Chapter 1

Method

General introduction to metodeafsnittet og causalitet

1.1 Causal discovery

In this section, we shall discuss the method for network deconvolution, originally proposed by [?]. 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 are made of some properties 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 1.1a 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 1.1b. With these initial comments, we proceed with the general setup and assumptions for network deconvolution based on observations.

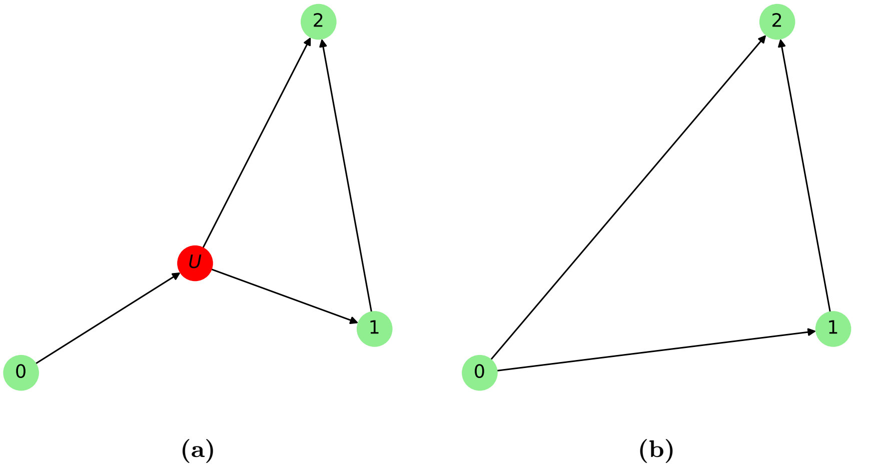


Figure 1.1: (a) An example of a causal structure depicted as a graph. When observing the network, only nodes 0, 1 and 2 are observed/recorded. (b) The resulting inferred graph from observational data. Although this is not a complete picture of the true underlying dynamics of the system, if only the observed 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.

1.1.1 Setup and assumptions

Suppose a set of N random variables (X_i) is given. 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 [?] then works as follows. Given an observed matrix $G_{obs} \in \mathbb{R}^{N \times N}$ of similarities between each pair of variables, we shall deduce a matrix $G_{dir} \in \mathbb{R}^{N \times N}$ 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 1.2 for a further discussion on these two measures and Section 1.3 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*.

illustration af hvordan algoritmen fungerer/formål

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 1.3.3, 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

$$G_{obs} = G_{dir} + G_{indir} \tag{1.1}$$

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

$$G_{indir} = G_{dir}^2 + G_{dir}^3 + \dots \tag{1.2}$$

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_{dir}^2) is the sum of products on edges e_{ik} and e_{kj} for all k

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

where e_{ij} is the (i,j) element of G_{dir}. This is of course not true in general, but through the error analysis in Subsection 1.3.3 and the examples in **??** this is not necessarily a problem. 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

$$G_{obs} = G_{dir} + G_{dir}^2 + G_{dir}^3 + \cdots = G_{dir} + G_{dir}\, G_{obs} \tag{1.3}$$

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. $\rho\left(G_{dir}\right) < 1$. Thus, assuming convergence we can rewrite the infinite series as

$$G_{obs} = G_{dir}\left(I - G_{dir}\right)^{-1} \tag{1.4}$$

It immediately follows that G_{dir} is given by (can be proved by directly inserting the above expression for G_{obs})

$$G_{dir} = G_{obs}\left(I + G_{obs}\right)^{-1} \tag{1.5}$$

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 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}). It follows that G_{dir} can be expressed in a simple (and later computationally efficient) way

$$G_{dir} = U\Lambda_{dir}U^T$$

where $\Lambda_{dir} = \Lambda_{obs} \left(I + \Lambda_{obs}\right)^{-1}$ which is also a diagonal matrix.

