Parallel Computation Framework for Discrete Copula Modeling

Dhyey Mavani

Amherst College

Submitted to the Department of Mathematics and Statistics of Amherst College in partial fulfillment of the requirements for the degree of Bachelor of Arts with honors.

Advisor(s):
Professor Shu-Min Liao

February 4, 2025

Table of contents

Abstract

Understanding regression dependencies among discrete variables in categorical data—especially with ordinal responses—is a significant challenge in fields like finance, where the natural order of variables can unlock deeper insights into underlying distributions and data generating processes (DGPs). While numerous model-based methods have been developed to examine these structures, there is a notable lack of flexible, model-free approaches. To address this gap, a novel model-free measure based on the checkerboard copula, was introduced by Wei and Kim (2021) to identify and quantify regression dependence in multivariate categorical data involving both ordinal and nominal variables. Building upon this foundation, my thesis focuses on developing scalable and modularized implementations of discrete checkerboard copula modeling in R and Python, utilizing parallel computing to enhance efficiency and accessibility for large-scale data analysis. Initial experimentation and deployment confirm the effectiveness of these tools, providing researchers with a powerful resource for exploratory modeling and a deeper investigation into regression dependence structures within complex categorical datasets.

Acknowledgements

I want to thank everyone who made my experience unforgettable during my thesis journey. Firstly, I want to convey my gratitude to Professor Shu-Min Liao for advising me throughout my time at Amherst College and believing in me to take on the challenge of developing a statistical software component encompassing her most recent research work. From my first research experience on campus building R-Blocks to introducing me to her research collaborator (Professor Daeyoung Kim), Prof. Liao played a pivotal role in my development. Additionally, I am indebted to Dr. Kim for his continuous encouragement and feedback while developing the software this past year.

I am also incredibly grateful to my college advisor and statistics major advisor, Professor Nicholas Horton, for always advocating for me and supporting me throughout the Amherst College experience. I also thank Professor Jun Ishii for teaching me Advanced Econometrics and Professor Amy Wagaman for teaching me Advanced Data Analysis, which helped me gain a clear and solid understanding of the foundational tools I could build on in this work.

Finally, I would like to express gratitude towards my family for their constant belief in my abilities. Special thanks to my mom, dad, sister, grandfather, and grandmother for making me capable of the opportunity to study abroad. Last but not least, I would like to thank my friends, peers, and colleagues on campus, who took courses, worked, and played sports with me. This academic, personal, and professional growth journey would not be possible without their support.

Chapter 1

Introduction

XXX

Chapter 2

Unraveling Notion of Dependence through Copulas

In this chapter, we aim to formalize concepts of dependence and association. To facilitate our understanding, we will use two bivariate random vectors and visualize their relationships through Python code.

2.1 Motivating Example

Consider (X_1, X_2) and (Y_1, Y_2) be bivariate random vectors, each consisting of 10000 independent data-points, which are distributed with the joint distributions F_X and F_Y respectively. Given these bivariate vectors, one might ask: How can I compare the relationship between (X_1, X_2) to the relationship between (Y_1, Y_2)? One of the measures that can help us compare and contrast these relationships is Pearson correlation coefficient (commonly denoted as $\rho_{pearson}$). After preliminary calculations on a Python3 kernel, we can see that $\rho_{pearson}(X_1, X_2) \approx 0.802$, but on the other hand, the correlation between

$\rho_{pearson}(Y_1, Y_2) \approx 0.755$. From these measure-values, it seems that the dependence between (X_1, X_2) is stronger than the dependence between (Y_1, Y_2). Although this agrees with our scatter plots in Figure 2.1, it is vital to note that $\rho_{pearson}$ only captures the linear dependence between the underlying random variables at hand.

Upon observing the Figure 2.1 closely, we note that the marginal distributions of X_1 and X_2 are close to normal, unlike the marginals of Y_1 and Y_2. Moreover, we can see that the relationship between Y_1 and Y_2 is non-linear. This vast difference in marginals takes away our trust from the appropriateness of the use of $\rho_{pearson}$ as a measure to compare dependence between the data vectors at hand.

