_8^P, \quad M_{10}^D$$

However, as $T_8 = T_8^P + T_8^D$ we shall also exclude either T_8 or T_8^D due to T_8^P being constant. In particular, T_8 and T_8^D can be interchanged at will when computing mutual information according to Corollary 3.2.1. Here, we have chosen to keep T_8^D as it is easier to relate to the other processes. Thus, we are only considering 58 random variables and hence $58 \cdot 57/2 = 1653$ pairs of variables as stated above.

We note that as the resolution in time is 0.001 we add uniform distributed noise from $[0, 0.001]$ to the durations and delays. In particular, we observe that in some cases equal durations are observed. These would result in non-uniform marginals like in Subsection 4.3.2 hence making the algorithm fail at computing the mutual information accurately. We note that the experiments to follow have been carried out multiple times to assess the influence of this perturbation on the durations and delays. However, as this did not influence our results in the slightest we will not discuss this further. Furthermore, instead of using a KDE as in Subsection 4.3.2 to transform the (continuous) observations to lie on the unit interval through the distribution function, we have used the empirical distribution function. This was done to ensure that all 58 random variables were transformed appropriately without having to consider transformations (like in Subsection 4.3.2).

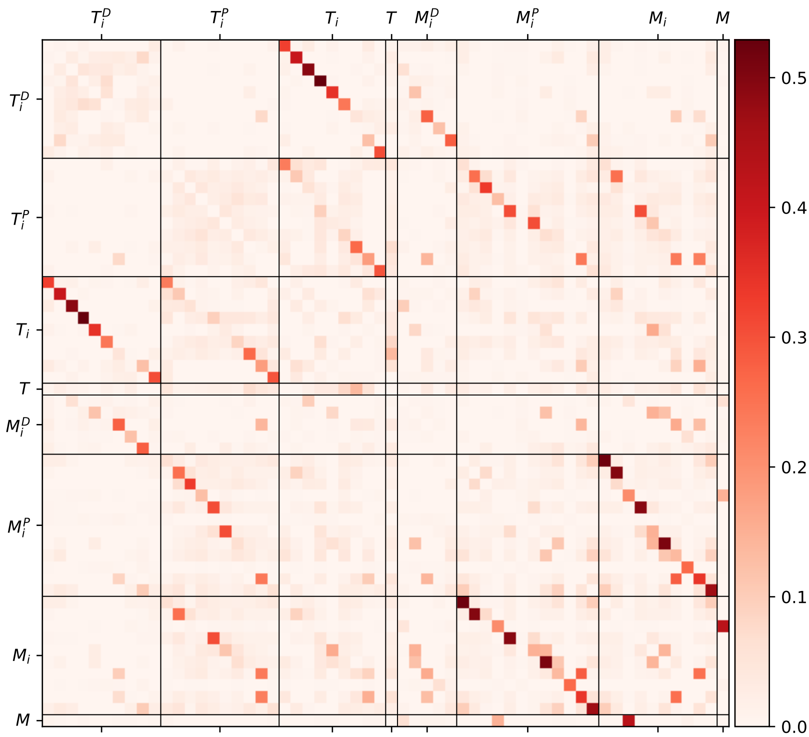


Figure 4.31: G_{dir} from a symmetric G_{obs}. Other than the accumulated variables, we observe strong dependencies between the duration of a process and the change in level during the process by comparing the columns and rows labeled T_i^P and M_i^P.

The resulting G_{obs} is shown in the appendix, Figure 6.16. We observe that there indeed is a strong dependence between the accumulated variables and their summands such as T_1 and T_1^D as well as T_1^P. We immediately use the deconvolution algorithm resulting in the G_{dir} seen in Figure 4.31

We have labeled sections of G_{dir} such that each section corresponds to an ordered list of variables. The variables in each section are

$$\begin{aligned}
T_i^D &= \{T_1^D, T_2^D, T_3^D, T_4^D, T_5^D, T_6^D, T_7^D, T_8^D, T_9^D, T_{10}^D\} \\
T_i^P &= \{T_1^P, T_2^P, T_{3.1}^P, T_{3.2}^P, T_{4.2}^P, T_5^P, T_6^P, T_7^P, T_9^P, T_{10}^P\} \\
T_i &= \{T_1, T_2, T_3, T_4, T_5, T_6, T_7, T_9, T_{10}\} \\
T &= \{T\} \\
M_i^D &= \{M_3^D, M_5^D, M_7^D, M_8^D, M_9^D\} \\
M_i^P &= \{M_1^P, M_2^P, M_{3.1}^P, M_{3.2}^P, M_{4.2}^P, M_{4.3}^P, M_5^P, M_6^P, M_7^P, M_8^P, M_9^P, M_{10}^P\} \\
M_i &= \{M_1, M_2, M_3, M_4, M_5, M_6, M_7, M_8, M_9, M_{10}\} \\
M &= \{M\}
\end{aligned} \tag{4.13}$$

I.e. the section labeled T_i^D consists of all the delays after each process and so on for the remaining section labels.

From Figure 4.31, we observe that the accumulated durations of each process T_i are indeed very dependent on both the delay and duration of the process. This is a sign that the algorithm performs well as by definition $T_i = T_i^D + \sum_{k \in \mathcal{P}_i} T_i^P$ (where $\mathcal{P}_i$ is defined as the set of subprocesses that constitute the process i).

For example, we observe that T_4^D is strongly associated with T_4 while $T_{4.2}^P$ is not as strongly associated with T_4. However, there is still a connection. If we plot T_4 vs. T_4^D and $T_{4.2}^P$ respectively (Figure 4.32) we indeed observe that the information between T_4^D and T_4 is much greater than $T_{4.2}^P$ and T_4. As we saw in chapter 2, this is because the delays after process 4 are much larger than the duration to begin with. Thus, even though T_4^D is 0, 56.25% of times, most of T_4 can be explained from this alone when compared to the scores between T_4 and the other random variables.

Furthermore, we observe that the total duration T is mostly explained by T_7 which in turn is explained mostly by T_7^P. Although there are many other interesting observations such as the most change in levels and masses occurs during process 3, addition of a material, and stirring. The level after process 3 is in turn mostly influenced by the stirring but also seems to have some unexplained variation. We observe the unexplained variation from the row corresponding to M_3 which has relatively small associations with the other delays, durations, and levels.

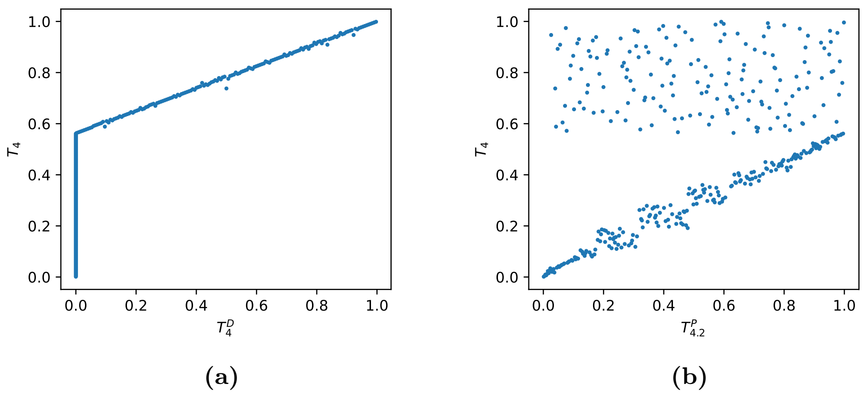
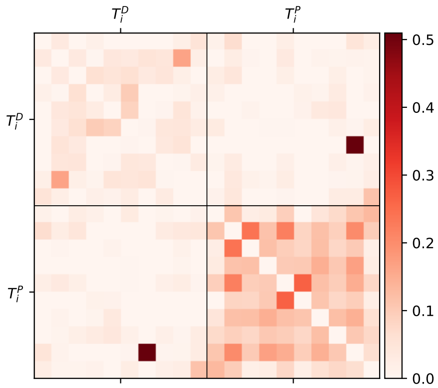


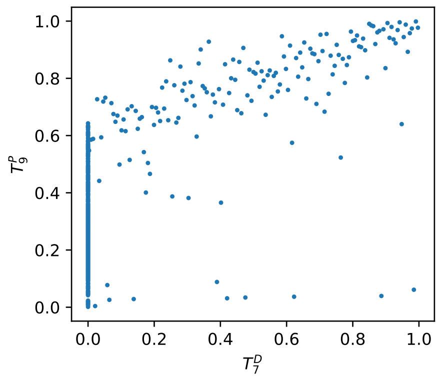
Figure 4.32: T_4 vs T_4^D and $T_{4.2}^P$ in (a) and (b) respectively. Notice that the continuous part of the variables has been transformed using the empirical distribution function such that the domain is $[0, 1]$ for all the variables. The transformation allows for an easier assessment of the mutual information between the pairs of variables.

At this point, we have observed that the algorithm rediscovers what we already know to be true. Namely, that the total masses and process durations are well explained by the individual processes. However, a more interesting question is how each of the processes affects each other. More precisely, if we only consider each process without the accumulated random variables T_i, T, M_i, and M, we hope to discover more interesting dependencies between the processes. Also, T_i, etc. can always be computed from the processes, so we do not care for these random variables if we want to understand the dynamics of the system.