We note that from the above one needs $\left(I + G_{obs}\right)^{-1}$ to be well-defined which is equivalent to $-1 \notin \sigma_{G_{obs}}$ i.e. -1 is not an eigenvalue of G_{obs}. To see that this is indeed the case whenever $\rho\left(G_{dir}\right) < 1$, and that $I + G_{obs}$ is thus invertible we use Equation 1.4 and simplify

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

which is clearly invertible. Furthermore, we note that under the assumption $\rho\left(G_{dir}\right) < 1$ we can not place any bound on the spectral radius of G_{obs}. Namely, if v is a eigenvector of G_{dir} with eigenvalue λ such that $|\lambda| < 1$, then v is also an eigenvector of G_{obs} as

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

i.e. $\left(\frac{\lambda}{1-\lambda}, v\right)$ is an eigenpair of G_{obs} and since $\frac{\lambda}{1-\lambda} \in (-1/2, \infty)$ for $\lambda \in (-1, 1)$ we can in general not bound the spectral radius of G_{obs}, although we should never observe an eigenvalue equal to or below $-1/2$ (which again proves that -1 is not an eigenvalue of G_{obs}).

As we shall later use some assumptions regarding causality i.e. directing the edges in the graph, we shall investigate the effect of letting G_{obs} be a triangular matrix which corresponds to a directed acyclic graph. 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 1.5, 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 which is true as $I + G_{obs}$ has only ones in the diagonal as we will later assume in Equation 1.8. A simple proof of this is as follows. Without loss of generality, we assume 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, $T = D\left(I + D^{-1}T_u\right)$ and hence, we have that

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

which is clearly also upper triangular. Thus, we conclude that G_{obs} is triangular if and only if G_{dir} is (under the assumption G_{dir} exists).

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.*

1.2 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.

1.2.1 Copula

Given a set of N random variables $X_1, \ldots, X_d$, a copula is loosely speaking a distribution function with support $[0,1]^d$ incorporating the dependence structure between the random variables. Given a joint distribution function F and (invertible) marginals $F_1, \ldots, F_N$ we define a copula C as

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

Letting $u_i = F_i\left(x_i\right) \in [0,1]$ it is clear that C is a distribution function as described above [?]. 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 1.1 (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 1.2 (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*

$$F(x_1, \ldots, x_d) = C(F_1(x_1), \ldots, F_d(x_d)) \tag{1.6}$$

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

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[**?**]. 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[**?**] is a bivariate random vector of independent variables $X \sim \text{Bern}(\pi_X)$ and $Y \sim \text{Bern}(\pi_Y)$. 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

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

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 1.6 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 1.2.1 (Coordinate transformation). *Under the assumptions of Theorem 1.2, given any set $(T_1, \ldots, T_d)$ of strictly increasing functions, if C is a copula of $(X_1, \ldots, X_d)$ then it is also a copula of $(T_1(X_1), \ldots, T_d(X_d))$.*

Proof. Suppose $(X_1, \ldots, X_d)$ permits 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 $\boldsymbol{X}$

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

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

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

Proposition 1.3. *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 $(X_1, \ldots, X_d)$ if and only if it is a copula of $(T_1(X_1), \ldots, T_d(X_d))$.*

1.2.2 Mutual information and Copula entropy

In this section we introduce Copula entropy as done in [?] 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. 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 1.4 (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 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 1.2.3. For now, we proceed with the definition of (joint) entropy for both discrete and continuous random vectors.

Definition 1.5 (Entropy)**.** *The (joint) entropy of a random vector $\boldsymbol{X}$ is defined as*

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

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 $h\left(\boldsymbol{X}\right)$ instead of $H\left(\boldsymbol{X}\right)$.

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 1.2.3.

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

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

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:

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

□

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

Definition 1.7 (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,

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

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

Theorem 1.8 (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 1.2, letting $x_i = F_i^{-1}(u_i)$, we can relate the copula density to the joint density of $\boldsymbol{X}$ and its marginals

$$\begin{aligned} c(u_1, \dots, u_n) &= \frac{\partial}{\partial \mathbf{u}} C(u_1, \dots, u_n) \\ &= \frac{\partial}{\partial \mathbf{u}} F\left(F_1^{-1}(u_1), \dots, F_n^{-1}(u_n)\right) \\ &= f\left(F_1^{-1}(u_1), \dots, F_1^{-1}(u_n)\right) \frac{1}{f_1\left(F_1^{-1}(u_1)\right) \dots f_n\left(F_1^{-1}(u_d)\right)} \end{aligned}$$

It follows directly that

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

where the third equality follows from a change of variables with the trivial substitution $u_i = F_i(x_i)$ such that $du_i = f_i(x_i)\,dx_i$. 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.