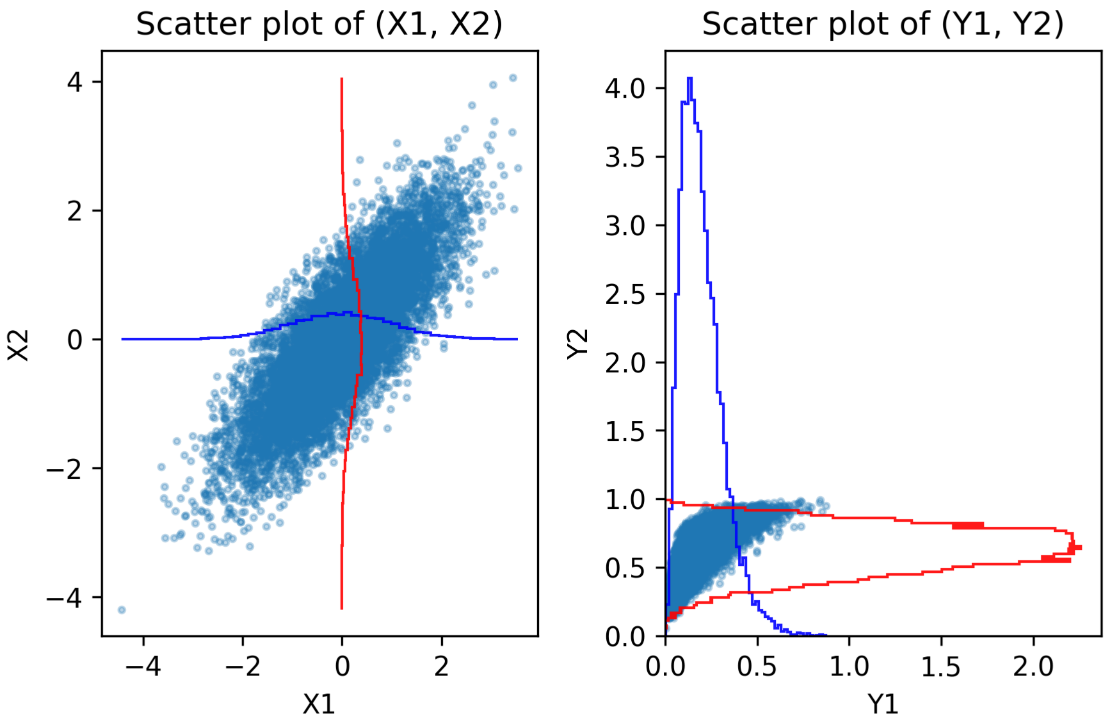

Figure 2.1: Scatter plots of 10000 independent observations of (X_1, X_2) and (Y_1, Y_2) with overlaid curves depicting respective marginal distributions.

Let's introduce a lemma that will help us transform the marginals so that the resulting marginals are more similar, and try to only capture or extract the "dependence" components, which will allow us to make fairer comparisons.

Lemma 2.1 (Probability Integral Transformation)**.** *(Hofert et al. 2018) Let F be a continuous distribution function and let $X \sim F$, then $F(X)$ is a standard uniform random variable, that is, $F(X) \sim U(0,1)$.*

Lemma 2.1 allows us to transform a continuous random variable to a random variable which has standard uniform distribution. So, by using this transformation, we can now convert our marginals X_1, X_2, Y_1, Y_2 individually to be distributed Uniform$(0,1)$. And, since now the resulting marginals will all be of the same type, it will allow us to compare the dependence between random variables on fairer grounds.

For instance, if we know that $X_1 \sim N(0,1) = F_1$, $X_2 \sim N(0,1) = F_2$, $Y_1 \sim Gamma(3,15) = G_1$, and $Y_2 \sim Beta(5,3) = G_2$, where F_1, F_2, G_1, G_2 denote the distribution functions of the respective random variables. By Lemma 2.1, we can say that $F_1(X_1), F_2(X_2), G_1(Y_1)$, and $G_2(Y_2)$ are each distributed Uniform$(0,1)$.

Looking at Figure 2.2, we can see that the transformed data vectors appear to be significantly similar. We can computationally verify this by quickly calculating the $\rho_{pearson}$ for $(F_1(X_1), F_2(X_2))$ and $(G_1(Y_1), G_2(Y_2))$, which turns out to be 0.788 for both data vector pairs, meaning that both have same dependence structures.

An alternative way to approach the problem (of comparing dependence of distinct pairs of marginals), is by transforming the marginals of (Y_1, Y_2) to be normal (same as marginals of (X_1, X_2)). As one can predict, in order to accomplish this transformation, we would need to "undo the current distributional mappings on (Y_1, Y_2)", which we can formally define as generalized inverse as follows:

Definition 2.1 (Quantile Function)**.** (Hofert et al. 2018) $F^{\leftarrow}$ (Quantile Function) is defined

