Bagged Copula-GPFA: A Framework for Estimating Dynamic Neuronal Dependency Relationships

Maximilian Walden

4th Year Project Report
Computer Science and Mathematics
School of Informatics
University of Edinburgh

2024

Abstract

Historically, the bottlenecks keeping researchers from being able to answer some of the central questions surrounding cognition and behavior have been the difficulty of acquiring meaningful neuronal data. However, advancements in recording techniques have somewhat fulfilled these limitations, shifting the needs within the field to powerful computational and statistical techniques. Enter *Copula-GP*, a recently developed state-of-the-art parametric mutual information estimator which was found to outperform other novel non-parametric methods when utilized on highly dimensional data. We utilized *Copula-GP* together with *Gaussian Process Factor Analysis* (GPFA), a novel dimensionality reduction technique, to investigate the information interaction between neuronal processes within the visual cortex of live mice and pupil dilation, naming the combination of methods *Copula-GPFA*. We found that this combination of methods was an effective means of investigating neuronal dependence, allowing flexibility in analysis and finding results in agreement with prior literature. We additionally extended *Copula-GP* with a bagging framework, allowing for the aggregation of model estimations and allowing for more accurate estimation accuracy and representation of dependency shape. We validated our bagging algorithm on simulated data sampled from known distributions, and utilized bagged *Copula-GPFA* on aforementioned neuronal data to find results in agreement with baseline *Copula-GPFA* but with more stability. We finally proposed several extensions, use cases, and possible research projects utilizing *Copula-GPFA* and *Copula-GP* estimator bagging, leveraging it's flexibility and effectiveness in analysis.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Maximilian Walden*)

Acknowledgements

I'd like to acknowledge friends and family that helped me find the drive and energy for this and other projects, as well as the invaluable advice given by my supervisor, as well as he and his fellows' work that culminated in the core paper.

Table of Contents

Chapter 1

Introduction

1.1 Project Setting, Motivation, and Problem Statement

Not quite as many topics in the sciences have captured peoples' imaginations as the mysteries of human thought, behavior, and learning, a topic which has inspired fields ranging from psychology to philosophy [67, 66]. We now know that human thought occurs within the brain, and understanding the intrinsics of how and where such processes occur has applications in fields such as prosthetics and neurosurgery, not to mention how such findings often find a way to spill into related fields, such as machine learning [31]. *Computational Neuroscience* aims to answer how cognition and behavior are encoded in the brain through the use of machine learning and statistical methods on large amounts of neuronal data, both simulated (*in silicon*) and real (*in vivo*) [49, 36, 69]. Some of the primary questions on this frontier include: How is information encoded in populations of neurons, and how is this information related to behaviors observed [36]?

As advances in recording techniques have allowed for the recording of hundreds to thousands of neurons at once [34, 20], the bottlenecks keeping researchers from answering these central questions in neuroscience have shifted away from data-related issues to the purely computational and statistical [69, 27, 54]. Answering such questions with machine learning comes down to analysis of the intricate highly-dimensional multivariate dependencies (i.e. dependencies and correlations between many random variables, many of which have different distributions) in recorded neuronal and behavioral data, much of which varies across spatial and temporal dimensions [36, 34, 27, 54]. This becomes especially challenging once one considers how different behaviors can occur on a scale of hours or even days, whereas related neuronal activity can occur on a scale of milliseconds [50, 20, 42], even more so as more and more variables are included in the data and the dependencies become even more computationally intense to analyze accurately (the "curse of dimensionality") [35]. Thus, in order to properly analyze such copious amounts of intricate high-dimensional data, we need methods that are both resistant to high dimensionality as well as equipped with the ability to properly analyze dependencies between temporally-disparate variables in time series.

1.2 Core Paper and Project Goals

The core paper by Kudryashova *et. al*, *Parametric Copula-GP model for analyzing multidimensional neuronal and behavioral relationships* [36], aims to fulfill this requirement via a novel framework for parametric estimation of multivariate distributions and mutual information through the use of *copulas*, a probabilistic structure adept at mutual information estimation. The core paper's contributions to the body of work includes (but is not limited to; see section 2.3) the development of the *Copula-GP* python package, a package which allows for the training of copulas, model selection, and copula entropy extraction. *Copula-GP* was found to outperform other non-parametric mutual information estimators when predicting mutual information for a larger number of variables (≥ 10) [36]. This combined with it's robustness to the "curse of dimensionality" makes it well suited for analysis of highly dimensional neuronal data.

In this project, we also validated usage of *Copula-GP* together with *Gaussian Process Factor Analysis* (GPFA), a dimensionality reduction method robust to qualities of neuronal data through which we are able to extract latent variables (we utilize the shorthand name of ***Copula-GPFA*** for the combination of methods). We validated combining such methods through confirming the statistical dependencies of simulated spike-data with latent variables of zero entropy, and through confirming statistical dependence of neuronal data extracted from mice on pupil dilation. Within the literature, there is very little use of GPFA with mutual information estimators, much less conditional parametric copulas on neuronal data (or on any data). This project aimed to fill that gap, with the goal of improving the interpretability of the model by capturing processes in the brain across groups of neurons as latent trajectories extracted from spike-train data and estimating the mutual information between latent trajectories $X_1, X_2, \cdots, X_N$ and a behavioral variable Y, as well as the information interaction $I(X_1 : X_2 : \cdots : X_N \leftarrow Y)$ (see equation (3.9)) measuring statistical dependence in $X_1, X_2, \cdots, X_N$ captured by Y.

We also aimed to extend *Copula-GP* with a method of ensemble model selection through the use of *bagging*, wherein we aggregate copula estimations of individual *Copula-GP* estimators fit on separate samples of data, validating the ensemble method via entropy estimation of a known distribution and culminating in usage of bagged *Copula-GPFA* on aforementioned real life neuronal data.

To recap, major project goals (with original goals and extensions noted) included:

- Validation of combining novel *Copula-GP* with GPFA (the combination of which we dub *Copula-GPFA* for this paper), with the expected benefits of robustness to dimensionality and efficient, detailed analysis of neuronal dependence (an original goal).
- Confirming dependencies between neuronal data from the visual cortex and pupil dilation through the use of *Copula-GPFA* (an original goal).
- Justification of possible improvement to the *Copula-GP* algorithm through the addition of bagging, confirmed via validation tests and application of bagged *Copula-GPFA* to real-life neuronal data (an extension).

1.3 Paper Outline and Project Contributions

In this introduction, we established the project setting, motivation, and goals. In the Background Chapter (chapter 2), we introduce basic information theory and neuroscience principles utilized in this paper, as well as an analysis of prior related work in the field. In the Methods chapter (chapter 3), we expand upon previously introduced principles through the formal introduction of copulas and copula entropy and discuss the implementation of *Copula-GP*. We also go over the justifications of a bagging extension (both formalistic and practical), as well as a description of the bagging algorithm utilized and implemented. We finally round off the Methods chapter with a description of GPFA, and intended application with *Copula-GP*. In the Data Providence chapter (chapter 4), we discuss the primary datasets used, including data background, acquisition, and cleaning. In the Baseline *Copula-GPFA* Validation and Experiments chapter (chapter 5), we delve into initial validation on *in silicon* data and experiments on *in vivo* data utilizing *Copula-GPFA*, with discussion of results found. We validate variations of the proposed bagging algorithm alongside baseline *Copula-GP* on simulated copula distributions and apply said bagging algorithm to real life mouse neuronal data in the Bagging Validation and Experiments chapter (chapter 6). Finally, we round things off with an overview of contributions made, possible project extensions, and future applications of methods in the Conclusion chapter (chapter 7).

A brief list of project contributions include:

- A validation of combining novel methods (*Copula-GPFA)* within the field of computational neuroscience.
- Application of such methods on *in silicon* and *in vivo* data for validation and experimental purposes.
- An extension of one of the novel methods (*Copula-GP*) via a bagging algorithm, complete with formalistic and practical justification.
- Identification of improvements of the bagging algorithm over baseline, and application to the aforementioned experimental dataset.
- Miscellaneous code contributions to the *Copula-GP* python code base (see appendix).

Chapter 2

Background

The aim of this chapter is to provide background knowledge on information theory and neuroscience concepts utilized in this paper, as well as an overview of related work in the literature.

2.1 Information Theory Essentials

(Note: knowledge of basic probability theory is assumed in this section.) Information theory, in broad terms, is the study of formal information quantification and communication [64]. The key measure of information in this field is *entropy*, defined as the negative expectation of the log-probability of an event X's outcome x:

$$H(X) = -\mathbb{E}_X(p(x)\log p(x)),\ x \sim X. \tag{2.1}$$

Essentially, entropy measures the number of *bits of information* needed to encode the total uncertainty of event X [64], and as such acts as a quantification of uncertainty. We also define the conditional entropy of X given the outcome y of event Y as

$$H(X|Y) = -\mathbb{E}_{X,Y}(p(x,y)\log\frac{p(x,y)}{p(y)}),\ x \sim X,\ y \sim Y, \tag{2.2}$$

and so the *mutual information* between X and Y is

$$I(X:Y) = H(X) - H(X|Y). \tag{2.3}$$

Mutual information thus represents the *reduction in bits of uncertainty* surrounding X given the outcome of Y. In addition, this quantity is *symmetric* [64], and so $I(X:Y) = H(Y) - H(Y|X)$ as well. In other words, **mutual information describes statistical co-dependence between events X and Y**. In addition, we note that mutual information is positive semi-definite (≥ 0), and so only ever represents a reduction in uncertainty (this is intuitive; the unconditioned outcome of an event naturally encompasses all conditions). We extend this to the *conditional mutual information* of X and Y given the outcome of event Z,

$$I(X:Y|Z) = H(X|Z) - H(X,Y|Z), \tag{2.4}$$

where $H(X,Y|Z)$ is the *conditional joint entropy* of X and Y given the outcome of Z, or the bits of uncertainty in the joint system of X and Y under condition z [64]. Finally, mutual information is easily extendable to a definition of *joint mutual information* via replacing singular entropy with it's joint definition (as traditional entropy itself is not the primary concern of this paper, we will skip rigorously defining these terms for brevity) [64]. Joint mutual information and it's conditional counterpart are the primary quantities of interest we wish to estimate in this paper, for reasons we outline in the next section of this chapter.

2.2 Basic Neuroscience Definitions and Computational Neuroscinece Ideas

Before we continue, we will first go over basic neuroscience definitions.

- A *neuron* is the primary cell in the brain, groups of which encode cognitive and behavioral processes.
- Neurons encode such processes through transmitting information to one-another in *spikes*, or instantaneous voltage transmissions to surrounding neurons [26, 47].
- A *spike train* is the experimental process of recording subject neuronal responses and behaviors to various controlled stimuli, and in the process gain an understanding of how neurons relate to behavior and stimuli [55].
- A spike train is usually split into *trials* of recordings replicating the above process to account for variation in neuronal response [55].
- We call real-life spike train data *in vivo* spike train data. Due to the difficulty of recording neurons in a live subject, we often seek to simulate *in silicon* spike train data of known characteristics, a method of which we utilize in this paper [33, 13].

If a system of neurons encodes the information that goes into behaviors or cognition, then we expect a statistical co-dependence between spikes and behaviors in recorded spike-trains, or in other terms a **high quantity of mutual information** [26, 47]. By extension, if we are able to accurately estimate mutual information then we should thusly be able to map systems of neurons to behaviors and cognition.

As stated in the introduction, computational neuroscience aims to answer questions surrounding the encoding of cognition and behavior in parts of the brain. In general, hurdles in answering such questions in the past stemmed from the difficulty of recording groups of neurons' behavior with certainty [36], however as of January 2022 we now have the technology needed for accurate single-neuron resolution spike train trial recordings in the form of *Neuropixel probes* [65]. As such, the hurdle of neuron-behavior-stimulus mutual information analysis is no longer centered on lack of data, moving instead to the need for accurate and efficient computational and statistical mutual information estimation techniques, as well as required robustness to dimensionality when a large number of individual neurons and behavioral variables are analysed [36, 48, 69].

2.3 *Copula-GP* and Other Neuronal Mutual Information Estimators

There are many choice methods in the realm of neuronal mutual information estimation. Popular novel ones include Bias-Improved Kraskov-Stögbauer-Grassberger (BI-KSG) [24] and the Mutual Information Neuronal Estimator (MINE) [4], both of which are effective non-parametric methods. The *Copula-GP* method implemented in the core paper by Kudryashova *et. al* [36] is a parametric estimator found to generally outperform MINE and BI-KSG in mutual estimation accuracy on highly-dimensional data ($\geq$ 10 dimensions), and as such is our chosen method of neuronal mutual information analysis. It is a method based on *copulas*, a multivariate distribution with uniform marginals representative of cumulative distributions and a history of applications ranging from scientific analysis of dependency (as they are generally used in computational neuroscience [32]) to the more purely analytical and predictive (i.e. predicting the outcome of an insurance claim [30]). We delve more into the rigorous definition of copulas in section 3.1.

Copulas have been used in mutual-information estimation and dependency analysis in a variety of *in vivo* and *in silicon* neuronal data, including spike data, 2-photon calcium imaging, and multi-modal neuronal datasets [48, 63, 5, 32], and have been successfully used for both mutual information estimation and dependency shape analysis in neuronal data [36, 63, 5]. However, recent advances in GPU accelerated processing power (through the use of popular libraries such as *PyTorch* [51] and *GPyTorch* [25]) have allowed for further refinement of copula implementations and expanded the number of practical use cases, allowing for the implementation of such a method for high-dimensional neuronal data, like that of *Copula-GP* [36].

2.3.1 Primary Contributions of Core Paper by Kudryashova *et. al*

While the core paper implemented the primary method used in this paper (*Copula-GP*), it would be unfair to call that the only (or even primary) contribution; the paper's primary strength is the adherence to formalism in their justifications and implementations of the methods used; the authors' adherence to formal copula theory allow for the flexibility in use and application of *Copula-GP* and adds robustness to variability in data dependency shape (see section 3.1.5). In addition, it's comparisons to other popular methods for neuronal mutual information estimation (MINE and BI-KSG) and the validation on *in vivo* neuronal data are able to highlight that *Copula-GP* even in the worst case tended not to produce biased or overestimated results, whereas MINE and BI-KSG often do. These comparisons to baseline methods (as well as the use of *Copula-GP* on *in vivo* datasets) firmly ground *Copula-GP* as a state-of-the-art method within the setting of computational neuroscience. For these reasons, *Copula-GP* is the cornerstone statistical method of choice in this paper.

Chapter 3

Methods

3.1 Copulas

When looking at the relationships between random variables, we often seek to examine the "shape" of variables' dependencies, even when such variables do not share a distribution [11]. One of the choice methods when it comes to dependency and mutual information analysis are *copulas* [59, 36, 44].

3.1.1 Definition and Basic Properties

Copulas are multivariate distributions with uniform marginal distributions, which themselves usually represent marginal variables' cumulative distributions [36, 59] A d-dimensional copula function $C(u_1,u_2,\ldots,u_d) : [0,1]^d \to [0,1]$ is defined as a *cumulative distribution function (CDF)* of a vector on the unit hyper-cube $[0,1]^d$ with uniform marginals $\mathcal{U}_{[0,1]}$:

$$C(u_1,u_2,\ldots,u_d) = \Pr(U_1 \leq u_1, U_2 \leq u_2, \ldots, U_d \leq u_d), \tag{3.1}$$

where $U_n \sim \mathcal{U}_{[0,1]}$ [36, 59]. Variables of non-uniform distribution are "attached" to the marginals through the use of CDFs (which by definition map to the interval $[0,1]$ [68]) as functions of non-uniformly distributed variables, creating a copula as a joint CDF [46]. *Sklar's theorem* states that for a d-dimensional random vector $\mathbf{X} = \{X_1, X_2, \cdots, X_d\}$ and it's CDF $F_{\mathbf{X}}$ with marginals $F_1, F_2, \cdots, F_d$, there exists a copula C such that $\forall \mathbf{x} \in \mathbb{R}^d$, $\mathbf{x} \sim \mathbf{X}$,

$$F_{\mathbf{X}}(x_1,x_2,\ldots,x_d) = C(F_1(x_1), F_2(x_2), \ldots, F_d(x_d)),\ x_i \in \mathbb{R}. \tag{3.2}$$

[59, 68] What is most valuable about this construction is the allowance for the random variables $X_1, X_2, \cdots, X_d$ to take *any* distribution, allowing for meaningful dependency and mutual information analysis between variables of different probabilistic distributions [59]. We can also condition the copula on some continuous variable y, allowing for the parametrization of the copula for a variable like time, phase, other marginals, etc [36]:

$$F_{\mathbf{X}}(x_1,x_2,\ldots,x_d|y) = C(F_1(x_1|y), F_2(x_2|y), \ldots, F_d(x_d|y)|y),\ x_i, y \in \mathbb{R}. \tag{3.3}$$

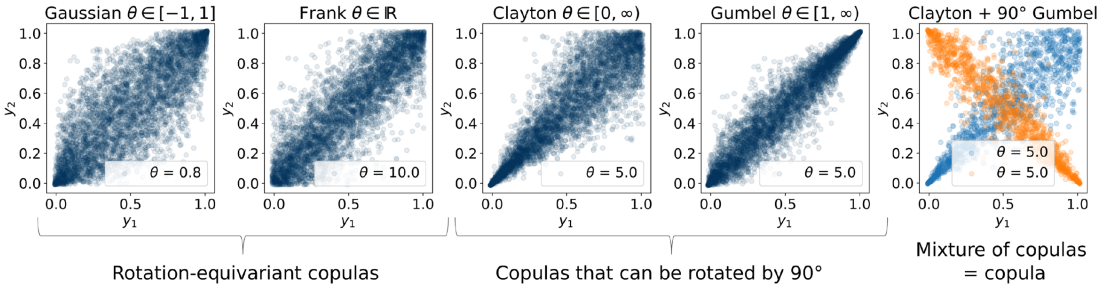


Figure 3.1: Various kinds of copulas. Note the difference in tail distribution representation, as well as how combining copula variants make a new *mixed* copula. Courtesy of [36].