1.2.3 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 this as 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:

$$\mathbb{P}\left(X^{\Delta} = x_i\right) = f(x_i)\Delta = \int_{i\Delta}^{(i+1)\Delta} f(x)\,dx \tag{1.7}$$

Clearly, this discretization is 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

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

Clearly, the first term in the above expression converges to the differential entropy $h\left(X\right)$ 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 1.6

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

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 1.4.

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

$$f\left(x_1^{(i)}, x_2^{(j)}\right)\Delta^2 = \int_{j\Delta}^{(j+1)\Delta}\int_{i\Delta}^{(i+1)\Delta} f(x_1, x_2)\,dx_1 dx_2, \quad \forall i, j \in \mathbb{Z}$$

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 right marginals and probability mass 1. 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 $I_{1,2}^{\Delta}$ we obtain

$$\begin{aligned} I_{1,2}^{\Delta} &= \sum_{i\in\mathbb{Z}} f_1\left(\tilde{x}_1^{(i)}\right)\log f_1\left(\tilde{x}_1^{(i)}\right)\Delta + \sum_{j\in\mathbb{Z}} f_2\left(\tilde{x}_2^{(j)}\right)\log f_2\left(\tilde{x}_2^{(j)}\right)\Delta \\ &\quad - \sum_{i\in\mathbb{Z}} f_1\left(\tilde{x}_1^{(i)}\right)\log f_1\left(\tilde{x}_1^{(i)}\right)\Delta - \log\Delta \\ &\to \infty \text{ as } \Delta \to 0 \end{aligned}$$

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.

1.2.4 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 1.2.1, 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 $\sigma_i = \sqrt{\operatorname{Var} X_i}$. 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.

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

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

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

again, under the assumption that Z_i has mean 0.

Thus, we conclude that indeed mutual information and correlation is very different measures of codependency (as correlation depends on the marginals whereas mutual information does not) and that 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 except if marginals should be insignificant to the network, Copula entropy is preferred.

1.3 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 1.8 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

$$G_{obs} = \begin{bmatrix} 0 & -CE_{12} & \dots & -CE_{1n} \\ -CE_{21} & 0 & \dots & -CE_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -CE_{n1} & -CE_{n2} & \dots & 0 \end{bmatrix} \tag{1.8}$$

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 (de)convolving 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 1.5, we would get the identity matrix which does not tell us much about the direct dependencies.

Algorithm 1 then follows immediately from Equation 1.8

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 **??**. 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 1.2.3, 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 1.1 (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 1.7 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*

