IMPERIAL

MEng Individual Project

Imperial College London

Department of Computing

Scaling up synergy estimators

Author:
Vlad-Bogdan Coroian

Supervisor:
Dr. Pedro Mediano

Second Marker:
Dr. Tolga Birdal

June 19, 2024

Abstract

This project focuses on the concept of synergy, a key component of the Partial Information Decomposition framework, which captures the information generated collectively by a system, but not contained in any individual component.

To address the current limitations of our understanding of synergy, this project pursues two primary objectives: improving the computational efficiency of existing synergy calculation tools for discrete systems and extending the concept of synergy to continuous distributions. Our contributions include the development of `dccp-syndisc`, an algorithm that utilizes disciplined convex-concave programming to extend the scalability of synergy computations. This method allows for the analysis of systems up to twice the size previously supported, enabling the exploration of larger and more complex datasets. Through detailed case studies, we demonstrate the utility of our computational improvements. These applications validate our approach and highlight the potential of synergy measures to offer insights into the interactions of complex systems.

Additionally, we extend the theoretical framework of synergy to continuous distributions, by leveraging results from copula theory. This project provides a mathematical basis for quantifying dependency among continuous variables, as well as numerical methods and software that enable practical synergy calculations in continuous distributions.

Acknowledgements

I would like to express my profound gratitude to Dr. Pedro Mediano for his invaluable guidance and insights over the past year. I am immensely thankful for his constant support that transformed this project into the highlight of my academic journey.

I am also incredibly grateful to my wonderful friends, whose support has been at the core of my experience during this degree, making my time at university truly unforgettable.

Finally, I would like to thank my family, whose unwavering love and guidance have been the foundation of my achievements. They have been my greatest supporters from the very beginning, offering encouragement, advice, and the occasional much-needed comfort and motivation when things got really tough.

Contents

Chapter 1

Introduction

1.1 Motivation

In a world increasingly driven by data, understanding the nuances of how information is structured and communicated within complex systems is essential. Originating from the idea that a general theory for transmitting information can be formulated, Information Theory began as a way to optimise communication in noisy channels. It has since expanded into a broad framework for measuring information content in different systems, and the results in this field can now be used to gain insights from raw data in a systematic way [1].

While traditional information theory provides essential tools for the study of bivariate systems, the real-world complexity often involves interactions across multiple variables or behaviours that cannot be addressed by simple pairwise analysis. Therefore, extending the notion beyond a system with two components is a natural problem to consider. There have been a number of attempts to generalise the basic building blocks of information theory, trying to formalise abstract notions like the total dependency within a system [2] or the information embedded in various combinations of a set of variables [3].

One conceptual framework that offers an elegant way of describing information interactions is the Partial Information Decomposition (PID) framework [4]. This framework divides the contributions of information within a system into three types of atoms: redundant information that is contained in multiple components, unique information that belongs to exactly one individual component, and synergistic information that emerges collectively but cannot be attributed to any single component. This framework has been proven to be essential for understanding complex dependencies in fields such as neuroscience, where it can be used to explain neural interdependencies [5], machine learning, for better understanding the learning process [6], and data privacy, for assessing the security implications of information sharing [7].

Synergy is a fundamental concept of the PID framework, and captures the idea that the collective behaviour of a system can account to more than simply the sum of individual parts. To build intuition behind this concept, consider the phenomenon of depth perception: to perceive depth accurately, input from both eyes is required. If either eye's input is obstructed or lost, the majority of depth information is compromised, illustrating how the combined inputs produce a value (depth perception) that neither can fully achieve independently. This raises an intriguing question about the origin of the additional information that appears in the system.

While informal descriptions of synergy are insightful, they do not have operational utility on their own. Therefore, in this project, we use a formal definition of synergy [8], and build the research on a robust mathematical foundation, focusing on the probability distributions that generate data in a complex system.

In the case of discrete distributions, [8] provides a clear method for computing synergy associated with a system, and the companion software package `syndisc` provides functionality for practical applications. However, these tools are limited by computational inefficiencies that restrict their applicability to larger systems. This means synergy cannot be used yet as a way of extracting insights from medium and large-scale systems.

In the case of continuous distributions, there is currently no formal way to extend [8] to describe a synergy-centered information decomposition. For the completeness of this approach, introducing a continuous extension of the discrete measure would be an ideal enhancement.

1.2 Objectives

This project is driven by two primary objectives aimed at addressing the gaps in current methods of quantifying synergy. Therefore, the project can be divided into two parts, focusing on computational improvements and extensions of theoretical work:

1. **Computational improvements:** In this first part of the project, the main objective is to improve the scalability of algorithms used for computing synergy in discrete distributions. The current algorithms can only be used for small systems, and in order to use synergy to analyse real-world datasets, a more efficient way of computing synergy is required.

2. **Extension to continuous distributions:** The main objective for the second part of the project is to explore the extensions of synergy to continuous distributions. The formal definition of synergy can be extended naturally to continuous distributions, but the algorithms used in [8] cannot be applied in this case. Therefore, the main goals are to improve the understanding of synergy in continuous distributions and to offer a method for measuring this property in such cases.

1.3 Contributions

With this project, we aim to provide a deeper understanding of synergy-centered measures, and add improvements from both a computational and theoretical point of view. To this end, the contributions can be structured in the following way:

- **Computational optimisations in** `syndisc`: Leveraging convex optimization techniques, we have developed `dccp-syndisc`[1], an optimized version of the original synergy measurement algorithm. This optimisation improves the scalability of our computational tools significantly, allowing us to handle systems of up to twice the size previously manageable. We also provide a thorough evaluation of the performance, and discuss various trade-offs between accuracy and execution time.

- **Extending the synergy measure to continuous distributions**: Starting from the definition of synergy in [8], we provide a natural extension to real-valued continuous distributions. Using results from copula theory, we derive a formal way of expressing synergy. Additionally, we introduce approximations of these theoretical constructs, which enable practical, numerical estimation of synergy across diverse systems. Our study also includes numerical simulations that show the connections between synergy in discrete and continuous distributions.

- **Real-World Applications and Case Studies**: The improved scalability of `dccp-syndisc` allows us to explore and analyze synergy in much larger systems than were previously accessible. We apply this algorithm to a variety of real-world datasets, particularly those derived from neuroscience studies, to uncover and quantify complex interactions within these systems. Through detailed case studies, we use hypothesis testing to validate the presence of synergy in certain systems. These applications not only demonstrate the practical utility of our contributions, but also highlight the potential of using synergy measures in data analysis to provide deeper insights into the interactions in these systems.

[1] Implementation available at https://github.com/vladcoroian/dccp-syndisc.

Chapter 2

Background

2.1 Fundamentals of Information Theory

The field of Information Theory was developed with the aims of optimizing communication methods and solving the problem of reliable data transmission. Since its initial development, Information Theory has proved to be an essential tool to analyze a broad range of systems, extending its applications far beyond the study of communication channels. One of its fundamental contributions lies in quantifying information content and providing a mathematical framework for measuring the amount of uncertainty or surprise associated with a particular event. This section gives a brief overview of the fundamental measures of this field, as presented in [1].

2.1.1 Central measures of Information Theory

For the following definitions, consider X, Y, Z discrete random variables, with countable alphabets $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ respectively, and probability mass functions $p_X(x) = Pr\{X = x\}, \forall x \in \mathcal{X}, p_Y(y) = Pr\{Y = y\}, \forall y \in \mathcal{Y}, p_Z(z) = Pr\{Z = z\}, \forall z \in \mathcal{Z}$.

Definition 2.1.1 (Shannon information content)**.** *The Shannon information content (or surprisal) of an outcome $x \in \mathcal{X}$ is defined to be*

$$h(x) = \frac{1}{p_X(x)}.$$

Definition 2.1.2 (Entropy)**.** *The entropy associated with the random variable X is defined as the average Shannon information content:*

$$H(X) = \mathbb{E}[h(X)] = \sum_{x \in \mathcal{X}} p_X(x) \log \frac{1}{p_X(x)},$$

with the convention that for $p_X(x) = 0$, $0 \times \log 1/0 = 0$. For convenience, $H(X)$ may also be written as $H(\boldsymbol{p})$, where $\boldsymbol{p} = (p_1, p_2, \ldots, p_I)$ is a probability vector.

Intuitively, entropy represents the uncertainty in the outcome of a given random variable, the expected surprise given a distribution.

The choice of the base of the logarithm determines the unit of measure for entropy. In this project, the entropy will be measured in *bits*, using logarithm of base 2 (with $\log$ being a shorthand notation for $\log_2$).

Definition 2.1.3 (Joint Entropy)**.** *The joint entropy of two discrete random variables X, Y is defined to be:*

$$H(X, Y) = \sum_{x,y \in \mathcal{X} \times \mathcal{Y}} p(x, y) \log \frac{1}{p(x, y)},$$

where $p(x, y)$ is the probability mass function corresponding to the joint distribution (X, Y).

Definition 2.1.4 (Conditional Entropy)**.** *The conditional entropy of X conditioned on Y is given by:*

$$H(X|Y) = \sum_{y \in \mathcal{Y}} p_Y(y) H(X|Y = y),$$

where $H(X|Y = y) = -\sum_{x \in \mathcal{X}} p_X(x|y) \log p_X(x|y)$.

Observing the definitions above, one can note that $H(X, Y) = H(X|Y) + H(Y)$, which is called the chain rule for entropy.

Definition 2.1.5 (Mutual Information)**.** *The mutual information between X and Y is given by*

$$I(X;Y) = \sum_{x,y \in \mathcal{X} \times \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}.$$

Mutual information is a fundamental quantity in Information Theory, which quantifies how much information is shared between two variables. By computation, one observes that $I(X;Y) = H(X) + H(Y) - H(X,Y) = H(X) - H(X|Y)$. Intuitively, $I(X;Y)$ describes the reduction of uncertainty in X after knowing Y (or vice-versa, as mutual information is symmetric).

Definition 2.1.6 (Conditional Mutual Information)**.** *The mutual information between X and Y given Z is defined as*

$$I(X;Y|Z) = \sum_{x,y,z \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Z}} p(x,y,z) \log \frac{p(x,y|z)}{p(x|z)p(y|z)}.$$

Below the KL-divergence, a key concept in Information Theory, is introduced. Roughly, KL-divergence is type of statistical distance, and signals difference between two probability distributions. While not a distance, KL-divergence can also be used to compute projections from a distribution to a statistical manifold.

Definition 2.1.7 (KL-divergence)**.** *The relative entropy or the Kullback-Leibler divergence between two probability distributions P(x) and Q(x) (defined over the same alphabet) is:*

$$D_{KL}(P\|Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}.$$

2.1.2 Extending to continuous variables

These definitions can naturally be extended to continuous random variables, but one has to be cautious in order to preserve the key properties of these measures. The key idea behind extending the measures for continuous random variables X involves discretizing the state space in bins of equal width Δx and taking the limit as $\Delta x \to 0$. As mutual information is the key information-theoretic measure used in this project, the definition is shown below, while the rest can be extended similarly.

Definition 2.1.8 (Mutual Information in continuous variables)**.** *Let X and Y be continuous random variables with pdfs $f_X(x)$ and $f_Y(y)$ respectively. The mutual information between X and Y is defined as:*

$$I(X;Y) = \int_{x,y \in \mathcal{X} \times \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy.$$

It is important to note that in both the discrete and the continuous case, mutual information shared between two variables is a non-negative quantity.

As a final result, the following statement introduces an inequality between mutual information terms in a Markov chain.

Theorem 2.1.1 (Data Processing Inequality)**.** *Let X, Y, Z be random variables forming a Markov chain $X - Y - Z$. Then, the following inequality between the mutual information between X and Y and the mutual information between X and Z holds:*

$$I(X;Y) \geq I(X;Z).$$

2.1.3 Partial Information Decomposition

In this section, the Partial Information Decomposition (PID) framework [4] is introduced, which extends the notion of mutual information beyond the bivariate case, and proposes a way of decomposing information into a series of interpretable, non-negative terms.

Let W be a *target* random variable, and $\mathbf{X} = \{X_1, X_2, \dots X_n\}$ a set of n random variables denoted by *sources*. The goal of PID is to decompose $I(W; (X_1, X_2, \dots X_n))$ in terms of the partial information contributed by various subsets of $\mathbf{X}$. This decomposition contains three types of terms:

- *Unique information* that is provided by only one of the sources.
- *Redundant information* that is provided in each individual source.
- *Synergistic information* that is information the sources provide jointly, and not individually.

Consider the simplest case of such a system, composed of three random variables, target variable W and sources X_1, X_2. The following relations between the quantities described above hold:

$$\begin{aligned} I(W; X_1) &= \texttt{Red}(W; X_1, X_2) + \texttt{Un}(W; X_1) \\ I(W; X_2) &= \texttt{Red}(W; X_1, X_2) + \texttt{Un}(W; X_2) \\ I(W; X_1, X_2) &= \texttt{Red}(W; X_1, X_2) + \texttt{Un}(W; X_1) + \texttt{Un}(W; X_2) + \texttt{Syn}(W; X_1, X_2) \end{aligned} \tag{2.1}$$

The Partial Information Decomposition can be visualised by considering an analogy to set theory and matching the quantities that the different regions represent to PID terms as in the diagram below.

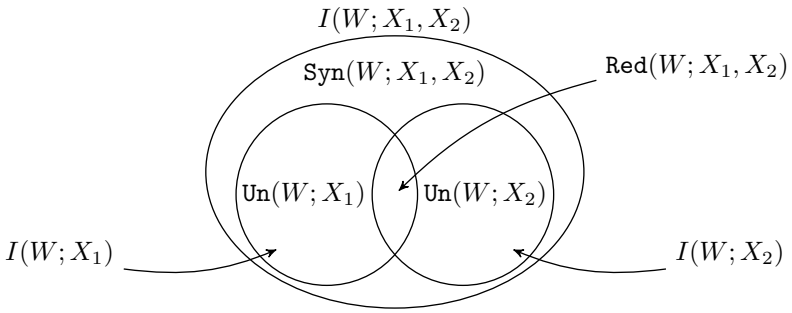


Figure 2.1: Structure of multivariate information for 3 variables [4]

The terms on the left-hand side in Eq. (2.1) can be easily computed using the standard definition of mutual information. This leaves four terms to be computed, representing the atoms giving the structure of multivariate information. On its own, PID simply describes the formal relationship between the basic building blocks of information, without providing further details on how to compute these terms. This makes the system above underdetermined, as there are four unknowns and just three equations.

The underdetermined nature of the system indicates that an extra concept is necessary to fully understand the decomposition of multivariate information. Providing a method to compute one of the terms involved implies that the other terms can be derived by solving the above system of equations. Thus, for this decomposition to be practical and useful, it is essential to have a method for calculating at least one of the terms.

Depending on the method used to transition to a determined system, the estimations of the atoms can vary, and the focus can shift to a specific block in the decomposition. Several methods to tackle the decomposition have been proposed, by providing methods to compute one of the terms (redundancy [4], synergy [9], unique information [10]), or alternative methods, such as those inspired from information geometry [11]. However, there is no consensus on a universally accepted method for quantifying these terms.

In this project, not all terms in the decomposition are studied, and the focus of the project is solely on the synergy term. The project uses a synergy-first approach, and a concrete, operational meaning of synergy is used. Related work is reviewed in the next section.

2.2 Synergy in discrete random variables

In this section, the main starting point for this project, a synergy-centered information decomposition, rooted in data privacy field [12] is introduced. Through this approach, synergistic information is viewed as the information that can be disclosed about a system without revealing the state of any of its individual components. The focus is on this approach specifically, as it offers a way of operating with synergy and using it as a measure for analysing the efficiency of systems. Next, a brief overview of the scenario and the main results presented in [8, 12] are presented.

2.2.1 Initial scenario

Suppose that we are provided with a dataset $\mathbf{X} = \{X_1, X_2, \dots X_n\}$ which should be kept private, and a variable of interest W correlated with the dataset, that a user wishes to share. For example, X may represent sensitive measurements of a patient's vital signals, while W may be a unique health indicator. Although it is ideal for the patient to disclose W for emergency alerts, the actual data samples should remain confidential to prevent unintentional disclosure of personal information.

The privacy framework designed to prevent inference attacks proposes to share another variable Y, obtained through a mapping of X. It is also essential to maintain perfect sample privacy, meaning that Y should not provide any information that enables statistical inference on individual components $X_i, \forall i$. Hence, we require that for all i, $X_i \perp\!\!\!\perp Y$ (X_i and Y are statistically independent), as well as $W - X - Y$ forming a Markov chain (given dataset X, W and Y need to be independent). A natural measure for the quality of the estimation Y provides for W is the mutual information shared between Y and W, $I(Y;W)$ [12].

Note that while Y is independent of any individual component, it might still be globally interdependent of the rest of the system.

Example 2.2.1 (Exclusive OR). *Let $X = (X_1, X_2)$ be two independent fair coins. Then, the random variable $Y = X_1 \oplus X_2$ (XOR operation), given by*

$$X_1 \oplus X_2 = \begin{cases} 0 & if\ X_1 = X_2 \\ 1 & if\ X_1 \neq X_2 \end{cases},$$

is independent of each of them, while $I(Y; X_1, X_2) > 0$. Therefore, Y reveals a collective property (whether entries are equal or not), while not revealing any information about X_1 or X_2.

Notation and preliminaries

Throughout the project, the notation from [12, 8] is used, and it is described here again, for convenience.

Capital letters denote random variables, lower case letters denote the realisation of these random variables, and capital letters in calligraphic font denote the alphabets of random variables. Matrices and vectors are denoted by letters in bold (capital for matrices and lower case for vectors).

"For integers $m \leq n$, we define the discrete interval $[m : n] \triangleq \{m, m+1, \dots, n\}$. For an integer $n \geq 1$, $\mathbf{1}_n$ denotes an n-dimensional all-one column vector. For a random variable $X \in \mathcal{X}$, with finite $|\mathcal{X}|$, the probability simplex $\mathcal{P}(\mathcal{X})$ is the standard $(|\mathcal{X}| - 1)-$simplex given by

$$\mathcal{P}(\mathcal{X}) = \{\, \mathbf{v} \in \mathbb{R}^{|\mathcal{X}|} \mid \mathbf{1}_{|\mathcal{X}|}^T \cdot \mathbf{v} = 1, v_i \geq 0, \forall i \in [1 : |\mathcal{X}|] \,\}.$$

To each probability mass function (pmf) on X, denoted by $p_X(\cdot)$ (or written simply p_X), corresponds a probability vector $p_X \in \mathcal{P}(\mathcal{X})$, whose i-th element is $p_X(x_i)$ ($i \in [1 : |\mathcal{X}|]$). Likewise, for a pair of random variables (X, Y) with joint pmf $p_{X,Y}$, the probability vector $\mathbf{p}_{X|y}$ corresponds to the conditional pmf $p_{X|Y}(\cdot|y), \forall y \in \mathcal{Y}$, and $\mathbf{P}_{X|Y}$ is an $|\mathcal{X}| \times |\mathcal{Y}|$ matrix with columns $\mathbf{p}_{X|y}, \forall y \in \mathcal{Y}$" [12].

Consider $W, X_1, \dots, X_n$ discrete random variables (with finite alphabets) and a given joint distribution $p_{W,X_1,\dots,X_n}$. Let $X \triangleq (X_1, \dots, X_n)$, and consider the alphabet $\mathcal{X}$ given by:

$$\mathcal{X} = \{\, (x_1, \dots, x_n) \in \prod_{i=1}^{n} \mathcal{X}_i \mid p_X(x_1, \dots x_n) > 0 \,\}.$$

The set of *admissible stochastic mappings* from the dataset X to $\mathcal{Y}$ can be defined as

$$\mathcal{A}_X = \{\, p_{Y|X} \mid Y \perp\!\!\!\perp X_i, \forall i \in [1:n] \,\}.$$

The quantity needed to be maximised is the information that is shared between W and Y, under perfect sample privacy. This can be formalised through the *private disclosure capacity*, which is defined as

$$I_s \triangleq \max_{\substack{p_{Y|X} \in \mathcal{A}_X \\ W-X-Y}} I(W;Y).$$

A random variable Y satisfying the constraints $p_{Y|X} \in \mathcal{A}_X$ and $W-X-Y$ Markov chain is called a *private disclosure channel*. The probability distribution $p_{Y|X}$ which maximises the expression above is denoted by $p^*_{Y|X}$.

The binary matrix $\mathbf{P}$ is also defined as: $\mathbf{P} = \begin{bmatrix} \mathbf{P}_{X_1|X} \\ \vdots \\ \mathbf{P}_{X_n|X} \end{bmatrix}_{G \times |\mathcal{X}|}$, where $G \triangleq \sum_{i=1}^{n} |\mathcal{X}_i|$. Note that the probabilities in $\mathbf{P}$ are either 0 or 1, as X_i is uniquely defined by X.

2.2.2 Extending the notion of synergy

In the previous sections, through the disclosed variable Y, the privacy of the individual components of the system is maintained. This notion can be extended to preserve the privacy of specific subsystems of the initial dataset X. In [8], an extension of this approach is presented, where the private channel is chosen such that a specific partition of the system remains private.

A subsystem of X is represented as $\alpha = \{n_1, \dots n_k\} \subset [n]$ and corresponds to the set of components $X^{\boldsymbol{\alpha}} = (X_{n_1}, X_{n_2}, \dots X_{n_k})$. Similar to the previous definition of a private disclosure channel, a synergistic channel with respect to a subsystem $\{\alpha_1, \dots, \alpha_L\}$ is given by $p_{Y|X}$, where Y is chosen such that $Y \perp\!\!\!\perp X^{\alpha_i}, \forall i \in [L]$.

2.2.3 Finding the optimal mapping

This section is an overview of the main results in [12], and describes a practical method for computing the optimal mapping under perfect sample privacy.

The following theorem represents the key starting point for this project, as it provides a concrete algorithm for computing the optimal mapping and proves its correctness.

Theorem 2.2.1. *The maximizer for the private disclosure capacity can be obtained as a solution to a standard linear program.*

Sketch of proof. Consider the singular value decomposition of $\mathbf{P}$, which gives $\mathbf{P} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$. Here, $\mathbf{V}$ is the matrix of right eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{|\mathcal{X}|}$ (where only the first rank($\mathbf{P}$) correspond to positive eigenvalues). Define $\mathbf{A} \triangleq [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{\text{rank}(\mathbf{P})}]^T$ and the convex polytope in $\mathcal{P}(\mathcal{X})$,

$$\mathbb{S} \triangleq \{\, \mathbf{t} \in \mathbb{R}^{|\mathcal{X}|} \mid \mathbf{At} = \mathbf{Ap}_X, \mathbf{t} \geq 0 \,\}.$$

Since Null($\mathbf{A}$) = Null($\mathbf{P}$) and $W - X - Y$ is a Markov chain, the following property which links elements of $\mathcal{A}_X$ to elements of $\mathbb{S}$ holds:

$$W - X - Y, p_{Y|X} \in \mathcal{A}_X \iff \mathbf{p}_{X|y} \in \mathbb{S}, \forall y \in \mathcal{Y}.$$

Using the property above, I_s can be rewritten as

$$\begin{aligned} I_s &= H(W) - \min_{\substack{p_{Y|X} \in \mathcal{A}_X \\ W-X-Y}} H(W|Y) \\ &= H(W) - \min_{\substack{p_Y(\cdot), \mathbf{p}_{X|y} \in \mathbb{S}, \forall y \in \mathcal{Y} \\ \sum_y p_Y(y)\mathbf{p}_{X|y} = \mathbf{p}_X}} \sum_y p_Y(y) H(\mathbf{P}_{W|X}\mathbf{p}_{X|y}). \end{aligned} \tag{2.2}$$

This property rephrases the original minimisation problem over all targets Y (so on $\mathcal{A}_X$) to a minimisation on elements of $\mathbb{S}$, where the geometrical aspects of the problem can be exploited.

By noticing that every point in $\mathbb{S}$ can be written as a linear combination of the extreme points of the convex polytope and using the concavity of entropy, it is sufficient to consider only the extreme points in the minimisation.