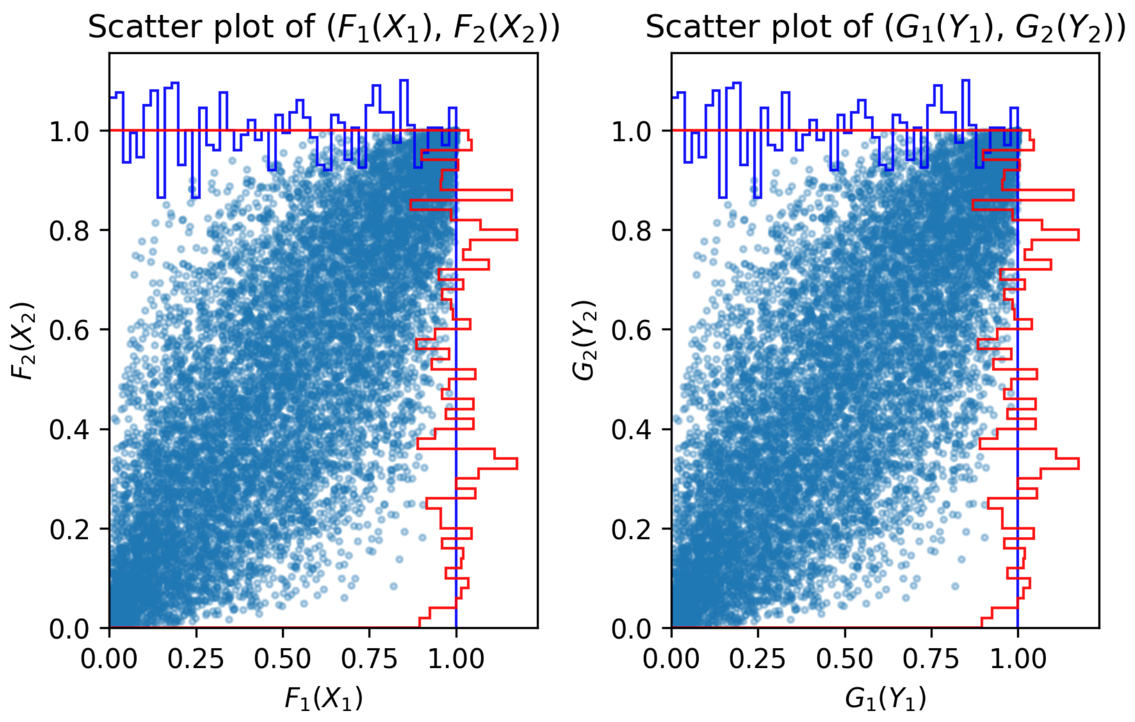

Figure 2.2: Scatter plots of 10000 independent observations of $(F_1(X_1), F_2(X_2))$ and $(G_1(Y_1), G_2(Y_2))$ with overlaid curves depicting respective marginal distributions.

as $F^{\leftarrow}(y) = \inf\{x \in \mathbb{R} | F(x) \geq y\}$, where $y \in [0, 1]$, and inf is the infimum of a set.

⚠ Warning

The quantile function $F^{\leftarrow} = F^{-1}$ only when F is continuous and strictly increasing. Thus it is important to note that, in other cases, the ordinary inverse F^{-1} need not exist. (Hofert et al. 2018)

With the above definition of $F^{\leftarrow}$, let's introduce a lemma from (Hofert et al. 2018) that will help us perform the transformation to normal.

Lemma 2.2 (Quantile Transformation). *(Hofert et al. 2018) Let $U \sim$ Unif$(0, 1)$ and let F be any distribution function be a distribution function. Then $F^{\leftarrow}(U) \sim F$, that is, $F^{\leftarrow}(X)$ is distributed with density F.*

Note

Lemma 2.2 is valid for non-continuous densities F as well. (Hofert et al. 2018)

Let's start with the transformations where we left off in Figure 2.2, since we have uniform densities there. Applying Lemma 2.2 on $G_1(Y_1)$ and $G_2(Y_2)$ using quantile functions $F_1^{\leftarrow} = F_1^{-1}$ and $F_2^{\leftarrow} = F_2^{-1}$ respectively gives us that $F_1^{-1}(G_1(Y_1)) \sim F_1$ and $F_2^{-1}(G_2(Y_2)) \sim F_2$.