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

1.3.1 Network deconvolution

At this point, we have obtained a convolved matrix of information G_{obs} and are ready to use Equation 1.5. We present the original algorithm from [**?**] 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: $G_{obs}$, $\alpha$, $\beta$
    $[G_{obs}]_{ii} \leftarrow 0$, $\forall i \in \{1, \dots, N\}$        ▷ ensure zero-diagonal
    $[G_{obs}]_{ij} \leftarrow 0$, when $[G_{obs}]_{ij} < Q_\alpha (G_{obs})$
    Compute eigendecomposition $U, \Lambda$ of $G_{obs}$
    $\lambda^+ \leftarrow \max (\lambda^{\max}, 0)$
    $\lambda^- \leftarrow \min (\lambda^{\min}, 0)$
    $c_s^+ \leftarrow \frac{1-\beta}{\beta} \lambda^+$
    $c_s^- \leftarrow \frac{1+\beta}{\beta} \lambda^-$
    $c_s \leftarrow \max (c_s^+, -c_s^-)$
    $\hat{\Lambda} \leftarrow \Lambda \left(c_s^{-1} I + \Lambda\right)^{-1}$
    return $U \hat{\Lambda} U^T$
```

where $Q_\alpha (G_{obs})$ 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 1.3.3.

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 **??**. 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 [**?**], 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.

1.3.2 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 [?] 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. $c_s = \max\left(\frac{1-\beta}{\beta}\lambda^+, -\frac{1+\beta}{\beta}\lambda^-\right)$ 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 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 [?] (which we have redone to know why the original implementation and derivation differs), we obtain that for a λ in $[0, \infty)$, $(-c_s^{-1}, 0)$ or $(-\infty, 0]$ to be mapped to $[-\beta, \beta]$ we have that

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

Thus, the smallest c_s^{-1} we can choose to ensure that $\rho\left(\tilde{G}_{dir}\right) \leq \beta$ is by $c_s = \min\left(\frac{1-\beta}{\beta}\lambda^+, -\frac{1+\beta}{\beta}\lambda^-\right)$ 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

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

made in the original discussion of the parameter β in [?]. 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 1.4. 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.

1.3.3 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 [?], 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/Euclidian 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 ([**?**]), and that we shall assume that $\rho\left(G_{obs}\right), \rho\left(\tilde{G}_{obs}\right) < 1$.

Definition 1.9 (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 $G_{dir} - \tilde{G}_{dir}$:

$$\begin{aligned}
\left|\left|G_{dir} - \hat{G}_{dir}\right|\right| &= \left|\left|G_{obs}\left(I + G_{obs}\right)^{-1} - \hat{G}_{obs}\left(I + \hat{G}_{obs}\right)^{-1}\right|\right| \\
&= \left|\left|-\sum_{k\geq 1}\left(-G_{obs}\right)^k + \sum_{k\geq 1}\left(-\hat{G}_{obs}\right)^k\right|\right| \\
&\leq \sum_{k\geq 1}\left|\left|G_{obs}^k - \left(G_{obs} + N\right)^k\right|\right| \\
&\leq \sum_{k\geq 1}\sum_{i=1}^{k}\binom{k}{i}||N||^i\,||G_{obs}||^{k-i} \\
