A Model-Free Dependence Measure for High-Dimensional Contingency Tables via the Checkerboard Copula and its Potential as a Goodness of Fit Measure

Kenny Chen

April 20, 2022

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:
Shu-Min Liao

Abstract

Categorical data analysis with ordinal responses is important in fields such as the social sciences because when we take into consideration the intrinsic ordering of ordinal variables, we can often obtain more powerful inferences. One step in categorical analysis is exploring the various dependence structures among the variables for exploratory modeling. A dependence structure of particular interest is that of the regression dependence which many model-based approaches have been constructed. However, there are comparatively fewer model-free approaches to examining dependence structures in categorical data, and most of these do not focus on regression dependence. To address this, Wei & Kim (2021) proposed a new model-free measure based on the checkerboard copula and demonstrated its ability to identify and quantify the regression dependence in multivariate categorical data with an ordinal response variable and categorical (nominal or ordinal) explanatory variables in an exploratory manner. This thesis explores their novel measure and the methodology behind it. In addition, we extend their work by proposing a model-based estimator of their measure. We conduct simulation studies to evaluate the performance of the model-free and model-based measure. Initial results demonstrated that model-based estimates of the measure from well-fitted models compared similarly to the model-free estimator of the measure, suggesting further exploration into the possibility of using the model-free estimator as a goodness of fit measure.

Acknowledgments

I am indebted to everyone who has assisted me along my thesis journey. First and foremost, I'd like to thank Shu-Min Liao for advising this thesis and pushing me to do what I thought was impossible. From my first statistics course here at Amherst to my last, she has supported me and enabled me to thrive.

I want to also acknowledge some of the other faculty here at Amherst that have a played a crucial part in my development as a budding statistician. Thank you to Nicholas Horton who has always advocated for me and provided me opportunities to learn and mature. Thank you to Yongheng Zhang who made math exciting and appealing to me again.

Lastly, I would like to express gratitude for my friends and family. Thank you to my partner for always providing warmth and encouragement during my lows and my highs. I'd like to thank my mom for never giving up on me and pushing me to try, try, try. I also want to thank my friends for not understanding the topic of my thesis but pretending to anyways and last, but not least, I thank my peers in the Mathematics and Statistics department for being on this journey with me.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

> All models are wrong, but some are useful.
>
> George E. P. Box.

Categorical data analysis with ordinal responses is important in a variety of fields, ranging from the social sciences to the public health sciences. In the social sciences, an ordinal response of interest might be the opinions of people. Examples might include opinions on government spending (i.e., (1) too high, (2) just enough, (3) too low) or the Likert psychometric response scale with 5 possible choices (i.e., (1) strongly disagree, (2) disagree, (3) neither agree nor disagree, (4) agree, (5) strongly agree). In the medical and public health sciences, it can be used to denote the severity of injuries or health risks. It is also important to note that ordinal variables should not be confused with nominal variables which are categorical variables with no intrinsic ordering. Examples of these include a person's ethnicity or a person's favorite brand of toothpaste. There exist many well established statistical methods like the Pearson χ^2 test of independence that treat the response variable as nominal, which means they do not utilize the order of the variables. But when we take into account the intrinsic ordering of the variables, our ordinal statistical analyses can give more powerful results than ones that ignore the ordinality (Agresti, 2010).

Wei & Kim (2021) explain that an important primary step in data analysis with ordinal responses is the exploration and examination of the various dependence structures among the random variables for exploratory and descriptive modeling. It is important for the researchers to identify dependency patterns, summarize the dependencies and obtain ideas on what may affect the ordinal variable. One of these dependence structures of interest is that of the regression dependence where we are interested in the relationship between an ordinal response variable and categorical explanatory variable(s).

Model-based approaches to modeling regression dependence in ordinal response data have been constructed and popular methods include the cumulative logit model, adjacent-categories logit model, and latent variable models. One could also utilize a range of model-free approaches to ordinal response data such as ordinal odds ratios and rank-based methods like Kendall's tau and Spearman's rank-based correlation. These model-free approaches do not require the explicit specification of the underlying dependence structure among the variables. However, some of the model-free methods mentioned above do not work well when one is interested in the regression dependence of multivariate ordinal data with an ordinal response variable. These methods are mainly designed to explore bivariate association between two ordinal variables and some of these methods treat the two ordinal variables symmetrically, meaning there exists no distinction between the response variable and the explanatory variable(s).

To address this problem, Wei & Kim (2021) propose a novel model-free data dependent measure, called the checkerboard copula regression association measure (CCRAM), to identify and quantify the regression dependence in multivariate categorical data with an ordinal response variable and categorical (nominal or ordinal) explanatory variables — based on the checkerboard copula, a certain type of joint distribution function. The checkerboard copula is just one type of copula model used when we are dealing with noncontinuous variables. As we will see in Chapter 2, in the continuous case, copula models are very useful in modeling the dependence structure between variables and their joint distribution. For instance, in the field of hydrology, Genest & Favre (2007) propose the use of copulas in exploring the pairwise dependence between characteristics of water such as depth, volume, and duration of flows.

This thesis, motivated and based on Wei & Kim (2021), aims to evaluate the performance of the CCRAM in a model-free manner and understand the methodologies used in its construction. As an extension of their work, we propose a model-based version of the CCRAM later on in Section 4.5.2 of Chapter 4 to assist in our evaluation of the model-free CCRAM. The difference between their model-free CCRAM and our proposed model-based CCRAM is in how we estimate the joint probabilities. The model-free CCRAM estimates joint probabilities directly from the data while the model-based CCRAM estimates the joint probabilities using a model. Because Wei & Kim (2021) demonstrated in their paper that the model-free CCRAM is able to identify and quantify the regression dependence without having to assume any assumptions about the underlying dependence structures, we are interested to see how a model-based version of the CCRAM compares. Our goal with this proposed method is to see if the model-free CCRAM can be used as a goodness of fit measure in ordinal response analyses.

The rest of the thesis is organized as follows. In Chapter 2, we provide a literature review on copulas in the continuous case and establish key theorems, concepts, and principles that make copulas useful for dependence modeling. In Chapter 3, we examine the challenges with copula modeling when we have discrete random variables. In Chapter 4, we introduce the checkerboard copula used when a discrete variable is present, the model-free CCRAM from Wei & Kim (2021), and our proposed model-based CCRAM. In Chapter 5, we simulate contingency tables under various conditions to evaluate the performance of both CCRAMs; a real world application is included in this chapter as well. In Chapter 6, we conclude with a summary of our findings and future work.

Chapter 2 Literature Review on Continuous Copulas & Dependence Measures

> Li's Gaussian copula formula will go down in history as instrumental in causing the unfathomable losses that brought the world financial system to its knees.
>
> Felix Salmon

2.1 Preliminary Concepts

Let $\mathbb{R}$ denote the real line $(-\infty, \infty)$ and $\mathbb{R}^2$ denote the real plane $(\mathbb{R}\times\mathbb{R})$. A rectangle in the real plane is the Cartesian product of two closed intervals: $[x_1, x_2] \times [y_1, y_2]$ where the vertices of the rectangle are $(x_1, y_1), (x_1, y_2), (x_2, y_1), (x_2, y_2)$. The unit interval is denoted by $\mathbb{I} = [0, 1]$ and so $\mathbb{I}^2 = \mathbb{I} \times \mathbb{I} = [0, 1]^2$ denotes the unit square. We will use the following two Lemmas throughout this chapter:

Lemma 1 (Probability Integral Transformation; Hofert et al. (2018), p.3)**.** Let F be a continuous distribution function and let $X \sim F$. Then the random variable $F(X)$ is a standard uniform random variable, i.e., $F(X) \sim U(0, 1)$.

Definition 1 (Quantile Function). $F^{\leftarrow}$ is the quantile function defined by:

$$F^{\leftarrow}(y) = \inf\{x \in \mathbb{R} : F(x) \geq y\}, \quad y \in [0,1].$$

Note that for continuous and strictly increasing distribution functions F, $F^{\leftarrow}$ is equivalent to the ordinary inverse, i.e., $F^{\leftarrow} = F^{-1}$. But if a margin is not strictly increasing, then it does not possess an inverse in the usual sense.

Lemma 2 (Quantile Transform; Hofert et al. (2018), p.4). Let $U \sim U(0,1)$ and let F be any distribution function. Then $F^{\leftarrow}(U) \sim F$.

2.2 A Motivating Example

Suppose we are handed two bivariate data sets, each consisting of 1000 independent observations from a bivariate random vector (X_1, X_2) and (Y_1, Y_2), respectively (displayed in Figure 2.1) and are asked to compare them in terms of their "dependence" between the variables, meaning how X_1 and X_2 relate compared to the relationship between Y_1 and Y_2. How would we go about approaching this question? One way we could do this is to calculate the Pearson correlation coefficient within each data set. After some calculations, we find that the correlation between X_1 and X_2 is approximately 0.69 while the correlation between Y_1 and Y_2 is approximately 0.6. This suggests that the dependence between X_1 and X_2 is stronger. However, it is important to remember that the correlation coefficient is only useful for capturing the linear dependence between the underlying variables. If we look at the right plot of Figure 2.1, we quickly notice that the relationship between Y_1 and Y_2 does not have a linear shape. Looking at the marginals on the side of the plots, we see that both X_1 and X_2 on the left plot have normal marginals which makes Pearson's appropriate. On the other hand, Y_2 appears to follow an exponential or highly right-skewed distribution.

Because the marginal distribution of Y_2 differs from the marginals of X_1 and X_2, we might be dubious about using the correlation coefficient to compare the dependence between X_1 and X_2 and the dependence between Y_1 and Y_2. Perhaps if the marginals were the same, comparisons can be made on fairer grounds.

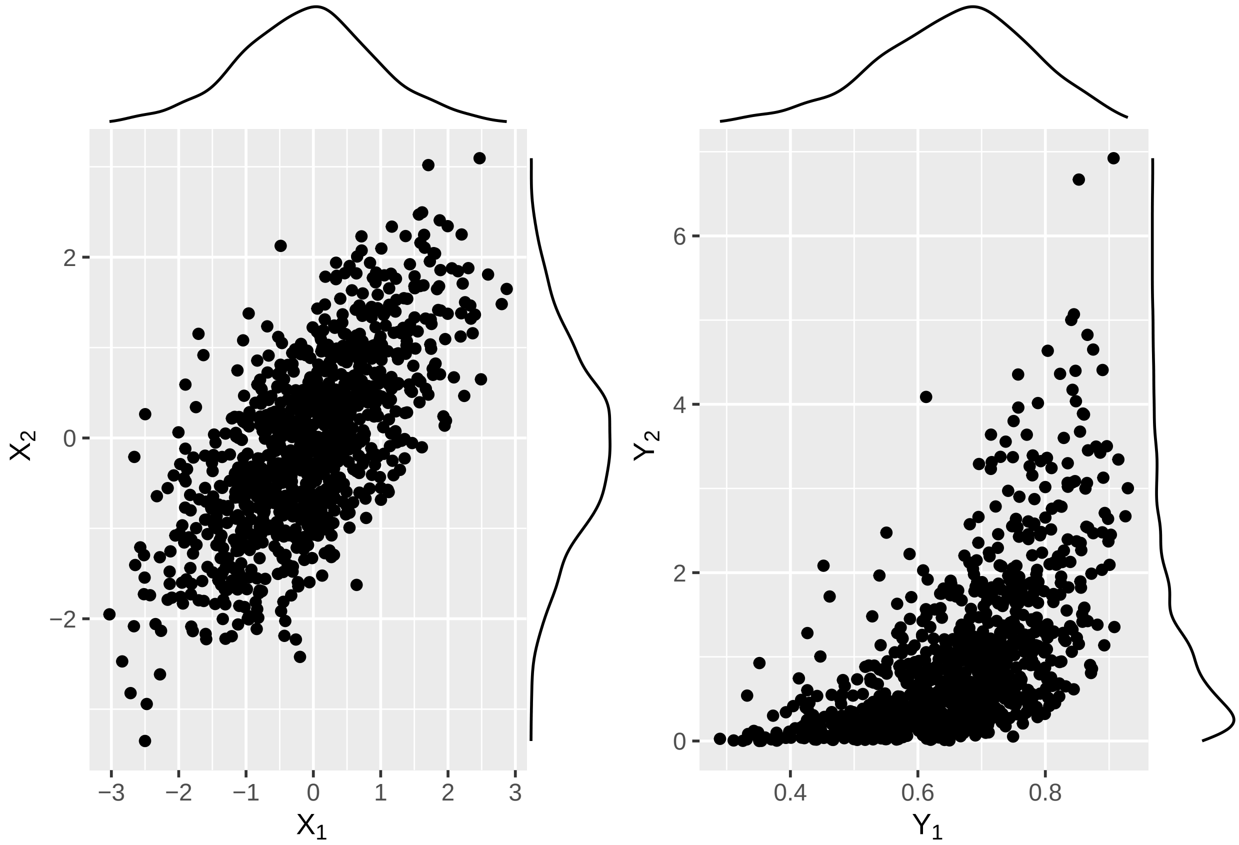

Figure 2.1: Scatter plots of 1000 independent observations of (X_1, X_2) and (Y_1, Y_2). Marginal densities are given along the sides.

One method of transforming the marginals is via Lemma 1. Suppose we know that $X_1 \sim N(0,1)$, $X_2 \sim N(0,1)$, $Y_1 \sim Beta(5,10)$, $Y_2 \sim Exp(1)$. Let F_1, F_2, G_1 and G_2 denote the distribution functions of X_1, X_2, Y_1 and Y_2 respectively. Knowing the distribution functions, we transform X_1, X_2, Y_1 and Y_2 into $F_1(X_1), F_2(X_2), G_1(Y_1)$ and $G_2(Y_2)$, all of which follow a Uniform(0,1) distribution according to Lemma 1. The transformed results are illustrated in Figure 2.2.

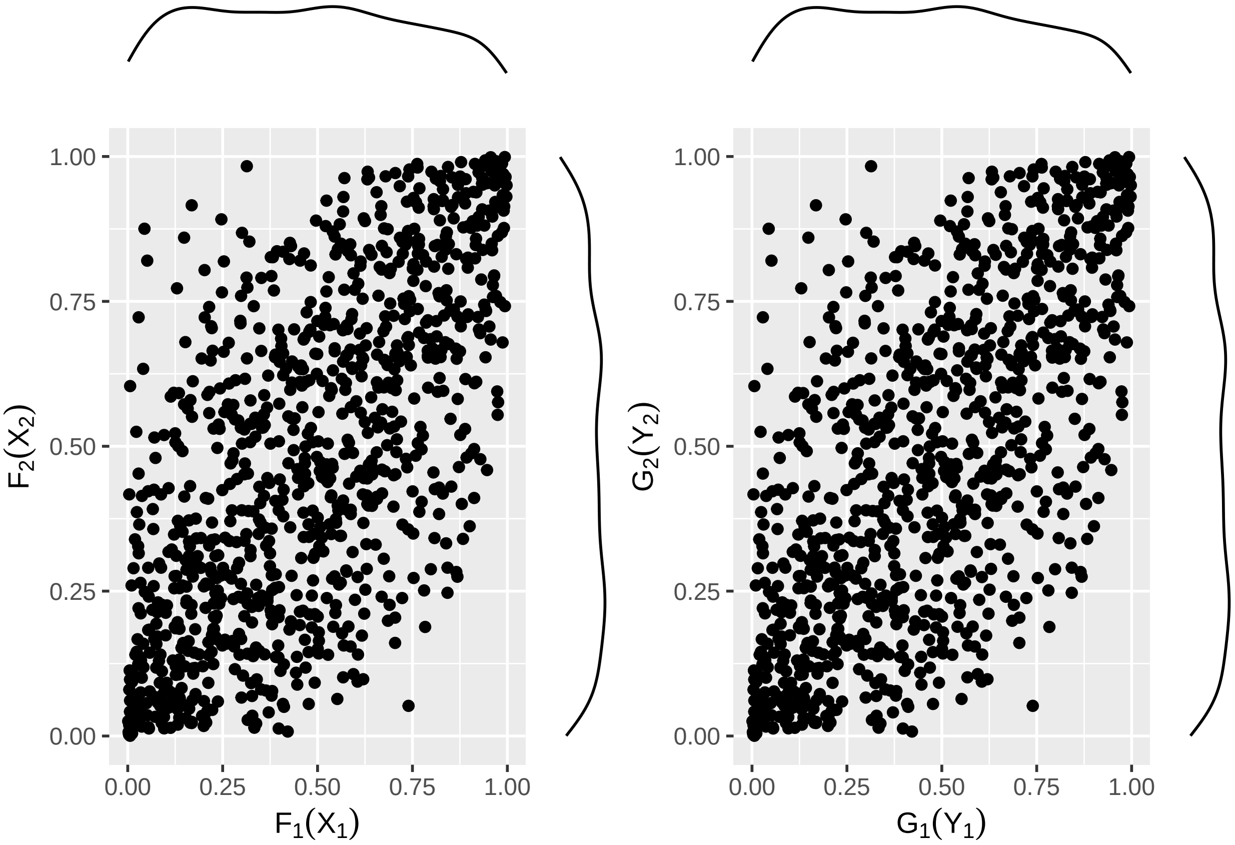

Figure 2.2: Scatter plots of 1000 independent observations of $(F_1(X_1), F_2(X_2)))$ and $(G_1(Y_1), G_2(Y_2))$. Marginal densities are given along the sides.

In Figure 2.2, both transformed data sets appear to be similar.[1] And now, for either dataset, we get a Pearson correlation coefficient of approximately 0.68, implying both bivariate associations have the same dependence.

We can consider another approach using Lemma 2. Instead of transforming both data sets to make all marginals uniform, we transform the second one to have standard normal marginals to match the first dataset. To do this though, we actually need to first apply Lemma 1 before applying Lemma 2. Recall that $Y_1 \sim G_1$ and $Y_2 \sim G_2$, where $G_1 = Beta(5, 10)$ and $G_2 = Exp(1)$. We first apply Lemma 1 and get $G_1(Y_1) \sim U(0, 1)$ and $G_2(Y_2) \sim U(0, 1)$. Then Lemma 2 tells us that applying the quantile transform using F_1^{-1} and F_2^{-1} on G_1 and G_2 results in $F_1^{-1}(G_1(Y_1)) \sim N(0, 1)$

[1] In fact, we constructed them in an identical manner.

and $F_2^{-1}(G_2(Y_2)) \sim N(0,1)$.

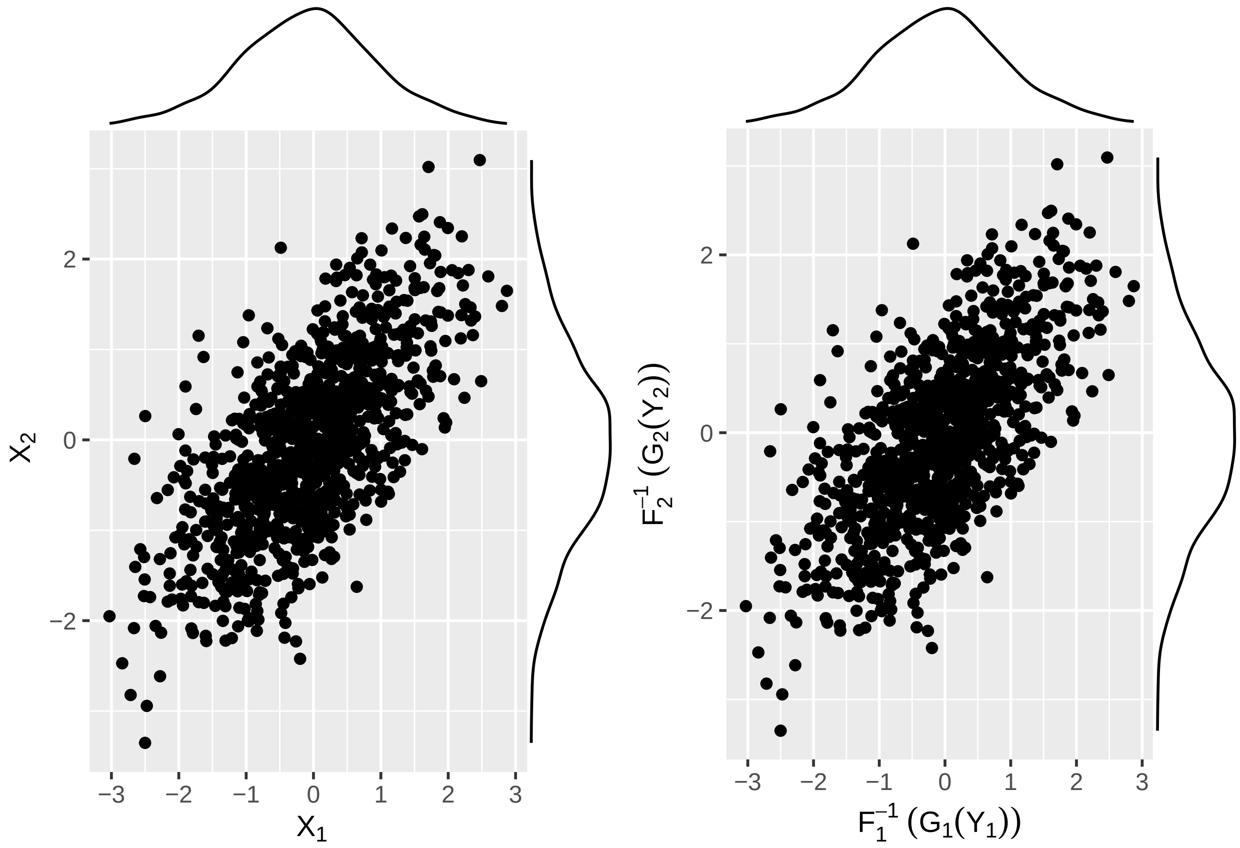

Figure 2.3: Scatter plots of 1000 independent observations of (X_1, X_2) and $F_1^{-1}(G_1(Y_1)), F_2^{-1}(G_2(Y_2))$. Marginal densities are given along the sides.

Using this second approach, the Pearson correlation coefficient between $F_1^{-1}(G_1(Y_1))$ and $F_2^{-1}(G_2(Y_2))$ is 0.69, the same as the correlation between X_1 and X_2. This flexibility is why copulas are so attractive. Copulas represent the idea that the dependence between components of a random vector should not be influenced by its marginal distributions. In other words, the statement, "(X_1, X_2) and (Y_1, Y_2) have the same dependence" can be thought of as "(X_1, X_2) and (Y_1, Y_2) have the same copula" (Hofert et al. 2018).

2.3 Copulas

Copulas allow the study of dependence of variables separate from their marginals. Informally, copulas are a particular kind of multivariate distribution function with standard uniform marginals. They can be thought of as functions that "couple" joint distribution functions to their marginal distribution functions.[2] More on this will be discussed in Section 2.6, but first, let's establish some more key preliminary definitions.

Definition 2. A 2-dimensional subcopula (2-subcopula) is a function $C^S : D_1 \times D_2 \to \mathbb{I}$ where $\{0, 1\} \subseteq D_i \subseteq \mathbb{I}$ for $i = 1, 2$ with the following characteristics:

- Grounded, i.e., : $C^S(u, 0) = 0 = C^S(0, v), \ \ \forall u \in D_1, \forall v \in D_2$
- $C^S(u, 1) = u, \ \ \forall u \in D_1$ and $C^S(1, v) = v, \ \ \forall v \in D_2$
- 2-increasing, i.e., : $C^S(u_2, v_2) - C^S(u_1, v_2) - C^S(u_2, v_1) + C^S(u_1, v_1) \geq 0$ where $u_1 \leq u_2$ and $v_1 \leq v_2$.

Definition 3. A 2-dimensional copula (2-copula) is a function $C : D_1 \times D_2 \to \mathbb{I}$ where $D_1 = \mathbb{I} = D_2$ with the following characteristics:

- Grounded, i.e., : $C(u, 0) = 0 = C(0, v), \ \ \forall u \in D_1, \forall v \in D_2$
- $C(u, 1) = u, \ \ \forall u \in D_1$ and $C(1, v) = v, \ \ \forall v \in D_2$
- 2-increasing, i.e., : $C(u_2, v_2) - C(u_1, v_2) - C(u_2, v_1) + C(u_1, v_1) \geq 0$ where $u_1 \leq u_2$ and $v_1 \leq v_2$.

Here are some remarks about the previous definitions:

[2] Fun fact: The name "copula" was chosen to emphasize how a copula "couples" a joint distribution function to its marginal distributions. In Latin, "copula" means to "link" or to "tie".

- These two definitions are quite similar but it is important to note that for copulas, the domain is the entire unit square and this occurs when X and Y are both continuous random variables. Moreover, subcopulas and copulas are the same when the marginal random variables are continuous. As we will see in Chapter 4, there is a way to extend the sub-copula to the whole unit square.
- In this thesis, we mostly consider the bivariate case when describing and defining various concepts but many definitions and theorems have analogous multivariate versions. However, one must be cautious when generalizing as there are exceptions. See section 2.10 in Nelsen (2006) for more details.
- It is worth noting that it is not absolutely necessary that copulas have standard uniform margins. Having standard uniform margins is just an easy way to work with copulas, but one could have used other margins too.
- A copula C is considered absolutely continuous if it admits a density and it admits a density if

  $$c(u,v) = \frac{\partial^2}{\partial u \partial v} C(u,v), \quad (u,v) \in (0,1)^2,$$

  exists and is integrable.
- We can also derive the conditional cumulative distribution function from the copula itself:

  $$P(V \leq v | U = u) = \frac{\partial}{\partial u} C(u,v).$$

2.4 Bivariate Copula Examples

Let's see some examples of bivariate copulas to ground our discussion. One of the simplest copulas is the independence copula:

$$\Pi(u, v) = u \times v, \quad (u, v) \in [0, 1]^2 \tag{2.1}$$

which is the distribution function of the random vector (U, V) where $U \sim Unif(0, 1)$ and $V \sim Unif(0, 1)$ are independent. As the name alludes, two random variables X and Y are independent if and only if the copula $C = \Pi$. We use the R package `copula` written by Hofert et al. (2020) to plot the surface plot and contour plot for an independence copula.

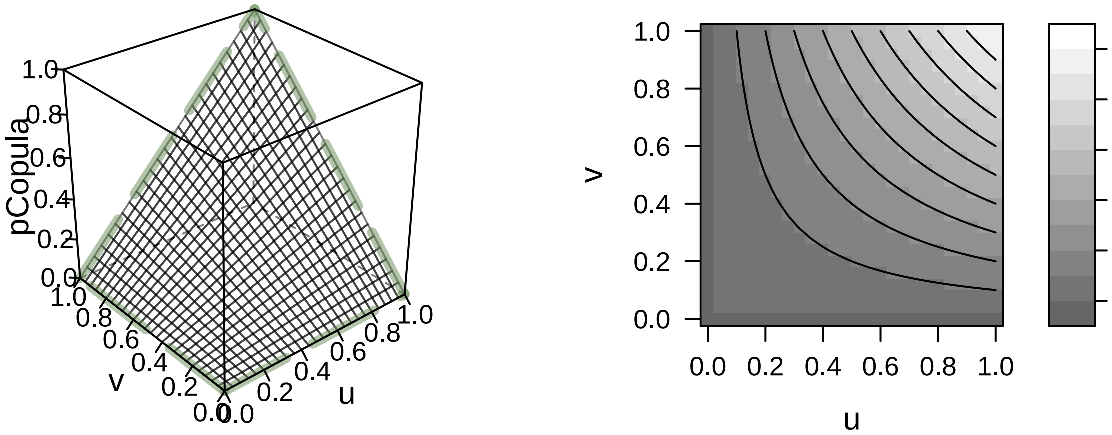

Figure 2.4: Surface plot (left) and contour plot (right) of an independence copula.

The independence copula is indeed a copula as it has the following properties:

- Grounded: For all $(u, v) \in [0, 1]^2$ where $u = 0$, $C(u, v) = C(0, v) = 0$. A similar case can be made for $v = 0$.
- For all $(u, v) \in [0, 1]^2$ where $u = 1$, $C(u, v) = C(1, v) = v$. A similar case can be made for $v = 1$.

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

There are also copulas that belong to parametric families. One of those families is the Frank family of copulas. It is parametrized by $\theta \in \mathbb{R} \setminus \{0\}$ and the copulas are defined by:

$$C_\theta^F = -\frac{1}{\theta} \log\left(1 + \frac{(\exp(-\theta u) - 1)(\exp(-\theta v) - 1)}{\exp(-\theta) - 1}\right), \quad (u, v) \in [0,1]^2 \tag{2.2}$$

where $C_0^F = \Pi$ due to the fact that it converges to Π as $\theta \to 0$. The parameter θ in Equation (2.2) controls the dependence between the components of $(U, V) \sim C_\theta^F$. Using the R package `copula`, we plot the surface plot and contour plot of the Frank copula and it's density given $\theta = -9$ as shown in Figure 2.5.

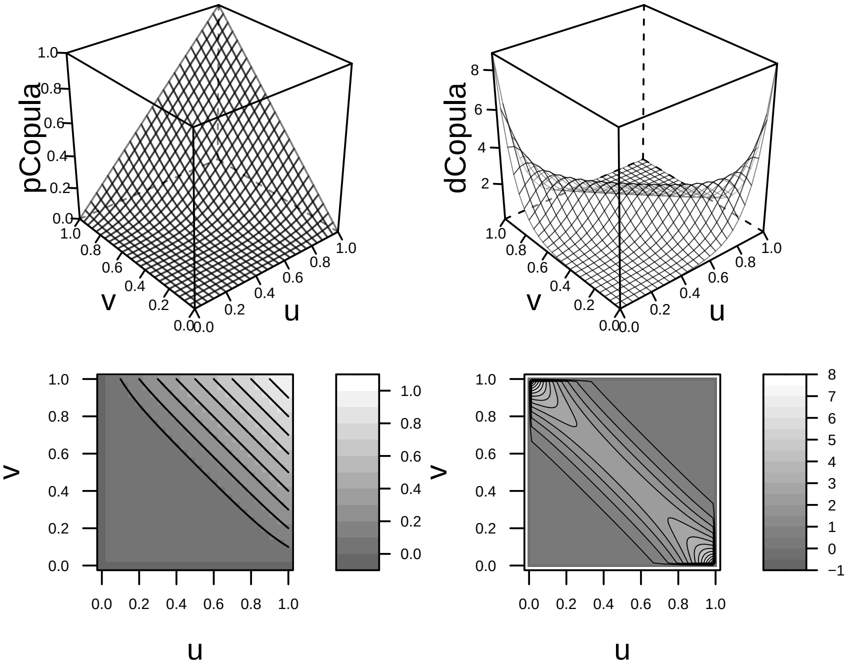

Figure 2.5: Surface plot and contour plots of C_θ^F (left) and its corresponding density (right) for $\theta = -9$.

In addition, Figure 2.6 illustrates how θ controls the dependence by sampling 1000 independence observations from C_θ^F given $\theta \in \{-9, 0, 9\}$ using the `rCopula()` function within the package.

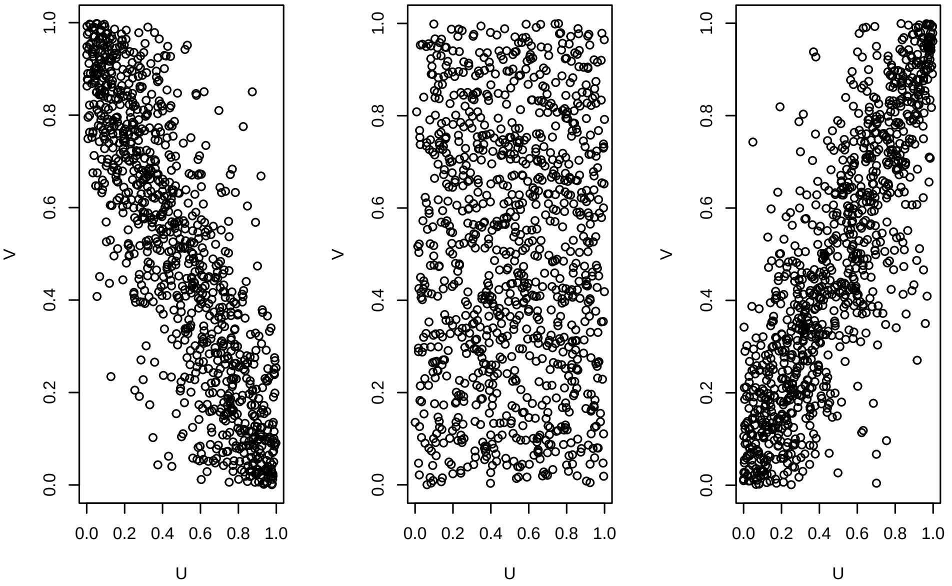

Figure 2.6: 1000 independent observations of $(U, V) \sim C_\theta^F$ for $\theta = -9$ (left), $\theta = 0$ (middle), $\theta = 9$ (right).

Observe in Figure 2.6, that by changing θ from -9 to 0 to 9, the components of $(U, V) \sim C_\theta^F$ went from negatively dependent in the sense that larger values of U tend to be associated with smaller values of V (left plot) to positively dependent in the plot on the right, meaning larger values of U tend to be associated with larger values of V.

Table 2.1: Families of Bivariate Copulas

Copula Family	Distribution Function	Parameter
Clayton	$C_\theta(u, v) = \max\{u^{-\theta} + v^{-\theta} - 1, 0\}^{-\frac{1}{\theta}}$	$\theta \in (0, \infty)$
Farlie-Gumbel-Morgernstern	$C_\theta(u, v) = uv(1 + \theta(1 - u)(1 - v))$	$\theta \in [-1, 1]$
Frank	$-\frac{1}{\theta}\log(1 + \frac{(\exp(-\theta u)-1)(\exp(-\theta v)-1)}{\exp(-\theta)-1})$	$\theta \in \mathbb{R} \setminus \{0\}$
Gaussian	$C_\theta(u, v) = \Phi_\theta\{\Phi^{-1}(u), \Phi^{-1}(v)\}$	$\theta \in (-1, 1)$

Table 2.1 lists some other commonly used parametric copula families used in modeling dependence structures. As another example, Figure 2.7 shows various plots regarding a Gaussian copula where $\theta \approx 0.707$. Top left is the surface plot of the density of the Gaussian copula, top right is the contour plot of the Gaussian copula, bottom left is the contour plot of the density, and bottom right is a scatter plot of a generated sample of 1000 points from the copula.

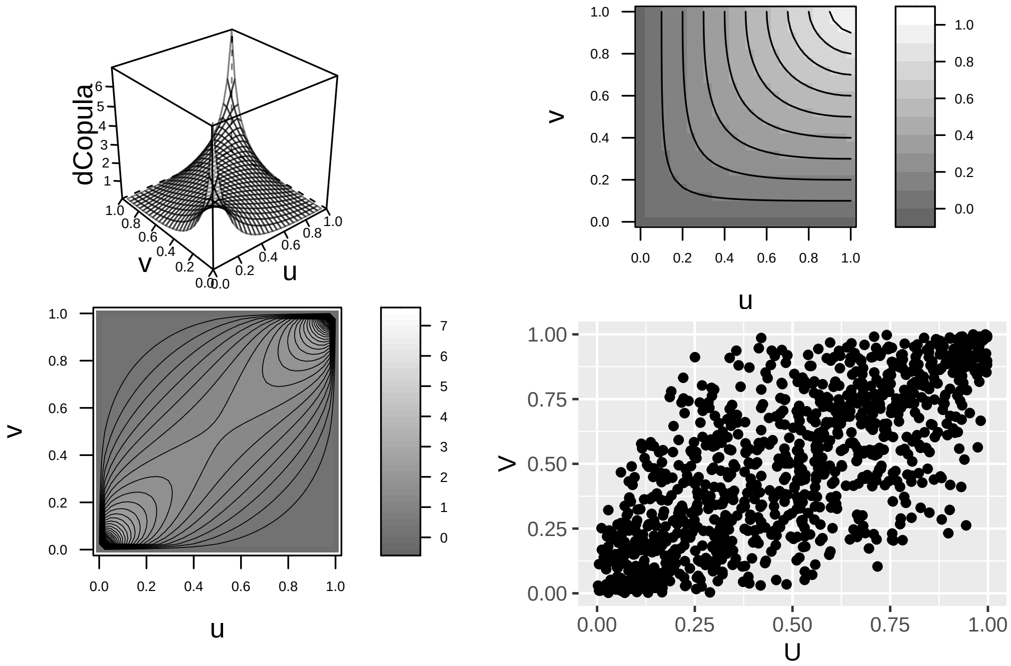

Figure 2.7: Surface plot of the density of a normal copula with $\theta \approx 0.707$ (top left), contour plot of the normal copula (top right), of its density (bottom left), and a scatter plot of 1000 observations from the normal copula (bottom right).

Observe the elliptical shape of the generated observations in the bottom right plot of Figure 2.7. The beauty behind various copulas is that it provides us great flexibility in modeling dependence structures. There exists many different copulas, along with numerous copula-specific transformations and constructions (e.g. rotations, mixtures, Khoudraji's Device), that one can use to capture various dependence structures, symmetric or not. See Chapter 3 of Hofert, Kojadinovic, Martin, & Yan (2018) for more copulas.

2.5 Fréchet-Hoeffding Bounds

One of the key concepts behind copulas is the following theorem which can be attributed to Hoeffding (1940) and Fréchet (1951). This theorem states that any copula C is pointwise bounded below by the lower Fréchet-Hoeffding Bound W and above by the upper Fréchet-Hoeffding Bound M.

Theorem 1 (Fréchet-Hoeffding Bounds; Hofert et al. (2018), p.19)**.** For any 2-dimensional copula C,

$$W(u,v) \leq C(u,v) \leq M(u,v), \quad u,v \in [0,1] \tag{2.3}$$

where $W(u,v) = \max\{u+v-1, 0\}$ and $M(u,v) = \min\{u,v\}$.

There also exists a stochastic representation of the above theorem. It can be verified that

$$(U, 1-U) \sim W \;\text{ and }\; (U,U) \sim M \tag{2.4}$$

where $U \sim Unif(0,1)$ (Hofert et al. 2018). To be more precise, Theorem 1 is considered the "analytic" version whereas Equation (2.4) is the "stochastic" representation of that same idea. This stochastic representation is particularly helpful in simulations, especially for generating random samples from copulas. For instance, the following example code can be used to generate a random sample from the W and M copulas; corresponding results are shown in Figure 2.8.

```r
set.seed(713) # reproducibility
par(mfrow = c(r = 1, c = 2)) # 2x2 grid
M <- runif(100) # sample 100 from a standard uniform
plot(cbind(M, 1 - M), xlab = "U", ylab = "V") # W
plot(cbind(M, M), xlab = "U", ylab = "V") # M
```

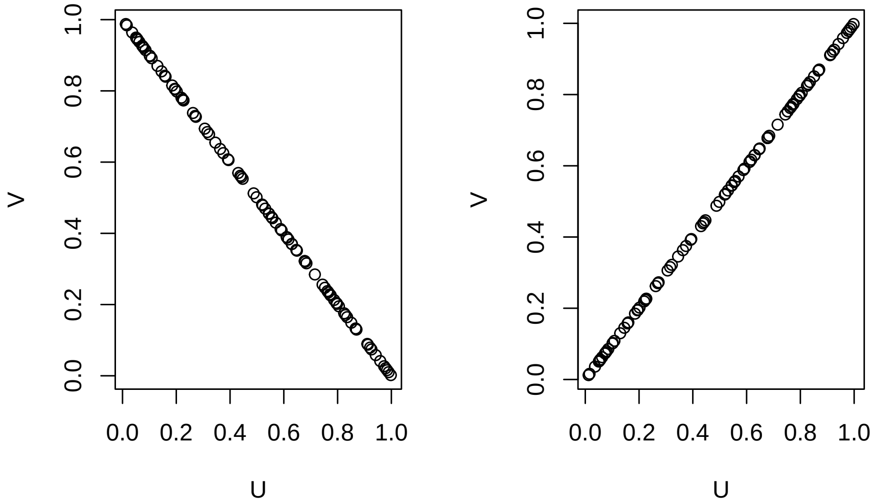

Figure 2.8: Scatterplot of n = 1000 independent observations from W (left) and M (right).

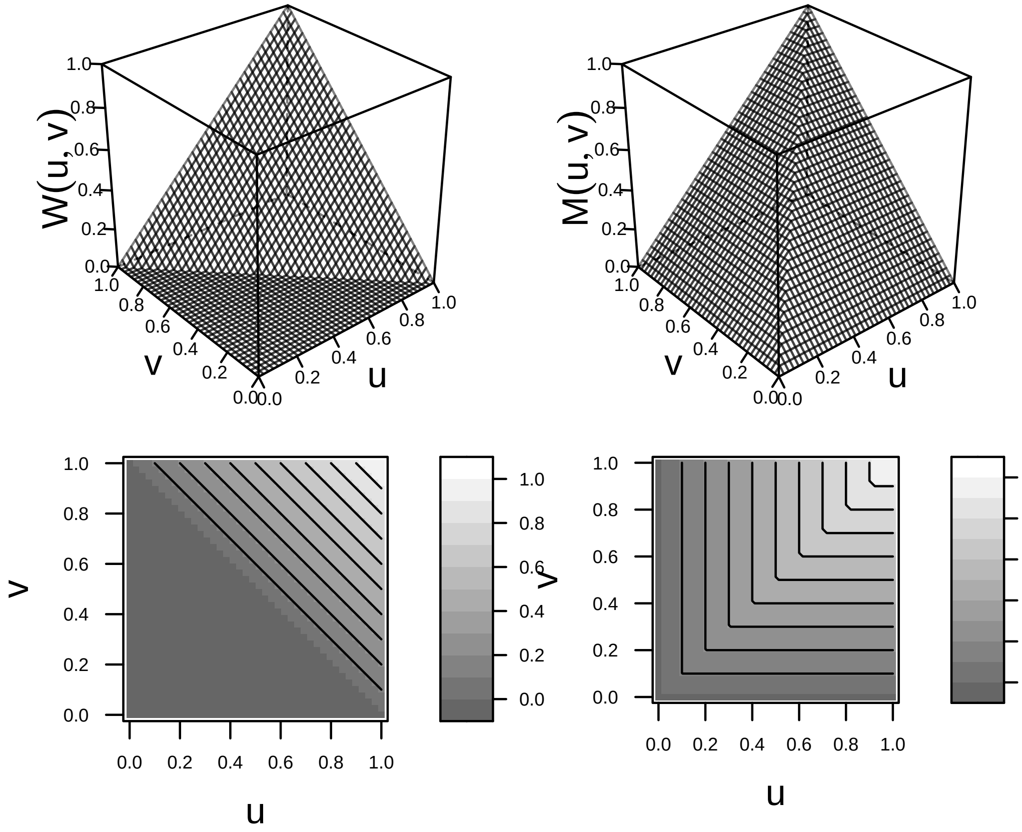

Figure 2.9: Surface plots (top) and contour plots (bottom) of W (left) and M (right).

The wireframe and contour plots of W and M are displayed in Figure 2.9. On the left is the W copula and on the right is the M copula. Note that the graph of all subcopulas and therefore copulas, lies between these two surfaces, $z = W(u, v)$ and $z = M(u, v)$. Consider the Frank copulas shown in Figure 2.6 as an example. It can be shown that, as the parameter θ increases, the scatter plot of a random sample from the corresponding Frank copula will get closer to the right plot of Figure 2.9 which represents the M copula. On the other hand, when θ decreases, the scatter plot will get closer to the left plot of Figure 2.9 which represents the W copula. In other words, the Frank copula family is bounded between the W and M copulas.

In addition, W is known as the countermonotone copula and M is known as the comonotone copula. Moreover, the dependence between the components of (U, U) modeled by M is referred to as the perfect positive dependence (if one component increases, the other increases almost surely, with probability 1), On the other hand, the dependence between the components of $(U, 1 - U)$ modeled by W is referred to as perfect negative dependence (if one component increases, the other decreases almost surely, with probability 1). Note that for W, this notion of perfect negative dependence cannot be extended to 3 or more dimensions. If two components of a random vector are perfectly negative dependent, then they both cannot be perfectly negative dependent with a third component.

2.6 Sklar's Theorem

This next theorem, attributed to Sklar (1959), is the real bread and butter of copula theory as it elucidates the relationship between a multivariate joint distribution and its univariate margins.

Theorem 2 (Sklar's Theorem)**.**

Part 1. Let H be a joint distribution function with univariate marginal distribution functions F and G. Then there exists a copula C such that for all x, y in $\mathbb{R}$,

$$H(x,y) = C(F(x), G(y))). \tag{2.5}$$

When F and G are continuous, C is unique but otherwise C is uniquely determined on $\text{Ran}F \times \text{Ran}G$ where $\text{Ran}F$, $\text{Ran}G$ denote the range of F and G respectively. The copula C is given by:

$$C(u,v) = H(F^{\leftarrow}(u), G^{\leftarrow}(v)), \quad (u,v) \in \text{Ran}F \times \text{Ran}G. \tag{2.6}$$

Part 2. Conversely, given a bivariate copula and univariate marginal distribution functions F and G. H defined above is a 2-dimensional distribution function with margins F and G.