The problem is now equivalent to a standard LP:

$$\begin{aligned} I_s = H(W) - \min_{\mathbf{u}\geq 0} \left[H(\mathbf{P}_{W|X}\mathbf{p}_1)\dots H(\mathbf{P}_{W|X}\mathbf{p}_K)\right]\cdot\mathbf{u} \\ \text{subject to } [\mathbf{p}_1\ \mathbf{p}_2\dots\ \mathbf{p}_K]\cdot\mathbf{u} = \mathbf{p}_X, \end{aligned} \tag{2.3}$$

where $\mathbf{u}$ is a $K-$dimensional weight vector, and $\mathbf{p}_1, \mathbf{p}_2 \dots, \mathbf{p}_K$ are the extreme points of $\mathbb{S}$. Note that the elements of $\mathbf{u}$ correspond to the coefficients in the linear combination based on the extreme points, and the constraint $\mathbf{1}_K^T \cdot \mathbf{u} = 1$ is implicitly satisfied if the constraint in the LP is satisfied. □

2.2.4 Algorithm

Using the theorem above, the problem can be solved in a two-step algorithm. First the extreme points of $\mathbb{S}$ are computed, then linear programming is used to obtain the optimal mapping. The following algorithms are taken from [13] and can be used to obtain the optimal mapping:

Algorithm 1: Finding the extreme points of $\mathbb{S}$

function `FindExtremePoints`($\boldsymbol{p}_X$)**:**
  $\mathbf{P}$ = `BuildBinaryMatrix` ($\mathbf{p}_X$)
  $\mathbf{U}, \mathbf{\Sigma}, [\mathbf{v}_1, \dots, \mathbf{v}_{|\mathcal{X}|}]^T$ = `SVD` ($\mathbf{P}$)
  $\mathbf{A} = [\mathbf{v}_1, \dots, \mathbf{v}_{\mathrm{rank}(\mathbf{\Sigma})}]^T$
  $\mathbf{b} = \mathbf{A}\mathbf{p}_X$
  K = 0
  $\mathcal{B}_1, \dots \mathcal{B}_J$ = All subsets of $\{1, \dots, |\mathcal{X}|\}$ with $\mathrm{rank}(\mathbf{\Sigma})$ elements
  for $j = 1 \dots J$ **do**
    `/* Let` $\mathtt{A}_{\mathcal{B}}$ `be a` $\mathrm{rank}(\mathtt{P}) \times \mathrm{rank}(\mathtt{P})$ `matrix whose columns are the columns of A indexed by the indices in` $\mathcal{B}$ `*/`
    if $\boldsymbol{A}_{\mathcal{B}_j}^{-1}\boldsymbol{b} \geq 0$ **then**
      K = K + 1
      $\mathbf{p}_K = [\mathbf{A}_{\mathcal{B}_j}^{-1}\mathbf{b}, \mathbf{0}_{\mathrm{nul}(\mathbf{P})}]^T$
    end
  end
  return $\boldsymbol{p}_1, \dots \boldsymbol{p}_K$

Using the algorithm for finding extreme points in the polytope, the initial problem can be turned into a standard linear program, which can be easily solved using traditional optimisation techniques.

Algorithm 2: Computing the optimal disclosure mapping $p^*_{Y|X}$.

function `FindOptimalDisclosureMapping`($\boldsymbol{P}_{W|X}, \boldsymbol{p}_X$)**:**
  $\mathbf{p}_1, \dots \mathbf{p}_K$ = `FindExtremePoints` ($\mathbf{p}_X$)
  Find $\mathbf{u}^* = \mathrm{argmin} \sum_{k=1}^{K} u_k H(\mathbf{P}_{W|X}\mathbf{p}_k)$ subject to $[\mathbf{p}_1\ \mathbf{p}_2 \dots\ \mathbf{p}_K]\cdot\mathbf{u} = \mathbf{p}_X$
  $\mathbf{p}_Y = [u_{i_1}, \dots u_{i_L}]$
  `/* where` $i_1, \dots, i_L$ `are the indices of the nonzero elements of u */`
  $\mathbf{P}_{X|Y} = [\mathbf{p}_{X|Y=i_1}\ \mathbf{p}_{X|Y=i_2} \cdots\ \mathbf{p}_{X|Y=i_L}]$
  $\mathbf{P}_{Y|X} = \mathrm{diag}(\mathbf{p}_Y)\mathbf{P}_{X|Y}^T \mathrm{diag}(\mathbf{p}_X)^{-1}$
  return $\boldsymbol{P}_{Y|X}$

The Python package `syndisc` is the companion software of [8]. The package provides utility functions that can be used to compute synergy in discrete distributions based on the algorithm presented above. While useful in the analysis of modest systems of random variables, the complexity of the computations makes the package impractical for extracting insights from medium and large-scale systems. More precisely, it cannot provide insights for setups that use 6 or more sources, such as $X = (X_1, \dots, X_6)$, where each X_i represents an independent fair coin, and the target variable is $W = f(X_1, \dots X_6)$ (for instance, $W = \bigwedge_{i=1}^{6} X_i$). The computational complexity involved makes it challenging to handle these kinds of scenarios effectively.

Note that even though the algorithms above can be used to compute the optimal mapping, the complexity of the computations required grows very quickly as the dataset X increases in size. The main bottleneck in the computation is the search over all subsets of columns of $\mathbf{A}$ which is needed for computing the extreme points in $\mathbb{S}$. This is one of the main issues the project tries to address.

Remark 2.2.1. *It is important to recognize that the task of identifying synergy within a system fundamentally involves an optimization problem. This is due to the definition of I_s, the synergy of a system, which is calculated as the supremum over the set of private disclosure channels, which represents the feasible set. Additionally, the structure of the objective function is favorable, as this consists of smooth functions with known convexity. Hence, this scenario is well-suited for the application of optimization principles, particularly the methods of convex optimization.*

2.3 Convex Optimisation

Convex optimisation studies the problem of minimising convex functions subject to a set of equality and inequality constraints and maintains many of the useful theoretical properties of predecessors, such as linear programming. Many optimisation problems can be formulated or approximated using convex programs, making this method suitable for a broad range of problems.

In this section, some of the main numerical algorithms used for solving convex optimisation models are introduced, as well as a methodology that can be used to construct models which allow much of the work to be automated.

With these methods, the aim is to formulate the problem of computing discrete synergy as a convex optimisation problem and then use one of the efficient numerical algorithms (implemented in commercial solvers such as GUROBI [14]) to approximate the private disclosure capacity and find appropriate mappings $p_{Y|X}$.

The basics of convex optimisation are reviewed below, as presented in [15].

2.3.1 Convex programming

Definition 2.3.1 (Convex Set)**.** *A set C is convex if the line segment between any two points in C lies in C, i.e., if for any $x_1, x_2 \in C$ and any $\theta \in [0,1]$, $\theta x_1 + (1-\theta)x_2 \in C$.*

Definition 2.3.2 (Convex function)**.** *A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $\mathbf{dom}\, f$ is a convex set and if $\forall x, y \in \mathbf{dom}\, f$, and $\theta \in [0,1]$, $f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$.*

A function f is strictly convex if strict inequality holds in whenever $x \neq y$ and $\theta \in (0,1)$. The function f is concave if $-f$ is convex, and strictly concave if $-f$ is strictly convex.

A convex program is an optimisation problem of the following form

$$\begin{aligned} &\text{minimize } f_0(x) \\ &\text{subject to } f_i(x) \leq 0, i = 1, \dots m, \\ &\qquad\qquad h_j(x) = 0, j = 1, \dots p, \end{aligned} \tag{2.4}$$

where $x \in \mathbb{R}^n$. The variable x is called the *optimisation variable*, $f_0 : \mathbb{R}^n \to \mathbb{R}$ is called *objective* function, $f_1, \dots f_m$ are *inequality constraints* and $h_1 \dots h_p$ are the *equality constraints*. A point x is called *feasible* if it satisfies all the equality and inequality constraints. The problem is a *convex* optimisation problem if $f_0, f_1, \dots f_m$ are convex and $h_0, \dots h_p$ are affine (linear function plus constant) [15].

2.3.2 Numerical algorithms

Numerical algorithms designed to solve convex optimisation problems often use various transformations to reduce the initial problem to an equivalent problem suitable for finding a numerical solution.

Interior-point methods

Interior-point methods remove the inequality constraints, by adding another term in the objective function which penalizes constraint violations. More precisely, this method involves substituting the set of inequality constraints $\mathcal{S} \triangleq \{ x \in \mathbb{R}^n \mid f_i(x) \leq 0,\ i = 1, \dots m \}$ with a twice differentiable convex barrier function $\phi \colon \mathbb{R}^n \to \mathbb{R}$, with $\mathbf{dom}\, \phi = \text{Int}\mathcal{S}$. This transformation results in a modified optimisation problem:

$$\begin{aligned} &\text{minimize } f_0(x) + t\phi(x) \\ &\text{subject to } h_j(x) = 0, j = 1, \dots p. \end{aligned}$$

As $t \to 0$, the solution for this adjusted problem converges to the solution of the original problem (under mild conditions). Each iteration performs a Newton minimisation step and reduces the value of t [16]. A variant of this algorithm called "primal-dual interior-point method" is used in many commercial solvers such as GUROBI.

First-order methods

First-order methods are iterative optimisation methods that use only first-order information about the objective function (gradients, subgradients and function value). This restriction allows them to have a small cost per iteration and they scale well to large problems.

Consider now the equality-constrained convex optimisation problem with the following form:

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } Ax = b \end{aligned} \tag{2.5}$$

where $x \in \mathbb{R}^n, A \in \mathbb{R}^{m \times n}$ and $f \colon \mathbb{R}^n \to \mathbb{R}$ is convex. The dual ascent method is an iterative method that takes advantage of the fact that the Lagrange dual problem is always convex and uses gradient ascent on the dual variable to find the optimal value of problem (2.5).

Starting from an initial guess, at every iteration it takes a small step in the direction of the gradient and then repeats the process. As long as the step size is chosen appropriately and some assumptions on the structure of the problem hold (KKT conditions), the algorithm converges to an optimal point.

2.3.3 Disciplined convex-concave programming

Convex optimization provides a powerful set of algorithms that can be used to optimise a certain class of functions. To implement these techniques programmatically and carry out numerical optimization, using a commercial solver is the ideal choice, due to the superior performance these tools have. However, these solvers often require framing the problem in a specific format dictated by the solver, which can be challenging.

Moreover, these optimization methods might not be directly applicable if the function lacks the necessary curvature. However, when dealing with sufficiently well-behaved objective functions, there are still strategies available to appropriately reformulate the problem, and take advantage of the convex optimisation techniques.

In this section, the `CVXPY` modeling language and disciplined convex-concave programming are introduced, which can be used to overcome these challenges encountered in convex optimisation.

Disciplined convex programming and `CVXPY`

`CVXPY` [17] is a Python-Embedded modeling language for convex optimisation. Phrasing the problem in the restrictive language imposed by certain solvers is challenging, and `CVXPY` allows the user to express the problem in a natural syntax and automates the process of translation. Moreover, if a problem is written in the `CVXPY` format many useful properties can be automatically verified. `CVXPY` uses a simple, readable syntax and greatly simplifies the process of preparing a problem for optimisation using a commercial solver.

Example 2.3.1 (Solving a system of equations [17]). *Given $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, the system of equations $Ax = b$ can be formulated in `CVXPY` as:*

```
# Construct the problem.
x = Variable(n)
objective = Minimize(sum_squares(A*x - b))
constraints = [0 <= x, x <= 1]
prob = Problem(objective, constraints)

# Solve the problem.
result = prob.solve()

# Output the optimal value.
print(x.value)
```

A restriction of `CVXPY` is that, in its initial form, `CVXPY` uses disciplined convex programming (DCP) to verify problem convexity. This means that in order to verify convexity of the problem, `CVXPY` uses a fixed set of functions for which the convexity (constant, affine, convex or concave) and monotonicity is known, which are then combined to produce new expressions. To allow the automation process of rewriting the problem, DCP does not support all convex optimisation problems. DCP allows constructing expressions where checking the curvature is fairly straightforward (e.g. sum of convex functions or composition between increasing functions and convex functions). Notably, a limitation of DCP is the "*No-product rule*", as it cannot accurately check the convexity of products of expressions. Also, the only problem types supported by DCP are: minimising a convex objective (with zero or more convex constraints), maximising a concave objective (with zero or more convex constraints) and feasibility problems with no objective and one or more convex constraints [16].

Convex-concave procedure

The convex-concave procedure (CCP) [18] is a heuristic that can be used to find local optimal points for difference of convex (DC) problems. DC problems are problems of the form:

$$\begin{aligned} &\text{minimize } f_0(x) - g_0(x) \\ &\text{subject to } f_i(x) - g_i(x) \leq 0, i = 1 \dots m \end{aligned} \tag{2.6}$$

where $x \in \mathbb{R}^n$ is the optimization variable, and the functions $f_i, g_i \colon \mathbb{R}^n \to \mathbb{R}$ are convex. This heuristic finds a local minimum by replacing the concave terms with a convex upper bound and solving the associated convex optimisation problem (which is a restriction of the original problem). By including both $f_i(x) - g_i(x) \leq 0$ and $g_i(x) - f_i(x) \leq 0$ as constraints, these problems can also model equality constraints. Note that the problem (2.6) becomes a convex optimization problem in the case where when g_i are all affine [18].

Note that CCP is a heuristic method, and the solutions obtained through this approach are local optimal points, but they may be far from the global optimal point.

Disciplined convex-concave programming

Disciplined convex-concave programming (DCCP) [19] combines concepts from DCP with the CCP heuristic, and offers a structured way to express a large range of optimisation problems that do not fit in the category of problems supported through DCP. In addition to the problems DCP supported, DCCP adds support for difference of convex problems. In particular, `DCCP` makes it possible to try to solve programmatically the following set of problems:

$$\begin{aligned} &\text{minimize/maximize } o(x) \\ &\text{subject to } l_i(x) \sim r_i(x), i = 1 \dots m \end{aligned}$$

where o, l_i, and r_i are functions of x with curvature that can be verified by the DCP rules, and the operator $\sim$ denotes a comparison operator $(=, \leq, \geq)$.

The methods proposed in [19] have been implemented as a Python package `DCCP`, which extends `CVXPY`. When `solve` is used with parameter `method='dccp'`, `DCCP` first verifies that all functions involved satisfy the DCP rules and then transforms the problem into an equivalent DCP problem, which is subsequently translated by `CVXPY`, and sent to the commercial solver for optimisation.

In particular, `DCCP` supports optimisation problems that look to minimise a concave function over a convex set. Therefore, these optimisation technique seems to be a promising avenue to explore further. The function needed to be optimized in the case of synergy is a concave function, and minimizing this kind of functions is supported by commercial solvers through `DCCP`.

Chapter 3

Scaling synergy in discrete distribution

In this chapter, we focus on the synergy in discrete distributions and we present the extensions added to the algorithm described in [12] for finding the optimal disclosure mapping. By taking advantage of convex optimisation techniques, the optimised software provides useful estimations for much larger systems. We also evaluate the optimised algorithm on a range of synthetic and real-world data and present various applications in analysing neuroscience datasets.

3.1 Formulating the optimisation problem

Throughout this chapter, we will be working with the system presented in section 2.2.1. The setup consists of a set of discrete random variables: the sources $\mathbf{X} = \{X_1, X_2, \dots X_n\}$ and a variable of interest W. The goal is to share another discrete random variable Y, obtained as a map of the dataset $\mathbf{X}$, which shares as much information about W as possible, while keeping the dataset private.

Formally, the aim is to maximise the mutual information between W and Y, where Y is an admissible stochastic mapping of the dataset $\mathbf{X}$ and the Markov condition $W - X - Y$ is satisfied. Considering the equivalent formulation of Eq. (2.2), the expression for computing the optimal mapping in the discrete case can be reduced to the following optimisation problem:

$$\begin{aligned} &\text{minimise} \quad \sum_{y \in \mathcal{Y}} p_Y(y) H(\mathbf{P}_{W|X} \mathbf{p}_{X|y}) \\ &\text{subject to } p_Y(\cdot) \text{ - probability distribution,} \\ &\qquad \forall y \in \mathcal{Y},\ \mathbf{p}_{X|y} \in \mathbb{S}, \sum_{y \in \mathcal{Y}} p_Y(y) \mathbf{p}_{X|y} = \mathbf{p}_X \end{aligned} \tag{3.1}$$

In the rephrasing above, the objective function $\sum_{y \in \mathcal{Y}} p_Y(y) H(\mathbf{P}_{W|X} \mathbf{p}_{X|y})$ is optimised over the set of probability distributions p_Y and the set of extreme points of the polytope $\mathbb{S}$. Note that this optimisation problem also involves setting the optimal alphabet $\mathcal{Y}$.

The optimal disclosure mapping to be computed is $p_{Y|X}$, and this can easily be constructed starting from the optimal solution of problem (3.1) using Bayes' rule. The variable Y that minimizes the optimization problem above, also maximizes the mutual information $I(W;Y)$, and is the optimal disclosure mapping. Therefore, solving the optimisation problem above also gives the optimal disclosure mapping.

3.2 Two step optimisation algorithm

3.2.1 Challenges with brute force optimisation

Formulating the optimisation problem as above provides a great advantage, as it becomes well suited for convex optimisation methods. These methods are known to be reliable and efficient, so

this approach can provide scaling opportunities without compromising on accuracy. The problem can now be expressed in a convenient way to allow convex optimisation solvers to apply various techniques programmatically and optimise the objective very efficiently.

However, one of the main challenges involved when using such a solver is the restrictive language required by the program. Before giving the problem as input data for a solver, problem (3.1) needs to be modelled in a more specific way. In particular, we wish to make the problem compatible with the disciplined convex-concave programming framework.

Note that the problem is not convex if taken in this initial form, since the products of variables in the last constraint break the convexity of the feasible set. As much more efficient methods exist for solving convex optimisation problems, it can be useful to rephrase the problem in a way that allows the solution to be found through these methods.

A numerical example is provided below and shows that the problem is not convex in its initial form.

Example 3.2.1. *Consider two independent fair coins X_1, X_2, and target $W = X_1 \wedge X_2$. Formally, the system is comprised of two binary random variables and $p_{\mathbf{X}} = [0.25, 0.25, 0.25, 0.25]^T$. The associated matrix A can be computed and is equal to:*

$$A = \begin{bmatrix} -0.5 & -0.5 & -0.5 & -0.5 \\ 0.71 & 0. & 0. & -0.71 \\ 0. & 0.71 & -0.71 & 0. \end{bmatrix}.$$

For $p_Y(\cdot)$ a vector with three elements, the following two points $(p_Y, \{\mathbf{p}_{X|y}, y \in \mathcal{Y}\})$ are in the feasible set of the optimization problem:

$$p_1 = \left([0.5, 0.5, 0], \left([0, 0.5, 0.5, 0]^T, [0.5, 0, 0, 0.5]^T, [0, 0.5, 0.5, 0]^T\right)\right)$$

$$p_2 = \left([0.5, 0, 0.5], \left([0, 0.5, 0.5, 0]^T, [0.5, 0, 0, 0.5]^T, [0.5, 0, 0, 0.5]^T\right)\right).$$

However, the linear combination $\lambda p_1 + (1-\lambda)p_2$ for $\lambda = 0.5$ gives the point:

$$\left([0.5, 0.25, 0.25], \left([0, 0.5, 0.5, 0]^T, [0.5, 0, 0, 0.5]^T, [0.25, 0.25, 0.25, 0.25]^T\right)\right),$$

which is not in the feasible set, as it gives

$$\sum_{i=1}^{3} p_Y(y_i)\mathbf{p}_{X|y_i} = [0.1875, 0.3125, 0.3125, 0.1875]^T \neq p_{\mathbf{X}}.$$

The example illustrates that the product of variables in the consistency condition, given by the sum of the conditional distributions $\sum_{y\in\mathcal{Y}} p_Y(y)\mathbf{p}_{X|y} = \mathbf{p}_X$, can break the convexity of the feasible set. Thus, it is necessary to find a strategy to linearize the constraints and eliminate the variable products in the constraints.

3.2.2 Making the problem convex

In problem (3.1), the optimization variables can be categorized into two distinct groups: the elements of the probability distribution p_Y and the probability distributions $\mathbf{p}_{X|y}$, which represent extreme points in the polytope $\mathbb{S}$. The challenge regarding the convexity of problem (3.1) arises from the presence of product terms in both the objective function and the constraints. However, we observe that these products exclusively involve variables from these two separate sets.

The observation above motivates using a coordinate descent [20] algorithm for tackling the optimisation problem: we can sequentially optimize over one set of variables while keeping the other fixed. By fixing one set of variables, the problem simplifies into a convex form and can be naturally formulated in DCCP format.

We note that the difference in size between the sets is significant and grows linearly with the size of states of the source set. For a fixed $\mathcal{Y}$, p_Y has $|\mathcal{Y}|$ elements, while the extreme points in the polytope contain a total of $|\mathcal{X}| \cdot |\mathcal{Y}|$ variables. Consequently, fixing p_Y in the first step will likely have less influence on the optimal solution, given its smaller size relative to the set of extreme points, so we decided to fix this one first. Initially, lacking additional information about the optimal solution, the assigned values for p_Y may be arbitrary. However, they should form a

valid probability distribution, so that the optimal values obtained for $\{ \mathbf{p}_{X|y} \mid y \in \mathcal{Y} \}$ together with p_Y form a feasible solution.

Two simplified versions of the optimisation problem (3.1) are considered, in which the optimisation over the two sets of variables happens in turns.

Polytope optimisation step

Note that in the equation below, the optimisation is done only on points in the polytope $\mathbb{S}$, and p_Y represents a fixed set of coefficients.

$$\begin{aligned} &\underset{\mathbf{p}_{X|y}}{\text{minimise}} \quad \sum_{y \in \mathcal{Y}} p_Y(y) H(\mathbf{P}_{W|X} \mathbf{p}_{X|y}) \\ &\text{subject to } \forall y \in \mathcal{Y},\ \mathbf{p}_{X|y} \in \mathbb{S}, \sum_{y \in \mathcal{Y}} p_Y(y) \mathbf{p}_{X|y} = \mathbf{p}_X \end{aligned} \tag{3.2}$$

The objective in problem (3.2) is a concave function. Also, $\mathbb{S} = \{ \mathbf{t} \in \mathbb{R}^{|\mathcal{X}|} \mid \mathbf{At} = \mathbf{Ap}_X, \mathbf{t} \geq 0 \}$ is defined as the set of points satisfying a linear constraint and the last equality is linear, so all the constraints are affine. This means the curvature of the objective and constraints can be verified by the DCP rules and the problem is compatible with DCCP.

Distribution optimisation step

The next step in the optimisation is finding the optimal distribution p_Y. For this step, the optimisation is done only on terms $p_Y(y), y \in \mathcal{Y}$. The terms $\mathbf{p}_{X|y}$ are fixed for all $y \in \mathcal{Y}$, so the terms of the form $H(\mathbf{P}_{W|X} \mathbf{p}_{X|y})$ in the objective function are fixed and can be computed beforehand.

$$\begin{aligned} &\underset{p_Y(y)}{\text{minimise}} \quad \sum_{y \in \mathcal{Y}} p_Y(y) H(\mathbf{P}_{W|X} \mathbf{p}_{X|y}) \\ &\text{subject to } p_Y(\cdot) \text{ - probability distribution,} \\ &\qquad \sum_{y \in \mathcal{Y}} p_Y(y) \mathbf{p}_{X|y} = \mathbf{p}_X \end{aligned} \tag{3.3}$$

The objective and constraints in the equation above are linear in p_Y, so problem (3.3) becomes a standard linear program [15].

Splitting the initial optimisation in two subproblems, problem (3.2) and problem (3.3), makes the scenario suitable for the convex optimisation techniques described in section 2.3. Specifically, DCCP methods can be applied to solve the convex-concave problem (3.2), while problem (3.3) can be solved using standard optimisation techniques.

3.2.3 Iterative algorithm

We can now describe the iterative algorithm used to find the optimal disclosure mapping.

One iteration algorithm

The algorithm below describes one iteration of the optimisation using DCCP. In the two simplified versions of the main optimisation problem, we build a list of constraints corresponding to each problem and then minimise the objective function using disciplined convex-concave programming. Leveraging the techniques used for convex optimisation, this technique is able to effectively find local optimum points, which can be considered lower bounds for the optimal value of problem (3.1) (as these are feasible solutions). As we present later in the evaluation section, this method often achieves good estimates of the maximal mutual information available.

The functions `DccpOptimise` and `LinearProgramOptimise` represent calls to `CVXPY`, that translate the problem and give it as input to the commercial solver GUROBI [14]. The return values from these functions are considered to be the optimal solutions to the problem.