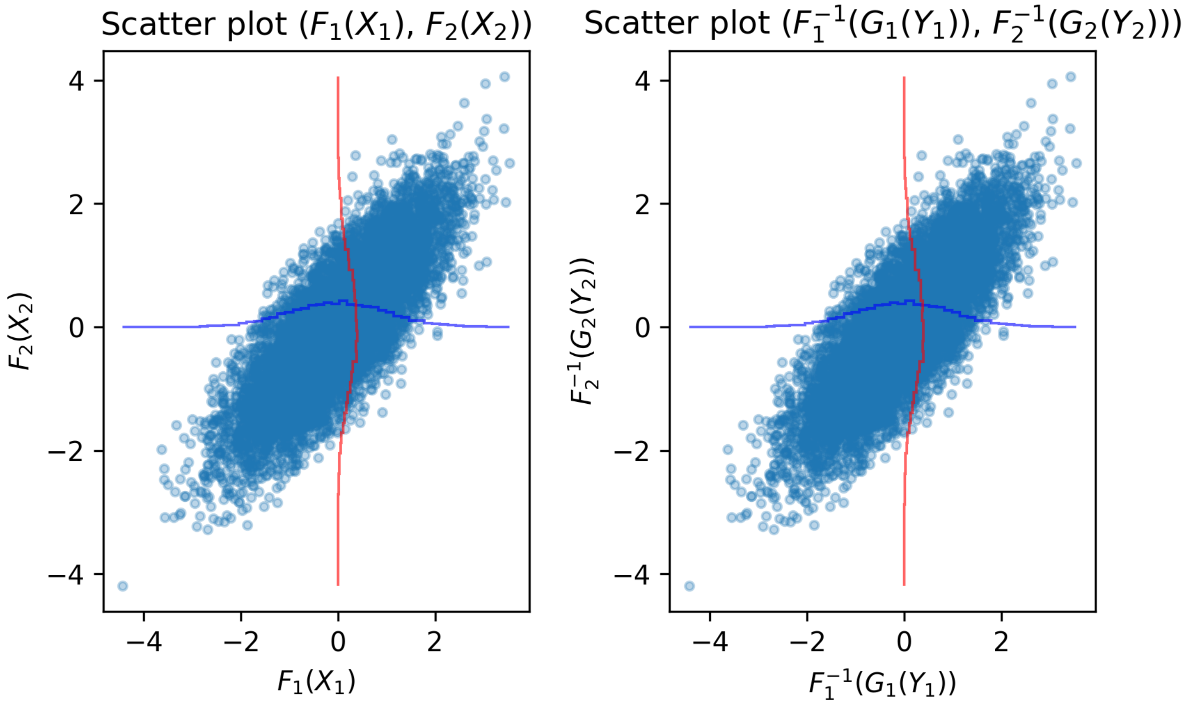

Figure 2.3: Scatter plots of 10000 independent observations of (X_1, X_2) and $(F_1^{-1}(G_1(Y_1)), F_2^{-1}(G_2(Y_2)))$ with overlaid curves depicting respective marginal distributions.

Notice in Figure 2.3 that the resulting transformed distribution through this alternative method resembles that of (X_1, X_2). Hence, we can conclude that they have the same dependence. Furthermore, through a quick calculation, we can see that $\rho_{pearson}(F_1^{-1}(G_1(Y_1)), F_2^{-1}(G_2(Y_2))) = 0.802$, which is the same as the Pearson correlation coefficient between X_1 and X_2. This is the level of flexibility that a combination of transformations presented in Lemma 2.1 and Lemma 2.2 can lend us.

Note

"(X_1, X_2) and (Y_1, Y_2) have the same dependence" $\iff$ "(X_1, X_2) and (Y_1, Y_2) have the same copula" (Hofert et al. 2018)

2.2 Copulas: A Unified Framework for Dependence

Copulas are a class of multivariate distribution functions with $Unif(0, 1)$ marginals. The motivating example in the previous section explains the usage of copulas as the structures capturing margin-independent dependence between random variables.

Note

The choice of $Unif(0, 1)$ as a post-transformation margin for the data at hand is somewhat arbitrary although it does simplify further results. One can use modifications of Lemma 2.1 and Lemma 2.2 to define copulas with respect to any margin of choice without affecting the final conclusions about the dependence between the data at hand. (Hofert et al. 2018)

In order to understand copulas better, for now, let's restrict ourselves to the 2-D (2-dimensional) case. Firstly, let's introduce the definition of a broader class of functions called subcopulas as a preliminary, which will help us mathematically define copulas as a special case. (Nelsen 2006)

Definition 2.2 (2-Dimensional Subcopula). (Erdely 2017) A **two-dimensional subcopula** (2-subcopula) is a function $C^S : D_1 \times D_2 \longrightarrow [0, 1]$, where $\{0, 1\} \subseteq D_i \subseteq [0, 1]$ for $i \in \{1, 2\}$ with the following conditions satisfied:

- *Grounded:* $C^S(u, 0) = 0 = C^S(0, v)$, $\forall u \in D_1, \forall v \in D_2$.

- *Marginal Consistency:* $\forall u \in D_1$ and $\forall v \in D_2$, $C^S(u,1) = u$ and $C^S(1,v) = v$.

- *2-increasing:* $\forall u_1, u_2 \in D_1$ and $\forall v_1, v_2 \in D_2$ such that $u_1 \leq u_2$ and $v_1 \leq v_2$, $C^S(u_1,v_1) - C^S(u_2,v_1) + C^S(u_2,v_2) - C^S(u_1,v_2) \geq 0$.