Depending on how we define a copula's CDF, we can dynamically change the shape of the copula to represent different dependency relationships, with the intention of better fitting the marginal variables' dynamic dependency shape (i.e. dynamic tail dependencies, transitions into different copula families over time, dynamic increases or decreases in marginal distributions' correlation) [59, 36]. Doing so imposes strong assumptions on data however, and can introduce biases in analysis when the shape of the copula does not fit the true relationship of the individual variables [59]. Various copula families used in the core paper of this project [36] are depicted in figure 3.1.

3.1.2 Copula Entropy

As before, we consider a random vector $\mathbf{X} = \{X_1, X_2, \ldots, X_d\}$, where F_i describes a marginal distribution of X_i. let $\mathbf{u} = [u_1, u_2, \ldots, u_d]$, with $u_i = F_i(x_i)$. By (3.2) we know $F_{\mathbf{X}}$ is described by a copula C. We define their copula entropy as

$$H_C(\mathbf{X}) = -\int_C c(\mathbf{u}) \log c(\mathbf{u}) d\mathbf{u}, \tag{3.4}$$

where $c(\mathbf{u})$ represents the probability density of the copula at $\mathbf{u}$

$$c(\mathbf{u}) = \frac{\partial^d C}{\partial u_1 \partial u_2 \ldots \partial u_d}. \tag{3.5}$$

[41, 32] Let $\mathbf{x}$ be a random sample $\mathbf{x} \sim \mathbf{X}$. We find that the joint mutual information between the marginals of $\mathbf{X}$ is equal to the negative copula entropy of the distribution:

$$\begin{aligned} I(\mathbf{X}) &= \int_{\mathbf{x}} F_X(\mathbf{x}) \log \frac{F_X(\mathbf{x})}{\prod_i F_i(x_i)} d\mathbf{x} \\ &= \int_{\mathbf{x}} c(\mathbf{u}_\mathbf{x}) \prod_i F_i(x_i) \log c(\mathbf{u}_\mathbf{x}) d\mathbf{x} \\ &= \int_{\mathbf{x}} F_X(\mathbf{x}) c(\mathbf{u}_\mathbf{x}) \log(c(\mathbf{u}_\mathbf{x})) d\mathbf{x} \\ &= \int_C c(\mathbf{u}) \log c(\mathbf{u}) d\mathbf{u} \\ &= -H_C(\mathbf{X}). \end{aligned} \tag{3.6}$$

[41, 32] In other words, **mutual information is negative copula entropy.** This also implies that copula entropy is negative semi-definite as opposed to traditional entropy,

which is positive semi-definite. Suppose we have some random variable $y \sim Y$. We can extend the definition of copula entropy in (3.4) to be parameterized and thus conditioned in a outside variable y as

$$H_C(\mathbf{X}|y) = -\int_C c(\mathbf{u}|y)\log c(\mathbf{u}|y)d\mathbf{u},\ c(\mathbf{u}|y) = \frac{\partial^d C}{\partial F_{1|y}\partial F_{2|y}\dots\partial F_{d|y}} \tag{3.7}$$

(where each $F_{i|y}$ is the respective conditional marginal probability of X_i) [41] and can extend further to the definition of *conditional copula entropy* as

$$\begin{aligned} H_C(\mathbf{X}|Y) &= \int_y H_C(\mathbf{X}|y_i) \\ &= -\int_y I(\mathbf{X}|y_i) \\ &= -I(\mathbf{X}|Y). \end{aligned} \tag{3.8}$$

We also utilize the *interaction information* between $\mathbf{X}$ and Y,

$$I(\mathbf{X} \leftarrow Y) = I(\mathbf{X}|Y) - I(\mathbf{X}) = H_C(\mathbf{X}) - H_C(\mathbf{X}|Y), \tag{3.9}$$

representing the change in the mutual information between marginals of $\mathbf{X}$ when Y is learned. In the setting of this paper, we hope for the information interaction between the two to be firmly *negative*, implying information surrounding $\mathbf{X}$ is captured in Y and a statistical dependence between the two being posited. Working under the assumption that the brain is a source of behavioral and motor functions such as pupil dilation, then if a part of the brain is tied to such a function we expect the information interaction to be significantly negative.

Instead of calculating the difference between the copula entropies, we can also calculate the sum, which we find through the chain rule of mutual information is equivalent to the negative joint mutual information between marginals of X and Y, and thus their copula entropy:

$$\begin{aligned} H_C(\mathbf{X}) + H_C(\mathbf{X}|Y) = -(I(\mathbf{X}) + I(\mathbf{X}|Y)) &= -(I(X_1 : X_2 : \dots) + I(X_1 : X_2 : \dots|Y)) \\ &= -I(X_1 : X_2 : X_3 : \dots : Y) = H(X_1, X_2, \dots, Y). \end{aligned} \tag{3.10}$$

Thus, through this method of copula entropy estimation we are able to extract several metrics surrounding mutual information, both statistical dependence in the form of their information interaction and the joint mutual information between marginal variables of $\mathbf{X}$ and Y.

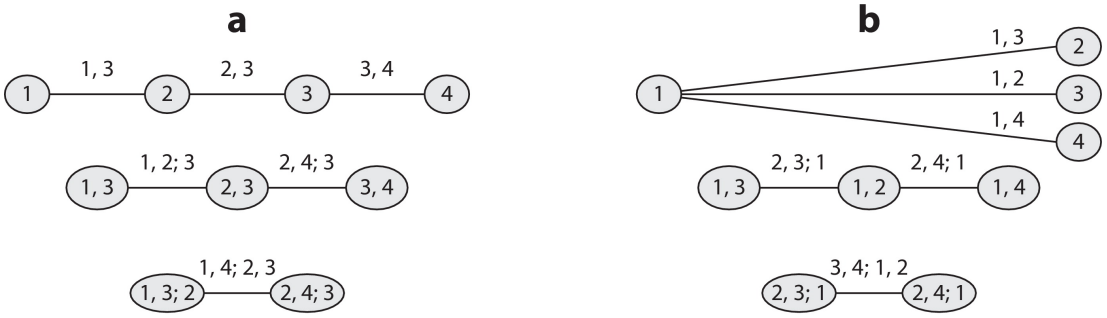
3.1.3 Vine Copulas

To scale against higher dimensions, copulas can take a *vine copula construction*, which factorize multivariate distributions into conditional distributions modelled as singular bivariate copulas [49, 44]. A single "vine" in this construction may be represented as a hierarchy of trees, where each node represents a single CDF (with increasing conditioning for each tree) and each edge is a pair copula. The cumulative distributions

of each tree's bivariate building blocks is then used as the nodes of the next tree in the hierarchy, and so extends the set of conditioning variables into the next tree (See figure 3.2 for a visualisation of a copula vine tree hierarchy in $d = 4$ dimensions) [17]. A multivariate distribution $f_{\mathbf{X}}$ with univariate marginal variable distributions $X_1, X_2, \cdots, X_d$ can be factorized in this way as

$$f_{\mathbf{X}}(x_1, \cdots, x_d) = \prod_{k=1}^{d} f(x_k) \prod_{j=1}^{d-1} \prod_{i=1}^{d-j} C_{j,i|1,\cdots,j-1} \left(F(x_j | x_1, \cdots, x_{j-1}), F(x_{i+j} | x_1, \cdots, x_{j-1}) \right). \tag{3.11}$$

This specific factorization is called a *canonical vine*, or C-vine [49]. What is special about vine copulas is both assumed independence between particular marginals in a tree (due to conditional independence) and the decomposition of a high dimensional distribution into bivariate building blocks. As such sampling from and analysis of a high dimensional vine copula is not as affected by the curse of dimensionality given robustness in the bivariate case [17, 45]. This, combined with the ability for parameterization in time and/or space, make vine copulas well suited for analysis of complex neuronal recordings [49, 36].



Czado C, Nagler T. 2022
Annu. Rev. Stat. Appl. 9:453–77

Figure 3.2: Two possible constructions for a copula vine in $d = 4$ dimensions. Panel A) describes a D-Vine construction, whereas panel B) describes a C-Vine. Nodes in these trees represent distributions of enumerated variables (1 to 4), possibly conditioned on some variable(s), whereas edges represent copulas modelling the bivariate distribution with node distributions as marginals. Courtesy of [17].

3.1.4 Gaussian-Process for Dependence Parameterization

We further expand upon vine copulas with the addition of *Gaussian Process (GP)* as a method of estimating parameterization of the relationship between a copula's marginals and dependency set [36, 28]. A GP estimator is a regression attempting to estimate the true relationship between dependent and independent variables by defining a distribution over functions such that observed points might follow a Gaussian distribution (or, in short, by means of Bayesian inference) [61]. In more rigid terms, we assume output y of a function f given a set of observations $\mathbf{x}$ can be given as

$$y = f(\mathbf{x}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2), \quad \mathbf{f} \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x'})) \tag{3.12}$$

where σ_ε^2 takes the usual definition as the variance of ε, m is the expected value of $\mathbf{f}$ given $\mathbf{x}$, and k is the covariance between the outputs of two separate sets of observations $\mathbf{x}$ and $\mathbf{x'}$ (this specific parameter is often called the *kernel* of the estimated distribution $\mathcal{GP}$) [61]. Different methods of estimating k, m, and σ_ε^2 exist [61, 21, 28], and so a method should be selected based on use case. Incorporating GP into our copula vine, we are able to extend our multivariate distribution to have dynamic dependency relationships estimated by means of GP. To express this mathematically, we redefine individual copulas to take the approximate form

$$F_{\mathbf{X}}(x_1,\cdots,x_d|y') = C(F_1(x_1|\theta_1(y')),F_2(x_2|\theta_2(y')),\cdots,F_d(x_d|\theta_d(y'))|\theta'(y')), \quad (3.13)$$

where $\theta_1,\cdots,\theta_d,\theta' : \mathbb{R} \to \mathrm{dom(y)}$ are functions mapping some parameter y' onto the domain of the conditioning variable y representing the relationships between $x_1,\cdots,x_d$ and y', often called *GPLink* functions [36, 61]. These GPlink functions are what we predict via GP, and how GPLinks are estimated varies depending on the specific use case and implementation of GP [61, 36, 28]. The main innovation extending copulas in this way is both uncertainty in how the relationship between parameters and marginals are defined, as well as parameterization of marginals' dependence. The latter is particularly useful when the actual dependence relationships in the data are not static and may instead be dynamic over time [36].

Table 3.1: Different GPLink functions used in *Copula-GP* for different copula types. GPLinks are used to parameterize dependence of the particular copula in time (or any other continuous variable. Courtesy of [36].

Copula	Domain	GPLink(f): $\mathbb{R} \to \mathrm{dom}(c_j)$
Independence	-	-
Gaussian	$[-1,1]$	$\mathrm{Erf}(f/1.4)$
Frank	$(-\infty,\infty)$	$0.3\cdot f+\mathrm{sign}(f)\cdot(0.3\cdot f)^2$
Clayton	$[0,\infty)$	$\mathrm{Exp}(0.3\cdot f)$
Gumbel	$[1,\infty)$	$1+\mathrm{Exp}(0.3\cdot f)$

3.1.5 Benefits of Parametric Vs. Non-parametric Copulas

One possible criticism surrounding copulas is that they impose assumptions surrounding the marginal distributions' dependency shape; imposing a certain copula can introduce bias in the data if said copula's assumptions are not met. While this bias could effect our findings in the non-parametric case, utilization of copula parameterization and usage of mixed copula construction (see eq. (3.14)) circumvents this through dynamic tailoring of copula variant to the parameterizing value [36, 48].

3.2 *Copula-GP*: A Novel Framework For Dependency Analysis

While copulas have been utilized in modelling multivariate neuronal dependence and mutual-information [36, 48, 44], the GP-treated vine copulas described previously had

only been extensively used on financial time series prior to the core paper [36, 28]. The *Copula-GP* package was implemented in Python and deployed in Kudryashova *et. al's* paper [36] (which was published in 2022), and is one of the few (if not only) practical implementations of a GP-treated copula vine model designed with GPU based acceleration of computation via *Pytorch* [51]. It is also the first of such models designed with use on neuronal data in mind, and as such literature on the uses of GP vine copulas on modelling neuronal data is heavily limited, with the main source for such claims being the core paper. In simplest terms, the package serves as a framework for GP copula model parameter estimation, copula model selection, and model deployment as a vine, with additional built in visualization and entropy extraction utility. The core paper also utilizes comparison to other popular methods of neuronal mutual information extraction, yielding superior results in highly-dimensional data (see section 2.3). **Any subsequent claim surrounding the *Copula-GP* package without a citation attached in this section comes courtesy of [36]**.

3.2.1 Implementation

The vine construction implemented used copula building blocks from five distinct families: independence, Gumbel, Gaussian, Frank, and Clayton copulas, with the independence copula preferred for independent variables (see figure 3.1). The factorization of the vine is that of the C-vine described by (3.11), with accommodations made for GP-parameterization. To fully capture the tail dependencies and negative correlations in relationships between marginal distributions, *mixed copulas* were utilized. In the core paper, these are defined as

$$C_{mixed}(\mathbf{X}|Y) = \sum_{j=1}^{K} \phi_j(Y) C_j(\mathbf{X}|\theta_j(Y)), \tag{3.14}$$

where K is the number of elements, ϕ_j is the *concentration* of the jth copula (c_j), and θ_j is the jth copula's parameter GPLink. The GPLink for each copula is determined by it's copula variant, as shown in figure 3.1, with θ being defined by GPLink(f), where f is sampled from $\theta \sim \mathcal{N}(\mu, K_\lambda(X,X))$ (the choice of GPLink depends on the kind of copula; see 3.1). GP is also utilized to parameterize the concentrations ϕ_j, which are defined as

$$\phi_j = (1-t_j)\prod_{m=1}^{j-1} t_m,\ \ t_m = \Phi\left(\tilde{f}_m + \Phi^{-1}\left(\frac{M-m-1}{M-m}\right)\right),\ \ t_M = 0, \tag{3.15}$$

where Φ is the CDF of a standard normal distribution and $\tilde{f}_m$ is sampled from $\tilde{\mathbf{f}}_\mathbf{m} \sim \mathcal{N}(\tilde{\mu}_m,\ \tilde{K}_{\tilde{\lambda}_m}(Y,Y))$. This gives us $2M-1$ sets of hyper parameters to estimate, $\{\lambda\}_M$ kernel hyperparameters for each GPLink θ and $\{\tilde{\lambda}\}_{M-1}$ kernel hyperparameters for each concentration function ϕ, estimated via the methods described in the next section.

3.2.2 Model Selection and Parameter Estimation

As stated in section 2.2.1, the parameters for the distributions used in GP must be estimated. In the *Copula-GP* framework, this is accomplished through the use of

stochastic variational inference (SVI), with SVI being scaled to high dimensions by means of *Kernel Interpolation for Scalable Structured Gaussian Process (KISS-GP)* [70]. These methods were specifically used for efficient implementation in aforementioned python libraries *PyTorch* and *GPyTorch* [51, 25]. For model hyper-parameter selection, *Watanabe-–Akaike information criterion (WAIC)* was used, a metric which aims to maximise the Akaike information criterion (AIC) by means of a Bayesian-approach (we more rigorously define AIC in section 3.3). Given f_1, f_2 and the true distribution g, WAIC examines the difference in log-likelihood i.e. $I(g:f_1)-I(g:f_2)=-\mathbb{E}(\log f_1(X)-\log f_2(X))$, and selects the model with the greater mutual information [1]. The space of models is too large to find the optimal model when considering the number of combinations of copulas shown in 3.1, and as such the core paper implements a greedy algorithm of minimising WAIC which can be used with all copula types and a heuristic algorithm specifically tuned for certain combinations of copula types. For this project, we utilize the heuristic approach of copula selection.

3.2.3 Entropy and Mutual Information Estimation

As *Copula-GP* models a copula distribution, possible states of variables can be sampled from it. As such, the joint mutual information $I(X_1:X_2:\cdots:Y)=H_C(\mathbf{X}|Y)-H_C(\mathbf{X})$ between parameter Y and multivariate distribution $\mathbf{X}=\{X_1,X_2,\dots\}$ can be derived from $C(\mathbf{X}|Y)$ (the C-vine copula *Copula-GP* estimates). Computing this directly via integration over the estimated copula density (the "direct integration approach") is computationally intensive due to nested integrals, and as such a "estimated approach" is explored. We choose to find the *monte carlo* estimate [58] of $H(X)$ (which is also utilized in the core paper), which involves sampling from the distribution $C(X|\hat{y}_i)$ with parameterization in N random values $\hat{y}_i\sim\mathcal{U}(0,1)$ and estimating the conditional entropy as the mean

$$H_C(X|Y)=\int_{\mathrm{dom}(y)}H_C(X|y_i)dy_i\approx\frac{1}{N}\sum_{i=1}^{N}H_C(X|\hat{y}_i). \tag{3.16}$$

Similarly, we estimate the unconditional entropy as the mean entropy of the system found when fed true values of the parameterizing variable, i.e

$$H_C(X)=\mathbb{E}_y(H_C(X|y))\approx\frac{1}{N}\sum_{n=1}^{N}H_C(X|y_n). \tag{3.17}$$

Due to the law of large numbers, as N goes to infinity both approximations become more accurate. To find the copula entropy values $H_C(X|y_i)$, we utilize the *Copula-GP*'s vine implementation's `entropy()` function to estimate conditional entropy for a fixed conditioning y via integrating over the estimated probability density of the copula.

3.2.4 Performance and Evaluation of Model Implementation in Core Paper

As discussed in the core paper and background chapter, the *Copula-GP* framework was successful as a method of mutual information estimation. When compared to state of the

art methods such as DEMINE [39] and BI-KSG [29], *Copula-GP* outperforms them in the high dimensional neuronal data used in the core paper (both artificial data in the form of a baseline multivariate Gaussian, a low-entropy multivariate student-t distribution, and a no-exact-fit transformed Gaussian distribution) as the comparison methods underperforms when used with low-sample high-dimensional data. In order to escape the "curse of dimensionality," assumptions surrounding the data must be made that BI-KSG and DEMINE do not do by themselves. *Copula-GP* is able to make said assumptions through assuming certain dependency shapes between marginals by means of selection of copula variant, tailoring the model's assumptions to fit those present in the data. The model was found to be a more reliable and accurate estimator of mutual-information, capturing more mutual information without overestimation than non-parametric models. In addition, when validated on a neuronal recording in the V1 cortex of live mice, it was able to identify behaviorally-relevant locations within the V1 cortex without prior knowledge of the tasks the mice were performing; copulas estimated via *Copula-GP* modelling the dependencies between neuronal data and behavioral data (licks) observed an increase in negative copula entropy (and thus an increase in mutual information) at the same time that a reward stimulus was given.