Algorithm 3: One iteration algorithm

function `OptimisePolytope`($\boldsymbol{p}_X, p_Y, \boldsymbol{P}_{W|X}, \boldsymbol{A}$)**:**
  `/* Prepare constraints for the polytope optimisation step */`
  cons = []
  for $i = 1 \dots |\mathcal{Y}|$ **do**
    `/* Ensure` $\mathtt{p}_{X|y_i}$ `is a valid pmf and inside` $\mathbb{S}$ `(`$\mathtt{Ap}_{X|y_i} = \mathtt{Ap}_X$`) */`
    cons.append(`BuildConstraints` ($\mathbf{p}_{X|y_i}, \mathbf{A}, \mathbf{p}_X$))
  end
  `/* Add consistency check for conditional distributions` $\mathtt{p}_{X|y}$ `*/`
  cons.append($\sum_{y\in\mathcal{Y}} p_Y(y)\mathbf{p}_{X|y} = \mathbf{p}_X$)
  objective = $\sum_{y\in\mathcal{Y}} p_Y(y) H(\mathbf{P}_{W|X}\mathbf{p}_{X|y})$
  $\mathbf{P}_{X|Y}$ = `DccpOptimise` (minimize(objective), cons, variables=$\{\, \mathbf{p}_{X|y_i} \mid i = \overline{1, |\mathcal{Y}|} \,\}$)
  return $\boldsymbol{P}_{X|Y}$
function `OptimiseDistribution`($\boldsymbol{p}_X, \boldsymbol{P}_{X|Y}$)**:**
  `/* Prepare constraints list for the distribution optimisation step */`
  cons = [$\sum_{y\in\mathcal{Y}} p_y = 1, p_Y \geq 0$]
  objective = $\sum_{y\in\mathcal{Y}} p_Y(y) H(\mathbf{P}_{W|X}\mathbf{p}_{X|y})$ `/* where` $H(\mathtt{P}_{W|X}\mathtt{p}_{X|y})$ `are precomputed */`
  p_Y = `LinearProgramOptimise` (minimize(objective), cons, variables=p_Y)
  return p_Y
function `DccpIteration`($\boldsymbol{P}_{W|X}, \boldsymbol{p}_X, |\mathcal{Y}|$)**:**
  $\mathbf{P}$ = `BuildBinaryMatrix` ($\mathbf{p}_X$)
  $\mathbf{U}, \mathbf{\Sigma}, [\mathbf{v}_1, \dots, \mathbf{v}_{|\mathcal{X}|}]^T$ = `SVD` ($\mathbf{P}$)
  $\mathbf{A} = [\mathbf{v}_1, \dots, \mathbf{v}_{\mathrm{rank}(\mathbf{\Sigma})}]^T$
  rand_p_Y = `RandomDistribution` ($|\mathcal{Y}|$)
  $\mathbf{P}_{X|Y}$ = `OptimisePolytope` ($\mathbf{p}_X$, rand_p_Y, A)
  p_Y = `OptimiseDistribution` ($\mathbf{p}_X, \mathbf{P}_{X|Y}$)
  $\mathbf{p}_{Y|X}$ = `BayesRule` ($\mathbf{P}_{X|Y}, \mathbf{p}_X, p_Y$)
  synergy = `MI` ($\mathbf{P}_{W|X}, \mathbf{p}_{Y|X}, \mathbf{p}_X$)
  return synergy, $\boldsymbol{p}_{Y|X}$

Given the nature of the CCP heursitic that DCCP is based on, the one iteration algorithm above can potentially be stuck in a local optimum which is far from the global optimum solution. Therefore, this motivates an extension that considers multiple runs of the single iteration algorithm, as the optimisation can explore the space more thoroughly and potentially find better solutions to the optimisation problem.

Extension and final algorithm

We introduce below the optimised algorithm for computing the optimal disclosure mapping, which we call `dccp-syndisc`.

Algorithm 4: Multiple iteration algorithm - `dccp-syndisc`

function `FindOptimalDisclosureMapping`($\boldsymbol{P}_{W|X}, \boldsymbol{p}_X, |\mathcal{Y}|$, iterations)**:**
  `/* Initialise optimal solution */`
  best_synergy, $\mathbf{p}^*_{Y|X} = 0$, None
  `/* Run multiple iterations of DCCP optimisation */`
  for $i = 1 \dots$ iterations **do**
    curr_synergy, $\mathbf{p}_{Y|X}$ = `DccpIteration` ($\mathbf{P}_{W|X}, \mathbf{p}_X, |\mathcal{Y}|$)
    `/* If solution is better than the optimal found so far and is precise enough, update optimal solution */`
    if curr_synergy $>$ best_synergy *and* `ConstraintChecking` ($\boldsymbol{p}_{Y|X}, \boldsymbol{p}_X, \boldsymbol{P}_{W|X}, \epsilon$) **then**
      best_synergy, $\mathbf{p}^*_{Y|X}$ = curr_synergy, $\mathbf{p}_{Y|X}$
    end
  end
  return best_synergy, $\boldsymbol{p}^*_{Y|X}$

Note that this can offer better estimations than the alternative algorithm, which again considers multiple iterations of `DccpOptimisation`, but continues with the optimisation of the initial solution obtained by a single run of the one iteration algorithm. This is because DCCP is very effective at finding local optimum points, so it can be confidently concluded that any solution obtained from one iteration of DCCP cannot be further optimised by considering the same region of the feasible set. However, the estimations of the optimal disclosure can potentially be much better when starting from new distributions p_Y and exploring other regions in the feasible set.

3.2.4 Choosing the starting parameters

Note that, compared to the initial version of `syndisc` in Algorithm 2, Algorithm 4, presented in the previous section, requires two more parameters ($|\mathcal{Y}|$ and `iterations`) that can heavily influence the efficiency of the estimation.

Choosing alphabet $\mathcal{Y}$

Considering theorem 2.2.1, we recall that the optimal disclosure channel is a point inside the convex polytope $\mathbb{S}$. This theorem shows that the optimal disclosure channel can be written as a linear combination of the extreme points in the polytope and it can be determined by solving the linear program in Eq. (2.3). Therefore, the alphabet $\mathcal{Y}$ describes how fine-grained the distribution p_Y needs to be in the optimal case and how many extreme points need to be considered for the optimal mapping $\mathbf{p}_{X|Y}$ (which is then used to compute $\mathbf{p}_{Y|X}$).

One of the advantages of the original `syndisc` algorithm is that it does not require setting $\mathcal{Y}$ beforehand. To achieve this, the complete set of extreme points is computed beforehand. This is useful, as the optimisation can consider all these points and decide which ones to use for the global optimum, but finding the extreme points is a very expensive operation from a computational point of view.

To avoid the expensive computation of the extreme points, Algorithm 4 requires a specific size of $\mathcal{Y}$ to be set. Therefore, fixing a $\mathcal{Y}$ that is too small might restrict the capacity of the disclosure channel to a point where it becomes very difficult to accurately estimate the optimal disclosure capacity. Conversely, opting for a very large $\mathcal{Y}$ can lead to computational challenges due to the problem's linear increase with $\mathcal{Y}$ size and memory constraints during optimization.

Through our experiments, we have observed that the best trade-off between computational efficiency and accuracy of estimation is achieved when $|\mathcal{Y}|$ is set to a moderately sized range (e.g. 10-20 elements), and estimation gains are marginal when increasing $\mathcal{Y}$ beyond that range. This observation is supported by comparing it with the optimal solutions achievable using the original `syndisc`, which are often within this range.

Choosing the number of iterations

As presented in the previous section, the Algorithm 4 is based on the CCP heuristic, which finds local optimal points that form lower bounds for the optimal disclosure. Due to the nature of the heuristic, doing multiple runs of `DccpIteration` allows the program to explore the space more thoroughly and can potentially lead to finding higher lower bounds. Naturally, increasing the number of runs will lead to better estimations, but this also increases the run time linearly. Throughout our experiments, we have observed that a fairly small number of iterations is enough to achieve good estimates, and we have decided to set the number of iterations for further experiments to 20.

3.3 Evaluation

Algorithm 4 presents a potential improvement to the original `syndisc` algorithm. It aims to provide accurate estimations of the optimal disclosure capacity given probability distributions $\mathbf{p}_X$ and $\mathbf{P}_{W|X}$, while offering improved scalability compared to the previous version. However, due to its heuristic nature, it is essential to thoroughly evaluate its practical utility. To achieve this, we utilize a combination of synthetic test data representing systems of various sizes and real-world data sourced from neuroscience studies.

3.3.1 Evaluation Metrics

When assessing the synergy computation's performance, we will focus on two key metrics for the evaluation:

- **Disclosure efficiency**: This metric represents the proportion of synergy identified by the optimized algorithm in comparison to the optimal synergy.
- **Scalability**: This metric measures the practical performance of the algorithm. Given the computational complexity of `syndisc`, the original package struggles to provide insights for systems with more than 5 sources. Thus, our objective is to surpass this limitation and explore larger systems.

A key property of a synergistic channel is keeping the set of sources private when sharing information with an outside source. Therefore, it is essential for Algorithm 4 to prioritise obtaining a channel $p_{Y|X}$ that satisfies with great precision the constraints. Therefore, all the results shown below are obtained for channels which satisfy the constraints within 10^{-10} precision. While these optimised channels give valid channels Y, and offer a useful lower bound for synergy, they may not necessarily provide the optimal disclosure.

3.3.2 Synthetic data experiments

Example 3.3.1 (Correlated AND gate)**.** *As a first example, an experiment from [8] is reproduced, where the authors consider a correlated AND gate with two sources. Take* $\boldsymbol{X}$ *to be a system with two binary sources, and the target* $W = X_1 \wedge X_2$*. Below, we compare ground truth results with the estimates of* `dccp-syndisc`*:*

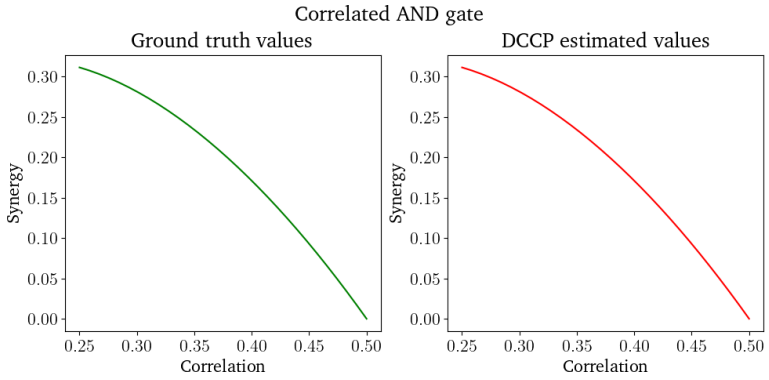


Figure 3.1: Correlated AND gate

The estimates obtained from the optimised software are precise, and the curve obtained from `dccp-syndisc` *is the same as the one obtained from the initial algorithm.*

The results in the experiment above are promising, as it shows that for simple scenarios, `dccp-syndisc` achieves optimal performance. In the following section, a more thorough evaluation is presented.

Small-system testing

The original `syndisc` algorithm only supports relatively small systems, so comparisons with ground truth values can only be made for such systems. In practice, it struggles to compute the optimal disclosure mapping for systems with more than 5 binary sources and 1 target. Hence, initial tests involve selecting random distributions of a small number of sources, $\mathbf{p}_X$, and considering targets W, given by $\mathbf{P}_{W|X}$.

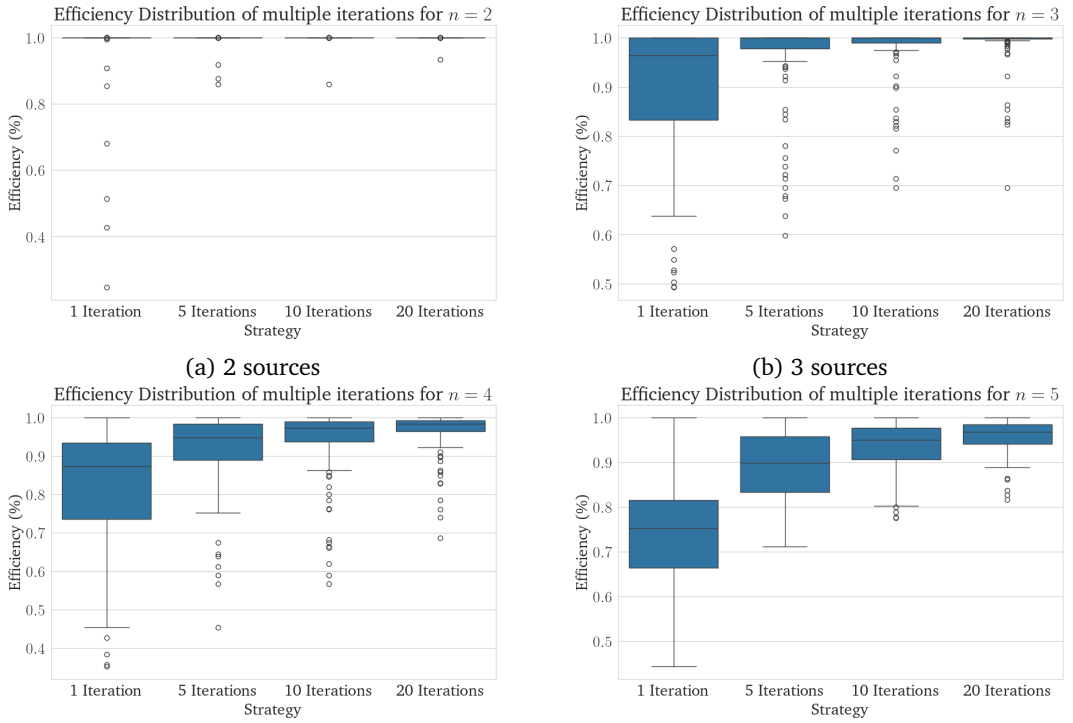
Systems of various size are considered for the evaluation process. The evaluation results from 100 runs of the described algorithms are presented below, and the outcomes of `dccp-syndisc` are compared to the ground truth values derived from `syndisc`.

Table 3.1: Evaluation results on small systems

n	Efficiency (Mean $\pm$ Std. Dev.)	Worst case efficiency	`syndisc` time	`dccp-syndisc` time
2	99.93% $\pm$ 0.65%	93.48%	0.0038s	2.4506s
3	98.59% $\pm$ 4.61%	69.54%	0.015s	3.7448s
4	96.04% $\pm$ 5.86%	68.65%	0.035s	4.2869s
5	95.57% $\pm$ 3.92%	81.66%	1.092s	5.3975s

The results above show that `dccp-syndisc` achieves on average very high efficiency, with all mean efficiency values above 95%. Also, the execution time of the original `syndisc`, while very short for small systems, increases quickly, which is particularly clear in the case of $n = 5$, and is unable to finish the computation for $n \geq 6$. In contrast, the execution time for `dccp-syndisc` increases more gradually as n increases, suggesting its potential for more efficient scalability in larger systems.

Despite the generally strong performance, it is important to discuss the variation in minimum efficiencies, which can drop as low as 68%. This highlights the heuristic nature of the DCCP approach, indicating that while most estimates are above 85% accuracy, there can be significant deviations in some cases.



(a) 2 sources

(b) 3 sources

(c) 4 sources

(d) 5 sources

Figure 3.2: Efficiency distribution for small systems. Boxes and whiskers contain most of the test data, while outliers are represented as dots. As number of iterations increases, the performance increases and becomes more stable.

In the plots above, it can be observed how increasing the number of iterations affects the performance of the algorithm. While all four variants of the algorithm achieve good average results, increasing the number of iterations significantly affects the variance of these estimations. However, the execution time increases linearly with the number of iterations. Therefore, we have considered 20 iterations to be a good trade-off between accuracy and execution speed for general tasks, but this parameter can be changed depending on the restrictions of the problem.

The efficiency of the estimations is also influenced by the chosen size of the private disclosure channel's alphabet. The table below shows how performance is affected by this parameter.

Table 3.2: Efficiency for different sizes of $\mathcal{Y}$ in systems with 4 sources

Alphabet size	Efficiency (Mean $\pm$ Std. Dev.)	Mean runtime
$\lvert\mathcal{Y}\rvert = 5$	79.76% $\pm$ 20.87%	2.4865s
$\lvert\mathcal{Y}\rvert = 10$	97.02% $\pm$ 4.07%	4.1432s
$\lvert\mathcal{Y}\rvert = 15$	99.52% $\pm$ 1.5%	5.9490s
$\lvert\mathcal{Y}\rvert = 20$	99.71% $\pm$ 1.87%	13.6866s
$\lvert\mathcal{Y}\rvert = 25$	99.89% $\pm$ 0.085%	16.6089s
$\lvert\mathcal{Y}\rvert = 30$	99.98% $\pm$ 0.012%	19.9190s

Similarly to the number of iterations, this parameter can be varied to provide more precision. However, note that after a certain threshold, the performance gains obtained for this parameter are marginal, but the execution time increases quickly. We have decided to use moderately sized alphabets (between 10 and 20 elements), as these offer a good overall performance, while still finding the optimal value relatively quickly.

Stress testing

The next step in the evaluation process is testing the scalability of Algorithm 4, by considering an increasingly larger sets of sources. Experiments revealed that the algorithm efficiently computes synergy in systems containing up to 13 binary sources. However, extending beyond this threshold is constrained by memory limitations due to the need for loading the entire system into memory before applying convex optimization techniques.

Ideally, as the size of the system increases, the disclosure efficiency of the algorithm is continuously assessed and compared to ground truth values. However, given the limits of `syndisc`, ground truth values cannot be computed for systems this large. To circumvent this limitation, the following testing strategy is proposed:

1. Generate a random distribution of sources $\mathbf{p}_X$ and target $\mathbf{P}_{W|X}$ for a small system ($n \leq 5$).
2. Extend the distribution to a larger system ($n \leq 11$), by adding duplicates of some of the sources in the system.
3. As synergy in this system will not change when adding the duplicates, synergy in the small system can be computed using `syndisc`, and then compared with the output of the optimised algorithm when given the large system.

Table 3.3: Evaluation results on large systems

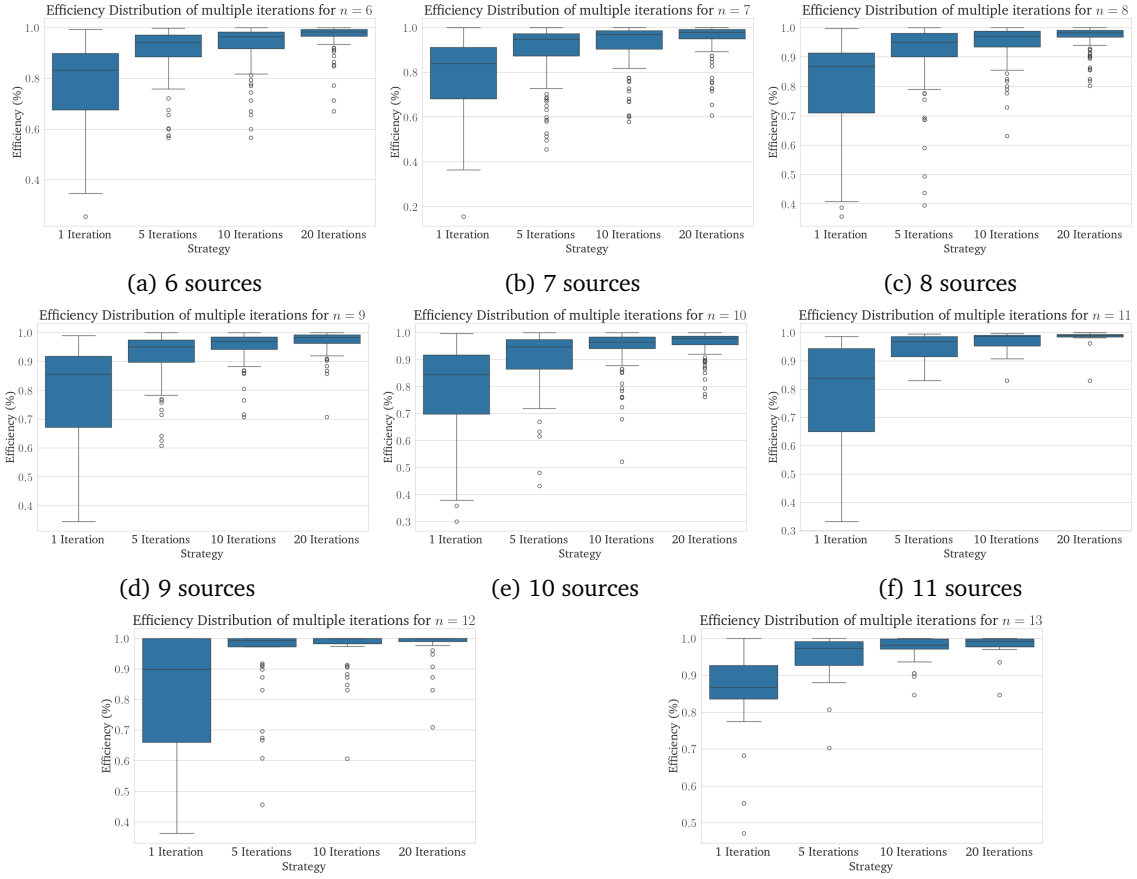
n	Efficiency (Mean $\pm$ Std. Dev.)	Worst case efficiency	`dccp-syndisc` time	`syndisc` time [1]
6	96.57% $\pm$ 5.38%	67.04%	8.6553s	$>$ 6h
7	94.48% $\pm$ 8.21%	60.57%	13.6003s	$>$ 6h
8	96.71% $\pm$ 4.15%	80.16%	24.8697s	$>$ 6h
9	97.01% $\pm$ 3.97%	70.70%	55.0243s	$>$ 6h
10	96.04% $\pm$ 4.79%	76.17%	137.8581s	$>$ 6h
11	97.84% $\pm$ 4.05%	83.00%	436.6510s	$>$ 6h
12	97.97% $\pm$ 5.35%	70.90%	951.7550s	$>$ 6h
13	98.14% $\pm$ 3.21%	84.57%	2158.4898s	$>$ 6h

From the table above, it can be observed that the performance of `dccp-syndisc` remains relatively stable in larger systems and the size of the system does not influence the performance of the optimisation.

In the figures below, the performance of the multiple iteration algorithm is assessed once again. The results are consistent with the results obtained in the case of small systems. In particular, note

[1] In all cases, when attempting to run `syndisc` with large systems as input, the computation ran for this amount of time without finishing.

that the algorithm using 20 iterations of the optimisation has a much smaller variance compared to the other strategies. This shows that the estimations from `dccp-syndisc` are robust, and the algorithm becomes suitable for practical applications.



(a) 6 sources (b) 7 sources (c) 8 sources

(d) 9 sources (e) 10 sources (f) 11 sources

(g) 12 sources (h) 13 sources

Figure 3.3: Efficiency distribution for large systems

3.3.3 Summary of systems supported

The complexity in the systems considered can either come from the number of sources in the system, or from the number of states each component has. Summarising the results of this evaluations, the following set of pairs of system size n and source alphabet $\mathcal{X}$ that are supported by each algorithm:

Table 3.4: Supported systems by `syndisc`

n \ $\lvert\mathcal{X}\rvert$	2	3	4	5	6	7
$n = 2$	✓	✓	✓	✗	✗	✗
$n = 3$	✓	✓	✗	✗	✗	✗
$n = 4$	✓	✗	✗	✗	✗	✗
$n = 5$	✓	✗	✗	✗	✗	✗
$n \geq 6$	✗	✗	✗	✗	✗	✗

Table 3.5: Supported systems by `dccp-syndisc`

n \ $\lvert\mathcal{X}\rvert$	2	3	4	5	6	7
$n = 2$	✓	✓	✓	✓	✓	✓
$n = 3$	✓	✓	✓	✓	✓	✓
$n = 4$	✓	✓	✓	✓	✓	✗
$n = 5$	✓	✓	✓	✓	✗	✗
$n = 6$	✓	✓	✓	✗	✗	✗
$n = 7$	✓	✓	✓	✗	✗	✗
$n = 8$	✓	✓	✗	✗	✗	✗
$n = 9$	✓	✗	✗	✗	✗	✗
$n = 10$	✓	✗	✗	✗	✗	✗
$n = 11$	✓	✗	✗	✗	✗	✗
$n = 12$	✓	✗	✗	✗	✗	✗
$n = 13$	✓	✗	✗	✗	✗	✗

3.4 Applications to real-world data

Synergistic relationships are present in a wide range of systems, and these can be used in discovering and quantifying complex interactions between components. Hence, when developing methods to measure synergy, we should always consider the real-world applications these have. While the previous version of `syndisc` was useful for analyzing small systems, it is not appropriate for real-world datasets that typically involve larger distributions. In contrast, the optimised `dccp-syndisc` extension discussed earlier offers significantly better scalability, making it highly promising for practical applications.

Nervous systems are a prime example of systems that can potentially exhibit synergistic relationships, and in this section we present a few potential applications of `dccp-syndisc` in this context. We describe the methodology that can be used when analysing a real-world dataset, and we also briefly discuss the results obtained when working with some of these datasets.