Definition 2.3 (2-Dimensional Copula). (Erdely 2017) A **two-dimensional copula** (2-copula) is a function $C : [0,1] \times [0,1] \longrightarrow [0,1]$, with the following conditions satisfied:

- *Grounded:* $C(u,0) = 0 = C(0,v)$, $\forall u \in [0,1], \forall v \in [0,1]$.

- *Marginal Consistency:* $\forall u \in [0,1]$ and $\forall v \in [0,1]$, $C(u,1) = u$ and $C(1,v) = v$.

- *2-increasing:* $\forall u_1, u_2 \in [0,1]$ and $\forall v_1, v_2 \in [0,1]$ such that $u_1 \leq u_2$ and $v_1 \leq v_2$, $C(u_1,v_1) - C(u_2,v_1) + C(u_2,v_2) - C(u_1,v_2) \geq 0$.

Note

A 2-D copula is essentially a 2-subcopula with a full unit square as domain ($D_1 = D_2 = [0,1]$). Furthermore, copula and subcopula are the same within a domain with continuous variables. Later in this chapter, we will discuss why this doesn't hold when one of the variables is discrete.

In this work, we will mainly deal with 2-D copulas and subcopulas, but the definitions above can be generalized to n-D case with some notable exceptions detailed (with proofs) in section 2.10 of Nelsen (2006). Moreover, there are many different families of copulas bearing peculiar properties and corresponding margins, we are not covering them in detail since that is not the focus of this work, and a comprehensive summary of many of these families can be found in chapter 3 of Hofert et al. (2018).

2.3 Fréchet-Hoeffding Bounds

XXX

2.4 Sklar's Theorem and its Corollaries

XXX

2.5 The Invariance Principle

XXX

2.6 Measures of Association and Copula Estimation

XXX

Note on Moving from Continuous to Discrete Case

Main High-Level Idea

2.7 Does Everything Work in Discrete Case as well?

XXX + Motivation from end

2.7.1 Unidentifiability Issue

XXX

2.7.2 Margin-Dependence of Concordance Association Measures

XXX

2.8 Chapter 2 Code

The following code was used to create Chapter 2.