Initially, we consider the durations and delays only. Namely, we choose the submatrix of G_{obs} corresponding to the sets T_i^D and T_i^P from Equation 4.13. Still using a symmetric G_{obs}, G_{dir} is computed as shown in Figure 4.33(a). Now, when removing the *trivial* random variables, as they are known to be the sum of others, a clearer image of how the durations and delay of each process depend on each other emerges. A strong association between T_9^P and T_7^D is observed. When plotting the observations of the random variables (again with the continuous part transformed through the empirical density function) in Figure 4.33(b), we indeed observe that for many batches, if the delay for process 7 is non-zero, then the duration of process 9 is greater than if the delay after process 7 was 0. Furthermore, as we have removed transitive effects, it seems as if this association is direct. I.e. something happens during the delay after process 7, the reaction process, which heavily *influences* to the duration of the cooling process 9. Interestingly, from Figure 4.33(a) it does not seem like T_7^P is associated with T_9^P. In Figure 6.15 we have shown these two variables plotted against each other and indeed observe that neither is particularly descriptive of the other.



(a) G_{dir}



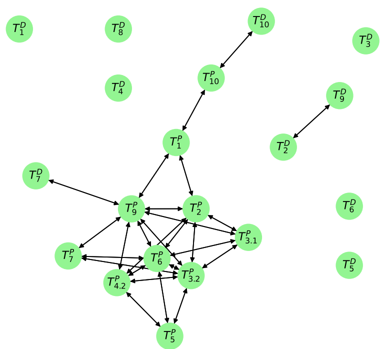
(b) T_9^P vs T_7^D

Figure 4.33: When only using the durations and delays, we observe G_{dir} as in (a). A very strong similarity is calculated between T_9^P and T_7^D, which when plotted in (b) is not an unreasonable finding. Namely, if the delay after process 7 is 0, T_9^P is primarily below the 0.6-quantile and otherwise appears to be linearly related to T_7^D.

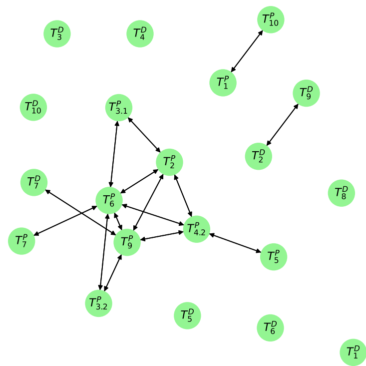
In figure Figure 4.34, we have shown G_{dir} (from Figure 4.33(a)) as a graph with varying thresholds. It is clear that the durations of each process are dependent on each other and depending on the threshold we conclude anything from a simple network structure (Figure 4.34(b) and Figure 4.34(c)) to a much more complicated dependency structure (Figure 4.34(a)) between the durations of the processes. As is also clear from Figure 4.33(a), the delays of the processes are the first random variables we conclude to be independent of both each other and all other process durations except for T_2^D and T_9^D corresponding to the delays when adding material and cooling respectively and the link between T_7^D and T_9^P discussed above.

Like in Subsection 4.3.3, we have a good understanding of the topological structure of the processes. Namely, as the delay of processes is always *after* the process we have that in terms of a topological structure of the random variables, T_i^D is always after T_i^P. For example, T_3^D is always realized *after* $T_{3.1}^P$ and $T_{3.2}^P$ (in that order). Likewise, as shown in Figure 2.1, as the processes are executed one by one, the topological structure is a simple chain. This should not be confused with the actual causal structure being a chain as in Section 4.1.

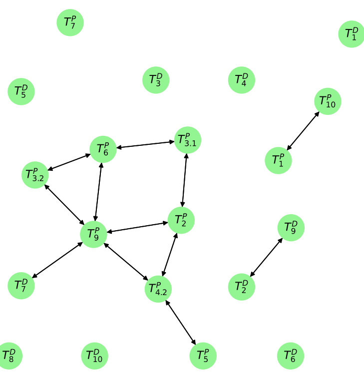
As the topological structure is a chain, it is also unique i.e. there is exactly one ordering of the random variables. The topological structure is summarized in Equation 4.14



(a) $t = 0.1$



(b) $t = 0.118$



(c) $t = 0.12$

Figure 4.34: G_{dir} from Figure 4.33 represented as a graph for different thresholds t. In general, we observe that the delays appear to be indescribable from the other variables and vice versa. As for the actual durations of the processes, depending on the threshold, we observe a more or less complicated relational structure. Varying the threshold t outside the considered range [0.1, 0.12] does not change the graph noticeably.

$$\begin{aligned} T_1^P &\longrightarrow T_1^D \longrightarrow T_2^P \longrightarrow T_2^D \longrightarrow T_{3.1}^P \longrightarrow T_{3.2}^P \longrightarrow T_3^D \\ \longrightarrow T_{4.2}^P &\longrightarrow T_4^D \longrightarrow T_5^P \longrightarrow T_5^D \longrightarrow T_6^P \longrightarrow T_6^D \longrightarrow T_7^P \\ \longrightarrow T_7^D &\longrightarrow T_8^D \longrightarrow T_9^P \longrightarrow T_9^D \longrightarrow T_{10}^P \longrightarrow T_{10}^D \end{aligned} \tag{4.14}$$

Using this to order the rows and columns of G_{obs} and finally only keeping the upper triangular part, we can deconvolve the network once again, but this time using the topological structure. The resulting G_{dir} is shown in Figure 4.35.

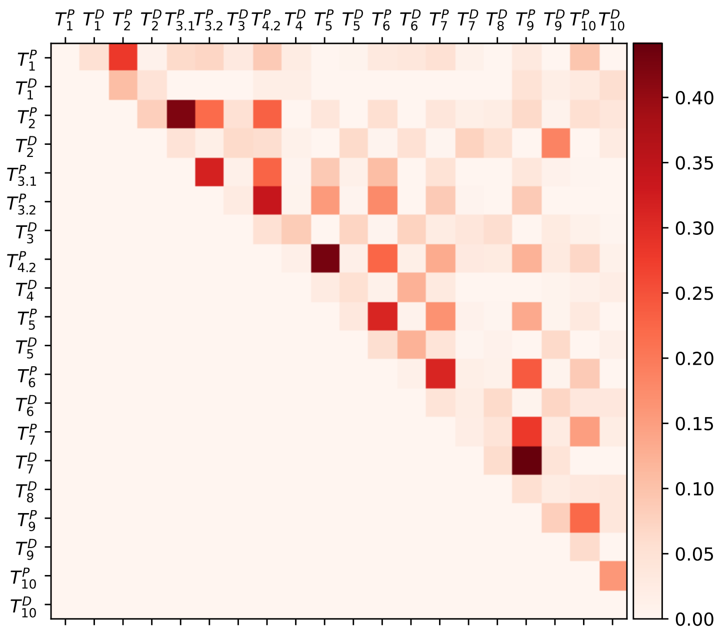
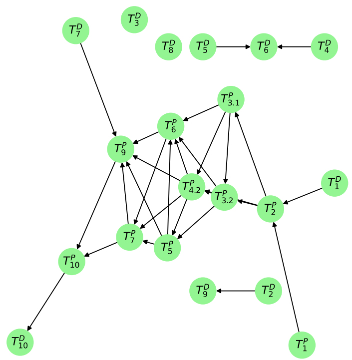
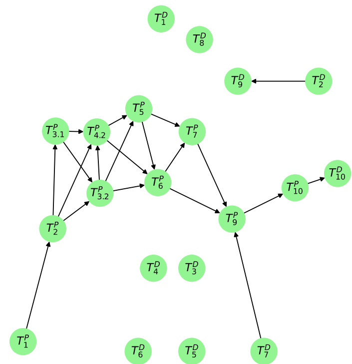


Figure 4.35: G_{dir} based on durations and delays but now with the added assumption of the topological order of the variables. Notice that the order of the variables is different from that of Figure 4.33(a) and hence not easily comparable. We refer to the graphical representations in Figure 4.34 and Figure 4.36 for comparison of dependence and causal structure.

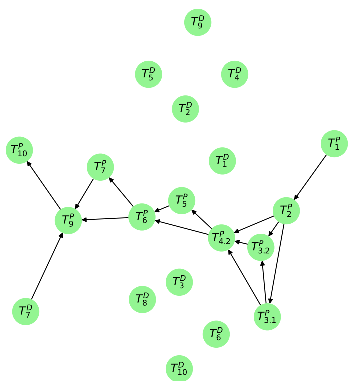
Although it is hard to compare this G_{dir} with the one obtained without the assumption of topological structure in Figure 4.33(a), we see some differences. T_1^P and T_{10}^P are not as directly dependent as originally inferred from the symmetric G_{obs}. The differences are however more noticeable when comparing the graphs from before with those in Figure 4.36 where the directed G_{obs} has been used instead.



(a) $t = 0.1$



(b) $t = 0.15$



(c) $t = 0.2$

Figure 4.36: Using a triangular G_{obs} resulting in a directed graph, we observe once again that the delays are largely unrelated. This hints that most delays should be treated separately when we only concern ourselves with the duration of processes. An important difference from these graphs to the previous where no topological order was assumed, is that now T_1^P only influences T_{10}^P indirectly where it was a direct link before. In general, we observe that durations only have a direct influence on the next process or the ones after that again. I.e. a more chain-like structure emerges compared to the previous results.

Depending on the chosen threshold t, we observe that still, the delays are the least predictable from other observations as they are not connected to other random variables. This was also the conclusion from Figure 4.34. The major difference is that now T_1^P influences T_{10}^P indirectly whereas in Figure 4.34 they were always directly connected. Also, a threshold in [0.1, 0.2] seems to result in a good balance between the connectedness and complexity of the graph. In our opinion Figure 4.36(b) obtains a good balance while also making sense in terms of how processes of a production flow, structured as in Figure 2.1, could influence later processes.