3.4.1 Experiment setup

In the experiments using real-world data, the aim is to evaluate the significance of discovered synergy among n sources $X_1, X_2, \ldots, X_n$ and one target W, based on data samples obtained from the joint distribution.

The synergy measure is based on discrete distributions, so the first step is pre-processing the data, so that the set of values is discrete. The experiments use multiple modalities for the distributions, by converting the data to discrete form, using bins corresponding to different percentiles. For each modality, a one-sided hypothesis test is used to determine if the system is synergistic. Based on multiple permutations of the data, it is decided whether the null hypothesis can be rejected. Note that the spatial correlation is important in the case of data related to the structure of the brain. Therefore, the permutations need to preserve this property, and to achieve this, spin tests are used, as described in [21]. In these, the brain data is projected on a sphere, which is then rotated to preserve spatial relations between the measurements.

Another important observation is that the experiment can be heavily influenced by the estimation process for the joint distribution, and the number of bins used for the quantization of the data is a critical parameter in this context. The domain of the data is continuous, so a lot of information can potentially be lost in the quantization process. This is especially relevant given that the number of bins supported by the algorithms for measuring synergy is still fairly small. It is also important to consider that given the quick growth in the number of states relative to number of bins, the number of data samples might not be enough to accurately estimate the joint distribution (especially when basic methods such as plug-in estimators are used).

3.4.2 Dataset description

Brain data: Neuromaps experiments

The brain data used in this study is sourced from the neuromaps project [22]. This dataset includes multiple high-resolution brain maps that capture brain activity across specific regions. The neuromaps dataset provides detailed brain maps that illustrate the spatial distribution of neural activity and connectivity patterns.

The neuromaps project offers a good environment for `dccp-syndisc` to be evaluated, because the diverse brain maps can be used to create systems with various number of sources. In our experiments, we aim to observe the synergistic relationships that emerge when considering groups of annotations from the same type as sources (such as Microstructure, Metabolism, or Electrophysiology) and the functional gradient as the target. The functional gradient is an ideal target because it captures gradual changes in functional connectivity and activity patterns across different brain regions, reflecting the brain's underlying organizational principles. By examining multiple sources from the same type of annotation together, we can evaluate their effectiveness in predicting changes in the gradient.

The following sets of annotations are used in the experiments:

1. **Microstructure**: There are two microstructure maps, corresponding to T1w/T2w ratio and cortical thickness, which are used as sources. These are derived from magnetic resonance

imaging (MRI) data, provided by the Human Connectome Project (HCP). The annotations show measurements of relative cortical myelin content and cortical thickness [23].

2. **Metabolism**: This setup contains four sources, corresponding to Cerebral Blood Flow (CBF), Cerebral Blood Volume (CBV), Oxygen metabolism (CMRO2), and Glucose metabolism (CMRGlu) [24].

3. **Electrophysiology**: This experiment uses MEG (Magnetoencephalography) power distributions from the Human Connectome Project, and contains five sources, corresponding to Delta, Theta, Alpha, Beta and Gamma powers. Neural communication in the brain is typically estimated through analysis of electromagnetic time-series, so this set of sources can potentially offer insights in the functional connectivity patterns in the brain [25].

Gene data: Enigma Toolbox

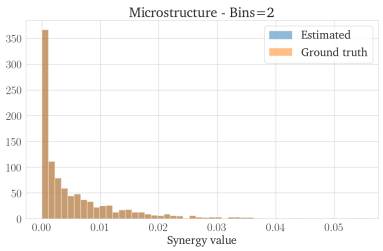
Another potential application for the synergy measure is in genetic data, where synergy can be used to observe interactions between genes and associated behaviors. For these experiments, we utilized data from the ENIGMA Toolbox [26], which offers a wide range of information useful for studying the genetic basis of various neurological disorders.

The ENIGMA toolbox provides curated datasets of genes that have been statistically associated with specific disorders. The toolbox also includes detailed data for these conditions, such as measurements of cortical thickness between case and control subjects. Therefore, this toolbox provides a good setting to apply `dccp-syndisc`. In the context of synergy, we wish to examine whether by considering sets of such genes, we can observe synergistic systems when the target represents the cortical thickness data associated with the disorder.

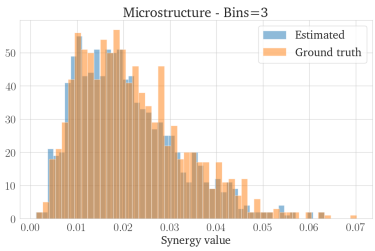
3.4.3 Experiment results: Neuromaps

Consistency check with `syndisc`

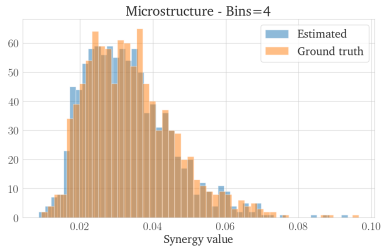
Lastly, a consistency check is performed, to assess once again the performance of `dccp-syndisc`. With 4 bins or less, the experiment setup can be used to compare the estimations with the ground truth. Therefore, the same experiment is run using both versions of the software and the results are then compared. The histograms for two bins match exactly, whereas those for three and four bins show highly similar values. This analysis confirms the strong performance of `dccp-syndisc` and the precise estimates of synergy it can achieve.



(a) Bins = 2



(b) Bins = 3



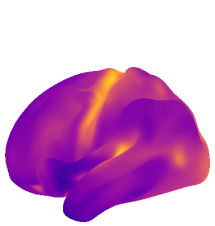
(c) Bins = 4

Figure 3.4: Histograms comparing the results from `syndisc` (orange) and `dccp-syndisc` (blue). The histograms are very similar in all cases. For 3 and 4 bins, the original algorithm reports slightly higher synergy values.

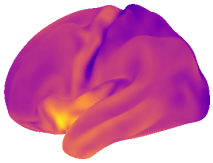
Microstructure

Given that the system comprises only two sources, `dccp-syndisc` is flexible enough to support distributions with an increased number of bins. Therefore, this scenario allows us to explore how the level of quantization impacts the results of the experiment.

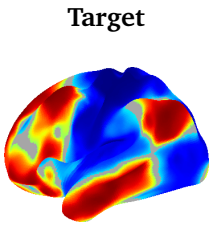
We expect that using only very few bins, such as converting the data into a binary format, will result in substantial information loss and might even obscure any synergistic results. Indeed, observations confirm that with two or three bins, the experimental results are not significant, and we fail to reject the null hypothesis. However, when more than three bins are used, the evidence of synergy in the system becomes significant (at 5% level) and the null hypothesis is rejected.



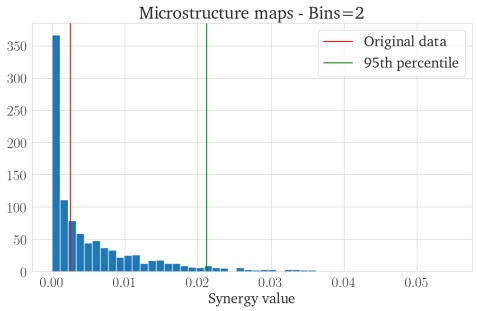
(a) T1w/T2w ratio



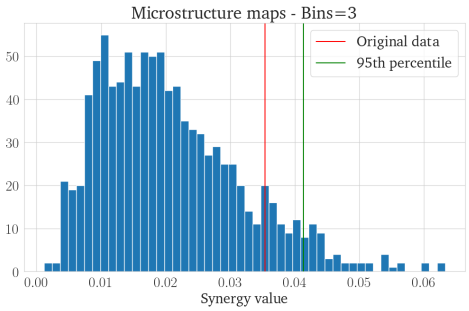
(b) Cortical Thickness



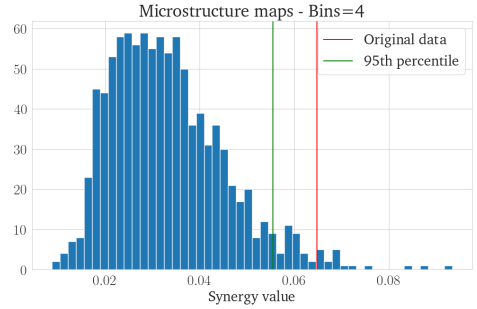
(c) Functional Gradient



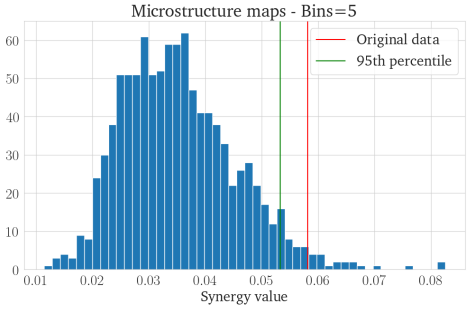
(d) Bins = 2



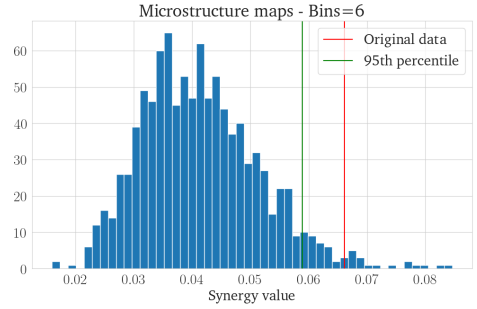
(e) Bins = 3



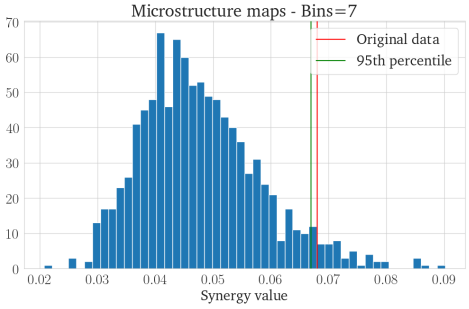
(f) Bins = 4



(g) Bins = 5



(h) Bins = 6

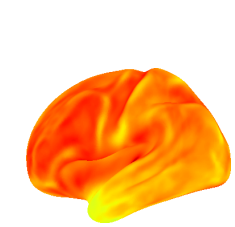


(i) Bins = 7

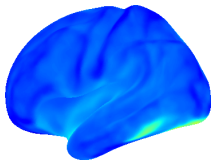
Figure 3.5: Experiment results for Microstructure annotation maps. In the fist two histograms, we fail to reject the null hypothesis, while in the last four, the null hypothesis is rejected at 5% level.

Metabolism

Similar to the previous scenario, the number of sources allow us to experiment with multiple levels of quantization, so the experiment is repeated with 4, 5, and 6 bins. However, in all cases, the experiment results are not significant at 5% level, and we fail to reject the null hypothesis.

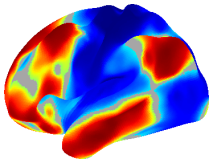


(a) Cerebral Blood Flow

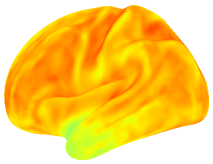


(b) Cerebral Blood Volume

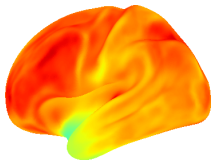
Sources **Target**



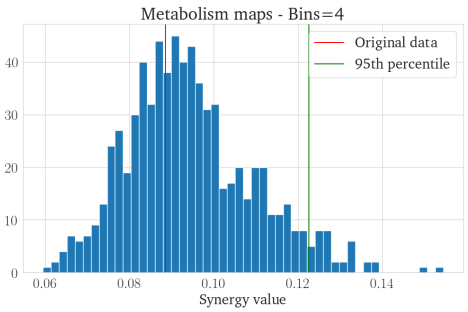
(c) Functional Gradient



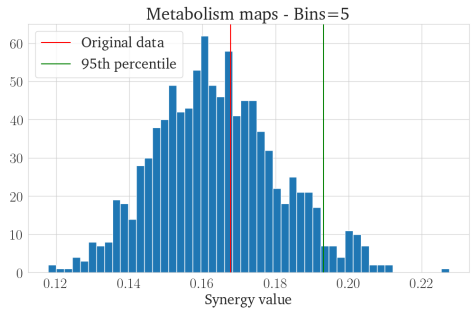
(d) Oxygen metabolism



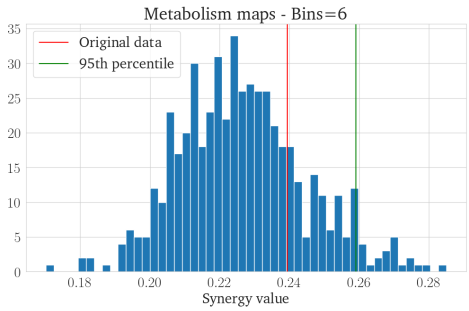
(e) Glucose metabolism



(f) Bins = 4



(g) Bins = 5

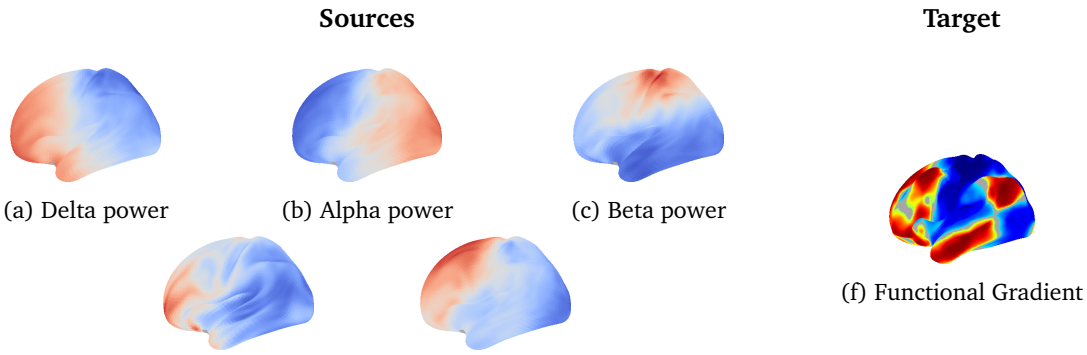


(h) Bins = 6

Figure 3.6: Experiment results for Metabolism annotation maps. In all cases, the results are not significant, and we fail to reject the null hypothesis.

Electrophysiology

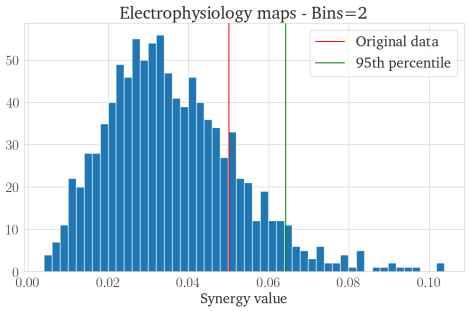
In this scenario, the number of sources allows `dccp-syndisc` to consider up to 5 bins for the quantization process. Again, as in the experiments for metabolism maps, the data does not have significant traces of synergy.



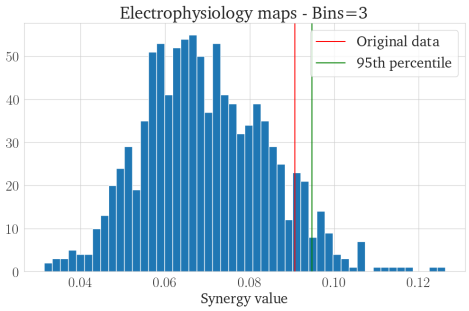
(a) Delta power (b) Alpha power (c) Beta power

(f) Functional Gradient

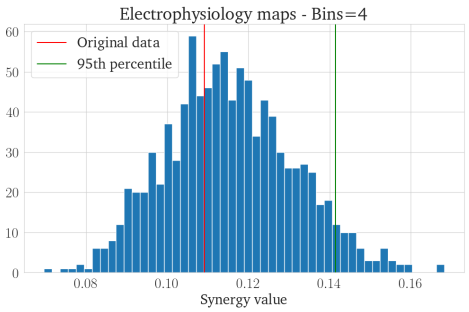
(d) Gamma power (e) Theta power



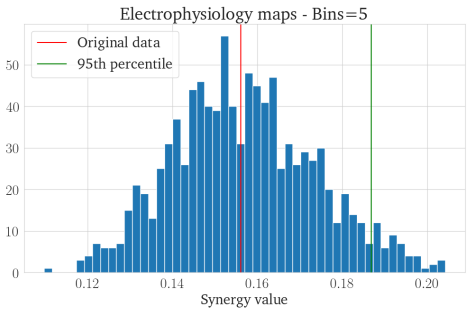
(g) Bins = 2



(h) Bins = 3



(i) Bins = 4



(j) Bins = 5

Figure 3.7: Experiment results with Electrophysiology annotation maps. In all cases, the results are not significant, and we fail to reject the null hypothesis.

3.4.4 Experiment results: Enigma Toolbox

A potentially interesting application of `dccp-syndisc` is determining how common synergistic subsets are in real-world datasets. The large number of genes related to epilepsy makes the data available through the Enigma Toolbox appropriate for such an experiment.

The Enigma Toolbox reports 8 risk genes associated with the epilepsy disorder, and considering these as sources creates a large system, and truly shows the potential of `dccp-syndisc`. The Cohen's d-values available in this toolbox are used as target. These represent changes in cortical thickness for brain regions in studies comparing subjects with and without the disorder. Given the large number of null values in this distribution (corresponding to no change between case and control datasets), this dataset was converted to binary format.

The experiment setup involves evaluating multiple gene subsets for synergy to uncover potential links between genes and neurological disorders, specifically epilepsy. For a comprehensive experiment, three sets of genetic data for the subsets are used:

1. **Related Genes:** We consider each subset of related genes as a potential source set. Following a spin test for each subset, we determine if there are significant indications of synergy.
2. **Unrelated Genes:** From the large number of genes considered unrelated, we select random subsets of varying sizes and reproduce the previous experiment.
3. **Control dataset (Shuffled, random genetic data):** To establish a control standard for the synergy tests, we consider a strategy that breaks all dependencies within the data. This involves using data from genes not associated with epilepsy and randomly shuffling all selected gene data. We then repeat the previous spin tests. This dataset serves as a baseline to estimate the lower bound of the expected number of synergistic subsets.

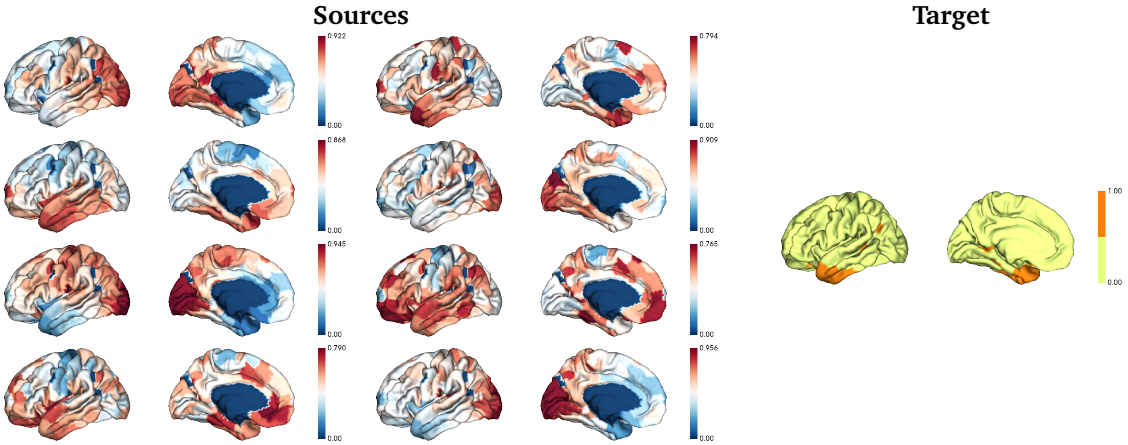


Figure 3.8: System used in experiments using genetic data. The sources represent the 8 risk genes linked to epilepsy (as reported by Enigma Toolbox), while the target represents the binarised Cohen's d-values between case and control datasets.

In the plot below, we show the results for the experiment described above. The plot represents the fraction of subsets that resulted in significant synergy at the 5% level. The x-axis indicates the size of each subset, with corresponding values for fractions of related genes subsets, unrelated genes subsets, and control subsets.

Firstly, one key observation from these results is that a significant number of both related and unrelated genes subsets exhibit synergy. This suggests that synergy is a fairly common characteristic in datasets sourced from neuroscience studies, indicating promising potential for its application in various real-world scenarios.

We now continue with a thorough analysis of the results. As expected, given that the significance level we considered is 5%, the control data shows a constant 5% fraction of synergistic subsets. This sets a baseline for interpreting results from both related and unrelated genetic data.

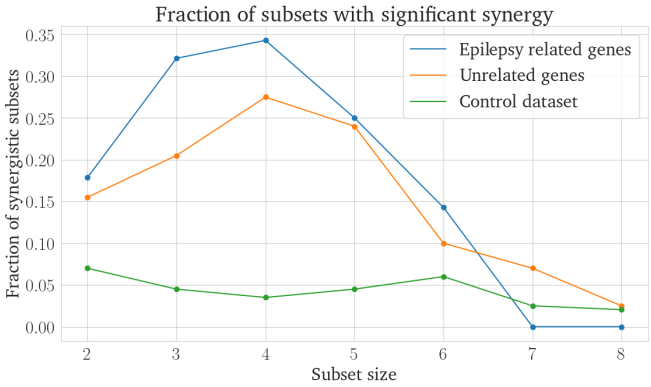


Figure 3.9: Fraction of subsets synergistic with epilepsy data. Related genes exhibit the highest fraction of synergistic subsets, while the unrelated genes show lower values. The control dataset is around 5% for all subset sizes.

A substantial fraction of related genes subsets show significant traces of synergy. This is a somewhat expected result, as these genes have been selected as a result of statistical analysis on their presence in different regions of the brain. However, it is worth exploring further which particular sets of related genes are synergistic and to investigate the common characteristics of the genes that frequently appear in these synergistic groups. Also note that, as the size of the gene subsets increases, the presence of synergy decreases, and when considering subsets of 7 genes, or the whole set of related genes, there are no significant traces of synergy.

The synergistic subsets of unrelated genes also offer an interesting hypothesis. Although these subsets show lower synergy compared to those of related genes, a notable number of gene subsets, traditionally not associated with epilepsy, showed significant synergy. This finding could suggest previously unrecognized interactions between genes and disorders like epilepsy, presenting a new way for data analysis in this field.

3.5 Discussion and interim conclusion

Throughout this chapter, we have explored the scalability and efficiency of `dccp-syndisc`, a novel extension of the synergy calculation software `syndisc`. This tool uses disciplined convex-concave programming (DCCP) to optimize the synergy calculation process for systems containing discrete random variables. We have highlighted both the mathematical foundations of the approach and its practical implementation, presenting extensive evaluation results on both synthetic and real-world data.

We summarise here the key findings of this section:

1. **Scalability and Performance**: Our evaluations demonstrate that `dccp-syndisc` offers a significant improvement in scalability compared to its predecessor. It manages to compute synergy efficiently for systems with up to 13 binary sources, a considerable advancement over the five-source limit of the original `syndisc` algorithm. This capability makes it a more viable tool for analyzing complex real-world systems where large datasets are common.

2. **Efficiency of Estimation**: The method consistently achieves high efficiency in synergy estimation across various system sizes and conditions. While there are occasional deviations due to the heuristic nature of the DCCP approach, the average performance remains robust. This reliability supports the utility of `dccp-syndisc` in both academic research and potentially clinical applications where accurate measurement of data interaction is crucial.

3. **Real-World Application**: The application of `dccp-syndisc` to neuroscience and genetic data illustrates its potential in discovering complex synergistic relationships in high-dimensional data. For instance, the analysis of brain mapping data from the neuromaps project and gene association data from the Enigma Toolbox has shown that `dccp-syndisc` can identify significant synergy patterns that may be vital for understanding underlying biological mechanisms.

Chapter 4

Synergy in continuous distributions

In this chapter, we extend the notion of synergy, initially defined only on discrete probability distributions in [12], to continuous distributions. By taking advantage of results from copula theory, we present a method for computing the synergy in this case.

4.1 Copula theory

Copulas are multivariate cumulative distribution functions for which the marginal probability distribution of each variable is uniform on the interval [0, 1]. These functions simplify the analysis of joint probability distributions by abstracting the marginal distributions of the random variables and capturing the dependencies between them. In this section, we cover the fundamentals of copula theory as outlined in [27], along with advanced topics such as pair-copulas and vine tree decompositions [28], and checkerboard copulas [29, 30].

4.1.1 Fundamental properties of copulas

In this section, the concept of copula is introduced, along with its fundamental properties, as presented in [27].