2.8.1 Code within chapter

```
# Load knitr package
library(knitr)

# Python Engine Setup
knit_engines$set(python3 = knit_engines$get("python"))

# Load packages
library(tidyverse)
library(gt)

# Set default ggplot theme for document
theme_set(theme_classic())
# If using kableExtra tables, print blank cells instead of `NA`
options(knitr.kable.NA = "")

# Load NBA Data
load("data/temp_wnba.RData")
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import beta, expon, norm, gamma, binom
```

```
import os

# Create directory if not exists
fig_dir = "fig"
os.makedirs(fig_dir, exist_ok=True)

# Generate Data
np.random.seed(8990)
n = 10000
mean = [0, 0]
cov = [[1, 0.8], [0.8, 1]]
X = np.random.multivariate_normal(mean, cov, size=n)
X1, X2 = X[:, 0], X[:, 1]

# Transform U_X1 and U_X2 to uniform [0, 1] using the CDF of the normal distribution
U_X1 = norm.cdf(X1)
U_X2 = norm.cdf(X2)

# Transform U_X1 and U_X2 into Gamma and Beta distributions
Y1 = gamma.ppf(U_X1, a=3, scale=1/15)
Y2 = beta.ppf(U_X2, a=5, b=3)

# Calculate Pearson Correlation Coefficients
```

```
rho_X = np.corrcoef(X1, X2)[0, 1]
rho_Y = np.corrcoef(Y1, Y2)[0, 1]
print("Pearson correlation for (X1, X2):", rho_X)
print("Pearson correlation for (Y1, Y2):", rho_Y)

# Create Layout design and Set Size-Ratio
fig, axes = plt.subplots(1, 2, figsize=(6, 4))

# Scatter plot for (X1, X2)
axes[0].scatter(X1, X2, alpha=0.3, s=5)
axes[0].set_title("Scatter plot of (X1, X2)")
axes[0].set_xlabel("X1")
axes[0].set_ylabel("X2")

# Add marginal histograms
axes[0].hist(X1, bins=50, density=True, alpha=0.9, color='blue', orientation='vertical', hist
axes[0].hist(X2, bins=50, density=True, alpha=0.9, color='red', histtype='step', orientation=

# Scatter plot for (Y1, Y2)
axes[1].scatter(Y1, Y2, alpha=0.3, s=5)
axes[1].set_title("Scatter plot of (Y1, Y2)")
axes[1].set_xlabel("Y1")
axes[1].set_ylabel("Y2")
```

```

# Add marginal histograms
axes[1].hist(Y1, bins=50, density=True, alpha=0.9, color='blue', orientation='vertical
axes[1].hist(Y2, bins=50, density=True, alpha=0.9, color='red', histtype='step', orient

# Organize into a tight layout as per matplotlib
plt.tight_layout()

# Save figure instead of showing it
fig_path = os.path.join(fig_dir, "motivating_example.png")
plt.savefig(fig_path, dpi=300, bbox_inches='tight')

# Close the figure to prevent rendering output
plt.close(fig)
knitr::include_graphics("fig/motivating_example.png")

# Set random seed for reproducibility
np.random.seed(8990)

# Apply probability integral transformation to all variables to make them uniform
U_Y1 = gamma.cdf(Y1, a=3, scale=1/15)
U_Y2 = beta.cdf(Y2, a=5, b=3)

# Calculate Pearson Correlation Coefficients
```

```python
rho_U_X = np.corrcoef(U_X1, U_X2)[0, 1]
rho_U_Y = np.corrcoef(U_Y1, U_Y2)[0, 1]
print("Pearson correlation for ($F_1(X_1)$, $F_2(X_2)$):", rho_U_X)
print("Pearson correlation for ($G_1(Y_1)$, $G_2(Y_2)$):", rho_U_Y)

# Combine transformed data
uniform_data = np.vstack([U_X1, U_X2, U_Y1, U_Y2]).T

# Verify the uniformity of transformed data (Should be 0.5 in value)
print("U_X1 mean:", U_X1.mean(), "U_X2 mean:", U_X2.mean())
print("U_Y1 mean:", U_Y1.mean(), "U_Y2 mean:", U_Y2.mean())

# Create Layout design and Set Size-Ratio
fig, axes = plt.subplots(1, 2, figsize=(6, 4))

# Scatter plot for (U_X1, U_X2)
axes[0].scatter(U_X1, U_X2, alpha=0.3, s=5)
axes[0].set_title("Scatter plot of ($F_1(X_1)$, $F_2(X_2)$)")
axes[0].set_xlabel("$F_1(X_1)$")
axes[0].set_ylabel("$F_2(X_2)$")

# Add marginal histograms
axes[0].hist(U_X1, bins=50, density=True, alpha=0.9, color='blue', orientation='vertical', hi
axes[0].hist(U_X2, bins=50, density=True, alpha=0.9, color='red', histtype='step', orie

# Scatter plot for (U_Y1, U_Y2)
axes[1].scatter(U_Y1, U_Y2, alpha=0.3, s=5)
axes[1].set_title("Scatter plot of ($G_1(Y_1)$, $G_2(Y_2)$)")
axes[1].set_xlabel("$G_1(Y_1)$")
axes[1].set_ylabel("$G_2(Y_2)$")

# Add marginal histograms
axes[1].hist(U_Y1, bins=50, density=True, alpha=0.9, color='blue', orientation='vertica
axes[1].hist(U_Y2, bins=50, density=True, alpha=0.9, color='red', histtype='step', orie

# Organize into a tight layout as per matplotlib
plt.tight_layout()

# Save figure instead of showing it
fig_path = os.path.join(fig_dir, "transformed_motivating_example.png")
plt.savefig(fig_path, dpi=300, bbox_inches='tight')

# Close the figure to prevent rendering output
plt.close(fig)
knitr::include_graphics("fig/transformed_motivating_example.png")