Finally, we shall try once again to use the levels of the tank after each process along with a topological assumption. However, as it is unclear whether it is the duration of a process that influences the change in levels or the other way around, we shall only make G_{obs} almost triangular. In particular, the entries in G_{obs} related to durations and levels of the same process such as T_1^P and M_1^P are symmetric along the diagonal whilst everything else is removed. This makes G_{obs} *almost* triangular in the sense that it is triangular but with entries in the subdiagonal (in case of an upper triangular G_{obs}).

The resulting G_{obs} can be seen in the appendix, Figure 6.17. As we would expect, level changes and durations for the same process are often very related and share much information. Unsurprisingly, $M_{4.3}^P$ does not seem to be related to any other process. This is likely because the process consists of waiting for a control operator. This also holds when deconvolving the network as seen in Figure 4.37 and from the original result in Figure 4.31 where all the random variables were used (although it is harder to see).

When comparing G_{dir} in Figure 4.37 to G_{obs} in Figure 6.17, we observe that much of the association between processes originate from indirect effects (as expected) either 1, 2 or 3 processes ago. In particular, the delays, durations, and level changes of process 9 (cooling) appear to be caused or at least explained well by what takes place during process 7 (reaction) which is not unlikely from a chemical point of view. Process 7, in turn, appears to be influenced mostly by process 6 (product transfer).

In Figure 4.38, we have shown G_{dir} represented as a graph with a threshold $t = 0.09$ as this filtered out most small values in G_{dir} while keeping a reasonable structure with only the edges with the most information carried along. As mentioned above $M_{4.3}^P$ is alone. Furthermore, we observe that the delays related to the process 3 (after adding the final solids) and 8 (post-reaction process) are by themselves. The duration of the delay and the change in level are however connected although weakly as per Figure 4.37. This also agrees with Figure 4.36(b) and Figure 4.36(c) where when we only used the durations and delays of the processes, we observed that T_3^D and T_8^D where unconnected nodes. As another example, M_8^P is not connected to any of the other variables invariant to the choice of threshold t. This also makes sense when comparing to Figure 6.1, Figure 6.2 and Figure 6.3 where the change in level during the post-reaction (process 8) does not seem to be related to any of the other random variables.

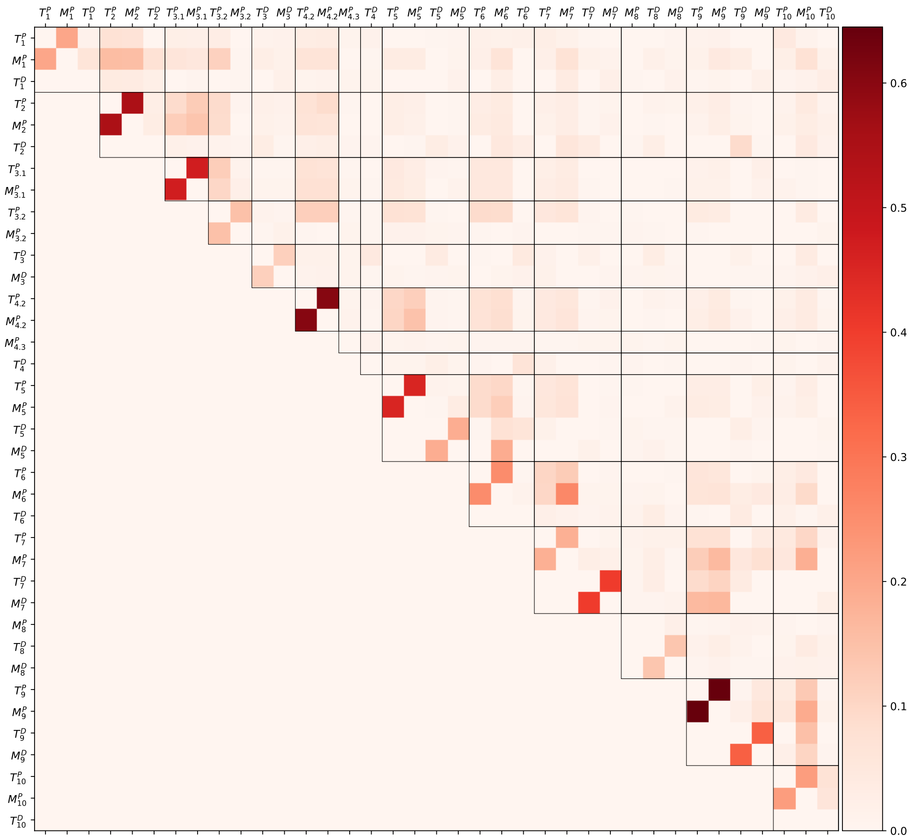


Figure 4.37: G_{dir} from an *almost* upper triangular G_{obs}. Like before, we observe a chain-like structure. However, with the added random variables of changes in levels during a process, the most significant dependencies are between these and the corresponding duration.

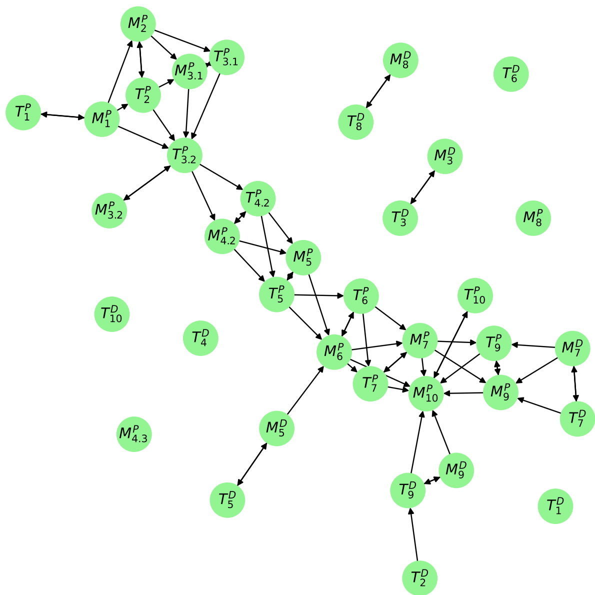


Figure 4.38: G_{dir} represented as a graph using a threshold $t = 0.09$. Once again, we observe a chain-like structure, but the added random variables corresponding to changes in levels, $T_{3.2}^P$ becomes a bottleneck, in the sense most of the behavior of the production system after process 3.2 is irrelevant when knowing the duration of process 3.2. This could also make sense from a practical point of view as the duration of the stirring contains much-combined information of the initial processes and hence is a good descriptor of the behavior of the processes later on.

Other interesting observations can be made from the graph depending on the interests of the reader. However, we round of our discussion of the pharmaceutical data by noting that we have obtained a causal structure for the random variables that in many ways reflect both our understanding of the production layout from chapter 2 and our observations of data. In particular, we observe a relatively simple structure, where many of the processes are only affected by the previous process. Notable deviations of this are the initial processes 1, 2, and 3.1 where raw materials are added. Each of these processes seems to have an influence on the duration of process 3.2 (the stirring). This also makes sense as the more material is added to the tank, the longer it probably needs to be agitated to ensure a consistent mixture of materials to allow for chemical reactions.

Finally, the change in level for process 10 (the transfer of material), appears to be influenced primarily by processes 7 (the chemical reaction) and 9 (cooling) while the duration of process 10 only seems to be related to the change in level and no other process directly. This would make sense practically, as we would think that it is the amount of material that is really influenced by the other processes and the duration of transferring the material is only as needed to be in order to remove the material produced.

CHAPTER 5

Conclusion and further perspective

In the chapter, we shall summarize our findings regarding network deconvolution and estimation of mutual information. In particular, we shall conclude on the observed properties of the framework including shortcomings, wherefrom these originate and potential fixes. Moreover, we shall state further perspectives and possible future studies. This includes ways to correct for some of these shortcomings, other methods for estimating mutual information and applications of the framework

5.1 Conclusion

Using network deconvolution, we observed that for linear chains and networks that can be represented by a directed acyclic graph, the causal structure could be recovered perfectly when the true correlation between pairs of random variables where known as well as the topological structure. Namely, for any network where a node X_i is given by $X_i = \epsilon_i + \sum_{j \in N_i^-} \vec{\rho}_{j,i} X_j$ where N_i^- denotes the parents or in-neighbors of X_i and ϵ_i is some independent noise, we observe that we can recover the coefficients $\vec{\rho}_{j,i}$. However, removing the assumption of a topological order, we observed a bleeding effect, where especially for chains, this resulted in an inability to perfectly recover the causal structure. In particular, the weak links in the chain with small $\vec{\rho}_{j,i}$ broke in the deconvolved network, whereas the more strongly connected subchains where observed to generate new edges, connecting random variables that should otherwise not be directly connected. This issue was not observed to the same extend when a more complex causal structure was used to generate the samples. Thus, when using correlations as the measure of similarity, we conclude that an assumption of the topological order of the random variables is important for reliable results, especially in the presence of chain-like substructures.