&= \sum_{k\geq 1}\sum_{i=1}^{k}\binom{k}{i}\gamma^i\delta^{k-i} \\
&= \sum_{k\geq 1}\left((\gamma+\delta)^k - \delta^k\right) \\
&= \frac{\gamma+\delta}{1-\gamma-\delta} - \frac{\delta}{1-\delta} \\
&= \frac{\gamma}{(1-\gamma-\delta)(1-\delta)}
\end{aligned} \tag{1.9}$$

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 and we observe 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 [**?**], which is very useful when doing matrix perturbation analysis.

Theorem 1.10 (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 $G_{dir} - \tilde{G}_{dir}$

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

where the third equality follows from Theorem 1.10. 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.

$$\left|\left|G_{dir} - \tilde{G}_{dir}\right|\right| \leq ||N|| \left|\left|\left(I + G_{obs}\right)^{-1}\right|\right| \left|\left|\left(I + G_{obs} + N\right)^{-1}\right|\right| \tag{1.10}$$

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 1.9. Equation 1.10 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 **??** we discuss the error from using mutual information in the case of a multi-variate Gaussian.

From Equation 1.10, by another application of Theorem 1.10, 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 [**?**], 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

$\left|\left|(I+G_{obs})^{-1}\right|\right|_2, \left|\left|\left(I+\tilde{G}_{obs}\right)^{-1}\right|\right|_2 \leq 2$. Using this with the above equivalence on the Euclidean norm with Equation 1.10, we conclude that

$$\begin{aligned} \left|\left|G_{dir} - \tilde{G}_{dir}\right|\right|_F &\leq 4\sqrt{d}\,||N||_F \\ \left|\left|G_{dir} - \tilde{G}_{dir}\right|\right|_{\max} &\leq 4d\,||N||_{\max} \end{aligned} \tag{1.11}$$

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.

1.4 Estimating mutual information

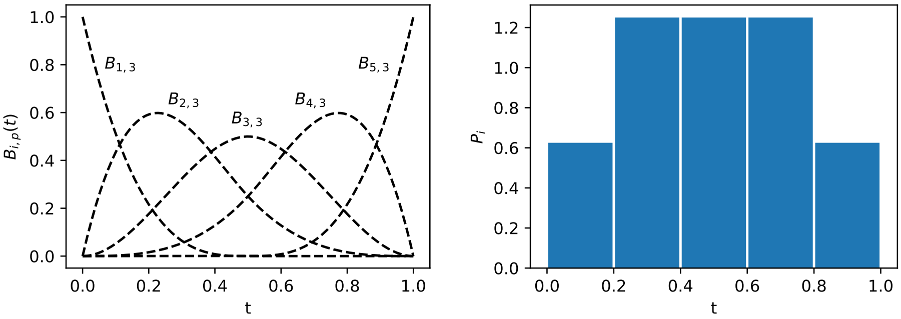
1.4.1 B-splines

B-spline - non-uniform marginals

[?] Cox-de Boor recursion formula

$$B_{i,0}\left(t\right) = 1, \quad t \in [t_i, T_{i+1}) \tag{1.12}$$

$$B_{i,k}\left(t\right) = \frac{t - t_i}{t_{i+k} - t_i} B_{i,k-1}\left(t\right) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} B_{i+1,k-1}\left(t\right) \tag{1.13}$$



(a) **(b)**

Have the useful property that $\sum_i B_{i,k}(t) = 1$ for all k and t. However, as we will be using this when trying to estimate mutual information from a copula, we have a very specific problem, namely, the marginals are not discrete uniform. $i \in \{1, \ldots, n\}$ basis functions

Let $\boldsymbol{X}$ be a random vector of d, i.i.d uniformly distributed random variables X_j. Then the random variable corresponding to the probability mass assigned to bin i, P_i, is given by

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

thus, the expected value of P_i is (independently of d)

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

Ideally, this should be equal for all i as we want the marginal to be uniform as per definition of a Copula density. However, this is not the case. In fact,

In particular, we can calculate the expected probability mass for each bin i as

1.4.2 M-splines