3.2.5 Time Complexity of fitting *Copula-GP* C-Vine

GP parameter inference in *Copula-GP* works on $O(n)$ time (n being the length of data), and as this is repeated for each bivariate copula we find a naive algorithmic complexity of $O(n \cdot d^2)$, where d is the number of dimensions. However, it was found that the practical complexity of the algorithm is instead much smaller as independence copulas require no parameterization. The number of non-independence copulas N_{ni} is much smaller than $d(d-1)/2$ and as such the effective number of dimensions for calculation of computational complexity is $m_{eff} \sim \sqrt{N_{ni}}$, and the true computational complexity of the model was found to scale on $O(n \cdot N_{ni}) \sim O(n \cdot m_{eff}^2)$. As all copulas (even independent copulas) take some time to train no matter how small the number of data points, an estimate for the true complexity of C-vine estimation is thus given as $n \cdot (m_{eff}^2 + c \cdot m^2)$, where c is a small constant. This low-exponent polynomial scaling in dimensions for parameter inference is yet another example of *Copula-GP*'s robustness to dimensionality, and makes *Copula-GP* an efficient method of examining the dependence structures of a large amount of neurons.

3.3 Extension of *Copula-GP*: Bagging

Copula-GP assumes continuous data for copula fit. However, often we find data is batched into controlled trials to measure properties of a one-time neuronal response against consistent stimuli [3, 10]. As such, we implemented an extension to allow for analysis via *bagging*. Bagging is a bootstrapping technique wherein multiple "weak" models are trained and their outcomes aggregated in some way to create a "stronger" model [8], typically reducing variance and increasing model bias. We propose a formal justification of copula bagging, taking inspiration from the "Random Forest" algorithm for bagged regression [9]; we assume the existence of some true copula C describing a multivariate distribution X. We take N samples $\{S_1, S_2, \ldots, S_N\}$ and get a copula

$C^{(n)}$ representing the best copula fit possible on an individual sample S_n. We then come to a final estimate via a mean aggregation of our estimates. Suppose $C^{(n)}$ is a C-vine estimate, and let $C^{(n)}_{i,j}$ represent the i-th copula in the j-th layer of the n-th estimate. We can create a mean mixture copula $\hat{C}_{i,j}$ representing the final estimate for the corresponding true copula $C_{i,j}$:

$$\hat{C}_{i,j} = \frac{1}{N}\sum_{n=1}^{N} C^{(n)}_{i,j}.$$

As $N \to \infty$, by the law of large numbers (and the fact that copulas are distributions) we find $\hat{C}_{i,j} \to C_{i,j}$, and so $\hat{C} \to C$. In other words, by fitting copulas on individual spike-train trials and taking the aggregate, we estimate a copula that more closely aligns with the true probabilistic relationship than those aggregated.

3.3.1 Weighted Aggregation Methods

A simple unweighted mean is appropriate in the infinite case. In practice however, we often find that a naive mean is insufficient when examining a finite number of samples; suppose we have finite N uniform length samples S_n and thus have a set of observations $S = \{S_1, S_2, \ldots, S_N\}$, fitting corresponding copula estimates $\hat{C}_n$. If the n-th sample is an outlier sample and so $\hat{C}_n$ does not resemble the true copula C_n, we naturally should discard it. Our first aggregation method utilized the mean bucketed goodness-of-fit $\overline{R}^2$ score found in the core paper [36],

$$\overline{R}^2 = \mathbb{E}_S(R^2(\hat{C}, S_n)), \tag{3.18}$$

where R^2 takes the usual definition of explained variance between the conditional empirical cdf (ecdf) of observations in a portion of all observations S_n and the corresponding conditional copula cdf (ccdf) of the fit copula,

$$R^2(\hat{C}, S_n) = 1 - \sum_{u_1, u_2 \in S_n} \frac{(\text{ecdf}(u_1|u_2, S_n) - \text{ccdf}_{\hat{C}}(u_1|u_2, S_n))^2}{(\text{ecdf}(u_1|u_2, S_n) - 0.5)^2}. \tag{3.19}$$

Given the copula $\overline{R}^2$ scores, we can discard all copulas with scores not around the best fit found and aggregate only copulas with high explained-variance. Alternatively, we may also utilize weighted aggregation based on information criterion. We utilized a Bayesian model aggregation approach utilized by S. Hu *et. al* [30], which aggregated copulas via *Bayesian Information Criterion* (BIC). The BIC value for the n-th estimated copula is given as

$$\text{BIC}_\text{n} = -2\log \mathcal{L}(\hat{C}_n|X, \Theta) + p_n \log|X|, \tag{3.20}$$

where $\log \mathcal{L}(\hat{C}_n|X, \Theta)$ is the log-likelihood of copula $\hat{C}_n$ under observations X and estimated model parameters Θ, and p_n is the total number of estimated parameters of $\hat{C}_n$ (in the case of *Copula-GP*, each bivariate copula estimated is a mixture copula and so possesses mixing and dependence parameters for each copula mixed; if the mixture is a singleton mixture, we only count dependence parameter. Note independence copulas have no dependence parameter). BIC essentially rewards model log-likelihood with

penalty in number of parameters scaling with sample-size [62, 30], with small BIC values are being preferred (as negative as possible). We arrive at weights for the n-th model as

$$W_{n,\text{BIC}} = \frac{\exp(-\frac{1}{2}BIC_n)}{\sum_m \exp(-\frac{1}{2}BIC_n)}. \tag{3.21}$$

[30] Note this weighting is *logarithmic* in BIC_n; consider two models $\hat{C}_1$ with BIC of -0.5 and $\hat{C}_2$ with BIC of -0.1. Then the weights of their aggregation as per (3.21) are

$$\begin{aligned} W_{1,\text{BIC}} &= \frac{\exp(-\frac{1}{2}(-0.5))}{\exp(-\frac{1}{2}(-0.5)) + \exp(-\frac{1}{2}(-0.1))} \\ &\approx 0.55 = 1 - 0.45 \approx 1 - W_{2,\text{BIC}} = 1 - \frac{\exp(-\frac{1}{2}(-0.1))}{\exp(-\frac{1}{2}(-0.5)) + \exp(-\frac{1}{2}(-0.1))}. \end{aligned} \tag{3.22}$$

We also utilize the *Akaike Information Criterion* (AIC) for comparison, which is given for the n-th copula estimation as

$$\text{AIC}_\text{n} = -2\log \mathcal{L}(X|\hat{C}_n, \Theta) + 2 * p_k, \tag{3.23}$$

and extract weights $W_{n,\text{AIC}}$ accordingly via (3.21), replacing BIC_n with AIC_n. AIC is similar to BIC in that it rewards log-likelihood but has a more relaxed penalty in number of parameters. If a more complex copula matches the true distribution, this can lead to possible improvements in aggregation accuracy, but can also lead to over-fitting via over-parameterization [72] (as we are already utilising mixtures of copulas with a high ceiling in number of parameters, we expect the latter to be true). We additionally include dynamic weighting of estimates via calculating the above information criteria and their respective weights point-wise as opposed to setting weights to be constant over the observations X; we call point-wise aggregation *dynamic bagging* and constant weighting *static bagging*. We validate our bagging methods with different aggregation methods on data samples from randomly generated bivariate copulas in section 6.1.

To aggregate C-vines, we can bag one layer at a time; we first bag copulas in the current layer normally, and gather cumulative conditional probabilities as pseudo-observations via the bagged copulas' ccdfs. We then utilize them for BIC, AIC, and R^2 calculations in the next layers' bagging process, and in doing so effectively propagate the previous layers' weightings to the next layer. As subsequent bagged copulas will have worse BIC, AIC, and R^2 calculations if their corresponding C-vines' previous layers were found to be bad fits, we in effect prioritize relationships with weaker conditioning using this method.

3.3.2 Aggregation Implementation

When *Copula-GP* estimates a parametric C-vine along a given 1-D parameterization input X, it models each individual copula as a mixture and gets estimated dependence parameterizations $\{\theta_{i,j}(X)\}$ and estimated mixing parameterizations $\{\phi_{i,j}(Y)\}$ for each of i mixture copulas, with each mixture copula being a mix of $j^{(i)}$ copula variants (see section 3.2). As such, our actual implemented bagging procedure is to aggregate along

these estimated parameters for each unique copula variant in each mixture (we count each rotation variant of a Clayton or Gumbel copula as unique). This algorithm is outlined more formally in algorithm 1. Example application of this algorithm with mean bagging can be seen in figure A.3. In addition, we implemented model selection for each C-vine we aggregate, allowing for the bagged vine to contain copula mixtures possibly unexamined in the heuristic based model selection process. The C-vine aggregation process is much simpler in implementation, requiring aggregation of copulas by layer and storing the cumulative distributions of each copula via the bagged copula ccdfs for weight extraction in the next layer. As such we omit it's implementation in this paper (for now; it may be added to the appendix later on). Note that aggregation was implemented as a module in a local *Copula-GP* code-base.

3.3.2.0.1 Complexity of Implementation Suppose we have N samples of uniform length n and dimensionality m. Then the algorithmic complexity of the described bagging algorithm would thus be $O(N \cdot n \cdot m^2)$. This is the same algorithmic complexity achieved via fitting a C-vine via *Copula-GP* on the concatenation of samples. However, if we recall from section 3.2.5 we require a small amount of time c to train a bivariate copula no matter the length of the trial or size of copula. Thus we find the practical complexity becomes $N \cdot (n \cdot m_{eff}^2 + c \cdot m^2)$, becoming slightly larger than the practical complexity of fitting on the full trials $N \cdot m_{eff}^2 + c \cdot m^2$ by about $(N-1) \cdot c \cdot m^2$. In addition, when utilizing model selection for each *Copula-GP* estimator, we find the time it takes for model selection scales linearly in the number of estimators, which can be impractical if the number of estimators is high. For example if fitting a C-vine estimator takes 5 hours in the non-ensemble case, then in the 10 estimator ensemble case it will take roughly 50 hours. As such, for highly-dimensional data we recommend to utilize a low number of estimators (single digit) if time-to-fit is a concern. As time-to-fit indeed was a concern for this project, we utilized 4 estimators for bagged estimations made.

3.3.3 Practical Justification

A averaging rule for bagging has been used to great effect with both GP [14] and copulas [30]. T Chen *et. al* [14] verified that a simple averaging bagging regime can boost GP accuracy to great effect, especially when the GP aims to predict largely unseen and/or uncertain (difficult-to-measure) variables utilizing known and measurable (easy-to-measure) variables, and in the case of the *in vivo* dataset used *Copula-GP* utilized GP to model a copula parameterization variable (a difficult-to-measure variable) as an outcome of pupil dilation (a easy-to-measure variable). As our bagging algorithm implementation (see alg.1) effectively aggregated GP outcomes, T Chen *et. al*'s findings may be applicable. In addition, a recent paper by S Hu *et. al* [30] found that weighted bagging of copula families allowed for a unified estimation of tail dependency and increased an estimated distribution's resemblance to the true distribution. As such, we posit that extending *Copula-GP* with bagging through copula variant and parameterization aggregation may allow for more accurate representation of dynamic dependency shape. Finally, A.2 trained estimators have higher uncertainty for parameterizations which deviate off of mean, as is seen later on copulas fit on the experimental dataset (see figure A.2). Bagging typically reduces variance through aggregation [8, 14], and thus a bagged

Algorithm 1 Implemented bivariate mixture copula bagging algorithm. Here, mixture copulas consist of mixed copula variants *variants*, their corresponding dependency parameter values Θ (for each of the m inputs), and their corresponding mixing parameters *mix*. We assume weights have been defined over all m input points.

Input
    mixture: A list of mixture copulas to aggregate.
    weights: Corresponding mixture weights.
Output
    mixture(Variant List, Mix Parameters, Θ): A new mixture copula of variants *Variant List* with corresponding mixture parameters *Mix Parameters* and dependency parameters Θ.
procedure BAGCOPULAS(*mixtures, weights*)
    $N \leftarrow 0$ ▷ Unique variant counter.
    indexes $\leftarrow$ *dict()* ▷ Indexing dictionary.
    Variants Seen $\leftarrow$ *set()* ▷ Set of Copula Variants
    Variant Counts $\leftarrow$ *dict()* ▷ Counting dictionary.
    Total Variant Weight $\leftarrow$ *dict()* ▷ Weighting dictionary.
    for *i, mixture* in enumerated(*mixtures*) **do**
        for *n, variant* in enumerated(*mixture.variants*) **do**
            if *variant* not in *Variants Seen* **then**
                Variants Seen.add(variant)
                $indexes[variant] \leftarrow N$
                $Variant\ Counts[N] \leftarrow 0.0$
                $Total\ Variant\ Weight[N] \leftarrow 0.0$
                $N \leftarrow N + 1$
            $idx \leftarrow indexes[variant]$
            $Variant\ Counts[idx] \leftarrow self + 1$
            $indexes[(i,n)] \leftarrow idx$
            $Total\ Variant\ Weight[idx] \leftarrow self + weights[i]$
    Variant List $\leftarrow$ *list*(N) ▷ Final list of variants.
    Mix Parameters $\leftarrow \mathbf{0}_{(N \times m)}$ ▷ Mixture parameters.
    $\Theta \leftarrow \mathbf{0}_{(N \times m)}$ ▷ Dependency parameters.
    for *i, mixture* in enumerated(*copulas*) **do**
        for *n, variant* in enumerated(*mixture.variants*) **do**
            $idx \leftarrow indexes[(i,n)]$
            $Variant\ List[idx] \leftarrow variant$
            $Mix\ Parameters[idx] \leftarrow self + mixture.mix[n] * weights[i]$
            $\Theta[idx] \leftarrow self + mixture.\Theta[n] * weights[i] / Total\ Variant\ Weight[idx]$
    return new *mixture(Variant List, Mix Parameters,* Θ)

estimator may give a more robust prediction when encountering outlier parameterizing variables.

3.4 Gaussian-Process Factor Analysis

Our final method covered is our choice dimensionality reduction technique. In general, the aim of dimension reduction techniques is to discover a handful of unobserved latent variables that can be used to accurately describe all of the observed variables in our data. Doing so allows us to both mitigate the problems associated with high dimensionality, as well as possibly improve the interpretability and informativaty of our model and/or data [53, 57]. A very common technique in the realm of dimension reduction for highly dimensional neuronal data is *Gaussian-Process Factor Analysis* (GPFA), which factors a matrix of observations X into a product of a loading matrix Ψ and a scoring matrix Θ as $X = \Psi\Theta$ (the *Factor Analysis* step) producing a low-dimensional group of trajectories, and then smooths said data by means of fitting a GP (the *Gaussian Process* step) [53]. This operates under the assumption of a linear relationship between observations X and latent trajectories Θ (utilizing Ψ as a transformation matrix) [71], and models the GP with bias d as

$$X = \Psi\Theta + d + \varepsilon,\ \varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2) \tag{3.24}$$

with parameters $(d, \Psi, \sigma_\varepsilon^2)$.

3.4.1 Python Implementation Used

The implementation of GPFA utilized is sourced from the *Elephant* (Electrophysiology Analysis Toolkit) python library, with the package's 1.0.0 release (used for the contents of this paper) being published November 10, 2023 [19]. The *Elephant* package was specifically designed for use on neuronal data, motivated by a push to release a standardized python package for use in computation neuroscience. The GPFA module has specifically seen use in recent papers [52, 3], and since release has become quite popular. The GPFA module receives some dimension n to reduce down to and time-bucket size m (we utilize 10ms and 20ms buckets) to instantiate a GPFA python object. This object can then be fit on a number of spike train recordings of uniform temporal length given the start and end time of each recording, summing the number of spikes for each time bucket and utilizing the bucketed spike counts as the observation matrix in equation (3.24). As each projection corresponds to the expectation of the latent trajectories $\mathbb{E}(\Theta|\Psi)$, these parameters are estimated in the *Elephant* implementation via expectation maximization [71].

3.4.2 Utilization with *Copula-GP* and Proposed Benefits

The *Copula-GPFA* process consists of GPFA applied to neuronal data to extract latent trajectory estimations $\hat{X}_1, \hat{X}_2, \ldots$, followed by fitting a *Copula-GP* C-vine estimator on neuronal trajectories. Doing so, we are able to efficiently extract mutual information estimates describing not just the information interaction between the examined part of the brain and some parameterizing variable, but the mutual information between the

brain processes driving neurons as instead of individual neurons themselves [71, 41, 36], enhancing the interpretability of individual copulas estimated. In addition, by utilizing GPFA to reduce a large number of neurons to a more manageable amount, we reduce the computational intensity of mutual information analysis; if we extract 13 trajectories from $\sim$150 neurons (as is done in section 5.2.1), then from the naive estimate for the complexity of training *Copula-GP* estimators (see section 3.2.5) we can come to an upper-bound of $150^2 \div 13^2 \approx 133$ times faster *Copula-GP* training.

3.4.3 Dimensionality Selection

Before we can utilize GPFA, we first must isolate which number of dimensions n to reduce down to. For the *in silicon* dataset, we utilize $n = 3$ dimensions, which is the dimensionality of a Lorenz system (see section 5.1). On the experimental *in vivo* dataset we chose n via investigating the log likelihood of GPFA fits extracted from a 3-fold cross validation, from target dimensionality $n = 1$ to 50. We then plotted the log likelihoods and saw both where the elbow, or the point of maximum concave slope curvature representing a "good enough" dimension to reduce down to [2] of the log likelihood curve was. While there are computational methods for elbow selection [2], there are few enough points to allow for the elbow to be selected ourselves visually. While it is entirely possible that optimal log-likelihood may lie further beyond 50 dimension, fitting *Copula-GPFA* on $n > 50$ dimensions is beyond the computational capabilities of this project.