Here are some remarks stemming from this theorem:

1.) From Sklar's theorem, copulas are bivariate distribution functions which combine univariate marginal distribution functions to create a 2-dimensional distribution function H. This is what it means for copulas to "couple" or link multivariate distribution functions to their univariate margins. Moreover, this implies that the univariate margins and the multivariate dependence structure can be separated and that the dependence structure is characterized by the copula C.

2.) If F and G are continuous, then the copula C is unique, otherwise, C is uniquely defined on $\text{Ran}F \times \text{Ran}G$. When dealing with continuous random variables, there is only one copula that characterizes the dependence structure; this is why copulas are often used when concerning continuous random vectors.

3.) Sklar's Theorem can be used to verify that a random vector has a continuous distribution function H if and only if it has continuous univariate marginal distributions.

Part 1 of Theorem 2 is often used in statistical applications. For a given continuous distribution function, part 1 implies the uniqueness of the underlying copula which is unknown. But because it is unique, it justifies its estimation from data and once we estimate the margins and the copula, we can couple them as in part 1 to obtain the estimated multivariate distribution function. Part 2 is also of interest to those who might need to create flexible multivariate distribution functions with given univariate margins.

Let's look at two examples to see the utility of copulas.

2.6.1 Sklar's Theorem: Decomposition

Using Equation (2.6) from Part 1 of Sklar's Theorem, we can create copula families from existing multivariate distribution families. Suppose we started with a Normal bivariate distribution, H, and we know that the marginal distributions of H are univariate normal. We can use this to help us construct the normal (Gaussian) copula family. We demonstrate with the following code from `mvtnorm` (Genz et al. (2021)) and `copula` where we show that the same output was returned whether we apply the bivariate normal distribution function (using `pmvnorm()`) or the Normal copula distribution function (using `pCopula()`).

```r
set.seed(713) # reproducibility
d <- 2 # dimension
rho <- 0.4 # off-diag entry of the correlation matrix
u <- runif(d) # generate a random point
x <- qnorm(u) # applying the quantile transform
# bivariate normal distribution
mvtnorm::pmvnorm(
  upper = x, corr = matrix(c(1, rho, rho, 1), nrow = 2),
  keepAttr = FALSE
)
```

```
[1] 0.00338
```

```r
# normal copula
nc <- normalCopula(rho)
copula::pCopula(u, copula = nc)
```

```
[1] 0.00338
```

2.6.2 Sklar's Theorem: Composition

We can use the second part of Sklar's Theorem to generate a variety of multivariate distribution functions from a given copula C. To this end, we employ the `copula` package and create two different distribution functions.

```r
H1 <- copula::mvdc(fgmCopula(1),
  margins = c("beta", "exp"),
  paramMargins = list(list(shape1 = 7, shape2 = 3), list(rate = 1))
)

H2 <- copula::mvdc(fgmCopula(1),
  margins = c("norm", "norm"),
  paramMargins = list(list(mean = 3, sd = 2), list(mean = 0, sd = 1))
)
```

Here, we use the `mvdc()` function to first create a bivariate distribution function, H_1 with a Farlie-Gumbel-Morgenstern copula and Beta(7, 3) and Exp(1) marginals and then H_2 with a Farlie-Gumbel-Morgenstern copula and Normal(3, 2) and Exp(1) marginals. Figure 2.10 shows the scatter plots of a sample of 1000 observations from each of H_1 and H_2.

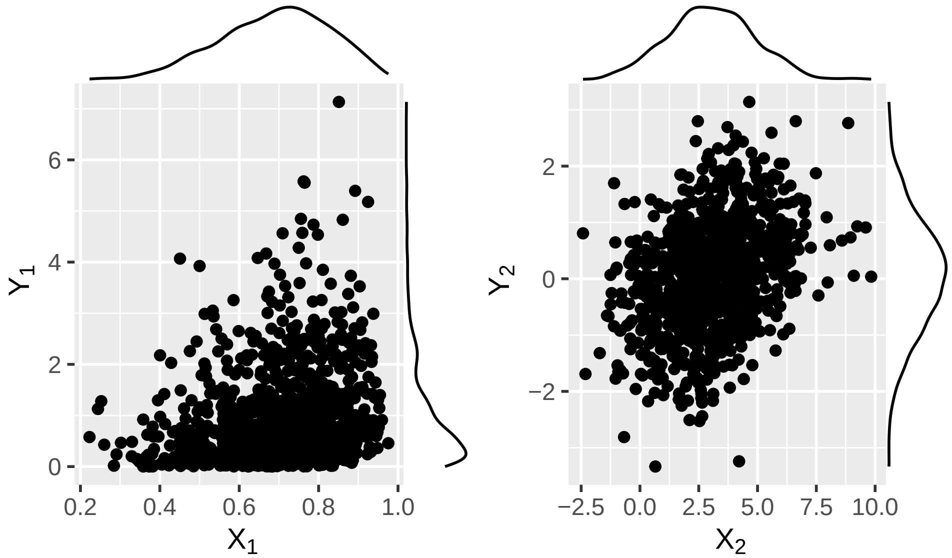

Figure 2.10: Scatterplots of 1000 observations from two dfs H_1 and H_2, created using a FGM-copula with Beta(7,3) and Exp(1) marginals (left) and Normal(3,2) and Exp(1) marginals (right).

From this demonstration, one can see that for a given copula, it is very easy and flexible to create a collection of bivariate or multivariate distributions with whatever marginal distributions we desire!

Similar to the Fréchet-Hoeffding Bounds, we present a stochastic analog of Sklar's Theorem for continuous random vectors that will be used in later sections.

Lemma 3 (Stochastic Analog of Sklar's Theorem; Hofert et al. (2018), p.35)**.** Let (X, Y) be a bivariate random vector with continuous univariate marginal distribution functions F and G. Then (X, Y) has a copula C if and only if $(F(X), G(Y)) \sim C$.

Given $(X, Y) \sim H$ with continuous univariate margins F and G, Lemma 3 enables the construction of a random vector $(U, V) \sim C$ where C is the underlying unique copula C. This can be seen as the stochastic analog of Part 1 of Theorem 2. On the other hand, given $(U, V) \sim C$ and univariate marginal dfs F and G, Lemma 3 tells us we can construct $(X, Y) \sim H$ with copula C through:

$$(F^{\leftarrow}(U), G^{\leftarrow}(V)) \sim H.$$

And this can be seen as the stochastic analog of Part 2 of Theorem 2.

2.7 Invariance Principle

Theorem 3 (Invariance Principle; Hofert et al. (2018), p.39)**.** Let $(X, Y) \sim H$, with continuous marginal distribution functions F, G and copula C. If T_X, T_Y are strictly increasing transformations on $\text{Ran}X, \text{Ran}Y$ respectively, then $(T_X(X), T_Y(Y))$ also has copula C.

Theorem 3 states that copulas are invariant under strictly increasing transformations on the ranges of the underlying random variables. Notice that for any 2-dimensional continuous random vector with strictly increasing marginals F and G, then it is clear that the invariance principle applies when $F = T$. It also applies to any 2-dimensional random vector with standard uniform marginal distributions when the quantile functions are strictly increasing.

Example 1 (From a bivariate normal distribution, to a normal copula, to a meta-normal model)**.** To demonstrate the invariance principle, we generated a sample of 1000 independent observations of (X, Y) that follows a standard bivariate normal distribution with $\rho = 0.8$ on the off-diagonal entries. Since $X \sim N(0, 1)$ and $Y \sim N(0, 1)$, we apply to each component sample the corresponding distribution function and Lemma 3 tells us that we have a sample from a normal copula, which we denote as C_ρ. Again by Lemma 3, we can apply our quantile functions of interest to each component sample in order to obtain multivariate observations from a meta-normal model which is a distribution function obtained from a given normal copula.

Suppose our quantile functions of interest were the quantile function of the Beta distribution with the following parametrization, $\alpha = 10$ and $\beta = 3$ and the quantile function of the Exponential distribution with $\lambda = 2$. By applying one quantile function to each component sample, we thus have observations from a multivariate distribution (meta-normal model). And Theorem 3 tells us that the copula remains the same. Figure 2.11 illustrates this idea. On the left is the scatter plot of 1000 independent observations from a bivariate normal distribution. In the middle, we have the corresponding sample after applying the Probability Integral Transform from a normal copula. And on the right, once we applied our quantile functions, we have a sample from a meta-normal model.

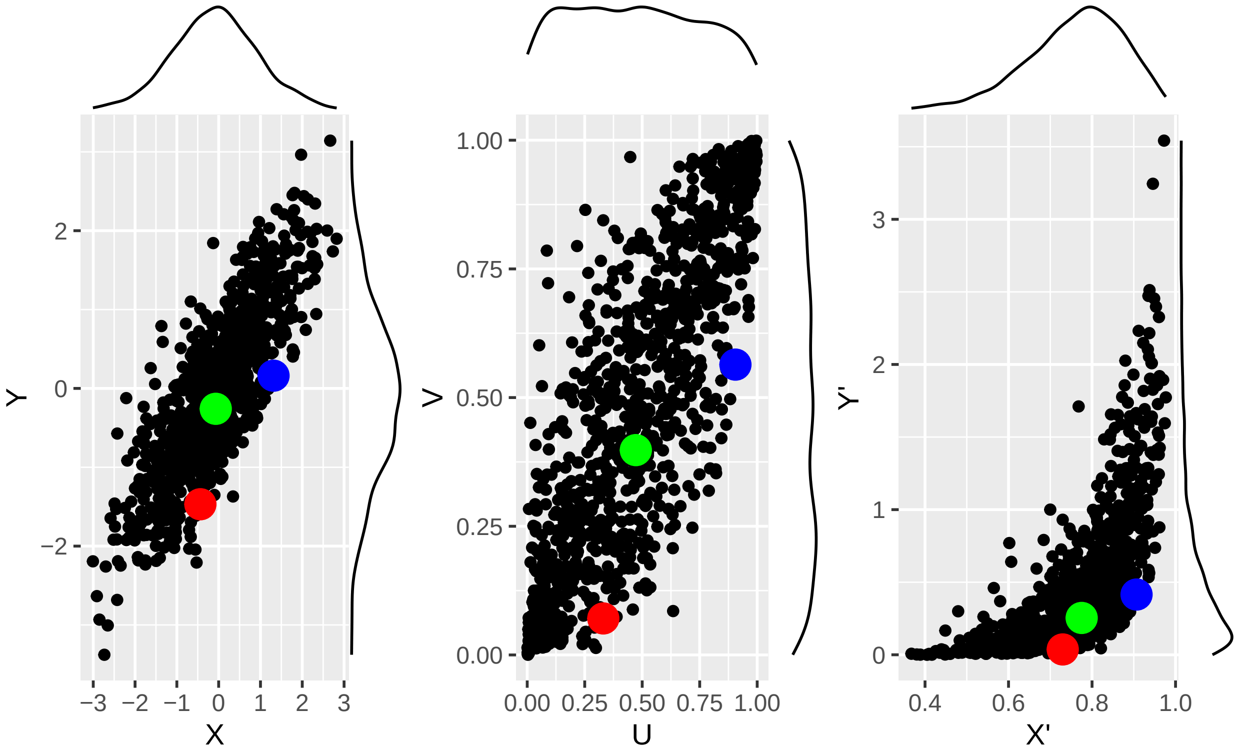

Figure 2.11: Scatterplot of 1000 independent observations from a bivariate normal distribution (left), corresponding sample after applying the Probability Integral Transform from a normal copula (middle), and quantile transformed meta-normal sample (right).

Notice that the location of the three highlighted points in Figure 2.11 relative to one another does not change. Along the X-axis, from left to right it is always Red, Green, Blue and on the Y-axis, from bottom to top it is Red, Green, Blue. This shows that componentwise, their ranks remain the same. Componentwise ranks were not affected by the transformation since ranks are invariant to strictly increasing transformations (the normal distribution function and the quantile function of the beta and exponential distribution are all strictly increasing).

In summary, the Fréchet-Hoeffding Bounds, Sklar's Theorem, and the Invariance Principle in tandem make copulas extremely useful in joint modeling. We are able to create all sorts of multivariate distributions with different marginal distribution functions. Moreover, our choice of marginals does not change the copula nor does it affect the underlying dependence structure, meaning no information about the dependence structure lies within the marginals but with the copula C. Moreover, because the copula C remains scale invariant under strictly increasing transformations, non-parametric measures of dependence such as Kendall's tau and Spearman's rho can be shown to be functions of just the copula itself.

2.8 Measures of Association

Since we now have a comprehensive idea about how copulas can model the dependence structure by separating the joint distribution function from its marginals distribution functions, it is desirable to quantify the dependence between two random variables X and Y. In statistics, we often do this using numerical summaries called measures of association. The most popular one is the Pearson correlation coefficient. It is probably the first association measure most students learn in an Introductory Statistics course and can be useful in the right situation. However, it has certain limitations.

2.8.1 Pearson's Correlation and its Fallacies

Definition 4 (Pearson correlation coefficient)**.** Given a random vector (X, Y) with finite variances, then:

$$\operatorname{Cor}(X,Y) = \frac{\operatorname{Cov}(X,Y)}{\sqrt{\operatorname{Var}(X)}\sqrt{\operatorname{Var}(Y)}} = \frac{\mathbb{E}((X-\mathbb{E}(X))(Y-\mathbb{E}(Y)))}{\sqrt{\mathbb{E}((X-\mathbb{E}(X))^2)\sqrt{\mathbb{E}((Y-\mathbb{E}(Y))^2)}}}. \tag{2.7}$$

Here are some well-known properties of the Pearson correlation coefficient:

1. $-1 \leq \operatorname{Cor}(X,Y) \leq 1$.

2. $|\operatorname{Cor}(X,Y)| = 1$ if and only if there exist $a, b \in \mathbb{R}$ where $a \neq 0$ such that $Y = aX + b$ almost surely with $a < 0$ if and only if $\operatorname{Cor}(X,Y) = -1$, and $a > 0$ if and only if $\operatorname{Cor}(X,Y) = 1$. In either case, X and Y are said to be perfectly linearly dependent.

3. If X and Y are independent, then $\operatorname{Cor}(X,Y) = 0$.

4. It is invariant under strictly increasing linear transformations.

It becomes clear that the Pearson correlation coefficient is a measure of linear dependence and it is really only useful for elliptical distributions like bivariate normal distributions. There are also other shortcomings to the Pearson correlation coefficient that we briefly list below.

1. Existence: $Cor(X,Y)$ does not exist for every random vector (X,Y).

2. Invariance: $Cor(X,Y)$ is not invariant under strictly increasing transformations on $\operatorname{Ran}(X), \operatorname{Ran}(Y)$.

3. Uniqueness: The marginal distributions and the correlation coefficient do not uniquely determine the joint distribution.

4. Uncorrelatedness implies independence: $Cor(X, Y) = 0$ does not necessarily imply X and Y are independent.

5. Attainability: Given marginal distribution functions F and G, not every $Cor(X, Y) \in [-1, 1]$ can be attained by choosing an appropriate copula for (X, Y).

Hofert et al. (2018) summarizes the main limitations of the Pearson correlation coefficient as follows:

1. The correlation does not exist for all random vectors, only those with finite second moments.

2. Correlation depends on the marginal distribution functions, meaning it cannot be expressed in terms of the unique underlying copula alone.

3. Correlation is invariant under strictly increasing linear transformations, not under strictly increasing transformations in general.

2.8.2 Rank-Based Correlation Measures

Since the Pearson correlation coefficient might not cut it, what can we turn to? Because of Theorem 3, Nešlehová (2007) writes

> “If the random variables under study have continuous distribution functions, the corresponding copula is unique and remains the same if the random variables are (almost surely) subject to strictly increasing transformations, such as the change of scale or location. As monotonic dependence also has this invariance property… it can be determined from the corresponding copula alone. Consequently, concordance measures like

> Kendall's tau and Spearman's rho can be expressed solely in terms of the corresponding copula."

So let's turn our attention to Spearman's rho and Kendall's tau, two well-known rank-based measures of association which are invariant under strictly increasing transformations. Because rank-based correlation coefficients only depend on the underlying copula C in the case of continuous random vectors, these measures overcome many of the limitations that the Pearson correlation coefficient faces.

Definition 5 (Spearman's Rho)**.** Let (X, Y) be a bivariate random vector with continuous marginal distribution functions F and G. The population version of Spearman's rho is defined by

$$\rho_s = \rho_s(X, Y) = Cor(F(X), G(Y)). \tag{2.8}$$

One may interpret Spearman's rho as the linear correlation coefficient of the random vector $(F(X), G(X))$ which can be obtained by applying the Probability Integral Transform (Lemma 1). It is clear that Spearman's rho only depends on the underlying copula C after transformation.

Definition 6 (Kendall's Tau)**.** Let (X, Y) be a bivariate random vector with continuous marginal distribution functions F and G and let (X', Y') be an independent copy of (X, Y). The population version of Kendall's tau is defined by

$$\tau = \tau(X, Y) = \mathbb{E}(\text{sign}((X - X')(Y - Y')))$$

where sign(x) denotes the sign of x, i.e., sign(x) $= \begin{cases} -1, & \text{if } x < 0, \\ 0, & \text{if } x = 0, \\ 1, & \text{if } x > 0. \end{cases}$

These rank-based measures are also known as measures of concordance. Consider two points in $\mathbb{R}^2$, (x_1, y_1) and (x_2, y_2). The points are considered concordant if $(x_1 - x_2)(y_1 - y_2) > 0$ and discordant if $(x_1 - x_2)(y_1 - y_2) < 0$. Using this idea and the definition of Kendall's tau, we can rewrite Kendall's tau as: $\tau(X, Y) = P((X_1 - X_2)(Y_1 - Y_2) > 0) - P((X_1 - X_2)(Y_1 - Y_2) < 0)$ where it is seen as the probability of concordance minus the probability of discordance. Similarly, we can write Spearman's rho as:

$$\rho(X, Y) = 3P((X_1 - X_2)(Y_1 - Y_2) > 0) - P((X_1 - X_2)(Y_1 - Y_2) < 0).$$

We can also represent Spearman's rho and Kendall's tau in terms of the underlying copula C. Let (X, Y) be a bivariate random vector with continuous marginal distributional functions and copula C. Then we can define:

$$\tau = \tau(C) = 4 \int_{[0,1]^2} C(u, v) dC(u, v) - 1 \tag{2.9}$$

$$\rho_s = \rho_s(C) = 12 \int_{[0,1]^2} C(u, v) du dv - 3. \tag{2.10}$$

When X and Y are continuous, it can be shown that $\tau(C) = \tau(X, Y)$ and $\rho(C) = \rho(X, Y)$ (Nelsen, 2006). If one of these measures is -1 or 1, then the copula C must either be W or M. If it is 0, then the copula must be the independence copula. In addition, these measures always exist, are invariant under strictly increasing transformations, and can reach any value in $[-1, 1]$.

Rank-based measures of dependence and the Pearson correlation coefficient are not the only type of dependence measures out there. There are various measures that look at tail dependence or quadrant dependence. See Chapter 5 of Nelsen (2006) for other ways to measure dependence.

2.9 Estimation

In the continuous case, we know that copulas characterize the dependence structure and that there only exists one unique copula so a natural question a Statistician might ask next is "how do we estimate the copula?" Copula estimation methods can be parametric, semiparametric, or nonparametric. We will briefly introduce some methods below, but won't go into too much detail. Hofert, Kojadinovic, Martin, & Yan (2018) goes more in depth for those that are interested in the various estimation techniques.

First, let's assume that we have a random sample, $(X_1, Y_1), \cdots, (X_n, Y_n)$, of n bivariate random vectors with a bivariate distribution function H and continuous univariate margins F and G. By Theorem 2, there exists a unique copula C such that

$$H(x, y) = C(F(x), G(y)), \ \ (x, y) \in \mathbb{R}^2. \tag{2.11}$$

2.9.1 Estimation Under a Parametric Assumption on the Copula

We can estimate under a parametric assumption on the copula where we assume that a copula C belongs to some continuous parametric family of copulas

$$\mathcal{C} = \{C_\theta : \boldsymbol{\theta} \in \Theta\}, \tag{2.12}$$

where Θ is the parameter space and is a subset of $\mathbb{R}^d$ where $d \geq 1$. Under this parametric assumption, we assume there exists some $\theta \in \Theta$ such that $C = C_\theta$. So in order to estimate C, it becomes a matter of estimating the parameter vector.

Moreover, if we know the margins F and G of H, then the sample

$$(U,V)_i = (F(X_i), G(Y_i)), \quad i \in \{1, \cdots, n\} \tag{2.13}$$

would be observable and independently and identically distributed. By Lemma 3, it would be a sample from C. Then we could turn to techniques such as maximum likelihood estimation. However, we do not know the margins of H and so the margins are nuisance parameters that we have to estimate so that we can estimate the parameter vector.

We can estimate the margins either parametrically or nonparametrically. If we assume our margins belong to continuous parametric families of univariate distribution functions, we can use maximum likelihood techniques to estimate both the marginals and the copula. However, if the margins are misspecified, the estimation of our parameter vector will be biased (Hofert et al. 2018). There can also be a computational burden if we want high-dimensional optimization. Instead, there is another approach called the inference functions for margins estimator that Hofert et al. (2018) and the references therein go more in depth on. On the other hand, we can avoid issues of model misspecification for the marginals with nonparametric estimates by finding the empirical distribution functions, F_n and G_n of the component samples of $(X_1, Y_1), \cdots, (X_n, Y_n)$. We estimate F_n and G_n nonparametrically by

$$F_n(x) = \frac{1}{n+1} \sum_{i=1}^{n} 1(X_i \leq x), \quad x \in \mathbb{R}; \tag{2.14}$$

$$G_n(x) = \frac{1}{n+1} \sum_{i=1}^{n} 1(Y_i \leq y), \quad y \in \mathbb{R}.^3 \tag{2.15}$$

[3] We divide by $n+1$ instead of n to ensure that the sample lies in the interior of the unit square.

We then use the estimated margins to form the sample

$$(U_n, V_n)_i = (F_n(X_i), G_n(Y_i)), \quad i \in \{1, \cdots, n\} \tag{2.16}$$

This sample is regarded as a consistently estimated version of the unobservable independent and identically distributed sample, $(U, V)_i$ from Equation (2.13) and is often called the "pseudo-observations" from C. Note that the U_ns and V_ns are not true observations and they are also not independent since F_n and G_n depend on the Xs.

Furthermore, these estimators for the margins F and G are actually also functions of the ranks of the observations. Let R_{X_i} be the rank of X_i where $i = 1, \cdots, n$ and R_{Y_i} be the rank of Y_i where $i = 1, \cdots, n$. It is easy to verify that $F_n(X_i) = R_{X_i}/(n+1)$ and $G_n(Y_i) = R_{Y_i}/(n+1)$; that is, the sample is the sample of multivariate scaled ranks:

$$(U_n, V_n)_i = \frac{1}{n+1}(R_{X_i}, R_{Y_i}), \quad i \in \{1, \cdots, n\}. \tag{2.17}$$

With these nonparametrically estimated margins, one can then estimate the parameter vector of interest. Two of the most popular estimation methods in this case are the method of moments approaches based on Kendall's tau and Spearman's rho and the Maximum Pseudo-likelihood estimator. There are plenty of other semiparametric approaches and we refer the reader to Chapter 2.6 and 4 of Hofert et al. (2018) for these methods and more.

2.9.2 Nonparametric Estimation of the Copula

We can also make no parametric assumptions about the marginals or the copula C and instead rely on nonparametric estimates. To this extent, we consider the empirical copula which is a consistent estimator of C defined by:

$$C_n(u,v) = \frac{1}{n}\sum_{i=1}^{n} 1(U_{n,i} \leq u, V_{n,i} \leq v), \quad (u,v) \in [0,1]^2 \tag{2.18}$$

where $(U_n, V_n)_i, i = 1, 2, \cdots, n$ are the pseudo-observations from Equation (2.16) (recall this sample uses the empirical distributions of F_n and G_n).

2.10 Big idea

There was a lot of information presented in this chapter and various ideas were included in this chapter for the sake of a somewhat complete yet brief literature review of copulas in the continuous case. The big idea the reader should keep in the back of their mind is that in the continuous case, **things work out nicely**. There exists a unique copula which models the dependence structure separate from the marginal distribution functions. Because the copula is synonymous with the dependence structure, the problem of examining the dependence between random variables boils down to estimating the unique copula that joins them together. However, there is no unique copula when we travel into the discrete world, what we have now are “possible copulas” and this can cause all sorts of issues as we will see in the next chapter.

Chapter 3 Challenges with Dependence Measures for Discrete Variables

> If all the statisticians in the world were laid head to toe, they wouldn't be able to reach a conclusion.
>
> Anon

Recall the definitions of subcopula and copula in Chapter 2. Copulas are subcopulas whose entire domain is the unit square so the two would be the same when the domain of the subcopula is the unit square. However, if this is not the case, then the subcopula is only uniquely defined on the domain $D_1 \times D_2$ where we defined D_1, D_2 to be the respective ranges of the margins F and G. In this scenario, any copula would work here as long as it agrees on the domain, leading to an issue of identifiability. In Chapter 4, we discuss one such copula but for now, we dedicate this chapter to showing the issues that arise when at least one variable is discrete.

3.1 Issue of Identifiability

Recall Theorem 2 which states we can write the joint distribution function H in terms of some copula C and its marginals F and G.

Consider Example 2 which shows that the only values of the copula C that have any effect on H are those that agree on the domain.

Example 2. Let (X, Y) be a bivariate random vector from a bivariate Bernoulli distribution such that: $P(X = 0, Y = 0) = 1/8,\ \ P(X = 1, Y = 1) = 3/8,\ \ P(X = 0, Y = 1) = 2/8,\ \ P(X = 1, Y = 0) = 2/8$. From Theorem 2, we know that $P(X \leq x, Y \leq y) = C(P(X \leq x), P(Y \leq y))$ for all x, y and some copula C. Since $\text{Ran}(F) = \text{Ran}(G) = \{0, 3/8, 1\}$, the only constraint on C is that $C(3/8, 3/8) = 1/8$. Any copula fulfilling this constraint is a copula of (X, Y) and there are infinitely many such copulas.

Another way we can think about this is that, in the discrete case, the domain of the copula C is a proper subset of $\mathbb{I}^2$ (including 0 and 1), so there are "gaps" in $\mathbb{I}^2 \backslash \text{Ran}(F) \times \text{Ran}(G)$ that need to be filled in (Geenens, 2020). And we can fill in these gaps in all sorts of ways leading to an identifiability issue. One method described by Rüschendorf (2009) and Faugeras (2017) is the Distributional Transform in which we essentially jitter the points. More precisely, consider the random variable X with some distribution function F and let $V \sim U(0, 1)$, independent of X. Rüschendorf (2009) defines the distributional transform of X by:

$$U := F(X, V) = P(X < x) + VP(X = x). \tag{3.1}$$

The big idea is that at any jump point of the distribution function, F, one uses V to randomize the jump height. An equivalent representation is:

$$U = F(X-) + V(F(X) - F(X-)) \tag{3.2}$$

where $F(X-)$ denotes the left limit of X.

We demonstrate this transform in Example 3 with an empirical simulation in the univariate case.

Example 3. Suppose $X \sim F$ where F is the Poisson distribution with $\lambda = 1$. We simulate 1000 independent copies of X, compute U using Equation (3.1) and plot the empirical distribution of $F(X)$ on the left and the empirical distribution of U on the right. In Figure 3.1 below, one can clearly see that the distribution of U, unlike that of X, is continuous.

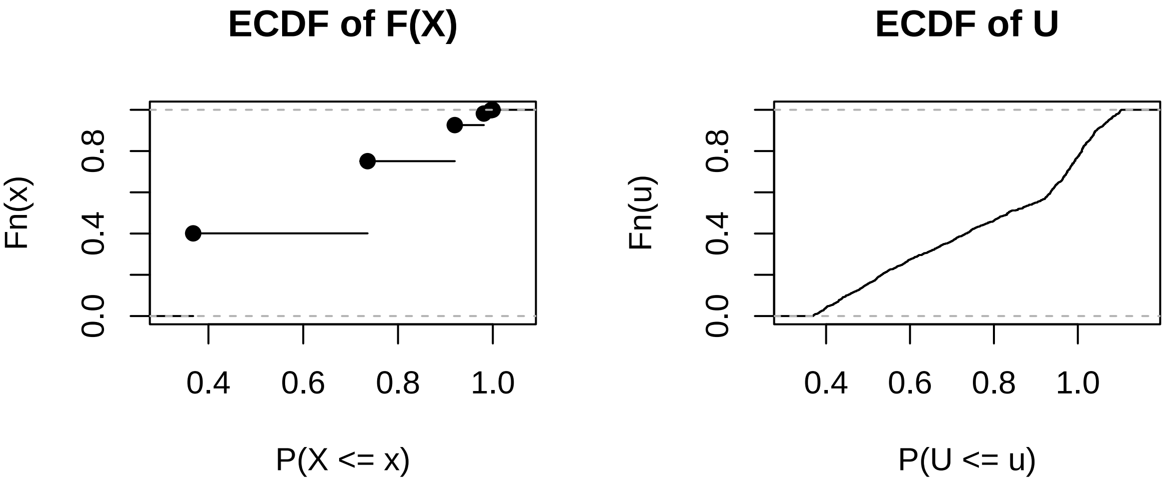

Figure 3.1: Empirical distribution functions of 1000 copies of X simulated from a Poisson distribution with $\lambda = 1$ (left) and the Distributional Transformed Sample (right).

Notice in Figure 3.1 the distributional transform, roughly speaking, smooths out the cumulative distribution function. However, the fact that there are an infinite amount of ways to go about this makes the solution non-unique (just think of all the different values in [0,1] that you could "jitter" by).[1]

After understanding that copulas in the discrete case are not unique, one might ask what is the huge issue about unidentifiability? We discuss some of the key issues that arise when unidentifiability occurs next.

[1] It can be shown the set of copula functions $C \in C_H$ compatible with the joint distribution function H can be quite large and lies between bounds referred to as Carley's bounds C_H^+, C_H^-.

3.2 The Copula Alone Does Not Model the Dependence Between X and Y

In the continuous case, X and Y have a unique underlying copula. Furthermore, X and Y are independent if and only if the copula $C = \Pi$. Recall that Π, the independence copula, is defined as:

$$\Pi(u, v) = uv, \; u, v \in (0, 1).$$

While $C = \Pi$ still implies the independence between X and Y, the other way is no longer true. We use an example from Genest & Nešlehová (2007) to demonstrate this occurrence. We note that, unless stated otherwise, the examples later on are also from Genest & Nešlehová (2007).

Example 4. Let X and Y be independent Bernoulli random variables with $P(X = 0) = p$ and $P(Y = 0) = q$. Then C is a copula model for (X,Y) if and only if $C(p, q) = P(X = 0, y = 0) = pq$. However, suppose that $p = q = 1/2$ and consider $C = (W + M)/2$ where W and M are the lower and upper Fréchet-Hoeffding Bounds. Then $C(1/2, 1/2) = 1/4 = pq$ but $C \neq \Pi$. This shows discrete copulas do not alone characterize the dependence between X and Y. Thus, it is an issue because we can model (in)dependence with different copulas, yet be compatible with $C(p, q) = pq$.

3.3 Concordance Measures Are Margin-Dependent

In the continuous case, measures like Kendall's tau and Spearman's rho provide margin-free measures of dependence and thus they can be used to construct estimation methods for the copula under specific parametric assumptions. But when the random variables are discrete, copula-based measures of dependence such as Kendall's tau or Spearman's rho are margin-dependent. If the dependence cannot be modeled by the copula alone and depends on the marginals, copulas become less useful in characterizing the dependence structure.

Example 5. Let X and Y be Bernoulli RVs with $P(X = 0) = p$ and $P(Y = 0) = q$. Let $r = P(X = 0, Y = 0) \in [\max(0, p + q - 1), \min(p, q)]$. Then

$$\tau(X, Y) = \rho(X, Y) = r - pq.$$

In the previous example, Kendall's tau and Spearman's rho now both rely on p and q which are the marginal probabilities of $P(X = 0)$ and $P(Y = 0)$ respectively.

3.4 The Stochastic and Analytical Definitions of τ and ρ Do Not Match

Recall that, when X and Y are continuous random variables with a unique underlying copula, then $\tau(X, Y) = \tau(C)$ and $\rho(X, Y) = \rho(C)$. But when we have discrete random variables, depending on our choice of copula, it may lead to different values for $\tau(C)$ and $\rho(C)$. Fortunately, there does exist a copula, namely the checkerboard copula, such that the stochastic and analytical definitions of τ and ρ do coincide.

3.5 Perfect monotone dependence does not imply that $|\tau|$ and $|\rho| = 1$

Let's consider the following example that illustrates this issue.

Example 6. Suppose X and Y are Bernoulli random variables where $P(X = 0) =$

$P(Y = 0) = P(X = 0, Y = 0) = p$ where $p \in (0, 1)$. Then $Y = X$ almost surely and $\tau(X, Y) = \rho(X, Y) = p(1 - p) < 1$.

3.6 Copula Model Chosen Might Be Valid for Some Restricted Discrete Distributions

Example 7. Let X and Y be independent Bernoulli random variables with $P(X = 0) = p$ and $P(Y = 0) = q$ and $P(X = 0, Y = 0) = r$. For $i, j = 1, 2$, let n_{ij} represent the number of times that $X = i, Y = j$ in a random sample of size n. The maximum likelihood estimates of the three parameters are:

$$\hat{p}_n = \frac{n_{00} + n_{01}}{n}, \ \ \hat{q}_n = \frac{n_{00} + n_{10}}{n}, \ \ \hat{r}_n = \frac{n_{00}}{n}. \tag{3.3}$$

Moreover, when we can write the joint distribution H of (X, Y) as $C(p, q)$ for some copula $C \in C_\theta$ (family of copulas parametrized by θ), one has $C_\theta(p, q) = r$ so that the maximum likelihood estimate, $\hat{\theta}_n$ is the unique value of θ such that $C_{\hat{\theta}_n}(\hat{p}_n, \hat{q}_n) = \hat{r}_n$. Genest & Nešlehová (2007) assumes that this bivariate Bernoulli distribution stems from a combination of univariate Bernoulli marginal distributional functions with a Farlie-Gumbel-Morgenstern copula which is defined analytically for $(u, v) \in [0, 1]^2$:

$$C_\theta(u, v) = uv + \theta uv(1 - u)(1 - v), \ \theta \in [-1, 1]. \tag{3.4}$$

Then to find θ, we switch some variables around to get:

$$\theta = \frac{r - pq}{pq(1 - p)(1 - q)}. \tag{3.5}$$

Next, to find the maximum likelihood estimator for θ, replace p, q, r with their maximum likelihood estimators defined in Equation (3.3) and get:

$$\hat{\theta}_n = \frac{\hat{r}_n - \hat{p}_n\hat{q}_n}{\hat{p}_n\hat{q}_n(1-\hat{p}_n)(1-\hat{q}_n)}. \tag{3.6}$$

Now suppose that the true values for this model were $p = 0.3, q = 0.4$ and $r = 0.1452$, then we would get $\theta = 0.5$. Everything works out fine and dandy here since we have a plausible θ that is within bounds. However, Faugeras (2017) points that if we assume the true values were $p = 0.3, q = 0.4$ and $r = 0.175$, then $\theta = 1.09127 > 1$ which violates the bounds we placed on θ, meaning it's impossible given the Farlie-Gumbel-Morgenstern model! Now suppose you are estimating the previous p, q, r from some sample and get $\hat{p} \approx 0.31, \hat{q} \approx 0.41, \hat{r} \approx 0.174$ which would give us $\hat{\theta} \approx 0.906$ which is valid but given the true parameter values, we know that it is not valid! The other way can very well happen too where you get estimates that indicate it is not valid when it is in fact valid.

In this situation, the researcher can make both mistakes of inferring a seemingly correct copula parameter value in an impossible model, or rejecting the correctly specified model from an apparently incorrect copula parameter value.

3.7 The Copula Model is Unidentifiable

Consider Example 7 mentioned above. Because the Fairlie-Gumbel-Morgenstern family of copulas model relatively weak dependence, we can instead choose another model that we think might better represent the dependence structure. A few other choices that one might pick are Plackett's copula, or Ali-Mikhail-Haq (AMH) copula, or Gaussian copulas. If we assume the true values are $p = 0.3, q = 0.4, r = 0.1452$, then for each copula, it can be shown that we get the following θ values:

1. Farlie-Gumbel-Morgenstern copula where $\theta \in [-1, 1]$: $\theta = 0.5$

2. Plackett copula where $\theta > 0$, then $\theta = 1.6389$

3. AMH where $\theta \in [-1, 1]$, then $\theta = 0.413223$

This issue demonstrates one of the major problems that may arise when the copula model is unidentifiable. Although it is possible to estimate the copula parameter once we have assumed a certain copula model, there is no guarantee that the real data generating process stemmed from our chosen copula model. Choosing the model is now simply a matter of preference. In other words, for each model, we have inferred some value for the dependence parameter, but how do we reconcile these different values? After all, the value of the copula parameter only has meaning within the arbitrarily chosen copula family. This makes it difficult, if not impossible, to make any statement about the dependence structure, because each value of the copula parameter only makes sense in the context of that particular copula family which is arbitrarily chosen.

3.8 Are Copula Models for Discrete Data Interesting at All?

What should we do if we want to apply theoretically-sound inference methods to measure dependence between discrete variables? Can we still use methods mentioned in Chapter 2 regarding estimation and dependence measures? Some argue in favor of using copulas. Regarding the lack of a unique copula, Trivedi & Zimmer (2017) write, "this usually does not pose a problem in applied settings, as researchers use copulas because the joint distribution $F(y_1, y_2)$ is either not known or is difficult to work with." Moreover, statisticians like Genest & Nešlehová (2007), argue that $H(x, y) = C(F(x), G(y))$ is a "bona fide" bivariate distribution. They also suggest that H inherits most (but not all) of the dependence properties of the copula from which it came from and that θ can still be interpreted as a dependence parameter. In addition, they claim "the fact that there exist (infinitely many) copulas for the same discrete joint distribution does not invalidate models of this sort." The mathematical construction of discrete copula models is still valid and under certain constraints, can prove to be viable and useful in simulations and robustness studies. However, when dealing with count data, modeling and interpreting dependence through copulas is subject to caution and inference for copula parameters from discrete data is difficult as we have seen in this chapter.

The reader might also be curious if we even need to use copulas for discrete variables. Can we use inference methods and construct measures based on the unique subcopula instead? The answer to that question is yes but they have their own array of problems as well. Tasena (2021) proposes an estimator of the subcopula and Erdely (2016) proposes a dependence measure based on the subcopula. However, further discussion on this matter strays from the focus of the thesis.

In conclusion, a lot of issues come up when the copula is unidentifiable. When we can use more than one copula to characterize the dependence structure between random variables, which one should we use? It turns out that there is actually one copula that best represents the dependence structure: the checkerboard copula. Genest, Nešlehová, & Rémillard (2017) writes

> The weak convergence of the empirical checkerboard copula process is shown to be sufficiently strong to derive the asymptotic behavior of a broad class of functionals that are directly relevant for the development of rigorous statistical methodology for copula models with arbitrary margins.

With this in mind, we move on to discussing the checkerboard copula.

Chapter 4 Checkerboard Copulas and Checkerboard Copula Based Dependence Measure

> Correlation doesn't imply causation, but it does waggle its eyebrows suggestively and gesture furtively while mouthing 'look over there'.
>
> Randall Munroe, - xkcd

Now that we have a solid foundation on copulas and also know the reasons why copulas sort of sputter out when dealing with discrete variables, we now turn our attention to the paper by Wei and Kim (2021) that motivated this thesis. In their paper, they used a copula called the checkerboard copula to identify and measure the regression dependence between an ordinal response variable and categorical explanatory variables in contingency tables.

Recall the motivation in Chapter 1 behind the need for such methods. Categorical data analysis with ordinal responses is important in several fields and taking into consideration the intrinsic ordering of ordinal variables can give more powerful inferences. One step in categorical analysis is exploring the various dependence structures among the variables for exploratory modeling. A dependence structure of particular interest is that of the regression dependence which many model-based approaches have been constructed. Comparatively, there are fewer model-free approaches to examining dependence structures in categorical data, and most of these approaches do not distinguish between explanatory variable(s) and response variable, i.e., they do not focus on regression dependence. To address this, Wei & Kim (2021) propose a new model-free measure, based on the checkerboard copula, to identify and quantify the regression dependence in multivariate categorical data with an ordinal response variable and categorical (nominal or ordinal) explanatory variables . It can be shown that the checkerboard copula constructed through bilinear interpolation of the bivariate distribution function, uniquely links the marginal distributions of discrete random variables to their joint distribution function. Research has demonstrated that the checkerboard copula has many good properties and can best represent the dependence structure among discrete variables (Genest & Nešlehová, 2007; Genest, Nešlehová, & Rémillard, 2014, 2017; Nešlehová, 2007).

In the upcoming sections, we will first introduce and define the checkerboard copula in Section 4.1 before going into detail regarding their proposed methodology, which is comprised of three parts:

- The checkerboard copula score [Section 4.2]: This is newly proposed score for ordinal variables that takes into account their intrinsic ordering.
- The checkerboard copula regression [Section 4.3]: This is a model-free approach towards identifying the regression dependence in categorical data. Note then we can use it to predict the category of the ordinal response variable from the combination of categories of explanatory variables.
- The checkerboard copula association measure (CCRAM) [Section 4.4]: This is an index to quantify the strength of association identified by the checkerboard

copula regression. It is essentially the average proportion of variance in the ordinal response variable (with respect to its checkerboard copula score and its marginal distribution) attributable to the checkerboard copula regression.

One might think of the CCRAM as an R^2-like measure but for the checkerboard copula regression. We will then explain in Section 4.5 how Wei & Kim (2021) estimated the CCRAM in a model-free way while proposing a new idea to estimate the CCRAM based on a parametric model. For the sake of simplicity and clarity, we will explain and demonstrate this methodology in 2 dimensions in this chapter, but it should be noted that all apply to higher dimensions too.

4.1 Checkerboard Copula

Suppose X, Y are ordinal variables with I and J ordered categories: $\{x_1 < \ldots < x_I\}$ and $\{y_1 < \ldots < y_J\}$ respectively. We can form a 2-way contingency table with joint probability mass function of X and Y denoted as $P = \{p_{ij}\}$, where $i = 1, \ldots I$; $j = 1, \ldots, J$; and $\sum_{i=1}^{I} \sum_{j=1}^{J} p_{ij} = 1$. The i^{th} row and j^{th} column marginal probability mass functions are denoted by $p_{i\bullet} = \sum_{j=1}^{J} p_{ij}$ and $p_{\bullet j} = \sum_{i=1}^{I} p_{ij}$. The conditional probability mass functions of Y given X and vice versa are denoted by $p_{j|i} = \frac{p_{ij}}{p_{i\bullet}}$ and $p_{i|j} = \frac{p_{ij}}{p_{\bullet j}}$, if $p_{\bullet j} \neq 0$ and $p_{i\bullet} \neq 0$, respectively, and zero otherwise.

Furthermore, we denote the range of the marginal distributions of X and Y to be $D_1 = \{u_0, \ldots, u_i, \ldots, u_I\}$ where $u_0 = 0, u_I = 1$ and $u_i = \sum_{i=1}^{I} p_{i\bullet}$ and $D_2 = \{v_0, \ldots, v_j, \ldots, v_J\}$ where $v_0 = 0, v_J = 1$, and $v_j = \sum_{j=1}^{J} p_{j\bullet}$. Then, from Theorem 2, there exists a unique subcopula C^S on $D_1 \times D_2$ such that:

$$H(x, y) = C^S(F(x_i), G(y_j)) = C^S(u_i, v_j) = \sum_{s \leq i} \sum_{t \leq j} c^S(u_s, v_t) \tag{4.1}$$

where $H(x, y)$ is the joint distribution function for X and Y; $F(x), G(y)$ are its marginal distributions and $c^S(u_i, v_j) = p_{ij}$ is the probability mass function of $C^S(u_i, v_j)$.