M-spline - need to normalize distributions

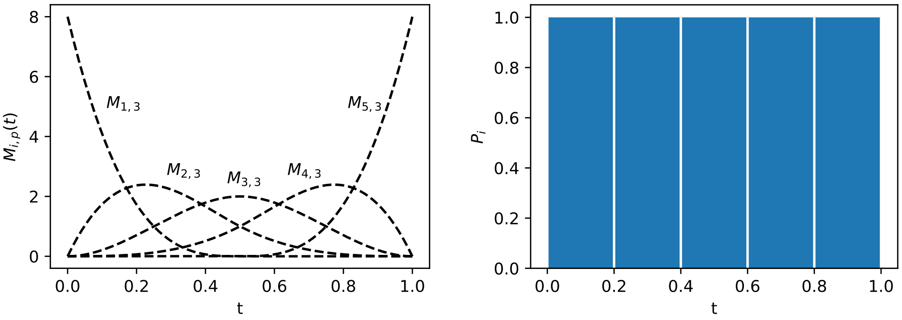
[?] a set of n basis functions indexed by i. Letting the order of the basis function equal $p+1$ such that the basis functions are polynomials of degree p. Define knots to be $t_1 = \cdots = t_k = 0$ and $t_{n+1} = \cdots = t_{n+k} = 1$. furthermore, we choose the remaining $n-k$ (interior) knots to be equally distributed on $[0, 1]$ i.e. if 3 interior points, they are 1/4, 1/2 and 3/4 respectively. (slightly reformualted to agree with the notation for B splines)

$$M_{i,0} \left(t \right) = \frac{1}{t_{i+1} - t_i}, \quad t \in \left[t_i, T_{i+1} \right) \tag{1.14}$$

$$M_{i,k} \left(t \right) = \frac{k \left(\left(t - t_i \right) M_{i,k-1} \left(t \right) + \left(t_{i+k} - t \right) M_{i+1,k-1} \left(t \right) \right)}{\left(k - 1 \right) \left(t_{i+k} - t_i \right)} \tag{1.15}$$

such that $M_{i,p}$ is a degree p polynomial on $\left[t_i, t_{i+p+1} \right)$

it follows that each of these $M_{i,p+1}$ basis splines integrate to 1 which, as we saw before is important for uniform marginals. In particular, if we instead let the basis functions be $\tilde{M}_{i,p} = \frac{1}{n} M_{i,p}$, The marginal is uniform. However now there is the downside that basis functions do not sum to 1 as with the B-spline. Namely, points near the boundary are weighted more than points on the inside as is illustrated below



(a)

(b)

Figure 1.3: P_i is area of each rectangle i.e. 0.2.

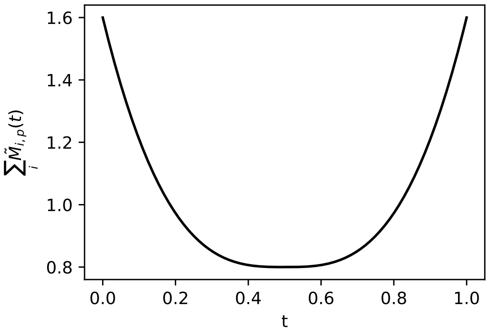


Figure 1.4

1.4.3 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 1.5 as boxes in the case $n = 5$.

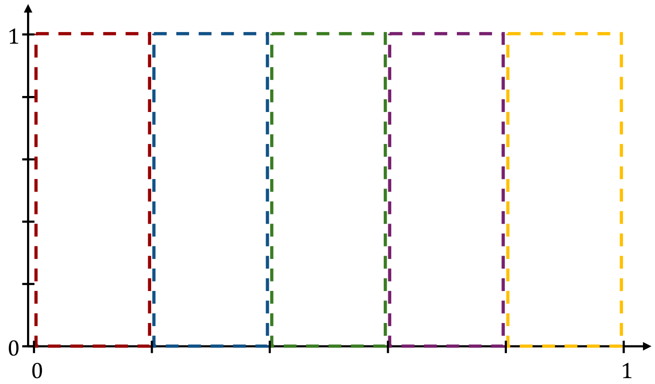


Figure 1.5

A better spline

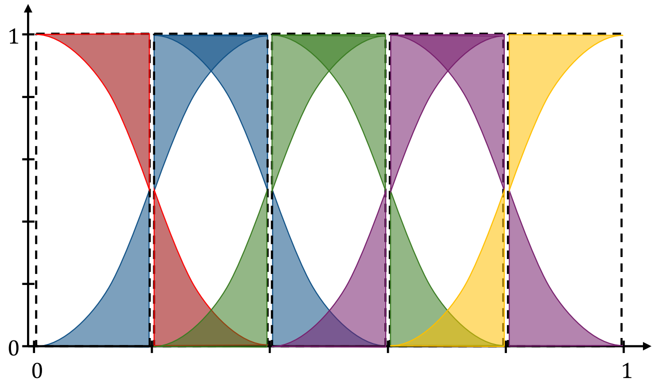
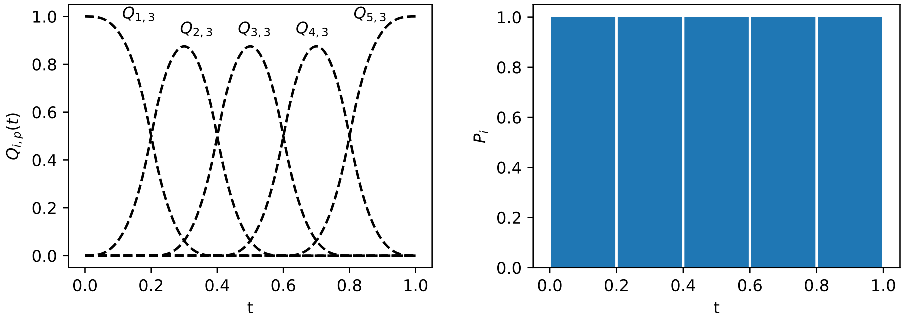


Figure 1.6

Cut off function has derivative 0 at top and is halfway down at the sides

This example does however not work well for many bins, in particular, due to the splines not overlapping that much i.e. a poor regularization on the smoothness. However, the method used for constructing this set of splines can be extended to bleed into multiple neighboring bins just as for B- and M-splines.



(a) **(b)**

Figure 1.7: P_i is area of each rectangle i.e. 0.2.

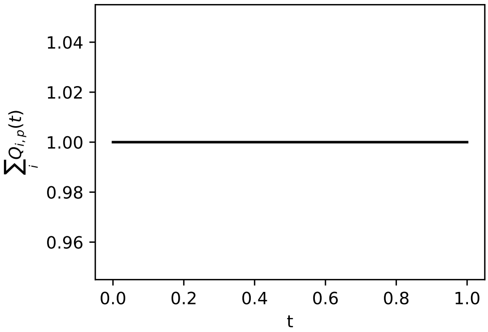


Figure 1.8

1.4.4 Naïve KDE

As we shall see in REFERENCE TIL RESULTAT AFSNIT OM EST AF MI UD FRA PERFEKT DENSITET, 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

$$\hat{f}(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{\prod_{j=1}^{d} \boldsymbol{h}_{i,j}} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{\boldsymbol{h}_i}\right) \tag{1.16}$$

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 1.16 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

$$K\,(\boldsymbol{x}) = \prod_{i=1}^{d} K_i\,(x_i) \tag{1.17}$$

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 j'th dimensino.

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

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

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

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 [?] 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