Furthermore, using only 400 samples from a network with 10 nodes, we show that the noise in the estimates of the correlations is so small that after a threshold is applied to the resulting deconvolved similarity matrix G_{dir}, we can in practice perfectly regain the network structure. From this, we show that when using mutual information instead of correlation, we introduce errors such that the underlying assumption does not hold true for how information is convolved. Namely, for mutual information, it no longer holds that $G_{obs} = \sum_{k=1}^{\infty} G_{dir}^{k}$. The error from this assumption is exemplified in the case of Gaussian chains which are observed to result in the largest errors. We find that as long as the pairwise correlations in the chain is at most 0.9 in terms of absolute value, the errors introduced by using mutual information are so small that the inferred causal structures are near identical to those of using correlation. In particular, we conclude that if the underlying causal structure is a linear DAG with Gaussian noise, and the observed random variables are homeomorphic to the underlying random variables such that the mutual information between the observed random variables is the same as for the underlying causal structure, we can in most cases perfectly recover the causal structure.

- Long chains can be a problem when the links have varying strength. We observe that the chain might break at the weak points and more strongly connected parts bleed into neighboring nodes resulting in *dense* subgraphs.

- Long chains using MI with highr correlation if symmetric seem to be connected $i - i + 1$ and $i - i + 2$. If very large MI might want to treat differently.

- Works well, trough experimentation, on linear networks. If $X_j = f(\sum X_i)$ for in-neighbors it is not necessarily well, however as MI is independent of marginals, we expect better performance on these systems than if one had used correlation. In particular, if it is a chain, transforming each variable gives exactly the same result.

5.2 Further perspectives

Chapter 6

Appendix

6.1 Jordan normal form of infinite matrix sum

In this section, we show that for a matrix A with spectral radius less than one, the eigenvalues of $B = \sum_{k=1}^{\infty} A^k$ are exactly $\frac{\lambda_i}{1-\lambda_i}$ for $\lambda_i \in \sigma(A)$. It is easy to show that if (λ, v) is an eigenpair of B, then $\left(\frac{\lambda}{1-\lambda}, v\right)$ is an eigenpair of B. However, it is not immediately clear if these are all the eigenpairs of B and hence the eigenvalues. Namely, if the algebraic and geometric multiplicities of the eigenvalues are not the same. To show this, let J be the Jordan normal form of A such that A is similar to J by some invertible matrix P. In particular, the Jordan normal form consists of Jordan blocks J_i such that

$$J = \begin{bmatrix} J_1 & & & \\ & J_2 & & \\ & & \ddots & \\ & & & J_p \end{bmatrix}$$

Hence, B can be written as

$$B = \sum_{k=1}^{\infty} A^k = \sum_{k=1}^{\infty} P^{-1} J^k P = P^{-1} \sum_{k=1}^{\infty} J^k P$$

Note that J^k is given by

$$J^k = \begin{bmatrix} J_1^k & & & \\ & J_2^k & & \\ & & \ddots & \\ & & & J_p^k \end{bmatrix}$$

Focusing on a single Jordan block J_i given by

$$J_i = \begin{bmatrix} \lambda_i & 1 & & \\ & \lambda_i & 1 & \\ & & \ddots & \\ & & & \lambda_i \end{bmatrix},$$

we see that $J_i^k = \lambda_i^k I + T$ where T is a strictly upper-triangular matrix. From this, we see that $\sum_{k=1}^{\infty} J_i^k = \frac{\lambda_i}{1-\lambda_i} I + T'$ where T' is another strictly upper-triangular matrix. Thus, $\sum_{k=1}^{\infty} J_i^k$ has $\frac{\lambda_i}{1-\lambda_i}$ as diagonal elements, hence there exists some invertible matrix P_i such that $\sum_{k=1}^{\infty} J_i^k = (P_i)^{-1} J_i' P_i$ where J_i' is a Jordan normal form (with $\frac{\lambda_i}{1-\lambda_i}$ as diagonal elements) of $\sum_{k=1}^{\infty} J_i^k$. Combining the above, we have that

$$\begin{aligned}
\sum_{k=1}^{\infty} J^k &= \begin{bmatrix} \sum_{k=1}^{\infty} J_1^k & & & \\ & \sum_{k=1}^{\infty} J_2^k & & \\ & & \ddots & \\ & & & \sum_{k=1}^{\infty} J_p^k \end{bmatrix} \\
&= \begin{bmatrix} P_1^{-1} J_1' P_1 & & & \\ & P_2^{-1} J_2' P_2 & & \\ & & \ddots & \\ & & & P_p^{-1} J_p' P_p \end{bmatrix} \\
&= \begin{bmatrix} P_1 & & & \\ & P_2 & & \\ & & \ddots & \\ & & & P_p \end{bmatrix}^{-1} \begin{bmatrix} J_1' & & & \\ & J_2' & & \\ & & \ddots & \\ & & & J_p' \end{bmatrix} \begin{bmatrix} P_1 & & & \\ & P_2 & & \\ & & \ddots & \\ & & & P_p \end{bmatrix}
\end{aligned}$$

Let P' be the above block diagonal matrix, consisting of P_is and J' be the block diagonal matrix consisting of J_i's and hence also a Jordan normal form. Then, we can finally write B as

$$B = (P'P)^{-1} J'P'P$$

In particular, B is similar to the Jordan normal form J' with diagonal elements $\frac{\lambda_i}{1-\lambda_i}$. In particular, the spectrum of B is exactly the elements of the spectrum of A mapped by $\lambda \mapsto \frac{\lambda}{1-\lambda}$.

6.2 Pharmaceutical duration and level changes plots

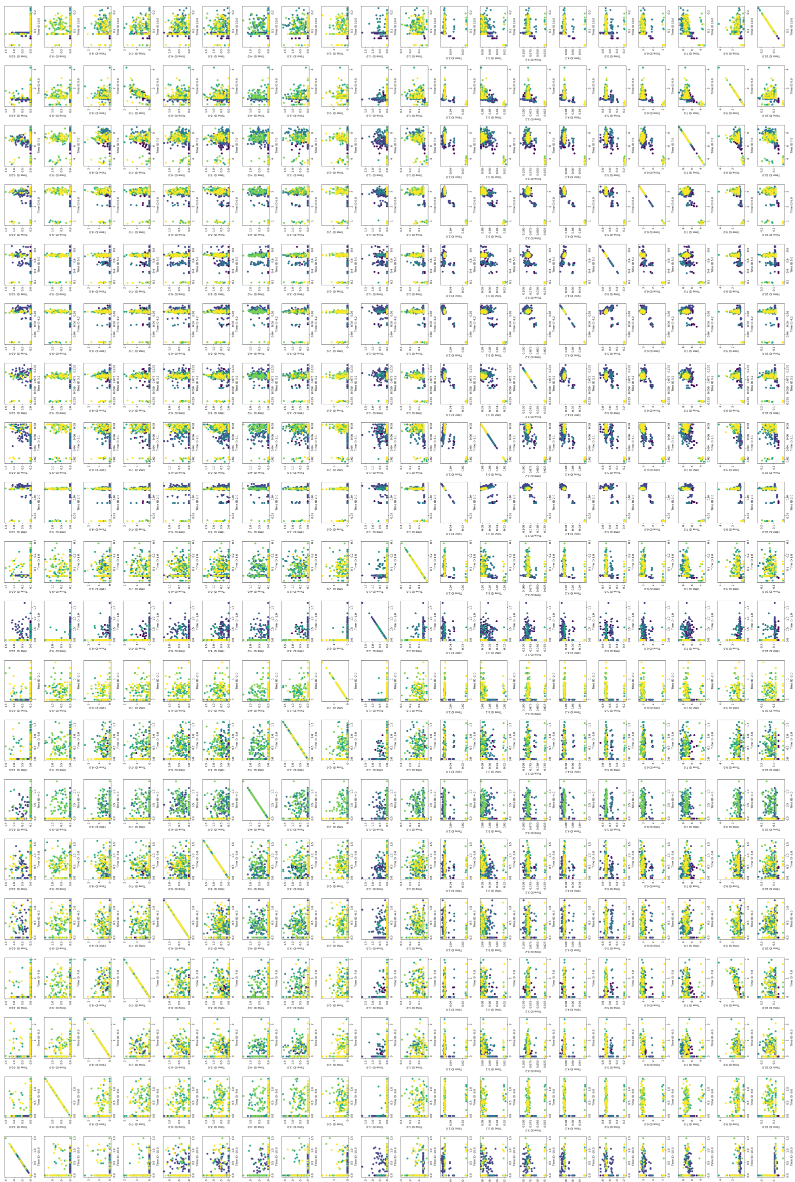


Figure 6.1: Durations and delays vs durations and delays. Some combinations of variables show a tendency, although a causal structure can not be inferred from this alone.