One can then extend the subcopula C^S on $D_1 \times D_2$ to a copula, say C, on $\mathbb{I}^2 = [0,1]^2$ via the so-called bilinear extension which we explain below. This is called the bilinear extension copula, also known as the checkerboard copula. We thus define the checkerboard copula and its density function as such:

Definition 7. Let C^S be the subcopula on $D_1 \times D_2$ satisfying (4.1). For any $(u, v) \in [0,1]^2$, let u_1, u_2 be the least and greatest elements of $\overline{D}_1$, the closure of set D_1 such that $u_1 \leq u \leq u_2$. And let v_1, v_2 be the least and greatest elements of $\overline{D}_2$, the closure of set D_2 such that $v_1 \leq v \leq v_2$. Then the bilinear extension copula (checkerboard copula), $C^{\maltese}(u, v)$, is defined by:

$$\begin{aligned} C^{\maltese}(u,v) = (1-\lambda)(1-\mu)C^S(u_1, v_1) + (1-\lambda)\mu C^S(u_1, v_2) \\ +\lambda(1-\mu)C^S(u_2, v_1) + \lambda\mu C^S(u_2, v_2) \end{aligned} \tag{4.2}$$

where $\lambda = \frac{u-u_1}{u_2-u_1}, \mu = \frac{v-v_1}{v_2-v_1}$. Note that if $u \in \overline{D}_1$, then $u_1 = u = u_2$; if $v \in \overline{D}_2$, then $v_1 = v = v_2$. Also, if $u_1 = u_2$, then $\lambda = 1$, similarly, if $v_1 = v_2$, then $\mu = 1$.

Figure 4.1 demonstrates the idea behind the checkerboard copula in which we bilinearly interpolate C^S on the unit square. The horizontal axis is the marginal distribution function of X, i.e., $F(x) = u$ and the vertical axis is the marginal distribution function of Y, i.e., $G(y) = v$. The points labeled $(u_1, v_1), (u_1, v_2), (u_2, v_1), (u_2, v_2)$ are as defined in Definition 7 and $(u, v) \in [0,1]^2$ is the point we want to interpolate at. Applying the bilinear extension, i.e., Equation (4.2), to all points on the unit square would smooth out the subcopula, so that the resulting checkerboard copula would be uniquely determined and have continuous uniform marginals. See Figure 4.2 for a 3D visualization of this idea.

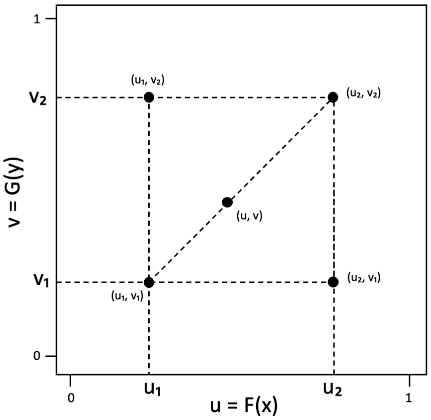

Figure 4.1: Plot of the Checkerboard Copula

By taking the derivatives of $C^{\maltese}$ with respect to u, v, the checkerboard copula density function $c^{\maltese}(u, v)$ is defined:

$$c^{\maltese}(u,v) = \frac{p_{ij}}{p_{i\bullet}p_{\bullet j}}, \text{ if } u_{i-1} < u \leq u_i, v_{j-1} < v \leq v_j. \tag{4.3}$$

The conditional density of V given U is defined as:

$$c^{\maltese}(u,v) = \frac{c^{\maltese}(v|u)}{c^{\maltese}(u)} = \frac{p_{j|i}}{p_{\bullet j}}. \tag{4.4}$$

Observe that from Definition 7, it follows that $C^{\maltese}$ coincides with the unique subcopula C^S of Equation (4.1). Moreover, Nešlehová (2007), Genest, Nešlehová, & Rémillard (2017), Rüschendorf (2009) obtained a stochastic representation of $C^{\maltese}(u, v)$ where $C^{\maltese}(u, v)$ is the joint distribution of two standard uniform variables U and V, the distributional transform of the ordinal variables X and Y:

$$U = F(X-) + [F(X) - F(X-)]W_1 \text{ and } V = G(Y-) + [G(Y) - G(Y-)]W_2. \tag{4.5}$$

Here, $F(X-)$ refers to the left limit of F, while W_1 and W_2 denote independent uniform random variables on $[0, 1]$, independent of X and Y. Roughly speaking, the stochastic representation in Equation (4.5) allows us to "jitter" the discontinuous function using W_1, W_2. Overall, the idea behind the checkerboard copula is that it is a smooth version of the subcopula in that it spreads the mass uniformly over the hyper rectangle.

Throughout the next several sections in this chapter, we will use a toy example provided by Wei & Kim (2021) to help the readers understand this methodology.

Example 8. Suppose X and Y represent the dose of a treatment drug for acute migraines and the severity of migraine pain after treatment respectively with $I = 5$ and $J = 3$ ordered categories. $(x_1, x_2, x_3, x_4, x_5) =$ (very low, low, medium, high, very high) and $(y_1, y_2, y_3) =$ (mild, moderate, severe).

$X \backslash Y$	y_1	y_2	y_3
x_1	0	0	2/8
x_2	0	1/8	0
x_3	2/8	0	0
x_4	0	1/8	0
x_5	0	0	2/8

Table 4.1: Joint pmf of X and Y, $P = \{p_{ij}\}$

Table 4.1 displays the joint probability mass function of X and Y. Here, we note that Y has a quadratic relationship with X since the level of Y decreases and then increases as the level of X increases. Note that Y is a function of X but not vice versa. For a given category of X, there is only one category of Y with joint probability that is non-zero. The marginal probability mass functions of X and Y are $p_{i\bullet} \in \{2/8, 1/8, 2/8, 1/8, 2/8\}$ and $p_{\bullet j} \in \{2/8, 2/8, 4/8\}$. The ranges of the marginal cdfs of X and Y are $D_1 = \{u_0, u_1, u_2, u_3, u_4\} = \{0, 2/8, 3/8, 5/8, 6/8, 1\}$ and

$D_2 = \{u_0, u_1, u_2\} = \{0, 2/8, 4/8, 1\}$, respectively. We illustrate the subcopula C^S and its density, as well as the checkerboard copula $C^{\maltese}$ and its density in Figure 4.2.

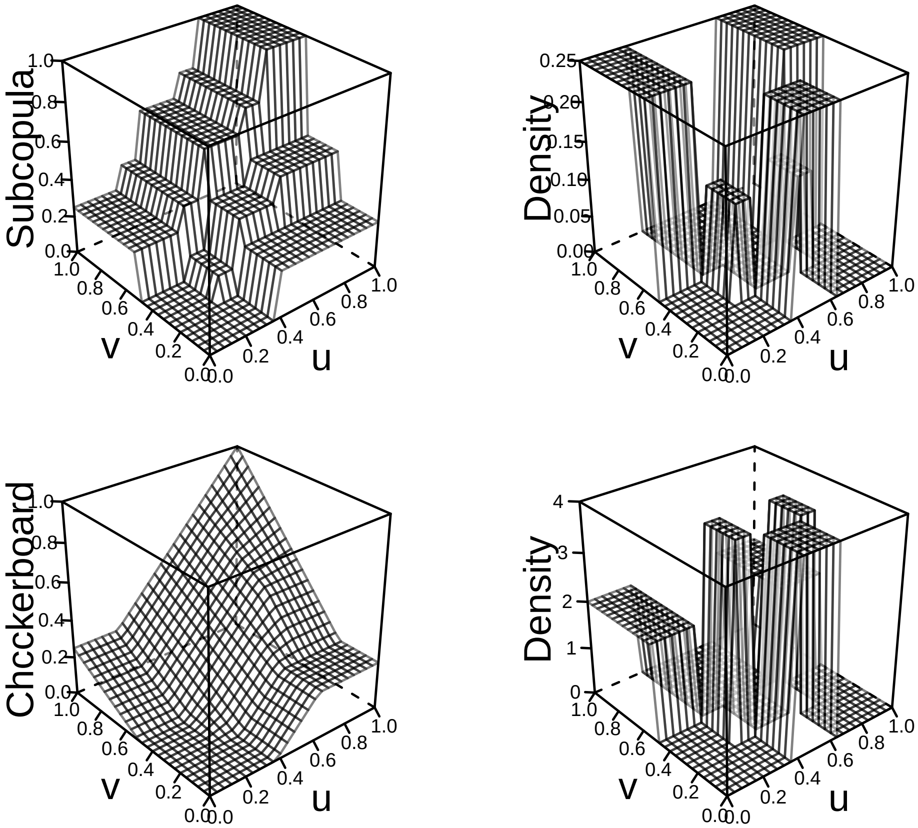

Figure 4.2: Surface plots of the subcopula (top left) and its density (top right) and surface plots of the checkerboard copula (bottom left) and its density (bottom right).

Figure 4.2 shows the subcopula (top left) and its density (top right) alongside the checkerboard copula (bottom left) and its density (bottom right). Notice how the checkerboard copula is a “smoothed” version of the subcopula. In addition, we include Figure 4.3 from Wei & Kim (2021) which is the projection of the bottom right 3D plot in Figure 4.2 to a 2D plot. Here, intensity of color represents the corresponding density which shows that the checkerboard copula density inherits the dependence between X and Y. In their example, they use U_1 and U_2 which in our notation is $U_1 = U$ and $U_2 = V$.

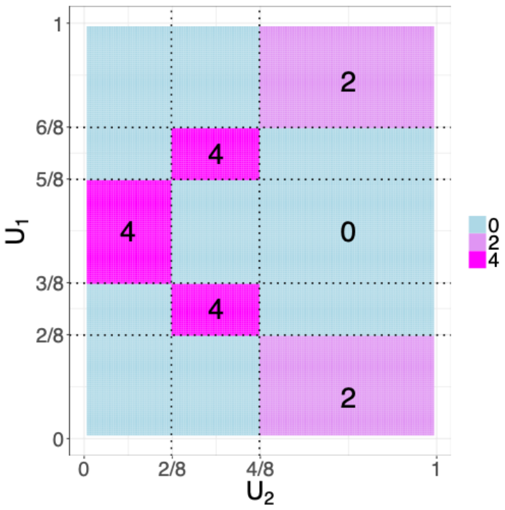

Figure 4.3: Checkerboard Copula Density (Wei & Kim, 2021)

4.2 Checkerboard Copula Score

Wei & Kim (2021) proposed a new type of score for ordinal variables obtained from the checkerboard copula in order to take advantage of the intrinsic ordering. As defined in Definition 7 and shown in Figure 4.3, the checkerboard copula $C^{\maltese}$ is a smooth version of the subcopula associated with the ordinal random vector (X, Y) in that it spreads the mass uniformly over each rectangle $[u_{i-1}, u_i] \times [v_{j-1}, v_j]$ where u_i, v_j are elements of the ranges of the marginal cdfs of X and Y. Furthermore, $C^{\maltese}$ is the joint distribution function of (U, V) in Equation (4.5) which is the distributional transform of X, Y. Motivated by these properties, they define a new random variable S_j where $j \in \{1, 2\}$ to be a transformation of X and Y via U and V: $S_1 = E[U \mid X]$ and $S_2 = E[V \mid Y]$ respectively. Wei & Kim (2021) proved that S_1 and S_2 are ordinal random variables with numerical support values $\{s_1^1, \cdots, s_i^1, \cdots, s_I^1\}$ and $\{s_1^2, \cdots, s_j^2, \cdots, s_J^2\}$, respectively, where $s_i^1 = \frac{(u_{i-1}+u_i)}{2}$ and $s_j^2 = \frac{(v_{j-1}+v_j)}{2}$ and that S_1 and S_2 have the same probability mass functions as X and Y respectively. They then proposed the support values of S_1 and S_2 as a new type of score for X and Y, which they called checkerboard copula scores.

Definition 8. The checkerboard copula scores of ordinal variables X and Y are:

$$\{s_1^1, \dots, s_I^1\}, \quad s_i^1 = \frac{(u_{i-1} + u_i)}{2} \tag{4.6}$$

$$\{s_1^2, \dots, s_J^2\}, \quad s_j^2 = \frac{(v_{j-1} + v_j)}{2} \tag{4.7}$$

for $i \in \{1, \cdots, I\}, j \in \{1, \cdots, J\}$ and u_i, v_j are given previously.

We can think of the checkerboard copula scores as the set of the average of the marginal cumulative distributions evaluated at every two consecutive categories of X and Y respectively. For those who have dealt with ordinal categorical data analysis before, this might look very familiar as it goes by another term, ridits (Bross, 1958).

Here are some properties of the checkerboard copula scores; for proofs, see the Appendix of Wei & Kim (2021):

- The scores have the same ordering as the categories of X and Y: $0 < s_1^1 < \cdots < s_i^1 < \cdots < s_I^1 < 1$ and $0 < s_1^2 < \cdots < s_j^2 < \cdots < s_J^2 < 1$.
- The conditional expectation of the stochastic representation in Equation (4.5) given $X = x_i$ with respect to U and is equal to the i^{th} checkerboard score for X: $E(U|X = x_i) = s_i^1$. Similarly, the conditional expectation of the stochastic representation in Equation (4.5) given $Y = y_j$ with respect to V and is equal to the j^{th} checkerboard score for Y: $E(V|Y = y_j) = s_j^2$.
- Mean and variance of S_1 (S_2) are $\mu_{S_1} = 0.5$ ($\mu_{S_2} = 0.5$) and $\sigma_{S_1}^2 = \frac{1}{4}\sum_{i=1}^{I} u_{i-1}u_i p_{i\bullet}$ ($\sigma_{S_2}^2 = \frac{1}{4}\sum_{j=1}^{J} v_{j-1}v_j p_{\bullet j}$). The maximum variance is attained when $p_{i\bullet} = \frac{1}{I}$ ($p_{\bullet j} = \frac{1}{J}$).

Example 9 (Example 8 Continued)**.** The checkerboard copula scores for X and Y

are (2/16, 5/16, 8/16, 11/16, 14/16) and (2/16, 6/16, 12/16) respectively. The means and variances of $S_1 = E[U \mid X]$ and $S_2 = E[V \mid Y]$ are (0.5, 81/1024) for S_1 and (0.5, 9/128) for S_2.

4.3 Checkerboard Copula Regression

Next, Wei & Kim (2021) proposed the checkerboard copula regression as follows:

Definition 9. Let U and V from Equation (4.5) be standard uniform variables associated with the checkerboard copula $C^{\maltese}(u, v)$ from Equation (4.2). The checkerboard copula regression function of V on U is defined as follows for $u_{i-1} < u < u_i$,

$$r_{V|U}(u) \equiv E_{c^{\maltese}}(V|U = u) = \int_0^1 vc^{\maltese}(v|u)dv = \sum_{j=1}^{J} p_{j|i}s_j^2, \tag{4.8}$$

where $c^{\maltese}(v|u)$ is the conditional density function of V given U. This function can be viewed as the mean checkerboard copula score of Y with respect to the conditional distribution at the i^{th} category of X.

Example 10 (Example 8 continued)**.** Observe from Table 4.2 and Figure 4.4 that the regression of V on U captures the quadratic dependence because it reflects the changes in D_2 associated with Y according to the changes in the D_1 associated with X and it is only equal to one of the checkerboard scores of Y for each interval in U.

Table 4.2: Checkerboard copula regression of V on U.

u	$r_{V\|U}(u)$
$[0, 2/8]$	12/16
$(2/8, 3/8]$	6/16
$(3/8, 5/8]$	2/16
$(5/8, 6/8]$	6/16
$(6/8, 1]$	12/16

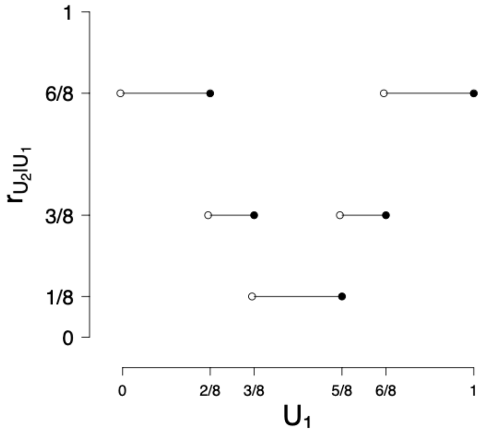

Figure 4.4: Regression of V on U (Here $U_1 = U$ and $U_2 = V$) (Wei & Kim, 2021).

Note that this regression function can also be used in predicting the category of the ordinal response variable given a category of the explanatory variable. Suppose Y is the response variable and X is the explanatory variable. For a given category of X, we can find the corresponding $u^* \in D_1$ and obtain the predicted value of the checkerboard copula regression, $v^* = r_{V|U}(u^*)$. From the range D_2 of the marginal distribution Y, we get j^* and v_{j^*} such that $v_{j^*-1} < v^* \leq v_{j^*}$ and obtain the predicted category of the response variable Y, y_{j^*}. Wei & Kim (2021) show that the prediction of the response variable is invariant under permutation of categories of explanatory variables, implying it can also be used for nominal explanatory variables.

Example 11 (Example 8 continued)**.** Using the checkerboard copula regression, we obtain predictions of Y for each category of X. Say we are given $X = 2$, then the corresponding $u_2^* \in D_1 = \{0, 2/8, 3/8, 5/8, 6/8, 1\}$ is 3/8 and then $r_{V|U}(3/8) = 3/8$ and because $0.25 \leq 3/8 \leq 0.50$, the predicted category of Y given $X = 2$, is the 2nd category of Y. We can apply the same prediction method and predict Y for all levels of X (category of X: predicted Y) = (1:3), (2:2), (3:1), (4:2), (5:3). Note that these predictions also reflect the quadratic relationship.

4.4 Checkerboard Copula Regression Association Measure (CCRAM)

The checkerboard copula regression association measure is then built upon the checkerboard copula regression.

Definition 10. For the ordinal contingency table of X and Y in a $I \times J$ table, the checkerboard copula regression association measure of Y on X is:

$$\rho^2_{(X\to Y)} \equiv \frac{Var(r_{V|U}(U))}{Var(V)} = \frac{E[(r_{V|U}(U) - 1/2)^2]}{1/12} = 12\sum_{i=1}^{I}\left(\sum_{j=1}^{J} p_{j|i} s_j^2 - 1/2\right)^2 p_{i\bullet}. \tag{4.9}$$

Here are some properties of the checkerboard copula regression association measure:

1. $0 \leq \rho^2_{(X\to Y)} \leq 12\sigma^2_{S_2} < 1$ where $\sigma^2_{S_2}$ is the variance of S_2. Recall that $S_2 = E(V|Y)$.

2. If X and Y are independent, then $\rho^2_{(X\to Y)} = 0$.

3. If $\rho^2_{(X\to Y)} = 0$ then $r_{V|U}(U) = E(V) = 1/2$ and $\text{cor}(U, V) = 0$.

4. $\rho^2_{(X\to Y)} = 12\sigma^2_{S_2}$ if and only if $Y = g(X)$ almost surely for some measurable function g.

5. $\rho^2_{(X\to Y)} < \frac{Var(r_{V|U}(U))}{\sigma^2_{S_2}}$.

6. $\rho^2_{(X\to Y)}$ is invariant over permutation on the categories of X.

7. $\rho^2_{(X\to Y)}$ is invariant over permutation on the categories of Y only when Y is binary.

Let's add some comments regarding the above properties:

1. From properties 1 - 4, we can see that the proposed measure can identify linear/nonlinear relationships between X and Y.

2. From property 1, the measure ranges from 0 to $12\sigma_{S_2}^2$.

3. Properties 3 and 4 tell us that if the measure is 0, then it means the explanatory variables contribute nothing to the construction of the checkerboard copula regression function and $12\sigma_{S_2}^2$ is an upper bound for the measure.

4. Property 5 tells us that the measure is the lower bound on the average proportion of variance for Y with respect to its checkerboard copula scores and its marginal distribution explained by the checkerboard copula regression.

5. Properties 6 and 7 imply that the measure can be applied to nominal explanatory variables and a binary response variable.

Notice from property 1 that the upper bound of the measure in fact depends on the marginal distribution of Y. Thus, Wei & Kim (2021) further propose a scaled version such that the scaled CCRAM $\rho^2_{(X\to Y)}$ ranges from 0 to 1:

$$\rho^{2*}_{(X\to Y)} = \frac{\rho^2_{(X\to Y)}}{12\sigma_{S_2}}. \tag{4.10}$$

Example 12 (Example 8 continued)**.** Using the checkerboard copula regression of V on U, we calculated the measure, upper bound and scaled measure to be 27/32, 27/32, and 1 respectively. The measure being 1 implies that X perfectly explains the variation of Y. Note that this result stems from the fact that the regression equals only one checkerboard score of Y for each interval in U.

4.5 Estimation

4.5.1 Model-Based Estimation of the Checkerboard Copula Regression Association Measure From Wei & Kim (2021)

Wei & Kim (2021) also provide model-free estimators of the checkerboard copula score, checkerboard copula regression, and checkerboard copula based association measure where they use the observed cell count $\mathbf{n} = \{n_{ij}\}$ in an $I \times J$ contingency table for two ordinal variables X, Y. The marginal sums of the i^{th} category of X are denoted as $n_{i\bullet} = \sum_{j=1}^{J} n_{ij}$ and the marginal sums of the j^{th} category of Y are denoted as $n_{\bullet j} = \sum_{i=1}^{I} n_{ij}$. Estimators for $p_{ij}, p_{i\bullet}, p_{\bullet j}, p_{j|i}, p_{i|j}$ as well as the definition of n are given below:

1. $p_{ij} : \hat{p}_{ij} = n_{ij}/n$.
2. $p_{i\bullet} : \hat{p}_{i\bullet} = n_{i\bullet}/n$.
3. $p_{\bullet j} : \hat{p}_{\bullet j} = n_{\bullet j}/n$.
4. $p_{i|j} : \hat{p}_{i|j} = \hat{p}_{ij}/\hat{p}_{\bullet j}$ if $\hat{p}_{\bullet j} \neq 0$, else 0.
5. $p_{j|i} : \hat{p}_{j|i} = \hat{p}_{ij}/\hat{p}_{i\bullet}$ if $\hat{p}_{i\bullet} \neq 0$, else 0.
6. $n : \sum_{i=1}^{I} \sum_{j=1}^{J} n_{ij}$.

In addition, the range of the marginal cdf of X, D_1, is estimated by $\widehat{D_1} = \{\hat{u}_0 \cdots \hat{u}_i \cdots \hat{u}_I\}$ where $\hat{u}_0 = 0$ and $\hat{u}_i = \sum_{s=1}^{i} \hat{p}_{s\bullet}$. Similarly, the range of the marginal cdf of Y, D_2, is estimated by $\widehat{D_2} = \{\hat{v}_0 \cdots \hat{v}_j \cdots \hat{v}_J\}$ where $\hat{v}_0 = 0$ and $\hat{v}_j = \sum_{t=1}^{j} \hat{p}_{\bullet t}$.

We can use the above estimators to find the following estimates in a model-free way:

- Checkerboard copula score for Y: $\{\hat{s}_1^2, ..., \hat{s}_J^2\}$ where $\hat{s}_j^2 = (\hat{v}_{j-1} + \hat{v}_j)/2$

- Variance of S_2: $\hat{\sigma}^2_{\hat{S}_2} = \left(\sum_{j=1}^{J} \hat{v}_{j-1}\hat{v}_j\hat{p}_{\bullet j}\right)/4$
- Checkerboard copula regression of V on U:

$$\hat{r}_{V|U}(u) = \sum_{j=1}^{J} \hat{p}_{j|i}\hat{s}_j^2 \quad \text{for } \hat{u}_{i-1} < u \leq \hat{u}_i. \tag{4.11}$$

With the estimated checkerboard copula regression and the prediction procedure from Example 10, we can obtain the predicted category of a response variable for each category of the explanatory variable. Like before, for some category of X, we can find the corresponding $\hat{u}^* \in \hat{D}_1$ and obtain the estimated value of the checkerboard copula regression, $\hat{v}^* = \hat{r}_{V|U}(\hat{u}^*)$. Then from $\hat{D}_2$, we can find an j^* and $\hat{v}_j^*$ such that $\hat{v}_{j^*-1} < \hat{v}^* \leq \hat{v}_{j^*}$ and obtain the predicted category of Y, $\hat{y}_{j^*}$.

Finally, we estimate the measure $\rho^2_{(X\to Y)}$ by:

$$\hat{\rho}^2_{(X\to Y)} = 12\sum_{i=1}^{I}\left(\sum_{j=1}^{J}\hat{p}_{j|i}\hat{s}_j^2 - \frac{1}{2}\right)^2 \hat{p}_{i\bullet} \tag{4.12}$$

and the scaled version by:

$$\hat{\rho}^{2*}_{(X\to Y)} = \frac{\hat{\rho}^2_{(X\to Y)}}{12\hat{\sigma}^2_{\hat{S}_2}}. \tag{4.13}$$

4.5.2 Model-Based Estimation of the Checkerboard Copula Regression Association Measure

We propose a new way of estimating the CCRAM, one that relies on parametric model-based estimates of the joint probabilities. To this end, we consider regression models used for ordinal responses to take advantage of the category ordering. A popular method is the cumulative logit model which can be thought of as a generalization of the logistic regression model when the response variable has more than two categories. For a response variable Y, the cumulative probability for outcome category n

is

$$P(Y \leq j) = \sum_{i=1}^{j} P(Y = i)$$

where $j = 1, \cdots, c$ and c denotes the number of categories. Then the logits of the cumulative probabilities, called cumulative logits, are:

$$\text{logit}[P(Y \leq j)] = \log \left[\frac{P(Y \leq j)}{1 - P(Y \leq j)} \right]$$

for $j = 1, \cdots, c - 1$. Note that we do not use the last category since the probability of observing any category less than or equal to the last category is 1 (Agresti, 2019). See Fullerton & Anderson (2021) for a comprehensive overview of ordered regression models.

Consider a categorical variable X and an ordinal variable Y with I and J ordered levels respectively. Denote the levels of X as $(x_1, \cdots, x_i, \cdots, x_I)$ and Y as $(y_1, \cdots, y_j, \cdots, y_J)$. Suppose we fit a proportional odds cumulative logit model (proportional odds means we assume the explanatory variables has the same effect for all cumulative logits). Once fitted, we are able to obtain estimated conditional probabilities of $p^M_{j|i}$, which we denote $\widehat{p}^M_{j|i}$, to indicate it is model-based, i.e., $\widehat{p}^M_{j|i} = P(Y = y_j | X = x_i)$ where $i = 1, \cdots, I$ and $j = 1, \cdots, J$. Then we estimate $p_{i\bullet}$ by $\hat{p}_{i\bullet} = n_{i\bullet}/n$ and with these two estimates, we estimate the joint probability mass function of X and Y, denoted as $\hat{P}^M = \{\hat{p}^M_{ij}\}$ by the chain rule:

$$\widehat{p}^M_{j|i} \widehat{p}_{i\bullet} = \widehat{p}^M_{ij}.$$

With our model-based estimate of the joint pmf of X and Y, we propose model-based estimates for the following:

1. $p_{ij} : \hat{p}^M_{ij}$.

2. $p_{i\bullet} : \hat{p}^M_{i\bullet} = \sum_{j=1}^{J} \hat{p}^M_{ij}$.

3. $p_{\bullet j} : \hat{p}^M_{\bullet j} = \sum_{i=1}^{I} \hat{p}^M_{ij}$.

4. $p_{i|j} : \hat{p}^M_{i|j} = \hat{p}^M_{ij} / \hat{p}^M_{\bullet j}$ if $\hat{p}^M_{\bullet j} \neq 0$ else 0.

5. $p_{j|i} : \hat{p}^M_{j|i} = \hat{p}^M_{ij} / \hat{p}^M_{i\bullet}$ if $\hat{p}^M_{i\bullet} \neq 0$ else 0.

In addition, the range of the marginal cdf of X, D_1, is estimated by $\widehat{D}^M_1 = \{\hat{u}^M_0 \cdots \hat{u}^M_i \cdots \hat{u}^M_I\}$ where $\hat{u}^M_0 = 0$ and $\hat{u}^M_i = \sum_{s=1}^{i} \hat{p}^M_{s\bullet}$. Similarly, the range of the marginal cdf of Y, D_2, is estimated by $\widehat{D}^M_2 = \{\hat{v}^M_0 \cdots \hat{v}^M_j \cdots \hat{v}^M_J\}$ where $\hat{v}^M_0 = 0$ and $\hat{v}^M_j = \sum_{t=1}^{j} \hat{p}^M_{\bullet t}$.

We can use the above estimators to find the following estimates in a model-based way:

- Checkerboard copula score for Y: $\{\hat{s}^{2M}_1, ..., \hat{s}^{2M}_J\}$ where $\hat{s}^{2M}_j = (\hat{v}_{j-1} + \hat{v}_j)/2$
- Variance of S_2: $\hat{\sigma}^{2M}_{\hat{S}^M_2} = \left(\sum_{j=1}^{J} \hat{v}^M_{j-1} \hat{v}^M_j \hat{p}^M_{\bullet j}\right)/4$
- Checkerboard copula regression of V on U:

$$\hat{r}^M_{V|U}(u) = \sum_{i=1}^{I} \hat{p}^M_{j|i} \hat{s}^{2M}_j \quad \text{for } \hat{u}^M_{i-1} < \hat{u} \leq \hat{u}^M_i. \tag{4.14}$$

Finally, we estimate the model-based $\rho^{2,M}_{(X \to Y)})$ by:

$$\hat{\rho}^{2M}_{(X \to Y)} = 12 \sum_{i=1}^{I} \left(\sum_{j=1}^{J} \hat{p}^M_{j|i} \hat{s}^{2M}_j - \frac{1}{2} \right)^2 \hat{p}^M_{i\bullet} \tag{4.15}$$

and the scaled version by:

$$\hat{\rho}^{2*M}_{(X \to Y)}) = \frac{\hat{\rho}^{2M}_{(X \to Y)})}{12 \hat{\sigma}^{2M}_{\hat{S}^M_2}} \tag{4.16}$$

With this novel measure at our disposal and two methods of estimation, we now turn towards simulations to observe its properties as well as applying it on some real world data.

Chapter 5 Simulations and Real World Application

> Don’t think—use the computer.
>
> Dyke (tongue in cheek) (1997)

In this section, the performance of the model-free CCRAM is evaluated by simulating various types of contingency tables and comparing it to model-based measures of CCRAM. A discussion follows regarding the potential of the model-free CCRAM as a goodness of fit test. The proposed measure is then applied to a real world data set.

5.1 Aims

The aim of this simulation study is to evaluate the model-free CCRAM under various types of association (no association, linear, and nonmonotone nonlinear) in a 2-way $I \times J$ contingency table with a categorical (nominal / ordinal) explanatory variable X and an ordinal response variable Y. Moreover, we compare the measure when estimated from the data itself (i.e., model-free) to measures estimated from both a model that is well-fitted and a model that is a poorly-fitted. Note that, in order to extend their simulation studies, we emulate the simulation setup of Wei & Kim (2021) which can be found in Appendix B of Wei & Kim (2021).

5.2 Simulation Set Up for an Ordinal Variable X

We consider five simulation factors in our design:

1. The type of association between X and Y

2. The magnitude of association

3. The marginal distributions of X and Y

4. Sample Size

5. Table Size, the number of categories of X and Y.

For the association scenarios, we considered three association patterns between X and Y:

1. No association - X and Y have no association

2. Linear pattern - Y increases linearly as X increases

3. Nonmonotone nonlinear pattern - Y increases quadratically as X linearly increases.

In order to simulate the contingency table with different association patterns, parametric ordinal response models also known as proportional odds cumulative logit models (CLM) were considered:

1. $\text{logit}[P(Y \leq y|X)] = \alpha_y$, for no association

2. $\text{logit}[P(Y \leq y|X)] = \alpha_y - \beta X$, for linear and monotone nonlinear association

3. $\text{logit}[P(Y \leq y|X)] = \alpha_y - \beta_1 X - \beta_2 X^2$ for nonmonotone nonlinear association

where $y = 1, 2, \cdots, J-1$, i.e. denotes all but the last category of Y. The betas, β, β_1, and β_2 are the regression coefficients and the alphas, α_ys, are the intercepts such that $\alpha_1 < \alpha_2 < ... < \alpha_{J-1}$. As will be explained later, the regression coefficients determine the magnitude of association and the intercepts determine the marginal distribution of Y.

For no association and linear association, a CLM with one predictor X was used with five values of beta indicating the magnitude of association, $\beta = (0, 0.25, 0.85, 1.4, 2)$: (no, weak, moderate, strong, very strong). Notice that $\beta = 0$ implies no effect of X on Y which is exactly what we want for no association. For nonmonotone nonlinear association, we used linear and quadratic terms of X in our cumulative logit model where $(\beta_1, \beta_2) = (12, 3)$ for tables of size 3x3 and 3x5 and $(\beta_1, \beta_2) = (18, 3)$ for tables of size 5x3 and 5x5.[1]

For the desired associations to be obtained, we also specified the marginal distributions of X and Y. For no association, linear association, and nonmonotone nonlinear association, we considered a discrete uniform distribution for X and Y which we denote as $(X, Y) = (\text{Unif}(1, I), \text{Unif}(1, J))$.

For our sample size, we considered $n = (500, 1000, 2000)$ and considered $I \times J$ contingency tables of size $3 \times 3, 3 \times 5, 5 \times 3, 5 \times 5$.

Under each association pattern (none, linear, nonmonotone nonlinear), we considered:

1. No Association: 1 association level × 1 marginal distribution of $(X, Y) \times$ 3 sample sizes × 4 table sizes = 12 experimental conditions.

2. Linear Association: 4 association levels × 1 marginal distribution of $(X, Y) \times 3$

[1] In the original paper of Wei and Kim (2021), it stated that $\beta_1 = 12$ across all table sizes; however, we were unable to replicate their results. We emailed the authors and discovered that they used different β_1 values to get the desired simulation setup. For comparison purposes, we used the same β_1 values as used in their study for different table sizes.

sample sizes × 4 table sizes = 48 experimental conditions

3. Nonmonotone nonlinear association: 1 association level × 1 marginal distribution of $(X, Y) \times 3$ sample sizes × 4 table sizes = 12 experimental conditions.

Under these various conditions, we simulated 1000 contingency tables using the following algorithm:

- Generate data given some cumulative logit model under the experimental condition, i.e., find the theoretical marginal probability mass function of X, which we denote with P_X^T, and the conditional probability mass function from the model, $P_{j|i}^T$, where $i \in \{1, \cdots, I\}$ and $j \in \{1, \cdots, J\}$ and sample from a multinomial distribution using the true joint probabilities, $P^T = \{p_{ij}^T\}$. For instance, if X is discretely uniform from 1 to 3, the theoretical probability of observing each value (1, 2, or 3) is $P(X = x) = 1/3$, where $x \in \{1, 2, 3\}$; using the chosen model we get $P(Y = y|X)$ and calculate the theoretical joint probability mass function, $P(X = x, Y = y)$.

- Next, fit a cumulative logit model of good fit and a cumulative logit model of poor fit. For no association, fit an intercept only model as the well-fitted model and a linear model as the poorly-fitted model. For linear association, fit a linear model as the well-fitted model and fit an intercept only model as the poorly-fitted model. For nonmonotone nonlinear association, fit $y \sim x + x^2$ as the well-fitted and consider two poorly-fitted models, a linear model and an intercept only model.

- Use Pearson's Goodness of Fit test to ensure that the first model is a good fit and that the other model(s) are a poor fit. If both the good model is a good fit and the poor model is a poor fit, we move on to the next step, else repeat step one. Refer to section 3.5.1 of Agresti (2010) for more detail on how to conduct Goodness of Fit tests.

- Then, we consider the estimation procedure proposed in Section 4.5.2 where we calculate the model-based estimated joint probability mass function from the estimated conditional probability mass function and the estimated marginal probability function, i.e., $\widehat{p}_{i\bullet} * \widehat{p}^{M}_{j|i} = \widehat{p}^{M}_{ij}$. Similarly, the model-free estimated joint pmf is calculated using just the data, i.e., $\widehat{p}_{ij} = \frac{n_{ij}}{n}$.

- Calculate the model-free CCRAM using Equation (4.12) and the model-based CCRAM using Equation (4.15) for both the well-fitted models and the poorly-fitted models.

We then present the simulation results of estimated CCRAMs using boxplots. Considering we have 72 experimental conditions, not all boxplots were included in this section. Instead, we include boxplots of size 3x5 for no association and linear association and all boxplots of the nonmonotone nonlinear association for the sake of brevity. The remaining boxplots can be found in Appendix B.

5.3 Simulation for No Association and Linear Association

To perform our simulations of contingency tables with no association and linear association, we apply the following the cumulative logit model (CLM) with a single predictor:

$$g_y(X) = \log(\frac{P(Y \leq y|X)}{P(Y > y|X)}) = \alpha_y - \beta X, \;\; y = 1, \cdots, J-1 \tag{5.1}$$

to simulate the contingency tables with no association and linear association.

Table 5.1: Simulation setup for no association and linear association for the CLM with an ordinal explanatory variable X. $\beta = (0, 0.25, 0.85, 1.4, 2)$ for no, weak, moderate, strong, and very strong association, respectively. The α_ys are given in the right column below, in the ascending order of the association levels. X is uniformly distributed from 1 to 3 or 5 corresponding to the table size.

Table Size	CLM Model	α_ys for No, Weak, Moderate, Strong, Very Strong Association
3x3	$g_y(X) = \alpha_y - \beta X_1$, $y = 1, 2$	$\alpha_y = [-0.69, 0.69], \alpha_y = [-0.21, 1.2]$, $\alpha_y = [0.93, 2.48], \alpha_y = [1.9, 3.7], \alpha_y = [2.9, 5.1]$
3x5	$g_y(X) = \alpha_y - \beta X_1$, $y = 1, 2, 3, 4$	$\alpha_y = [-1.39, -0.41, 0.41, 1.39]$, $\alpha_y = [-0.9, 0.09, 0.91, 1.9]$, $\alpha_y = [0.17, 1.25, 2.16, 3.23]$, $\alpha_y = [1.05, 2.27, 3.33, 4.55]$, $\alpha_y = [1.90, 3.34, 4.66, 6.10]$
5x3	$g_y(X) = \alpha_y - \beta X_1$, $y = 1, 2$	$\alpha_y = [-0.69, 0.69], \alpha_y = [0.04, 1.47]$, $\alpha_y = [1.63, 3.47], \alpha_y = [2.95, 5.45], \alpha_y = [4.30, 7.69]$
5x5	$g_y(X) = \alpha_y - \beta X_1$, $y = 1, 2, 3, 4$	$\alpha_y = [-1.39, -0.41, 0.41, 1.39]$, $\alpha_y = [-0.67, 0.33, 1.17, 2.17]$, $\alpha_y = [0.76, 2, 3.1, 4.33]$, $\alpha_y = [1.84, 3.46, 4.95, 6.55]$, $\alpha_y = [2.9, 5, 7, 9]$

In the case of no association and linear association, data for X were uniformly distributed over 1 to I (denoted as Unif(1, I)). We use the same values of α_ys from Appendix B of Wei & Kim (2021) which were selected such that the marginal distribution of Y is uniformly distributed over 1 to J, denoted as Unif(1, J)). Table 5.1 contains the specific values of the coefficients and intercepts used to obtain the desired association scenarios and distributions.

5.4 Nonmonotone Nonlinear Association

For the simulation of a contingency table with a nonmonotone nonlinear association, we consider the CLM with linear and quadratic terms:

$$g_y(X) = \log(\frac{P(Y \leq y|X)}{P(Y > y|X)}) = \alpha_y - \beta X - 3X^2, \ \ y = 1, \cdots, J-1, \qquad (5.2)$$

Table 5.2: Simulation setup for nonmonotone nonlinear association for the CLM with an ordinal explanatory variable X. $\beta = -12$ in the 3x3 and 3x5 table and $\beta = -18$ in the 5x3 and 5x5 table. X is uniformly distributed from 1 to 3 or 5 corresponding to the table size.

Table Size	CLM Model	α_ys for Weak, Moderate, Strong, Very Strong Association
3x3	$g_y(X) = \alpha_y + 12\beta X - 3X^2$, $y = 1, 2$	$\alpha_y = [-10.92, -8.91]$
3x5	$g_y(X) = \alpha_y + 12\beta X - 3X^2$, $y = 1, 2$	$\alpha_y = [-11.98, -10.46, -9.28, -8.11]$
5x3	$g_y(X) = \alpha_y + 12\beta X - 3X^2$,2 $y = 1, 2$	$\alpha_y = [-24.52, -16.62]$
5x5	$g_y(X) = \alpha_y + 12\beta X - 3X^2$, $y = 1, 2$	$\alpha_y = [-25.92, -23.91, -19.51, -15.01]$

where $\beta = -12$ for 3x3 and 3x5 contingency tables and $\beta = -18$ for 5x3 and 5x5 contingency tables.

Data for X were generated so that they were uniformly distributed over 1 to I. Then values of α_ys were selected such that the marginal distribution of Y is uniformly distributed over 1 to J. Table 5.2 displays the values of the coefficients and intercepts to obtain the desired association scenarios and distributions. See footnote 1 on page 65 for more details about these β values.

5.5 Simulation Set Up for a Nominal Variable X

For our simulations where X is a nominal explanatory variable with I levels, we employ the proportional odds cumulative logit model where we consider X as a factor with $I - 1$ indicator variables:

$$g_y(X) = \text{logit}[P(Y \leq y|X)] = \alpha_y + \tau_1\omega_1 + \cdots + \tau_{I-1}\omega_{I-1},$$

where $y = 1, \cdots, J-1$, α_ys are the intercepts such that $a_1 < \cdots < a_{J-1}$, $\omega_1, \cdots, \omega_{I-1}$ are the indicator variables of X and $\tau_1, \cdots, \tau_{I-1}$ are the coefficients that determine the effect of each level of X. For identifiability, $\tau_I = 0$, i.e., the last category of X is used as the reference. Moreover, the coefficients were chosen to determine the magnitude of the association between X and Y and the intercepts were chosen to determine the marginal distribution of Y.

Under the cumulative logit model we specified, we considered these factors: table size, sample size, and magnitude of association. For our table size and sample size, we repeated the values from the previous simulation, so 3x3, 3x5, 5x3, 5x5 and $n =$ (500, 1000, 2000). For magnitude of association, we considered the levels of no association, weak, moderate, strong, very strong association. Moreover, we employed the discrete uniform distribution for X, and α_ys were similarly chosen to get a discrete uniform marginal distribution for Y. Likewise, we followed the aforementioned algorithm for each experimental condition to get 1000 tables and computed the measure and presented with boxplots. Table 5.3 displays the values of the coefficients and intercepts to obtain the desired association scenarios and distributions.

5.6 Simulation Study Results

In this section, we present the results of our simulations by presenting boxplots of 3x5 contingency tables for no association and linear association, those of all table sizes for nonmonotone association, and those of 3x5 tables for cases with a nominal explanatory variable. Boxplots of model-free CCRAM and model-based CCRAM (well-fitted and poorly-fitted) are displayed side by side.

Table 5.3: Simulation setup for the CLM with a nominal explanatory variable X. Pairings of (α_y, τ_x) are listed for no, weak, moderate, strong, very strong association in each contingency table.

Table Size	CLM Model	α_ys and τs for No, Weak, Moderate, Strong, Very Strong Association
3x3	$g_y(X) = \alpha_y + \tau_1\omega_1 + \tau_2\omega_2$, $y = 1, 2$	$\alpha_y = [-0.69, 0.69], \tau_x = [0, 0]$, $\alpha_y = [-0.70, 0.70], \tau_x = [0.25, -0.25]$, $\alpha_y = [-0.77, 0.77], \tau_x = [0.85, -0.85]$, $\alpha_y = [-0.91, 0.91], \tau_x = [1.4, -1.4]$, $\alpha_y = [-1.11, 1.11], \tau_x = [2, -2]$
3x5	$g_y(X) = \alpha_y + \tau_1\omega_1 + \tau_2\omega_2$, $y = 1, 2, 3, 4$	$\alpha_y = [-1.39, -0.41, 0.41, 1.39], \tau_x = [0, 0]$, $\alpha_y = [-1.40, -0.41, 0.41, 1.40], \tau_x = [0.25, -0.25]$, $\alpha_y = [-1.53, -0.45, 0.45, 1.53], \tau_x = [0.85, -0.85]$, $\alpha_y = [-1.76, -0.53, 0.53, 1.76], \tau_x = [1.4, -1.4]$, $\alpha_y = [-2.10, -0.66, 0.66, 2.10], \tau_x = [2, -2]$
5x3	$g_y(X) = \alpha_y + \tau_1\omega_1 + \tau_2\omega_2 + \tau_3\omega_3 + \tau_4\omega_4$, $y = 1, 2$	$\alpha_y = [-0.69, 0.69], \tau_x = [0, 0, 0, 0]$, $\alpha_y = [-0.72, 0.72], \tau_x = [0.55, 0.25, -0.25, -0.55]$, $\alpha_y = [-0.82, 0.82], \tau_x = [1.1, 0.85, -0.85, -1.1]$, $\alpha_y = [-1.02, 1.02], \tau_x = [1.7, 1.4, -1.4, -1.7]$, $\alpha_y = [-1.31, 1.31], \tau_x = [2.3, 2, -2, -2.3]$
5x5	$g_y(X) = \alpha_y + \tau_1\omega_1 + \tau_2\omega_2 + \tau_3\omega_3 + \tau_4\omega_4$, $y = 1, 2, 3, 4$	$\alpha_y = [-1.39, -0.41, 0.41, 1.39]$, $\tau_x = [0, 0, 0, 0]$, $\alpha_y = [-1.43, -0.42, 0.42, 1.43]$, $\tau_x = [0.55, 0.25, -0.25, -0.55]$, $\alpha_y = [-1.61, -0.48, 0.48, 1.61]$, $\tau_x = [1.1, 0.85, -0.85, -1.1]$, $\alpha_y = [-1.93, -0.61, 0.61, 1.93]$, $\tau_x = [1.7, 1.4, -1.4, -1.7]$, $\alpha_y = [-2.37, -0.79, 0.79, 2.37]$, $\tau_x = [2.3, 2, -2, -2.3]$

5.6.1 Simulation Results for No Association and Linear Associations

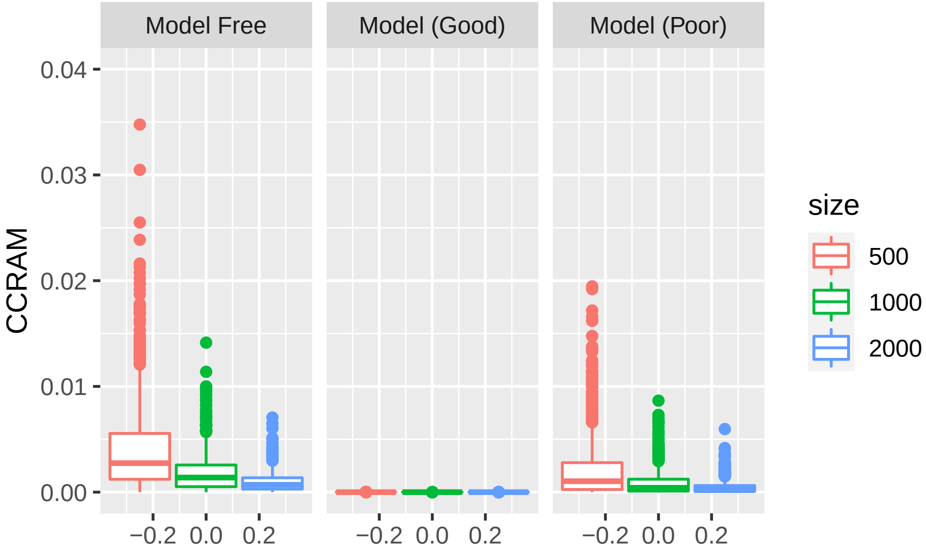

Figure 5.1: No Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

Figure 5.1 shows the boxplots of the model-free and two model-based measures from simulated 3x5 tables with no association for three sample sizes (n=500, 1000, 2000).