Definition 4.1.1. *A copula is a multivariate distribution function $C : [0,1]^n \to [0,1]$ with uniform marginals on the interval $[0,1]$. Specifically, C satisfies the following properties:*

- *For every $u \in [0,1]^n$, $C(u_1, \ldots, u_n)$ is d-non-decreasing (the volume in each hyperrectangle is non-negative)*
- *$C(u_1, \ldots, u_{i-1}, 0, u_{i+1}, \ldots, u_d) = 0$, the copula is zero if any one of the arguments is zero*
- *$C(1, \ldots, 1, u, 1, \ldots, 1) = u$, the copula is equal to u if one argument is u and all others 1.*

Theorem 4.1.1 (Sklar's Theorem)**.** *Let H be a joint distribution function with marginals $F_1, \ldots, F_n$. Then there exists a copula C such that for all $x_1, \ldots, x_n \in \mathbb{R}$,*

$$H(x_1, \ldots, x_n) = C(F_1(x_1), \ldots, F_n(x_n)).$$

If $F_1, \ldots, F_n$ are continuous, then the copula C is unique. Conversely, if C is a copula and $F_1, \ldots, F_n$ are distribution functions, then H defined above is a joint distribution function with marginal probability distributions $F_1, \ldots, F_n$.

The copula function above can be seen as a multivariate probability distribution. Based on this, the corresponding copula density can be derived:

$$c(u_1, \ldots, u_n) = \frac{\partial^n C(u_1, \ldots, u_n)}{\partial u_1 \ldots \partial u_n}.$$

The copula density provides a useful relationship between the pdfs of the marginals and the joint distribution, as by taking partial derivatives in all directions, Sklar's Theorem becomes:

$$h(x_1, \ldots, x_n) = c(F_1(x_1), \ldots, F_n(x_n)) \prod_{i=1}^{n} p_i(x_i),$$

where $p_1, \ldots, p_n, h$ are the probability density functions for $X_1, \ldots, X_n, (X_1, \ldots, X_n)$ respectively.

A few important properties of copulas are introduced below.

Theorem 4.1.2. *Let $X_1, X_2, \ldots X_n$ be continuous random variables, with copula C. Then $X_1, \ldots, X_n$ are independent if and only if C is the independence copula $\Pi_n(u_1, u_2 \ldots, u_n) = u_1 u_2 \ldots u_n$.*

In this case, $c(u_1, u_2 \ldots, u_n) = 1, \forall u_1 \ldots u_n \in [0, 1]$, where $c(u_1, u_2 \ldots, u_n)$ is the copula density.

Theorem 4.1.3 (Fréchet–Hoeffding copula bounds)**.** *For any copula C and any $u, v \in [0, 1]$, the following bounds hold:*

$$W(u, v) \leq C(u, v) \leq M(u, v),$$

where $W(u, v) = \max(u + v - 1, 0)$ is the Fréchet–Hoeffding lower bound and $M(u, v) = \min(u, v)$ is the Fréchet–Hoeffding upper bound. The copula W corresponds to the case of countermonotonicity, and the copula M corresponds to comonotonicity.

Remark 4.1.1. *Note that the copula density associated to the comonotonicity copula is similar in nature to the Dirac-delta function, and has the following form:*

$$c(u, v) = \begin{cases} \infty & \text{if } u = v, \\ 0 & \text{otherwise.} \end{cases}$$

Definition 4.1.2 (Total correlation)**.** *[2] The total correlation of the set composed of random variables $\{X_1, X_2, \ldots, X_n\}$ is defined as:*

$$\begin{aligned} C(X_1, X_2, \ldots, X_n) &= \mathrm{D}_{\mathrm{KL}}\left[p(X_1, \ldots, X_n) \| p(X_1)p(X_2)\cdots p(X_n)\right] \\ &= \sum_{i=1}^{n} H(X_i) - H(X_1, X_2, \ldots, X_n) \end{aligned}$$

where D_{KL} represents the Kullback-Leibler divergence, $H(X_i)$ denotes the entropy of the random variable X_i, and $H(X_1, X_2, \ldots, X_n)$ is the joint entropy of all the random variables. This measure quantifies the total amount of mutual information shared among the variables, reflecting their joint statistical dependence.

The following theorem provides an alternative way of expressing total correlation of random variables based on the copula of the joint distribution.

Theorem 4.1.4 (Copula Entropy and Total correlation)**.** *[31] Let $\boldsymbol{X} = \{X_1, X_2, \ldots X_n\}$ be random variables with marginal distribution functions $u = [F_1, \ldots, F_n]$ and copula density $c(u)$. The copula entropy of $X_1, X_2, \ldots X_n$ is defined as $H_c(\boldsymbol{X}) = -\int_u c(u) \log c(u)\, du$.*

The total correlation of the random variables is then equivalent to the negative of their copula entropy:

$$C(X_1, X_2, \ldots, X_n) = \int_{\mathcal{X}} p(x_1, \ldots, x_n) \log \frac{p(x_1, \ldots, x_n)}{\prod_{i=1}^{n} p(x_i)} d\boldsymbol{X} = -H_c(\boldsymbol{X}).$$

Two useful examples of families of copula distribution functions in the bivariate case are introduced below, as shown in [32]. These definitions can also be extended to multiple dimensions.

Definition 4.1.3 (Gaussian Copula)**.** *Given a bivariate random vector $(X_1, X_2) \sim \mathcal{N}_2(0, \Sigma)$ where Σ is the 2×2 correlation matrix of X, the Gaussian copula C_Σ^{Gauss} is defined as:*

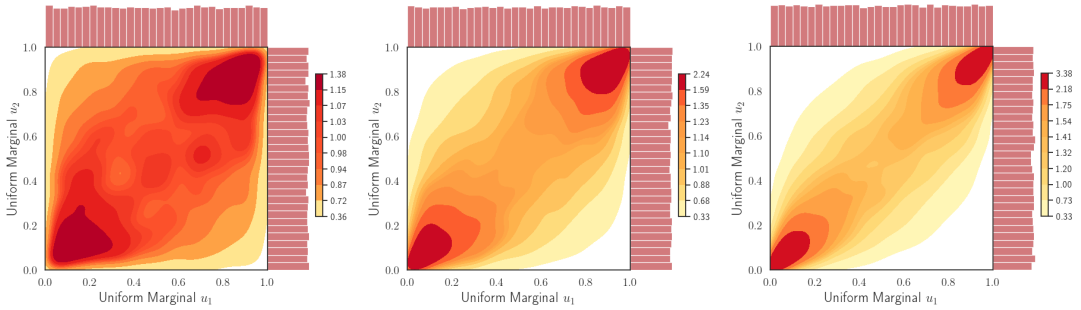
$$C_\Sigma^{Gauss}(u_1, u_2) \triangleq \Phi_2(\Phi^{-1}(u_1), \Phi^{-1}(u_2); \Sigma),$$

where $\Phi(\cdot)$ denotes the standard univariate normal CDF and $\Phi_2(\cdot, \cdot; \Sigma)$ denotes the bivariate normal CDF with correlation matrix Σ. The density of the copula is given by:

$$c(u_1, u_2) = \frac{1}{\sqrt{1 - \rho_{12}^2}} \exp\left(-\frac{\rho_{12}^2(x_1^2 + x_2^2) - 2\rho_{12}x_1x_2}{2(1 - \rho_{12}^2)}\right),$$

where ρ_{12} is the correlation between X_1 and X_2, $x_1 = \Phi^{-1}(u_1)$, and $x_2 = \Phi^{-1}(u_2)$.

This copula can be visualised in the figure below, which also highlights the uniform marginal property of copulas by projecting the samples on the axes.



(a) $\rho = 0.2$ (b) $\rho = 0.5$ (c) $\rho = 0.7$

Figure 4.1: Gaussian copula simulation with different correlations, using 50000 samples of $(X_1, X_2)-$ bivariate normal distribution. Projecting the marginals on the axes shows that this density has uniform marginals.

Definition 4.1.4 (Gumbel Copula)**.** *The bivariate Gumbel copula, parameterized by $\theta \geq 1$, is defined as:*

$$C_\theta^{Gum}(u_1, u_2) \triangleq \exp\left(-\left((-\log u_1)^\theta + (-\log u_2)^\theta\right)^{1/\theta}\right).$$

For $\theta = 1$, the copula represents independence, and as $\theta \to \infty$, it approaches the comonotonicity copula.

4.1.2 Checkerboard copulas

Checkerboard copulas are a special type of copula constructed by partitioning the unit cube into smaller, non-overlapping cubes. This partitioning allows for a flexible method to approximate any continuous copula distribution. The main motivation behind using checkerboard copulas is that besides their computational simplicity, these offer a useful connection between the case of discrete synergy and the continuous extension. This section introduces the formal definition, as well as a set of useful results concerning checkerboard copulas.

Consider $I_n = \{0, \frac{1}{n}, \frac{2}{n}, \dots, \frac{n-1}{n}, 1\}, \forall n \in \mathbb{N}$. Also, recall the notation $[m : n] = \{m, m+1, ..., n\}$.

Definition 4.1.5 (Discrete copula)**.** *[29] A discrete copula is a function $C_{n,m} : I_n \times I_m \to [0, 1]$ with the following properties:*

- $C_{n,m}(\frac{i}{n}, 0) = C_{n,m}(0, \frac{j}{m}) = 0, \forall i \in [0 : n], j \in [0 : m]$
- $C_{n,m}(\frac{i}{n}, 1) = \frac{i}{n}, C_{n,m}(1, \frac{j}{m}) = \frac{j}{m}, \forall i \in [0 : n], j \in [0 : m]$
- $C_{n,m}(\frac{i}{n}, \frac{j}{m}) + C_{n,m}(\frac{i-1}{n}, \frac{j-1}{m}) \geq C_{n,m}(\frac{i-1}{n}, \frac{j}{m}) + C_{n,m}(\frac{i}{n}, \frac{j-1}{m}), \forall i \in [1 : n], j \in [1 : m]$.

Remark 4.1.2. *[29] The discrete function $C_{n,m} : I_n \times I_m \to [0, 1]$ can be extended to a valid copula distribution on $[0, 1]^2$. The associated copula distribution function (called checkerboard copula) is piecewise constant on each rectangle of the partition of $[0, 1]^2$.*

Denote by $\mathcal{DC}_n$ the set of checkerboard copulas of size n.

Remark 4.1.3. *The definition above can be naturally extended to a d-dimensional definition of a discrete copula on the cube $[0, 1]^d$. The first two properties are formulated equivalently by taking all but one argument to be either all 0 or all 1, while the last property is extended to an inequality ensuring that the volume of the unit hyperrectangle is non-negative. In this section we focus on the two dimensional case for clarity, but all concepts can be extended to high-dimensional spaces.*

Definition 4.1.6. *[29] A $n \times n$ matrix $\mathbf{A} = (a_{ij})_{i,j=1}^n$ is called doubly stochastic if:*

- *All its elements are non-negative, i.e. $a_{ij} \geq 0$ for all i, j.*
- *Each row and each column sums to 1, i.e. $\sum_{j=1}^n a_{ij} = 1$ for all i and $\sum_{i=1}^n a_{ij} = 1$ for all j.*

Proposition 4.1.1. *[29] For a function $C_{n,n} : I_n^2 \to [0,1]$ the following statements are equivalent:*

1. *$C_{n,n}$ is a discrete copula;*
2. *there is a doubly stochastic matrix $A = (a_{ij})_{i,j=1}^n$ such that for $i, j \in \{0, 1, 2, \ldots, n\}$*

$$C_{n,n}\left(\frac{i}{n}, \frac{j}{n}\right) = \frac{1}{n} \sum_{k=1}^{i} \sum_{m=1}^{j} a_{km}.$$

Remark 4.1.4. *Note that the doubly stochastic matrix in the previous proposition represents the scaled copula density of the associated checkerboard copula. This is because the doubly stochastic matrix can be identified with the probability mass that the probability measure assigns to each unit square. The copula density matrix can be obtained by considering the matrix $(c_{ij})_{i,j=1}^n$, where $c_{ij} = n \cdot a_{ij}$.*

Example 4.1.1. *Let **A** be the doubly-stochastic matrix:*

$$\boldsymbol{A} = \begin{bmatrix} 1/7 & 3/7 & 3/7 \\ 2/7 & 1/7 & 4/7 \\ 4/7 & 3/7 & 0 \end{bmatrix}.$$

*By applying the transformation from proposition 4.1.1, we obtain the discrete copula C associated with **A***

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1/21 & 4/21 & 1/3 \\ 0 & 3/21 & 7/21 & 2/3 \\ 0 & 1/3 & 2/3 & 1 \end{bmatrix}.$$

*By scaling **A**, the copula density c can be obtained, and the piecewise constant property of the copula density can be visualised in the following figure:*

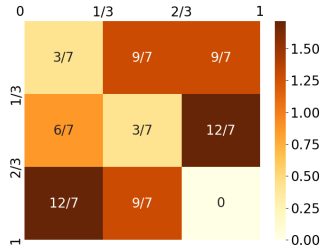


Figure 4.2: Corresponding copula density. The copula density c is constant on squares of area $\frac{1}{3^2}$.

The following results link the discrete copulas with some known combinatorial results, and are discussed in [30].

Theorem 4.1.5. *The set of discrete copulas is convex. That is, if C_1 and C_2 are discrete copulas and $t \in [0,1]$, then $tC_1 + (1-t)C_2$ is also a discrete copula.*

Theorem 4.1.6 (Birkhoff-von Neumann)**.** *The extreme points of the polytope representing the set of $n \times n$ doubly stochastic matrices are the $n \times n$ permutation matrices. That is, $n \times n$ matrices such that each row and each column has exactly one entry equal to 1 and all other entries equal to 0.*

Remark 4.1.5. *Birkhoff-von Neumann Theorem gives a characterisation of the extreme points of the discrete copula polytope on a grid of size n. By Remark 4.1.4, the copula density for a discrete copula can be obtained through a scaling of a doubly stochastic matrix with a factor of n. The extreme points are therefore matrices such that each row and each column has exactly one entry equal to n and all other entries equal to 0.*

Theorem 4.1.7. *[33] For every copula distribution C, there exists a sequence of checkerboard copulas $(\hat{C}_n)_{n=0}^{\infty}$ that converge to C uniformly. In particular, the sequence $(\hat{C}_n)_{n=0}^{\infty}$ can be chosen such that $\hat{C}_n$ is a checkerboard copula on a grid of size n and:*

$$\sup_{(x,y)\in[0,1]^2} |C(x,y) - \hat{C}_n(x,y)| \leq \frac{2}{n}, \forall n \in \mathbb{N}.$$

The theorem above shows that by considering checkerboard copulas on grids of increasing granularity, any copula can be uniformly approximated.

4.1.3 Pair-copula Constructions

Pair-copula constructions are a powerful method for building flexible multivariate distributions by decomposing complex dependencies into simpler, bivariate copulas. This section introduces the fundamental concepts behind these constructions.

The key idea behind pair-copula constructions is that a high-dimensional probability distribution can be decomposed by successively rewriting joint distributions in terms of conditional distributions.

Pair-copula decomposition

In this section, the pair-copula constructions used in decomposing high-dimensional distributions are introduced, as presented in [28]. Consider $X_1, X_2, \dots X_d$ continuous (real-valued) random variables. In the derivations below, $F(\cdot)$ and $p(\cdot)$ denote conditional cdf's and probability density functions, respectively. Throughout this chapter, the full expression of a conditional copula and its abbreviation are used interchangeably:

$$c_{i,j|i_1,\dots,i_k} \triangleq c_{i,j|i_1,\dots,i_k}(F(x_i \mid x_{i_1},\dots,x_{i_k}), F(x_j \mid x_{i_1},\dots,x_{i_k})),$$

for distinct indices $i, j, i_1, \dots, i_k$ with $i < j$ and $i_1 < \cdots < i_k$.

Consider the following decomposition using conditional pdfs:

$$\begin{aligned} p(x_1,\dots,x_d) &= p(x_d \mid x_1,\dots,x_{d-1})p(x_1,\dots,x_{d-1}) \\ &= \cdots = \prod_{t=2}^{d} p(x_t \mid x_1,\dots,x_{t-1})p(x_1). \end{aligned} \tag{4.1}$$

Sklar's theorem for dimension $d = 2$ gives $p(x_1, x_2) = c_{12}(F_1(x_1), F_2(x_2))p_1(x_1)p_2(x_2)$, where $c_{12}(\cdot,\cdot)$ is an arbitrary bivariate copula density. Using Eq. (4.1), the conditional density of X_1 given X_2 can be expressed as

$$p(x_1 \mid x_2) = c_{12}(F_1(x_1), F_2(x_2))p_1(x_1). \tag{4.2}$$

Using Eq. (4.2), the pdf $p(x_t \mid x_1,\dots,x_{t-1})$ can now be expressed recursively as

$$\begin{aligned} p(x_t \mid x_1,\dots,x_{t-1}) &= c_{1,t|2,\dots,t-1} \cdot p(x_t \mid x_2,\dots,x_{t-1}) \\ &= \left[\prod_{s=1}^{t-2} c_{s,t|s+1,\dots,t-1}\right] \cdot c_{(t-1),t} \cdot p_t(x_t). \end{aligned} \tag{4.3}$$

Using Eq. (4.3) in Eq. (4.1) and taking $s = i, t = i + j$ gives:

$$\begin{aligned} p(x_1,\dots,x_d) &= \left[\prod_{t=2}^{d}\prod_{s=1}^{t-2} c_{s,t|s+1,\dots,t-1}\right]\left[\prod_{t=2}^{d} c_{(t-1),t}\right]\left[\prod_{k=1}^{d} p_k(x_k)\right] \\ &= \left[\prod_{j=1}^{d-1}\prod_{i=1}^{d-j} c_{(i,(i+j))|(i+1),\dots,(i+j-1)}\right]\left[\prod_{k=1}^{d} p_k(x_k)\right]. \end{aligned} \tag{4.4}$$

The decomposition above is called a *pair-copula decomposition*, and this decomposition allows any high-dimensional decomposition to be written only in terms of pair-wise interactions (through the bivariate copulas) and the marginal distributions.

Regular vine decomposition

According to [28], Bedford and Cooke [34] observed that pair-copula decompositions above can be represented graphically as *vine trees*, a sequence of nested trees with undirected edges, where edges denote indices used for the conditional copula densities. A regular vine on d variables consists of connected trees $T_1, \dots, T_{d-1}$ with nodes N_i and edges E_i for $i = 1, \dots, d-1$, satisfying:

1. T_1 has nodes $N_1 = \{1, \ldots, d\}$ and edges E_1.

2. For $i = 2, \ldots, d-1$, tree T_i has nodes $N_i = E_{i-1}$.

3. Two edges in T_i are joined in T_{i+1} if they share a common node in T_i.

Edges in T_i are denoted by $jk|D$, where $j < k$ and D is the conditioning set. If D is empty, the edge is denoted jk. Consider two edges $a = j(a), k(a)|D(a)$ and $b = j(b), k(b)|D(b)$ in T_{i-1} that share a node. Nodes corresponding to a and b are then joined in T_i by edge $e = j(e), k(e)|D(e)$, where $D(e)$ are the random variables that appear in both a and b, while $j(e)$ and $k(e)$ represent the components that appear only in a or b, respectively.

Regular tree sequences can therefore help visualise the interactions between random variables of a system and these can give multiple equivalent decompositions for the same joint probability distribution.

A regular vine tree is called

- drawable vine tree (D-vine tree) if each node in the sequence $(T_i)_{i=1}^{d-1}$ has at most 2 edges
- canonical vine tree (C-vine tree) if each tree T_i has a node with degree $d - i$.

Therefore, a D-vine tree is fully specified by the initial tree T_1, while the C-vine tree is fully specified by the root nodes in each T_i.

4.2 Formulate extension of synergy to continuous distributions

The synergy measure described in [12, 8] is originally defined for discrete probability distributions, but it can be directly extended to continuous probability distributions. The key idea behind discrete private disclosure capacity is to construct a discrete random variable that remains private with respect to all sources while sharing as much information as possible about the target variable. Therefore, the scenario can be kept identical to the discrete case, but with discrete random variables replaced by continuous ones. The setup is described again for clarity.

Scenario

Consider $W, X_1, \ldots, X_n$ continuous random variables with continuous domains $\mathcal{W}, \mathcal{X}_1, \ldots \mathcal{X}_n$, and a given joint probability density function $p_{W,X_1,\ldots,X_n}$. By considering the same definitions for the set of admissible stochastic mappings $\mathcal{A}_X = \{\, p_{Y|X} \mid Y \perp\!\!\!\perp X_i, \forall i \in [1:n] \,\}$, the definition for the private disclosure capacity remains the same:

$$I_s \triangleq \max_{\substack{p_{Y|X} \in \mathcal{A}_X \\ W - X - Y}} I(W;Y). \tag{4.5}$$

This chapter focuses on two cases of synergy in continuous distributions. We begin by exploring the case of self-disclosure, when the target variable is the system itself, and then proceed to the case of synergy with an arbitrary target, where all random variables in the system are real-valued. In both cases, to build intuition, we begin by examining the simpler case of two sources, and then prove generalisations of the results for an arbitrary number of sources.

4.3 Self-disclosure

We start by presenting a particular case of the formal definition of synergy introduced in the previous section, by focusing on the amount of information that a system can disclose about itself. In this section, we introduce the notion of *self-disclosure* and discuss some properties of this measure.

Definition 4.3.1. *The self-disclosure of $\boldsymbol{X} = \{X_1, X_2, \ldots, X_n\}$ is defined as the disclosure capacity when the target is given by $W = \boldsymbol{X}$ in Eq. (4.5). Formally, this is defined as:*

$$I_s \triangleq \max_{p_{Y|X} \in \mathcal{A}_X} I(\boldsymbol{X};Y).$$

We initially focus on the special case of just two sources, as the insights gained from this case help in understanding self-disclosure involving multiple sources. In this case, the definition above becomes:

$$I_s = \max_{Y \perp\!\!\!\perp X_1, Y \perp\!\!\!\perp X_2} I(X_1, X_2; Y).$$

4.3.1 Modelling constraints through copulas

Lemma 4.3.1. *For any private disclosure channel Y ($p_{Y|X} \in \mathcal{A}_X$), the joint probability distribution $p(y, x_1, x_2)$ can be written using pair copula constructions as:*

$$p(y, x_1, x_2) = p(y)p(x_1)p(x_2)c_{12|\theta(u_y)}(u_1, u_2),$$

where $c_{12|\theta(u_y)}$ represents the conditional copula of (X_1, X_2) given Y (where $(X_1, X_2)|Y$ depends on Y only through the parametrisation function $\theta(u_y)$).

Proof. Consider the following D-vine sequence of trees $(T_i)_{i=1}^2$:

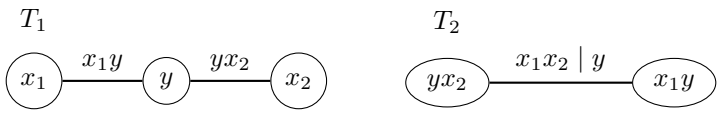


Figure 4.3: D-vine tree decomposition for the joint distribution (Y, X_1, X_2).

By the pair-copula construction Eq. (4.4) for $d = 3$, for any joint distribution (Y, X_1, X_2), the following relation holds:

$$p(y, x_1, x_2) = p(y)p(x_1)p(x_2)c_{12|y}(u_{1|y}, u_{2|y})c(u_y, u_1)c(u_y, u_2).$$

By $p_{Y|X} \in \mathcal{A}_X$, we also have that $Y \perp\!\!\!\perp X_1$ and $Y \perp\!\!\!\perp X_2$, so $\forall u_y, u_1, u_2 \in [0, 1], c(u_y, u_1) = c(u_y, u_2) = 1$, and $u_{1|y} = F_{X_1|Y}(x_1|y) = F_{X_1|Y}(x_1) = u_1, u_{2|y} = F_{X_2|Y}(x_2|y) = F_{X_2|Y}(x_2) = u_2$. Substituting these in the relation above, we obtain the decomposition. □

A similar expression for the decomposition above was given in [35]. Here, we apply it to the synergy measure we work with, and we also provide a complete proof of the decomposition.