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

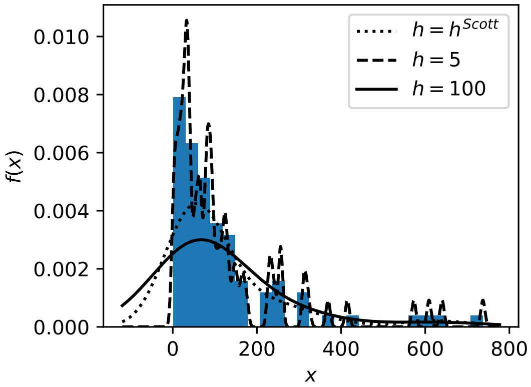


Figure 1.9: The suicide data from MHE. The 86 observations are shown as a histogram of densities along with Gaussian KDEs with bandwidth $h = 5$ and $h = 100$ and h chosen from Scott's rule $h = \hat{\sigma} n^{-1/5} \approx 59.86$.

In Figure 1.9, 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 data from [?], tabulated in [?] which has been used in [?] 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 **??**. 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 1.10 where this method is applied to the same example as above, this tends to underestimate the peaks especially near the boundary.

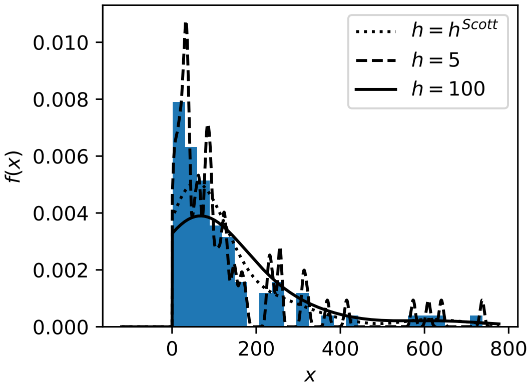


Figure 1.10: Using a rescaled version of $\hat{f}$ on the interval $[0, \infty)$ and disregarding any probability mass below $x = 0$ we obtain proper probability distributions once again. However, neither of the methods capture the peak near the boundary $x = 0$. In particular, although h^{Scott} still seem to be a good choice for h, the KDE does not capture the tendency observed in the data.

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.

1.4.5 Boundary corrected KDE

Before introducing the boundary corrected kernels presented by [?], 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

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

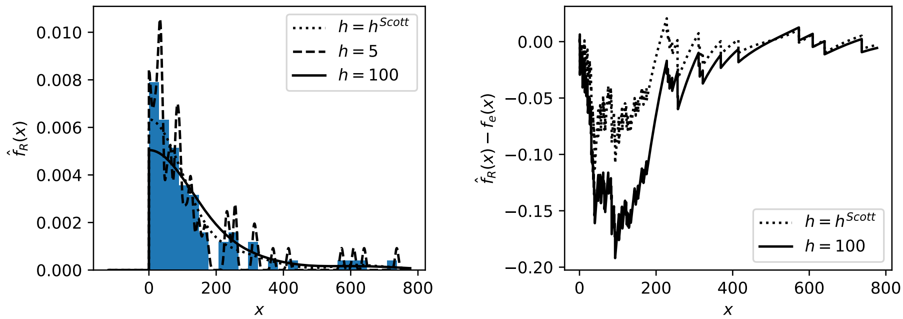
Clearly, $\hat{f}_R$ is non-negative, and it follows from the below that it is also a proper density function as the probability mass is 1

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

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

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

If we once again apply this to the suicide data, comparing to Figure 1.10 we see a big improvement near the boundary as shown in Figure 1.11a. However, we still proceed with the method originally presented in [?]. 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 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



(a)

(b)

Figure 1.11

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 1.11b. 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 1.1 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 1.1: $(\sqrt{n} + 0.12 + 0.11/\sqrt{n})\, D$

We now turn our attention to the boundary corrected KDE from [?]. They shortly described this in [?] 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, [?] 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 [?] 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 1.17.

To expand on the boundary corrected KDE to include an upper bound on the domain, we define the functions $a_m(x)$. Note that in [?], they define a_m as a

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

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

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

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

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