Because this simulation was set up so that the data had no association, we see that all three measures performed similarly in obtaining values near 0. For the model-free measure, we see that it is skewed right but the skewness decreased as the sample size increased. Even with right skew, the values of $\hat{\rho}^2_{X\to Y}$ did not vary too much with range less than 0.05. Moreover, the center slightly decreased as the sample size increased. For the well-fitted model, we see it had no variation in the distribution of the model-based measure while for the poorly-fitted model, the sampling distributions of $\hat{\rho}^2_{X\to Y}$ skewed slightly right like the model-free one but this skewness decreased as the sample size increased.

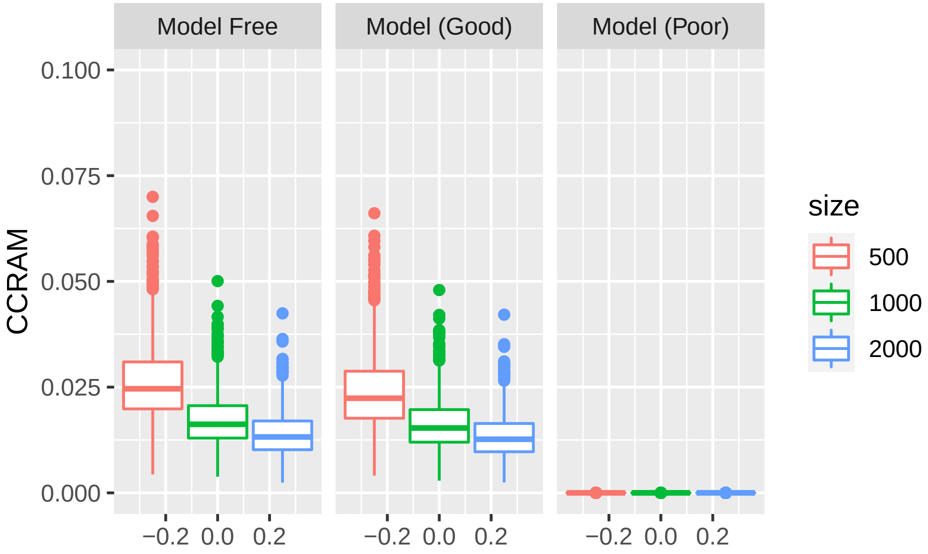

Figure 5.2: Weak Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

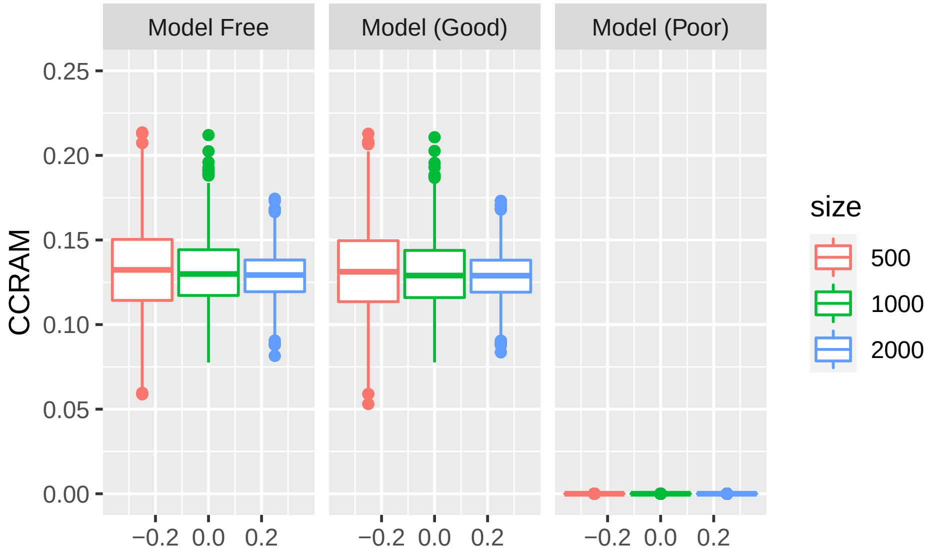

Figure 5.3: Moderate Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

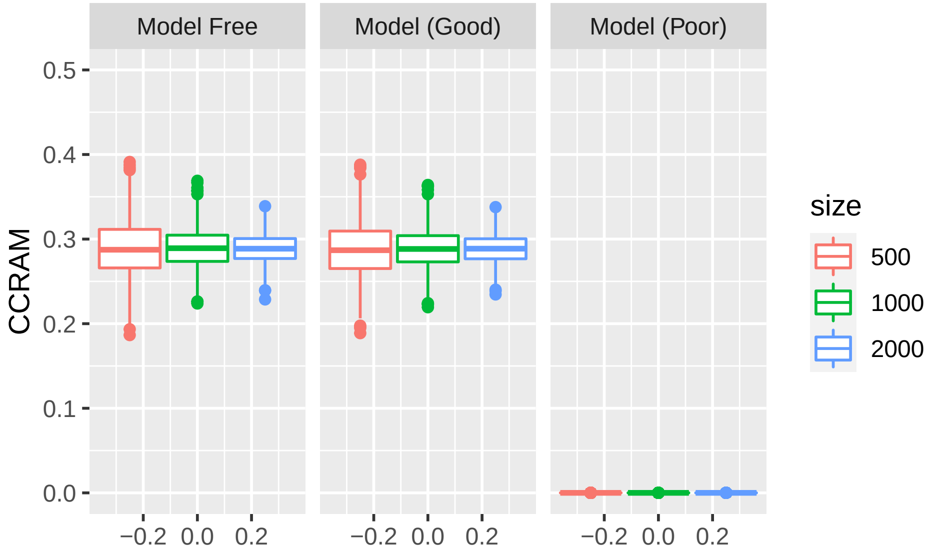

Figure 5.4: Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

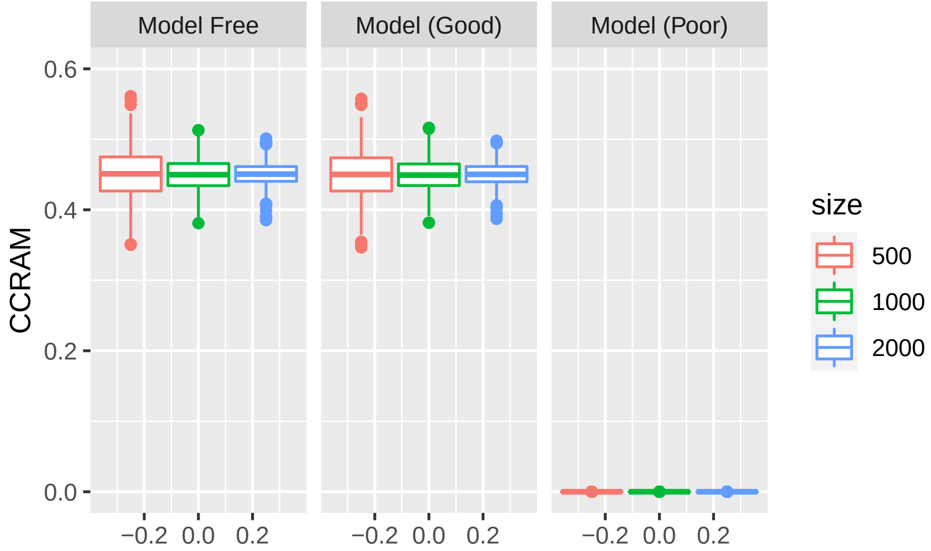

Figure 5.5: Very Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

Figures 5.2-5.5 display the boxplots of the model-free and two model-based measures from simulated 3x5 tables under 4 association levels (weak, moderate, strong, very strong) for three sample sizes (n=500, 1000, 2000). The model-free $\hat{\rho}^2_{X \to Y}$ values were skewed right and were similar in sampling distribution to the well-fitted model. Moreover, in both the model free and model (good) boxplots, as the sample size increased, the sampling variability decreased (as well as the skewness) while the center slightly decreased as the sample size increased. This disparity becomes less obvious when association is moderate or stronger. As the strength of the association increases, so does the model-free and model (good) measure. In Appendix B, it can be observed that $\hat{\rho}^2_{X \to Y}$ also increases as the table size gets bigger, especially as the number of categories in X increases. On the other hand, the poorly-fitted model did not have similar results as the other two. While the other measures compared similarly in distribution across sample sizes, the poorly-fitted model based measure did not deviate from 0, suggesting it was unable to detect nor quantify the dependence between X and Y.

5.6.2 Simulation Results for Nonmonotone Association

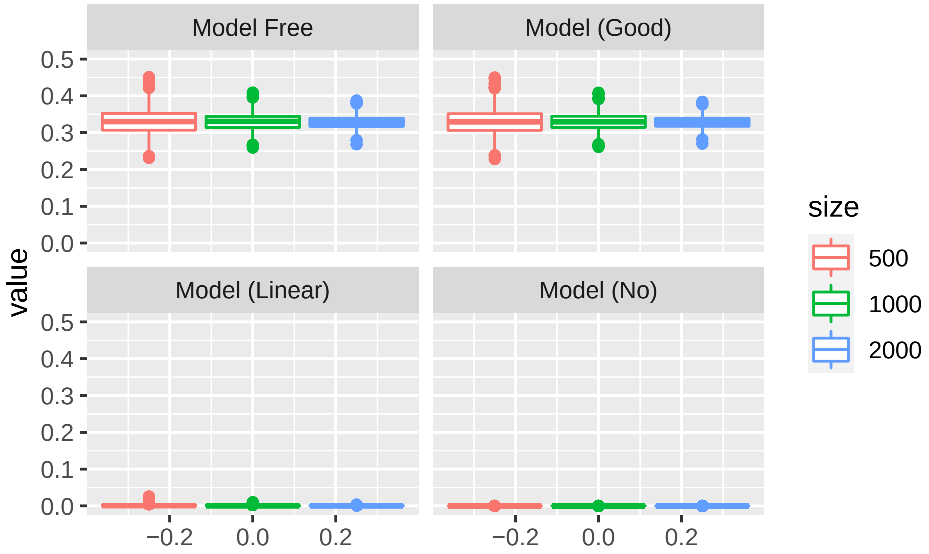

Figure 5.6: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×3 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

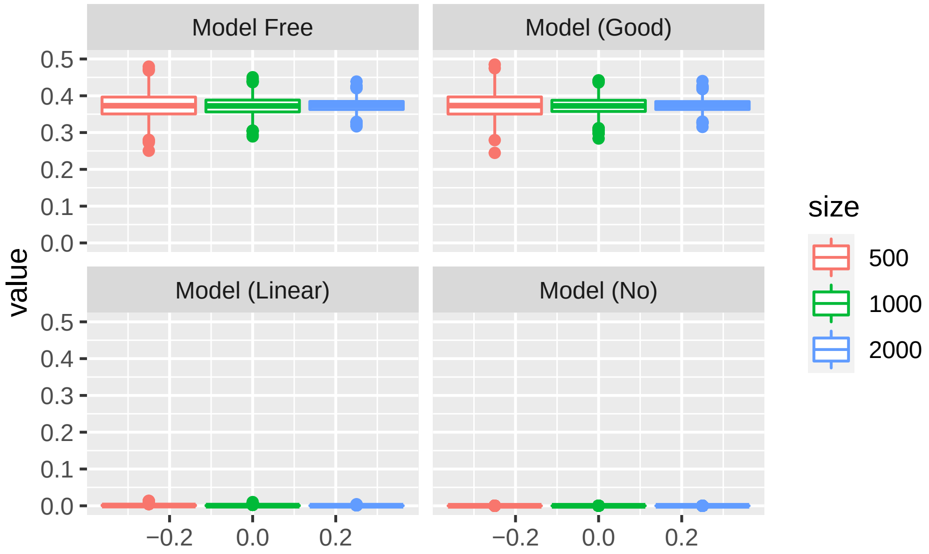

Figure 5.7: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×5 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

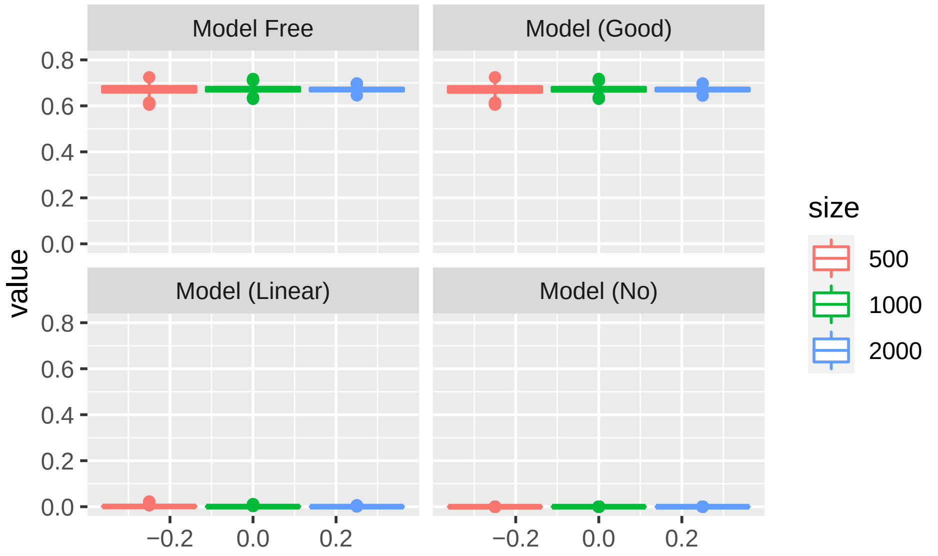

Figure 5.8: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5×3 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

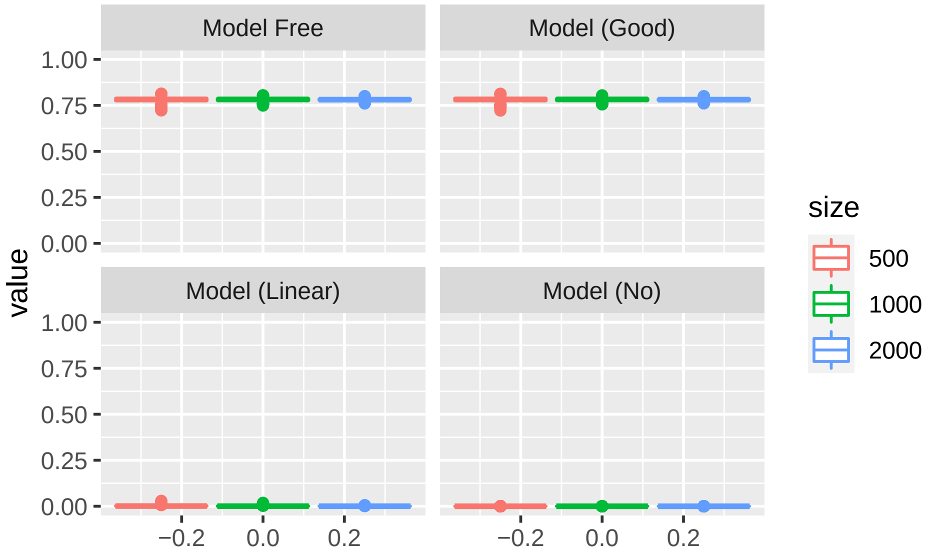

Figure 5.9: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5×5 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

Figures 5.6-5.9 show the boxplots of the model-free and three model-based measures (1 well-fitted model, 2 poorly-fitted models) from simulated tables of all table sizes considered (3x3, 3x5, 5x3, 5x5) with a nonmonotone association for three sample sizes (n=500, 1000, 2000). Similar to the previous observations made in Section 5.6.1

for the no association and linear association boxplots, the sampling variation of the model-free measure and the model-based measure from a model of good fit decreased as the sample size increased, while the centers of the two measures remained stable when the associations are nonmonotone nonlinear. It is also worth noticing that, when the table size (cell size) increases, the variation also decreases. On the contrary, for both poorly-fitted models, they were unable to identify the nonmonotone linear association. There were slight variations in both the intercept only and linear model but all estimated values did not deviate much from a value of 0.

5.6.3 Simulation Results for Nominal Explanatory Variable X

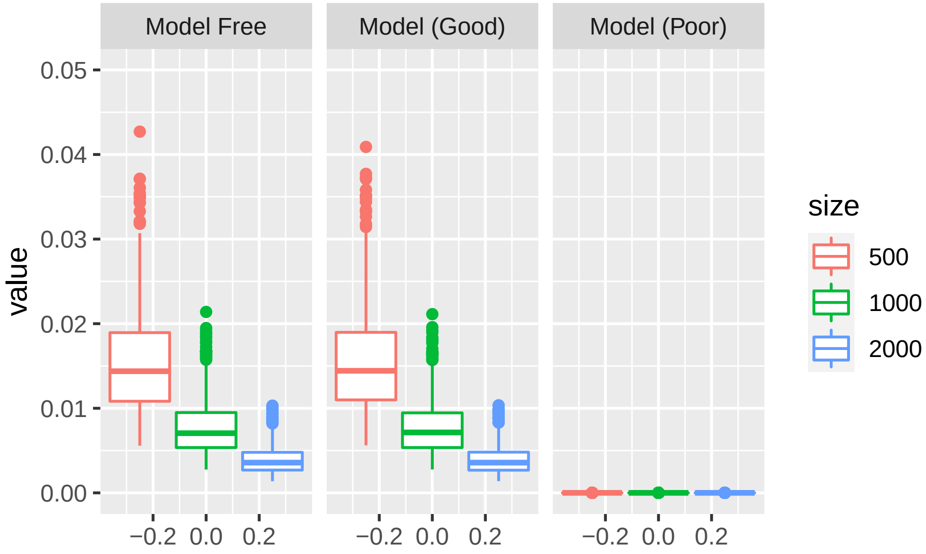

Figure 5.10: No Association. Boxplots of $\hat{\rho}^2_{X\rightarrow Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

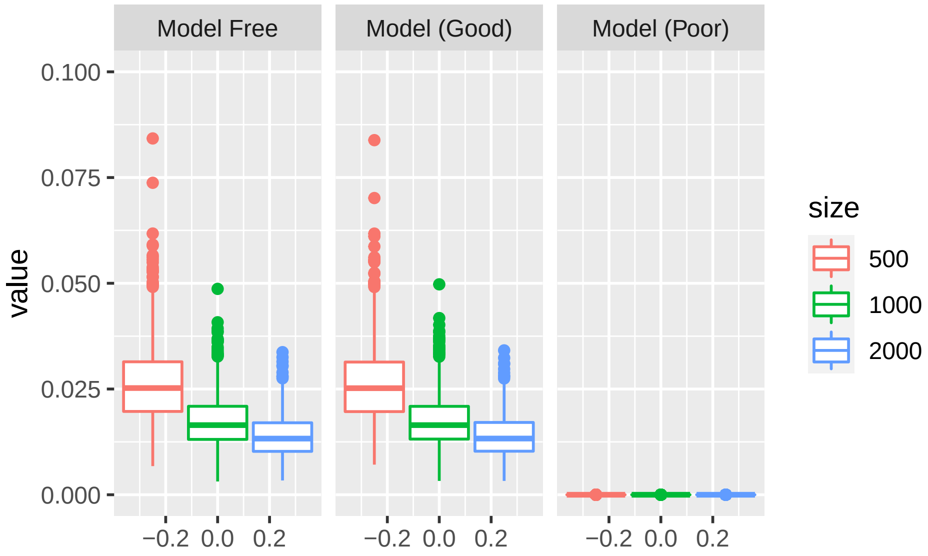

Figure 5.11: Weak Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

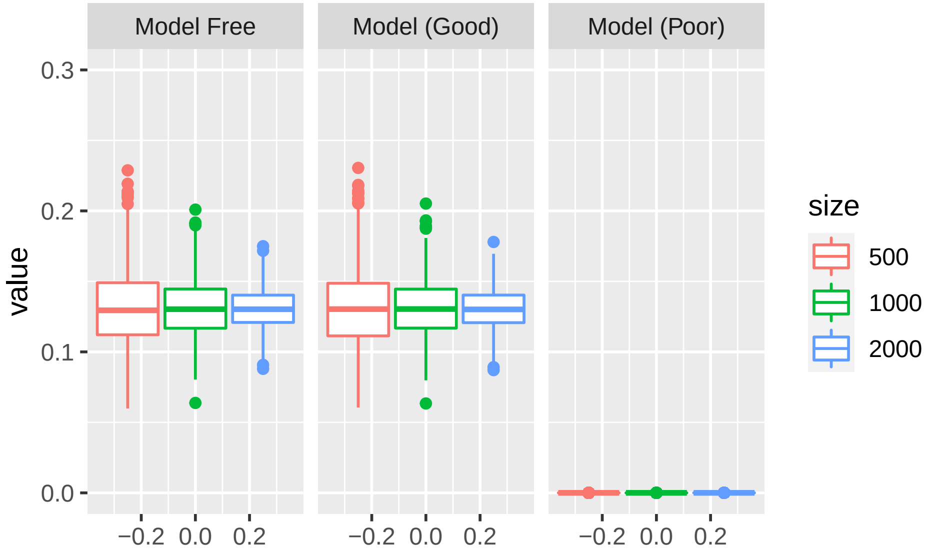

Figure 5.12: Moderate Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3×5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

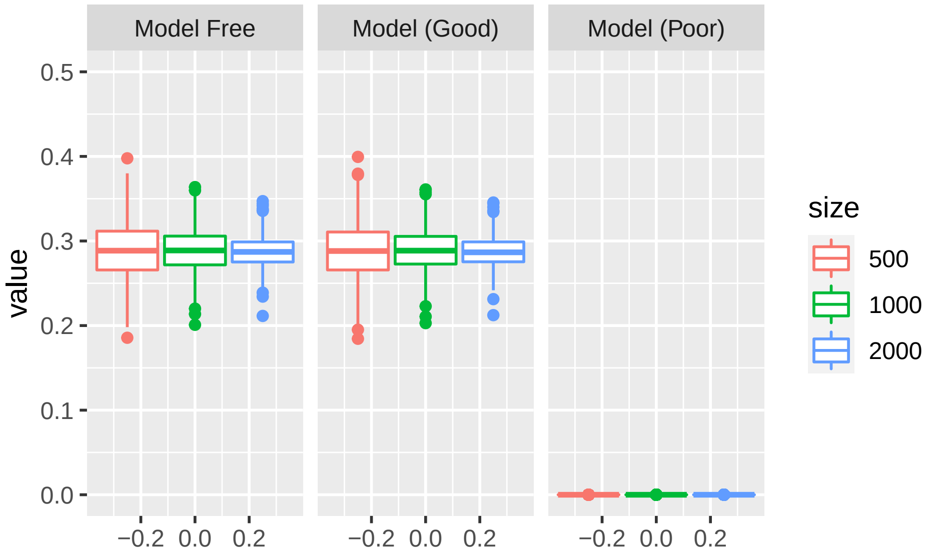

Figure 5.13: Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

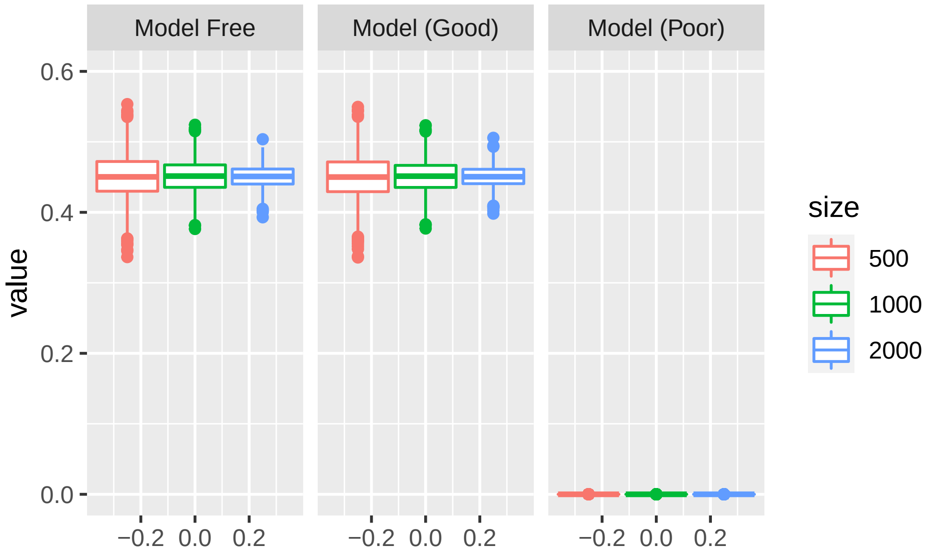

Figure 5.14: Very Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

Figures 5.10-5.14 show the boxplots of the model-free and two model-based measures from simulated 3x5 tables with a nominal X with 5 levels of association (none, weak, moderate, strong, and very strong) for three sample sizes (n=500, 1000, 2000). Similar to the previous observations made in Section 5.6.1 and Section 5.6.2, the sampling variability of the model-free measure and the model-based measure from a model of good fit decreased as the sample size increased. These two measures were also right skewed but the skewness decreased as the sample size increased. Moreover, centers decreased as the sample size increased when the association is zero or weak, but this disparity becomes less obvious when association is moderate or stronger. As the strength of association increased, the value of both measures increased as well. In Appendix B, it can be observed that $\hat{\rho}^2_{X\to Y}$ also increases as the table size gets bigger, especially as the number of categories in X increases. For the poorly-fitted model, the measure is centered around 0 without much deviation regardless of strength of association or table size (see Appendix B).

5.6.4 Discussion

In this simulation study, we observed similarities in the sampling distributions of the model-free CCRAM proposed by Wei & Kim (2021) and our proposed model-based CCRAM when the model is a good fit. The similarities between the proposed model-free measure and the model-based measure from a well-fitted model, both of which were very different than the poor one, reveal two important values of the model-free CCRAM. It not only successfully captures the structure of the dependence without model specification, but also has a potential of serving as a goodness of fit measure for model comparison and selection.

For the former, since it's not always easy or possible to identify a good parametric model for such data in real world applications, the model-free CCRAM provides us a good and practical estimate for this type of regression dependence which often exists in high-dimensional contingency tables, For the latter, for those interested in parametric modeling, the model-free CCRAM can also help identify or evaluate a chosen parametric model; if one would like to consider a parametric model for multivariate categorical data with regression dependence based on an ordinal response, they can calculate both the model-based CCRAM (based on the model of their choice) and the model-free CCRAM, and compare the two to see if they are roughly the same and not close to zero. If both were close to zero, it tells us that the chosen categorical explanatory variables might have very little contribution to the ordinal response variable. The basic idea behind this potential approach is that if the two non-zero measures were roughly the same, it suggests that the chosen model possibly fits the data well. On the other hand, disparities between the two measures may reflect the deviation of the chosen model from the data collected, urging the researcher to consider a different parametric model instead.

5.7 Real Data Analysis

One of the major political issues that Americans face is the polarization of politics. In a report titled, "Political polarization in the American public," the Pew Research Center writes, "Republicans and Democrats are more divided along ideological lines ... than at any point in the last two decades." Does one's political ideology align with party affiliations? Using data from the 2016 General Social Survey relating political ideology and political affiliation in the United States based on sex, we examine the exploratory utility of the model-free CCRAM in answering this question.[2]

To this end, we emulate Agresti (2019)'s construction of a multiway ($2 \times 2 \times 5$) contingency table in Table 5.4 with three variables: sex (S), political party (P), and political ideology (I). Subjects ($n = 661$) were chosen if they identified themselves as 1 for "strong Democrats" or 2 for "strong Republicans." Political ideology has a five-point ordinal scale, (1 = Very Liberal, 2 = Slightly Liberal, 3 = Moderate, 4 =

[2] The General Social Survey is a nationally representative survey of adults in the United States and collects data regarding people's opinions, attitudes, and behaviors. This survey is quite useful for policy-makers and researchers that are interested in the sociological contour of America.

Table 5.4: Political Ideology by Sex and Political Party Affiliation (n = 661)

		Political Ideology (I)				
Sex (S)	Political Party (P)	Very Liberal	Slightly Liberal	Moderate	Slightly Conservative	Very Conservative
Female	Democrat	25	105	86	28	4
	Republican	0	5	15	83	32
Male	Democrat	20	73	43	20	3
	Republican	0	1	14	72	32

Table 5.5: Estimated model-free CCRAMs and their BCa confidence intervals using Table 5.4 for three relationships. $P, S \to I$ denotes the relationship where sex and political party are explanatory variables and ideology is the response variable. $S \to I$ and $P \to I$ similarly denote relationships with only one explanatory variable, either sex or political party.

Total (n = 661)	$\hat{\rho}^2_{P,S\to I}$	$\hat{\rho}^2_{S\to I}$	$\hat{\rho}^2_{P\to I}$
Estimate	0.460	0.002	0.459
95% BCa bootstrap CI	(0.404, 0.511)	(0.000, 0.014)	(0.400, 0.501)

Slightly Conservative, 5 = Very Conservative) and sex was categorized by 1 = female, 2 = male.

Because the model-free CCRAM identifies regression dependence, we are interested in comparing the value of CCRAM measure when political ideology is the response variable with different explanatory variables. When we consider both sex and political party as explanatory variables, we denote this relationship by $S, P \to I$ Furthermore, we examine the measure with a single explanatory variable, either sex or political party, denoted $S \to I$ and $P \to I$ respectively. To examine these dependence structures, the calculated proposed measure for these three relationships and their 95% bootstrap bias-corrected and accelerated (BCa) confidence intervals are displayed in Table 5.5.

After computing the measures, we observe that $\hat{\rho}^2_{P,S\to I} = 0.460$ and the corresponding 95% confidence interval is $(0.404, 0.511)$. This suggests a moderately strong association between sex and party as explanatory variables on political ideology. This estimated association measure also tells us that the lower bound on the average proportion of variance for the checkerboard copula score of political ideology (I) explained by the checkerboard copula regression using sex (S) and political party (P) as explanatory variables is 46%. Then, the estimated upper bound of the model-free CCRAM is 0.934 ($= 12\hat{\sigma}^2_{\hat{S}_3}$) and so the rescaled CCRAM is $0.460/0.934 = 0.492$.

Now, let's turn our attention to each individual predictor. When sex is the only explanatory variable, we obtained $\hat{\rho}^2_{S\to I} = 0.002$ ($\hat{\rho}^{2*}_{S\to I} = 0.003$) indicating a weak association between sex and political ideology. When party is the only explanatory variable, we obtained $\hat{\rho}^2_{P\to I} = 0.459$ ($\hat{\rho}^{2*}_{P\to I} = 0.491$) suggesting a strong association between political party and political ideology. However, it is also important to identify potential interactions among the two explanatory variables for explanatory modeling, so bootstrap predictions were performed and the results are displayed in Figure 5.15.

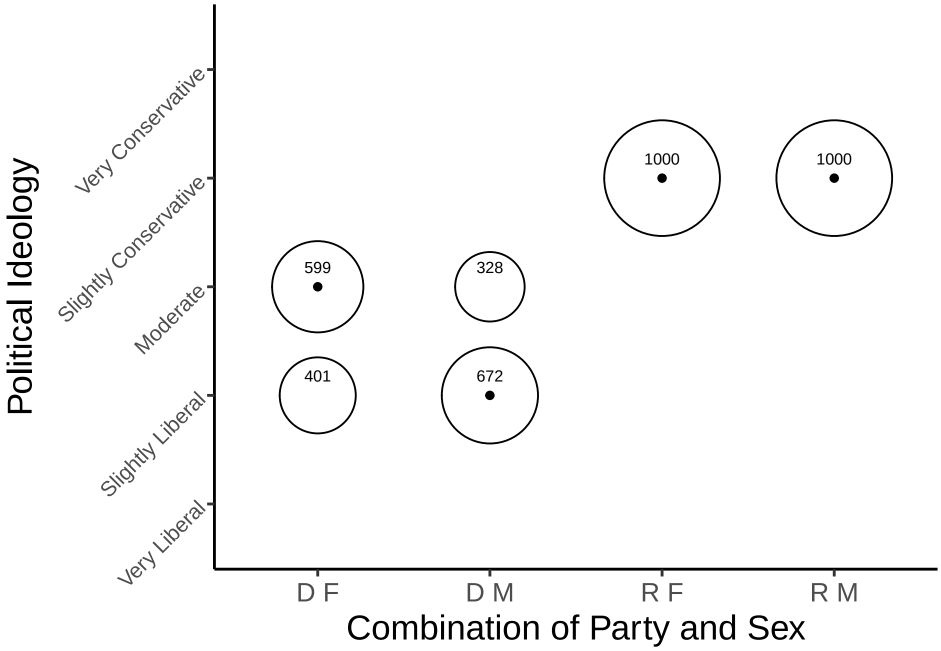

Figure 5.15: Predicted category of ideology by the checkerboard copula regression for each combination of party and sex. The first letter denotes the party (D = Democrat and R = Republican) and the second the sex (F = Female, M = Male). The size of the circle indicates proportion of each category of political ideology estimated by the regression in 1000 bootstraps. The dark dot at each combination represents the predicted level of political ideology for that combination.

Without fitting any model, Figure 5.15 reveals that there might be a potential interaction between sex and political party. Notice for Democrats (D), females tend to be moderate while males tend to be slightly liberal. While for Republicans (R), regardless of sex, the predicted levels of political ideology from all bootstrap samples are all slightly conservative. Using plots like the one above makes it easy to explore potential interactions in 3-way or higher-dimensional contingency tables.

After looking at these model-free measure and bootstrap predictions, we fit a cumulative logit model under the proportional odds assumption with an interaction term between sex and political party. We display the model output below.

```

Call:
VGAM::vglm(formula = cbind(y1, y2, y3, y4, y5) ~ sex * party,
    family = cumulative(parallel = TRUE), data = table_ideology)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept):1   -2.177      0.176  -12.36   <2e-16 ***
(Intercept):2    0.118      0.123    0.96     0.34
(Intercept):3    1.808      0.156   11.59   <2e-16 ***
(Intercept):4    4.604      0.239   19.28   <2e-16 ***
sex2             0.182      0.188    0.97     0.33
party2          -3.480      0.254  -13.71   <2e-16 ***
sex2:party2     -0.361      0.311   -1.16     0.25
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

Names of linear predictors: logitlink(P[Y<=1]),
logitlink(P[Y<=2]), logitlink(P[Y<=3]), logitlink(P[Y<=4])

Residual deviance: 8.45 on 9 degrees of freedom

Log-likelihood: -34.5 on 9 degrees of freedom

Number of Fisher scoring iterations: 4

No Hauck-Donner effect found in any of the estimates
```

```
Exponentiated coefficients:
       sex2      party2 sex2:party2
     1.1994      0.0308      0.6969
```

First, let's check if the proportional odds assumption holds as well as the overall fit of this model. To do the former, we consider the Brant test from Brant (1990) to check proportional odds using the `brant` package in R from Schlegel & Steenbergen (2020). The results of this test suggest the proportional odds assumptions hold so now we check the overall fit. From the summary output, the deviance for the cumulative logit model is 9.81 based on $df = 9$ and calculations tell us that the p-value is 0.489 so the model fits well.

Looking at the summary output, we see that the estimated odds that Republican responds in the liberal direction rather than the conservative direction is equal to $\exp(\hat{\beta}) = \exp(-3.48) = 0.03$ times the estimated odds for Democrats. Likewise, the estimated odds that Republican responds in the conservative direction rather than the liberal direction is equal to $\exp(\hat{\beta}) = \exp(3.48) = 32.46$ times the estimated odds for Democrats. Furthermore, there does not seem to be evidence of a sex effect or interaction effect. Recall that Figure 5.15 suggests a potential interaction between sex and political party, but this is not shown to be significant from this fitted model. While it's very likely that the interaction effect is indeed not strong enough to be significant (after all, Figure 5.15 is only for EDA purposes), it's worth noting that the above interpretations and findings are only valid if the model assumptions hold and if the sample size is large enough to ensure the corresponding asymptotic theory holds. While we can verify the former by checking the proportional odds assumption which the Brant test tells us is okay, the latter is often harder to confirm. Both of them remind us of the common challenges and limitations to parametric modeling.

Now, let's calculate the model-based rescaled CCRAMs based on the following proportional odds cumulative logit models for political ideology: (1) Sex only, (2) Party only, (3) Sex and Party without interaction, and (4) Sex and Party with interaction. From a model that had sex as the only predictor of political ideology, our model-based rescaled CCRAM is 0.003. Our model with only political party resulted in a model-based rescaled CCRAM of 0.488. For a model with sex and party and no interactions, the model-based rescaled CCRAM was 0.488. Finally, the model-based rescaled CCRAM for the model with the interaction term was 0.489.

These model-based CCRAMs are similar to the values obtained by the model-free CCRAM as shown in Table 5.5. One way we can think about these results in familiar terms is through the lens of R^2, the coefficient of determination, values for linear models. Recall that in parametric modeling, R^2 gives us a notion of how well the model fits or how much variation in the response variable the model explains for. In a similar light, that is what the CCRAM, model-based or model-free, represents, but for multi-way contingency tables. When we considered only sex in the model, we obtained very low model-based and model-free CCRAMs, suggesting that sex only explains almost no variation in ideology so the model is not a good fit. For the other three models, we see that all these CCRAM values, model-based and model-free are similar to each other which tells use 2 things. It tells us that these parametric models might be good choices and that among these three models, adding an additional term does not add much to the explained variation in the ordinal response variable. For instance, adding the interaction term to the model would only increase the explained variation by 1% (i.e. the difference in the rescaled model-based CCRAM between the models with and without the interaction is 0.489-0.488 = 0.01). If we were thinking in the sense of R^2, then what should we do? Well, if they are similar, we often go for the more parsimonious and simpler model, so we recommend the party only model for predicting ideology in this case.

Overall, our explanatory modeling using the model-free CCRAM aligns well with the results from the parametric modeling; it not only extracts similar information about the regression dependence without parametric modeling, but also has great potential in helping identify good parametric models — all of which echoes our findings from the simulation studies.

Chapter 6 Conclusion

> A conclusion is simply the place
> where you got tired of thinking.
>
> Dan Chaon, Stay Awake

In summary, this thesis explored a novel model-free regression dependence measure for ordinal response variables and categorical explanatory variables as well as its potential to be a goodness of fit measure. In order to arrive at the methodology, we first familiarized ourselves with copulas in the continuous case and several key results that make copulas useful in characterizing the dependence between random variables. The key results included Sklar's theorem, the Invariance Principle, and the Fréchet Hoeffding Bounds. After arming the reader with the tools they needed, we discussed some of the limitations when transitioning into the discrete case. Here we saw an array of problems that could occur, making copulas less than ideal for characterizing dependence. Issues ranged from the nonuniqueness of the copula to the lack of interpretation. However, we saw that there does exist one copula, the checkerboard copula, that bridges the gap between the discrete case and the continuous case.

We then turned our attention to how the checkerboard copula is used in the creation of a regression based association measure comprised of three parts: the score, the regression function, and the measure. The checkerboard copula score can be seen as the set of the average of the marginal distributions calculated at every two consecutive categories of some variable. This score was used in defining the checkerboard copula regression function which can be interpreted as the mean checkerboard score of some response variable with respect to the conditional distribution of the explanatory variable(s). Based on the regression function, we examined the model-free checkerboard copula regression association measure which represents the magnitude of the explanatory power of the explanatory variable(s) in the checkerboard copula regression. In addition to this model-free measure, we proposed a model-based version of this measure where the joint probabilities are estimated using a model as opposed to directly from the data.

With both measures in hand, we simulated contingency tables to evaluate the performance of the model-free and the model-based measure. Specifically, we examined the model-free and model-based measures under various types and strengths of association as well as the sample size and the size of the table itself. We observed similar sampling distribution patterns when comparing the model-free measure to the model-based measure from a model that is well fitted to the data. On the other hand, a model that was poorly fitted to the data differed drastically in the sampling distribution of the measure. In addition, we applied our measure onto a real world data set concerning the political ideology of people cross-classified by sex and political party where we saw that our model-free exploratory modeling hinted at results corroborated by parametric modeling. While it is important to note that the proposed measure does not indicate the statistical significance of parametric ordinal models with the same explanatory variables, the simulations and real world applications demonstrate the utility of this measure as an exploratory approach in not only quantifying the association between multiple categorical variables, but also helping identify complex regression dependence structures between an ordinal response variable and multiple categorical (ordinal or nominal) explanatory variables.

Our exploratory results suggest several advantages of the model-free CCRAM. First, it is a model-free approach meaning it does not get bogged down by some of the steps we take in parametric modeling; that is, specifying a dependence structure in advance, ensuring conditions are met and running model diagnostics. In addition, one should keep in mind that sparseness is always an issue, especially for high-dimensional categorical data, and one might not be able to even fit a parametric model on a multi-way contingency table. Moreover, the model-free measure acts as a compliment to existing approaches. It provides a new way to fully explore categorical data for EDA purposes by quantifying the association between multiple categorical variables, visualizing multi-way contingency tables and potential interactions. Lastly, our simulation results hint at the possibility of using the model-free measure as a goodness of fit measure. We can think of it as a R^2-like measure for categorical data and explore what the model free CCRAM can potentially do for a regression model based on an ordinal response, just like what R^2 can do in a multiple linear regression model — based on a quantitative response.

Given the time constraints of a thesis, we were able to only explore certain association scenarios. It would be beneficial to consider other associations like monotone nonlinear ones as well as consider simulations with more than one explanatory variable (which will involve simulating multi-way contingency tables with an ordinal response variable) to see if our initial results are similar. Furthermore, an exciting future work in this area is using the model-free CCRAM as a goodness of fit point estimator and derive the asymptotic theory for testing the fit of a parametric model.

Appendix A Hidden Code Chunks

This first appendix includes all of the R chunks of code that were hidden throughout the document.