This gives the following parametrisation of the space $\mathcal{A}_X$:

Lemma 4.3.2. *The copula density $c_{12|y}$, together with $p(y)$, parametrise the feasible set $\mathcal{A}_X$.*

Proof. Consider $Y \in \mathcal{A}_X$. This is specified by the conditional $p_{Y|X}$ (given for each $y \in \mathcal{Y}, (x_1, x_2) \in \mathcal{X}^2$). This is equivalent to specifying the joint pdf $p(y, x_1, x_2)$. By the previous proposition, this is given by $p(y)p(x_1)p(x_2)c_{12|\theta(u_y)}(u_1, u_2)$. Therefore, as $p(x_1), p(x_2)$ are known, the private disclosure channels are fully specified by the pair $(p(y), c_{12|y})$ (where $p(y)$ is obtained from $p_{Y|X}$ and $\mathbf{p}_X$). Conversely, for a given pair $(p(y), c_{12|y})$, the joint pdf $p(y, x_1, x_2)$ is uniquely defined by setting the copulas $c(u_y, u_1)$ and $c(u_y, u_2)$ to be independence copulas. □

Proposition 4.3.1. *Self-disclosure I_s in the special case of $n = 2$ has the following form:*

$$I_s = \max_{\theta(\cdot)} \int_{[0,1]^3} du_y du_1 du_2 c_{12|\theta(u_y)}(u_1, u_2) \log \frac{c_{12|\theta(u_y)}(u_1, u_2)}{c(u_1, u_2)}.$$

Proof. In the self-disclosure case $W = (X_1, X_2)$, and $I_s = \max_{Y \in \mathcal{A}_X} I(W; Y)$ Take $Y \in \mathcal{A}_X$, arbitrarily chosen. The mutual information chain rule gives $I(W; Y) = I(X_2; Y) + I(X_1; Y|X_2)$. Since Y is independent of X_2, $I(X_2; Y) = 0$, so $I(W; Y) = I(X_1; Y|X_2)$. By the definition of mutual information:

$$\begin{aligned} I(X_1; Y|X_2) &= \int dy dx_1 dx_2 p(y, x_1, x_2) \log \frac{p(y, x_1|x_2)}{p(y|x_2)p(x_1|x_2)} \\ &= \int dy dx_1 dx_2 p(y)p(x_1)p(x_2)c(u_y, u_1, u_2) \log \frac{p(y)p(x_1)p(x_2)c(u_y, u_1, u_2)}{p(y)p(x_1)p(x_2)c(u_1, u_2)} \\ &= \int_{[0,1]^3} du_y du_1 du_2 c(u_y, u_1, u_2) \log \frac{c(u_y, u_1, u_2)}{c(u_1, u_2)}, \end{aligned}$$

where the last equality follows by the changes of variables $p(y)dy = du_y, p(x_1)dx_1 = du_1, p(x_2)dx_2 = du_2$.

Considering the decomposition in lemma 4.3.1, $c(u_y, u_1, u_2) = c_{12|\theta(u_y)}(u_1, u_2)$. By lemma 4.3.2, I_s is obtained by maximising over the all parametrisation functions $\theta(\cdot)$:

$$\begin{aligned} I_s &= \max_{Y \in \mathcal{A}_X} \int_{[0,1]^3} du_y du_1 du_2 c(u_y, u_1, u_2) \log \frac{c(u_y, u_1, u_2)}{c(u_1, u_2)} \\ &= \max_{\theta(\cdot)} \int_{[0,1]^3} du_y du_1 du_2 c_{12|\theta(u_y)}(u_1, u_2) \log \frac{c_{12|\theta(u_y)}(u_1, u_2)}{c(u_1, u_2)}. \end{aligned}$$

□

The property above shows that when constructing the optimal private disclosure channel, the constraints can be separated from the optimisation problem. The random variable Y is constructed through the joint pdf $p(y, x_1, x_2)$, and through 4.3.1 we can set copula terms individually to reconstruct the joint distribution. More precisely, the two joint copulas $c(u_y, u_1)$ and $c(u_y, u_2)$ are set to the independence copulas, and the copula $c_{12|\theta}$ can be set to an arbitrary copula which models the interaction between X_1 and X_2 when conditioned on Y.

The next property shows that, assuming regularity conditions regarding continuity, in the optimal case the parametrisation $\theta(\cdot)$ does not depend on the argument, and $\theta(\cdot)$ can be set to a constant value over the entire domain.

Proposition 4.3.2. *Self-disclosure I_s, in the case of $n = 2$, is maximised for a constant parametrisation function $\theta(u_y) = \theta^*$.*

The self-disclosure then becomes:

$$I_s = \int_{[0,1]^2} du_1 du_2 c_{12|\theta^*}(u_1, u_2) \log \frac{c_{12|\theta^*}(u_1, u_2)}{c(u_1, u_2)}.$$

Proof. Take $\theta(\cdot)$ a parametrisation function, with domain $[0, 1]$. Now consider the function $S_{\theta(\cdot)} : [0, 1] \to \mathbb{R}$,

$$S_{\theta(\cdot)}(u_y) = \int_{[0,1]^2} du_1 du_2 c_{12|\theta(u_y)}(u_1, u_2) \log \frac{c_{12|\theta(u_y)}(u_1, u_2)}{c(u_1, u_2)}.$$

The function $S_{\theta(\cdot)}$ is a continuous function defined on the compact interval $[0, 1]$, and therefore attains its maximal value in a point $\theta^* \in [0, 1]$. By definition of θ^*, we get the point-wise inequality:

$$\int_{[0,1]^2} c_{12|\theta(u_y)}(u_1, u_2) \log \frac{c_{12|\theta(u_y)}(u_1, u_2)}{c(u_1, u_2)} \leq \int_{[0,1]^2} c_{12|\theta^*}(u_1, u_2) \log \frac{c_{12|\theta^*}(u_1, u_2)}{c(u_1, u_2)}, \forall u_y \in [0, 1].$$

By integrating in $[0, 1]$:

$$\begin{aligned} \int_{[0,1]^3} du_y du_1 du_2 c_{12|\theta(u_y)}(u_1, u_2) \log \frac{c_{12|\theta(u_y)}(u_1, u_2)}{c(u_1, u_2)} \leq \\ \leq \int_{[0,1]^2} du_1 du_2 c_{12|\theta^*}(u_1, u_2) \log \frac{c_{12|\theta^*}(u_1, u_2)}{c(u_1, u_2)}. \end{aligned}$$

Therefore, by defining another parametrisation function $\theta_2(u_y) = \theta^*$ we obtain a larger disclosure, so I_s is achieved for a constant parametrisation function. □

Without the continuity of $S(\cdot)$ in the result above, we can assume simplification of the problem which considers a constant parametrisation $\theta(\cdot) = \theta^*$. In [36], the authors examine whether such a parametrisation is general enough, and claim that this restriction is not too severe.

Note that in the expression for I_s above, any parametrisation $c_{12|\theta}$ leads to a valid private disclosure channel. Therefore, there are no other constraints on the conditional copula, and the maximisation of I_s happens over the entire copula set.

Proposition 4.3.3. *The function $S : \mathcal{C} \to [0, \infty]$, defined on the set of bivariate copulas, with $S(V) = \int_{[0,1]^2} v(u_1, u_2) \log \frac{v(u_1,u_2)}{c(u_1,u_2)} du_1 du_2$ is convex, the supremum is achieved and lies on the boundary of the set of copulas.*

Proof. Take $V_1, V_2 \in \mathcal{C}$ arbitrary bivariate copulas with v_1, v_2 the respective copula densities. The set of copulas is convex, so for any $\lambda \in [0,1]$, $V = \lambda V_1 + (1-\lambda)V_2$ is a bivariate copula, with density $v = \lambda v_1 + (1-\lambda)v_2$. Since $x \mapsto x \log \frac{x}{c}$ is convex for any $c > 0$, from Jensen's inequality we have:

$$v(u_1,u_2)\log\frac{v(u_1,u_2)}{c(u_1,u_2)} \leq \lambda v_1(u_1,u_2)\log\frac{v_1(u_1,u_2)}{c(u_1,u_2)} + (1-\lambda)v_2(u_1,u_2)\log\frac{v_2(u_1,u_2)}{c(u_1,u_2)},$$

for $u_1, u_2 \in [0,1]$ fixed. Since for points where $c(u_1,u_2) = 0$, both sides are ∞, by integrating in $[0,1]^2$ we get the following inequality:

$$\int_{[0,1]^2} v(u_1,u_2)\log\frac{v(u_1,u_2)}{c(u_1,u_2)} \leq \lambda\int_{[0,1]^2} v_1(u_1,u_2)\log\frac{v_1(u_1,u_2)}{c(u_1,u_2)} + (1-\lambda)\int_{[0,1]^2} v_2(u_1,u_2)\log\frac{v_2(u_1,u_2)}{c(u_1,u_2)},$$

which shows $S(V) \leq \lambda S(V_1) + \lambda S(V_2)$, so S is convex. As the domain of S is convex, we get that S achieves its maximum value on the boundary of the domain (the set of bivariate copulas is compact, so the set is closed and bounded, and the maximum value of S is achieved). □

The proposition above shows that since the optimisation of I_s happens over a convex set, the optimal value is achieved for an extreme copula. In order to find this optimal value, we now consider a discretised version of the optimisation problem, and through the use of checkerboard copulas defined in the previous section, we show that the optimal disclosure is unbounded.

4.3.2 Approximation with checkerboard copula

In this section, we consider the same scenario of self-disclosure in the case of two sources, with $X = \{X_1, X_2\}$ a system of continuous random variables with copula density $c(u_1,u_2)$. We also consider V_n a checkerboard copula defined on a grid of size n for some $n \in \mathbb{N}$, with copula density $v_{ij}, \forall i,j \in [1:n]$ (the value of the copula density in the rectangle $[\frac{i-1}{n}, \frac{i}{n}] \times [\frac{j-1}{n}, \frac{j}{n}]$).

Let $(c_{ij})_{i,j=0}^n$ be the mean of logarithm of the copula density of (X_1, X_2) in the rectangle $[\frac{i-1}{n}, \frac{i}{n}] \times [\frac{j-1}{n}, \frac{j}{n}]$:

$$c_{ij} = n^2 \int_{[\frac{i-1}{n},\frac{i}{n}]\times[\frac{j-1}{n},\frac{j}{n}]} \log c(u_1,u_2)du_1du_2.$$

The self-disclosure capacity of V_n is equal to:

$$\begin{aligned}
S(V_n) &= \int_{[0,1]^2} v(u_1,u_2)\log\frac{v(u_1,u_2)}{c(u_1,u_2)}du_1du_2 \\
&= \sum_{i,j=1}^n \int_{[\frac{i-1}{n},\frac{i}{n}]\times[\frac{j-1}{n},\frac{j}{n}]} v(u_1,u_2)\log\frac{v(u_1,u_2)}{c(u_1,u_2)}du_1du_2 \\
&= \sum_{i,j=1}^n \int_{[\frac{i-1}{n},\frac{i}{n}]\times[\frac{j-1}{n},\frac{j}{n}]} v_{ij}\log\frac{v_{ij}}{c(u_1,u_2)}du_1du_2 \\
&= \sum_{i,j=1}^n \frac{1}{n^2}v_{ij}(\log v_{ij} - c_{ij}) = \frac{1}{n^2}\sum_{i,j=1}^n v_{ij}(\log v_{ij} - c_{ij}).
\end{aligned}$$

The set of discrete copulas of size n is a convex polytope, and it can be shown that $S(V_n)$ is a convex function on the set of discrete copulas of size n (using similar arguments to proposition 4.3.3). The maximum value of $S(V_n)$ is therefore achieved when V_n is on the boundary of the feasible set, and corresponds to an extreme point in the polytope of discrete copulas of size n. By 4.1.6, the extreme points of this convex polytope correspond to the extreme points of Birkhoff's polytope, and represent permutations of nI_n, where I_n is the identity matrix of size n.

For a fine enough grid, we can assume $c(u_1,u_2)$ is constant in the square $[\frac{i-1}{n}, \frac{i}{n}] \times [\frac{j-1}{n}, \frac{j}{n}]$ (with value c_{ij}). With this assumption, $S(V_n)$ can be simplified, and we will use the following expression throughout this section:

$$S(V_n) = \frac{1}{n^2}\sum_{i,j=1}^n v_{ij}\log\frac{v_{ij}}{c_{ij}}. \tag{4.6}$$

Proposition 4.3.4. *For any extreme point V_n of $\mathcal{DC}_n$, $\sum_{i,j=1}^n v_{ij} \log v_{ij} = n^2 \log n$.*

Proof. By 4.1.6, the extreme points of the convex polytope $\mathcal{DC}_n$ are permutations of nI_n. Considering the convention $0 \times \log 0 = 0$, we have exactly n elements in the sum that are equal to $n \log n$, while the rest are equal to 0. Therefore, $\sum_{i,j=1}^n v_{ij} \log v_{ij} = n^2 \log n$, for any extreme point V_n. □

Proposition 4.3.5. *There exists a permutation $(v_{ij})_{i,j=1}^n$ of nI_n such that $\sum_{i,j=1}^n v_{ij} \log c_{ij} \leq 0$.*

Proof. The function above is linear in $(v_{ij})_{i,j=1}^n$, so the minimal value is achieved for an extreme point of the polytope $\mathcal{DC}_n$. By Birkhoff-von Neumann theorem 4.1.6, this is achieved for a permutation of nI_n. Let the minimal value be $n \sum_{i=1}^n \log c_{i\pi(i)} = n \log \prod_{i=1}^n c_{i\pi(i)}$, for some permutation $\pi : [1:n] \to [1:n]$.

As $(c_{ij})_{i,j=0}^n$ corresponds to the copula density of (X_1, X_2), we have $c_{ij} \geq 0$ and $\sum_{i=0}^n c_{ij} = \sum_{j=0}^n c_{ij} = n$.

By the inequality between arithmetic mean and geometric mean, we get

$$\sqrt[n^2]{\prod_{i,j=1}^n c_{ij}} \leq \frac{\sum_{i,j=1}^n c_{ij}}{n^2} = 1 \implies \prod_{i,j=1}^n c_{ij} \leq 1.$$

There exists a permutation π such that $\prod_{i=1}^n c_{i\pi(i)} \leq 1$, as otherwise the product above is > 1, leading to a contradiction with the copula density constraints on $c(u_1, u_2)$. Therefore, there exists a permutation such that $n \sum_{i=1}^n \log c_{i\pi(i)} \leq 0$. □

Theorem 4.3.1. *The self-disclosure capacity in a system with two sources is unbounded.*

Proof. Combining the two propositions above shows that for all $n \in \mathbb{N}$, there exists an extreme discrete copula $V_n^* \in \mathcal{DC}_n$ such that $S(V_n^*) \geq \frac{1}{n^2} \cdot n^2 \log n + 0 = \log n$. Since any discrete copula V_n is a valid copula distribution (with piecewise constant copula density on squares of area $\frac{1}{n^2}$), $S(V_n^*)$ provides a lower bound for I_s, and since $S(V_n^*) \to \infty$ as $n \to \infty$, the self-disclosure for a system with two sources is unbounded. □

Remark 4.3.1. *This result agrees with [13], where the authors also prove that the disclosure is unbounded in the case of $W = (X_1, X_2)$. However, the results are only shown for the case of two sources. In the following section, we show that the results derived through the use of pair-copula decompositions can be extended to arbitrary number of sources, and later on, to the general case of synergy.*

4.3.3 Associated optimisation problem

The optimisation of $S(V_n)$ for V_n a discrete copula can be seen as an optimisation problem, where $S(V_n)$ is the objective function and the conditions V_n needs to satisfy to be a discrete copula are the constraints of the problem. Therefore, the associated optimisation problem is:

$$\begin{aligned} &\text{maximise} \quad \frac{1}{n^2} \sum_{i,j=1}^n v_{ij} \log \frac{v_{ij}}{c_{ij}} \\ &\text{subject to} \quad \forall i,j \in [1:n] \; v_{ij} \geq 0, \\ &\forall i \in [1:n] \; \frac{1}{n} \sum_{j=1}^n v_{ij} = 1, \; \forall j \in [1:n] \; \frac{1}{n} \sum_{i=1}^n v_{ij} = 1. \end{aligned} \tag{4.7}$$

Note that in the optimisation problem above, $(c_{ij})_{i,j=0}^n$ are fixed coefficients. Therefore, the optimal solution of this problem is at the boundary of the feasible set. This is confirmed numerically, as the problem is a convex-concave optimisation problem (the objective of the maximisation problem is convex), and the problem can be expressed as a DCCP program. Hence, the solution can be computed by a solver programmatically, for a given set of coefficients representing the initial copula $c(u_1, u_2)$.

4.3.4 Self-disclosure in systems with arbitrary number of sources

Lemma 4.3.3. *For any private disclosure channel Y ($p_{Y|X} \in \mathcal{A}_X$), the joint probability distribution $p(y, x_1, \ldots, x_d)$ can be written using pair-copula constructions as:*

$$p(y, x_1, \ldots, x_d) = p(y) \prod_{i=1}^{d} p(x_i) c_{12\ldots d|\theta(u_y)}(u_1, \ldots, u_d),$$

where $c_{12\ldots d|\theta(u_y)}$ represents the conditional copula of $(X_1, \ldots, X_d)$ given Y.

Proof. Using Sklar's Theorem applied to the joint distribution $(X_1|Y, \ldots, X_d|Y)$, we get:

$$\begin{aligned} p(y, x_1, \ldots, x_d) &= p(x_1, \ldots, x_d|y)p(y) \\ &= p(y) \prod_{i=1}^{d} p(x_i) c_{1\ldots d|y}(u_{1|y}, u_{2|y}, \ldots, u_{d|y}) \\ &= p(y) \prod_{i=1}^{d} p(x_i) c_{1\ldots d|y}(u_1, \ldots, u_d), \end{aligned}$$

where the last equality follows from the fact that $Y \perp\!\!\!\perp X_i, \forall i \in [1:d]$. □

By assuming either a constant parametrisation function $\theta(u_y) = \theta^*$ or the continuity of the disclosure as a function of u_y, a similar derivation holds for arbitrary sources as well. Using the previous lemma 4.3.3, the self-disclosure capacity I_s becomes:

$$I_s = \int_{[0,1]^d} c_{12\ldots d|\theta^*}(u_1, \ldots, u_d) \log \frac{c_{12\ldots d|\theta^*}(u_1, \ldots, u_d)}{c(u_1, \ldots, u_d)} du_1 \ldots du_d.$$

This expression can be similarly approximated using checkerboard copulas, leading to an analogous optimisation problem:

$$\begin{aligned} &\text{maximise } \frac{1}{n^d} \sum_{i_1,\ldots i_d=1}^{n} v_{i_1\ldots i_d} \log \frac{v_{i_1\ldots i_d}}{c_{i_1\ldots i_d}} \\ &\text{subject to } \forall i_1 \ldots i_d \in [1:n]\ v_{i_1\ldots i_d} \geq 0, \\ &\qquad \forall k \in [1:d], i \in [1:n]\ \frac{1}{n^{d-1}} \sum_{i_1,\ldots i_{k-1}, i_{k+1}, \ldots i_d=1}^{n} v_{i_1,\ldots i_{k-1}, i, i_{k+1}, \ldots i_d} = 1. \end{aligned}$$

Note that the computational complexity of the associated optimisation problem increases very quickly with the number of sources d. This is because the optimisation in this case happens over a grid in d dimensions, so the number of variables for a grid size of n is n^d.

Theorem 4.3.2. *The self-disclosure capacity in a system with an arbitrary number of sources is unbounded.*

Sketch of proof. We can prove in a similar way that the disclosure is unbounded in this case. As the function we wish to maximise is convex, the optimal solution is in one of the extreme points of the polytope of checkerboard copulas in d dimensions.

Consider the points corresponding to permutations of the identity tensor in d dimensions. These represent the extreme points of the polytope [37]. For these points,

$$\frac{1}{n^d} \sum_{i_1,\ldots i_d=1}^{n} v_{i_1\ldots i_d} \log v_{i_1\ldots i_d} = \frac{1}{n^d} \cdot n \cdot n^{d-1} \log(n^{d-1}) = \log n^{d-1} \to \infty, \text{ as } n \to \infty.$$

For the other term, we can similarly apply the AM-GM inequality to get that there exists a permutation of the identity tensor in d dimensions π (which we represent as $\epsilon : [1:n]^d \to \{0,1\}$, where ϵ takes the value 1 for elements in the permutation, and 0 otherwise) for which $\sum_{i_1,\ldots i_d=1}^{n} \pi(i_1, \ldots, i_d) \log c_{i_1\ldots i_d} \leq 0$. This permutation corresponds to an extreme point, and by taking this extreme point we get that the disclosure is unbounded.

□

4.3.5 Numerical Simulations

The associated optimisation problem can be solved for a given grid size n, so we can formulate the problem using `CVXPY`, and use a solver such as `GUROBI` to analyse numerically the optimal values. We present below the numerical simulations in the case of two sources, for multiple initial copulas $c(u_1, u_2)$. As an example for the applications of checkerboard copulas approximations, in the simulations below we use a grid of size 10.

Impact of grid size

In this case, we consider independent sources and we have $c(u_1, u_2) = 1, \forall u_1, u_2 \in [0, 1]$. Therefore, $S(V_n) = \frac{1}{n^2} \sum_{i,j=1}^{n} v_{ij} \log v_{ij}$ for all $V_n \in \mathcal{DC}_n$. Therefore, based on proposition 4.3.4, we expect to have $S(V_n) = \log n$ for any extreme point of $\mathcal{DC}_n$. Indeed, this is confirmed numerically, as the optimal value for a grid of size n is always achieved for a permutation of nI_n and is equal to $\log n$.

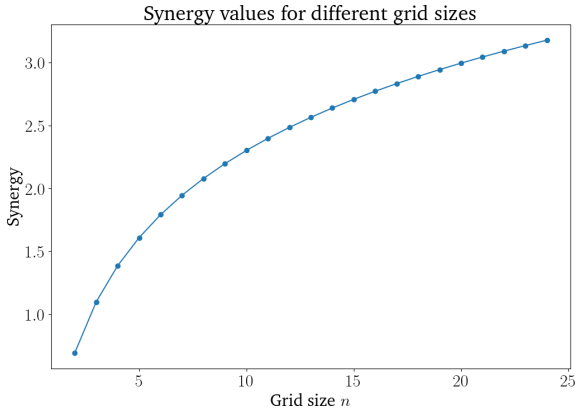


Figure 4.4: Self-disclosure with independent sources. The synergy grows with finer grids, and the optimal value is always the synergy lower bound $\log n$.

Gaussian copula

A potentially interesting application of self-disclosure is in Gaussian systems. Here, we now consider $(X_1, X_2) \sim \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}1 & \rho\\ \rho & 1\end{bmatrix}\right)$. We want to observe how self-disclosure is influenced by the correlation between the components in the system. We expect that the lowest value to be achieved for independent sources, where the self-disclosure is exactly $\log n$, and as the variables become more correlated, we expect the synergy to grow, as the contributions of $\sum_{i,j=1}^{n} v_{ij} \log c_{ij}$ become more meaningful.

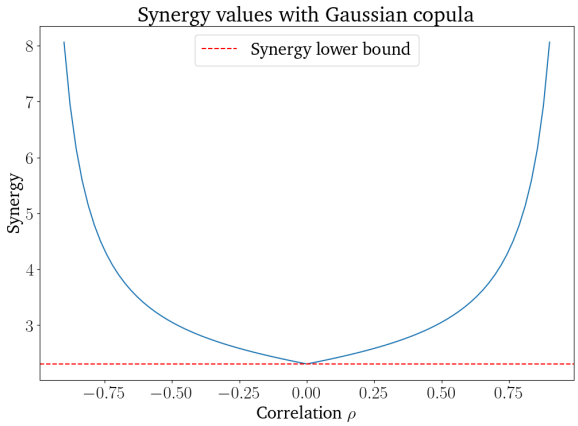


Figure 4.5: Self-disclosure in systems with Gaussian copula. The synergy grows as the sources become more correlated, and is always greater than the synergy lower bound $\log n$ (represented by the dashed, red line).

Gumbel copula

Another common choice for modelling dependence between random variables is the Gumbel copula. Note that when $\theta = 1$, the Gumbel copula is the independence copula, and when $\theta \to \infty$, the Gumbel copula converges to the comonotonicity copula. Therefore, we expect the self-disclosure to be equal to $\log n$ for $\theta = 1$, and we expect it to grow quickly with the value of θ.

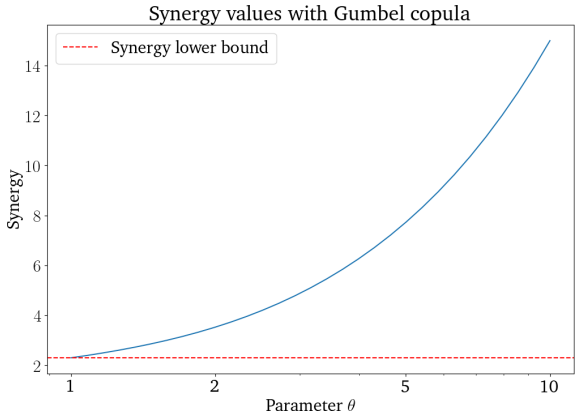


Figure 4.6: Self-disclosure in systems with Gumbel copula. The synergy grows as θ increases, and is always greater than the synergy lower bound $\log n$ (represented by the dashed, red line).