```

```python
# Set random seed for reproducibility
np.random.seed(8990)

# Transform (Y1, Y2) back to normal marginals using quantile transformation
F1_Y1 = norm.ppf(gamma.cdf(Y1, a=3, scale=1/15))
F2_Y2 = norm.ppf(beta.cdf(Y2, a=5, b=3))

# Calculate Pearson Correlation Coefficients
rho_F_Y = np.corrcoef(F1_Y1, F2_Y2)[0, 1]
print("Pearson correlation for transformed:", rho_F_Y)
print("Pearson correlation between X1 and X2:", rho_X)

# Plot the scatter plots with marginal histograms
fig, axes = plt.subplots(1, 2, figsize=(6, 4))

# Scatter plot for original normal marginals (X1, X2)
axes[0].scatter(X1, X2, alpha=0.3, s=10)
axes[0].set_title("Scatter plot ($F_1(X_1)$, $F_2(X_2)$)")
axes[0].set_xlabel("$F_1(X_1)$")
axes[0].set_ylabel("$F_2(X_2)$")
axes[0].hist(X1, bins=50, density=True, alpha=0.6, color='blue', histtype='step')
axes[0].hist(X2, bins=50, density=True, alpha=0.6, color='red', histtype='step', orientation=

# Scatter plot for transformed normal marginals (F1_Y1, F2_Y2)
axes[1].scatter(F1_Y1, F2_Y2, alpha=0.3, s=10)
axes[1].set_title("Scatter plot ($F_1^{-1}(G_1(Y_1))$, $F_2^{-1}(G_2(Y_2))$)")
axes[1].set_xlabel("$F_1^{-1}(G_1(Y_1))$")
axes[1].set_ylabel("$F_2^{-1}(G_2(Y_2))$")
axes[1].hist(F1_Y1, bins=50, density=True, alpha=0.6, color='blue', histtype='step')
axes[1].hist(F2_Y2, bins=50, density=True, alpha=0.6, color='red', histtype='step', or

# Layout adjustment and save the figure
plt.tight_layout()
fig_path = os.path.join(fig_dir, "quantile_transformed_motivating_example.png")
plt.savefig(fig_path, dpi=300, bbox_inches="tight")

# Close the figure to prevent rendering output
plt.close(fig)

knitr::include_graphics("fig/quantile_transformed_motivating_example.png")

# ======================================================================
# Sample R script for thesis template
#
# Cleans temp_raw_wnba.csv dataset, which contains data pulled from
# https://www.espn.com/wnba/stats/player on 2024/06/19
```

```r
#
# Last updated: 2024/06/19
# ======================================================================
library(tidyverse)

wnba <- read_csv("data/temp_raw_wnba.csv") |>
  janitor::clean_names() |>
  # Pull jersey numbers off of names and
  # turn height text into msmt (6'4" = 6.3333)
  mutate(jersey = str_extract(name, "[0-9]+$"),
         name = str_remove(name, "[0-9]+$"),
         ht_ft = parse_number(str_extract(ht, "^[0-9]")),
         ht_in = parse_number(str_extract(ht, '[0-9]+\\"$')),
         height = ht_ft * 12 + ht_in,
         weight = parse_number(wt),
         position = factor(pos,
                           levels = c("G", "F", "C"),
                           labels = c("Guard", "Forward", "Center"))) |>
  select(-c(ht, wt, ht_ft, ht_in, pos))

save(wnba, file = "data/temp_wnba.RData")
```

2.8.2 Code sourced from external scripts

```r
# ======================================================================
# Sample R script for thesis template
#
# Cleans temp_raw_wnba.csv dataset, which contains data pulled from
# https://www.espn.com/wnba/stats/player on 2024/06/19
#
# Last updated: 2024/06/19
# ======================================================================
library(tidyverse)

wnba <- read_csv("data/temp_raw_wnba.csv") |>
  janitor::clean_names() |>
  # Pull jersey numbers off of names and
  # turn height text into msmt (6'4" = 6.3333)
  mutate(jersey = str_extract(name, "[0-9]+$"),
         name = str_remove(name, "[0-9]+$"),
         ht_ft = parse_number(str_extract(ht, "^[0-9]")),
         ht_in = parse_number(str_extract(ht, '[0-9]+\\"$')),
         height = ht_ft * 12 + ht_in,
         weight = parse_number(wt),
         position = factor(pos,
                           levels = c("G", "F", "C"),
                           labels = c("Guard", "Forward", "Center"))) |>
  select(-c(ht, wt, ht_ft, ht_in, pos))

save(wnba, file = "data/temp_wnba.RData")
```

Chapter 3

Checkerboard Copula and Regression Association Measure

Start with motivation and connect with earlier chapters, then define concepts and weave in examples to solidify readers understanding.

3.1 Chapter 2 Code

The following code was used to create Chapter 2.

3.1.1 Code within chapter

```r
# Load packages
library(tidyverse)
library(gt)

# Set default ggplot theme for document
```

```r
theme_set(theme_classic())
# If using kableExtra tables, print blank cells instead of `NA`
options(knitr.kable.NA = "")

# Load data
load("data/temp_wnba.RData")
# ======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# ======================================================================

print("Hello, Amherst!")
```

3.1.2 Code sourced from external scripts

```r
# ======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# ======================================================================

print("Hello, Amherst!")
```

Chapter 4

Applications of Parallel Computing

Start with motivation, connect, and dive into details + examples of parallel computing along with use-cases.

4.1 Chapter 2 Code

The following code was used to create Chapter 2.

4.1.1 Code within chapter

```
# Load packages
library(tidyverse)
library(gt)

# Set default ggplot theme for document
theme_set(theme_classic())
# If using kableExtra tables, print blank cells instead of `NA`
```

```r
options(knitr.kable.NA = "")

# Load data
load("data/temp_wnba.RData")
# ======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# ======================================================================

print("Hello, Amherst!")
```

4.1.2 Code sourced from external scripts

```r
# ======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# ======================================================================

print("Hello, Amherst!")
```

Chapter 5

Software (Package) Implementation and Testing

Start basic with motivation, and breakdown the implementation and testing phases properly while making sure that they are accessible.