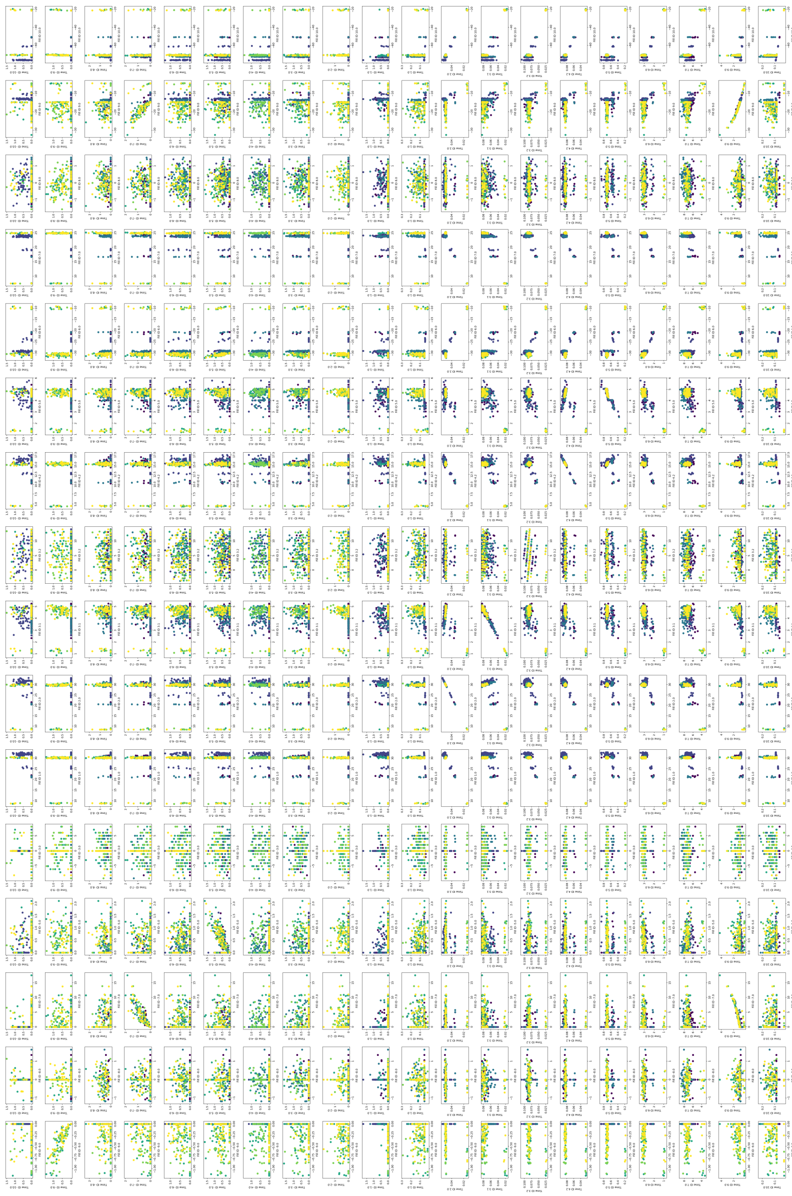


Figure 6.2: Changes in levels vs durations and delays.

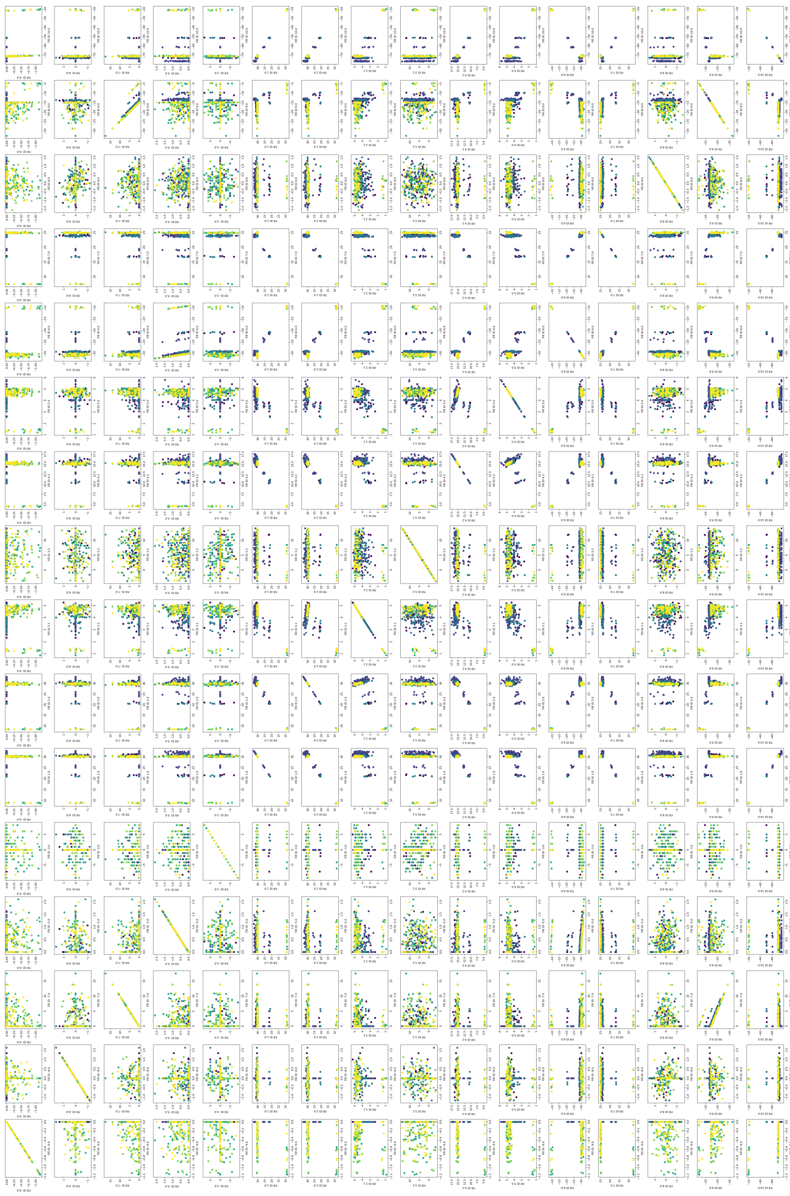


Figure 6.3: Changes in levels vs changes in levels.

6.3 Cleaning operations plots

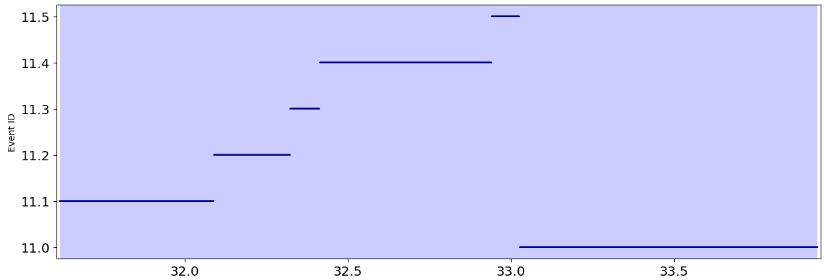


Figure 6.4: A single cleaning process and the process labels assigned during the duration. This blue rectangle corresponds to a single blue vertical line in Figure 6.5.

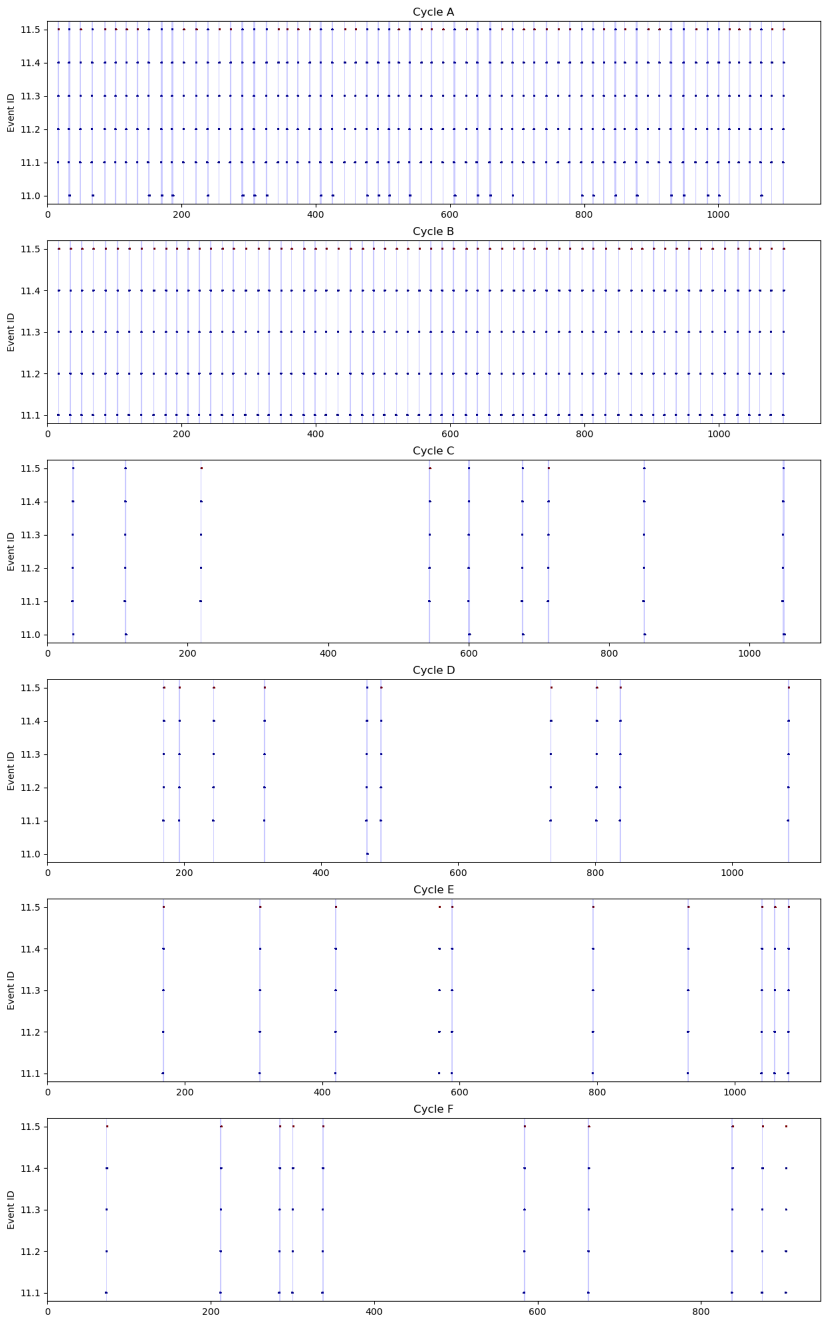


Figure 6.5: Each of the 6 cycles, cleaning (corresponding to `BatchID = 0` in the time series dataset). Each (Cleaning process), CIP, is highlighted by an opaque interval (the blue rectangles). The dots marked with red (only process label 11.5, but not all of these are red), is if the Cleaning ID is 0 in the data. Upon further inspection, this is assumed to be the existence of delays after a cleaning process, before the next batch is started.