3.4.4 Additional Post-GPFA Interim Processing Required

Copula-GP's C-vine framework is fit on single-trial continuous, however *Elephant*'s GPFA implementation produces trajectory data that is split into trials. A solution to this would be to concatenate trials trajectory wise, however per-trial drift in trajectory means can result in weak *Copula-GP* fit performance if these lead to large jump discontinuities and thus loss of smoothness in the data; the kernel for the GP-link functions used in parameterization will require more restrictions as it encodes the smoothness of the data (among other things) [61, 71]. For the *in vivo* data, we found in figure 5.3 that drift occurs in the 1 second inter-stimulus break in the average trial. As such, we crop this period out of each trial (50 points), and concatenate trials together trajectory-wise. While this is by far not a perfect solution, it allows the data to remain roughly smooth at the cost of continuity in time and residual (small) jump discontinuities, as well as isolating the data to only when stimulus presentations are occurring. See figure A.1 for a example of trial-to-trial discontinuities created by this interim step.

3.5 Python Version and Hardware Utilized

Computations were made on the Edinburgh University compute server utilizing 128 GB of system memory, an Intel Xeon Gold 6142 processor, an NVidia 2080, and an NVidia 2080Ti. Plots were made on a laptop with 8 GB of system memory and an Apple M2 processor. The python version utilized for all computations and plots made was 3.10.6;

Elephant 1.0.0 and *Copula-GP* 0.0.5 were utilized for computations, and *Matplotlib* 3.6.1 and *Seaborn* 0.10.0 were utilized for plot production.

Chapter 4

Data Providence

In this chapter, we briefly go over data background, acquisition, initial exploration, and any cleaning and preprocessing regimes. The two datasets utilized are an *in silicon* dataset extracted from a Lorenz system [40] and an *in vivo* dataset extracted from the visual cortex of live mice.

4.1 Toy *in silicon* data-set: Simulated Data for Model Validation

One of the primary methods of generating neurological data is simulating it using statistical methods that generate a random multivariate distribution of which we can draw sample data from. For example, neuron spike data was simulated in the core paper via a GLM and a exponential non-linearity Poisson emission model [36]. For this paper, we utilize simulated spike data derived from a *Lorenz system*. The Lorenz system, first discovered by Edward Lorenz in meteorological systems [40], is a standard chaotic dynamical system utilized for the production of synthetic neuronal spike data in the past [12, 33]. The chaotic yet deterministic nature of the system makes this method of spike data simulation an interesting edge case for model validation in this project.

In mathematical terms, a Lorenz system is essentially a 3-dimensional gradient system $\nabla F = [\partial x, \partial y, \partial z]^T$ parameterized in some values σ, ρ, β:

$$\nabla F(x,y,z) = \begin{bmatrix} \partial x = & \sigma(y-x) \\ \partial y = & x(\rho - z) - y \\ \partial z = & xy - \beta z \end{bmatrix}. \tag{4.1}$$

To model the system as latent trajectory data, we simply get a uniformly distributed sample of a solution to the system: start with an initial point $p_0 = (x_0, y_0, z_0)$ and iteratively find the next point at some time-step t in the future as

$$p_t = p_{t-1} + \Delta_t \nabla F_{\sigma,\rho,\beta}(p_{t-1}), \tag{4.2}$$

where Δ_t is a parameter for scaling in time [12, 33, 19]. We then project the points found onto the desired neuronal dimensionality via a projection matrix with added

gaussian noise. We can see the outcome of such a process alongside the true latent variables for a solution to this system in figure 5.1a. The motivation behind this method of data simulation is as a demonstrative toy dataset to explore with *Copula-GPFA*, as well as a soft validation method for *Copula-GPFA*: *Copula-GPFA* should find near zero information interaction between the system and time-step t using a C-vine fit onto the system parameterized in t; there should be 0 entropy in the system outside of projection noise (see section 4.1.1) as a Lorenz system's outcome given initial parameters is deterministic.

4.1.1 Production of Data

We utilized the method of spike simulation found in the *Elephant* package's tutorial for GPFA (found at `https://elephant.readthedocs.io/en/latest/tutorials/gpfa.html`), which was accomplished by modelling a random projection of a Lorenz system to a highly-dimensional space to represent the instantaneous spike-rates of 50 neurons over 30 seconds. Afterwards, we then extracted spike events via application of a Poisson process [38], utilizing code within the *Elephants* package and above tutorial. Picking parameters $\sigma = 5$, $\rho = 34$, $\rho = 1.77$, $p_0 = (0, 1, 1.25)$, $\Delta_t = 1$ ms for our system, we produce 40000 points and use the last 30000 to represent true neuronal trajectory values over 30 seconds at a temporal resolution of 1ms with dimensionality $n = 3$ (we call the 10000 entries not utilized the *transient period*). We stores these values in a matrix Θ and project these values onto a $m = 50$ dimensional space via a random projection matrix Ψ

$$\Theta \to \Psi \cdot \Theta, \Psi = \{m_{i,j} \sim \mathcal{N}(0, \frac{1}{3})\}. \tag{4.3}$$

Let this matrix product be Θ'. We then normalized Θ' by dividing it by it's max value, and get neuronal spike-rates via mapping each element of Θ' to some max neuronal spike-rate (we use 70Hz in data production) and multiplying together values of the resulting matrix column wise, producing an instantaneous spike-rate. We then got the average spike-rate over a 10ms interval via integration to produce neuronal spike-rates for 10ms time buckets over the full 30s. We can then produce the neuronal spike data at 1ms resolution via a Poisson process. The original system, it's projection, and resulting spike-rates of a single trial can be seen in figure 4.1c. We replicated this process 20 times, producing 20 trials with uniform duration of 30s. This produces a large volume of data (600000 entries; 30000 entries per trial), which is ideal for fitting both *Copula-GP* estimators and GPFA.

4.2 Experimental *in vivo* data-set: Neuropixels Dataset

While *in silicon* data is useful for reproducability, validation on *in silicon* data will never be as applicable to the real world as validation on *in vivo* data. As such, the main data set we wished to explore is the *Visual Coding: Neuropixels* dataset, which is publicly available via the `allensdk` python package (visit `https://portal.brain-map.org/` and navigate to "circuits and behavior", then "Neuropixels"). This dataset contains spike-signals recorded from the visual cortex of live mice at single neuron spatial resolution utilizing novel Neuropixel probes, a high-fidelity and -resolution brain probe

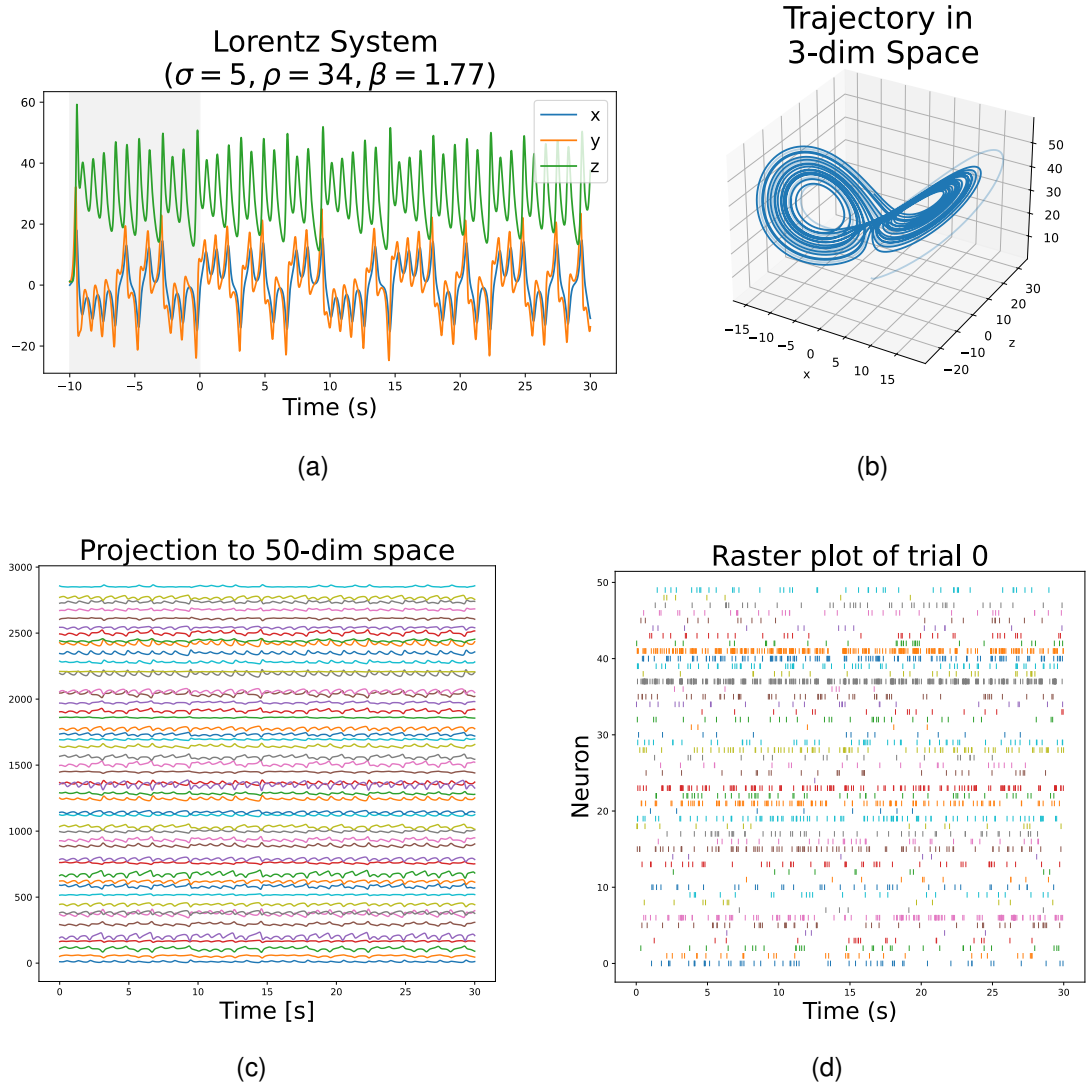


Figure 4.1: *In silicon* data generated from Lorenz system. Figure 4.1a shows the Lorenz system solution utilized for spike trial creation, with an area greyed out to represent the transient (unused) initial part of the system. Figure 4.1b is the system as represented in 3-dimensional space. We utilize these representations of the lorenz system as references for our averaged GPFA trajectories in figure 5.1. The system is projected onto a 50-dimensional space in 4.1c to represent waveforms of instantaneous spike rates, which are then utilized in production of spike trials, like that of figure 4.1d.

developed in 2021 [65]. Due to the volume of *in vivo* data recorded, as well as the quality metrics included in the data, it is an invaluable tool for analysis of underlying processes within the visual cortex. All technical claims surrounding the data in question is sourced from the technical white paper (found through the URL above, or through this hyperlink). For more information surrounding data production and specificities not covered in the body of this paper, i.e specifics on how probes were used and how visual stimuli were produced, one should see the technical white paper.

The spike data was collected via insertion of *Neuropixel* probes into the visual cortex of live mice. Probes record spike signals from points in space at single neuron resolution. These points are called "units," and are where neurons *might* be. Units have quality metrics attached, which can be used to filter for units of better quality (i.e. units with less noise, units which more certainly record spikes from only one neuron, etc). See section 4.2.1 for specifics on data retrieval and cleaning.

Pupil dilation has been found to have close ties to processes within the visual cortex [22, 7, 37]. As this data is directly from the visual cortex, we expected a significant statistical dependence between pupil dilation and visual cortex trajectories. As such, the motivation behind utilizing this dataset is as an experimental method of model validation; *Copula-GPFA* should be able to extract a significant amount of negative information interaction between trajectories extracted from the visual cortex and pupil dilation.

4.2.1 Data Retrieval and Cleaning

Visual Coding - Neuropixels is split into recording sessions, where during each session a mouse test subject was exposed to varying visual stimuli. Stimuli were presented in "blocks" of similar kinds of stimuli of varying duration. For this project, we examine specifically the first 100 stimuli presentations of the first "drifting gratings" block (block 2), as this way we mitigate variation in neuronal response due to differing stimulus length. In addition, within this block individual presentations are of uniform length (2 seconds of presentation, with a inter-presentation break of 1 second), and so the block is easily divisible into separate spike train trials (the GPFA implementation used is best suited for a trial-by-trial format; see section 5.2.1).

The session data contains the full spike recordings of all recorded units within the subject's visual cortex. As stated prior, these units possess quality metrics that correspond to how accurate and noisy unit recordings are. Of these, we utilize Signal to Noise Ratio (SNR) and Inter Spike Interval (ISI) Violation rate. SNR corresponds to the ratio of the maximum unit waveform amplitude to one standard deviation of the waveform, and acts as a metric of how noisy the unit recording is [6]. ISI violation rate is the percentage of unit spikes that occur during what should be the corresponding neuron's refractory period, and serves as a metric to determine whether multiple neurons and/or electronic interference are being picked up in a single unit's recordings [18].

Session data used in this project was pulled from a single session (session ID 756029989). By setting a lower bound for SNR and an upper bound for ISI violation rate, we are able to filter overly noisy and unreliable units' spike recordings out of the data set. Utilizing

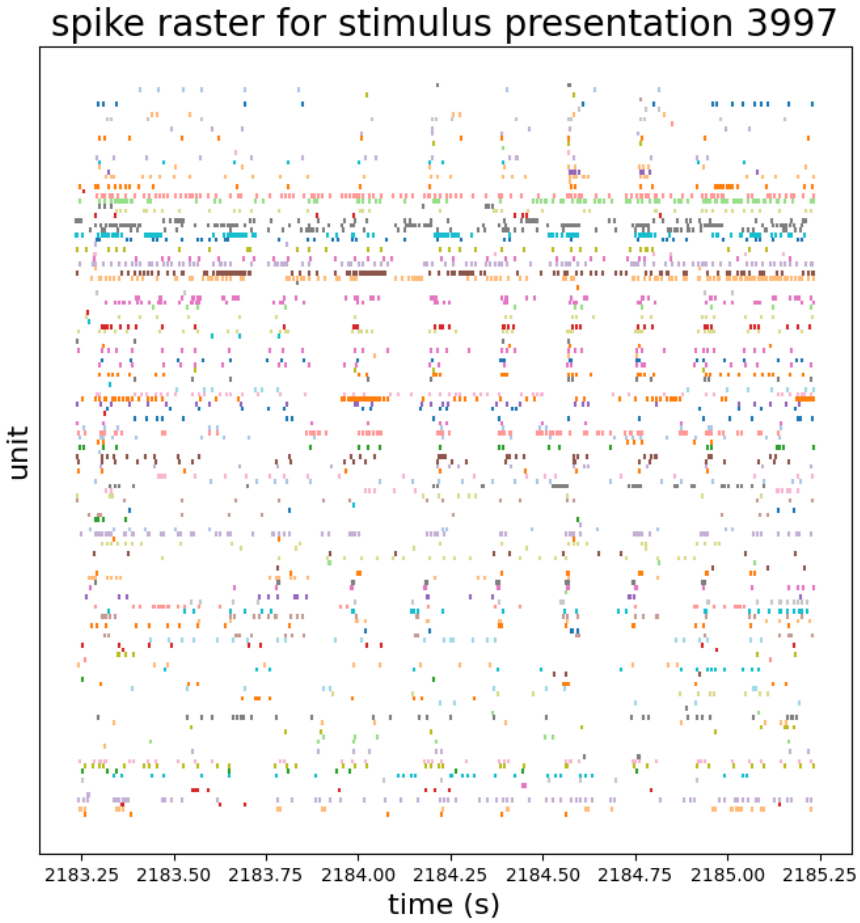


Figure 4.2: Spike raster of a single drifting gratings stimulus presentation (duration of 2s), from the session utilized in model validation (session ID 756029989). Only those spikes with SNR lower bound of 3 and ISI violation rate upper bound of 0.05 are shown. Note correlation in spike events present in the raster plot, as well as heightened neuronal spike-rate at the beginning of stimulus presentation.

a SNR lower bound of 3 and an ISI violation rate upper bound of 0.05 (5%), we found that in this session 26.0% of units meet these quality thresholds (178 out of the original total of 684). Example spike data for a single stimulus presentation of 2 seconds is shown in figure 4.2.