5.1 Chapter 2 Code

The following code was used to create Chapter 2.

5.1.1 Code within chapter

```r
# Load packages
library(tidyverse)
library(gt)

# Set default ggplot theme for document
```

```r
theme_set(theme_classic())
# If using kableExtra tables, print blank cells instead of `NA`
options(knitr.kable.NA = "")

# Load data
load("data/temp_wnba.RData")
# ======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# ======================================================================

print("Hello, Amherst!")
```

5.1.2 Code sourced from external scripts

```r
# ======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# ======================================================================

print("Hello, Amherst!")
```

Chapter 6

Conclusion

Like introduction, pull everything together and conclude the work!

6.1 Chapter 2 Code

The following code was used to create Chapter 2.

6.1.1 Code within chapter

```r
# Load packages
library(tidyverse)
library(gt)

# Set default ggplot theme for document
theme_set(theme_classic())
# If using kableExtra tables, print blank cells instead of `NA`
options(knitr.kable.NA = "")
```

```r

# Load data
load("data/temp_wnba.RData")
# =======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# =======================================================================

print("Hello, Amherst!")
```

6.1.2 Code sourced from external scripts

```r
# =======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
#
# Last updated: 2024/08/24
# =======================================================================

print("Hello, Amherst!")
```

References

Erdely, A. (2017), “A subcopula based dependence measure,” *Kybernetika*, Institute of Information Theory; Automation AS CR, 53, 231–243.

Hofert, M., Kojadinovic, I., Maechler, M., and Yan, J. (2018), *Elements of Copula Modeling with r*, Springer Use R! Series.

Nelsen, R. B. (2006), *An introduction to copulas*, Springer Science & business media.

Ushey, K., and Wickham, H. (2024), *Renv: Project environments*.

Wei, Z., and Kim, D. (2021), “On exploratory analytic method for multi-way contingency tables with an ordinal response variable and categorical explanatory variables,” *Journal of Multivariate Analysis*, 186, 104793. https://doi.org/10.1016/j.jmva.2021.104793.

Appendix A

Code availability

This thesis is written using Quarto with **renv** (Ushey and Wickham 2024) to create a reproducible environment. All materials (including the data sets and source files) required to reproduce this document can be found at the Github repository github.com/GITHUB-USERNAME/THESIS-REPO-NAME.

```r
# ======================================================================
# Sample R script for thesis template
#
# Cleans temp_raw_wnba.csv dataset, which contains data pulled from
# https://www.espn.com/wnba/stats/player on 2024/06/19
#
# Last updated: 2024/06/19
# ======================================================================
library(tidyverse)

wnba <- read_csv("data/temp_raw_wnba.csv") |>
  janitor::clean_names() |>
  # Pull jersey numbers off of names and
  # turn height text into msmt (6'4" = 6.3333)
  mutate(jersey = str_extract(name, "[0-9]+$"),
         name = str_remove(name, "[0-9]+$"),
         ht_ft = parse_number(str_extract(ht, "^[0-9]")),
         ht_in = parse_number(str_extract(ht, '[0-9]+\\"$')),
         height = ht_ft * 12 + ht_in,
         weight = parse_number(wt),
         position = factor(pos,
                           levels = c("G", "F", "C"),
                           labels = c("Guard", "Forward", "Center"))) |>
  select(-c(ht, wt, ht_ft, ht_in, pos))

save(wnba, file = "data/temp_wnba.RData")
```

```r
# ======================================================================
# Sample R script for thesis template
#
# Doesn't do anything useful
```

```python
#
# Last updated: 2024/08/24
# =======================================================================

print("Hello, Amherst!")
```

Appendix B

Corrections

This section may be excluded if no corrections are made to your thesis after initial submission to the department and before final submission to the college.

Per the Statistics Honors Thesis Regulations:

> Corrections to theses may be made after the date on which they are due in the Department's hands. Corrections may be made to the body of the thesis, but every such correction will be acknowledged in a list under the heading "Corrections," along with the statement "When originally submitted, this honors thesis contained some errors which have been corrected in the current version. Here is a list of the errors that were corrected." This list will be given on a sheet or sheets to be appended to the thesis. Corrections to spelling, grammar, or typography may be acknowledged by a general statement such as "30 spellings were corrected in various places in the thesis, and the notation for definite integral was changed in approximately 10 places." However, any correction that affects the meaning of a sentence or paragraph should be described in careful detail, and substantial

additions to the thesis will not be allowed. Questions about what should appear in the "Corrections" should be directed to the Chair. Electronic versions of the thesis, technical appendix, and necessary data and supplemental files must all be updated at the time of correction as well.

When originally submitted, this honors thesis contained some errors which have been corrected in the current version. Here is a list of the errors that were corrected.

1. ...
2. ...