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 [?] as done in e.g. [?] 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 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 i'th kernel center is then given by

$$K_i^L(x) = \frac{1}{h} \frac{a_2(x) - a_1(x)\frac{x-x_i}{h}}{a_0(x)\, a_2(x) - a_1^2(x)} K\left(\frac{x-x_i}{h}\right) \tag{1.18}$$

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 $\int u\,\phi\left(u\right)\,du = -\phi\left(u\right) + C$

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

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

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

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 1.12a 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 1.12b 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 1.18 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 [?], 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 $a_0(x), K(x) > 0$. The non-negative boundary corrected KDE, denoted $\hat{f}_P(x)$ is then defined as

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

This $\hat{f}_P$ is also shown in Figure 1.12 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

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

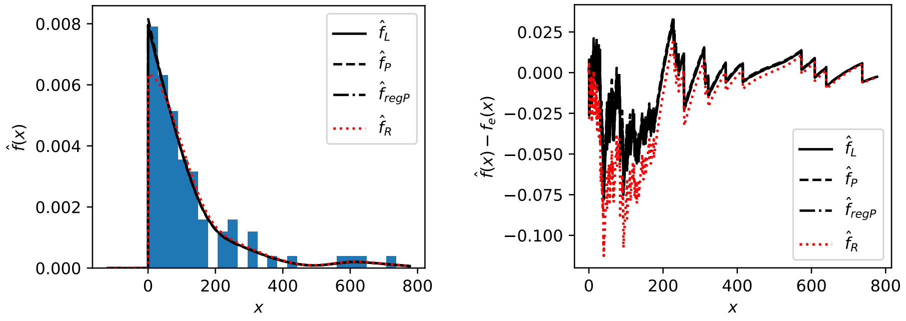
then, we introduce the regularizing parameters $\lambda \geq 0$ such that

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

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 1.12 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 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 [?], 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.



(a) **(b)**

Figure 1.12

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 [**?**]. 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 [**?**] [**?**] which shows great potential. Without going into too much detail, it works by considering the PDE

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

where $t = h^2$. The Gaussian kernel is then the unique solution when the domain is $\mathbb{R}$ with condition $\hat{f}[x;0] = \frac{1}{n}\sum_{i=1}^{n}\delta(x - x_i)$ 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

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

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

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

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 [?].

Example 1.2 (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 $F\left(x\right)=1-\left(1-x\right)^4$. Generating* 1000 *samples from the distribution and using the algorithm in [?] 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 1.13 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.*

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 **??**.

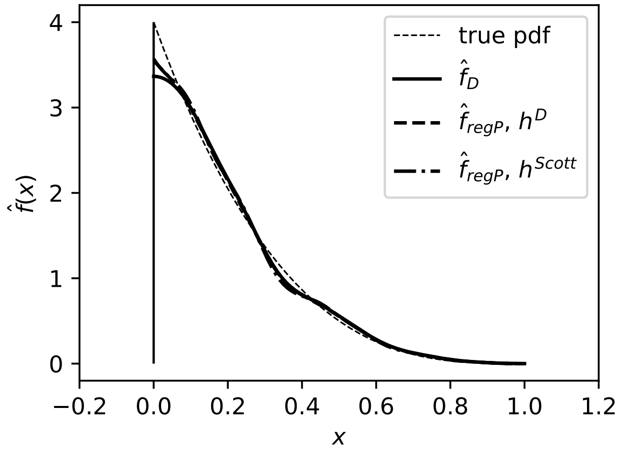


Figure 1.13: Using the diffusion based KDE on samples from a Beta $(1, 4)$ distribution, we see that in general it fits very well. Only at the boundary $x = 0$ there seems to be a problem which on the other hand, $\hat{f}_{regP}$ does not seem to suffer from to the same extend. In particular, repeating the experiment, $\hat{f}_{regP}$ is on average 4 at $x = 0$ while $\hat{f}_D$ constantly seem to undershoot.