We also utilized pupil area recordings (cm^2) to parameterize the copulas in pupil dilation. Measurements were recorded every 33ms and possess a relatively large range, with sporadic large spikes in pupil dilation (see pupil dilation curves in figure 4.3*a*). As such, we applied a rolling mean with a window of 10 entries for smoothing followed by a robust normalization procedure $\text{Robust}(X)$:

$$\text{Robust}(X) = \frac{X - Q_1(X)}{Q_3(X) - Q_1(X)}, \tag{4.4}$$

where $Q_1(X), Q_3(X)$ are the 1st and 3rd quartile values of X. Doing so, we reduce the impact of outliers in the data. As the input for parameterizing values for fitting a C-vine

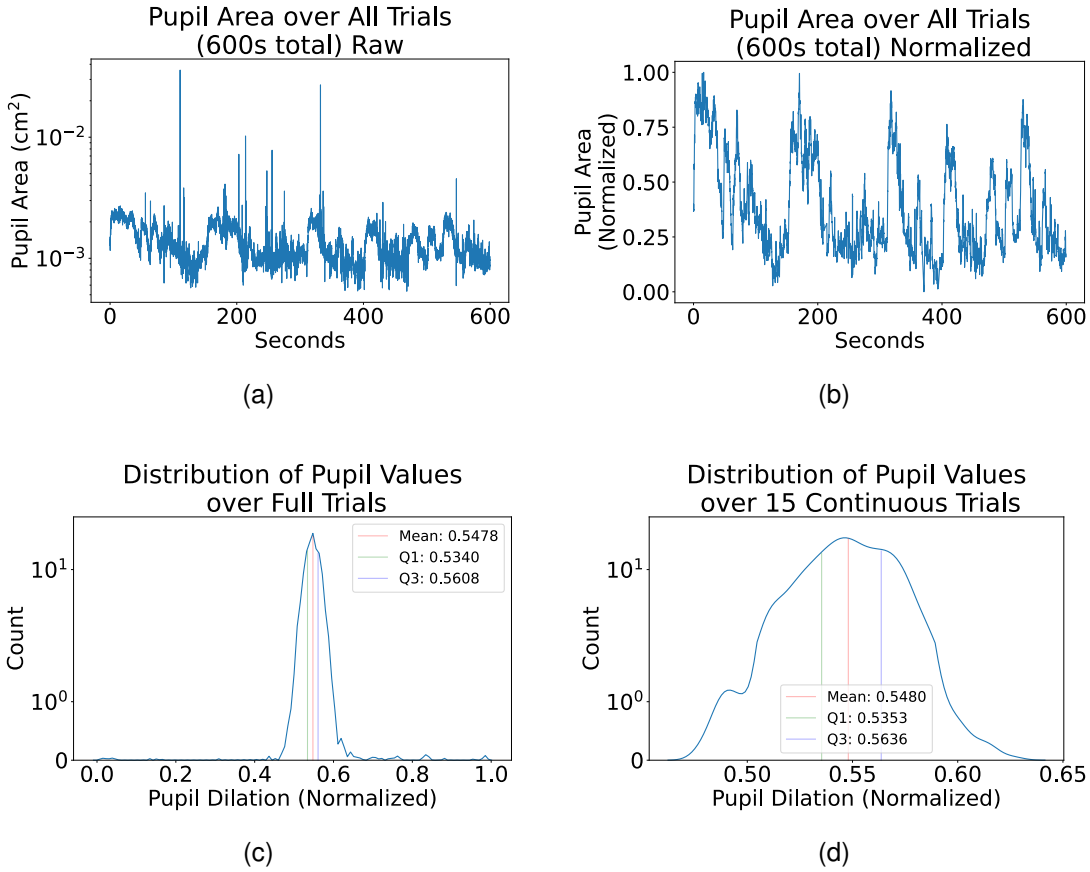


Figure 4.3: Figures 4.3a and 4.3b represent pupil dilation as function of time over all trials (duration 600s), raw (4.3a, pupil area in cm^2 scaled logistically) and processed (4.3b, pupil area smoothed and normalized). Processing consisted of rolling mean (window of 10 entries), a robust scaling, and a min-max scaling. Note the removal of strong outliers through processing, as well as the large range necessitating a logistic scale in values present in the raw recordings. Figure 4.3c represents the distribution over all pupil dilation data utilized (100000 continuous samples / 100 trials). Figure 4.3d represents the distribution over 1500 continuous samples (15 trials). Note the use of logistic scale in figures 4.3a, 4.3c, and 4.3d.

via *Copula-GP* must be the interval $[0,1]$, we map onto via an additional min-max normalization

$$\text{MinMax}(X) = \frac{X - \min(X)}{\max(X) - \min(X)}. \tag{4.5}$$

We can observe the difference between the raw and processed data in figure 4.3. See figure 4.3c for the distribution this preprocessing regime created, which appears roughly normal with few outliers.

Chapter 5

Baseline *Copula-GPFA* Validation, Experiments, and Results

5.1 Exploration of *In Silicon* Lorenz system Data

We first explored the use of *Copula-GPFA* on the known noisy deterministic simulated data-set extracted from a Lorenz attractor. In this section, we aim to visualize the GPFA process and utilize *Copula-GPFA* to confirm a near-zero information interaction between the Lorenz dynamics and time.

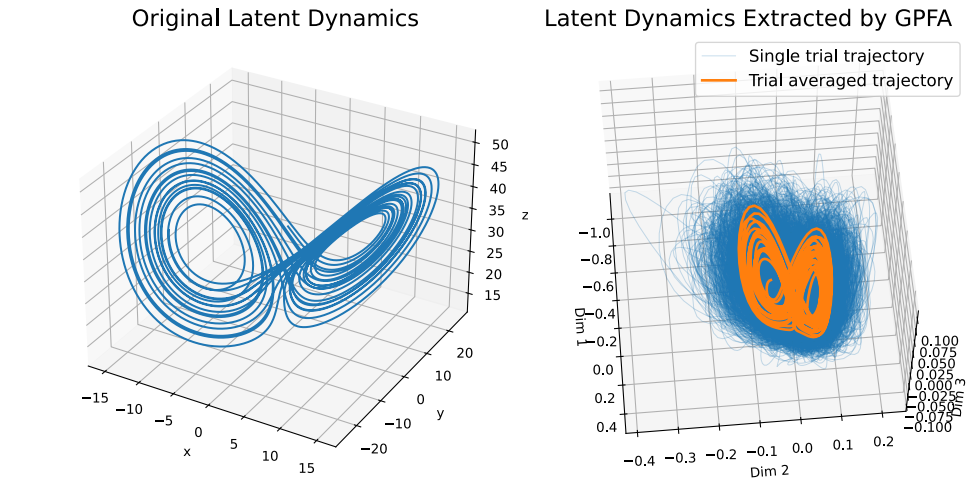
5.1.1 Extraction of Latent Dynamics via GPFA

We expected the true dynamics' to be sufficiently reconstructed visually via *Elephant* GPFA, which is straightforward to use. We created a GPFA instance with the dimensionality 3 and time buckets of 10ms, fitting the instance on and transforming the simulated neuronal spike train trials. The estimated dynamics can be compared to the true dynamics in figure 5.1. In figure 5.1b we can observe that GPFA extracted dynamics were re-centered on 0, with none of the extracted trajectories corresponding directly to the original dynamics. Despite this, from figure 5.1a we can see by comparing mean extracted dynamics to true dynamics that GPFA successfully captured the shape of the true Lorenz system. With the original trajectories having been sufficiently reconstructed, we could then utilize *Copula-GP* to analyze their dependencies.

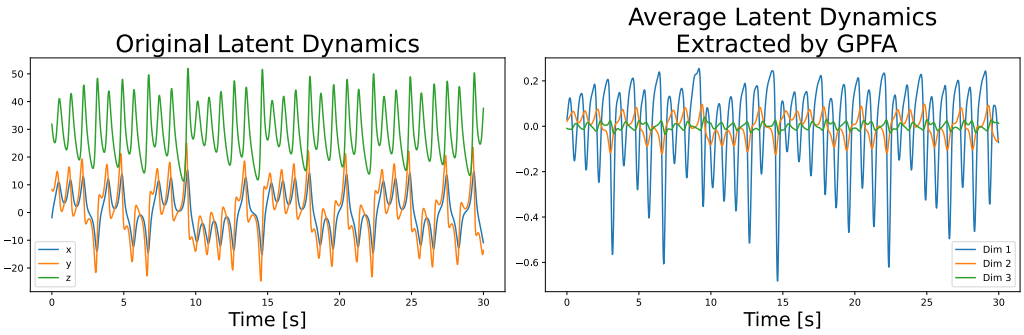
5.1.2 *Copula-GP* fit

We first utilized the additional interim processing described in section 3.4.4, cropping the final 1 second of trials and concatenating them. We then fit a C-vine estimator onto the trajectories parameterized in time, with the C-vine possessing 2 layers (3 copulas total). We calculated estimations of copula entropy $H_C(X)$ and conditional copula entropy $H_C(X|Y)$ via the estimation methods outlined in section 3.2.3. We found a mean negative copula entropy $-H_C(X)$ of 4.3622 and a mean negative conditional copula entropy $-H_C(X|Y)$ of 4.3698, yielding a near-zero information interaction of $I(X \leftarrow Y) \approx 0.0076$ (the slight deviation off zero can be attributed to gaussian noise

during data creation; see 5.1). In other-words, *Copula-GPFA* correctly identified little-to-no dependency between time bucket t and the extracted trajectories. With *Copula-GPFA* having passed initial validation, we moved on to utilization on the the *in vivo* experimental dataset.



(a)



(b)

Figure 5.1: Lorenz derived dynamics, both actual and estimated via GPFA. True trajectories were simulated with Lorenz system parameters $\sigma = 5$, $\rho = 34$, $\rho = 1.77$, $p_0 = (0, 1, 1.25)$, $\Delta_t = 1$ ms, and consists of a total of 30000 points. GPFA utilized time buckets of 20ms, and estimated trajectories encompass 1500 data points. Figure 5.1a are dynamics are spatial coordinates. Figure 5.1b are dynamics as functions through time. The left halves of each figure are true dynamics, while the right halves are averaged estimated dynamics, with single trial estimates shown in 5.1a. Note the close approximation of the shapes of the dynamics in 5.1a, as well as the recentering of dynamics on 0 and the difference scale and directionality present in 5.1b. The conclusion from this is that GPFA does not estimate trajectories exactly, but instead estimates trajectories while preserving statistically significant qualities. Other intrinsics surrounding the true dynamics are encompassed in the estimated scoring matrix Ψ.

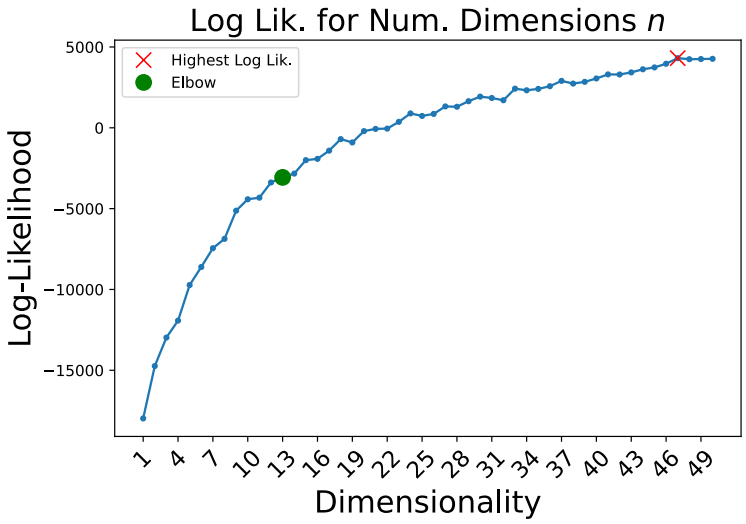


Figure 5.2: Log-likelihood curve for GPFA fit on trials as function of dimension n. Elbow and max log-likelihood found marked. Note that only dimension up to and including $n = 50$ were tested, and it is entirely possible log-likelihood to increase further in dimensions.

5.2 Experimental Exploration of *in vivo* Data: *Visual Coding - Neuropixels*

For this experiment, the objective was to utilize *Copula-GPFA* to confirm high statistical dependence between the visual cortex and pupil dilation. We hypothesised firmly negative information interaction, thereby effectively confirming that the visual cortex of the brain is linked to pupil dilation and agreeing with prior results in the literature [7, 23].

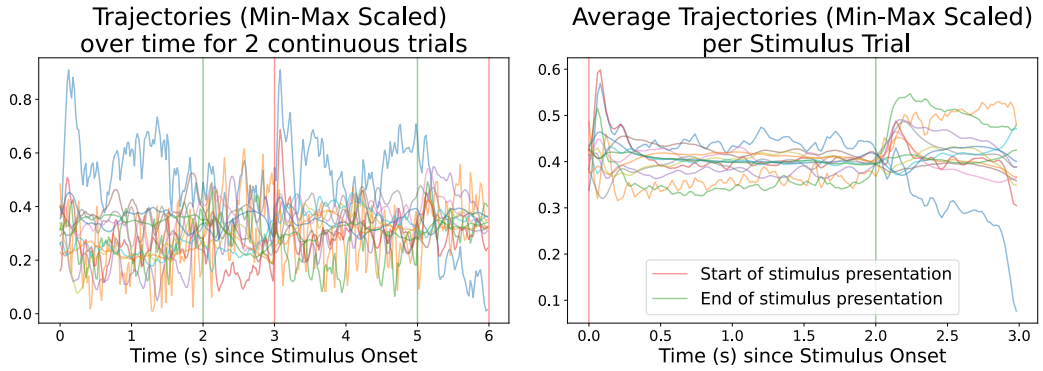
5.2.1 GPFA Application and Processing

GPFA was fit on 100 consecutive spike trains during Drifting Gratings stimulus presentations, each of which have a uniform temporal length of 2s followed by a 1s interval during which no stimulus is presented. We fit the model for the entirety of the dataset with the dimensionality of the elbow of the log-likelihood plot $n = 13$ (figure 5.2). Afterwards, we plot the trajectories for both a single trial and the averaged trial. We can observe from figure 5.3 a response in the trajectories immediately after both stimulus presentation start and stop, as well as clear vertical drift on a trial-by-trial basis. We removed the drift and concatenated the trials as per the process described in section 3.4.4, and moved on to *Copula-GP* application.

5.2.2 Application of *Copula-GP*

Application of *Copula-GP* involved training and model-selection methods described in section 3.2. We estimated a C-vine over the concatenation of GPFA-treated trials possessing 12 layers (78 copulas total) with parameterization in normalized pupil dila-

tion. We can observe in figure A.2 examples of singular bivariate copula estimations within the vine (low-level). We extracted the entropy of the system of trajectories via the methods outlined in section 3.2.3. In figure 5.4 we can see pupil dilation values y, corresponding parametric copula entropy $H_C(X|y)$, and the corresponding parametric information interaction $I(X \leftarrow y)$ plotted as functions of time-bucket t. Copula entropy appears to be highly correlated with pupil dilation in the figure, possessing a almost linear relationship. In addition, the information interaction stays firmly *negative*, implying the multivariate distribution of neuronal trajectories possesses statistical dependence on pupil dilation (thereby confirming our hypothesis). We found a mean negative copula entropy $-H_C(X)$ estimation of 13.4745 bits (95% CI of 3.0720) and a mean negative conditional copula entropy $-H_C(X|Y)$ empirical estimation of 7.2221 bits (95% CI of 6.1190), with X being the neuronal trajectories and Y being pupil dilation.



(a) (b)

Figure 5.3: Trajectories extracted via GPFA (dimension $n = 13$) representing dynamics driving recorded neuronal activity. Figure 5.3a are two trials' trajectory data as functions in time that have been concatenated, giving the appearance of continuity. Figure 5.3b is the mean trial. Note the clear vertical drift in mean and single trial trajectories post-stimulus presentation stop. Data is scaled to the range [0.01,0.99] to match *Copula-GP*'s input range of (0,1). Lines are transluscent for visibility purposes.

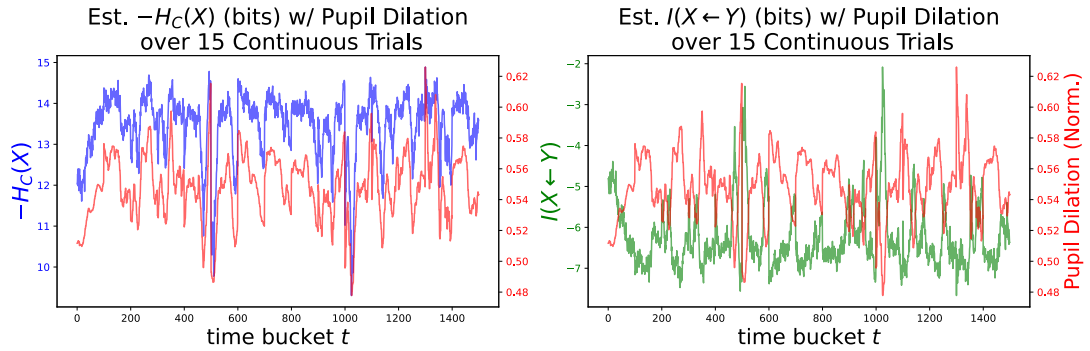


Figure 5.4: Pupil dilation (in red), estimated copula entropy of neuronal trajectories parametric in pupil dilation (in blue), and estimated corresponding information interaction (in green) through time. Copula entropy was estimated via estimating a distribution via *Copula-GP* and methods described in A.2.

Chapter 6

Bagging Validation and Experiments

In this section, we validated various *Copula-GP* bagging aggregation methods alongside baseline *Copula-GP* on randomly generated copulas; we validated copula entropy and mixture selection accuracy on data sampled from a simple bivariate copula, a highly mixed bivariate copula, and the simple bivariate copula parameterized in time. We then utilized bagging aggregation weighted dynamically by copula BIC on the experimental data-set to find a new estimation for the conditional entropy of the system of neuronal trajectories.

6.1 Initial Copula Entropy Accuracy Validation Tests

To validate our bagging methodology, we tested various *Copula-GP* aggregation methods alongside baseline unbagged *Copula-GP* on generated bivariate copulas (the "true" copula). Unfortunately, in depth validation on C-vines with increasing dimensionality could not be completed for paper submission, and as such robustness against dimensions for now goes unexamined. However, we note that C-vines are made up of bivariate copula building blocks, with C-vine model selection essentially being made up of consecutive bivariate copula selections. As such, validation alongside baseline *Copula-GP* on bivariate copulas alone might be indicative of possible improvements in C-vine entropy estimation via bagging methods described, however without aforementioned tests such conclusions cannot be confirmed.

For all validation tests in this section, the task was to accurately replicate the true copula's entropy and dependency shape. For validation, the weighted aggregation methods utilized were

- Copulas weighted point-wise dynamically by BIC / AIC on input,
- Copulas weighted statically by BIC / AIC on input,
- Average of copulas with close-to-best explained variance score R^2.

Computational metrics used for validation were:

- Copula entropy root point-wise mean squared error (RMSE) of baseline and bagging methods.
- Absolute error of mean (AEM) in copula entropy of baseline and bagging methods.
- Visual examination of baseline, BIC dynamically bagged, and true copulas.

To generate the data, we drew 10000 samples from the true copula, splitting samples and their corresponding parameterizations into train and test sets (80:20 split). Validation tests 1 and 2 utilized parameterization on a normally distributed variable $x \sim \mathcal{N}(0.5, 0.2)$, restricted to the interval $[0, 1]$. Validation test 3 utilized a parameterization in a time-based variable t scaling linearly from 0 to 1. Copula training and model selection utilized train set samples, with accuracy validation utilizing metrics described on test set samples. All validations utilized the heuristic algorithm for individual estimator model selection, and for bagged estimations 4 estimators underwent individual training and model selection routines. We include random seed, module version, and hardware information for test reproducability purposes in the appendix (see A.1).