A.1 Code for Chapter 2

Loading in packages used throughout the chapter

Code used in Section 2.2 for our motivating example. We first generate the two data sets.

```r
set.seed(713) # reproducibility
d <- 2 # dimensions
n <- 1000 # sample size
sigma <- matrix(c(1, 0.65, 0.65, 1), nrow = 2) # correlation matrix

norm_norm <- MASS::mvrnorm(n, mu = rep(0, 2), Sigma = sigma) |>
  as_tibble()
u <- norm_norm |>
  mutate(
    X1 = pnorm(V1),
    X2 = pnorm(V2)
  )
beta_exp <- u |>
  mutate(
    Y1 = qbeta(X1, shape1 = 10, shape2 = 5),
    Y2 = qexp(X2, rate = 1)
  )
```

Code for Figure 2.1.

```r
p1 <- ggplot(u, aes(x = V1, y = V2)) +
  geom_point() +
  ylab(expression(~ X[2])) +
  xlab(expression(~ X[1]))

p1 <- ggMarginal(p1)
p2 <- ggplot(beta_exp, aes(x = Y1, y = Y2)) +
  geom_point() +
  ylab(expression(~ Y[2])) +
  xlab(expression(~ Y[1]))
p2 <- ggMarginal(p2)
grid.arrange(p1, p2, nrow = 1)
```

Code for Figure 2.2.

```r
u_norm_norm <- norm_norm |>
  mutate(
    V1 = pnorm(V1),
    V2 = pnorm(V2)
  )
u_beta_exp <- beta_exp |>
  mutate(
    V1 = pbeta(Y1, shape1 = 10, shape2 = 5),
    V2 = pexp(Y2, rate = 1)
  )

p1_u <- ggplot(u_norm_norm, aes(x = V1, y = V2)) +
  geom_point() +
  ylab(expression(`F`[2](X[2]))) +
  xlab(expression(`F`[1](X[1])))

p1_u <- ggMarginal(p1_u)
p2_u <- ggplot(u_beta_exp, aes(x = V1, y = V2)) +
  geom_point() +
  ylab(expression(`G`[2](Y[2]))) +
  xlab(expression(`G`[1](Y[1])))
p2_u <- ggMarginal(p2_u)
grid.arrange(p1_u, p2_u, nrow = 1)
```

Code for Figure 2.3.

```
norm_norm_2 <- u_beta_exp |>
  mutate(
    V1 = qnorm(V1),
    V2 = qnorm(V2)
  )
p1_2 <- ggplot(norm_norm_2, aes(x = V1, y = V2)) +
  geom_point() +
  ylab(expression(`F`[2]^-1 ~ (G[2](Y[2])))) +
  xlab(expression(`F`[1]^-1 ~ (G[1](Y[1]))))

p1_2 <- ggMarginal(p1_2)

# grid.arrange(p1, p1_2, nrow= 1)
grid.arrange(p1, p1_2, nrow = 1)
```

Code for Figure 2.4.

```
# par(mfrow=c(r=1,c=2))
library(copula)
d <- 2 # 2 dimensions
# create an independent copula object
ic <- copula::indepCopula(dim = 2)
plot1 <- wireframe2(ic, FUN = pCopula, xlab = "u", ylab = "v")
plot2 <- contourplot2(ic, FUN = pCopula, xlab = "u", ylab = "v")
gridExtra::grid.arrange(plot1, plot2,
  nrow = 1
)
```

Code for Figure 2.5.

```
d <- 2 # dimension
theta <- -9 # copula parameter
fc <- frankCopula(theta, dim = d) # define a Frank copula

set.seed(713)
n <- 5 # number of evaluation points
u <- matrix(runif(n * d), nrow = n) # n random points in [0,1]^d
frankPlot <- wireframe2(fc,
  FUN = pCopula, # wireframe plot (copula)
  draw.4.pCoplines = FALSE,
  xlab = "u",
  ylab = "v",
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(bottom.padding = -8),
    layout.widths =
      list(right.padding = -20)
  )
)
frankDensityPlot <- wireframe2(fc,
  FUN = dCopula, delta = 0.001,
  lwd = 1 / 2,
  xlab = "u",
  ylab = "v",
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(bottom.padding = -8)
  )
) # wireframe plot (density)
frankPlotContour <- contourplot2(fc,
  FUN = pCopula,
  xlab = "u",
  ylab = "v",
  par.settings = list(
    axis.text = list(cex = .5),
    layout.widths =
      list(right.padding = -20)
  )
) # contour plot (copula)
frankDensityContour <- contourplot2(fc,
  FUN = dCopula,
  n.grid = 72, # contour plot (density)
  lwd = 1 / 2,
  xlab = "u",
  ylab = "v",
  par.settings = list(axis.text = list(cex = .5))
)
gridExtra::grid.arrange(frankPlot, frankDensityPlot,
  frankPlotContour, frankDensityContour,
  nrow = 2
  # heights=c(6,5)
  # widths=c(5,2)
```

```
)
```

Code for Figure 2.6.

```
par(mfrow = c(1, 3))
set.seed(713)
n <- 1000
U <- rCopula(n, copula = fc)
U0 <- rCopula(n, copula = setTheta(fc, value = 0))
U9 <- rCopula(n, copula = setTheta(fc, value = 9))
UPlot <- plot(U, xlab = "U", ylab = "V")
U0Plot <- plot(U0, xlab = "U", ylab = "V")
U9Plot <- plot(U9, xlab = "U", ylab = "V")
# gridExtra::grid.arrange(UPlot,U0Plot,
#                         U9Plot, nrow = 1)
```

Code for Figure 2.7.

```
nc <- normalCopula(iTau(normalCopula(), tau = 0.5))
set.seed(713)
U <- rCopula(1000, copula = nc) # sample from the normal copula
U1 <- wireframe2(nc,
  FUN = dCopula,
  delta = 0.025,
  xlab = "u", ylab = "v",
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(bottom.padding = -4)
  )
)
U2 <- contourplot2(nc,
  FUN = pCopula,
  xlab = "u", ylab = "v",
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(bottom.padding = -2)
  )
) # copula
U3 <- contourplot2(nc,
  FUN = dCopula, n.grid = 42,
  cuts = 33, lwd = 1 / 2,
  xlab = "u", ylab = "v",
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(top.padding = -4)
  )
) # density
U4 <- ggplot(data.frame(U), aes(x = X1, y = X2)) +
  geom_point() +
  labs(x = "U", y = "V") # scatter plot
gridExtra::grid.arrange(U1, U2,
  U3, U4,
  nrow = 2
)
```

Code for Figure 2.8.

```
set.seed(713) # reproducibility
par(mfrow = c(r = 1, c = 2)) # 2x2 grid
M <- runif(100) # sample 100 from a standard uniform
plot(cbind(M, 1 - M), xlab = "U", ylab = "V") # W
plot(cbind(M, M), xlab = "U", ylab = "V") # M
```

Code for Figure 2.9.

```
# par(mfrow=c(r=2,c= 2)) # 2x2 grid
u <- seq(0, 1, length.out = 40) # subdivision points in each dimension
u12 <- expand.grid("u" = u, "v" = u) # build a grid
W <- pmax(u12[, 1] + u12[, 2] - 1, 0) # values of W on grid
M <- pmin(u12[, 1], u12[, 2]) # values of M on grid
val.W <- cbind(u12, "W(u,v)" = W) # append grid
val.M <- cbind(u12, "M(u,v)" = M) # append grid
W_wire <- wireframe2(val.W,
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(bottom.padding = -8),
    layout.widths =
      list(right.padding = -20)
  )
)
```

```r
M_wire <- wireframe2(val.M,
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(bottom.padding = -8)
  )
)
W_contour <- contourplot2(val.W,
  xlim = 0:1,
  ylim = 0:1,
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights = list(bottom.padding = 0),
    layout.widths =
      list(right.padding = -20)
  )
)
M_contour <- contourplot2(val.M,
  xlim = 0:1, ylim = 0:1,
  par.settings = list(axis.text = list(cex = .5))
)
grid.arrange(W_wire, M_wire,
  W_contour, M_contour,
  nrow = 2
)
```

Code for Section 2.6.1.

```r
set.seed(713) # reproducibility
d <- 2 # dimension
rho <- 0.4 # off-diag entry of the correlation matrix
u <- runif(d) # generate a random point
x <- qnorm(u) # applying the quantile transform
# bivariate normal distribution
mvtnorm::pmvnorm(
  upper = x, corr = matrix(c(1, rho, rho, 1), nrow = 2),
  keepAttr = FALSE
)
# normal copula
nc <- normalCopula(rho)
copula::pCopula(u, copula = nc)
```

Code for Section 2.6.2.

```r
H1 <- copula::mvdc(fgmCopula(1),
  margins = c("beta", "exp"),
  paramMargins = list(list(shape1 = 7, shape2 = 3), list(rate = 1))
)

H2 <- copula::mvdc(fgmCopula(1),
  margins = c("norm", "norm"),
  paramMargins = list(list(mean = 3, sd = 2), list(mean = 0, sd = 1))
)
```

Code for Figure 2.10.

```r
X1 <- rMvdc(1000, mvdc = H1)
X2 <- rMvdc(1000, mvdc = H2)
H1_p <- ggplot(data.frame(X1), aes(x = X1, y = X2)) +
  geom_point() +
  ylab(expression(~ Y[1])) +
  xlab(expression(~ X[1]))
H2_p <- ggplot(data.frame(X2), aes(x = X1, y = X2)) +
  geom_point() +
  ylab(expression(~ Y[2])) +
  xlab(expression(~ X[2]))
H1_p <- ggMarginal(H1_p)
H2_p <- ggMarginal(H2_p)
gridExtra::grid.arrange(H1_p,
  H2_p,
  nrow = 1
)
```

Code for Figure 2.11.

```r
set.seed(713)
### Sampling from a normal copula
n <- 1000 # sample size
d <- 2 # dimension
rho <- 0.8 # off-diagonal entry in the correlation matrix P
P <- matrix(rho, nrow = d, ncol = d) # build the correlation matrix P
```

```r
diag(P) <- 1
set.seed(713)
# n ind. bivariate normal observations
X <- MASS::mvrnorm(n, mu = rep(0, d), Sigma = P)
U <- pnorm(X) # n ind. realizations from the corresponding copula
# transform U (normal copula) to beta and exp margins
Y <- cbind(qbeta(U[, 1], shape1 = 10, shape2 = 3), qexp(U[, 2],
                                                        rate = 2))

X <- X |>
  as_tibble()

X_p <- ggplot(X, aes(x = V1, y = V2)) +
  geom_point() +
  geom_point(data = X[3, ], aes(x = V1, y = V2),
             colour = "red", size = 5) +
  geom_point(data = X[78, ], aes(x = V1, y = V2),
             colour = "blue", size = 5) +
  geom_point(data = X[593, ], aes(x = V1, y = V2),
             colour = "green", size = 5) +
  labs(x = "X", y = "Y")

X_p <- ggMarginal(X_p)

U <- U |>
  as_tibble()

U_p <- ggplot(U, aes(x = V1, y = V2)) +
  geom_point() +
  geom_point(data = U[3, ], aes(x = V1, y = V2),
             colour = "red", size = 5) +
  geom_point(data = U[78, ], aes(x = V1, y = V2),
             colour = "blue", size = 5) +
  geom_point(data = U[593, ], aes(x = V1, y = V2),
             colour = "green", size = 5) +
  labs(x = "U", y = "V")

U_p <- ggMarginal(U_p)

Y <- Y |>
  as_tibble()
```

```
Y_p <- ggplot(Y, aes(x = V1, y = V2)) +
  geom_point() +
  geom_point(data = Y[3, ], aes(x = V1, y = V2),
             colour = "red", size = 5) +
  geom_point(data = Y[78, ], aes(x = V1, y = V2),
             colour = "blue", size = 5) +
  geom_point(data = Y[593, ], aes(x = V1, y = V2),
             colour = "green", size = 5) +
  labs(x = "X'", y = "Y'")
Y_p <- ggMarginal(Y_p)

grid.arrange(X_p, U_p, Y_p, nrow = 1)
```

A.2 Code for Chapter 3

Code for Figure 3.1.

```
set.seed(713)
par(mfrow = c(1, 2)) # Create a 2 x 2 plotting matrix
X <- rpois(1000, lambda = 1) # sampling from poisson
V <- runif(1000) # sampling from unif
U <- ppois(X, lambda = 1) + (V * dpois(X, lambda = 1)) # transform
plot(ecdf(ppois(X, 1)), xlab = "P(X <= x)", main = "ECDF of F(X)")
plot(ecdf(U), xlab = "P(U <= u)", ylab = "Fn(u)", main = "ECDF of U")
```

A.3 Code for Chapter 4:

In this chapter, the code below are functions we used to obtain various values in order to get the measure.

```
# get joint pmf from df where the
# rows are cases (i.e. one row is one observation)
getJointPMF <- function(data = df) {
  return(prop.table(table(data)))
}
```

```r
# gets frequency table
getJointPMF_freq <- function(data = df) {
  return(data |>
    group_by_all() |>
    dplyr::count() |>
    ungroup() |>
    mutate(freq = n / sum(n)))
}

# each joint pmf is a row
getJointPMF_2 <- function(data = df) {
  grid <- expand.grid(lapply(data, levels))
  colnames(grid) <- colnames(data)
  pmf <- getJointPMF_freq(data = data)
  combined <- suppressMessages(left_join(grid, dplyr::select(pmf, -n),
    by = colnames(grid)
  ))
  combined[is.na(combined)] <- 0
  result <- combined |>
    mutate(across(.cols = everything(), as.numeric))
  return(result)
}

# function to get the specific joint pmf of u,v pair
findJointPMF <- function(u = 0.3, v = 0.3, data = df) {
  joint_pmf <- getJointPMF(data = data)
  X <- findInterval(u, Xcdf, left.open = TRUE, rightmost.closed = TRUE)
  Y <- findInterval(v, Ycdf, left.open = TRUE, rightmost.closed = TRUE)
  return(joint_pmf[X, Y])
}

# marginal pmf of the var_index
getPMF <- function(var_index = 1, data = df) {
  if ("data.frame" %in% class(data)) {
    data <- getJointPMF(data = data)
  }
  if (var_index == 1) {
    return(rowSums(data))
  } else {
    return(colSums(data))
  }
}
```

```r
# generalized method to get PMF
getPMF_2 <- function(var_index = 1, data = df) {
  if ("data.frame" %in% class(data)) {
    return(data |>
      dplyr::count(data[, var_index]) |>
      dplyr::mutate(freq = n / sum(n)) |>
      dplyr::select(freq) |>
      unlist())
  } else {
    if (var_index == 1) {
      return(rowSums(data))
    } else {
      return(colSums(data))
    }
  }
}


# marginal pmf without var_index
getPMFwo <- function(var_index = 1, data = df) {
  without <- data[, -var_index] |>
    data.frame()
  colnames(without) <- colnames(data)[-var_index]
  return(without |>
    group_by_all() |>
    dplyr::count() |>
    ungroup() |>
    mutate(freq = n / sum(n)))
}


# getting conditional pmf tables
getConditionalPMF <- function(var_index = 2, data = df) {
  if ("data.frame" %in% class(data)) {
    data <- getJointPMF(data = data)
    # regardless of var_index, we make the number of rows to be
    # length of X and columns to be length of Y,
    # similar to the table in the paper
  }
  conditionalTable <- matrix(
    nrow = length(data[, 1]),
    ncol = length(data[1, ])
  )
  if (var_index == 2) {
    for (i in 1:length(data[, 1])) {
      row_sum <- sum(data[i, ])
      for (j in 1:length(data[1, ])) {
        conditionalTable[i, j] <- data[i, j] / row_sum
      }
    }
  } else {
    for (j in 1:length(data[1, ])) {
      col_sum <- sum(data[, j])
      for (i in 1:length(data[, 1])) {
        conditionalTable[i, j] <- data[i, j] / col_sum
      }
    }
  }

  conditionalTable <- as.data.frame(conditionalTable)
  # colnames(conditionalTable) <- levels(data)
  return(conditionalTable)
}

# getting conditional pmf of var_index
getConditionalPMFwo <- function(var_index = 1, data = df) {
  grouping_cols <- c(1:ncol(data))[-var_index]
  cond <- data |>
    group_by_all() |>
    dplyr::count() |>
    ungroup() |>
    mutate(freq = n / sum(n)) |>
    group_by(across(grouping_cols)) |>
    mutate(freq_cond = n / sum(n)) |>
    ungroup()
  return(cond)
}

# get the marginal cdf
getCDF <- function(var_index = 1, data = df) {
  append(cumsum(getPMF_2(var_index = var_index, data = data)),
    value = 0, after = 0
  )
```

```r
}

# function to get the joint cdf of u,v pair
getJointCDF <- function(u = 0.3, v = 0.3, data = df) {
  joint_pmf <- getJointPMF(data = data)
  Xcdf <- getCDF(var_index = 1, data = data)
  Ycdf <- getCDF(var_index = 2, data = data)

  X <- findInterval(u, Xcdf, left.open = TRUE, rightmost.closed = TRUE)
  Y <- findInterval(v, Ycdf, left.open = TRUE, rightmost.closed = TRUE)

  sum <- 0
  for (i in 1:X) {
    for (j in 1:Y) {
      sum <- sum + joint_pmf[i, j]
    }
  }
  return(sum)
}

# generalized function to get the joint cdf of u,v pair
getJointCDF_2 <- function(idx_vec = c(1, 2, 3), data = df) {
  grid <- expand.grid(lapply(data, levels))
  colnames(grid) <- colnames(data)
  pmf <- getJointPMF_freq(data = data)
  combined <- suppressMessages(left_join(grid, dplyr::select(pmf, -n),
    by = colnames(grid)
  ))
  combined[is.na(combined)] <- 0
  result <- combined |>
    mutate(across(.cols = everything(), as.numeric))

  for (i in 1:length(idx_vec)) {
    result <- result[result[, i] <= idx_vec[i], ]
  }
  return(sum(result$freq))
}

# get density of the checkerboard copula
getCCDensity <- function(pmf_table = pmf_table) {
  density_table <- matrix(
    nrow = length(pmf_table[, 1]),
    ncol = length(pmf_table[1, ])
  )

  for (i in 1:length(pmf_table[, 1])) {
    for (j in 1:length(pmf_table[1, ])) {
      density_table[i, j] <- pmf_table[i, j] / ((sum(pmf_table[i, ])
      * sum(pmf_table[, j])))
    }
  }
  return(density_table)
}

# generalized version to get density of the checkerboard copula
getCCDensity_2 <- function(data = df) {
  # browser()
  data_int <- unique(data) |>
    mutate(across(.cols = everything(), as.numeric))
  pmfs <- lapply(1:ncol(data), getPMF, data = data)
  cc_prod <- data.frame()
  for (i in 1:nrow(data_int)) {
    c <- c()
    for (j in 1:length(pmfs)) {
      c <- append(c, pmfs[[j]][data_int[i, j]])
    }
    cc_prod <- rbind(cc_prod, c)
  }
  cc_prod <- cc_prod |>
    mutate(prod = Reduce(`*`, cc_prod))
  return_data <- getJointPMF_freq(data = data) |>
    mutate(cc_value = freq / cc_prod$prod)
  return(return_data)
}

# get density for a given u and v from density table above
getDensity <- function(u = 0.5, v = 1 / 8, pmf_table = pmf_table) {
  Xcdf <- getCDF(var_index = 1)
  Ycdf <- getCDF(var_index = 2)
  X <- findInterval(u, Xcdf, left.open = TRUE, rightmost.closed = TRUE)
  Y <- findInterval(v, Ycdf, left.open = TRUE, rightmost.closed = TRUE)
  getCCDensity(pmf_table = pmf_table)[X, Y]
}
```

```r
# general version
getDensity_2 <- function(vec = c(0.25, 0.3), data = df) {
  # browser()
  cdfs <- lapply(1:ncol(data), getCDF, data = data)
  values <- mapply(FUN = function(x, y) {
    findInterval(y, x, left.open = TRUE, rightmost.closed = TRUE)
  }, x = cdfs, y = vec)
  density <- getCCDensity_2(data = data) |>
    mutate(across(.cols = everything(), as.numeric))
  for (i in 1:length(vec)) {
    density <- density[density[, i] == values[i], ]
  }
  return(density)
}

### Checkerboard copula score
# ridits, using the zoo package to find mean of
# adjacent values in the cdf
getScores <- function(data = Xcdf) {
  rollmean(data, 2)
}

# calculate CC
getCheckerboardCopula <- function(u = 0.3, v = 0.3, data = df) {
  Xcdf <- getCDF(var_index = 1, data = data)
  Ycdf <- getCDF(var_index = 2, data = data)
  if ("data.frame" %in% class(data)) {
    joint_pmf <- getJointPMF(data = data)
  } else {
    joint_pmf <- data
  }
  # get greatest and least elements such that inf_u <= u <= sup_u
  inf_u <- max(Xcdf[Xcdf <= u])
  sup_u <- min(Xcdf[Xcdf >= u])

  # get greatest and least elements such that inf_v <= v <= sup_v
  inf_v <- max(Ycdf[Ycdf <= v])
  sup_v <- min(Ycdf[Ycdf >= v])

  # get lambda and mu
  lambda <- ifelse(inf_u < sup_u, (u - inf_u) / (sup_u - inf_u), 1)
  mu <- ifelse(inf_v < sup_v, (v - inf_v) / (sup_v - inf_v), 1)
  # find category of X and Y that matches u and v
  inf_u_index <- findInterval(inf_u, Xcdf,
    left.open = TRUE,
    rightmost.closed = TRUE
  )
  sup_u_index <- findInterval(sup_u, Xcdf,
    left.open = TRUE,
    rightmost.closed = TRUE
  )

  inf_v_index <- findInterval(inf_v, Ycdf,
    left.open = TRUE,
    rightmost.closed = TRUE
  )
  sup_v_index <- findInterval(sup_v, Ycdf,
    left.open = TRUE,
    rightmost.closed = TRUE
  )

  # calculate the Checkerboard copula distribution function
  CC_df <- (1 - lambda) * (1 - mu) * getJointCDF(
    u = inf_u, v = inf_v,
    data = data
  ) +
    (1 - lambda) * (mu) *
      getJointCDF(u = inf_u, v = sup_v, data = data) +
    (lambda) * (1 - mu) *
      getJointCDF(u = sup_u, v = inf_v, data = data) +
    (lambda) * (mu) *
      getJointCDF(u = sup_u, v = sup_v, data = data)
  # find joint cdf
  return(CC_df)
}

# general
getCheckerboardCopula_2 <- function(idx_vec = c(0.4, 0.4),
                                    data = df) {
  # browser()
  cdfs <- lapply(1:ncol(data), getCDF, data = data)
  inf_values <- mapply(FUN = function(cdf, vec) {
    max(cdf[cdf <= vec])
  }, cdf = cdfs, vec = idx_vec)
  sup_values <- mapply(FUN = function(cdf, vec) {
    min(cdf[cdf >= vec])
  }, cdf = cdfs, vec = idx_vec)
  lambdas <- mapply(FUN = function(sup, inf, vec) {
    ifelse(inf < sup, (vec - inf) / (sup - inf), 1)
  }, sup = sup_values, inf = inf_values, vec = c)
  # find category of X and Y that matches u and v
  sup_idx <- mapply(FUN = function(cdf, vec) {
    findInterval(vec, cdf, left.open = TRUE, rightmost.closed = TRUE)
  }, cdf = cdfs, vec = sup_values)
  inf_idx <- mapply(FUN = function(cdf, vec) {
    findInterval(vec, cdf, left.open = TRUE, rightmost.closed = TRUE)
  }, cdf = cdfs, vec = inf_values)
  subsets <- sets::set_power(1:ncol(data))
  sum <- 0
  for (i in subsets) {
    prod <- 1
    idx_vector <- c()
    for (j in c(1:ncol(data))) {
      if (j %in% i) {
        prod <- prod * lambdas[j]
        idx_vector <- append(idx_vector, sup_idx[j])
      } else {
        prod <- prod * (1 - lambdas[j])
        idx_vector <- append(idx_vector, inf_idx[j])
      }
    }
    sum <- sum + (prod *
      getJointCDF_2(idx_vec = idx_vector, data = data))
  }
  return(sum)
}


# get variance of CC
getVariance <- function(var_index = 1, data = df) {
  # browser()
  pmf <- getPMF_2(var_index = var_index, data = data)
  cdf <- getCDF(var_index = var_index, data = data)
  var <- 0
  for (i in 1:nrow(unique((data[, var_index])))) {
```

```r
    var <- var + cdf[i] * cdf[i + 1] * pmf[i] / 4
  }
  return(var)
}

# var_index = variable of interest regressed on the other variable
# if var_index = 1, then we are regressing X on Y,
# else we regress Y on X
getRegression <- function(var_index = 1, data = df) {
  length_var1 <- length(levels(factor(data[, var_index])))
  length_var2 <- length(levels(factor(data[, -var_index])))
  Xcdf <- getCDF(var_index = 1, data = data)
  Xscores <- getScores(data = Xcdf)
  Ycdf <- getCDF(var_index = 2, data = data)
  Yscores <- getScores(data = Ycdf)


  # getting conditional pmfs
  Y_X <- getConditionalPMF(var_index = 2, data = data)
  X_Y <- getConditionalPMF(var_index = 1, data = data)

  # Y on X
  if (var_index == 2) {
    regression_table <- tibble(
      index = levels(cut(c(0, 1), breaks = Xcdf)), regression = NA
    )
    for (i in 1:length_var2) {
      sum <- 0
      for (j in 1:length_var1) {
        sum <- sum + Y_X[i, j] * Yscores[j]
      }
      regression_table[i, 2] <- sum
    }
  }
  # X on Y
  else {
    regression_table <- tibble(
      index = levels(cut(c(0, 1), breaks = Ycdf)), regression = NA
    )
    for (j in 1:length_var2) {
      sum <- 0
      for (i in 1:length_var1) {
        sum <- sum + X_Y[i, j] * Xscores[i]
      }
      regression_table[j, 2] <- sum
    }
  }

  return(regression_table)
}

# general version
getRegression_2 <- function(var_index = 1, data = df) {
  # browser()
  cdf <- getCDF(var_index = var_index, data = data)
  scores <- getScores(data = cdf)
  # getting conditional pmfs
  conditional_pmf <- getConditionalPMFwo(
    var_index = var_index,
    data = data
  ) |>
    mutate(across(.cols = everything(), as.numeric))

  grouping_cols <- c(1:ncol(data))[-var_index]
  scores_vec <- lapply(conditional_pmf[, var_index],
    FUN = function(x) scores[x]
  )
  combined <- cbind(conditional_pmf, scores_vec)
  colnames(combined)[ncol(combined)] <- "score"
  regression_data <- combined |>
    mutate(regression = freq_cond * score) |>
    group_by(across(grouping_cols)) |>
    summarize(reg = sum(regression))

  # regression_data <- cbind(regression_data,
  # levels(cut(c(0, 1), breaks = cdf)))

  return(regression_data)
}

# predicting from regression
# explanatory index is the category level of X
# so explanatory_index = 3 indicates that when X = 3,
# it predicts what Y is
```

```
getPredictionY <- function(explanatory_index = 3, data = df) {
  regression <- getRegression(var_index = 2, data = data)
  Ycdf <- getCDF(var_index = 2, data = data)
  x <- findInterval(regression[explanatory_index, 2], Ycdf)
  return(x)
}

getPredictionX <- function(explanatory_index = 3, data = df) {
  regression <- getRegression(var_index = 1, data = data)
  Xcdf <- getCDF(var_index = 1, data = data)
  x <- findInterval(regression[explanatory_index, 2], Xcdf)
  return(x)
}

# general version
getPrediction <- function(var_index = 1, idx_vec = c(1, 2),
                          data = df) {
  regression <- getRegression_2(var_index = var_index, data = data)
  cdf <- getCDF(var_index = var_index, data = data)
  # for each element in the idx_vec
  for (i in 1:length(idx_vec)) {
    # print(i)
    regression <- regression[regression[, i] == idx_vec[i], ]
  }
  pred_val <- findInterval(regression$reg, cdf)
  if (!length(pred_val)) {
    return("no prediction")
  }
  return(pred_val)
}

# function to get beccr, takes a dataframe or a table
getBECCR <- function(data = df, var_index = 2) {
  # browser()
  if ("data.frame" %in% class(data)) {
    data_pmf <- getJointPMF(data = data)
  }
  # data <- as.table(data)
  conditional_table <- getConditionalPMF(
    var_index =
      var_index, data = data
  )
  length_var1 <- length(conditional_table[1, ])
  length_var2 <- length(conditional_table[, 1])

  scores <- getScores(data = getCDF(
    var_index = var_index,
    data = data
  ))

  if (var_index == 2) {
    pmf <- getPMF(var_index = 1, data = data)

    sum_i <- 0
    for (i in 1:length_var2) {
      sum_j <- 0
      for (j in 1:length_var1) {
        sum_j <- sum_j + (conditional_table[i, j] * scores[j])
      }
      sum_i <- sum_i + ((sum_j - (1 / 2))^2 * pmf[i])
    }
    rho <- 12 * sum_i
  } else {
    pmf <- getPMF(var_index = 2, data = data)

    sum_j <- 0
    for (j in 1:length_var2) {
      sum_i <- 0
      for (i in 1:length_var1) {
        sum_i <- sum_i + (conditional_table[i, j] * scores[i])
      }
      sum_j <- sum_j + ((sum_i - (1 / 2))^2 * pmf[j])
    }
    rho <- 12 * sum_j
  }
  return(rho)
}

# generalized version
getBECCR_2 <- function(var_index = 2, data = df) {
  # browser()
  regression_table <- getRegression_2(
    var_index =
      var_index, data = data
  )
  pmfwo <- getPMFwo(var_index = var_index, data = data) |>
    mutate(across(everything(), as.numeric))
  regression_table <- left_join(regression_table,
    pmfwo,
    by = colnames(data[-var_index])
  )
  beccr <- regression_table |>
    mutate(temp = (reg - (1 / 2))^2 * freq)

  return(12 * sum(beccr$temp))
}
```

Code to generate a toy example used throughout the chapter as shown in Table 4.1.

```
# toy data from key paper
X <- factor(c("Very Low", "Low", "Medium", "High", "Very High"),
  levels = c("Very Low", "Low", "Medium", "High", "Very High")
)
Y <- factor(c("Severe", "Moderate", "Mild", "Moderate", "Severe"),
  levels = c("Mild", "Moderate", "Severe")
)
toy_data <- data.frame(
  row.names = 1:8,
  X = rep(X, times = c(2, 1, 2, 1, 2)),
  Y = rep(Y, times = c(2, 1, 2, 1, 2))
)
toy_pmf <- getJointPMF(data = toy_data)
```

Code to generate a grid of values to plot the surface plots of the subcopula, checkerboard copula as well as their dependencies.

```
source("scripts.R")
pmf_table <- getJointPMF(data = df)
CC <- getCCDensity(pmf_table = pmf_table)
n.grid <- 26
u <- seq(0, 1, length.out = n.grid)
```

```
grid <- expand.grid("u1" = u, "u2" = u)
grid_CsCDF <- grid %>%
  rowwise() %>%
  mutate(subcopula = getJointCDF(u = u1, v = u2, data = df))

grid_CsPMF <- grid %>%
  rowwise() %>%
  mutate(subcopula = findJointPMF(u = u1, v = u2, data = df))

grid_ccCDF <- grid %>%
  rowwise() %>%
  mutate(Checkerboard = getCheckerboardCopula(u = u1,
                                              v = u2, data = df))

grid_ccPMF <- grid %>%
  rowwise() %>%
  mutate(density = getDensity(u = u1, v = u2, pmf_table = pmf_table))

saveRDS(grid_CsCDF, "data/grid_CsCDF.rds")
saveRDS(grid_CsPMF, "data/grid_CsPMF.rds")
saveRDS(grid_ccCDF, "data/grid_ccCDF.rds")
saveRDS(grid_ccPMF, "data/grid_ccPMF.rds")
```

Code for Figure 4.2.

```
grid_CsCDF <- readRDS("data/grid_CsCDF.rds")
grid_CsPMF <- readRDS("data/grid_CsPMF.rds")
grid_ccCDF <- readRDS("data/grid_ccCDF.rds")
grid_ccPMF <- readRDS("data/grid_ccPMF.rds")
# subcopula CDF
a <- copula::wireframe2(grid_CsCDF,
  xlab = "u",
  ylab = "v",
  zlab = list("Subcopula", rot = 90),
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights =
      list(
        top.padding = 0,
        bottom.padding = -4
      ),
    layout.widths =
      list(
        left.padding = 0,
        right.padding = -4
      )
  )
)

# subcopula PMF
b <- copula::wireframe2(grid_CsPMF,
  xlab = "u",
  ylab = "v",
  zlab = list("Density", rot = 90),
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights =
      list(
        top.padding = 0,
        bottom.padding = -4
      ),
    layout.widths =
      list(
        left.padding = -20,
        right.padding = 0
      )
  )
)
# CC CDF
c <- wireframe2(grid_ccCDF,
  xlab = "u",
  ylab = "v",
  zlab = list("Chcckerboard", rot = 90),
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights =
      list(
        top.padding = -4,
        bottom.padding = 0
      ),
    layout.widths =
      list(
        left.padding = 0,
        right.padding = -4
      )
  )
)
# c+
d <- wireframe2(grid_ccPMF,
  xlab = "u",
  ylab = "v",
  zlab = list("Density", rot = 90),
  par.settings = list(
    axis.text = list(cex = .5),
    layout.heights =
      list(
        top.padding = -4,
        bottom.padding = 0
      ),
    layout.widths =
      list(
        left.padding = -20,
        right.padding = 0
      )
  )
)
gridExtra::grid.arrange(a, b,
  c, d,
  nrow = 2, ncol = 2
)
```

Code for Example 9.

```
getScores(data = getCDF(var_index = 1, data = toy_data))
getScores(data = getCDF(var_index = 2, data = toy_data))
getVariance(var_index = 1, data = toy_data)
getVariance(var_index = 2, data = toy_data)
```

Code for Table 4.2.

```
getConditionalPMF(var_index = 2, data = toy_data)
getRegression(var_index = 2, data = toy_data)
getRegression(var_index = 1, data = toy_data)
getConditionalPMF(var_index = 1, data = toy_data)
```

Code to obtain the CCRAM and variance for Example 12.

```
getBECCR(data = toy_data, var_index = 2)
12 * getVariance(var_index = 2, data = toy_data)
getBECCR(data = toy_data, var_index = 2) /
  (12 * getVariance(var_index = 2, data = toy_data))
```

A.4 Code for Chapter 5

In this section, we include the code to generate the simulations as well as graph the boxplots. For the actual boxplots, see Appendix B.

```
############# Preliminary Functions #############
# sample from discrete uniform
runifdisc <- function(n, min = 0, max = 1) {
  sample(min:max, n,
    replace = T
  )
}

# stretches a frequency table so that each row is an observation
countsToCases <- function(x, countcol = "Freq") {
  idx <- rep.int(seq_len(nrow(x)), x[[countcol]])
  x[[countcol]] <- NULL
  x[idx, ]
}
# get logit / invlogit
logit <- function(p) {
  return(log(p / (1 - p)))
}
invlogit <- function(p) {
  return(1 / (1 + exp(-p)))
```

```r
}
invlogit_2 <- function(p) {
  return(exp(p) / (1 + exp(p)))
}

############# Generate Data and Simulations #############
# generate data based on specified CLM model
generateData <- function(Icat = 3, Jcat = 3, alpha = c(-.69, 0.69),
                         beta = 0, size = 200) {
  # create true joint pmf by first getting conditional probabilities
  truematrix <- matrix(NA, nrow = Icat, ncol = Jcat)
  for (imat in 1:Icat) {
    for (jmat in 1:(Jcat - 1)) {
      truematrix[imat, jmat] <- invlogit(alpha[jmat] - beta * imat)
    }
  }
  truematrix[, Jcat] <- 1 # add ones to the end since Jcat-1
  px <- rep(1 / Icat, Icat) # true uniform margins for x
  truemat <- cbind(truematrix[, 1], truematrix[, 2:(Jcat)] -
    truematrix[, 1:(Jcat - 1)]) * px

  # simulate using rmultinom
  simtabmat <- matrix(rmultinom(1, size = size, prob = truemat),
    nrow = Icat, ncol = Jcat, byrow = F
  )
  simtab <- as.table(simtabmat) # convert to table
  rownames(simtab) <- 1:Icat
  colnames(simtab) <- 1:Jcat
  simtab <- as.data.frame(simtab) # convert to data frame
  names(simtab) <- c("x", "y", "Freq") # Changing names of columns
  simtab$x <- as.numeric(simtab$x) # Convert X to numeric
  # Convert X to ordered levels
  simtab$y <- ordered(simtab$y, levels = 1:Jcat)
  x <- countsToCases(simtab)$x
  y <- countsToCases(simtab)$y
  casedata <- data.frame(x = as.numeric(x), y = y)
  casedata$y <- as.factor(casedata$y)
  return(casedata)
}

# function so that I can simulate for
# no association or linear association
```

```
no_linear_simulation <- function(Icat = 3, Jcat = 3,
                                 alpha = c(-0.69, 0.69),
                                 beta = 0,
                                 sample_size_vec = c(500, 1000, 2000)) {

  # browser()
  return_data <- data.frame(
    size = NA, model_free = NA, model = NA,
    model_poor = NA, count = NA
  )
  # generate the data for each sample size
  for (size in sample_size_vec) {
    model_check <- FALSE
    count <- 0
    while (!model_check) {
      data <- generateData(
        Icat = Icat, Jcat = Jcat, alpha = alpha,
        beta = beta, size = size
      )
      # if beta = 0, find no association else find linear association
      # for the poor model, choose linear
      # assoc if there none and vice versa
      # then perform goodness of fit test to see if model is good
      if (beta == 0) {
        logit.m <- VGAM::vglm(
          formula = y ~ 1, data = data,
          family = cumulative(
            link = "logitlink",
            parallel = TRUE,
            reverse = FALSE
          )
        )
        logit.poor <- VGAM::vglm(
          formula = y ~ x, data = data,
          family = cumulative(
            link = "logitlink",
            parallel = TRUE,
            reverse = FALSE
          )
        )
      } else {
        logit.m <- VGAM::vglm(
          formula = y ~ x, data = data,
          family = cumulative(
            link = "logitlink",
            parallel = TRUE,
            reverse = FALSE
          )
        )
        logit.poor <- VGAM::vglm(
          formula = y ~ 1, data = data,
          family = cumulative(
            link = "logitlink",
            parallel = TRUE,
            reverse = FALSE
          )
        )
      }
      cond_tab <- cbind(invlogit(predict(logit.m,
        newdata = data.frame(x = 1:Icat)
      )), 1)
      cond_tab_poor <- cbind(invlogit(predict(logit.poor,
        newdata = data.frame(x = 1:Icat)
      )), 1)

      cond_pmf <- cbind(cond_tab[, 1], cond_tab[, 2:Jcat] -
        cond_tab[, 1:Jcat - 1])
      cond_pmf_poor <- cbind(
        cond_tab_poor[, 1],
        cond_tab_poor[, 2:Jcat] -
          cond_tab_poor[, 1:Jcat - 1]
      )

      # check if model is a good fit and check if
      # poor model is a good fit too
      # this is pearsons GoF calculation
      data_table <- table(data$x, data$y)
      sum <- 0
      sum_poor <- 0
      for (i in 1:Icat) {
        n_row <- sum(data_table[i, ])
        for (j in 1:Jcat) {
          mu_hat <- n_row * cond_pmf[i, j]
          mu_hat_poor <- n_row * cond_pmf_poor[i, j]
```

```
          sum <- sum + ((data_table[i, j] - mu_hat)^2 / mu_hat)
          sum_poor <- sum_poor + ((data_table[i, j] -
            mu_hat_poor)^2 / mu_hat_poor)
        }
      }
      if (beta == 0) {
        model_check <- TRUE
      } else if (1 - pchisq(sum, df = Icat *
        (Jcat - 1) - Jcat) >= 0.05 &
        1 - pchisq(sum_poor, df = Icat *
          (Jcat - 1) - (Jcat - 1)) < 0.05) {
        model_check <- TRUE
      }
      count <- count + 1
    }
    px <- getPMF(var_index = 1, data = data)
    model_tab <- cbind(cond_tab[, 1], cond_tab[, 2:(Jcat)] -
      cond_tab[, 1:(Jcat - 1)]) * px
    model_tab_poor <- cbind(
      cond_tab_poor[, 1],
      cond_tab_poor[, 2:(Jcat)] -
        cond_tab_poor[, 1:(Jcat - 1)]
    ) * px

    model_free_BECCR <- getBECCR_2(var_index = 2, data = data)
    model_BECCR <- getBECCR(data = model_tab)
    model_poor_BECCR <- getBECCR(data = model_tab_poor)
    return_data <- rbind(return_data, c(
      size, model_free_BECCR,
      model_BECCR, model_poor_BECCR, count
    ))
  }
  return(return_data[-1, ] |> remove_rownames() |>
    mutate(size = as.factor(size)))
}

# generate data based on nonmonotone model
# structure is the same as the other data
# generating functions except with a different model
generateData_nonmonotone <- function(Icat = 3, Jcat = 3,
                                     alpha = c(-10.92, -8.91),
                                     beta1 = -12, beta2 = 3,
                                     size = 200) {
  truematrix <- matrix(NA, nrow = Icat, ncol = Jcat)
  for (imat in 1:Icat) {
    for (jmat in 1:(Jcat - 1)) {
      truematrix[imat, jmat] <- invlogit(alpha[jmat] -
        beta1 * imat - beta2 * imat^2)
    }
  }
  truematrix[, Jcat] <- 1
  px <- rep(1 / Icat, Icat)
  truemat <- cbind(truematrix[, 1], truematrix[, 2:(Jcat)] -
    truematrix[, 1:(Jcat - 1)]) * px

  simtabmat <- matrix(rmultinom(1, size = size, prob = truemat),
    nrow = Icat, ncol = Jcat, byrow = F
  )
  simtab <- as.table(simtabmat)
  rownames(simtab) <- 1:Icat
  colnames(simtab) <- 1:Jcat
  simtab <- as.data.frame(simtab) # convert to data frame
  names(simtab) <- c("x", "y", "Freq") # Changing names of columns
  simtab$x <- as.numeric(simtab$x) # Convert X to numeric
  # Convert X to ordered levels
  simtab$y <- ordered(simtab$y, levels = 1:Jcat)
  x <- countsToCases(simtab)$x
  y <- countsToCases(simtab)$y
  casedata <- data.frame(x = x, y = y)
  return(casedata)
}

# function so that I can simulate for nonmonotone association
nonmonotone_simulation <- function(Icat = 3, Jcat = 3,
                                   alpha = c(-10.92, -8.91),
                                   beta1 = -12, beta2 = 3,
                                   sample_size_vec =
                                     c(500, 1000, 2000)) {
  return_data <- data.frame(
    size = NA, model_free = NA, model = NA,
    model_linear = NA, model_no = NA,
    count = NA
  )
  for (size in sample_size_vec) {
```

```r
    model_check <- FALSE
    count <- 0
    while (model_check == FALSE) {
      data <- generateData_nonmonotone(
        Icat = Icat, Jcat = Jcat,
        alpha = alpha, beta1 = beta1,
        beta2 = beta2, size = size
      )
      logit.m <- VGAM::vglm(
        formula = y ~ x + I(x^2), data = data,
        family = cumulative(
          link = "logitlink",
          parallel = TRUE,
          reverse = FALSE
        )
      )
      logit.no <- VGAM::vglm(
        formula = y ~ 1, data = data,
        family = cumulative(
          link = "logitlink",
          parallel = TRUE,
          reverse = FALSE
        )
      )
      logit.linear <- VGAM::vglm(
        formula = y ~ x, data = data,
        family = cumulative(
          link = "logitlink",
          parallel = TRUE,
          reverse = FALSE
        )
      )

      cond_tab <- cbind(invlogit(predict(logit.m,
        newdata =
          data.frame(x = 1:Icat)
      )), 1)
      cond_tab_no <- cbind(invlogit(predict(logit.no,
        newdata =
          data.frame(x = 1:Icat)
      )), 1)
      cond_tab_linear <- cbind(invlogit(predict(logit.linear,
        newdata =
          data.frame(x = 1:Icat)
      )), 1)

      cond_pmf <- cbind(cond_tab[, 1], cond_tab[, 2:Jcat] -
        cond_tab[, 1:Jcat - 1])
      cond_pmf_no <- cbind(cond_tab_no[, 1], cond_tab_no[, 2:Jcat] -
        cond_tab_no[, 1:Jcat - 1])
      cond_pmf_linear <- cbind(
        cond_tab_linear[, 1],
        cond_tab_linear[, 2:Jcat] - cond_tab_linear[, 1:Jcat - 1]
      )

      data_table <- table(data$x, data$y)
      sum <- 0
      sum_no <- 0
      sum_linear <- 0
      for (i in 1:Icat) {
        n_row <- sum(data_table[i, ])
        for (j in 1:Jcat) {
          mu_hat <- n_row * cond_pmf[i, j]
          mu_hat_no <- n_row * cond_pmf_no[i, j]
          mu_hat_linear <- n_row * cond_pmf_linear[i, j]

          sum <- sum + ((data_table[i, j] -
            mu_hat)^2 / mu_hat)
          sum_no <- sum_no + ((data_table[i, j] -
            mu_hat_no)^2 / mu_hat_no)
          sum_linear <- sum_linear +
            ((data_table[i, j] - mu_hat_linear)^2 / mu_hat_linear)
        }
      }

      if (1 - pchisq(sum, df = Icat * (Jcat - 1) -
        (Jcat - 1) - 2) >= 0.05 &
        1 - pchisq(sum_no, df = Icat * (Jcat - 1) -
          Jcat) < 0.05 &
        1 - pchisq(sum_linear, df = Icat * (Jcat - 1) -
          (Jcat - 1)) < 0.05) {
        model_check <- TRUE
      }
      count <- count + 1
```

```
    }

    px <- getPMF(var_index = 1, data = data)
    model_tab <- cbind(cond_tab[, 1], cond_tab[, 2:(Jcat)] -
      cond_tab[, 1:(Jcat - 1)]) * px
    model_tab_linear <- cbind(
      cond_tab_linear[, 1],
      cond_tab_linear[, 2:(Jcat)] -
        cond_tab_linear[, 1:(Jcat - 1)]
    ) * px
    model_tab_no <- cbind(
      cond_tab_no[, 1],
      cond_tab_no[, 2:(Jcat)] - cond_tab_no[, 1:(Jcat - 1)]
    ) * px

    model_free_BECCR <- getBECCR(var_index = 2, data = data)
    model_BECCR <- getBECCR(data = model_tab)
    model_BECCR_linear <- getBECCR(data = model_tab_linear)
    model_BECCR_no <- getBECCR(data = model_tab_no)

    return_data <- rbind(return_data, c(
      size, model_free_BECCR,
      model_BECCR, model_BECCR_linear,
      model_BECCR_no, count
    ))
  }

  return(return_data[-1, ] |> remove_rownames() |>
    mutate(size = as.factor(size)))
}

# generate data based on nominal model
generateData_nominal <- function(Icat = 3, Jcat = 3,
                                 alpha = c(-.69, 0.69),
                                 tau = c(0, 0, 0), size = 200) {
  truematrix <- matrix(NA, nrow = Icat, ncol = Jcat)
  for (imat in 1:Icat) {
    for (jmat in 1:(Jcat - 1)) {
      truematrix[imat, jmat] <- invlogit(alpha[jmat] + tau[imat])
    }
  }
  truematrix[, Jcat] <- 1
```

```r
  px <- rep(1 / Icat, Icat) # uniform margins for x
  truemat <- cbind(truematrix[, 1], truematrix[, 2:(Jcat)] -
    truematrix[, 1:(Jcat - 1)]) * px

  simtabmat <- matrix(rmultinom(1, size = size, prob = truemat),
    nrow = Icat, ncol = Jcat, byrow = F
  )
  simtab <- as.table(simtabmat) # convert to table
  rownames(simtab) <- 1:Icat
  colnames(simtab) <- 1:Jcat
  simtab <- as.data.frame(simtab) # convert to data frame
  names(simtab) <- c("x", "y", "Freq") # Changing names of columns
  simtab$x <- as.numeric(simtab$x) # Convert X to numeric
  # Convert X to ordered levels
  simtab$y <- ordered(simtab$y, levels = 1:Jcat)
  x <- countsToCases(simtab)$x
  y <- countsToCases(simtab)$y
  casedata <- data.frame(x = as.factor(x), y = y)
  casedata$y <- as.factor(casedata$y)
  return(casedata)
}

nominal_simulation <- function(Icat = 3, Jcat = 3,
                               alpha = c(-.69, 0.69),
                               tau = c(0, 0, 0),
                               sample_size_vec = c(
                                 500,
                                 1000, 2000
                               )) {
  return_data <- data.frame(
    size = NA, model_free = NA, model = NA,
    model_poor = NA, count = NA
  )
  # generate the data for each sample size
  for (size in sample_size_vec) {
    model_check <- FALSE
    count <- 0
    while (!model_check) {
      data <- generateData_nominal(
        Icat = Icat,
        Jcat = Jcat,
        alpha = alpha,
        tau = tau, size = size
      )

      # Getting model

      logit.m <- VGAM::vglm(
        formula = y ~ relevel(x, ref = Icat), data = data,
        family = cumulative(
          link = "logitlink",
          parallel = TRUE,
          reverse = FALSE
        )
      )
      logit.poor <- VGAM::vglm(
        formula = y ~ 1, data = data,
        family = cumulative(
          link = "logitlink",
          parallel = TRUE,
          reverse = FALSE
        )
      )

      coef <- append(logit.m@coefficients, 0,
        after = (Jcat + Icat - 1)
      )
      coef_poor <- rep(0, Icat)

      cond_tab <- matrix(NA, nrow = Icat, ncol = Jcat)
      cond_tab_poor <- matrix(NA, nrow = Icat, ncol = Jcat)

      for (imat in 1:Icat) {
        for (jmat in 1:(Jcat - 1)) {
          cond_tab[imat, jmat] <- invlogit(logit.m@coefficients[jmat] +
            coef[Jcat - 1 + imat])
          cond_tab_poor[imat, jmat] <- invlogit(
            logit.poor@coefficients[jmat] +
              coef_poor[imat]
          )
        }
      }
      cond_tab[, Jcat] <- 1
      cond_tab_poor[, Jcat] <- 1
      cond_pmf <- cbind(cond_tab[, 1], cond_tab[, 2:Jcat] -
        cond_tab[, 1:Jcat - 1])
      cond_pmf_poor <- cbind(
        cond_tab_poor[, 1],
        cond_tab_poor[, 2:Jcat] -
          cond_tab_poor[, 1:Jcat - 1]
      )

      data_table <- table(data$x, data$y)
      sum <- 0
      sum_poor <- 0
      for (i in 1:Icat) {
        n_row <- sum(data_table[i, ])
        for (j in 1:Jcat) {
          mu_hat <- n_row * cond_pmf[i, j]
          mu_hat_poor <- n_row * cond_pmf_poor[i, j]
          sum <- sum + ((data_table[i, j] -
            mu_hat)^2 / mu_hat)
          sum_poor <- sum_poor + ((data_table[i, j] -
            mu_hat_poor)^2 / mu_hat_poor)
        }
      }

      if (1 - pchisq(sum, df = Icat * (Jcat - 1) - (Jcat - 1) -
        (Icat - 1)) >= 0.05 &
        1 - pchisq(sum_poor, df = Icat * (Jcat - 1) -
          (Jcat - 1)) < 0.05) {
        model_check <- TRUE
      }
      count <- count + 1
    }

    px <- getPMF(var_index = 1, data = data)
    model_tab <- cbind(cond_tab[, 1], cond_tab[, 2:(Jcat)] -
      cond_tab[, 1:(Jcat - 1)]) * px
    model_poor_tab <- cbind(
      cond_tab_poor[, 1],
      cond_tab_poor[, 2:(Jcat)] -
        cond_tab_poor[, 1:(Jcat - 1)]
    ) * px
```

```
    model_free_BECCR <- getBECCR(var_index = 2, data = data)
    model_BECCR <- getBECCR(data = model_tab)
    model_poor_BECCR <- getBECCR(data = model_poor_tab)
    return_data <- rbind(return_data, c(
      size, model_free_BECCR,
      model_BECCR, model_poor_BECCR, count
    ))
  }

  return(return_data[-1, ] |> remove_rownames() |>
    mutate(size = as.factor(size)))
}
```