6.4 Suicide data

1	25	40	83	123	256
1	27	49	84	126	257
1	27	49	84	129	311
5	30	54	84	134	314
7	30	56	90	144	322
8	31	56	91	147	369
8	31	62	92	153	415
13	32	63	93	163	573
14	34	65	93	167	609
14	35	65	103	175	640
17	36	67	103	228	737
18	37	75	111	231	
21	38	76	112	235	
21	39	79	119	242	
22	39	82	122	256	

Table 6.1: The length of treatment of control patients in suicide study. The data originates from the Mental Health Enquiry (MHE) of England of Wales and was published in 1967.

6.4.1 A better spline

In theory, we could make a set of splines ourselves, that has both of the desired properties of B-splines and M-splines. I.e. a set of piecewise polynomials, which we shall denote $Q_{i,p}$, that has both the property that

$$\sum_{i=1}^{n} Q_{i,p}(t) = 1$$

and

$$\int_0^1 Q_{i,p}(t)\,dt = \frac{1}{n}$$

Actually, both M- and B-splines satisfies this for $p = 0$ as they are then both piecewise constant and non-zero on disjoint intervals. From this, one can the construct such sets of $Q_{i,p}$ in the following way. Namely, start with the piecewise constant functions, $Q_{i,0}$ depicted in Figure 6.6 as boxes in the case $n = 5$.

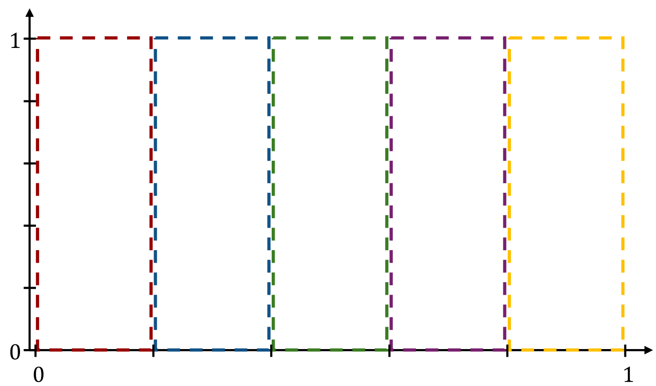


Figure 6.6: The initial splines from which we iteratively construct a set of splines summing to 1 and integrating to $1/n$ individually.

By cutting off equal areas of each of these rectangles, and moving them to the neighboring intervals as shown in Figure 6.7, we can obtain a set of splines as shown in Figure 6.8(a). In Figure 6.8(b) and Figure 6.9 we show the accumulated probability mass in each bin and the sum of the splines for all t and confirm that indeed the above properties hold. This process can be iterated to obtain more regularization/smoothing of the observations.

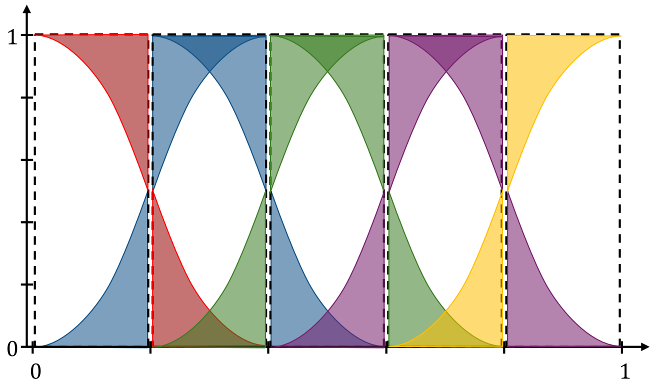
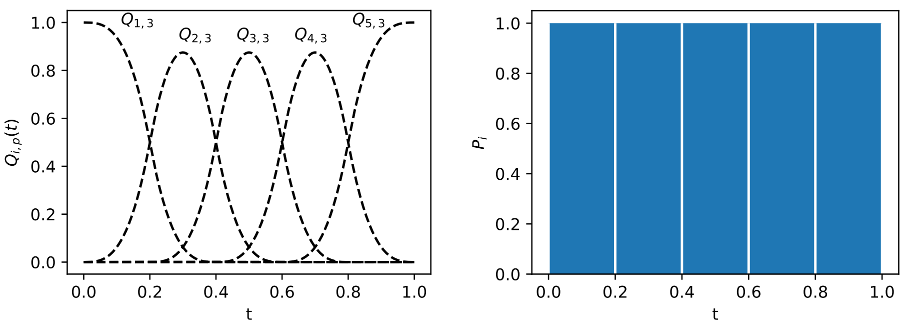


Figure 6.7: Each color represent an area removed from a rectangle and added to a neighboring interval such that the initial splines now are defined on multiple intervals, conserving the desired properties



(a) **(b)**

Figure 6.8: The resulting Q-splines (a) and the expected bin probability mass P_i is area of each rectangle i.e. 0.2.

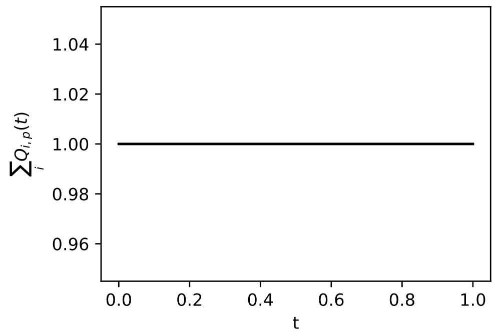
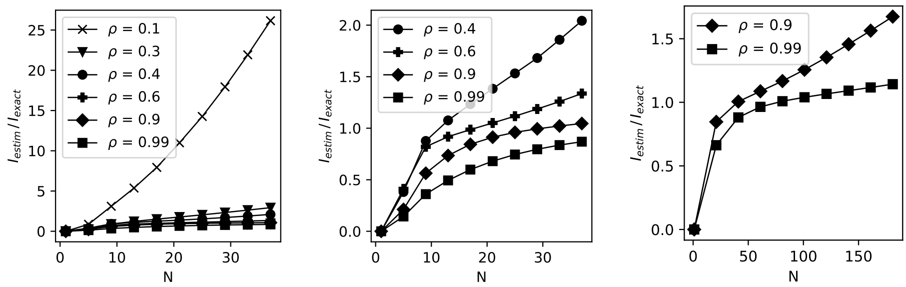


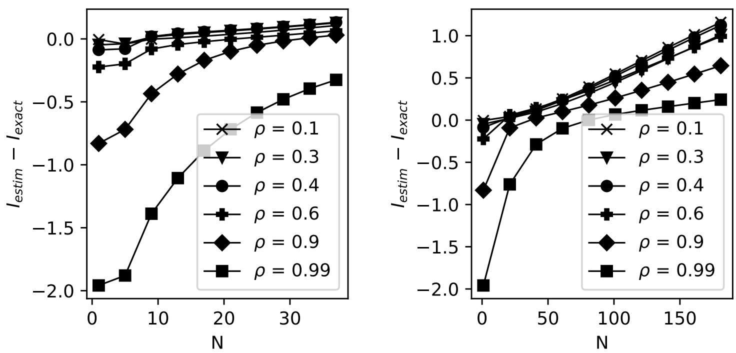
Figure 6.9: The sum of the set of Q-splines for all $t \in [0, 1]$. Indeed, they sum to 1 as per construction.

6.5 M-spline based MI estimation



(a) (b) (c)

Figure 6.10: Relative errors of the mutual information using M-splines. We observe the same behavior as with the B-splines. This indicates that neither is better than the other and thus no prefference between the two.



(a) (b)

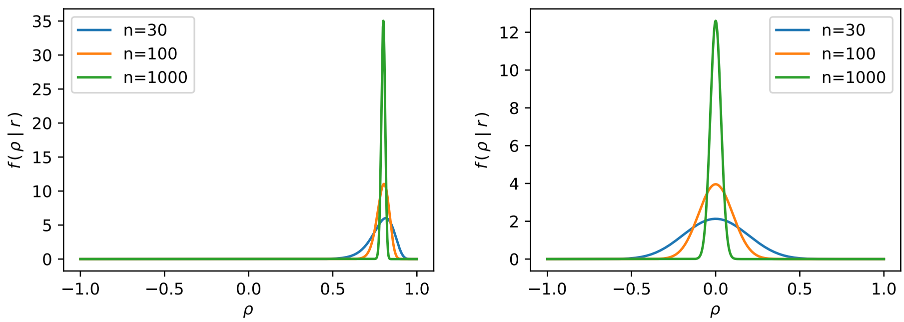
Figure 6.11: The actual error using the M-splines. Once again, we observe similar behavior to the B-splines.