6.1.1 Validation 1: Low-Entropy 4-Copula Mixture Copula

Table 6.1: True and predicted mean copula entropies $\overline{H}_C$ with 95% CI of validation test on low-entropy 4-copula mixture copula with parameterization in random variable $x \sim \mathcal{N}(0.5, 0.2)$ and resemblance to independence copula. Average Error in Mean (AEM) and Root Meas Square point-wise Error (RMSE) of predicted copula entropies included. Closest to actual / best scores in bold.

Model	$\overline{H}_C$	95% CI	AEM	RMSE
True Copula	-0.1440	$\pm$0.0154	-	-
Baseline	-0.0000	$\pm$0.0000	0.1440	0.1442
BIC Dynamic	**-0.0802**	**$\pm$0.0147**	**0.0638**	**0.0647**
BIC Static	-0.0791	$\pm$0.0117	0.0649	0.0656
AIC Dynamic	-0.0796	$\pm$0.0128	0.0644	0.0652
AIC Static	-0.0790	$\pm$0.0123	0.0650	0.0658
R2 Meaned	-0.0798	$\pm$0.0133	0.0642	0.0651

Our first validation test was on a low entropy copula with a resemblance to the independence copula we expected our bagging algorithm to perform well on. Baseline *Copula-GP* when facing such a copula can often "give up" early into heuristic model selection and select independence if it finds non-independence copulas possess too low WAIC. We expect that bagging is able to make up for this flaw in baseline heuristic model selection via it's bias-variance trade-off, and thus pick up tail dependencies in the model.

We found that our expectations were held. In table 6.1 we observe that bagging methods all performed better than baseline (which selected independence), with BIC dynamic bagging possessing the best model accuracy and capturing much of the variance in the true copula (i.e. the most accurate 95% CI metric). Finally, in figure 6.1 we find that

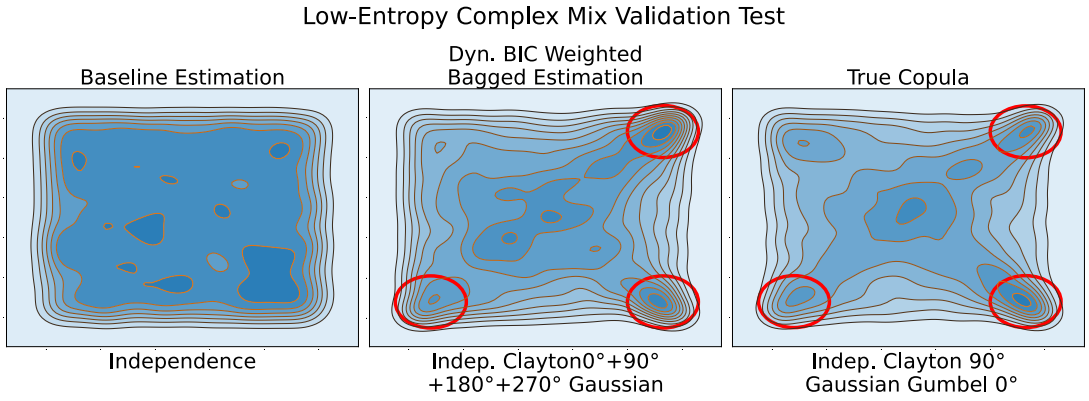


Figure 6.1: Select results of low entropy copula validation. Note how the distribution "tricks" the baseline estimator into picking the independence copula, but the bagged estimate is able to catch major tail dependencies in the true distribution (circled in red).

the BIC dynamically bagged estimated copula captures major tail distributions in the true copula, whereas the baseline estimation of independence does not.

6.1.2 Validation 2: High-Entropy 3-Copula Mixture Copula

Table 6.2: True and predicted mean copula entropies $\overline{H}_C$ with 95% CI of validation test on high-entropy 3-copula mixture copula, with parameterization in random variable $x \sim \mathcal{N}(0.5, 0.2)$. Average Error in Mean (AEM) and Root Mean Square point-wise Error (RMSE) of predicted copula entropies included. Closest to actual / best scores in bold.

Model	$\overline{H}_C$	95% CI	AEM	RMSE
True Copula	-0.6864	±0.1148	-	-
Baseline	-0.6680	±0.0688	0.0184	0.0694
BIC Dynamic	**-0.6764**	±0.0847	**0.0100**	0.0719
BIC Static	-0.6725	±0.0793	0.0139	0.0713
AIC Dynamic	-0.6664	±0.0870	0.0200	0.0748
AIC Static	-0.6662	±**0.0921**	0.0202	0.0658
R2 Meaned	-0.6738	±0.0778	0.0126	**0.0651**

For this validation test, we expected baseline to perform well and are interested in if the bias induced by bagging can result in worse performance on distributions the baseline is well suited for. The true copula here is a high-entropy copula with simple shape, something that in the core paper [36] was found to be well suited to baseline *Copula-GP* as a method.

We find from table 6.2 that all tested methods had similar performance. The BIC dynamically weighted estimate had the best AEM, but the R2 meaned estimate possessed better point-wise RMSE. In addition, we find that AIC weighted aggregation methods had worse AEM than baseline. In figure 6.2 we find that the BIC dynamically weighted estimation included a gaussian copula and as such makes incorrect assumption surrounding the dependency shape of the true copula. Despite this, the majority bagged copula estimates still result in similar if not slightly better performance in copula entropy prediction.

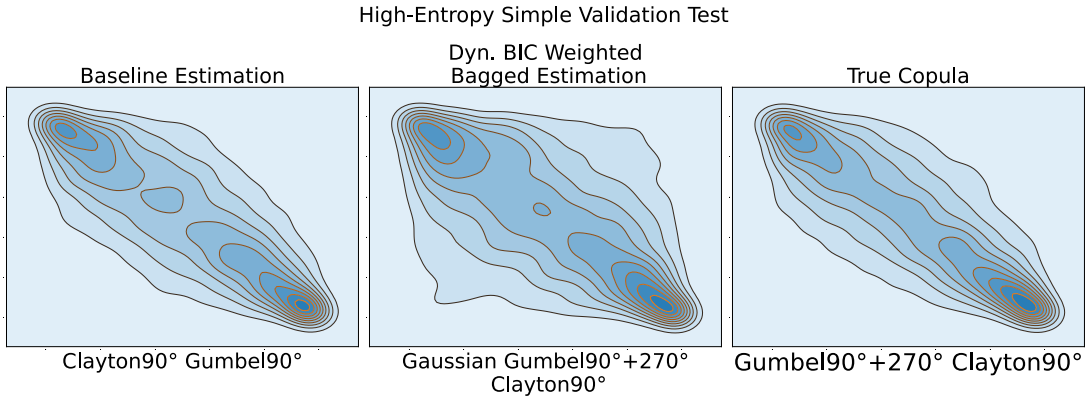


Figure 6.2: Select results of high entropy copula validation. We find that bagging methods can result in incorrect selection of copula variant leading to misrepresentation of dependency shape.

6.1.3 Validation 3: High-Entropy 3-Copula Mixture Copula, Parameterized Linearly in Time

Table 6.3: True and predicted mean copula entropies $\overline{H}_C$ with 95% CI of validation test on high-entropy 3-copula mixture copula, with time based parameterization in variable increasing linearly from $0 \to 1$. Average Error in Mean (AEM) and Root Meas Square point-wise Error (RMSE) of predicted copula entropies included. Closest to actual / best scores in bold.

Model	$\overline{H}_C$	95% CI	AEM	RMSE
True Copula	-0.6864	$\pm$0.1148	-	-
Baseline	-0.4556	$\pm$0.2038	0.2308	0.2587
BIC Dynamic	**-0.5628**	$\pm$0.0754	**0.1236**	**0.1388**
BIC Static	-0.5589	$\pm$0.0650	0.1275	0.1418
AIC Dynamic	-0.5382	$\pm$**0.0813**	0.1482	0.1615
AIC Static	-0.5389	$\pm$0.0703	0.1475	0.1596
R2 Meaned	-0.5580	$\pm$0.0722	0.1284	0.1420

Our final validation was on a copula parameterized in time. In the core paper, it was found that "transformed" cases with dependency shape changing as a function in time can be more challenging for *Copula-GP* to predict the entropy of accurately. As such, we utilize the copula in validation test 2 and change it's parameterization to be in time t scaling linearly from 0 to 1. We maintain continuity when splitting into train and test set, and as such the train set consists of parameterization in $t = 0.0$ to 0.8, and the test set consists of parameterization in $t = 0.8$ to 1.0. We hope that bagging distribution estimations made by estimators trained on continuous subsets of training data can induce some robustness to time-based transformation in true dependency shape.

Our hopes were confirmed, as we find in table 6.3 that bagging methods all find better accuracy than baseline. BIC dynamically weighted bagging results in the best AEM and RMSE (with a 0.1072 decrease in AEM over baseline), while capturing most of the variance in true copula entropy. From figure 6.3 we see that the baseline estimated copula possesses much less of a resemblance to the true copula.

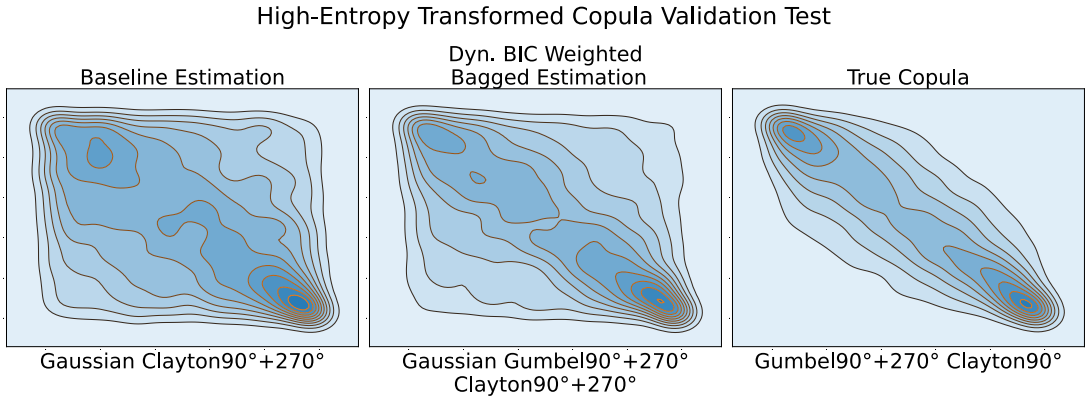


Figure 6.3: Select results of high entropy transformed copula validation. We find that the bagged estimation possesses much less of a resemblance to the true copula than the bagged estimation.

6.1.4 Validation Discussion

In general, we found that bagged copula estimations utilizing 4 estimators possesses similar or better accuracy in copula entropy estimation than baseline. The best bagging method appears to be BIC dynamically aggregated copulas, possessing the best AEM for all validation tests conducted. In addition, AIC weighted aggregation methods find worse performance than the BIC counterparts. Considering the difference between AIC from BIC is a more lenient penalty in number of parameters, this can be attributed to overfitting due to over-parameterization. From validation test 1, we see that bagged estimates can catch dependencies in distributions that the baseline perceives as independence. In C-vine model selection, as copulas gain more conditioning marginal variables' relationships tend to more closely resemble conditional independence [36]. As such, we may conclude that a bagged estimator might be more likely to pick up tail dependencies in conditioned relationships where the baseline estimator might assume independence. From validation tests 2 and 3 we find that the bagged estimator is more robust to transformations in dependency shape over time, and as such may yield better accuracy when predicting marginal variables' dependency shape for time-based data, such as spike train data. Finally, we note that the BIC dynamically bagged estimator does not over-estimate copula entropy in validation, much like was found of baseline *Copula-GP* in the core paper.

6.2 Application of BIC Dynamically Bagged *Copula-GPFA* on *in vivo* Experimental Data

Finally, we utilized the BIC dynamically weighted bagging method on the neuronal trajectories extracted from the experimental spike train data (described in section 5.2.1), and compare it's negative copula entropy $-H_C(X)$ over time with that of baseline *Copula-GP* over 15 continuous trials. We expected either similar or lower (more negative) copula entropy estimates, as was observed in validation.

We can see the outcome of $-H_C(X)$ and interaction information $I(X \leftarrow Y)$ calculations

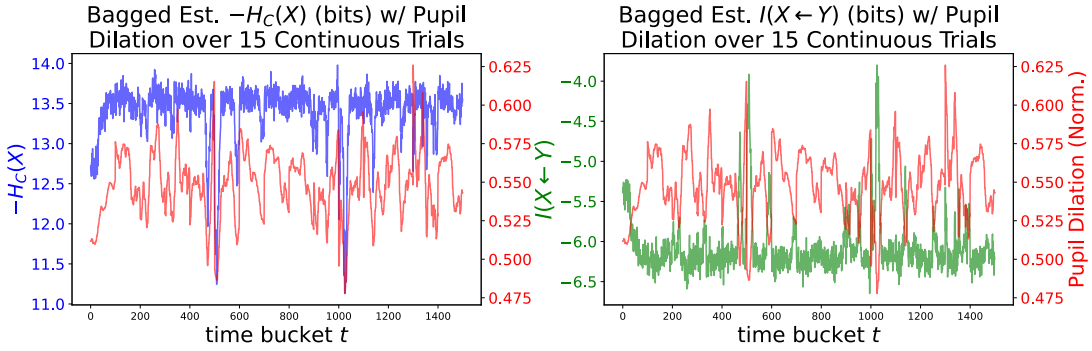


Figure 6.4: Pupil dilation (in red), copula entropy of neuronal trajectories parametric in pupil dilation utilizing bagged estimation (in blue), and corresponding information interaction (in green) through time utilizing bagged estimated. Copula entropy was estimated via estimating a distribution with BIC dynamically bagged *Copula-GP*.

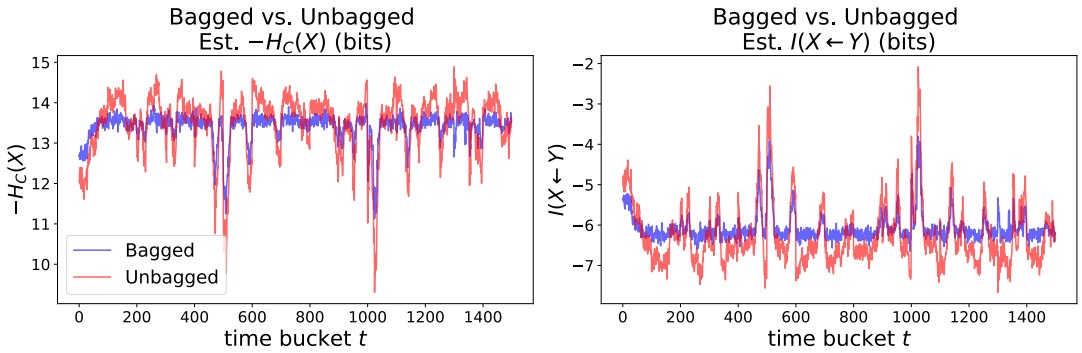


Figure 6.5: BIC dynamically bagged vs unbagged *Copula-GP* estimated entropies and information interactions. The BIC dynamically bagged entropies curve seems to hover steadily around a mean, and does not appear to be as volatile as unbagged entropies.

alongside normalized pupil dilation in figure 6.4, and comparison of $-H_C(X)$ calculations in figure 6.5. The bagged estimate found a negative mean copula entropy $-H_C(X)$ of 13.4027 bits (95% CI of 0.7069; only 0.0718 bit difference from unbagged estimate), and a mean negative conditional copula entropy $-H_C(X|Y)$ of 7.3323 bits (95% CI of 5.9693). From figures 6.4 and 6.5, we can see that like the original baseline estimate the bagged negative copula entropy appears to be dependent on pupil dilation, with large dips occurring with pupil dilation. However, we find the bagged estimate is less sensitive to small changes in pupil dilation than the baseline estimate. In addition, the bagged entropy estimate appears to steadily hover around the mean negative entropy found for most observed time-buckets, but will still dip severely at the same time as the baseline estimate. These differences aside, the bagged and baseline mean copula entropy and information interaction estimates were quite close, implying that the bagged estimator does not inherently out- or under perform baseline unbagged *Copula-GPFA* in the case of the *in vivo* data.

Chapter 7

Conclusions

In this chapter, we first go over the contributions to project goals (most of which were fully completed). Afterwards, we discuss future extensions to this paper, as well as remark on possible uses of *Copula-GPFA* and the bagging extension to *Copula-GP* in future work.

7.1 Contribution Overview

7.1.1 Validation of *Copula-GPFA*

In chapter 5, we validated *Copula-GPFA* as a mutual information and information interaction extraction method on *in silicon* data extracted from a Lorenz attractor, and confirmed significantly negative information interaction between neuronal trajectories and pupil dilation, agreeing with prior findings linking the visual cortex with pupil dilation [7, 23, 73]. In addition, *Copula-GPFA* also allows for easy interpretability of findings, allowing for visualization of both neuronal trajectories and information quantities over time. We also justify a significant reduction in computational complexity via GPFA utilized with *Copula-GP*, using complexity estimates made in the core paper. In other words, ***Copula-GPFA* is able to efficiently provide accurate and meaningful dependency and information quantification analysis and results.** As *Copula-GPFA* is not inherently a method restricted to the area of neuroscience, it is entirely plausible *Copula-GPFA* is able to be meaningfully applied to other fields where dependency analysis is useful (i.e. bio-informatics or quantitative finance). Finally, we note that due to paper length restriction we did not thoroughly investigate individual bivariate copulas within the C-vines estimated. However, *each bivariate copula itself represents a dependency relationship*, and as such there is room for even more analysis to glean from C-vines estimated.