A.4.1 Code to perform the simulations

Simulation code for no association.

```
set.seed(713)
no_assoc_33 <- replicate(1000,
  no_linear_simulation(
    Icat = 3, Jcat = 3,
    alpha = c(-0.69, 0.69), beta = 0,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

no_assoc_35 <- replicate(1000,
  no_linear_simulation(
    Icat = 3, Jcat = 5,
    alpha = c(
      -1.39, -0.41,
      0.41, 1.39
    ),
    beta = 0,
```

```r
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

no_assoc_53 <- replicate(1000,
  no_linear_simulation(
    Icat = 5, Jcat = 3,
    alpha = c(-0.69, 0.69),
    beta = 0,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

no_assoc_55 <- replicate(1000,
  no_linear_simulation(
    Icat = 5, Jcat = 5,
    alpha = c(
      -1.39, -0.41,
      0.41, 1.39
    ),
    beta = 0,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(no_assoc_33, "simulations/no_assoc_33.Rds")
saveRDS(no_assoc_35, "simulations/no_assoc_35.Rds")
saveRDS(no_assoc_53, "simulations/no_assoc_53.Rds")
saveRDS(no_assoc_55, "simulations/no_assoc_55.Rds")
```

Simulation code for weak association.

```
set.seed(713)
# weak linear associations

weak_assoc_33 <- replicate(1000,
  no_linear_simulation(
    Icat = 3, Jcat = 3,
    alpha = c(-0.21, 1.2),
    beta = 0.25,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

weak_assoc_35 <- replicate(1000,
  no_linear_simulation(
    Icat = 3, Jcat = 5,
    alpha = c(
      -0.9, 0.09,
      0.91, 1.9
    ),
    beta = 0.25,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )


weak_assoc_53 <- replicate(1000,
  no_linear_simulation(
    Icat = 5, Jcat = 3,
    alpha = c(0.04, 1.47),
    beta = 0.25,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

weak_assoc_55 <- replicate(1000,
  no_linear_simulation(
    Icat = 5, Jcat = 5,
    alpha = c(
      -0.67, 0.33,
      1.17, 2.17
    ),
    beta = 0.25,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(weak_assoc_33, "simulations/weak_assoc_33.Rds")
saveRDS(weak_assoc_35, "simulations/weak_assoc_35.Rds")
saveRDS(weak_assoc_53, "simulations/weak_assoc_53.Rds")
saveRDS(weak_assoc_55, "simulations/weak_assoc_55.Rds")
```

Simulation code for moderate association.

```
set.seed(713)
# moderate linear association
moderate_assoc_33 <- replicate(1000,
  no_linear_simulation(
    Icat = 3, Jcat = 3,
    alpha = c(0.93, 2.48),
    beta = 0.85,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

moderate_assoc_35 <- replicate(1000, no_linear_simulation(
  Icat = 3, Jcat = 5,
  alpha = c(
    0.17, 1.25,
    2.16, 3.23
  ),
  beta = 0.85,
  sample_size_vec = c(
    500, 1000,
    2000
  )
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )


moderate_assoc_53 <- replicate(1000, no_linear_simulation(
  Icat = 5, Jcat = 3,
  alpha = c(1.63, 3.47),
  beta = 0.85,
  sample_size_vec = c(
    500, 1000,
    2000
  )
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

moderate_assoc_55 <- replicate(1000, no_linear_simulation(
  Icat = 5, Jcat = 5,
  alpha = c(
    0.76, 2,
    3.1, 4.33
  ),
  beta = 0.85,
  sample_size_vec = c(
    500, 1000,
    2000
  )
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(moderate_assoc_33, "simulations/moderate_assoc_33.Rds")
saveRDS(moderate_assoc_35, "simulations/moderate_assoc_35.Rds")
saveRDS(moderate_assoc_53, "simulations/moderate_assoc_53.Rds")
saveRDS(moderate_assoc_55, "simulations/moderate_assoc_55.Rds")
```

Simulation code for strong association.

```
set.seed(713)
strong_assoc_33 <- replicate(1000,
  no_linear_simulation(
    Icat = 3,
    Jcat = 3,
    alpha = c(1.9, 3.7),
    beta = 1.4,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

strong_assoc_35 <- replicate(1000,
  no_linear_simulation(
    Icat = 3,
    Jcat = 5,
    alpha = c(
      1.05, 2.27,
      3.33, 4.55
    ),
    beta = 1.4,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

strong_assoc_53 <- replicate(1000,
  no_linear_simulation(
    Icat = 5,
    Jcat = 3,
    alpha = c(2.95, 5.45),
    beta = 1.4,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

strong_assoc_55 <- replicate(1000,
  no_linear_simulation(
    Icat = 5,
    Jcat = 5,
    alpha = c(
      1.84, 3.46,
      4.95, 6.55
    ),
    beta = 1.4,
    sample_size_vec = c(
      500, 1000,
      2000
    )
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(strong_assoc_33, "simulations/strong_assoc_33.Rds")
saveRDS(strong_assoc_35, "simulations/strong_assoc_35.Rds")
saveRDS(strong_assoc_53, "simulations/strong_assoc_53.Rds")
saveRDS(strong_assoc_55, "simulations/strong_assoc_55.Rds")
```

Simulation code for very strong association.

```
set.seed(713)
very_assoc_33 <- replicate(1000,
  no_linear_simulation(
    Icat = 3,
    Jcat = 3,
    alpha = c(2.9, 5.1),
    beta = 2, sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
```

```
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

very_assoc_35 <- replicate(1000,
  no_linear_simulation(
    Icat = 3, Jcat = 5,
    alpha = c(1.90, 3.34, 4.66, 6.10), beta = 2,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

very_assoc_53 <- replicate(1000,
  no_linear_simulation(
    Icat = 5, Jcat = 3,
    alpha = c(4.30, 7.69),
    beta = 2,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

very_assoc_55 <- replicate(1000,
  no_linear_simulation(
    Icat = 5, Jcat = 5,
    alpha = c(2.9, 5, 7, 9),
    beta = 2,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(very_assoc_33, "simulations/very_assoc_33.Rds")
saveRDS(very_assoc_35, "simulations/very_assoc_35.Rds")
saveRDS(very_assoc_53, "simulations/very_assoc_53.Rds")
saveRDS(very_assoc_55, "simulations/very_assoc_55.Rds")
```

Simulation code for nonmonotone nonlinear association.

```
set.seed(713)
nonmonotone_33 <- replicate(1000,
  nonmonotone_simulation(
    Icat = 3, Jcat = 3,
    alpha = c(-10.92, -8.91),
    beta1 = -12, beta2 = 3,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_linear, model_no),
    names_to = c("type")
  )


nonmonotone_35 <- replicate(1000,
  nonmonotone_simulation(
    Icat = 3, Jcat = 5,
    alpha = c(-11.98, -10.46, -9.28, -8.11),
    beta1 = -12, beta2 = 3,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_linear, model_no),
    names_to = c("type")
  )

nonmonotone_53 <- replicate(1000,
  nonmonotone_simulation(
    Icat = 5, Jcat = 3,
    # alpha = c(-25.49, -19.45),
    alpha = c(-24.52, -16.62),
    beta1 = -18, beta2 = 3,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_linear, model_no),
    names_to = c("type")
  )

nonmonotone_55 <- replicate(1000,
  nonmonotone_simulation(
    Icat = 5, Jcat = 5,
    alpha = c(-25.92, -23.91, -19.51, -15.01),
    # alpha = c(-26.82, -24.84, -22.54, -15.42),
    beta1 = -18, beta2 = 3,
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_linear, model_no),
    names_to = c("type")
  )

saveRDS(nonmonotone_33, "simulations/nonmonotone_33.Rds")
saveRDS(nonmonotone_35, "simulations/nonmonotone_35.Rds")
saveRDS(nonmonotone_53, "simulations/nonmonotone_53.Rds")
```

```r
saveRDS(nonmonotone_55, "simulations/nonmonotone_55.Rds")
```

Simulation code for no association for a nominal variable.

```r
set.seed(713)
no_nominal_33 <- replicate(1000,
  nominal_simulation(
    Icat = 3, Jcat = 3,
    alpha = c(-0.69, 0.69),
    tau = c(0, 0, 0),
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

no_nominal_35 <- replicate(1000,
  nominal_simulation(
    Icat = 3, Jcat = 5,
    alpha = c(-1.39, -0.41, 0.41, 1.39),
    tau = c(0, 0, 0),
    sample_size_vec = c(500, 1000, 2000)
  ),
  simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

no_nominal_53 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 3,
  alpha = c(-0.69, 0.69),
  tau = c(0, 0, 0, 0, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

no_nominal_55 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 5,
  alpha = c(-1.39, -0.41, 0.41, 1.39),
  tau = c(0, 0, 0, 0, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(no_nominal_33, "simulations/no_nominal_33.Rds")
saveRDS(no_nominal_35, "simulations/no_nominal_35.Rds")
saveRDS(no_nominal_53, "simulations/no_nominal_53.Rds")
saveRDS(no_nominal_55, "simulations/no_nominal_55.Rds")
```

Simulation code for weak association for a nominal variable.

```
set.seed(713)
# weak association
weak_nominal_33 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 3,
  alpha = c(-0.70, 0.70),
  tau = c(0.25, -0.25, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

weak_nominal_35 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 5,
  alpha = c(-1.40, -0.41, 0.41, 1.40),
  tau = c(0.25, -0.25, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

weak_nominal_53 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 3,
  alpha = c(-0.72, 0.72),
  tau = c(0.55, 0.25, -0.25, -0.55, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )


weak_nominal_55 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 5,
  alpha = c(-1.43, -0.42, 0.42, 1.43),
  tau = c(0.55, 0.25, -0.25, -0.55, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(weak_nominal_33, "simulations/weak_nominal_33.Rds")
saveRDS(weak_nominal_35, "simulations/weak_nominal_35.Rds")
saveRDS(weak_nominal_53, "simulations/weak_nominal_53.Rds")
saveRDS(weak_nominal_55, "simulations/weak_nominal_55.Rds")
```

Simulation code for moderate association for a nominal variable.

```
set.seed(713)
# moderate association
moderate_nominal_33 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 3,
  alpha = c(-0.77, 0.77),
  tau = c(0.85, -0.85, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

moderate_nominal_35 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 5,
  alpha = c(-1.53, -0.45, 0.45, 1.53),
  tau = c(0.85, -0.85, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )
```

```r
moderate_nominal_53 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 3,
  alpha = c(-0.82, 0.82),
  tau = c(1.1, 0.85, -0.85, -1.1, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

moderate_nominal_55 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 5,
  alpha = c(-1.61, -0.48, 0.48, 1.61),
  tau = c(1.1, 0.85, -0.85, -1.10, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(moderate_nominal_33, "simulations/moderate_nominal_33.Rds")
saveRDS(moderate_nominal_35, "simulations/moderate_nominal_35.Rds")
saveRDS(moderate_nominal_53, "simulations/moderate_nominal_53.Rds")
saveRDS(moderate_nominal_55, "simulations/moderate_nominal_55.Rds")
```

Simulation code for strong association for a nominal variable.

```r
set.seed(713)
strong_nominal_33 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 3,
  alpha = c(-0.91, 0.91),
  tau = c(1.4, -1.4, 0),
  sample_size_vec = c(500, 1000, 2000)
```

```
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

strong_nominal_35 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 5,
  alpha = c(-1.76, -0.53, 0.53, 1.76),
  tau = c(1.4, -1.4, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

strong_nominal_53 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 3,
  alpha = c(-1.02, 1.02),
  tau = c(1.7, 1.4, -1.4, -1.7, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

strong_nominal_55 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 5,
  alpha = c(-1.93, -0.61, 0.61, 1.93),
  tau = c(1.7, 1.4, -1.4, -1.7, 0),
  sample_size_vec = c(500, 1000, 2000)
```

```r
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(strong_nominal_33, "simulations/strong_nominal_33.Rds")
saveRDS(strong_nominal_35, "simulations/strong_nominal_35.Rds")
saveRDS(strong_nominal_53, "simulations/strong_nominal_53.Rds")
saveRDS(strong_nominal_55, "simulations/strong_nominal_55.Rds")
```

Simulation code for very strong association for a nominal variable.

```r
# very strong association
set.seed(713)
very_nominal_33 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 3,
  alpha = c(-1.11, 1.11),
  tau = c(2, -2, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

very_nominal_35 <- replicate(1000, nominal_simulation(
  Icat = 3, Jcat = 5,
  alpha = c(-2.10, -0.66, 0.66, 2.10),
  tau = c(2, -2, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

very_nominal_53 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 3,
  alpha = c(-1.31, 1.31),
  tau = c(2.3, 2, -2, -2.3, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

very_nominal_55 <- replicate(1000, nominal_simulation(
  Icat = 5, Jcat = 5,
  alpha = c(-2.37, -0.79, 0.79, 2.37),
  tau = c(2.3, 2, -2, -2.3, 0),
  sample_size_vec = c(500, 1000, 2000)
),
simplify = FALSE
) |>
  bind_rows() |>
  pivot_longer(
    cols = c(model_free, model, model_poor),
    names_to = c("type")
  )

saveRDS(very_nominal_33, "simulations/very_nominal_33.Rds")
saveRDS(very_nominal_35, "simulations/very_nominal_35.Rds")
saveRDS(very_nominal_53, "simulations/very_nominal_53.Rds")
saveRDS(very_nominal_55, "simulations/very_nominal_55.Rds")
```

A.4.2 Code to generate the boxplots

Code for no association boxplots.

```r
# New facet label names
model.labs <- c("Model (Good)", "Model Free", "Model (Poor)")
names(model.labs) <- c("model", "model_free", "model_poor")

no_assoc_33 <- readRDS("simulations/no_assoc_33.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
no_assoc_35 <- readRDS("simulations/no_assoc_35.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
no_assoc_53 <- readRDS("simulations/no_assoc_53.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
no_assoc_55 <- readRDS("simulations/no_assoc_55.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

no_assoc_33_p <- ggplot(no_assoc_33, aes(x = value, color = size)) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(0, 0.03)

no_assoc_35_p <- ggplot(no_assoc_35, aes(x = value, color = size)) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(0, 0.04)

no_assoc_53_p <- ggplot(no_assoc_53, aes(x = value, color = size)) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(0, 0.04)

no_assoc_55_p <- ggplot(no_assoc_55, aes(x = value, color = size)) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(0, 0.04)
```

Code for weak association boxplots.

```r
weak_assoc_33 <- readRDS("simulations/weak_assoc_33.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
weak_assoc_35 <- readRDS("simulations/weak_assoc_35.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
weak_assoc_53 <- readRDS("simulations/weak_assoc_53.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
weak_assoc_55 <- readRDS("simulations/weak_assoc_55.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

weak_assoc_33_p <- ggplot(
  weak_assoc_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .075))

weak_assoc_35_p <- ggplot(
  weak_assoc_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, 0.1))

weak_assoc_53_p <- ggplot(
  weak_assoc_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .15))

weak_assoc_55_p <- ggplot(
  weak_assoc_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .15))
```

Code for moderate association boxplots.

```
moderate_assoc_33 <- readRDS("simulations/moderate_assoc_33.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
moderate_assoc_35 <- readRDS("simulations/moderate_assoc_35.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
moderate_assoc_53 <- readRDS("simulations/moderate_assoc_53.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
moderate_assoc_55 <- readRDS("simulations/moderate_assoc_55.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

moderate_assoc_33_p <- ggplot(
  moderate_assoc_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .25))

moderate_assoc_35_p <- ggplot(
  moderate_assoc_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .25))

moderate_assoc_53_p <- ggplot(
  moderate_assoc_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .4))

moderate_assoc_55_p <- ggplot(
  moderate_assoc_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .5))
```

Code for strong association boxplots.

```
strong_assoc_33 <- readRDS("simulations/strong_assoc_33.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
strong_assoc_35 <- readRDS("simulations/strong_assoc_35.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
```

```
strong_assoc_53 <- readRDS("simulations/strong_assoc_53.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
strong_assoc_55 <- readRDS("simulations/strong_assoc_55.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

strong_assoc_33_p <- ggplot(
  strong_assoc_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .5))

strong_assoc_35_p <- ggplot(
  strong_assoc_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .5))

strong_assoc_53_p <- ggplot(
  strong_assoc_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .7))

strong_assoc_55_p <- ggplot(
  strong_assoc_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .7))
```

Code for very strong association boxplots.

```
very_assoc_33 <- readRDS("simulations/very_assoc_33.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
very_assoc_35 <- readRDS("simulations/very_assoc_35.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
very_assoc_53 <- readRDS("simulations/very_assoc_53.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
very_assoc_55 <- readRDS("simulations/very_assoc_55.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

very_assoc_33_p <- ggplot(
  very_assoc_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .6))

very_assoc_35_p <- ggplot(
  very_assoc_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .6))

very_assoc_53_p <- ggplot(
  very_assoc_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .8))

very_assoc_55_p <- ggplot(
  very_assoc_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .8))
```

Code for nonmonotone nonlinear association boxplots.

```
# New facet label names
nonmonotone.labs <- c(
  "Model (Good)", "Model Free",
  "Model (Linear)", "Model (No)"
)
names(nonmonotone.labs) <- c(
  "model", "model_free",
  "model_linear", "model_no"
)
nonmonotone_33 <- readRDS("simulations/nonmonotone_33.Rds") |>
  mutate(across(type, factor,
    levels = c(
      "model_free", "model",
      "model_linear", "model_no"
    )
  ))
nonmonotone_35 <- readRDS("simulations/nonmonotone_35.Rds") |>
  mutate(across(type, factor,
    levels = c(
      "model_free", "model",
      "model_linear", "model_no"
    )
  ))
nonmonotone_53 <- readRDS("simulations/nonmonotone_53.Rds") |>
  mutate(across(type, factor,
    levels = c(
      "model_free", "model",
      "model_linear", "model_no"
    )
  ))
nonmonotone_55 <- readRDS("simulations/nonmonotone_55.Rds") |>
  mutate(across(type, factor,
    levels = c(
      "model_free", "model",
      "model_linear", "model_no"
    )
  ))

nonmonotone_33_p <- ggplot(
  nonmonotone_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = nonmonotone.labs)) +
  xlim(c(0, .5))

nonmonotone_35_p <- ggplot(
  nonmonotone_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = nonmonotone.labs)) +
  xlim(c(0, .5))

nonmonotone_53_p <- ggplot(
  nonmonotone_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = nonmonotone.labs)) +
  xlim(c(0, .8))
```

```
nonmonotone_55_p <- ggplot(
  nonmonotone_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = nonmonotone.labs)) +
  xlim(c(0, 1))
```

Code for boxplots of no association for a nominal variable.

```
no_nominal_33 <- readRDS("simulations/no_nominal_33.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
no_nominal_35 <- readRDS("simulations/no_nominal_35.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
no_nominal_53 <- readRDS("simulations/no_nominal_53.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
no_nominal_55 <- readRDS("simulations/no_nominal_55.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

no_nominal_33_p <- ggplot(
  no_nominal_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .05))

no_nominal_53_p <- ggplot(
  no_nominal_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .05))

no_nominal_35_p <- ggplot(
  no_nominal_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .05))

no_nominal_55_p <- ggplot(
  no_nominal_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .06))
```

Code for boxplots of weak association for a nominal variable.

Code for boxplots of moderate association for a nominal variable.

```
moderate_nominal_33 <- readRDS(
  "simulations/moderate_nominal_33.Rds"
) |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
moderate_nominal_35 <- readRDS(
  "simulations/moderate_nominal_35.Rds"
) |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
moderate_nominal_53 <- readRDS(
  "simulations/moderate_nominal_53.Rds"
) |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
moderate_nominal_55 <- readRDS(
  "simulations/moderate_nominal_55.Rds"
) |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

moderate_nominal_33_p <- ggplot(
  moderate_nominal_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .3))

moderate_nominal_35_p <- ggplot(
  moderate_nominal_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .3))

moderate_nominal_53_p <- ggplot(
  moderate_nominal_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .4))

moderate_nominal_55_p <- ggplot(
  moderate_nominal_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .4))
```

Code for boxplots of strong association for a nominal variable.

Code for boxplots of very strong association for a nominal variable.

```
very_nominal_33 <- readRDS("simulations/very_nominal_33.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
very_nominal_35 <- readRDS("simulations/very_nominal_35.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
very_nominal_53 <- readRDS("simulations/very_nominal_53.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))
very_nominal_55 <- readRDS("simulations/very_nominal_55.Rds") |>
  mutate(across(type, factor,
    levels = c("model_free", "model", "model_poor")
  ))

very_nominal_33_p <- ggplot(
  very_nominal_33,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .6))

very_nominal_35_p <- ggplot(
  very_nominal_35,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
```

```r
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .6))

very_nominal_53_p <- ggplot(
  very_nominal_53,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .6))

very_nominal_55_p <- ggplot(
  very_nominal_55,
  aes(x = value, color = size)
) +
  geom_boxplot() +
  coord_flip() +
  facet_wrap(~type, labeller = labeller(type = model.labs)) +
  xlim(c(0, .6))
```

A.4.3 Code for real data

Code to calculate the CCRAM for Table 5.5.

```r
ideology <- read.csv("data/polviews.csv") |>
  pivot_longer(cols = c(y1, y2, y3, y4, y5), names_to = "ideology") |>
  mutate(ideology = as.numeric(str_extract(ideology, "[0-9]")))
ideology_observations <- countsToCases(ideology, countcol = "value")
female <- ideology_observations |>
  dplyr::filter(sex == 1)
male <- ideology_observations |>
  dplyr::filter(sex == 2)

getBECCR_2(var_index = 3, data = ideology_observations)
getBECCR_2(var_index = 3, data = ideology_observations) /
  (12 * getVariance(var_index = 3, data = ideology_observations))
# 0.4920222

sex_only <- select(ideology_observations, -party)
```

```r
party_only <- select(ideology_observations, -sex)

getBECCR_2(var_index = 2, data = sex_only)
getBECCR_2(var_index = 2, data = sex_only) /
  (12 * getVariance(var_index = 2, data = sex_only))
getBECCR_2(var_index = 2, data = party_only) /
  (12 * getVariance(var_index = 2, data = party_only))
```

```r
# confidence intervals
library(boot)
get_beccr <- function(data, indices, var_index) {
  d <- data[indices, ]
  getBECCR_2(var_index = var_index, data = d)
}

# CI for everything, ideology as response
boot_total_ideology <- boot(
  ideology_observations,
  var_index = 3,
  R = 1000,
  statistic = get_beccr
)
# CI for everything, party as response
boot_total_party <- boot(
  ideology_observations,
  var_index = 2,
  R = 1000,
  statistic = get_beccr
)

saveRDS(boot_total_ideology, "data/boot/boot_total_ideology.Rds")
saveRDS(boot_total_party, "data/boot/boot_total_party.Rds")


# CI for sex, ideology as response
boot_sex_ideology <- boot(
  sex_only,
  var_index = 2,
  R = 1000,
  statistic = get_beccr
)
```

```
# CI for party, party as response
boot_party_ideology <- boot(
  party_only,
  var_index = 2,
  R = 1000,
  statistic = get_beccr
)

saveRDS(boot_sex_ideology, "data/boot/boot_sex_ideology.Rds")
saveRDS(boot_party_ideology, "data/boot/boot_party_ideology.Rds")
```

```
boot_total_ideology <- readRDS("data/boot/boot_total_ideology.Rds")
boot_total_party <- readRDS("data/boot/boot_total_party.Rds")
boot_sex_ideology <- readRDS("data/boot/boot_sex_ideology.Rds")
boot_party_ideology <- readRDS("data/boot/boot_party_ideology.Rds")


boot_females_ideology <- readRDS("data/boot/boot_females_ideology.Rds")
boot_females_party <- readRDS("data/boot/boot_females_party.Rds")
boot_males_ideology <- readRDS("data/boot/boot_males_ideology.Rds")
boot_males_party <- readRDS("data/boot/boot_males_party.Rds")
boot::boot.ci(boot_total_ideology, type = "bca")
boot::boot.ci(boot_total_party, type = "bca")

boot::boot.ci(boot_sex_ideology, type = "bca")
boot::boot.ci(boot_party_ideology, type = "bca")



boot::boot.ci(boot_females_ideology, type = "bca")
boot::boot.ci(boot_females_party, type = "bca")
boot::boot.ci(boot_males_ideology, type = "bca")
boot::boot.ci(boot_males_party, type = "bca")
```

Code for Figure 5.15.

```
set.seed(713)
prediction_data <- expand.grid(
  sex = unique(ideology_observations$sex),
  party =
    unique(ideology_observations$party)
)
for (n in 1:1000) {
  resample_ideology <- resample(ideology_observations,
    nrow(ideology_observations),
    replace = TRUE
  )[, 1:3]
  resampled_prediction <- prediction_data |>
    rowwise() |>
    mutate(prediction_index = getPrediction(
      var_index = 3,
      idx_vec = c(sex, party),
      data = resample_ideology
    ))
  prediction_data <- data.frame(prediction_data,
    name =
      resampled_prediction$prediction_index
  )
}
saveRDS(prediction_data, "simulations/prediction_data.Rds")
```

```
prediction_data <- readRDS("simulations/prediction_data.Rds")

long_form <- prediction_data |>
  pivot_longer(cols = "name":"name.999", names_to = "sample")

counts <- long_form |>
  dplyr::count(sex, party, value)

combinations <- expand.grid(
  sex =
    unique(ideology_observations$sex),
  party =
    unique(ideology_observations$party),
  ideology =
    unique(ideology_observations$ideology)
)

full_data <- left_join(combinations, counts,
  by = c(
    "sex" = "sex",
    "party" = "party",
    "ideology" = "value"
  )
) |>
  replace_na(list(n = 0))

full_data <- full_data |>
  mutate(
    sex_name = ifelse(sex == 1, "F", "M"),
    party_name = ifelse(party == 1, "D", "R"),
    ideology_name = factor(case_when(
      ideology == 1 ~ "Very Liberal",
      ideology == 2 ~ "Slightly Liberal",
      ideology == 3 ~ "Moderate",
      ideology == 4 ~ "Slightly Conservative",
      ideology == 5 ~ "Very Conservative"
    ), levels = c(
      "Very Liberal", "Slightly Liberal",
      "Moderate", "Slightly Conservative",
      "Very Conservative"
    ))
  )


full_data$combination_sp <- paste(
  full_data$sex_name,
  full_data$party_name
)
full_data$combination_ps <- paste(
  full_data$party_name,
  full_data$sex_name
)

ggplot(full_data, aes(x = combination_ps, y = ideology_name)) +
  geom_point(aes(
    size =
      ifelse(n == 0, NA, n * 100), color = sex
  ),
  shape = 21, colour = "black",
  fill = "white", stroke = .5
  ) +
  scale_size_continuous(range = c(1, 20)) +
  geom_point(data = full_data, aes(
    x = combination_ps,
    y = ideology_name,
    size =
      ifelse(n == 0 | n <= 500,
        NA, .1
      )
  )) +
  # geom_point(full_data, aes(x = combination_ps, y = ideology_name)) +
  geom_text(aes(label = ifelse(n == 0, "", n)),
    vjust = -1.2, size = 2
  ) +
  theme(
    legend.position = "none",
    text = element_text(size = 13),
    axis.text.y = element_text(
      angle = 45, vjust = 0.5,
      hjust = 1, size = 8
    ),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.background = element_blank(),
    axis.line = element_line(colour = "black")
  ) +
  labs(
    x = "Combination of Party and Sex",
    y = "Political Ideology"
  )
```

Code for model output summary.

```
table_ideology <- read.csv("data/polviews.csv") |>
  mutate(sex = as.factor(sex), party = as.factor(party))
logit.real <- VGAM::vglm(cbind(y1, y2, y3, y4, y5) ~ sex * party,
  data = table_ideology,
  family = cumulative(parallel = TRUE)
)
summary(logit.real)
# logit.polr <- polr(as.factor(ideology) ~  sex * party,
# method="logistic",data = ideology_observations)
# brant::brant(logit.polr)
```

Code for model-based rescaled measures.

```r
table_ideology <- read.csv("data/polviews.csv")
logit.sex <- VGAM::vglm(cbind(y1, y2, y3, y4, y5) ~ sex,
  data = table_ideology,
  family = cumulative(
    link = "logitlink",
    parallel = TRUE,
    reverse = FALSE
  )
)
summary(logit.sex)
# check proportional odds
sex_model <- polr(as.factor(ideology) ~ sex,
  method = "logistic",
  data = ideology_observations
)
brant::brant(sex_model)
# gof test
1 - pchisq(413.054, df = 11)

sex_cond <- cbind(predict(logit.sex,
  newdata =
    data.frame(sex = 1:2), type = "response"
))
sex_px <- getPMF(var_index = 1, data = sex_only)
sex_model_tab <- sex_cond * sex_px
# scaled model_BECCR
getBECCR(var_index = 2, data = sex_model_tab) /
  (12 * getVariance(var_index = 2, data = sex_model_tab))
```

```r
# model for party only
logit.party <- VGAM::vglm(cbind(y1, y2, y3, y4, y5) ~ party,
  data = table_ideology,
  family = cumulative(
    link = "logitlink",
    parallel = TRUE,
    reverse = FALSE
  )
)
summary(logit.party)
```

```
# check proportional odds
party_model <- polr(as.factor(ideology) ~ party,
  method = "logistic",
  data = ideology_observations
)
brant::brant(party_model)
# gof test
1 - pchisq(9.9069, df = 11)

party_cond <- cbind(predict(logit.party,
  newdata = data.frame(party = 1:2),
  type = "response"
))
party_px <- getPMF_2(var_index = 2, data = ideology_observations)
party_model_tab <- party_cond * party_px
getBECCR(var_index = 2, data = party_model_tab) /
  (12 * getVariance(var_index = 2, data = party_model_tab))
# 0.488223
```

```
# model with both

logit.both <- VGAM::vglm(cbind(y1, y2, y3, y4, y5) ~ party + sex,
  data = table_ideology,
  family = cumulative(
    link = "logitlink",
    parallel = TRUE,
    reverse = FALSE
  )
)
summary(logit.both)
# check proportional odds
both_model <- polr(as.factor(ideology) ~ party + sex,
  method = "logistic",
  data = ideology_observations
)
brant::brant(both_model)
# gof test
1 - pchisq(9.8072, df = 10)

# first find the empirical probability of sex and party
sex_given_party <- getConditionalPMFwo(var_index = 1, data = party_sex)
```

```
party_pmf <- tibble(
  party_pmf =
    getPMF_2(
      var_index = 2,
      data =
        ideology_observations
    )
) |>
  cbind(party = 1:2)
# pmf of party and sex
pmf_party_sex <- left_join(sex_given_party,
  party_pmf,
  by = "party"
) |>
  mutate(joint_party_sex = freq_cond * party_pmf) |>
  select(sex, party, joint_party_sex)


# get the conditional pmf of ideology given sex and party
ideology_given_both <- getConditionalPMFwo(
  var_index = 3,
  data =
    ideology_observations
)
# model-based
newdata <- expand.grid(sex = 1:2, party = 1:2)

ideology_given_both <- data.frame(
  sex = c(1, 2, 1, 2),
  party = c(1, 1, 2, 2),
  predict(logit.both,
    newdata = newdata,
    type = "response"
  )
) |>
  pivot_longer(
    cols = y1:y5, names_to = "ideology",
    values_to = "freq_cond",
    names_transform =
      list(ideology = readr::parse_number)
  ) |>
  left_join(pmf_party_sex, by = c("sex", "party")) |>
  mutate(joint_pmf = freq_cond * joint_party_sex) |>
  select(sex, party, ideology, joint_pmf, freq_cond)

total_possibilities <- expand.grid(
  sex = 1:Icat, party = 1:Kcat,
  ideology = 1:Jcat
) |>
  left_join(ideology_given_both) |>
  mutate(
    joint_pmf = ifelse(is.na(joint_pmf), 0, joint_pmf),
    freq_cond = ifelse(is.na(freq_cond), 0, freq_cond)
  )

combinations <- expand.grid(
  sex = 1:Icat, party = 1:Kcat,
  ideology = 1:Jcat
)

# get the scores
ideology_pmf_1 <- total_possibilities |>
  group_by(ideology) |>
  summarize(pmf = sum(joint_pmf)) |>
  select(pmf)
ideology_cdf <- mutate(ideology_pmf_1, cdf = cumsum(pmf))
ideology_cdf <- append(ideology_cdf$cdf, value = 0, after = 0)
ideology_scores <- getScores(data = ideology_cdf)
ideology_scores <- data.frame(
  ideology = 1:5,
  scores = ideology_scores
)
ideology_pmf <- ideology_pmf_1$pmf
# get the variance
var <- 0
for (i in 1:5) {
  var <- var + ideology_cdf[i] *
    ideology_cdf[i + 1] *
    ideology_pmf[i] / 4
}

# getting the marginal probability without ideology
# getPMFwo(var_index = 3, data = ideology_observations)
PMFwo <- total_possibilities |>
```

```
  group_by(sex, party) |>
  mutate(pmfWO = sum(joint_pmf)) |>
  distinct(pmfWO)

total_possibilities <- left_join(total_possibilities,
  ideology_scores,
  by = "ideology"
) |>
  left_join(PMFwo)

measure_both <- total_possibilities |>
  group_by(sex, party) |>
  mutate(temp1 = freq_cond * scores) |>
  summarize(temp2 = ((sum(temp1) - .5)^2 * pmfWO)) |>
  distinct()

# model-based measure
12 * sum(measure_both$temp2) / (12 * var)
# 0.4884015 model-based
```

```
logit.interaction <- VGAM::vglm(
  cbind(y1, y2, y3, y4, y5) ~ party * sex,
  data = table_ideology,
  family = cumulative(
    link = "logitlink",
    parallel = TRUE,
    reverse = FALSE
  )
)
summary(logit.interaction)
# gof
1 - pchisq(8.4528, df = 9)
# brant test
interaction_model <- polr(as.factor(ideology) ~ party * sex,
  method = "logistic",
  data = ideology_observations
)
brant::brant(interaction_model)

# find predicted
ideology_given_both2 <- data.frame(
```

```r
  sex = c(1, 2, 1, 2),
  party = c(1, 1, 2, 2),
  predict(logit.interaction,
    newdata = newdata,
    type = "response"
  )
) |>
  pivot_longer(
    cols = y1:y5, names_to = "ideology",
    values_to = "freq_cond",
    names_transform =
      list(ideology = readr::parse_number)
  ) |>
  left_join(pmf_party_sex, by = c("sex", "party")) |>
  mutate(joint_pmf = freq_cond * joint_party_sex) |>
  select(sex, party, ideology, joint_pmf, freq_cond)

total_possibilities2 <- expand.grid(
  sex = 1:Icat,
  party = 1:Kcat,
  ideology = 1:Jcat
) |>
  left_join(ideology_given_both2) |>
  mutate(
    joint_pmf = ifelse(is.na(joint_pmf), 0, joint_pmf),
    freq_cond = ifelse(is.na(freq_cond), 0, freq_cond)
  )

# get the scores
ideology_pmf2 <- total_possibilities2 |>
  group_by(ideology) |>
  summarize(pmf = sum(joint_pmf)) |>
  select(pmf)

ideology_cdf2 <- mutate(ideology_pmf2, cdf = cumsum(pmf))
ideology_cdf2 <- append(ideology_cdf2$cdf, value = 0, after = 0)
ideology_scores2 <- getScores(data = ideology_cdf2)
ideology_scores2 <- data.frame(
  ideology = 1:5,
  scores = ideology_scores2
)
ideology_pmf2 <- ideology_pmf2$pmf
```

```r
# get the variance
var <- 0
for (i in 1:5) {
  var <- var + ideology_cdf2[i] *
    ideology_cdf2[i + 1] *
    ideology_pmf2[i] / 4
}

# getting the marginal probability without ideology
# getPMFwo(var_index = 3, data = ideology_observations)
PMFwo2 <- total_possibilities2 |>
  group_by(sex, party) |>
  mutate(pmfWO = sum(joint_pmf)) |>
  distinct(pmfWO)


total_possibilities2 <- left_join(total_possibilities2,
  ideology_scores2,
  by = "ideology"
) |>
  left_join(PMFwo2)

measure2 <- total_possibilities2 |>
  group_by(sex, party) |>
  mutate(temp1 = freq_cond * scores) |>
  summarize(temp2 = ((sum(temp1) - .5)^2 * pmfWO)) |>
  distinct()

# model-based measure
12 * sum(measure2$temp2) / (12 * var)
# 0.4894932 with interaction
```

Appendix B Simulation Results

This appendix includes code for the simulation study and real world application as well as the results for all the simulations. This appendix includes the boxplots of simulation results.