6.6 Confidence interval for absolute correlation in bivariate Gaussian

From [30], given a bivariate Gaussian, the confidence distribution of ρ given the empirical correlation r based on n observations is given by

$$f\left(\rho \mid r, \nu\right) = \frac{\nu(\nu-1)\Gamma(\nu-1)}{\sqrt{2\pi}\Gamma(\nu+\frac{1}{2})} \frac{\left(1-r^2\right)^{\frac{\nu-1}{2}}\left(1-\rho^2\right)^{\frac{\nu-2}{2}}}{\left(1-r\rho\right)^{\frac{2\nu-1}{2}}} F\left(\frac{3}{2}, -\frac{1}{2}, \nu+\frac{1}{2}, \frac{1+r\rho}{2}\right)$$

where $F\left(a, b, c, z\right)$ is the Gaussian hypergeometric function and $\nu = n-1$. That is, given a sample correlation r, what is the confidence in ρ in terms of a distribution. In the following figure, a sample correlation $r = 0.8$ and $r = 0$ has been used with varying number of observations (degrees of freedom) in figures Figure 6.12(a) and Figure 6.12(b) respectively. A key property is that f is *even*



(a) **(b)**

Figure 6.12: $f\left(\rho \mid r, \nu\right)$ shown for $r = 0.8$ and $r = 0$ in (a) and (b) with $n \in \{30, 100, 1000\}$. As one would expect, the power i.e. the width of the peak decreases with increasing n and for correlations closer to 0, the width is the largest.

symmetric in ρ, r. That is $f(\rho \mid r) = f(-\rho \mid -r)$. Thus, a confidence interval for ρ given r is the negative of the confidence interval given $-r$. In particular, if we only observe $|r|$, we can calculate a confidence interval for ρ up to the sign of the bounds of the interval. Furthermore, as we want a CI for $|\rho|$, it does not matter if r is negative or positive. Hence, without loss of generality, we assume that $r \geq 0$. At this point, to construct a confidence interval for $|\rho|$ we list the following desired properties. Firstly, it should be an exact confidence interval, meaning that for a given significance level α, the CI includes the true value exactly $1-\alpha$ fraction of the times. Secondly, if for a given r, it can not be rejected that ρ is 0, 0 should also be contained in the interval. Finally, if we reject that $\rho = 0$, we shall have $\alpha/2$ probability mass above and below the bounds of the interval. The above is enough to uniquely define a confidence interval in all cases. Before continuing with how this CI is calculated, we mention that as r is an unbiased estimator of ρ, we would preferably want $|r| \in CI_{1-\alpha}(|\rho|)$ (where $CI_{1-\alpha}(|\rho|)$ denotes the $1-\alpha$ confidence interval for $|\rho|$). However, although this will in almost every scenario be the case, we can not be sure of this from the above properties and in fact examples with large α can be constructed such that $|r|$ lies just outside the constructed CI.

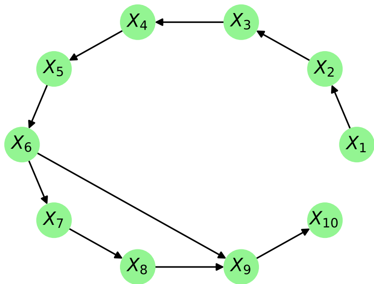
First, to conform with the second desired property, if it can not be rejected that $\rho = 0$ on a significance level α, we will initially compute a CI for ρ (not $|\rho|$) based on r (WLOG chosen to be non-negative). This CI will just be a symmetric CI in the sense that $\alpha/2$ of the probability mass will lie below the lower bound of the CI and above the upper bound of the CI respectively. If 0 is contained in this CI, we can not reject that $\rho = 0$ and vice versa on an α significance level. Thus, if 0 is contained in this initial CI for ρ, we will start the CI for $|\rho|$ at 0 and determine and upper bound b such that α probability mass is above this b. Otherwise, we shall find a and b such that $\alpha/2$ probability mass is below a and above b respectively. Choosing a and b this way also conforms with the third property. Finally, to ensure that the CI contains exactly $1-\alpha$, we define $\tilde{f}$ as the reflected f in ρ such that

$$\tilde{f}(\rho_a \mid r_a, \nu) = f(\rho_a \mid r_a, \nu) + f(-\rho_a \mid r_a, \nu), \quad \rho_a, r \in [0, 1]$$

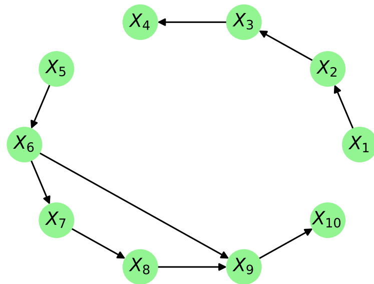
where ρ_a and r_a is the absolute correlation and empirical correlation respectively. With this $\tilde{f}$, the density at ρ_a is both the density for the negative and positive correlation ensuring that the $\tilde{f}$ has probability mass 1. Thus, if $a = 0$ (i.e. the CI must contain 0), we find b as the $1-\alpha$ percentile of $\tilde{f}$ and if $a \neq 0$, we take a as the $\alpha/2$ percentile and b as the $1-\alpha/2$ percentile of $\tilde{f}$.

As an example, suppose $r_a = 0.06$ with 1000 observations. Then a 95% CI for $|\rho|$ is $[0, 0.11164]$ whereas if on had observed $r_a = 0.07$ the CI would be $[0.01071, 0.1314]$. These CI could then be used to test the absolute correlation of a bivariate Gaussian i.e. for $r_a = 0.07$ based on 1000 observations would be rejected as stemming from a Gaussian with absolute correlation 0.01 on a 5% significance level.

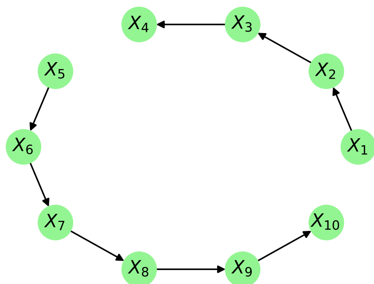
6.7 Gaussian chain deconvolution



(a) $t = 0.02$



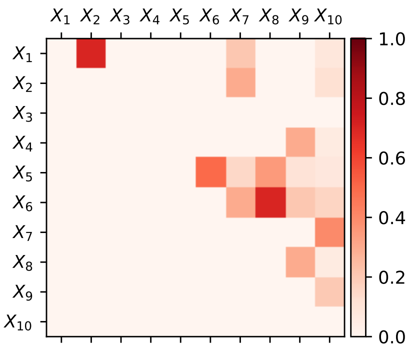
(b) $t = 0.021$



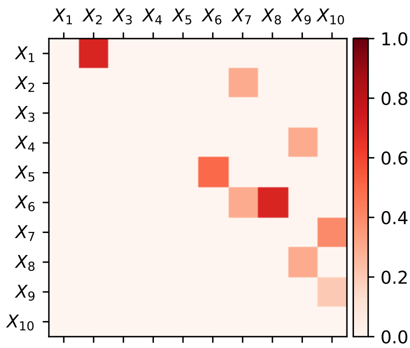
(c) $t = 0.0451$

Figure 6.13: G_{dir} represented as a graph for different cut-offs. Based on a triangular G_{obs} with mutual information as the measure of similarity. Varying the threshold, we observe that we are almost able to recover the true causal structure. Only the weak link between variables X_4 and X_5.

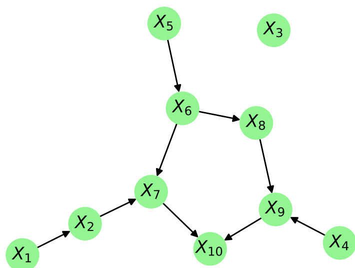
6.8 Gaussian network deconvolution



(a) G_{obs}



(b) G_{dir}



(c) G_{dir} as a graph

Figure 6.14: For the linear network defined in Equation 4.7, using a triangular G_{obs} (a) with the true topological structure and correlation as the measure of similarity we are able to perfectly rediscover the causal structure as seen in (b) and (c).

6.9 Pharmaceutical process deconvolution

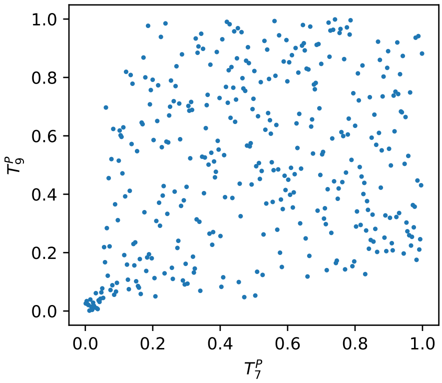


Figure 6.15: T_9^P vs T_7^P for the pharmaceutical data set. Although as small positive correlation is observed, it does not appear as if these variables are particularly descriptive of each other.