7.1.2 Extension of *Copula-GP* via Addition of Bagging Capability

In chapter 3, we introduced copula bagging as a means to possibly improve copula estimation, complete with both formalistic and practical justifications, and implemented bagging as an extension to the *Copula-GP* code-base (locally). In chapter 6, **we**

identified the implementation of bagged *Copula-GP*'s estimations' superiority over baseline in tail-dependency identification and estimated copula entropy accuracy in the bivariate case. In addition, we utilized the bagged extension on the experimental *in vivo* data, finding results in agreement with baseline estimation of mean copula entropy with more stability over time, however whether this stability is indicative of the true distribution or not goes unconfirmed. While our validation is flawed in that we did *not* test robustness of the bagged extension to higher dimensionality due to time and workload constraints, we did find similar results to baseline when utilized on the *in vivo* data with 13 dimensions. We also noted C-vine selection in *Copula-GP* essentially consists of individual selections of bivariate building blocks, and as such we believe that benefits found in the bivariate case may carry over into C-vines of higher dimensions. Our implementation of bagging also follows in the footsteps of the formalistic approach of the original *Copula-GP* implementation, and as such maintains flexibility in use cases outside the realm of spike train data and neuroscience. We also acknowledge that the ability to bag copula estimations allows for the aggregation of *Copula-GP* estimators to be fit on individual trials of data, thereby resolving possible violations of local smoothness and continuity assumptions caused by concatenation of GPFA treated spike train trials (a low number of estimators was utilized in the core data due to time-to-run concerns, and so this goes unutilized as of now). Finally, we note our bagging implementation can be used *without* heuristic model selection, and one could utilize our implementation to bag singleton mixtures (mixture copulas of only a single variant) as an alternative efficient means of model selection.

7.2 Further Work and Extensions

7.2.1 Further Validation, Optimization, and Use of Bagging Methods

Our validations find that weighted copula estimate aggregation can yield effective improvements in copula tail dependency identification and copula entropy accuracy. The obvious extension to validations made would of course be to confirm if such improvements firmly carry over to higher dimensions. In addition, further refinement of aggregation methods may be considered, i.e. clustering methods such as k-means as a way to select copula estimates for datasets. Use of parallel processing in training bagged estimators concurrently might also be a natural optimization for bagged *Copula-GP*; one could even bag bivariate copulas immediately after they've completed model selection on their subsets of data during vine training.

That being said, the validation tests made are still entirely applicable. Bivariate copulas are still effectively used in fields from computational finance [15] to bio-informatics [56]. As such, the improvements made to *Copula-GP* in bivariate copula estimation via the addition of dynamically weighted aggregation methods may be used to robustify bivariate copula selection effectiveness in such use cases.

7.2.2 Possible Future Use of *Copula-GPFA*

Computational neuroscience as a field is in it's infancy; only recently have demands in recording techniques and computational power been met through technologies like neuropixel probes [65] and GPU-based computational acceleration [51, 25]. As such, *Copula-GPFA* may be used to answer questions that have been plaguing the minds of neuro-scientists for decades.

For example, dysfunction of the cerebellum (a section of the brain located on it's back) has been hypothesized in the past to be a central cause of multiple cognitive disorders, such as Alzheimer's, Frontotemporal Dementia, and cognitive effects of Multiple Sclerosis [60, 43]. A possible experiment to discern what parts of the brain could be effected by cerebellar dysfunction utilizing *Copula-GPFA* could involve recording of different parts of the brain, followed by separate GPFA application on those parts of the brain. Afterwards, *Copula-GP* might be utilized to investigate dependencies between cerebellar neuronal trajectories and trajectories extracted from other parts of the brain, as well as how these dependencies change in response to cerebellar dysfunction. Ethical considerations not withstanding, such experiments could yield insight on how cognitive diseases progress and just how parts of the brain are involved.

Alternatively, experiments may also be conducted to further confirm neurological functions; there have been sub-networks identified in the thalamus with a diverse range of functions [16]. Many of these function remain unknown, and as such utilization of neuropixel probes to gather single-neuron resolution spike recordings of these networks combined with *Copula-GPFA* to uncover statistical dependencies between subnetwork dynamics might uncover how these subnetworks interface with one-another to accomplish processes in the thalamus. Such experiments that could leverage newly available detail and accuracy in data collection and analysis are only the tip of the iceberg of what's possible via recently developed novel technologies.

That is not to say that *Copula-GPFA* must be restricted to the realm of spike trial data, or even computational neuroscience; As stated prior, copulas and dependency analysis have heavy utilization in many fields. For example, bivariate copulas have been used to model co-expression dependencies in gene pairings [56], as well as for analysis of risk dependency in insurance claims [30]. The number of use cases for efficient dependency and mutual information analytical frameworks is quite large, and as such we hope *Copula-GPFA* might be used in a wide range of fields.

Bibliography

[1] *[PDF] A widely applicable Bayesian information criterion — Semantic Scholar*. URL: https://www.semanticscholar.org/paper/A-widely-applicable-Bayesian-information-criterion-Watanabe/d9c9b74c8cbfa5475c8e91312efac956bf (visited on 10/25/2023).

[2] Mário Antunes, Diogo Gomes, and Rui L. Aguiar. "Knee/Elbow Estimation Based on First Derivative Threshold". In: *2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*. Mar. 2018, pp. 237–240. DOI: 10.1109/BigDataService.2018.00042. URL: https://ieeexplore.ieee.org/abstract/document/8405717 (visited on 03/23/2024).

[3] Bence Bagi, Michael Brecht, and Juan Ignacio Sanguinetti-Scheck. "Unsupervised discovery of behaviorally relevant brain states in rats playing hide-and-seek". en. In: *Current Biology* 32.12 (June 2022), 2640–2653.e4. ISSN: 09609822. DOI: 10.1016/j.cub.2022.04.068. URL: https://linkinghub.elsevier.com/retrieve/pii/S0960982222007047 (visited on 03/24/2024).

[4] Mohamed Ishmael Belghazi et al. *MINE: Mutual Information Neural Estimation*. arXiv:1801.04062 [cs, stat]. Aug. 2021. DOI: 10.48550/arXiv.1801.04062. URL: http://arxiv.org/abs/1801.04062 (visited on 03/29/2024).

[5] P. Berkes, F. Wood, and J. Pillow. "Characterizing neural dependencies with copula models". In: Dec. 2008. URL: https://www.semanticscholar.org/paper/Characterizing-neural-dependencies-with-copula-Berkes-Wood/41d56cacdecbf781ee7c219780f45ee948f71bf2 (visited on 10/25/2023).

[6] Gérard Biau and Erwan Scornet. "A random forest guided tour". en. In: *TEST* 25.2 (June 2016), pp. 197–227. ISSN: 1863-8260. DOI: 10.1007/s11749-016-0481-7. URL: https://doi.org/10.1007/s11749-016-0481-7 (visited on 03/21/2024).

[7] Klaas Bombeke et al. "Pupil size directly modulates the feedforward response in human primary visual cortex independently of attention". In: *NeuroImage* 127 (Feb. 2016), pp. 67–73. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2015.11.072. URL: https://www.sciencedirect.com/science/article/pii/S105381191501109X (visited on 03/22/2024).

[8] Leo Breiman. "Bagging predictors". en. In: *Machine Learning* 24.2 (Aug. 1996), pp. 123–140. ISSN: 1573-0565. DOI: 10.1007/BF00058655. URL: https://doi.org/10.1007/BF00058655 (visited on 03/24/2024).

[9] Leo Breiman. “Random Forests”. en. In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324. URL: https://doi.org/10.1023/A:1010933404324 (visited on 03/26/2024).

[10] Emery N. Brown, Robert E. Kass, and Partha P. Mitra. “Multiple neural spike train data analysis: state-of-the-art and future challenges”. en. In: *Nature Neuroscience* 7.5 (May 2004). Publisher: Nature Publishing Group, pp. 456–461. ISSN: 1546-1726. DOI: 10.1038/nn1228. URL: https://www.nature.com/articles/nn1228 (visited on 03/24/2024).

[11] R. S. Calsaverini and R. Vicente. “An information-theoretic approach to statistical dependence: Copula information”. en. In: *Europhysics Letters* 88.6 (Dec. 2009), p. 68003. ISSN: 0295-5075. DOI: 10.1209/0295-5075/88/68003. URL: https://dx.doi.org/10.1209/0295-5075/88/68003 (visited on 10/17/2023).

[12] Yin-Jui Chang et al. “Multi-scale dynamics modeling of spike and field potential activity via biologically realistic neural ordinary differential equations”. In: *2022 56th Asilomar Conference on Signals, Systems, and Computers*. ISSN: 2576-2303. Oct. 2022, pp. 613–617. DOI: 10.1109/IEEECONF56349.2022.10051855. URL: https://ieeexplore.ieee.org/abstract/document/10051855 (visited on 02/27/2024).

[13] Lin Chen et al. “Detection of bursts in neuronal spike trains by the mean inter-spike interval method”. In: *Progress in Natural Science* 19.2 (Feb. 2009), pp. 229–235. ISSN: 1002-0071. DOI: 10.1016/j.pnsc.2008.05.027. URL: https://www.sciencedirect.com/science/article/pii/S1002007108003432 (visited on 03/20/2024).

[14] Tao Chen and Jianghong Ren. “Bagging for Gaussian process regression”. In: *Neurocomputing*. Advances in Machine Learning and Computational Intelligence 72.7 (Mar. 2009), pp. 1605–1610. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2008.09.002. URL: https://www.sciencedirect.com/science/article/pii/S0925231208004396 (visited on 03/26/2024).

[15] Umberto Cherubini et al. *Dynamic Copula Methods in Finance*. en. Google-Books-ID: wZaADwAAQBAJ. John Wiley & Sons, Nov. 2011. ISBN: 978-0-470-68307-1.

[16] John W. Crabtree. “Functional Diversity of Thalamic Reticular Subnetworks”. English. In: *Frontiers in Systems Neuroscience* 12 (Oct. 2018). Publisher: Frontiers. ISSN: 1662-5137. DOI: 10.3389/fnsys.2018.00041. URL: https://www.frontiersin.org/articles/10.3389/fnsys.2018.00041 (visited on 04/03/2024).

[17] Claudia Czado and Thomas Nagler. “Vine Copula Based Modeling”. In: *Annual Review of Statistics and Its Application* 9.1 (2022). _eprint: https://doi.org/10.1146/annurev-statistics-040220-101153, pp. 453–477. DOI: 10.1146/annurev-statistics-040220-101153. URL: https://doi.org/10.1146/annurev-statistics-040220-101153 (visited on 10/19/2023).

[18] Gabriela Czanner et al. “Measuring the signal-to-noise ratio of a neuron”. In: *Proceedings of the National Academy of Sciences* 112.23 (June 2015). Publisher: Proceedings of the National Academy of Sciences, pp. 7141–7146. DOI: 10.1073/pnas.1505545112. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1505545112 (visited on 03/20/2024).

[19] Michael Denker and Moritz Kern. *Elephant 1.0.0*. Nov. 2023. DOI: 10.5281/zenodo.10107318. URL: https://zenodo.org/records/10107318 (visited on 03/21/2024).

[20] Daniel A. Dombeck et al. "Imaging large-scale neural activity with cellular resolution in awake, mobile mice". eng. In: *Neuron* 56.1 (Oct. 2007), pp. 43–57. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2007.08.003.

[21] R. M. Dudley. "Sample Functions of the Gaussian Process". en. In: *Selected Works of R.M. Dudley*. Ed. by Evarist Giné, Vladimir Koltchinskii, and Rimas Norvaisa. Selected Works in Probability and Statistics. New York, NY: Springer, 2010, pp. 187–224. ISBN: 978-1-4419-5821-1. DOI: 10.1007/978-1-4419-5821-1_13. URL: https://doi.org/10.1007/978-1-4419-5821-1_13 (visited on 10/23/2023).

[22] Katrin Franke et al. "State-dependent pupil dilation rapidly shifts visual feature selectivity". en. In: *Nature* 610.7930 (Oct. 2022). Publisher: Nature Publishing Group, pp. 128–134. ISSN: 1476-4687. DOI: 10.1038/s41586-022-05270-3. URL: https://www.nature.com/articles/s41586-022-05270-3 (visited on 03/22/2024).

[23] Dan Alin Ganea et al. "Pupillary Dilations of Mice Performing a Vibrotactile Discrimination Task Reflect Task Engagement and Response Confidence". English. In: *Frontiers in Behavioral Neuroscience* 14 (Sept. 2020). Publisher: Frontiers. ISSN: 1662-5153. DOI: 10.3389/fnbeh.2020.00159. URL: https://www.frontiersin.org/articles/10.3389/fnbeh.2020.00159 (visited on 03/23/2024).

[24] Weihao Gao, Sewoong Oh, and Pramod Viswanath. "Demystifying Fixed k - Nearest Neighbor Information Estimators". In: *IEEE Transactions on Information Theory* 64.8 (Aug. 2018). Conference Name: IEEE Transactions on Information Theory, pp. 5629–5661. ISSN: 1557-9654. DOI: 10.1109/TIT.2018.2807481. URL: https://ieeexplore.ieee.org/abstract/document/8294268 (visited on 03/29/2024).

[25] Jacob Gardner et al. "GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration". In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper/2018/hash/27e8e17134dd7083b050476733207ea1-Abstract.html (visited on 10/23/2023).

[26] Wulfram Gerstner et al. *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. en. Google-Books-ID: D4j2AwAAQBAJ. Cambridge University Press, July 2014. ISBN: 978-1-107-06083-8.

[27] Joshua I. Glaser et al. "The roles of supervised machine learning in systems neuroscience". In: *Progress in Neurobiology* 175 (Apr. 2019), pp. 126–137. ISSN: 0301-0082. DOI: 10.1016/j.pneurobio.2019.01.008. URL: https://www.sciencedirect.com/science/article/pii/S0301008218300856 (visited on 10/16/2023).

[28] José Miguel Hernández-Lobato, James R Lloyd, and Daniel Hernández-Lobato. "Gaussian Process Conditional Copulas with Applications to Financial Time Series". en. In: ().

[29] Caroline M. Holmes and Ilya Nemenman. “Estimation of mutual information for real-valued data with error bars and controlled bias”. In: *Physical Review E* 100.2 (Aug. 2019). Publisher: American Physical Society, p. 022404. DOI: 10.1103/PhysRevE.100.022404. URL: https://link.aps.org/doi/10.1103/PhysRevE.100.022404 (visited on 10/26/2023).

[30] Sen Hu and Adrian O’Hagan. *Copula Averaging for Tail Dependence in Insurance Claims Data*. arXiv:2103.10912 [stat]. Mar. 2021. DOI: 10.48550/arXiv.2103.10912. URL: http://arxiv.org/abs/2103.10912 (visited on 03/26/2024).

[31] Aisha Jangid, Laxmi Chaudhary, and Komal Sharma. “Computational Neuroscience and Its Applications: A Review”. en. In: *Intelligent Energy Management Technologies*. Ed. by Mohammad Shorif Uddin et al. Singapore: Springer, 2021, pp. 159–169. ISBN: 9789811588204. DOI: 10.1007/978-981-15-8820-4_16.

[32] Rick L. Jenison and Richard A. Reale. “The Shape of Neural Dependence”. en. In: *Neural Computation* 16.4 (Apr. 2004), pp. 665–672. ISSN: 0899-7667, 1530-888X. DOI: 10.1162/089976604322860659. URL: https://direct.mit.edu/neco/article/16/4/665-672/6825 (visited on 10/25/2023).

[33] Justin Jude et al. “Robust alignment of cross-session recordings of neural population activity by behaviour via unsupervised domain adaptation”. In: (2022). Publisher: arXiv Version Number: 2. DOI: 10.48550/ARXIV.2202.06159. URL: https://arxiv.org/abs/2202.06159 (visited on 02/27/2024).

[34] James J. Jun et al. “Fully integrated silicon probes for high-density recording of neural activity”. en. In: *Nature* 551.7679 (Nov. 2017), pp. 232–236. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature24636. URL: https://www.nature.com/articles/nature24636 (visited on 10/09/2023).

[35] Mario Koppen. “The curse of dimensionality”. en. In: ().

[36] Nina Kudryashova et al. “Parametric Copula-GP model for analyzing multidimensional neuronal and behavioral relationships”. en. In: *PLOS Computational Biology* 18.1 (Jan. 2022). Publisher: Public Library of Science, e1009799. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1009799. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1009799 (visited on 10/26/2023).

[37] Rylan S. Larsen et al. *Activation of neuromodulatory axon projections in primary visual cortex during periods of locomotion and pupil dilation*. en. Pages: 502013 Section: New Results. Dec. 2018. DOI: 10.1101/502013. URL: https://www.biorxiv.org/content/10.1101/502013v1 (visited on 03/22/2024).

[38] Günter Last and Mathew Penrose. *Lectures on the Poisson Process*. en. Google-Books-ID: JRs3DwAAQBAJ. Cambridge University Press, Oct. 2017. ISBN: 978-1-107-08801-6.

[39] Xiao Lin et al. *Data-Efficient Mutual Information Neural Estimator*. arXiv:1905.03319 [cs, stat]. May 2019. DOI: 10.48550/arXiv.1905.03319. URL: http://arxiv.org/abs/1905.03319 (visited on 10/26/2023).

[40] Edward N. Lorenz. “Deterministic Nonperiodic Flow”. EN. In: *Journal of the Atmospheric Sciences* 20.2 (Mar. 1963). Publisher: American Meteorological Society Section: Journal of the Atmospheric Sciences, pp. 130–141. ISSN: 0022-4928, 1520-0469. DOI: 10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2.

URL: https://journals.ametsoc.org/view/journals/atsc/20/2/1520-0469_1963_020_0130_dnf_2_0_co_2.xml (visited on 02/27/2024).

[41] Jian Ma and Zengqi Sun. "Mutual Information Is Copula Entropy". In: *Tsinghua Science & Technology* 16.1 (Feb. 2011), pp. 51–54. ISSN: 1007-0214. DOI: 10.1016/S1007-0214(11)70008-6. URL: https://www.sciencedirect.com/science/article/pii/S1007021411700086 (visited on 03/21/2024).

[42] Alexander Mathis et al. "DeepLabCut: markerless pose estimation of user-defined body parts with deep learning". en. In: *Nature Neuroscience* 21.9 (Sept. 2018). Number: 9 Publisher: Nature Publishing Group, pp. 1281–1289. ISSN: 1546-1726. DOI: 10.1038/s41593-018-0209-y. URL: https://www.nature.com/articles/s41593-018-0209-y (visited on 10/17/2023).

[43] Paul B. McBeth et al. "Robotics in neurosurgery". In: *The American Journal of Surgery* 188.4, Supplement 1 (Oct. 2004), pp. 68–75. ISSN: 0002-9610. DOI: 10.1016/j.amjsurg.2004.08.004. URL: https://www.sciencedirect.com/science/article/pii/S0002961004003162 (visited on 04/02/2024).