4.4 Synergy with arbitrary target

In this section, we shift our attention back to the general case of synergy, and we offer a method that can be used to compute the synergistic disclosure with an arbitrary (real-valued) target. As in the previous section, we focus on the case where all the random variables involved are one-dimensional.

Let $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ be the set of sources consisting of continuous random variables, and W the continuous target variable. Using the equivalence between total correlation and copula entropy from Theorem 4.1.4, for any $Y \in \mathcal{A}_X$, we can write:

$$I(W;Y) = -H_c(W;Y) = \int_{[0,1]^2} c(u_w, u_y) \log c(u_w, u_y)\, du_w du_y. \tag{4.8}$$

To find the Y that maximises the quantity above, we can take advantage of copula theory described in the previous section. In particular, pair-copula constructions allow the decoupling of interactions within components in the system, making it significantly easier to satisfy the constraints.

4.4.1 Synergy in systems with two sources

As a first step, we consider systems with two sources. We begin by constructing a convenient representation of the joint distribution (Y, W, X_1, X_2) through a pair-copula decomposition.

Lemma 4.4.1. *For any private disclosure channel Y ($p_{Y|X} \in \mathcal{A}_X$, $W - X - Y$ Markov chain), the joint probability distribution $p(y, w, x_1, x_2)$ can be written using pair-copula constructions as:*

$$p(y, w, x_1, x_2) = p(y)p(w)p(x_1)p(x_2)c_{12|\theta(u_y)}(u_1, u_2)\frac{c(u_1, u_2, u_w)}{c(u_1, u_2)},$$

where $c_{12|\theta(u_y)}$ represents the conditional copula of (X_1, X_2) given Y (where $(X_1, X_2)|Y$ depends on Y only through the parametrisation function $\theta(u_y)$).

Proof. Take the following D-vine tree decomposition for the joint distribution (Y, W, X_1, X_2) :

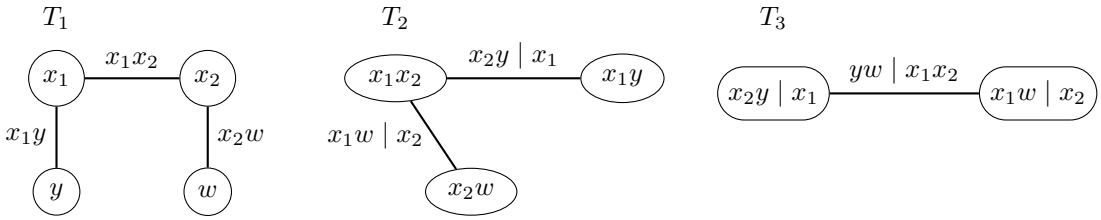


Figure 4.7: D-vine tree decomposition for the joint distribution (Y, W, X_1, X_2).

Based on this decomposition, the joint distribution can be written as:

$$p(y,w,x_1,x_2) = p(y)p(w)p(x_1)p(x_2)c_{x_1y}(u_1,u_y)c_{x_1x_2}(u_1,u_2)c_{x_2w}(u_2,u_w)\cdot \\ \cdot c_{yx_2|x_1}(u_{y|1},u_{2|1})c_{x_1w|x_2}(u_{1|2},u_{w|2})c_{yw|x_1x_2}(u_{y|x_1x_2},u_{w|x_1x_2}).$$

Since $W-X-Y$ form a Markov Chain, Y and W are independent given (X_1, X_2), so $c_{yw|x_1x_2} = 1$. From lemma 4.3.1, by symmetry, we also have that $c_{yx_1x_2} = c_{yx_1}c_{yx_2}c_{x_1x_2|y} = c_{yx_1}c_{x_1x_2}c_{yx_2|x_1}$. As $Y \in \mathcal{A}_X$, we have that c_{yx_1} and c_{yx_2} are independence copulas. Therefore, by setting these terms to 1, we get $c_{x_1x_2|y} = c_{x_1x_2}c_{yx_2|x_1}$. Substituting in the relation above, the joint distribution can be expressed as:

$$p(y,w,x_1,x_2) = p(y)p(w)p(x_1)p(x_2)c_{x_1x_2|y}c_{x_2w}c_{x_1w|x_2}.$$

Now, note that lemma 4.3.1 can also be applied to the joint distribution (W, X_1, X_2), which gives:

$$c(u_1,u_2,u_w) = c_{x_1x_2}c_{x_2w}c_{x_1w|x_2}.$$

Therefore, we get that $c_{x_2w}c_{x_1w|x_2} = \frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}$, so the distribution can be written as:

$$p(y,w,x_1,x_2) = p(y)p(w)p(x_1)p(x_2)c_{x_1x_2|y}(u_{1|y},u_{2|y})\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}.$$

In the equality above, $u_{1|y} = u_1$ and $u_{2|y} = u_2$, due to the independence constraints. Also, $c_{x_1x_2|y}$ is copula density of (X_1, X_2) given Y, and is given through a parametrisation $\theta(u_y)$, and $\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}$ is a quantity which can be computed from the initial data, as the distribution of (W, X_1, X_2) is known. □

In the previous proof, we have assumed that divisions with quantities involving copulas are always possible.

Similar to the self-disclosure case, this decomposition shows that the private disclosure channel is uniquely recovered by the marginal distribution $p(y)$ and the copula $c_{12|y}$, giving the following parametrisation lemma:

Lemma 4.4.2. *The copula density $c_{12|y}$, together with $p(y)$, parametrise the feasible set $\mathcal{A}_X$.*

Proof. Consider $Y \in \mathcal{A}_X$. Due to the Markov condition $W-X-Y$ and the independence conditions $Y \perp\!\!\!\perp X_1$, $Y \perp\!\!\!\perp X_2$, the joint pdf of (Y, W, X_1, X_2) is given by lemma 4.4.1:

$$p(y,w,x_1,x_2) = p(y)p(w)p(x_1)p(x_2)c_{12|y}(u_1,u_2)\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}.$$

As the marginal probability distribution functions $p(w), p(x_1), p(x_2)$ and the copula densities $c(u_1,u_2,u_w), c(u_1,u_2)$ are known, the marginal $p(y)$ and the conditional copula $c_{12|y}$ uniquely determine the joint distribution, parametrising the space of private disclosure channels. Conversely, given $p(y)$ and copula density $c_{12|y}$, a private disclosure channel can be uniquely constructed based on the parametrisation above. Note that this parametrisation ensures constraints are satisfied by default, through the properties of the pair-copula decomposition. □

Proposition 4.4.1. *The disclosure of a system with sources X_1, X_2 and target W can be expressed as:*

$$I_s = \max_{\theta(\cdot)} \int_{[0,1]^2} du_w du_y \int_{[0,1]^2} du_1 du_2 \, c_{12|\theta(u_y)}(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \cdot \log \left(\int_{[0,1]^2} du_1 du_2 \, c_{12|\theta(u_y)}(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \right).$$

Proof. Since copulas have uniform marginals, we get the following:

$$c(u_w, u_y) = \int_{[0,1]^2} c(u_w, u_y, u_1, u_2) du_1 du_2.$$

Substituting this in the expression for synergistic disclosure, Eq. (4.8) gives:

$$I(W;Y) = \int_{[0,1]^2} \left(\int_{[0,1]^2} c(u_w, u_y, u_1, u_2) du_1 du_2 \right) \log \left(\int_{[0,1]^2} c(u_w, u_y, u_1, u_2) du_1 du_2 \right) du_w du_y.$$

By lemma 4.4.1, for any $Y \in \mathcal{A}_X$, the following holds:

$$c(u_w, u_y, u_1, u_2) = c_{12|y}(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)}, \forall u_w, u_y, u_1, u_2 \in [0, 1].$$

By lemma 4.4.2, the space is parametrised by the conditional copula $c_{12|y}$ and $p(y)$. As $p(y)$ does not appear in the expression of $I(W;Y)$, the value of the disclosure associated with a private channel only depends on $c_{12|y}$. Combining the relations above gives:

$$I(W;Y) = \int_{[0,1]^2} du_w du_y \int_{[0,1]^2} du_1 du_2 \, c_{12|y}(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \cdot \log \left(\int_{[0,1]^2} du_1 du_2 \, c_{12|y}(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \right).$$

Now, maximising over the set of copulas $c_{12|y}$ (which is equivalent to maximising over the parametrisations $\theta(\cdot)$) gives the expression needed. □

We assume again that the conditional copula $c_{12|y}(u_{1|y}, u_{2|y})$ only depends on Y through the arguments, and not through the intrinsic shape of the copula at particular points. This assumption can also be replaced with an assumption on the continuity of the integrand in the previous expression. Therefore, similarly to the self-disclosure case, the optimal disclosure becomes:

$$I_s = \int_{[0,1]} du_w \int_{[0,1]^2} du_1 du_2 c_{12|\theta^*}(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \cdot \log \left(\int_{[0,1]^2} du_1 du_2 c_{12|\theta^*}(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \right). \tag{4.9}$$

Expression Eq. (4.9) highlights that in order to find the optimal synergy, it is enough to find θ^*. The synergy I_s depends on the private disclosure channel solely through the conditional copula $c_{12|\theta(u_y)}$, and this copula can be used to obtain $c(u_1, u_2, u_y)$, from which the joint distribution can be constructed.

Proposition 4.4.2. *The function $S : \mathcal{C} \to [0, \infty]$, defined on the set of bivariate copulas, with*

$$S(v) = \int_{[0,1]} du_w \int_{[0,1]^2} du_1 du_2 v(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \cdot \log \left(\int_{[0,1]^2} du_1 du_2 v(u_1, u_2) \frac{c(u_1, u_2, u_w)}{c(u_1, u_2)} \right)$$

is convex, the supremum is achieved and lies on the boundary of the set of copulas.

Proof. Take $V_1, V_2 \in \mathcal{C}$ arbitrary bivariate copulas with v_1, v_2 the respective copula densities. The set of copulas is convex, so for any $\lambda \in [0,1]$, $V = \lambda V_1 + (1-\lambda)V_2$ is a bivariate copula, with density $v = \lambda v_1 + (1-\lambda)v_2$. By linearity, we have

$$v(u_1,u_2)\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)} = \lambda v_1(u_1,u_2)\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)} + (1-\lambda)v_2(u_1,u_2)\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}.$$

Let $J(v,u_w) = \int_{[0,1]^2} du_1 du_2 v(u_1,u_2)\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}$. By integrating the equality above in $[0,1]^2$, we get:

$$J(v,u_w) = \lambda J(v_1,u_w) + (1-\lambda)J(v_2,u_w).$$

The function $x \mapsto x\log x$ is convex, so by Jensen's inequality, for any $u_w \in [0,1]$ for which $J(v,u_w), J(v_1,u_w), J(v_2,u_w) < \infty$, we get:

$$\begin{aligned} J(v,u_w)\log J(v,u_w) &= (\lambda J(v_1,u_w) + (1-\lambda)J(v_2,u_w))\log\left(\lambda J(v_1,u_w) + (1-\lambda)J(v_2,u_w)\right) \\ &\leq \lambda J(v_1,u_w)\log J(v_1,u_w) + (1-\lambda)J(v_2,u_w)\log J(v_2,u_w). \end{aligned}$$

Also, note that if the $J(v,u_w) = \infty$, the right hand side of the inequality is also ∞, so the inequality holds $\forall u_w \in [0,1]$. By integrating u_w in $[0,1]$, we get:

$$S(v) \leq \lambda S(v_1) + (1-\lambda)S(v_2).$$

Therefore, S is a convex function defined on a convex, compact set. Similar to the self-disclosure case, this means that the optimal value is achieved for V a copula on the boundary of the copula set. □

4.4.2 Approximation with checkerboard copulas

By the previous proposition, we have that the maximal value is achieved on the boundary of the copula set. To approximate the optimal disclosure, we will now consider a discretised version of the problem, and we will use a similar strategy as in the case of self-disclosure.

Let V_n a checkerboard copula defined on a grid of size n for some $n \in \mathbb{N}$, with copula density $v_{ij}, \forall i,j \in [1:n]$. The synergy associated to V_n is

$$S(V_n) = \frac{1}{n^3}\sum_{w=1}^{n}\left(\sum_{i,j=1}^{n} v_{ij}c_{ijw}\right)\log\left(\frac{1}{n^2}\sum_{i,j=1}^{n} v_{ij}c_{ijw}\right).$$

This can be shown by direct computation and using the fact that the copula density of V_n is constant on squares of area $\frac{1}{n^2}$. For fine enough grids, we can consider the copula of (X_1, X_2, W) to be constant in cubes of the form $[\frac{w-1}{n}, \frac{w}{n}] \times [\frac{i-1}{n}, \frac{i}{n}] \times [\frac{j-1}{n}, \frac{j}{n}]$. We define $(c_{ijw})_{i,j,w=0}^{n}$ to represent the constant coefficients the copula values determine on cubes $[\frac{w-1}{n}, \frac{w}{n}] \times [\frac{i-1}{n}, \frac{i}{n}] \times [\frac{j-1}{n}, \frac{j}{n}]$.

For $u_w \in [\frac{w-1}{n}, \frac{w}{n}], w \in [1:n]$, we set $\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)} = c_{ijw}$, and get:

$$\begin{aligned} \int_{[0,1]^2} du_1 du_2 v(u_1,u_2)\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)} &= \sum_{i,j=1}^{n}\int_{[\frac{i-1}{n},\frac{i}{n}]\times[\frac{j-1}{n},\frac{j}{n}]} v_{ij}\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}du_1du_2 \\ &= \sum_{i,j=1}^{n} v_{ij}\int_{[\frac{i-1}{n},\frac{i}{n}]\times[\frac{j-1}{n},\frac{j}{n}]}\frac{c(u_1,u_2,u_w)}{c(u_1,u_2)}du_1du_2 \\ &= \frac{1}{n^2}\sum_{i,j=1}^{n} v_{ij}c_{ijw}. \end{aligned}$$

The relation above can be used in the expression of $S(V_n)$ to get:

$$
\begin{aligned}
S(V_n) &= \int_{[0,1]} du_w \int_{[0,1]^2} du_1 du_2 v(u_1,u_2) \frac{c(u_1,u_2,u_w)}{c(u_1,u_2)} \log\left(\int_{[0,1]^2} du_1 du_2 v(u_1,u_2) \frac{c(u_1,u_2,u_w)}{c(u_1,u_2)} \right) \\
&= \sum_{w=1}^{n} \int_{[\frac{w-1}{n}, \frac{w}{n}]} du_w \left(\frac{1}{n^2} \sum_{i,j=1}^{n} v_{ij} c_{ijw} \right) \log\left(\frac{1}{n^2} \sum_{i,j=1}^{n} v_{ij} c_{ijw} \right) \\
&= \frac{1}{n^3} \sum_{w=1}^{n} \left(\sum_{i,j=1}^{n} v_{ij} c_{ijw} \right) \log\left(\frac{1}{n^2} \sum_{i,j=1}^{n} v_{ij} c_{ijw} \right).
\end{aligned}
$$

Similar to the case of self-disclosure, $S(V_n)$ is a convex function defined on the convex set $\mathcal{DC}_n$. Hence, the optimal in the case of general synergy on discrete copulas of size n is also achieved for an extreme point in this polytope.

4.4.3 Associated optimisation problem

Similar to the self-disclosure case, the optimisation of S for discrete copulas can be seen as an optimisation problem. The objective function is now the discrete representation of general disclosure from previous section, and the initial coefficients $(c_{ijw})_{ijw=1}^{n}$ represent the ratio of the copulas $c(u_1,u_2,u_w)$ and $c(u_1,u_2)$. The optimisation problem can therefore be formulated as:

$$
\begin{aligned}
&\text{maximise } \frac{1}{n^3} \sum_{w=1}^{n} \left(\sum_{i,j=1}^{n} v_{ij} c_{ijw} \right) \log\left(\frac{1}{n^2} \sum_{i,j=1}^{n} v_{ij} c_{ijw} \right) \\
&\text{subject to } \forall i,j \in [1:n]\ v_{ij} \geq 0, \\
&\forall i \in [1:n]\ \frac{1}{n} \sum_{j=1}^{n} v_{ij} = 1,\ \forall j \in [1:n]\ \frac{1}{n} \sum_{i=1}^{n} v_{ij} = 1.
\end{aligned}
\tag{4.10}
$$

4.4.4 Synergy in systems with arbitrary number of sources

The results from the previous section can be generalised in a straightforward way to systems of an arbitrary number of sources. Similarly to the case of two sources, a rephrasing of the problem using $d-$dimensional copulas is considered, and an associated discrete optimisation problem is built for the problem on the continuous domain.

In this section, a system of d sources $\mathbf{X} = \{X_1, X_2, \ldots, X_d\}$ is used.

Lemma 4.4.3. *For any private disclosure channel Y ($p_{Y|X} \in \mathcal{A}_X$, $W - X - Y$ Markov chain), the joint probability distribution $p(y,w,x_1,\ldots,x_d)$ can be written using pair-copula constructions as:*

$$
p(y,w,x_1,\ldots,x_d) = p(y)p(w) \prod_{i=1}^{d} p(x_i) c_{12\ldots d|\theta(u_y)}(u_1,\ldots,u_d) \frac{c(u_1,\ldots u_d,u_w)}{c(u_1,\ldots,u_d)},
$$

where $c_{12\ldots d|\theta(u_y)}$ represents the conditional copula of $(X_1,\ldots,X_d)$ given Y.

Proof. Take $Y \in \mathcal{A}_X$, arbitrarily chosen. By applying Eq. (4.3) for the distribution $(Y, X_1, \ldots, X_n, W)$, the following decomposition can be obtained:

$$
p(y \mid w, x_1, \ldots, x_d) = c_{yw|1,\ldots,d} \cdot p(y \mid x_1, \ldots, x_d). \tag{4.11}
$$

From lemma 4.3.3:

$$
p(y,x_1,\ldots,x_d) = p(y) \prod_{i=1}^{d} p(x_i) c_{12\ldots d|\theta(u_y)}(u_1,\ldots,u_d).
$$

Combining the relation above with Sklar's Theorem we get:

$$\begin{aligned} p(y \mid x_1, \ldots, x_d) &= \frac{p(y, x_1, \ldots, x_d)}{p(x_1, \ldots, x_d)} \\ &= \frac{p(y) \prod_{i=1}^{d} p(x_i) c_{12\ldots d|\theta(u_y)}(u_1, \ldots, u_d)}{\prod_{i=1}^{d} p(x_i) c_{12\ldots d}(u_1, \ldots, u_d)} \\ &= \frac{p(y) c_{12\ldots d|\theta(u_y)}(u_1, \ldots, u_d)}{c_{12\ldots d}(u_1, \ldots, u_d)}. \end{aligned}$$

Using this in Eq. (4.11) gives:

$$\begin{aligned} p(y, w, x_1, \ldots, x_d) &= p(y \mid w, x_1, \ldots, x_d) p(w, x_1, \ldots, x_d) \\ &= c_{yw|1,\ldots,d} \frac{p(y) c_{12\ldots d|\theta(u_y)}(u_1, \ldots, u_d)}{c_{12\ldots d}(u_1, \ldots, u_d)} p(w, x_1, \ldots, x_d) \\ &= c_{yw|1,\ldots,d} \frac{p(y) c_{12\ldots d|\theta(u_y)}(u_1, \ldots, u_d)}{c_{12\ldots d}(u_1, \ldots, u_d)} p(w) \prod_{i=1}^{d} p(x_i) c(u_1, \ldots u_d, u_w) \\ &= 1 \cdot p(y) p(w) \prod_{i=1}^{d} p(x_i) \frac{c_{12\ldots d|\theta(u_y)}(u_1, \ldots, u_d)}{c_{12\ldots d}(u_1, \ldots, u_d)} c_{12\ldots dw}(u_1, \ldots, u_d, u_w) \\ &= p(y) p(w) \prod_{i=1}^{d} p(x_i) c_{12\ldots d|\theta(u_y)}(u_1, \ldots, u_d) \frac{c(u_1, \ldots u_d, u_w)}{c(u_1, \ldots, u_d)}. \end{aligned}$$

In the derivation above, the fact that $W - X - Y$ is a Markov chain is also used, through $c_{yw|1,\ldots,d} = 1$ (which holds by the independence property). □

Using an analogous derivation to the case of two sources, the mutual information $I(W;Y)$ depends on Y only through the copula $c_{12\ldots n|\theta(u_y)}$, and it can be shown that the optimal parametrisation function $\theta(\cdot)$ is the constant function $\theta(u_y) = \theta^*$, where θ^* is similarly defined. The disclosure capacity I_s then becomes:

$$\begin{aligned} I_s = \int_{[0,1]} du_w \int_{[0,1]^d} du_1 \ldots du_d c_{12\ldots d|\theta^*}(u_1, \ldots, u_d) \frac{c(u_1, \ldots, u_d, u_w)}{c(u_1, \ldots, u_d)} \\ \cdot \log \left(\int_{[0,1]^d} du_1 \ldots du_d c_{12\ldots n|\theta^*}(u_1, \ldots, u_d) \frac{c(u_1, \ldots, u_d, u_w)}{c(u_1, \ldots, u_d)} \right). \end{aligned}$$

By considering a $d-$dimensional discrete copula V_n on a grid of size n, the problem can be reduced to finding the optimal value of:

$$S(V_n) = \frac{1}{n^{d+1}} \sum_{w=1}^{n} \left(\sum_{i_1, \ldots i_d = 1}^{n} v_{i_1 \ldots i_d} c_{i_1 \ldots i_d w} \right) \log \left(\frac{1}{n^d} \sum_{i_1, \ldots i_d = 1}^{n} v_{i_1 \ldots i_d} c_{i_1 \ldots i_d w} \right).$$

The DCCP optimisation problem has n^d variables, and can be formulated as:

$$\begin{aligned} &\text{maximise} \quad \frac{1}{n^{d+1}} \sum_{w=1}^{n} \left(\sum_{i_1, \ldots i_d = 1}^{n} v_{i_1 \ldots i_d} c_{i_1 \ldots i_d w} \right) \log \left(\frac{1}{n^d} \sum_{i_1, \ldots i_d = 1}^{n} v_{i_1 \ldots i_d} c_{i_1 \ldots i_d w} \right) \\ &\text{subject to} \quad \forall i_1 \ldots i_d \in [1:n] \; v_{i_1 \ldots i_d} \geq 0, \\ &\qquad \forall k \in [1:d], i \in [1:n] \; \frac{1}{n} \sum_{i_1, \ldots i_{k-1}, i_{k+1}, \ldots i_d = 1}^{n} v_{i_1, \ldots i_{k-1}, i, i_{k+1}, \ldots i_d} = 1. \end{aligned} \tag{4.12}$$

The challenges arise, however, due to the increased complexity of the computation. In the case of two sources, the discrete optimisation happens over a two-dimensional grid of varying resolution. However, the size of the grid increases quickly with the number of sources in the system, as for d sources, the associated optimisation problem contains an $d-$dimensional conditional copula, for which the domain is the unit hypercube in $d-$dimensions.

4.4.5 Convergence to the true optimal value

In this section, we present a final result which ensures the checkerboard copula approach gives a useful approximation. We show the sequence of approximations given by the the optimisation problem (4.12) convergences to the optimal disclosure I_s.

Theorem 4.4.1. *There exists a sequence of solutions to the optimisation problem* (4.12) *that converge to the maximal disclosure* I_s.

Proof. By proposition 4.4.2, the maximum value of $S(\cdot)$, corresponding to I_s, is achieved for some copula C^*. The copula C^* determines a random variable Y^* for which $I(Y^*;W) = I_s$ (by setting C^* to be the conditional copula $C_{12|y}$).

By theorem 4.1.7, there exists a sequence of increasingly finer checkerboard copulas $(\hat{C}_k)_{k=1}^{\infty}$ that converge to C^* uniformly. Let $(\hat{Y}_k)_{k=1}^{\infty}$ be a sequence of random variables, such that $\hat{Y}_k$ is determined by $\hat{C}_k, \forall k \in \mathbb{N}$. Therefore, by the definition of mutual information and the dominated convergence theorem, we get that:

$$I(\hat{Y}_k; W) \to I(Y^*; W) = I_s. \tag{4.13}$$

Now, let $(n_k)_{k=1}^{\infty}$ be an unbounded, increasing sequence of positive integers, such that $\hat{C}_k$ is a checkerboard copula defined on a grid of size $n_k, \forall k \in \mathbb{N}$. Let $(V_k)_{k=1}^{\infty}$ be a sequence of solutions to the optimisation problem (4.12). By considering the structure of the optimisation problem, for each k, $V_k \in \mathcal{DC}_{n_k}$ and V_k is an extreme checkerboard copula. Let $(Y_k)_{k=1}^{\infty}$ be the sequence of random variables determined by this sequence of checkerboard copulas. By definition, note that for all $k, I(Y_k; W) = S(V_k)$.