B.1 Boxplots for no association and linear association

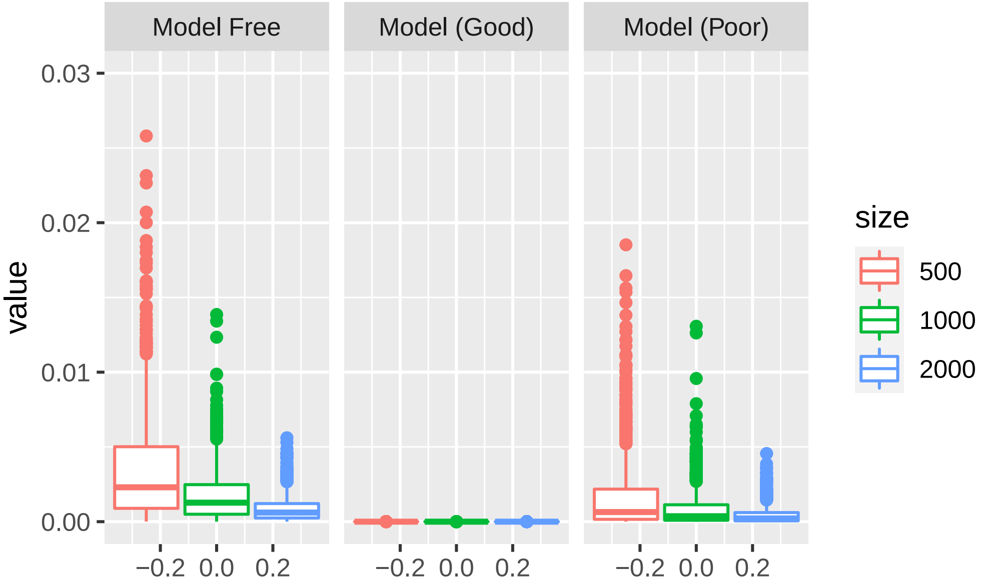

Figure B.1: No Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

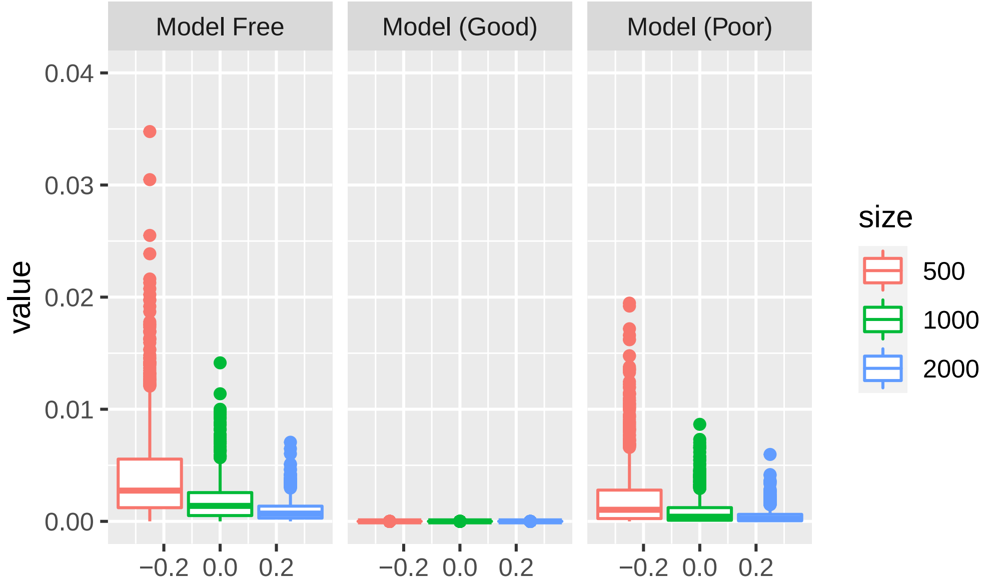

Figure B.2: No Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

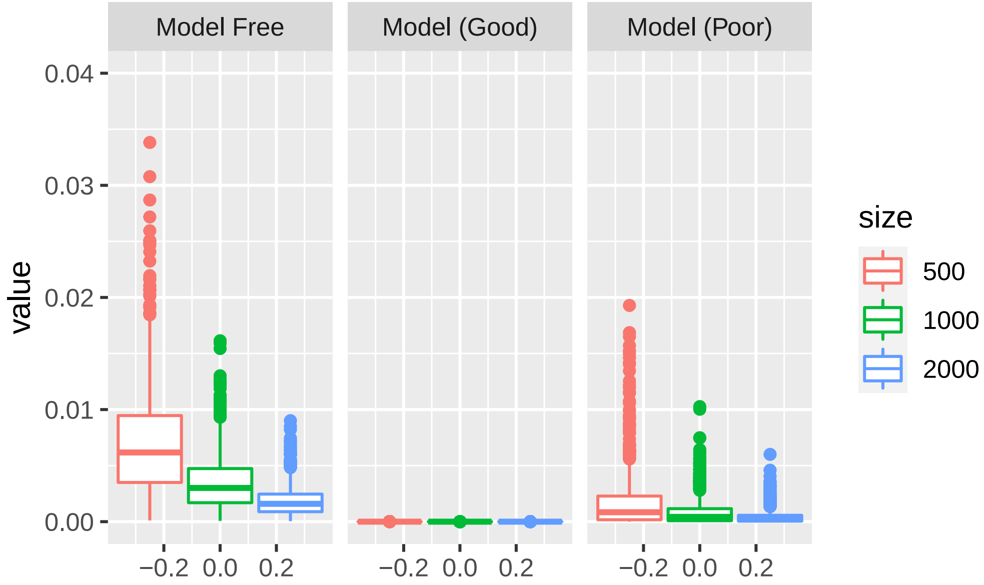

Figure B.3: No Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

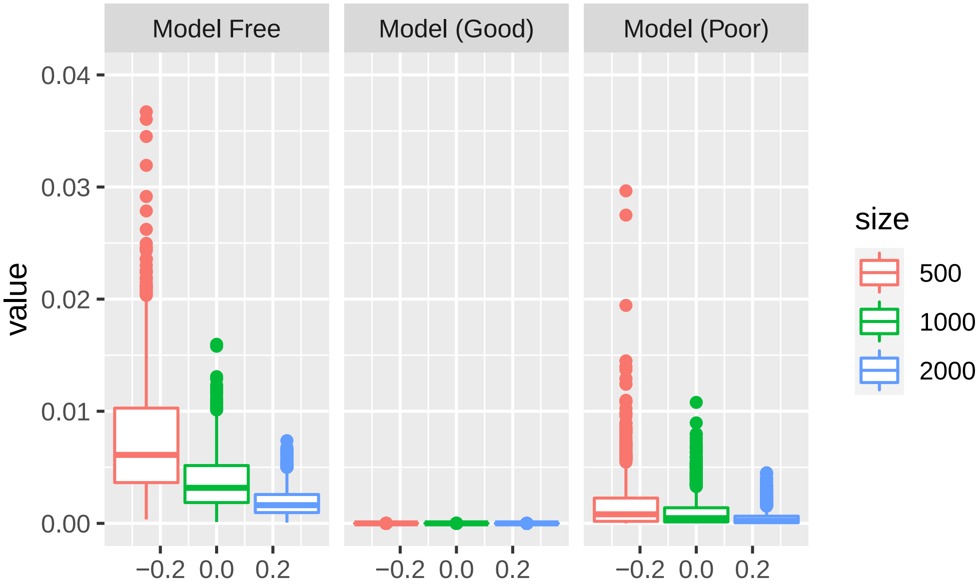

Figure B.4: No Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

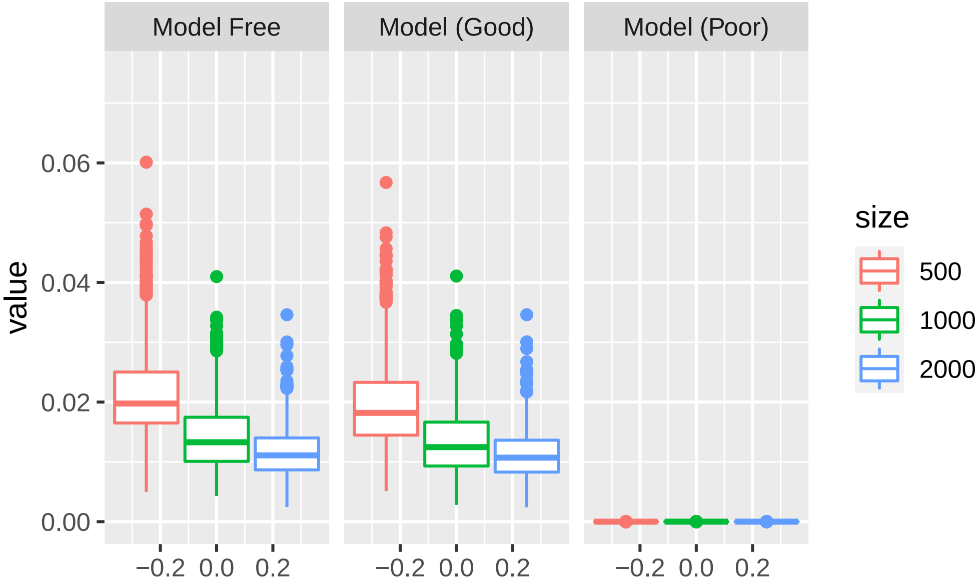

Figure B.5: Weak Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

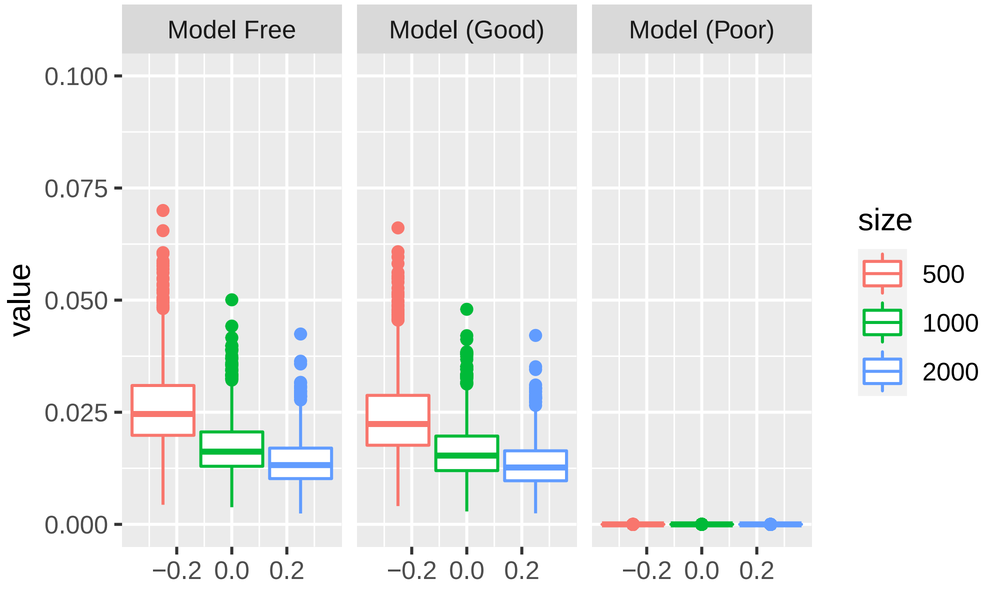

Figure B.6: Weak Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

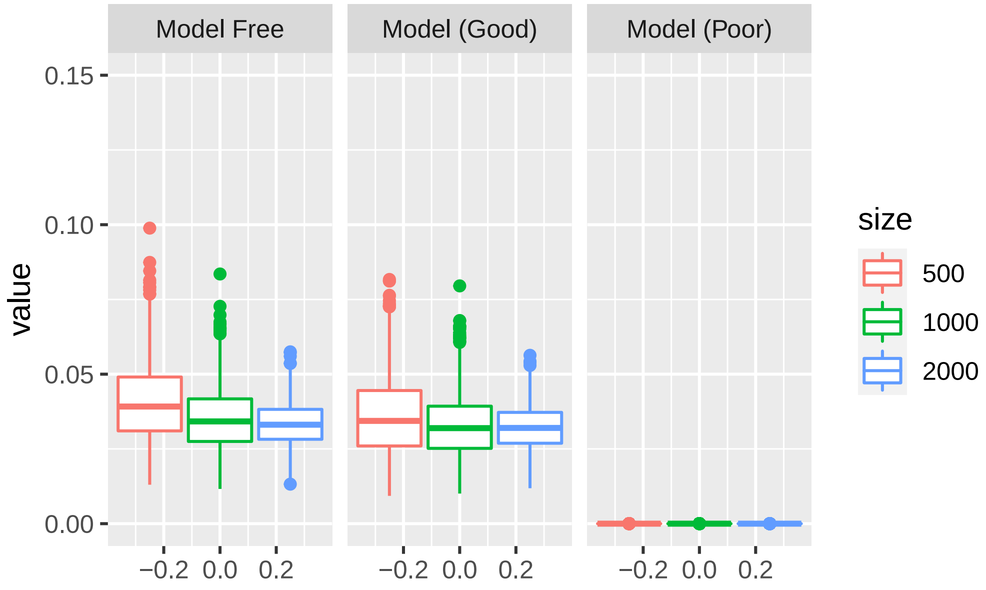

Figure B.7: Weak Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

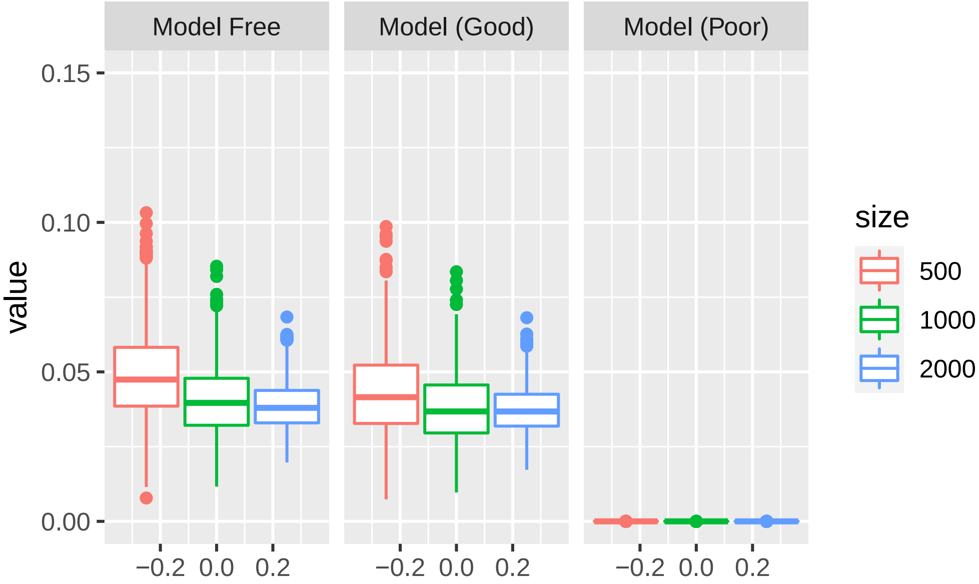

Figure B.8: Weak Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

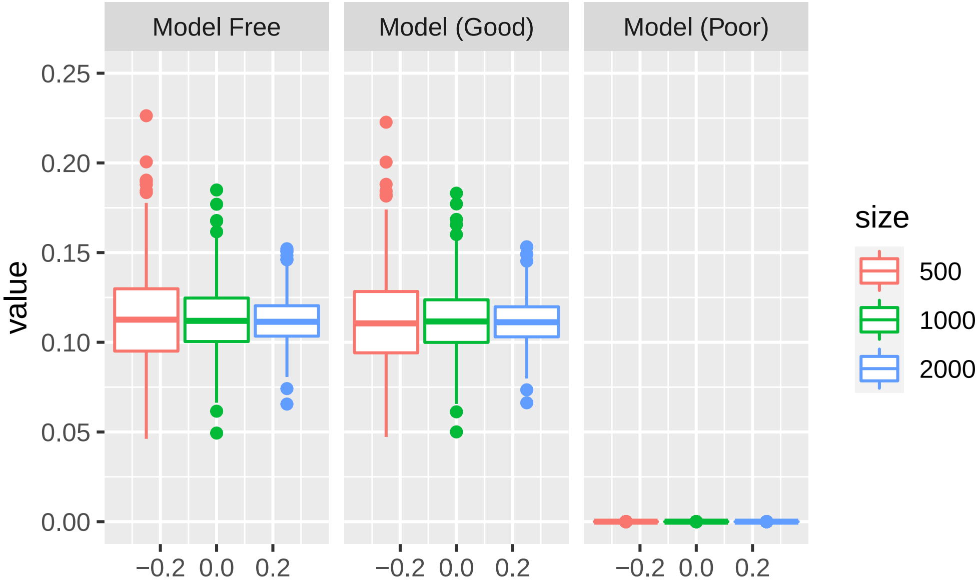

Figure B.9: Moderate Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

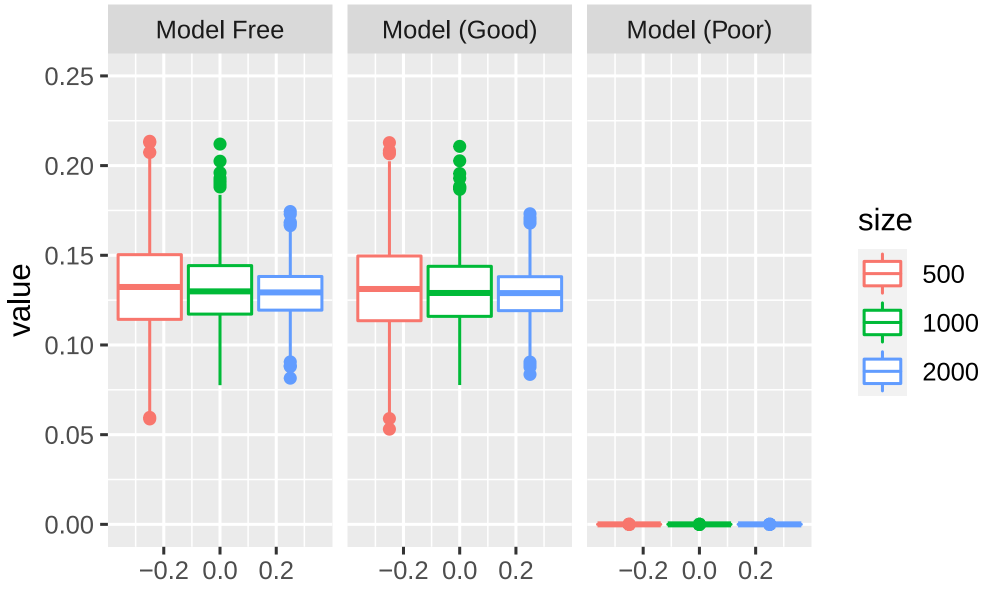

Figure B.10: Moderate Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3×5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

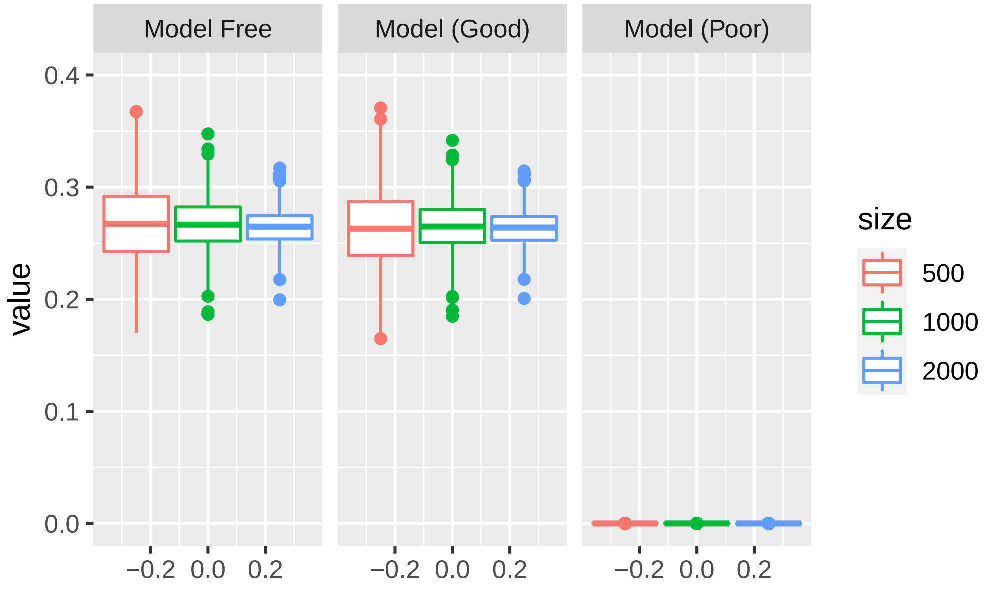

Figure B.11: Moderate Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5×3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

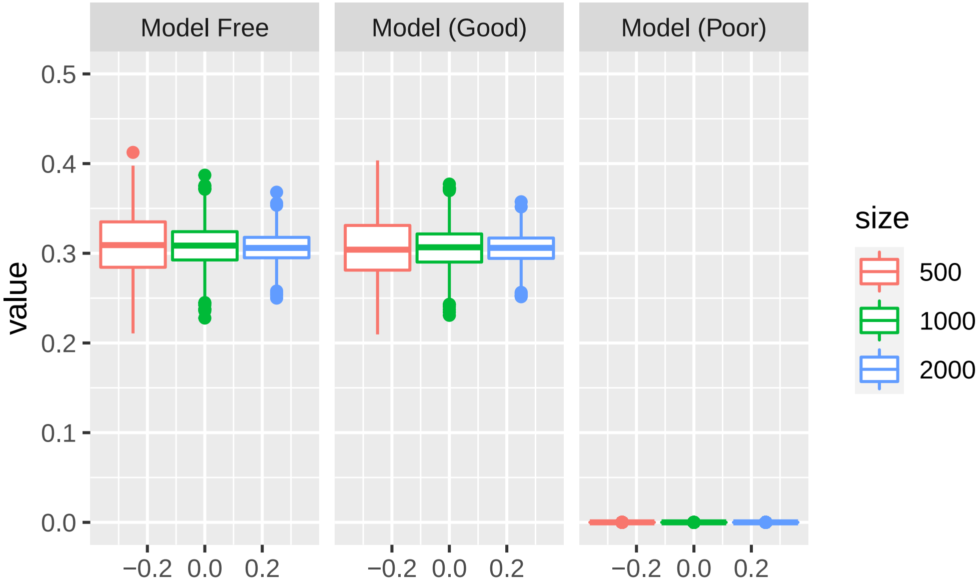

Figure B.12: Moderate Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5×5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

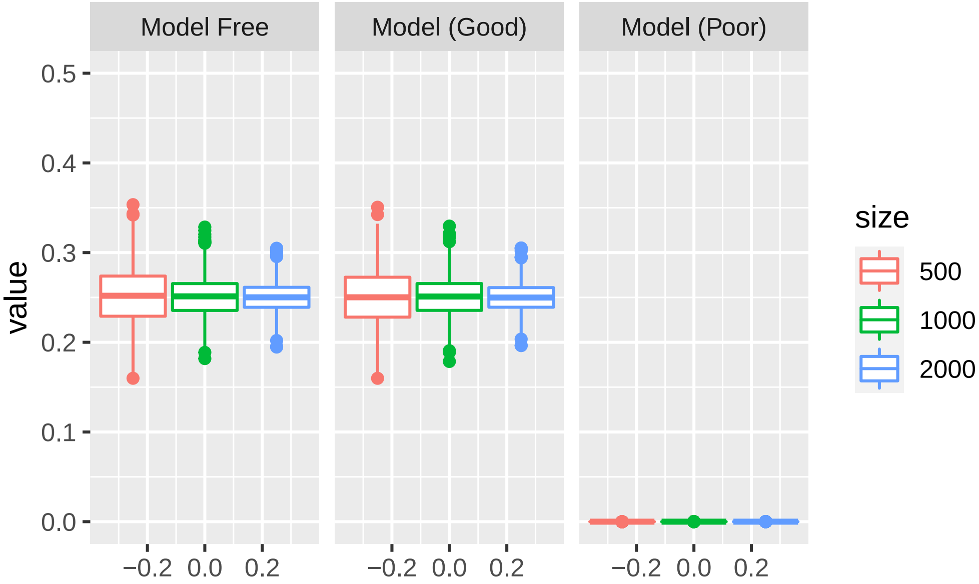

Figure B.13: Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

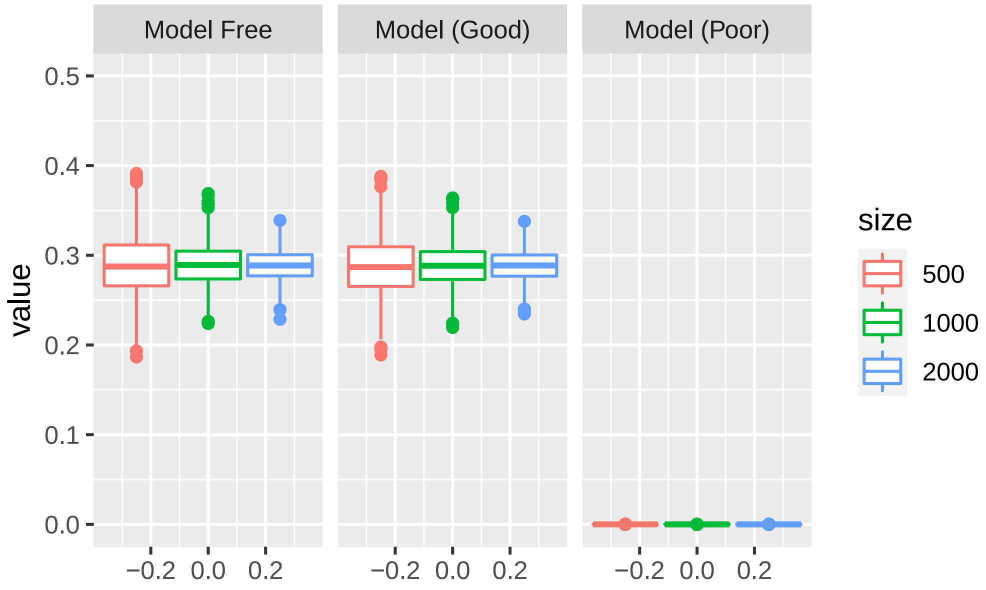

Figure B.14: (Strong Association) Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

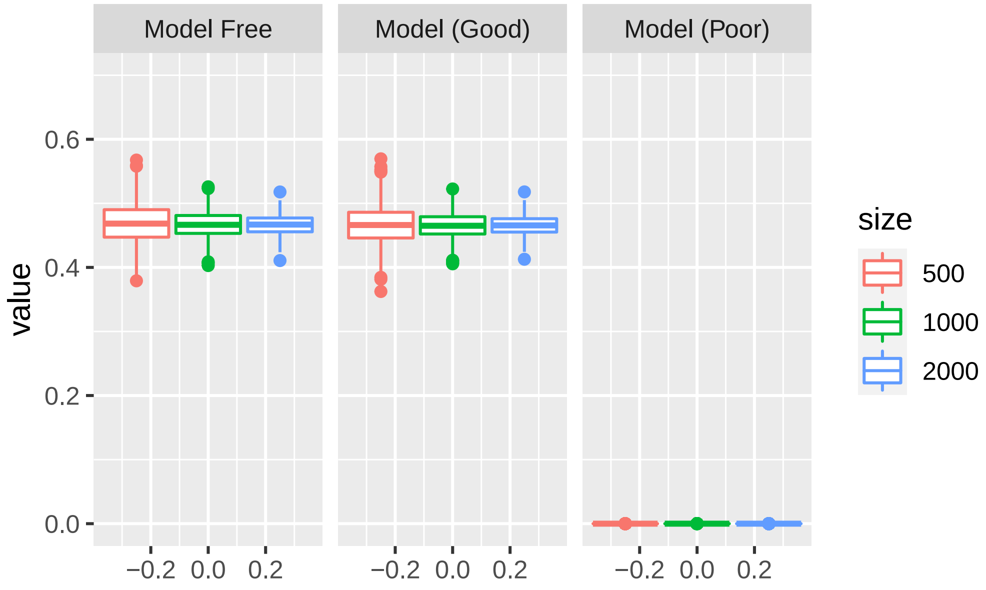

Figure B.15: Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5 × 3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

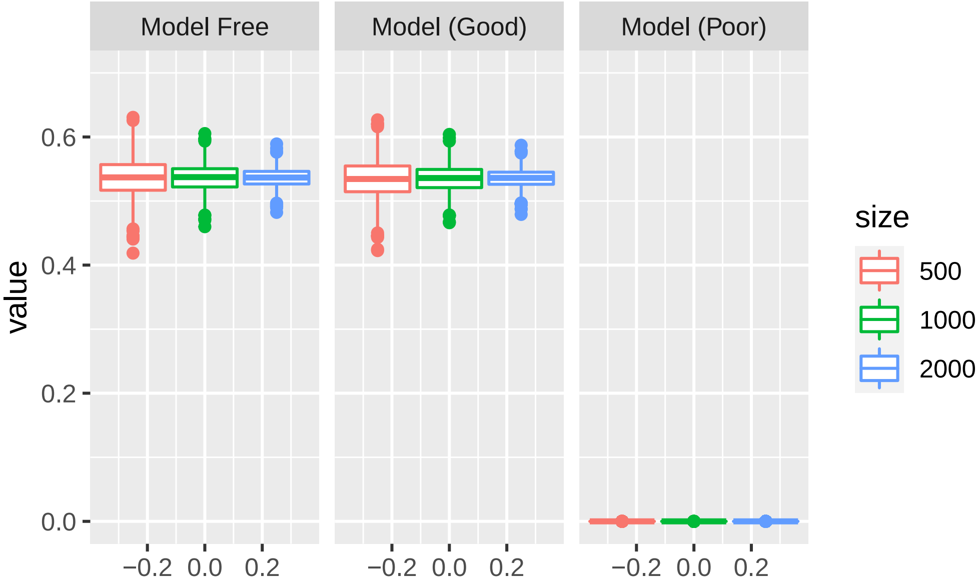

Figure B.16: Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

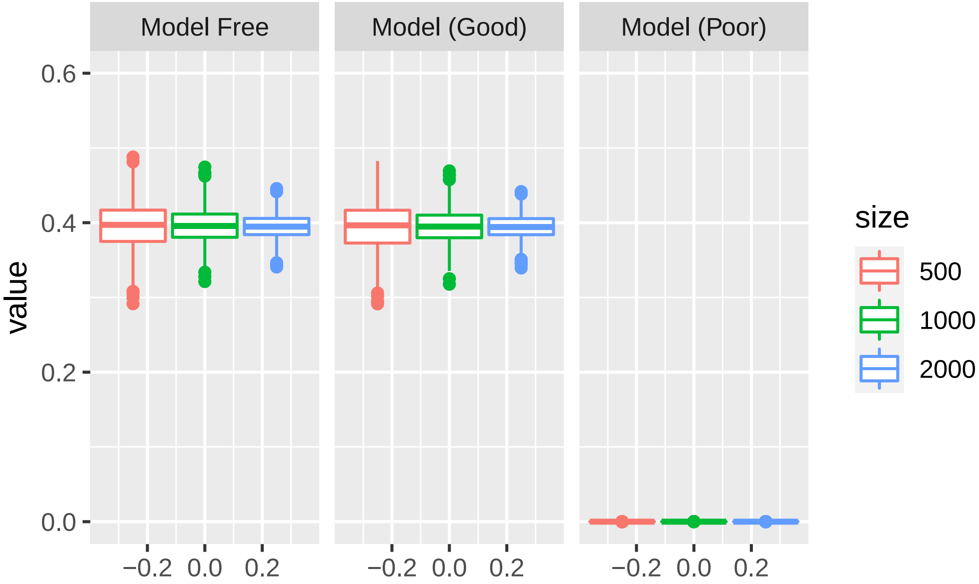

Figure B.17: Very Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

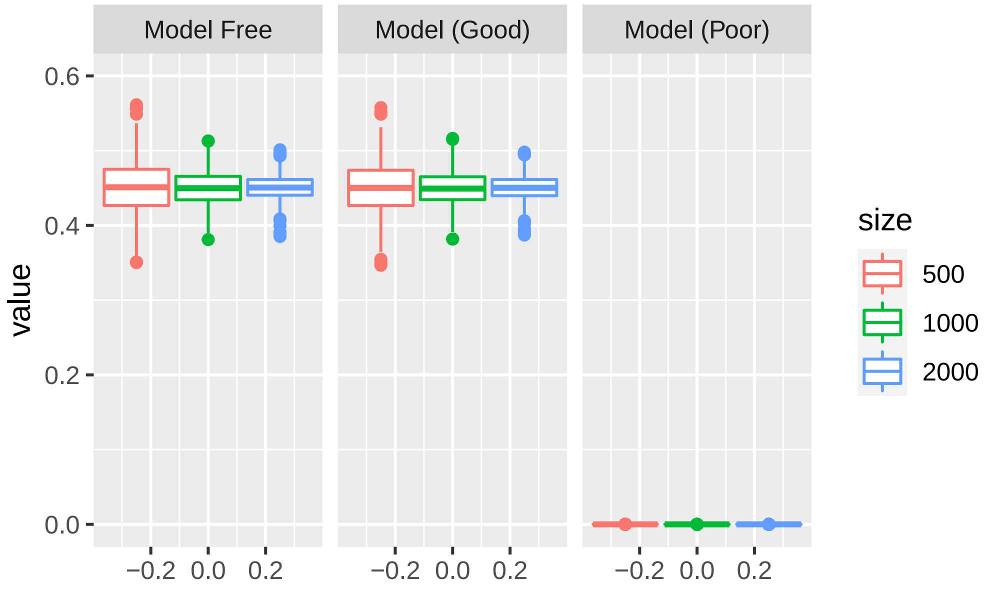

Figure B.18: Very Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5×3 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

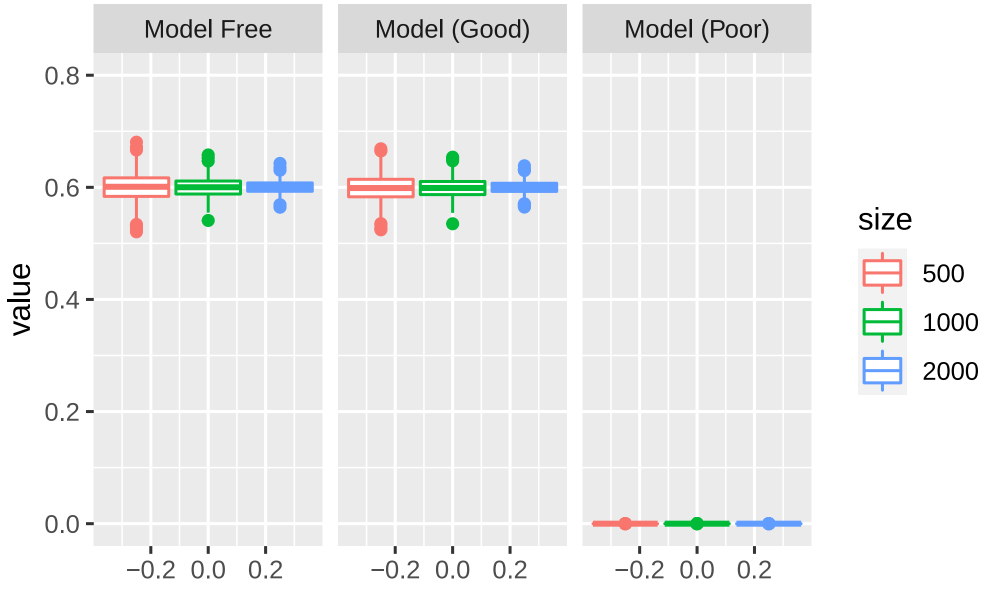

Figure B.19: Very Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

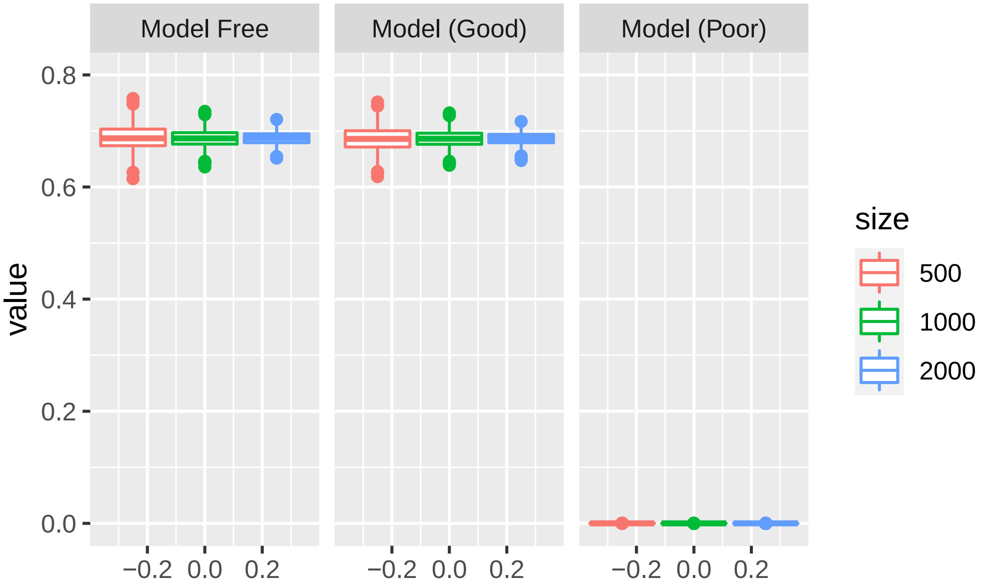

Figure B.20: Very Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5 × 5 table. Data were simulated from cumulative logit model with ordinal explanatory variable X.

B.2 Boxplots for nonmonotone nonlinear association

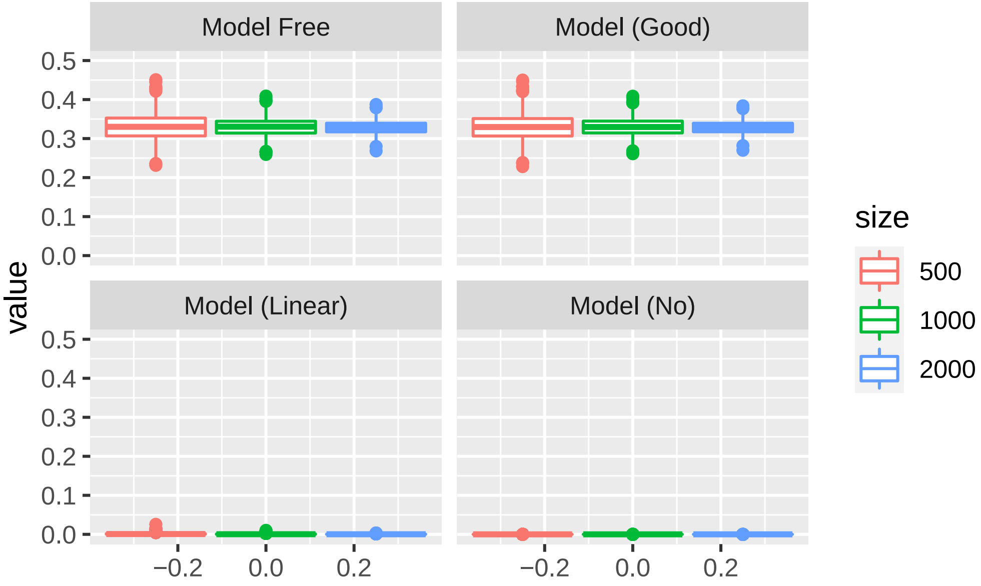

Figure B.21: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3×3 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

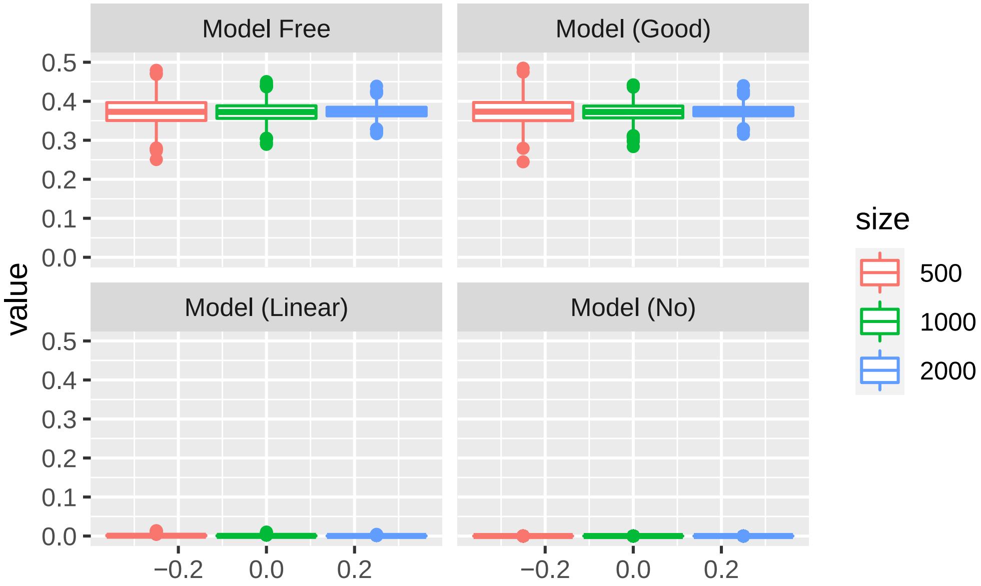

Figure B.22: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3×5 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

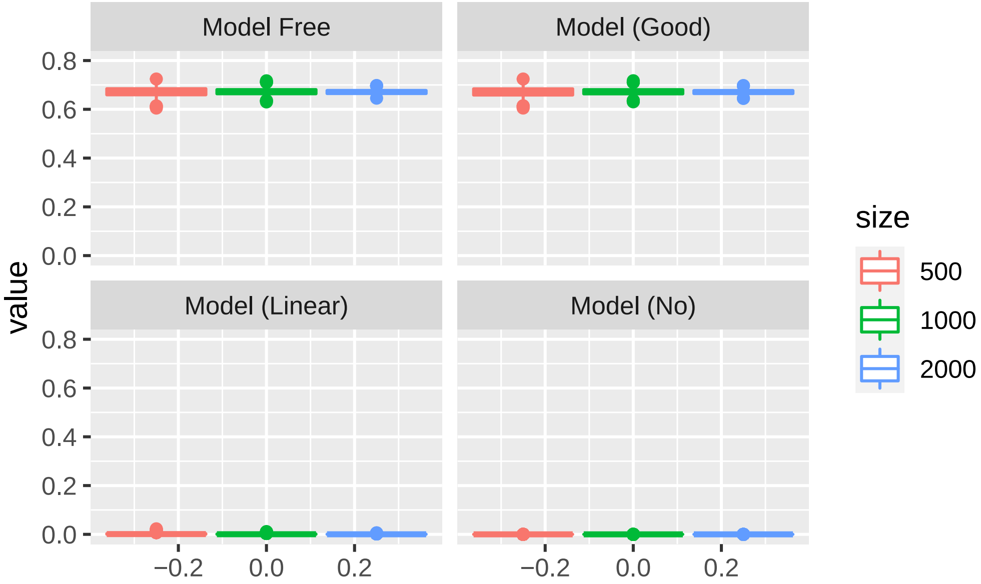

Figure B.23: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5×3 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

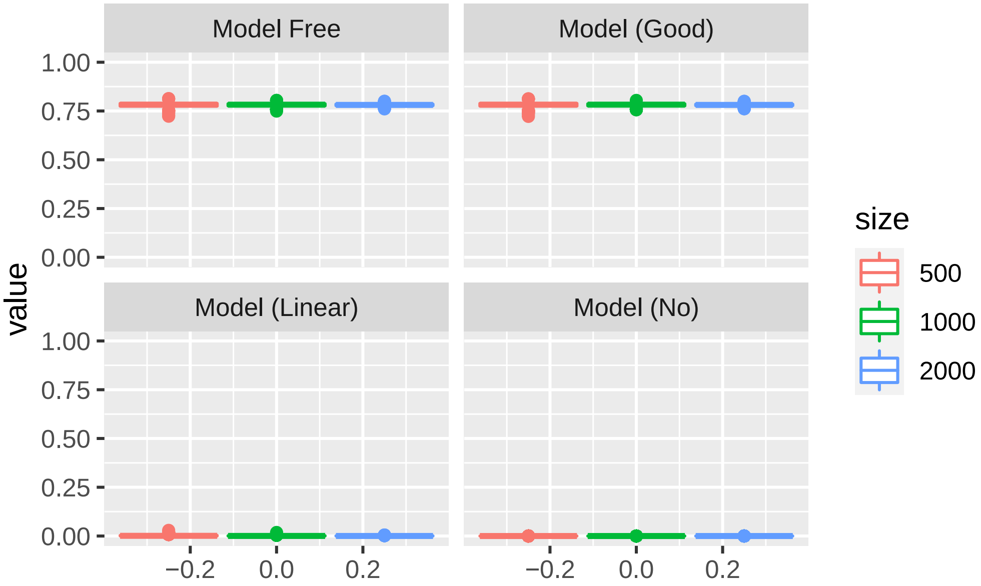

Figure B.24: Nonmonotone Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5×5 tables. Data were simulated from cumulative logit model with ordinal explanatory variable X.