[44] Lazaros Mitskopoulos, Theoklitos Amvrosiadis, and Arno Onken. "Mixed vine copula flows for flexible modeling of neural dependencies". In: *Frontiers in Neuroscience* 16 (Sept. 2022), p. 910122. ISSN: 1662-453X. DOI: 10.3389/fnins.2022.910122. URL: https://www.frontiersin.org/articles/10.3389/fnins.2022.910122/full (visited on 10/08/2023).

[45] Thomas Nagler and Claudia Czado. "Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas". In: *Journal of Multivariate Analysis* 151 (Oct. 2016), pp. 69–89. ISSN: 0047-259X. DOI: 10.1016/j.jmva.2016.07.003. URL: https://www.sciencedirect.com/science/article/pii/S0047259X16300471 (visited on 10/22/2023).

[46] Roger B. Nelsen. *An introduction to copulas*. en. 2. ed. Springer series in statistics. New York Berlin Heidelberg: Springer, 2006. ISBN: 978-0-387-28659-4 978-1-4419-2109-3.

[47] *Neuronal Dynamics*. URL: https://books.google.com/books/about/Neuronal_Dynamics.html?id=D4j2AwAAQBAJ (visited on 03/29/2024).

[48] A. Onken and S. Panzeri. "Mixed vine copulas as joint models of spike counts and local field potentials". In: Dec. 2016. URL: https://www.semanticscholar.org/paper/Mixed-vine-copulas-as-joint-models-of-spike-counts-Onken-Panzeri/07bb8182920fce1ec11522a1dbc82d546658efb1 (visited on 10/08/2023).

[49] Arno Onken and Stefano Panzeri. "Mixed vine copulas as joint models of spike counts and local field potentials". en. In: ().

[50] Janelle MP Pakan, Valerio Francioni, and Nathalie L Rochefort. "Action and learning shape the activity of neuronal circuits in the visual cortex". In: *Current Opinion in Neurobiology*. Systems Neuroscience 52 (Oct. 2018), pp. 88–97. ISSN: 0959-4388. DOI: 10.1016/j.conb.2018.04.020. URL: https://www.sciencedirect.com/science/article/pii/S0959438818300400 (visited on 10/17/2023).

[51] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.

cc/paper_files/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html (visited on 10/23/2023).

[52] Felix Pei et al. *Neural Latents Benchmark '21: Evaluating latent variable models of neural population activity*. arXiv:2109.04463 [cs, q-bio]. Jan. 2022. DOI: 10.48550/arXiv.2109.04463. URL: http://arxiv.org/abs/2109.04463 (visited on 03/24/2024).

[53] Gregor Pirš and Erik Štrumbelj. "Bayesian Gaussian process factor analysis with copula for count data". In: *Expert Systems with Applications* 197 (July 2022), p. 116645. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2022.116645. URL: https://www.sciencedirect.com/science/article/pii/S0957417422001336 (visited on 10/27/2023).

[54] Rodrigo Quian Quiroga and Stefano Panzeri. "Extracting information from neuronal populations: information theory and decoding approaches". en. In: *Nature Reviews Neuroscience* 10.3 (Mar. 2009). Number: 3 Publisher: Nature Publishing Group, pp. 173–185. ISSN: 1471-0048. DOI: 10.1038/nrn2578. URL: https://www.nature.com/articles/nrn2578 (visited on 10/17/2023).

[55] Rodrigo Quian Quiroga and Stefano Panzeri. *Principles of Neural Coding*. en. Google-Books-ID: MkTNBQAAQBAJ. CRC Press, May 2013. ISBN: 978-1-4398-5331-3.

[56] Sumanta Ray, Snehalika Lall, and Sanghamitra Bandyopadhyay. "CODC: a Copula-based model to identify differential coexpression". en. In: *npj Systems Biology and Applications* 6.1 (June 2020). Publisher: Nature Publishing Group, pp. 1–13. ISSN: 2056-7189. DOI: 10.1038/s41540-020-0137-9. URL: https://www.nature.com/articles/s41540-020-0137-9 (visited on 04/03/2024).

[57] G. Thippa Reddy et al. "Analysis of Dimensionality Reduction Techniques on Big Data". In: *IEEE Access* 8 (2020). Conference Name: IEEE Access, pp. 54776–54788. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2980942. URL: https://ieeexplore.ieee.org/abstract/document/9036908 (visited on 10/26/2023).

[58] Christian P. Robert and George Casella. "Monte Carlo Integration". en. In: *Monte Carlo Statistical Methods*. Ed. by Christian P. Robert and George Casella. New York, NY: Springer, 1999, pp. 71–138. ISBN: 978-1-4757-3071-5. DOI: 10.1007/978-1-4757-3071-5_3. URL: https://doi.org/10.1007/978-1-4757-3071-5_3 (visited on 03/22/2024).

[59] Houman Safaai et al. "Information estimation using nonparametric copulas". In: *Physical review. E* 98.5 (Nov. 2018), p. 053302. ISSN: 2470-0045. DOI: 10.1103/PhysRevE.98.053302. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6458593/ (visited on 10/17/2023).

[60] Jeremy D. Schmahmann. "The cerebellum and cognition". In: *Neuroscience Letters*. The Cerebellum in Health and Disease 688 (Jan. 2019), pp. 62–75. ISSN: 0304-3940. DOI: 10.1016/j.neulet.2018.07.005. URL: https://www.sciencedirect.com/science/article/pii/S0304394018304671 (visited on 04/03/2024).

[61] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. "A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions". In: *Journal of Mathematical Psychology* 85 (Aug. 2018), pp. 1–16. ISSN: 0022-2496. DOI:

10.1016/j.jmp.2018.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0022249617302158 (visited on 03/21/2024).

[62] Gideon Schwarz. "Estimating the Dimension of a Model". In: *The Annals of Statistics* 6.2 (1978). Publisher: Institute of Mathematical Statistics, pp. 461–464. ISSN: 0090-5364. URL: https://www.jstor.org/stable/2958889 (visited on 03/28/2024).

[63] Babak Shahbaba et al. "A Semiparametric Bayesian Model for Detecting Synchrony Among Multiple Neurons". en. In: *Neural Computation* 26.9 (Sept. 2014), pp. 2025–2051. ISSN: 0899-7667, 1530-888X. DOI: 10.1162/NECO_a_00631. URL: https://direct.mit.edu/neco/article/26/9/2025-2051/8006 (visited on 10/25/2023).

[64] Claude Elwood Shannon and Warren Weaver. *The mathematical theory of communication*. eng. OCLC: 40716662. Urbana: University of Illinois Press, 1998. ISBN: 978-0-252-72546-3. URL: https://www.loc.gov/catdir/enhancements/fy1616/98230924-b.html (visited on 03/29/2024).

[65] Nicholas A. Steinmetz et al. "Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings". In: *Science* 372.6539 (Apr. 2021). Publisher: American Association for the Advancement of Science, eabf4588. DOI: 10.1126/science.abf4588. URL: https://www.science.org/doi/full/10.1126/science.abf4588 (visited on 02/08/2024).

[66] Robert J. Sternberg and Edward E. Smith. *The Psychology of Human Thought*. en. Google-Books-ID: DZ44AAAAIAAJ. CUP Archive, Feb. 1988. ISBN: 978-0-521-31115-1.

[67] Dugald Stewart. *Elements of the Philosophy of the Human Mind*. en. Google-Books-ID: h98AAAAAYAAJ. E. and E. Hosford, 1822.

[68] Ben Van Vliet. *Abe Sklar's*. en. SSRN Scholarly Paper. Rochester, NY, Mar. 2023. DOI: 10.2139/ssrn.4198458. URL: https://papers.ssrn.com/abstract=4198458 (visited on 10/17/2023).

[69] Mai-Anh T. Vu et al. "A Shared Vision for Machine Learning in Neuroscience". en. In: *Journal of Neuroscience* 38.7 (Feb. 2018). Publisher: Society for Neuroscience Section: TechSights, pp. 1601–1607. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.0508-17.2018. URL: https://www.jneurosci.org/content/38/7/1601 (visited on 10/17/2023).

[70] A. Wilson and H. Nickisch. "Kernel Interpolation for Scalable Structured Gaussian Processes (KISS-GP)". In: Mar. 2015. URL: https://www.semanticscholar.org/paper/Kernel-Interpolation-for-Scalable-Structured-Wilson-Nickisch/767d0625c4767c6b8afa0b1b30deafed7e0e8f08 (visited on 10/25/2023).

[71] Byron M. Yu et al. "Gaussian-Process Factor Analysis for Low-Dimensional Single-Trial Analysis of Neural Population Activity". en. In: *Journal of Neurophysiology* 102.1 (July 2009), pp. 614–635. ISSN: 0022-3077, 1522-1598. DOI: 10.1152/jn.90941.2008. URL: https://www.physiology.org/doi/10.1152/jn.90941.2008 (visited on 10/08/2023).

[72] Zhi-Hua Zhou. "Why over-parameterization of deep neural networks does not overfit?" en. In: *Science China Information Sciences* 64.1 (Jan. 2021), p. 116101. ISSN: 1674-733X, 1869-1919. DOI: 10.1007/s11432-020-2885-6. URL:

https://link.springer.com/10.1007/s11432-020-2885-6 (visited on 04/02/2024).

[73] Ariel Zylberberg, Manuel Oliva, and Mariano Sigman. "Pupil Dilation: A Fingerprint of Temporal Selection During the "Attentional Blink"". English. In: *Frontiers in Psychology* 3 (Aug. 2012). Publisher: Frontiers. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2012.00316. URL: https://www.frontiersin.org/journals/psychology/articles/10.3389/fpsyg.2012.00316/full (visited on 03/23/2024).

Appendix A

Reproducability, Source Code Contributions, and Additional Figures

A.1 Validation Test Reproducability

All validation tests were done using variations of arguments for the `validate_bagging.py` python file.

Test 1 For test 1, we utilized:

- Random Seed: 859448723
- Num. Estimators: 4
- Maximum Copula Elements: 5
- Dimensions: 2
- Shuffling of Data: Yes
- Input Type: Random

Test 2 For test 2, we utilized:

- Random Seed: 859443
- Num. Estimators: 4
- Maximum Copula Elements: 3
- Dimensions: 2
- Shuffling of Data: Yes
- Input Type: Random

Test 3 For test 3, we utilized:

- Random Seed: 859443
- Num. Estimators: 4
- Maximum Copula Elements: 3
- Dimensions: 2
- Shuffling of Data: No
- Input Type: Linear

A.2 Source Code Contributions

Contributions to *Copula-GP* src can be found on the project repository (`https://github.com/mwalden00/HonProj`).

- `train/train_next_tree.py`: Calls to `worker` were missing function arguments. I believe these were meant to be applied via the `global` statement, but I could not get the implementation to function that way. As such, I simply added variables with the `global` attachment to arguments.
- `requirements.txt`: Missing `seaborn` v 0.10.0.
- `bvcopula/distributions.py`: In `GaussianCopula` class, there are sometimes tensor device mismatches that occur when `log_prob` is called.
- `bvcopula/conf.py`: Possibly ignore; when training a vine I *think Gpytorch* will sometimes have floating point errors resulting in constraint violations. This kills workers in vine training, resulting in failure when they are collected later on. As such, relaxed constraints by 0.0001, which might break some calculations if unlucky, but fixed the issue in testing. If *Gpytorch* fixes this issue later on, there won't be a problem.
- `synthetic_data/synthetic_data.py` Added minumum vine mixture copula elements argument for convenience.
- `vine/vine.py` + `bvcopula/distributions.py` The `entropy` functions can take a lot of memory, resulting in shortages when running on GPU. As such, added `gc.collect()` and `torch.cuda.empty_cache()` lines to optimize slightly. It will be better to fix these in a more elegant way sometime in the future.
- `select_model/bagging.py`: Bagging methods are here; It might be better to move the vine bits to train. In addition, a method to train singleton mixtures and combine them via bagging might be a worthy inclusion.
- Many training related files: Miscellaneous device optimizations were put in the code due to my own misunderstanding. I would recommend removing default device instantiating on CPU, as it can lead to head-aches in device handling later on if the user is not properly aware; forcing device specification would make things easier when learning to use *Copula-GP* and *PyTorch* device handling.

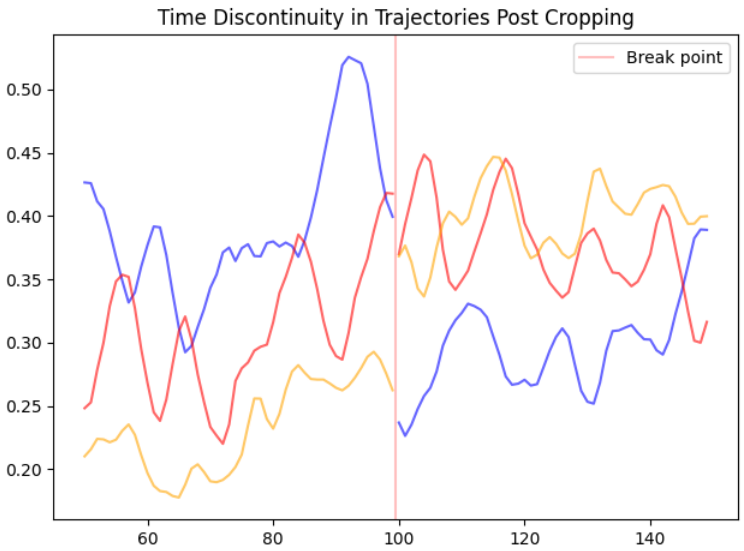


Figure A.1: Discontinuities present in trajectory data post-cropping and -concatenation (3 of 13 trajectories shown). Such discontinuities can negatively effect GP performance when estimating GP-link functions during the *Copula-GP* fit process.

A.3 Figures

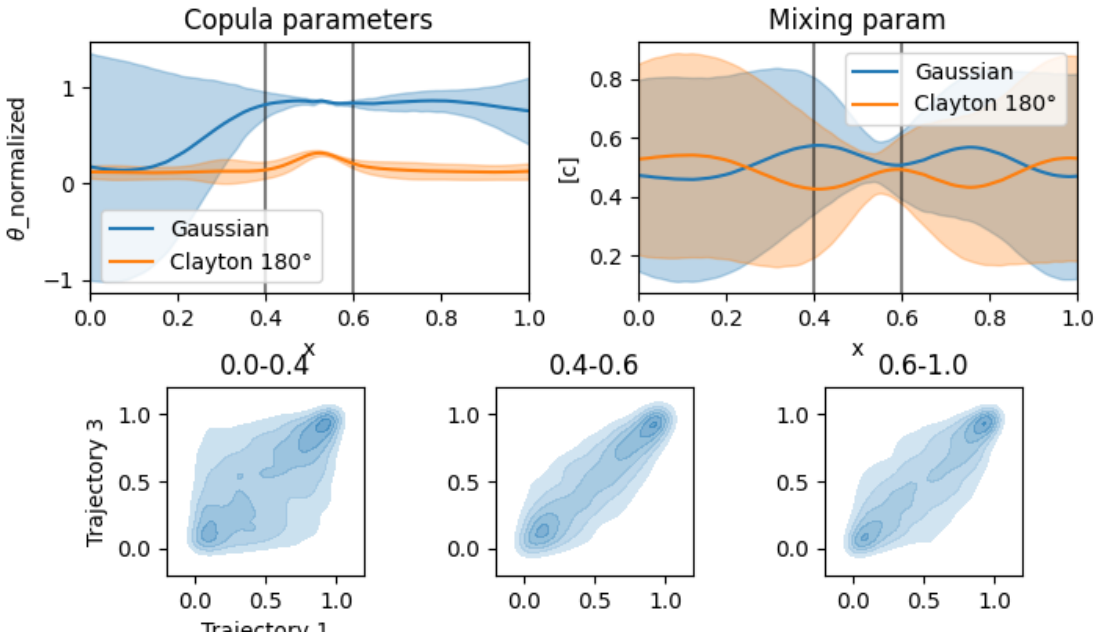


Figure A.2: Lowest level *Copula-GP* bivariate copula predictor in a C-vine predictor fit to neuronal data sourced from mice and mean predicted copulas. The top right of the figure is copula mixing parameters as a function of pupil-width, with a shaded region for uncertainty (0.95 CI). At the top left of the figure we find copula dependency parameters as a function of pupil-width, with a shaded region for uncertainty (0.95 CI). At the bottom are sample density plots of the copulas extracted for different ranges of pupil-width values (ranges [0.0,0.4], [0.4,0.6], and [0.6,1.0]), produced from 1500 continuous samples. Note that pupil dilation was robustly normalized, with a mean pupil dilation of 0.548 (see figure 4.3d for distribution). *Copula-GP* based estimators were fit utilizing 10000 data points as part of a C-vine modeling neuronal trajectories (13 total). Notice in A.2 the large increase in dependency- and mixing-parameter uncertainty as pupil-width deviates from the mean.

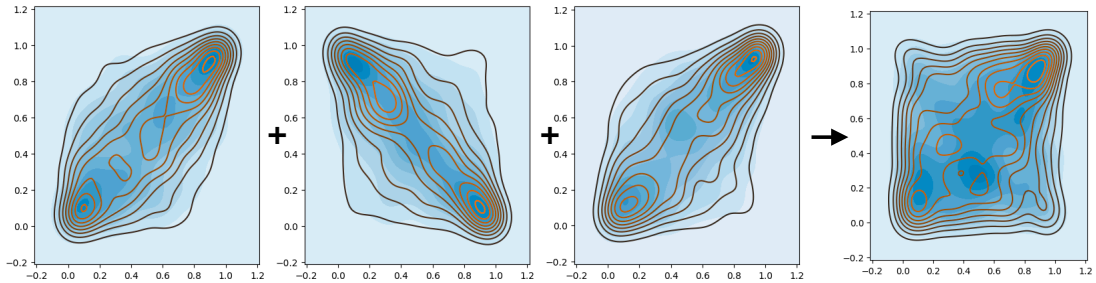


Figure A.3: Example of aggregated copula as a naive mean on *Copula-GP* estimated copulas via algorithm 1, with the right-most copula being the aggregation of the copulas to the left of it. Note how the shape of the aggregated copula and the copula tail distributions are aggregated from the other copulas.