Since, V_k is the solution to the optimisation problem on the polytope $\mathcal{DC}_{n_k}$ and $\hat{C}_k \in \mathcal{DC}_{n_k}$, we get $S(V_k) \geq S(\hat{C}_k), \forall k \in \mathbb{N}$. Hence, considering the corresponding random variables determined by these copulas, we get:

$$\forall k \in \mathbb{N}, I(Y_k; W) \geq I(\hat{Y}_k; W). \tag{4.14}$$

Now, by Eq. (4.13) and taking the limit as $k \to \infty$ in Eq. (4.14), we get that $\lim_{k\to\infty} I(Y_k; W) \geq I_s$. Considering the maximality of I_s over the set of copulas, and that V_k is a valid copula distribution for all values of k, we also get $\forall k \in \mathbb{N}, I(Y_k; W) \leq I_s$. Combining these inequalities, we get $\lim_{k\to\infty} I(Y_k; W) = I_s$, so the sequence of solutions converges to the global optimum I_s. □

Remark 4.4.1. *Note that, unlike the self-disclosure which is infinite, by the Data Processing Inequality 2.1.1, the synergy with an arbitrary target is bounded by the mutual information* $I(\boldsymbol{X}; W)$ *(as* $W - \boldsymbol{X} - Y$ *is a Markov chain). This shows that* I_s *is finite when* $I(\boldsymbol{X}; W)$ *is finite.*

4.4.6 Numerical Simulations

As an example for the results in the previous section, we consider the case of two sources. Phrasing the optimisation problem as above makes the scenario suitable for `CVXPY` and DCCP. The problem can then be solved using a commercial solver such as GUROBI, and we can analyse numerically the optimal values. In this section, we have chosen (X_1, X_2, W) to follow a multivariate normal distribution, which is characterised by $\rho_{12}, \rho_{13}, \rho_{23}$, the pairwise correlations between the components. The system can be visualised in the figure below:

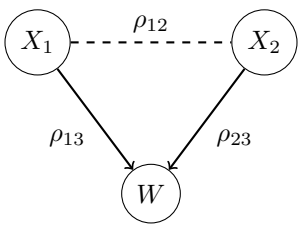


Figure 4.8: The system (X_1, X_2, W) follows a multivariate normal distribution. The sources X_1, X_2 are connected through correlation ρ_{12}, and they are linked to the target W through correlations ρ_{13} and ρ_{23}, respectively.

Impact of grid size

In this first example, we want to see the influence of finer grids on the optimal solution. Based on approximation theorem 4.1.7, we expect that the values converge as we consider finer grids. Indeed, we notice that initially the synergy increases quickly, and as n increases the values seem to converge. Synergy has largest values for independent sources, as in this case there is no overlap in the information provided by the sources. We also note that in the case where sources are not well correlated with the target, but highly correlated between them, the synergy is close to 0.

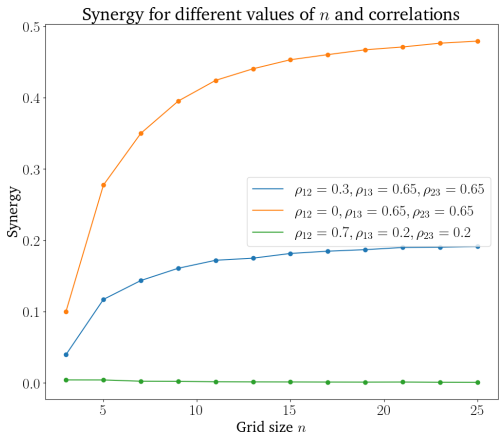


Figure 4.9: Influence of the grid size on the optimal disclosure estimated. As grid size increases, the estimated values of synergy start to converge.

Independent sources

We consider independent random variables X_1 and X_2 ($\rho_{12} = 0$). We will vary the correlation each of these have with the target and see how this affects synergy values.

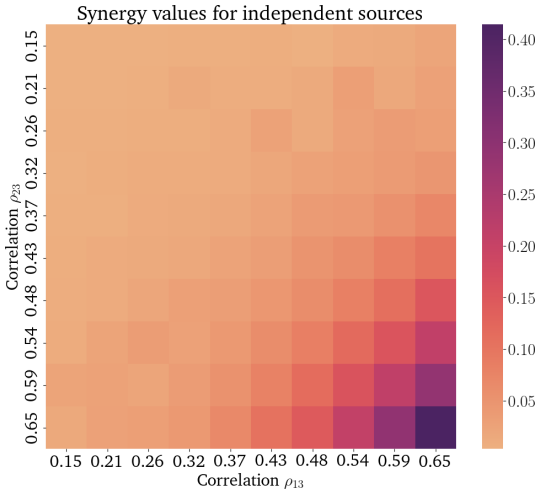


Figure 4.10: Synergy with independent sources. Sources that are better correlated with the target become better predictors, so synergy increases.

In the diagram presented, the correlation between sources and the target is in the range [0.15, 0.65]. The results are consistent with the intuition built for the notion of synergy, and the results from the numerical simulations in the discrete case. When the sources have a strong correlation with the target, the level of disclosure tends to increase. This is because Y is derived from X_1 and X_2, and as these sources correlate more closely with the target, they can more effectively predict W. Conversely, when the sources are not correlated with the target, the synergy is very close to 0.

Varying correlation between sources.

We now consider fixed correlations $\rho_{13} = \rho_{23}$, and vary the correlation ρ_{12} in the interval [0, 0.9]. As expected, the synergy contained in the system seems to decrease as the correlation increases, as there is less information in the system that can be used to predict the target. The synergy has the highest value when the sources are independent, and when the correlation is close to 1, the synergy in the system is close to 0.

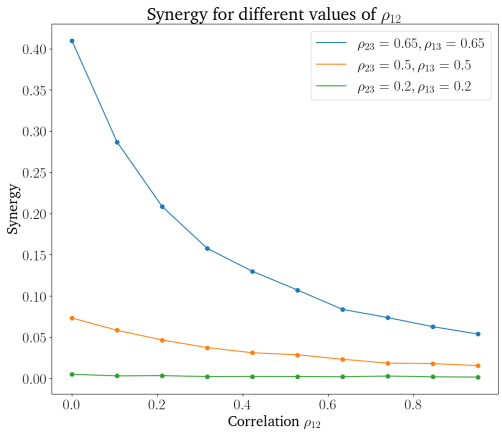


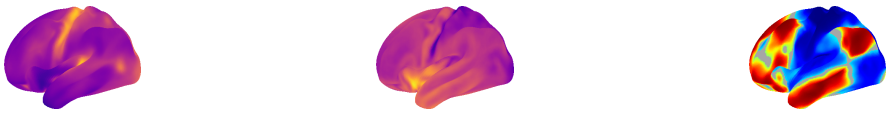
Figure 4.11: Varying correlation between the sources. As sources become more correlated, the synergy in the systems decreases, and the system contains a lot of redundant information.

4.4.7 Applications to real-world data

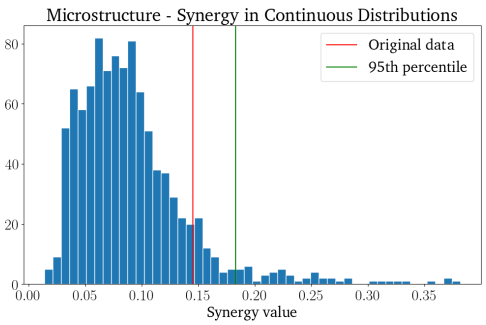
In this section, we apply the synergy measure to continuous distributions in real-world datasets. We will again use data sourced from the neuromaps project [22], and we consider an identical experiment setup.

The previous sections derived an expression for the synergy in continuous distributions and showed that checkerboard copulas are useful approximations to consider in this case. Numerical simulations also confirmed the intuition built for this notion. Building on these, an example applying synergy to continuous, real-world data is presented in this section. The experiment setup is identical to the experiments on discrete datasets.

Sources **Target**



(a) T1w/T2w ratio (b) Cortical Thickness (c) Functional Gradient



(d) Synergy measured on the continuous distribution.

Figure 4.12: Experiment results with Microstructure annotation maps. Results are not significant and we fail to reject the null hypothesis.

One challenge noted in earlier experiments with the neuromaps datasets is the necessity to discretise the data with multiple bins. In this analysis, we work with the original, continuous datasets to estimate the copulas. These estimations were conducted using the `pyvinecopulib` Python library [38]. This package considers the decomposition of the three-dimensional copula into pair copulas, and estimates each pair copula. As the neuromaps package provides many datapoints for these annotation maps (32000 datapoints), the estimation is difficult and the package cannot use any standard bivariate copula families in the process (the copulas are therefore obtained using a form of kernel density estimation).

After the copula density is estimated, we use Monte-Carlo integration [39] to approximate the values of the coefficients in the optimisation problem (4.10). The problem is then solved through DCCP optimisation.

In the figure above, the experiment results of 1000 spin tests are shown. Unlike the discrete case, the results are not significant, and we fail to reject the null hypothesis.

This variation in results could also be attributed to several factors related to our methodology. The process of fitting a copula without additional information about the shape of the distribution may lack precision. Also, we consider the copula to be constant on unit squares in the grid, and the reliability of these estimations is heavily dependent on the resolution of the grids used in the optimisation problem. A finer grid is desired, but also requires more computational resources and can introduce its own set of challenges regarding DCCP optimisation.

The above experiment demonstrates how the synergy measure can be applied to analyse data derived from continuous distributions. While the limited scalability of the estimation algorithms might obscure the true statistical outcomes, this example nonetheless illustrates a valuable application of the synergy concept, and shows that the theory derived in this chapter can have meaningful applications in real-world datasets.

Chapter 5

Conclusion and Future Work

Throughout this project, we have explored the notion of synergy in the context of the Partial Information Decomposition framework and approached the problem of measuring synergy from both a computational and theoretical point of view. We have improved the scalability and efficiency of synergy computation software in the case of discrete probability distributions, and we have developed a natural extension of the notion of synergy from discrete probability distributions to continuous, real-valued distributions.

5.1 Key Findings and Contributions

We briefly revisit the main contributions of the project:

- **Scalable and efficient** `dccp-syndisc`[1]: We have developed a novel extension of the synergy calculation software, `syndisc`. Through the use of disciplined convex-concave programming (DCCP), this software more than doubles the previous limits in synergy calculations, supporting systems of up to 13 discrete random variable sources, compared to the previous limit of 5 sources. After extensive evaluations, we conclude that the method consistently achieves high efficiency in synergy estimation in a variety of systems.
- **Real-world applications**: By addressing the computational challenges associated with previous methods and extending the application scope to real-world datasets, `dccp-syndisc` offers a valuable tool for understanding the complexities of system interactions. We presented a series of initial results that show the potential impact of the optimised software in real-world settings. These results also offer a new perspective for analysing dependencies in complex systems, suggesting that an analysis of the collective behaviors for different components can provide new information.
- **Generalization to continuous distributions**: We successfully extended the concept of synergy to continuous distributions by leveraging the properties of copulas. This generalization allows us to capture the interdependencies between continuous random variables. The introduction of checkerboard copulas and the formulation of the associated optimization problems are particularly important, as they provide practical tools for estimating synergy in complex systems. Finally, the convergence of the checkerboard copulas towards the true optimal disclosure validates our approach, illustrating the robustness of the estimations.

5.2 Outlook and Future work

The positive results in this project have several implications for future research in information theory and data analysis. Potential improvements can be viewed from two perspectives: theoretical advancements based on the findings of this project, and practical applications of synergy measures in real-world datasets.

From a theoretical point of view, the results presented face two main limitations:

[1] Implementation available at https://github.com/vladcoroian/dccp-syndisc.

- **Theoretical guarantees of** `dccp-syndisc`: As with any heuristic method, there is a risk of convergence to local optima rather than the global optimum. This issue highlights the need for developing strategies to ensure enough sampling guarantees a certain level of performance or at least ensure that the local solutions are sufficiently close to the global optimum. The performance of `dccp-syndisc` is sensitive to the choice of parameters such as the size of the alphabet $\mathcal{Y}$ and the number of iterations. While we have provided guidelines for choosing these parameters, further research could automate these choices or make the algorithm less sensitive to them.

- **Interpretability in continuous distributions**: The checkerboard copula approximation provides valuable estimates of synergy in continuous systems. Additionally, the convexity of the synergy function shows that a maximally synergistic variable exists. However, a theoretical link to construct the optimal disclosure channel, based on the approximations remains unestablished. The checkerboard copula approximations do not necessarily form a convergent sequence in the set of copulas. Therefore, there is currently no systematic approach to derive the optimal disclosure channel in the case continuous distributions.

- **Other methods for parametrisation of copulas**: While checkerboard copulas offer useful approximations when grids are sufficiently fine, the computational complexity of this approach increases quickly with the number of sources and the granularity of the grid. Other types of copula parametrisations, such as using neural networks to capture the dependency structure [40, 41], could provide flexible and potentially more powerful methods of parametrisation, which might overcome the limitations of the current grid-based approach.

The methods developed in this project and the initial results obtained by applying this measure to real-world data are promising. Further work can focus on applying the theoretical insights and companion software developed to practical scenarios:

- **Broader applications to real-world data**: The application of synergy measurements can extend beyond the realms of neuroscience and genetics. Information theory and the PID framework in particular have applications in a diverse range of fields. This motivates the potential use of synergy computation software in various collections of real-world data, where understanding complex interactions can provide useful insights.

- **Scalability of synergy in continuous distributions**: One of the limitations in the estimation of synergy in continuous distributions is the scalability issues with the number of sources. The number of variables in the associated optimisation problem increases exponentially with the number of sources. This makes estimations very challenging for large systems. To make synergy computation practical for real-world applications, the computation software must be optimized to efficiently handle larger systems.

5.3 Concluding remarks

In this project, we focused on synergy, a concept that beautifully explains the complex interactions of large systems. Through synergy, we can explain how information emerges from these systems collectively, offering deeper insights into their complexity.

The results in this project are both theoretical and practical, and lay a solid foundation for future research in this field. The work on synergistic relations within continuous distributions has significantly sharpened our understanding of this nuanced concept. On the other hand, the computational improvements to the algorithms measuring synergy in discrete distributions demonstrate the remarkable potential of synergy in data analysis.

Looking ahead, these developments promise to broaden the use of synergy measures. The key results in this project pave the way for applications that could offer new perspectives on the analysis of complex interactions.

Chapter 6

Ethical considerations

We believe that the research in this project is exempt from ethical approval, as the analysis is based on secondary data, which is already freely accessible online. Also, no additional personal data is collected as part of the project.

The utilisation of real-world datasets happens only in the evaluation phase of the project, where the data is sourced from public repositories like the Enigma Toolbox and the Neuromaps project. Given the nature of the data and the methodology employed, we believe that our research does not need additional ethical analysis or approval.

The datasets utilised in our research have been publicly released and are readily available, eliminating the need for explicit consent from individual participants involved in the study. Furthermore, the data has been anonymised to a degree where there is no risk of identification. Also, the collection of the original data adhered to the ethical standards outlined by the respective projects.

It is important to emphasize that our research does not involve direct interaction with human subjects, and our analysis is centered around anonymised data. As we solely rely on secondary data that has already undergone ethical considerations by the original data collectors, our work is within established ethical boundaries.

Bibliography

[1] Mackay DJC. Information theory, inference, and learning algorithms. *Cambridge, England: Cambridge University Press*, 2003.

[2] Watanabe S. Information theoretical analysis of multivariate correlation. *IBM Journal of Research and Development*, 4(1):66–82, 1960. doi: 10.1147/rd.41.0066.

[3] McGill W. J. Multivariate information transmission. *Psychometrika*, 19(2):97–116, 1954. ISSN 1860-0980. doi: 10.1007/BF02289159. URL https://doi.org/10.1007/BF02289159.

[4] Williams P. L. and Beer R. D. Nonnegative decomposition of multivariate information. *arXiv:1004.2515*, abs/1004.2515, 2010. URL https://api.semanticscholar.org/CorpusID:14794416.

[5] Luppi A. I., Mediano P. A. M., and Rosas F. E. et. al. A synergistic core for human brain evolution and cognition. *Nature Neuroscience*, 25(6):771–782, 2022. doi: 10.1038/s41593-022-01070-0.

[6] Proca A. M., Rosas F. E., Luppi A. I., Bor D., Crosby M., and Mediano P. A. M. Synergistic information supports modality integration and flexible learning in neural networks solving multiple tasks. *PLoS Computational Biology*, 20(6):e1012178, 2024. doi: 10.1371/journal.pcbi.1012178. Advance online publication.

[7] Rauh J., Kr. Banerjee P., Olbrich E., and Jost J. Unique information and secret key decompositions. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 3042–3046, 2019. doi: 10.1109/ISIT.2019.8849550.

[8] Rosas F. E., Mediano P. A. M., Rassouli B., and Barrett A. An operational information decomposition via synergistic disclosure. *Journal of Physics A: Mathematical and Theoretical*, 14(3):53, 2020.

[9] Griffith V. and Koch C. *Quantifying synergistic mutual information*, pages 159–190. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. ISBN 978-3-642-53734-9. doi: 10.1007/978-3-642-53734-9_6. URL https://doi.org/10.1007/978-3-642-53734-9_6.

[10] Bertschinger N., Rauh J., Olbrich E., Jost J., and Ay N. Quantifying unique information. *Entropy*, 16(4):2161–2183, April 2014. ISSN 1099-4300.

[11] Niu X. and Quinn C. A measure of synergy, redundancy, and unique information using information geometry. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 3127–3131, 2019. doi: 10.1109/ISIT.2019.8849724.

[12] Rassouli B., Rosas F., and Gündüz D. Latent feature disclosure under perfect sample privacy. *IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–7, 2018.

[13] Rassouli B., Rosas F. E., and Gündüz D. Data disclosure under perfect sample privacy. *IEEE Transactions on Information Forensics and Security*, 15:2012–2025, 2020. doi: 10.1109/TIFS.2019.2954652.

[14] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. URL https://www.gurobi.com.

[15] Boyd S. and Vandenberghe L. *Convex optimization*. Cambridge university press, 2004.

[16] Grant M., Boyd S., and Ye Y. Disciplined convex programming. In Liberti L. and Maculan N., editors, *Global Optimization: From Theory to Implementation*, pages 155–210. Springer US, Boston, MA, 2006. ISBN 978-0-387-30528-8.

[17] Diamond S. and Boyd S. Cvxpy: A python-embedded modeling language for convex optimization. *J. Mach. Learn. Res.*, 17(1):2909–2913, jan 2016. ISSN 1532-4435.

[18] Lipp T. and Boyd S. P. Variations and extension of the convex–concave procedure. *Optimization and Engineering*, 17:263–287, 2016. URL https://api.semanticscholar.org/CorpusID:14778227.

[19] Shen X., Diamond S., Gu Y., and Boyd S. Disciplined convex-concave programming, 2016. URL https://api.semanticscholar.org/CorpusID:7815963.

[20] Wright S. J. Coordinate descent algorithms. *Mathematical Programming*, 151(1): 3–34, 6 2015. doi: 10.1007/s10107-015-0892-3. URL https://doi.org/10.1007/s10107-015-0892-3.

[21] Alexander-Bloch A., Shou H., and Liu S. et. al. On testing for spatial correspondence between maps of human brain structure and function. *NeuroImage*, 178, 06 2018. doi: 10.1016/j.neuroimage.2018.05.070.

[22] Markello R., Hansen J., and Liu Z. et. al. Neuromaps: structural and functional interpretation of brain maps. *bioRxiv*, 01 2022. doi: 10.1101/2022.01.06.475081.

[23] Glasser M. F., Coalson T. S., Robinson E. C., and Hacker C. D. et. al. A multi-modal parcellation of human cerebral cortex. *Nature*, 536:171 – 178, 2016. URL https://api.semanticscholar.org/CorpusID:205249949.

[24] Vaishnavi S. N., Vlassenko A. G., and Rundle M. M. et. al. Regional aerobic glycolysis in the human brain. *Proceedings of the National Academy of Sciences*, 107(41):17757–17762, 2010. doi: 10.1073/pnas.1010459107. URL https://www.pnas.org/doi/abs/10.1073/pnas.1010459107.

[25] Van Essen D. C., Smith S. M., and Barch D. M. et. al. The wu-minn human connectome project: An overview. *NeuroImage*, 80:62–79, 2013. ISSN 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2013.05.041. URL https://www.sciencedirect.com/science/article/pii/S1053811913005351. Mapping the Connectome.

[26] Lariviere S., Paquola C., and Park B. et. al. The enigma toolbox: multiscale neural contextualization of multisite neuroimaging datasets. *Nature Methods*, 18, 06 2021. doi: 10.1038/s41592-021-01186-4.

[27] Nelsen R. B. *An Introduction to Copulas*. Springer Series in Statistics. Springer New York, NY, 2 edition, 2007. ISBN 978-0-387-28678-5. URL https://doi.org/10.1007/0-387-28678-0.

[28] Czado C. Pair-copula constructions of multivariate copulas. In *Copula Theory and Its Applications*, pages 93–109, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-12465-5.

[29] Kolesarova A., Mesiar R., Mordelova J., and Sempi C. Discrete copulas. *IEEE Transactions on Fuzzy Systems*, 14(5):698–705, 2006. doi: 10.1109/TFUZZ.2006.880003.

[30] Perrone E. and Durante F. Extreme points of polytopes of discrete copulas. In *Joint Proceedings of the 19th World Congress of the International Fuzzy Systems Association (IFSA), the 12th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT), and the 11th International Summer School on Aggregation Operators (AGOP)*, pages 596–601. Atlantis Press, 2021. ISBN 978-94-6239-423-0. doi: 10.2991/asum.k.210827.080. URL https://doi.org/10.2991/asum.k.210827.080.

[31] Ma J. and Sun Z. Mutual information is copula entropy. *Tsinghua Science and Technology*, 16 (1):51–54, 2011. doi: 10.1016/S1007-0214(11)70008-6.

[32] Aas K., Czado C., Frigessi A., and Bakken H. Pair-copula constructions of multiple dependence. *Insurance: Mathematics and Economics*, 44(2):182–198, 2009. ISSN 0167-6687. doi: https://doi.org/10.1016/j.insmatheco.2007.02.001. URL https://www.sciencedirect.com/science/article/pii/S0167668707000194.

[33] Li X., Mikusiński P., and Taylor M.D. Strong approximation of copulas. *Journal of Mathematical Analysis and Applications*, 225(2):608–623, 1998. ISSN 0022-247X. doi: https://doi.org/10.1006/jmaa.1998.6056. URL https://www.sciencedirect.com/science/article/pii/S0022247X98960565.

[34] Bedford T. and Cooke R. M. Vines–a new graphical model for dependent random variables. *The Annals of Statistics*, 30(4):1031 – 1068, 2002. doi: 10.1214/aos/1031689016. URL https://doi.org/10.1214/aos/1031689016.

[35] Pakman A., Nejatbakhsh A., Gilboa D., Makkeh A., and Mazzucato L. et. al. Estimating the unique information of continuous variables. *Advances in neural information processing systems*, 34:20295–20307, 2021. URL https://api.semanticscholar.org/CorpusID:231741184.

[36] Hobæk Haff I., Aas K., and Frigessi A. On the simplified pair-copula construction — simply useful or too simplistic? *Journal of Multivariate Analysis*, 101(5):1296–1310, 2010. ISSN 0047-259X. doi: 10.1016/j.jmva.2009.12.001.

[37] Li X., Mikusiński P., Sherwood H., and Taylor M. D. In quest of Birkhoff's theorem in higher dimensions. *Lecture Notes-Monograph Series*, 28:187–197, 1996. ISSN 07492170. URL http://www.jstor.org/stable/4355892.

[38] Vatter T. and Nagler T. Pyvinecopulib 0.6.4, 2022. URL https://vinecopulib.github.io/pyvinecopulib/. Accessed: 2024-06-14.

[39] Robert C. P. and Casella G. *Monte Carlo Statistical Methods*. Springer, New York, NY, USA, 2 edition, 2004. ISBN 978-1-4419-1939-7.

[40] Zeng Z. and Wang T. Neural copula: A unified framework for estimating generic high-dimensional copula functions. 05 2022. doi: 10.48550/arXiv.2205.15031.

[41] Letizia N. and Tonello A. Copula density neural estimation. 11 2022. doi: 10.48550/arXiv.2211.15353.