B.3 Boxplots for no association with a nominal variable

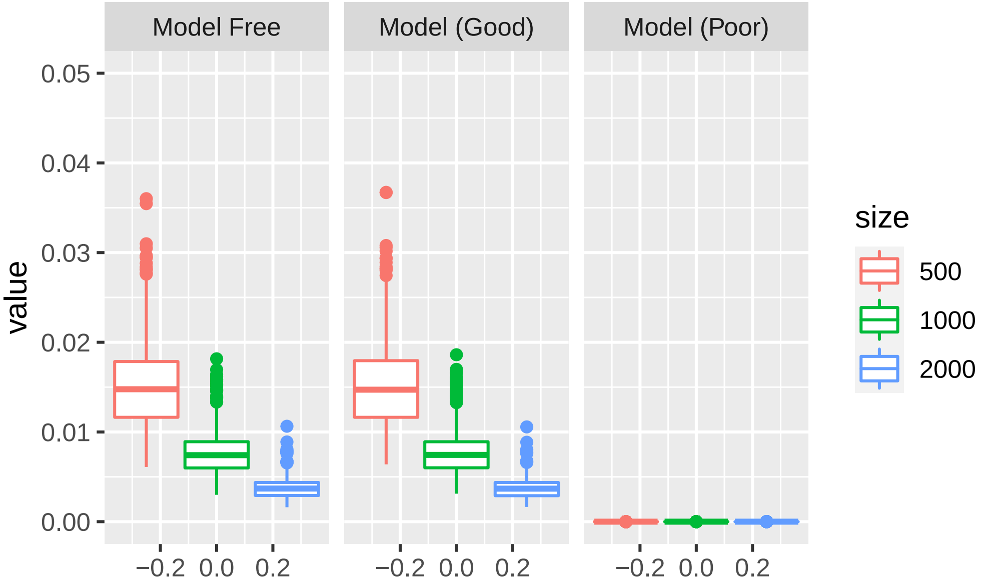

Figure B.25: No Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

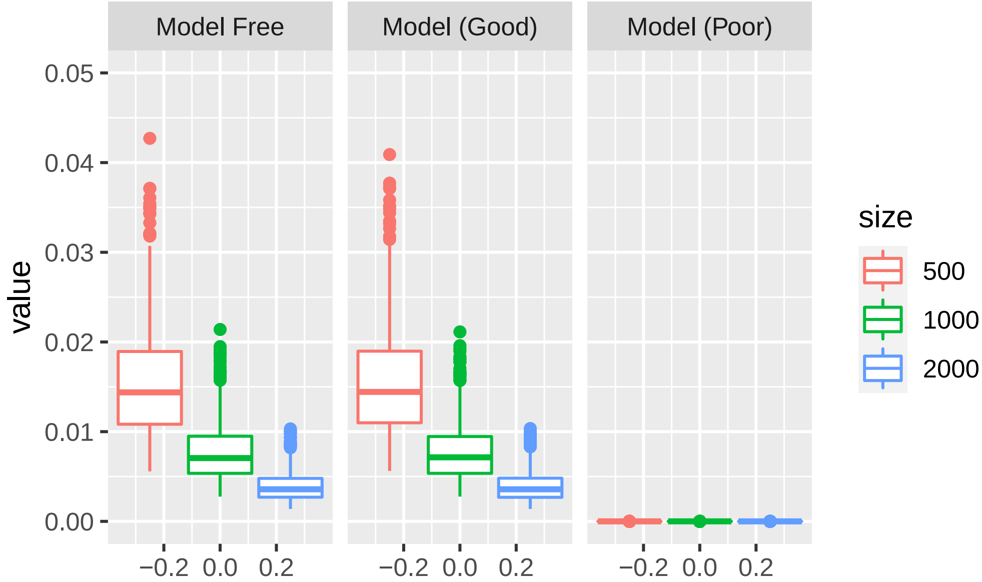

Figure B.26: No Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

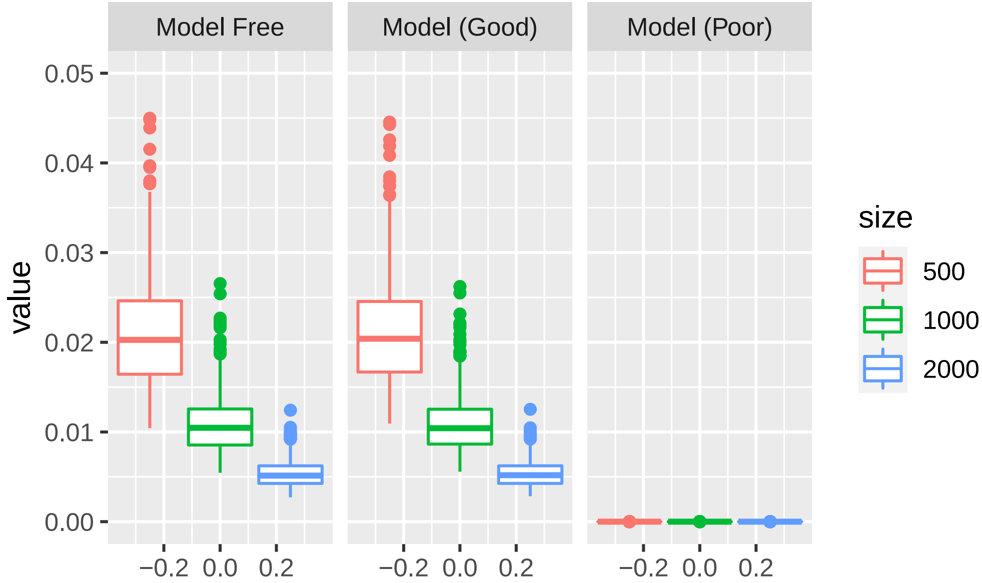

Figure B.27: No Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5 × 3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

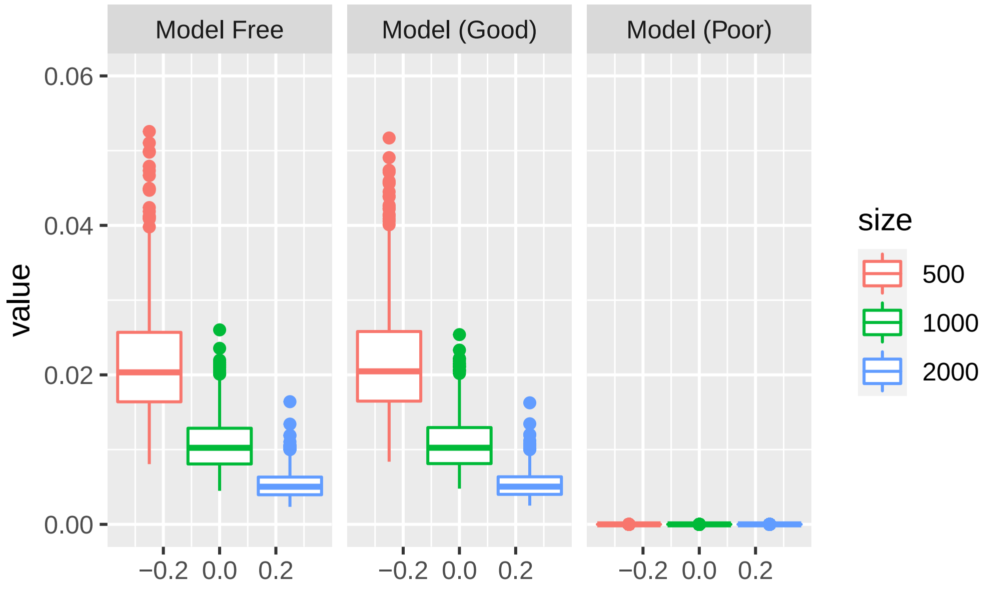

Figure B.28: No Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

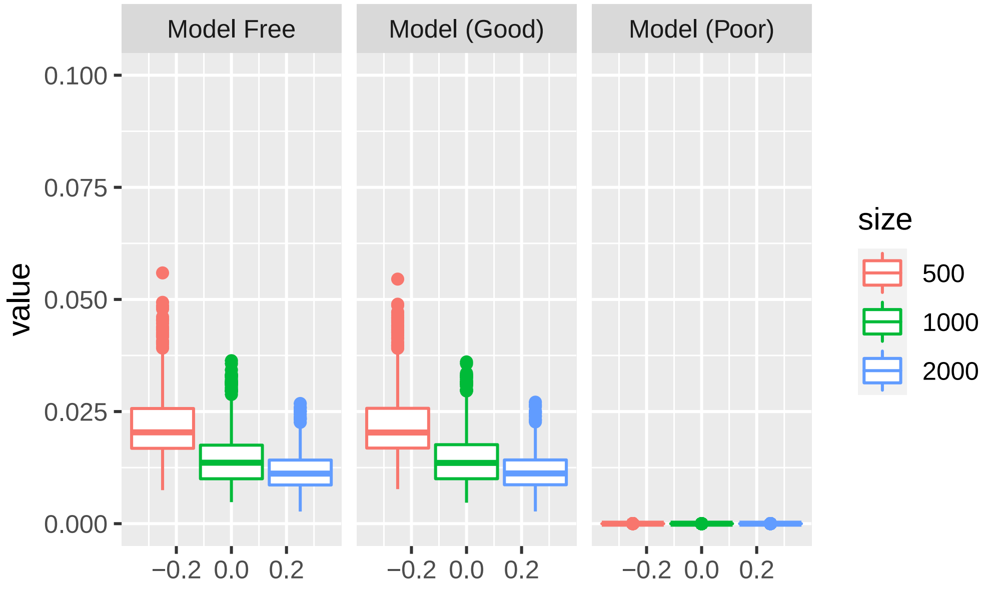

Figure B.29: Weak Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

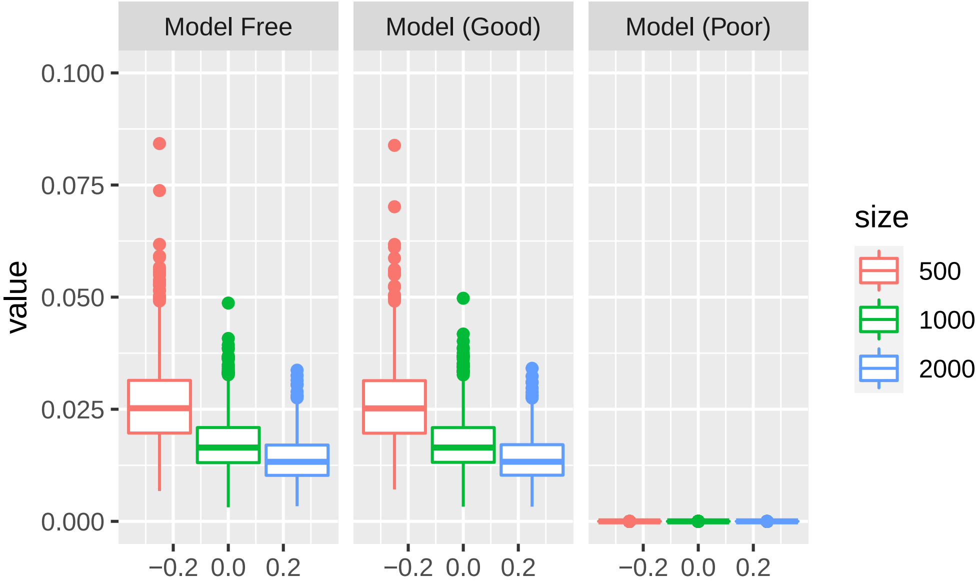

Figure B.30: Weak Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

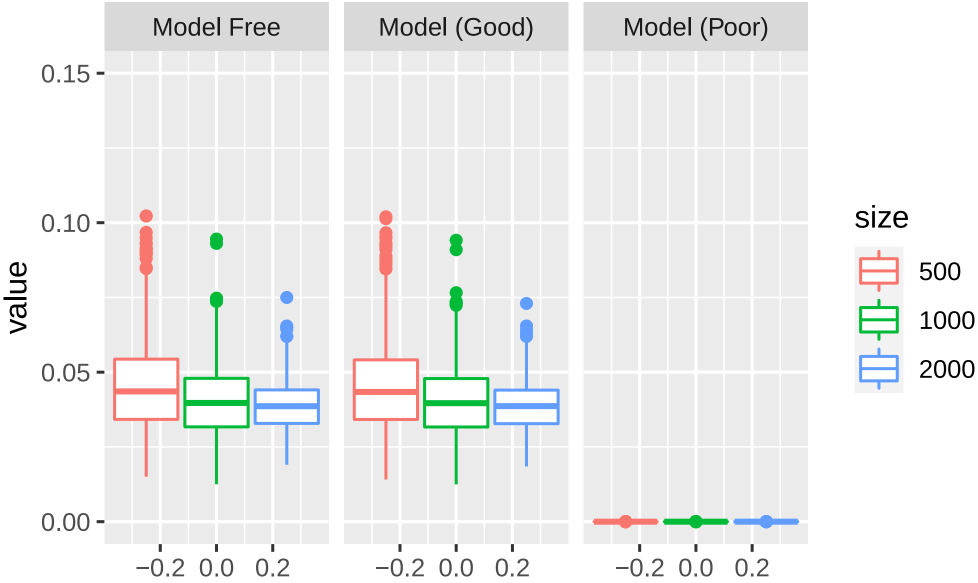

Figure B.31: Weak Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5 × 3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

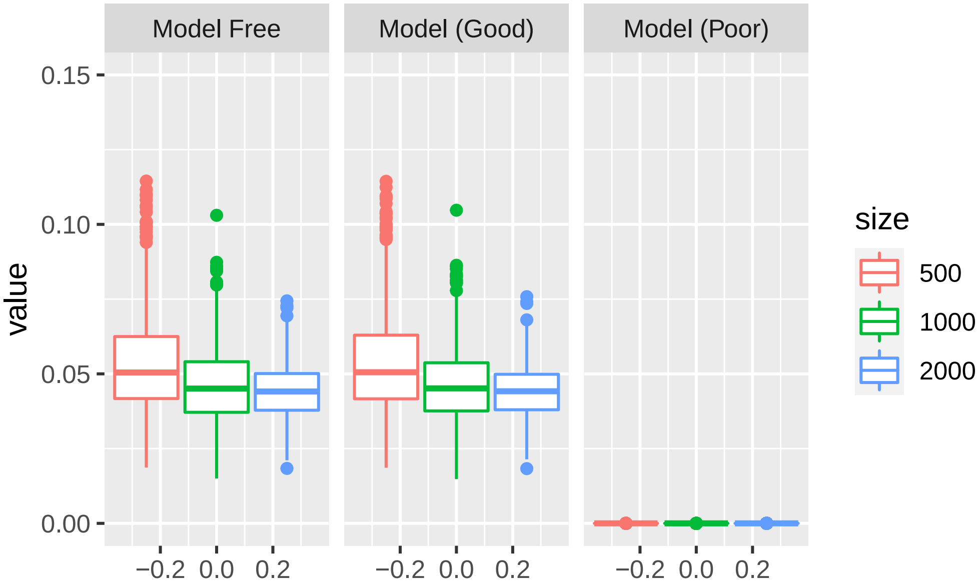

Figure B.32: Weak Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5×5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

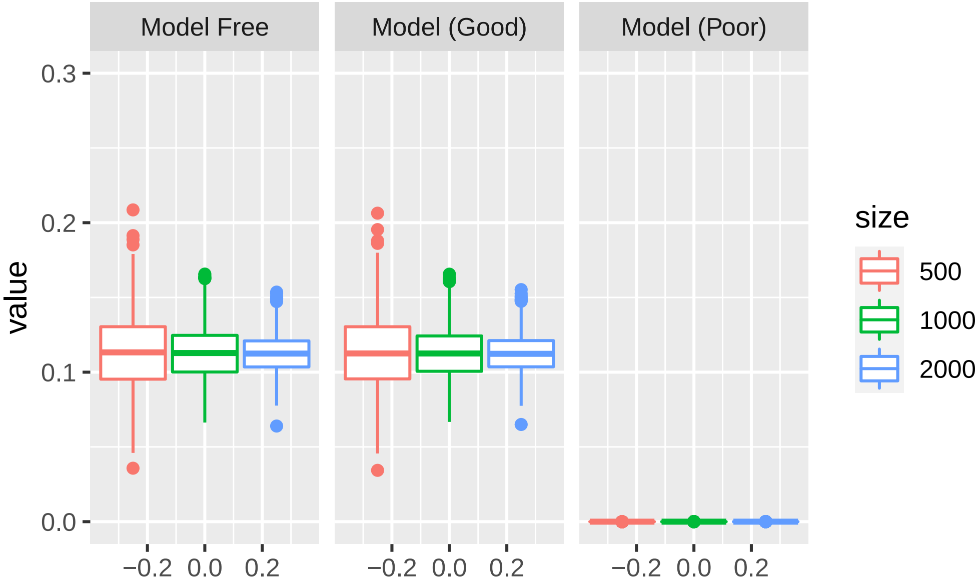

Figure B.33: Moderate Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3×3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

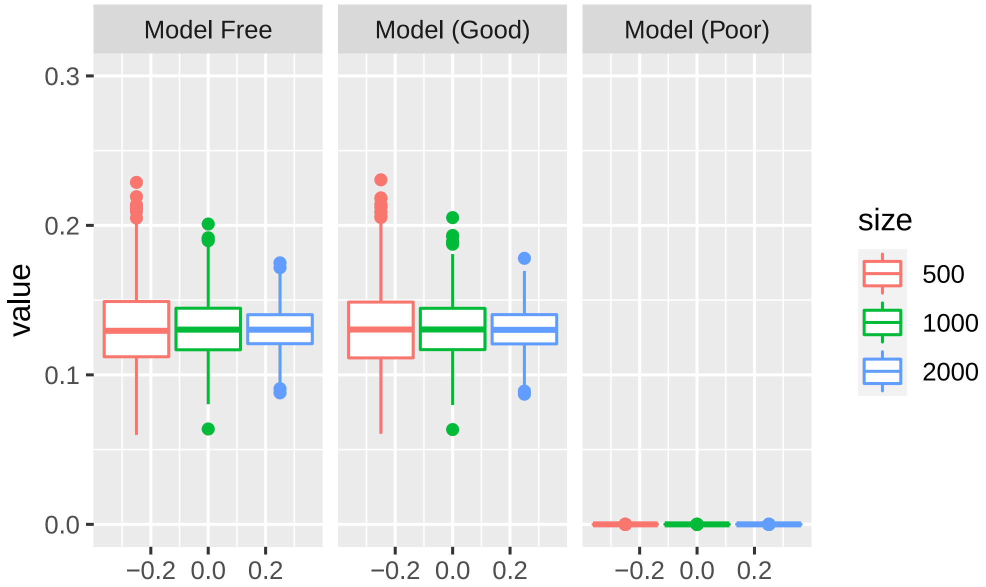

Figure B.34: Moderate Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

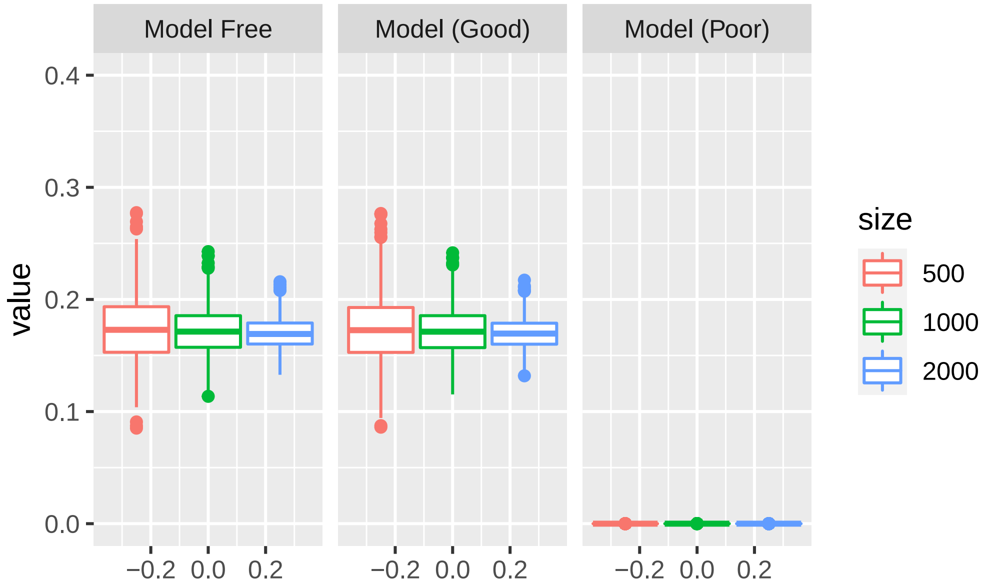

Figure B.35: Moderate Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

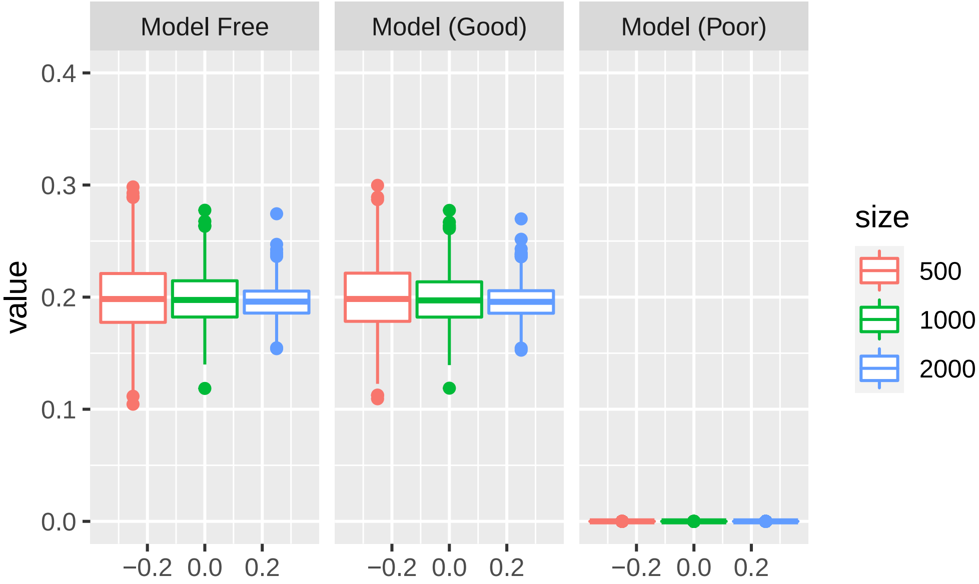

Figure B.36: Moderate Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

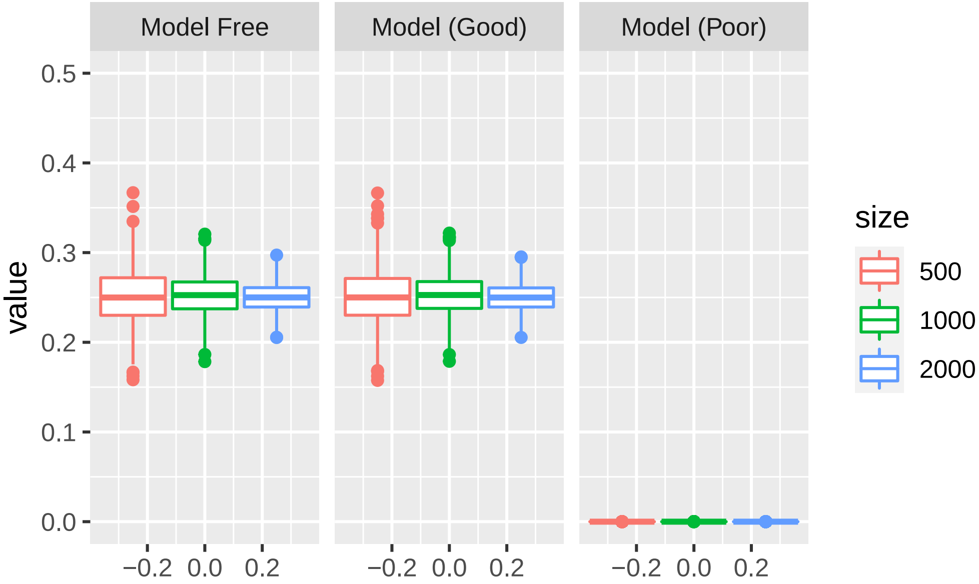

Figure B.37: Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

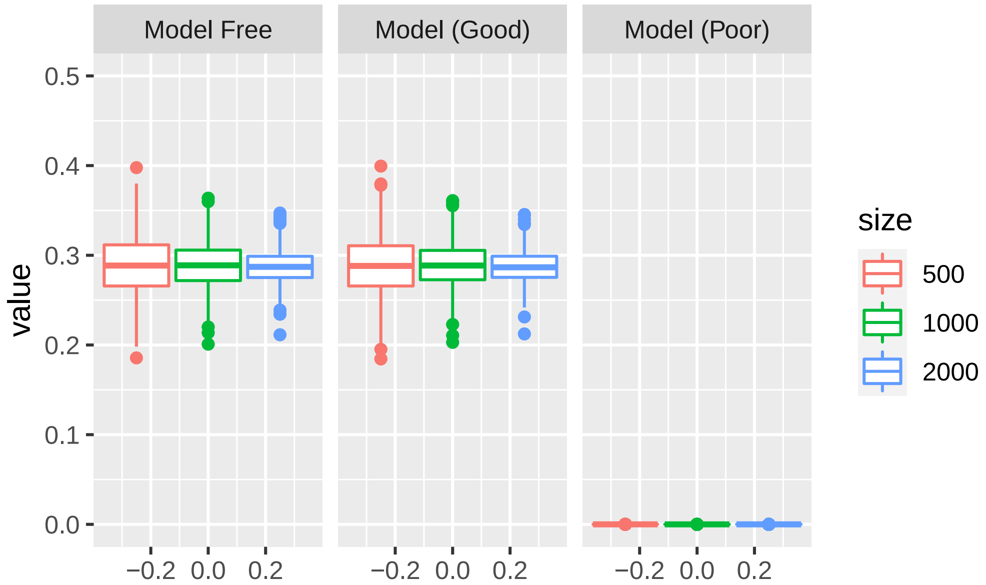

Figure B.38: Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 3×5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

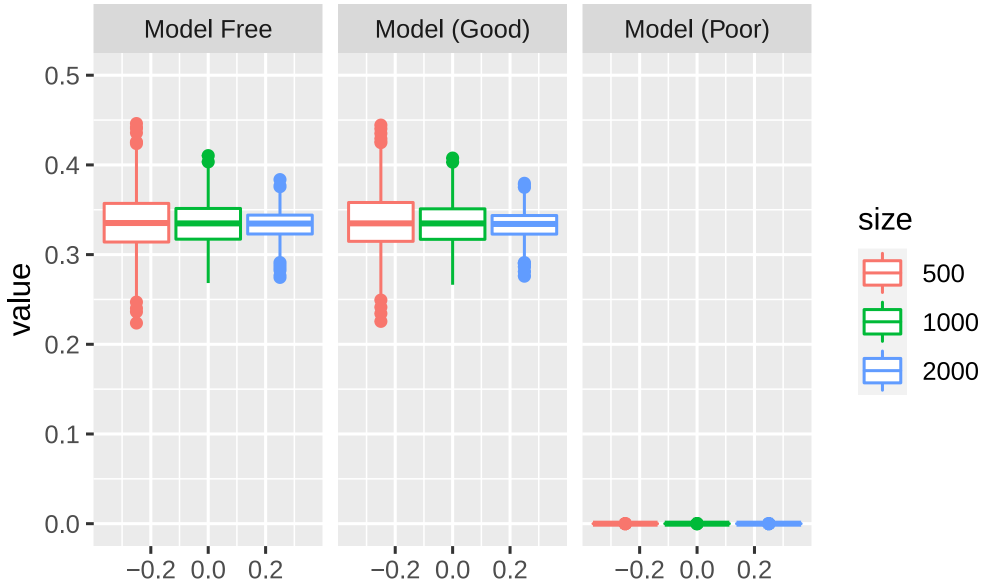

Figure B.39: Strong Association. Boxplots of $\hat{\rho}^2_{X\to Y}$ for 5×3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

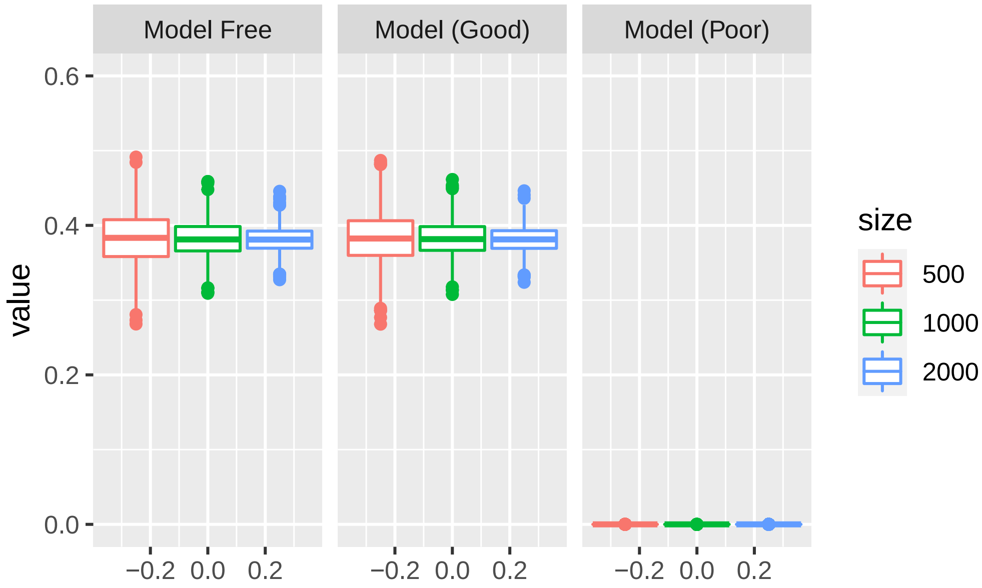

Figure B.40: Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

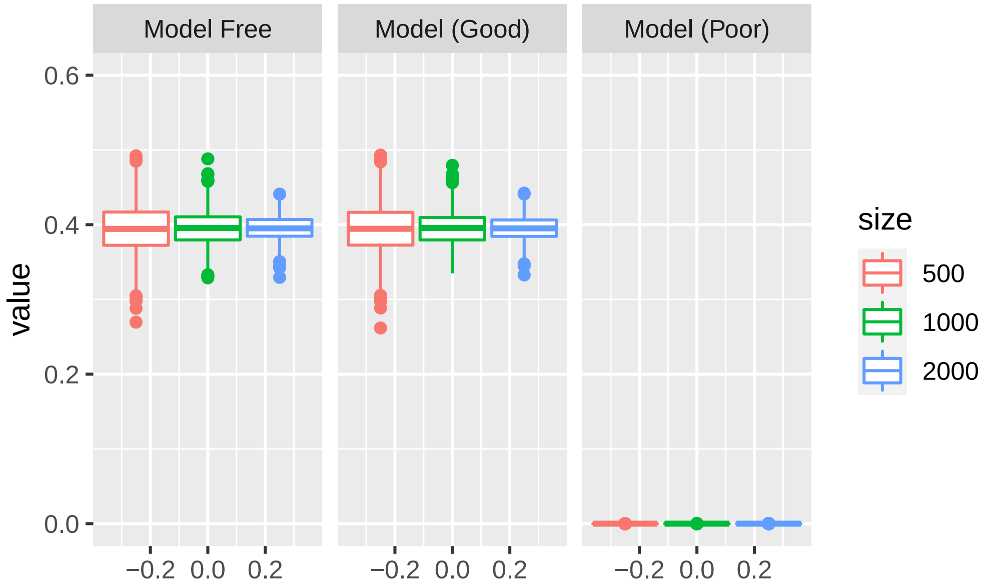

Figure B.41: Very Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

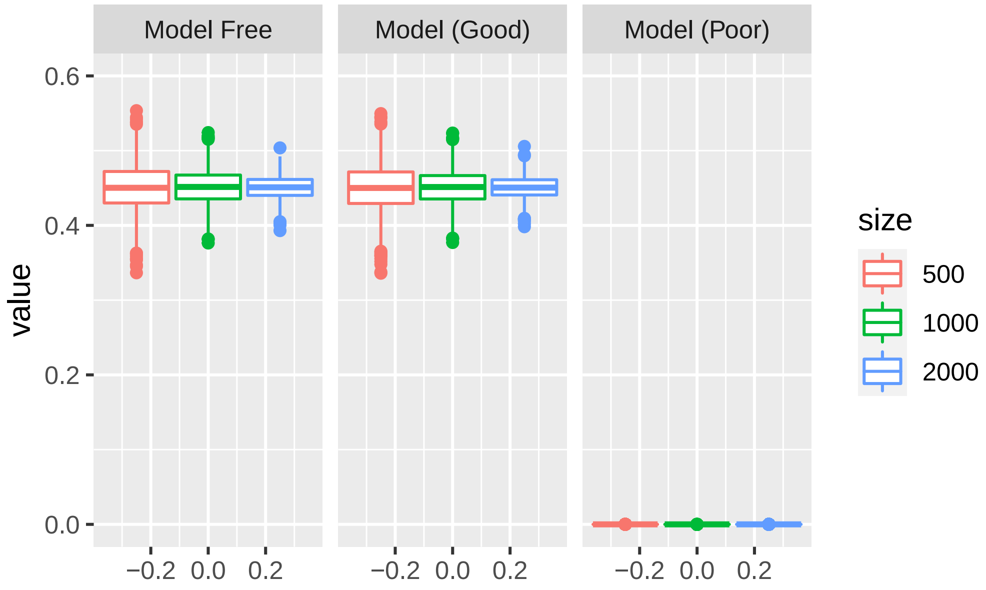

Figure B.42: Very Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 3 × 5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

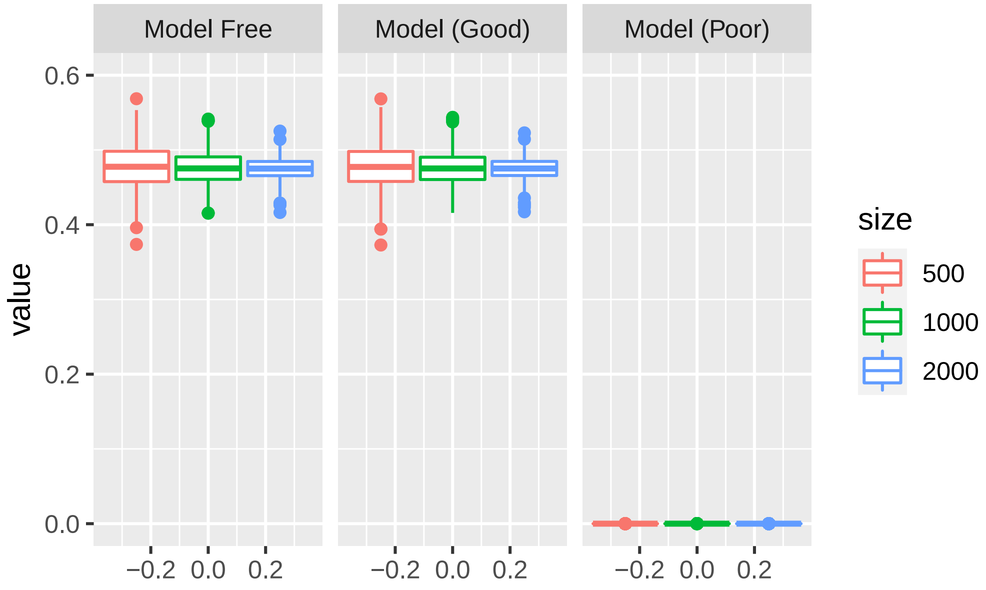

Figure B.43: Very Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5 × 3 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

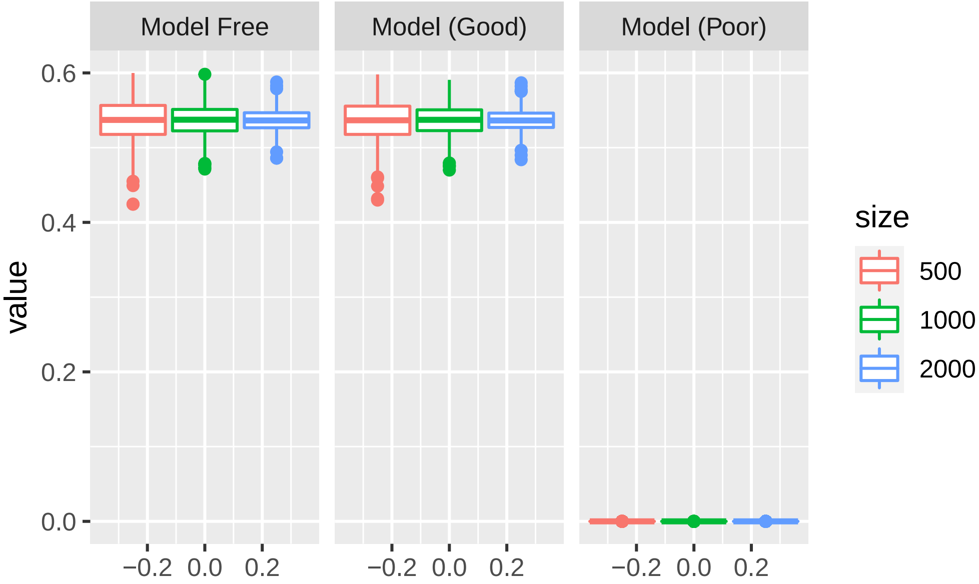

Figure B.44: Very Strong Association. Boxplots of $\hat{\rho}^2_{X \to Y}$ for 5×5 tables. Data were simulated from cumulative logit model with nominal explanatory variable X.

Corrections

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.

Abstract, pg i.

“Categorical data analysis with ordinal responses is important in fields such as the social sciences and taking into consideration the intrinsic ordering of ordinal variables can give more powerful inferences” has been changed to “Categorical data analysis with ordinal responses is important in fields such as the social sciences because when we take into consideration the intrinsic ordering of ordinal variables, we can often obtain more powerful inferences”

Page 2

The phrase “,hence the lack of modeling required” was removed.

Page 6

The sentence that included “$\cdots$ underlying variables and if we look at the right $\cdots$” has been broken up into two sentences, “$\cdots$ underlying variables. If we look at the right $\cdots$”

Page 9

“This is why copulas are so exciting” has been changed to “This flexibility is why copulas are so attractive.”

Page 10

Fixed definition of subcopula and copula. Before they were:

Definition 11. A 2-dimensional subcopula (2-subcopula) is a function $C^S : D_1 \times D_2 \to \mathbb{I}$ where $\{0,1\} \subseteq D_i \subseteq \mathbb{I}$ for $i = 1, 2$ with the following characteristics:

- Grounded, i.e., : $C^S(u, 0) = 0 = C^S(0, v)$
- $C^S(u, 1) = 1 = C^S(1, v) \;\; \forall u \in D_1, \forall v \in D_2$
- 2-increasing, i.e., : $C^S(u_2, v_2) - C^S(u_1, v_2) - C^S(u_2, v_1) + C^S(u_1, v_1) \geq 0$ where $u_1 \leq u_2$ and $v_1 \leq v_2$.

Definition 12. A 2-dimensional copula (2-copula) is a function $C : \mathbb{I} \times \mathbb{I} \to \mathbb{I}$ with the following characteristics:

- Grounded, i.e., : $C(u, 0) = 0 = C(0, v)$
- $C(u, 1) = 1 = C(1, v) \;\; \forall u \in D_1, \forall v \in D_2$
- 2-increasing, i.e., : $C(u_2, v_2) - C(u_1, v_2) - C(u_2, v_1) + C(u_1, v_1) \geq 0$ where $u_1 \leq u_2$ and $v_1 \leq v_2$.

and after:

Definition 13. A 2-dimensional subcopula (2-subcopula) is a function $C^S : D_1 \times D_2 \to \mathbb{I}$ where $\{0,1\} \subseteq D_i \subseteq \mathbb{I}$ for $i = 1, 2$ with the following characteristics:

- Grounded, i.e., : $C^S(u, 0) = 0 = C^S(0, v), \;\; \forall u \in D_1, \forall v \in D_2$
- $C^S(u, 1) = u, \;\; \forall u \in D_1$ and $C^S(1, v) = v, \;\; \forall v \in D_2$
- 2-increasing, i.e., : $C^S(u_2, v_2) - C^S(u_1, v_2) - C^S(u_2, v_1) + C^S(u_1, v_1) \geq 0$ where $u_1 \leq u_2$ and $v_1 \leq v_2$.

Definition 14. A 2-dimensional copula (2-copula) is a function $C : D_1 \times D_2 \to \mathbb{I}$ where $D_1 = \mathbb{I} = D_2$ with the following characteristics:

- Grounded, i.e., : $C(u, 0) = 0 = C(0, v), \;\; \forall u \in D_1, \forall v \in D_2$

- $C(u, 1) = u, \;\; \forall u \in D_1$ and $C(1, v) = v, \;\; \forall v \in D_2$

- 2-increasing, i.e., : $C(u_2, v_2) - C(u_1, v_2) - C(u_2, v_1) + C(u_1, v_1) \geq 0$ where $u_1 \leq u_2$ and $v_1 \leq v_2$.

Page 17

"The wireframe and contour plots of W and M are demonstrated in $\cdots$" has been changed to "The wireframe and contour plots of W and M are displayed in $\cdots$"

Page 28

Moved paragraph beginning with "One may interpret $\cdots$" directly under Definition 5 (Spearman's Rho).

Page 30

Added "Hofert, Kojadinovic, Martin, & Yan (2018) goes more in depth for those that are interested in the various estimation techniques." after the sentence, "We will briefly introduce some methods below, but won't go into too much detail."

Page 39

Added "In the previous example, Kendall's tau and Spearman's rho now both rely on p and q which are the marginal probabilities of $P(X = 0)$ and $P(Y = 0)$ respectively" to better explain the previous example.

Page 45

The sentence with the phrase "they used a certain kind of copula called the checkerboard copula" was changed to "they used a copula called the checkerboard copula."

Page 53

The phrase "see Appendix of $\cdots$" has been changed to "see the Appendix of $\cdots$"

Page 70

The sentence “For magnitude of association, we considered no association, weak, moderate, strong, very strong association” was changed to “For magnitude of association, we considered the levels of no association, weak, moderate, strong, very strong association.”

Additionally, 45 grammatical, typography, and formatting changes were corrected in various places in this thesis. Figure captions were edited formatting wise in approximately 5 places. Notation for standard Uniform distributions were changed in approximately 5 places.

References

Agresti, A. (2010). *Analysis of ordinal categorical data*. Wiley.

Agresti, A. (2019). *An introduction to categorical data analysis*. Wiley.

Brant, R. (1990). Assessing proportionality in the proportional odds model for ordinal logistic regression. *Biometrics*, *46*(4), 1171. http://doi.org/10.2307/2532457

Bross, I. D. J. (1958). How to use ridit analysis. *Biometrics*, *14*(1), 18. http://doi.org/10.2307/2527727

Erdely, A. (2016). A subcopula based dependence measure. *Kybernetika*, *53*. http://doi.org/10.14736/kyb-2017-2-0231

Faugeras, O. P. (2017). Inference for copula modeling of discrete data: A cautionary tale and some facts. *Dependence Modeling*, *5*(1), 121–132. http://doi.org/doi:10.1515/demo-2017-0008

Fréchet, M. (1951). Sur les tableaux de corrélations dont les marges sont données, 53–77.

Fullerton, A. S., & Anderson, K. F. (2021). Ordered regression models: A tutorial. *Prevention Science*. http://doi.org/10.1007/s11121-021-01302-y

Geenens, G. (2020). Copula modeling for discrete random vectors. *Dependence Modeling*, *8*(1), 417–440. http://doi.org/doi:10.1515/demo-2020-0022

Genest, C., & Favre, A.-C. (2007). Everything you always wanted to know about copula modeling but were afraid to ask. *Journal of Hydrologic Engineering - J HYDROL ENG*, *12*. http://doi.org/10.1061/(ASCE)1084-0699(2007)12:4(347)

Genest, C., & Nešlehová, J. (2007). A primer on copulas for count data. *ASTIN Bulletin*, *37*(2), 475–515. http://doi.org/10.1017/S0515036100014963

Genest, C., Nešlehová, J. G., & Rémillard, B. (2014). On the empirical multilinear copula process for count data. *Bernoulli*, *20*(3). http://doi.org/10.3150/13-bej524

Genest, C., Nešlehová, J. G., & Rémillard, B. (2017). Asymptotic behavior of the empirical multilinear copula process under broad conditions. *Journal of Multivariate Analysis*, *159*, 82–110. http://doi.org/10.1016/j.jmva.2017.04.002

Genz, A., Bretz, F., Miwa, T., Mi, X., Leisch, F., Scheipl, F., & Hothorn, T. (2021). *mvtnorm: Multivariate normal and t distributions*. Retrieved from https://CRAN.R-project.org/package=mvtnorm

Hoeffding, W. (1940). Massstabinvariante korrelationstheorie, 181–233.

Hofert, M., Kojadinovic, I., Martin, M., & Yan, J. (2018). *Elements of copula modeling with r*. Springer.

Nelsen, R. B. (2006). *An introduction to copulas (springer series in statistics)*. Berlin, Heidelberg: Springer-Verlag.

Nešlehová, J. (2007). On rank correlation measures for non-continuous random variables. *Journal of Multivariate Analysis*, *98*(3), 544–567. http://doi.org/10.1016/j.jmva.2005.11.007

Rüschendorf, L. (2009). On the distributional transform, sklar's theorem, and the empirical copula process. *Journal of Statistical Planning and Inference*, *139*(11), 3921–3927. http://doi.org/https://doi.org/10.1016/j.jspi.2009.05.030

Schlegel, B., & Steenbergen, M. (2020). *Brant: Test for parallel regression assumption*. Retrieved from https://CRAN.R-project.org/package=brant

Sklar, A. (1959). Fonctions de répartition à n dimensions et leurs marges, 229–231.

Tasena, S. (2021). On subcopula estimation for discrete models. *Asian Journal of Economics and Banking*, *5*(2), 102–110. http://doi.org/10.1108/ajeb-04-2021-0052

Trivedi, P., & Zimmer, D. (2017). A note on identification of bivariate copulas for discrete count data. *Econometrics*, *5*(1), 10. http://doi.org/10.3390/econometrics5010010

Wei, Z., & 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. http://doi.org/https://doi.org/10.1016/j.jmva.2021.104793