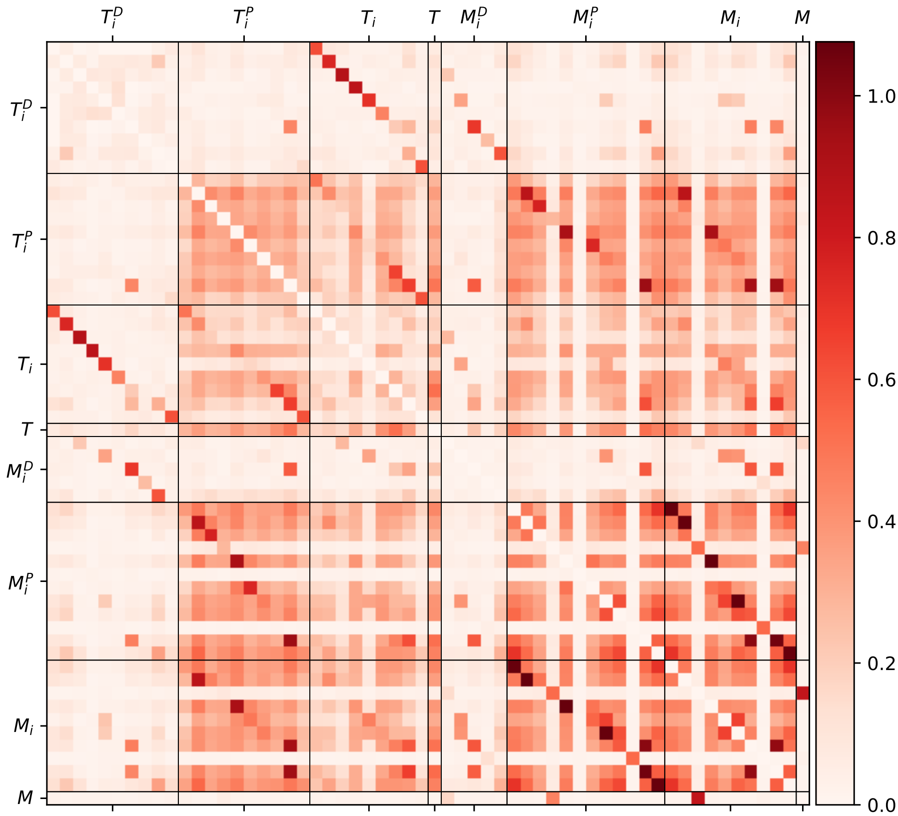


Figure 6.16: G_{obs} from pharmaceutical production data. Strong dependencies between variables and the related accumulated variables are observed.

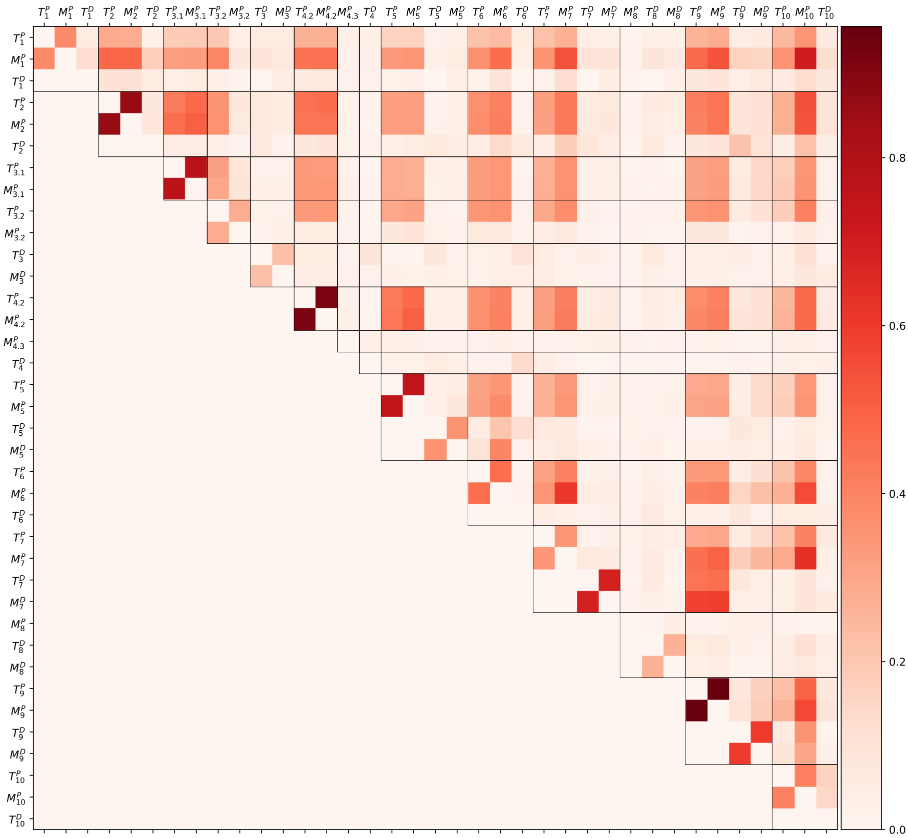


Figure 6.17: G_{obs} for both durations, delays and level changes for the simulated data from [9]. Labels related to the same process are divided by lines vertically and horizontally to easier observe what is related to what.

Bibliography

[1] "Novo nordisk supply update." `https://www.novonordisk-us.com/supply-update.html`. Accessed: 20124-07-28.

[2] M. J. Wainwright and M. I. Jordan, *Graphical Models, Exponential Families, and Variational Inference.* Hanover, MA, USA: Now Publishers Inc., 2008.

[3] N. Friedman, M. Linial, I. Nachman, and D. Pe'er, "Using bayesian networks to analyze expression data," in *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology*, RECOMB '00, (New York, NY, USA), p. 127–135, Association for Computing Machinery, 2000.

[4] N. Friedman, "Inferring cellular networks using probabilistic graphical models," *Science*, vol. 303, no. 5659, pp. 799–805, 2004.

[5] M. Weigt, R. A. White, H. Szurmant, J. A. Hoch, and T. Hwa, "Identification of direct residue contacts in protein-protein interaction by message passing," *Proceedings of the National Academy of Sciences*, vol. 106, no. 1, pp. 67–72, 2009.

[6] S. Feizi, D. Marbach, M. Médard, and M. Kellis, "Network deconvolution as a general method to distinguish direct dependencies in networks," *Nature biotechnology*, vol. 31, pp. 726 – 733, 2013.

[7] Y.-N. Sun, W. Qin, and Z. Zhuang, "Nonparametric-copula-entropy and network deconvolution method for causal discovery in complex manufacturing systems," *Journal of Intelligent Manufacturing*, vol. 33, 08 2022.

[8] W. Qin, D. Zha, and J. Zhang, "An effective approach for causal variables analysis in diesel engine production by using mutual information and network deconvolution," *Journal of Intelligent Manufacturing*, vol. 31, pp. 1661–1671, 08 2020.

[9] M. L. Vicente, J. F. Granjo, R. Tan, and F. D. Bähner, "A benchmark model to generate batch process data for machine learning testing and comparison," in *32nd European Symposium on Computer Aided Process Engineering* (L. Montastruc and S. Negny, eds.), vol. 51 of *Computer Aided Chemical Engineering*, pp. 217–222, Elsevier, 2022.

[10] J. Peters, D. Janzing, and B. Schlkopf, *Elements of Causal Inference: Foundations and Learning Algorithms.* The MIT Press, 2017.

[11] J. B. Copas and M. J. Fryer, "Density estimation and suicide risks in psychiatric treatment," *Journal of the Royal Statistical Society. Series A (General)*, vol. 143, no. 2, pp. 167–176, 1980.

[12] D. Kurowicka and H. Joe, *Dependence Modeling: Vine Copula Handbook.* World Scientific Publishing Co Pte Ltd, 12 2010.

[13] G. Geenens, "Copula modeling for discrete random vectors," *Dependence Modeling*, vol. 8, pp. 417–440, 12 2020.

[14] W. Gao, S. Kannan, S. Oh, and P. Viswanath, "Estimating mutual information for discrete-continuous mixtures," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[15] R. A. Horn and C. R. Johnson, *Matrix Analysis.* Cambridge University Press, 2 ed., 2012.

[16] H. V. Henderson and S. R. Searle, "On deriving the inverse of a sum of matrices," *SIAM Review*, vol. 23, no. 1, pp. 53–60, 1981.

[17] G. H. Golub and C. F. Van Loan, *Matrix Computations.* The Johns Hopkins University Press, third ed., 1996.

[18] J. J. Faith, B. Hayete, J. T. Thaden, I. Mogno, J. Wierzbowski, G. Cottarel, S. Kasif, J. J. Collins, and T. S. Gardner, "Large-Scale Mapping and Validation of Escherichia coli Transcriptional Regulation from a Compendium of Expression Profiles," *PLOS Biology*, vol. 5, pp. 1–13, January 2007.

[19] C. d. Boor, *A Practical Guide to Splines.* New York: Springer Verlag, 1978.

[20] J. O. Ramsay, "Monotone regression splines in action," *Statistical Science*, vol. 3, no. 4, pp. 425–441, 1988.

[21] D. W. Scott, "Multivariate density estimation: Theory, practice, and visualization," in *Wiley Series in Probability and Statistics*, 1992.

[22] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. London: Chapman & Hall, 1986.

[23] M. C. Jones and T. Buch-Kromann, "Simple boundary correction for kernel density estimation," *Statistics and Computing*, vol. 3, pp. 135–146, 1993.

[24] J. Dai and S. Sperlich, "Simple and effective boundary correction for kernel densities and regression with an application to the world income and engel curve estimation," *Computational Statistics & Data Analysis*, vol. 54, pp. 2487–2497, 08 2010.

[25] M. Jones and P. Foster, "A simple nonnegative boundary correction method for kernel density estimation," *Statistica Sinica*, vol. 6, 01 1996.

[26] M. C. Jones and D. A. Henderson, "Kernel-type density estimation on the unit interval," *Biometrika*, vol. 94, no. 4, pp. 977–984, 2007.

[27] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," *Phys. Rev. E*, vol. 69, p. 066138, Jun 2004.

[28] Z. Botev, "A novel nonparametric density estimator," *The University of Queensland, Tech Rep*, 01 2006.

[29] Z. I. Botev, J. F. Grotowski, and D. P. Kroese, "Kernel density estimation via diffusion," *The Annals of Statistics*, vol. 38, no. 5, pp. 2916 – 2957, 2010.

[30] G. Taraldsen, "Confidence in correlation," 11 2020.