PAsso: an R Package for Assessing Partial Association between Ordinal Variables

by Shaobo Li, Xiaorui Zhu, Yuejie Chen, Dungang Liu

Abstract Partial association, the dependency between variables after adjusting for a set of covariates, is an important statistical notion for scientific research. However, if the variables of interest are ordered categorical data, the development of statistical methods and software for assessing their partial association is limited. Following the framework established by Liu et al. (2021), we develop an R package **PAsso** for assessing **P**artial **Asso**ciations between ordinal variables. The package provides various functions that allow users to perform a wide spectrum of assessments including quantification, visualization and hypothesis testing. In this paper, we discuss the implementation of **PAsso** in detail, and demonstrate its utility through an analysis of the 2016 American National Election Study.

Introduction

Partial association analysis plays an important role in scientific research. It uncovers the dependency between variables after adjusting for a set of covariates, which are often considered as potential confounding factors. For continuous data, a conventional approach to assessing partial association consists of two steps: (1) regress each variable on the same set of covariates using linear regression, and (2) inspect the association between the residuals from each regression model. However, it remains challenging when the data of interest are recorded in ordinal scales. One of the challenges is that the regression models for ordinal data, such as the cumulative link models (McCullagh, 1980), do not have well-defined residuals that maintain the same properties as those from linear regression. Simply treating ordinal data as continuous and applying the conventional approach may lead to misleading results (Agresti, 2010). To this end, Liu et al. (2021) proposed to use the surrogate residuals (Liu and Zhang, 2018), a type of residual developed for ordinal regression, to assess partial associations between ordinal data. They developed a unified framework allowing quantification, hypothesis testing and visualization of partial association. Their proposed methods can capture linear, monotonic and non-monotonic associations. To make their methods readily and widely applicable in practice, we develop the R package **PAsso** (**P**artial **Asso**ciation). The goal of this paper is to introduce this package in both implementation and utility.

Consider a pair of ordinal variables $Y_1 = \{1, ..., J_1\}$ and $Y_2 = \{1, ..., J_2\}$, where the recorded values represent labels of ordered categories. Let $\{X_1, ..., X_p\}$ be a set of covariates to be adjusted for. We consider such a bivariate scenario in this paper unless indicated otherwise. To assess the partial association between Y_1 and Y_2, Liu et al. (2021) proposed to assess the association between their corresponding surrogate residuals (Liu and Zhang, 2018). It mimics the conventional approach as if Y_1 and Y_2 were continuous variables. Specifically, they first apply ordinal regression models, such as the cumulative link model, to Y_1 and Y_2 separately, and derive corresponding surrogate residuals R_1 and R_2. Then, assessing the partial association between Y_1 and Y_2 is equivalent to assessing the association between R_1 and R_2. The validity of this approach is supported by the key result in Liu et al. (2021), which shows that the independence between the surrogate residuals R_1 and R_2 is a sufficient and necessary condition for the partial independence between the ordinal variables Y_1 and Y_2. We provide a more detailed review of their framework subsequently in this paper.

The R package **PAsso** provides three types of association measures discussed in Liu et al. (2021). They are Pearson-correlation-based measure ϕ_ρ, Kendall-tau-based measure ϕ_τ, and a copula-based measure, Schweizer-Wolff's sigma-based ϕ_σ. Specifically,

$$\phi_\rho = \rho(R_1, R_2) = \mathrm{Cov}(R_1, R_2)/\sqrt{\mathrm{Var}(R_1)\mathrm{Var}(R_2)}; \tag{1}$$

$$\phi_\tau = \tau(R_1, R_2) = \Pr\{(R_1 - R_1^*)(R_2 - R_2^*) > 0\} - \Pr\{(R_1 - R_1^*)(R_2 - R_2^*) < 0\}; \tag{2}$$

$$\phi_\sigma = 12 \iint_{[0,1]^2} |C(u,v) - uv| du dv, \quad \textit{where } C(u,v) = \Pr\{G_1(R_1) \le u, G_2(R_2) \le v\}. \tag{3}$$

These measures are proposed in order to capture linear, monotonic and non-monotonic relationships, respectively. For the purpose of comparison, the package also computes the Pearson correlation coefficient, Kendall's tau coefficient (Kendall, 1938) and Schweizer-Wolff's sigma (Schweizer and Wolff, 1981), for marginal associations between Y_1 and Y_2. This is convenient and useful as the marginal association is often firstly examined and set to be compared with the partial association. Furthermore, the package allows analyses for multiple variables beyond the bivariate case, so that it delivers an association matrix as one of the key outputs.

In addition, bootstrap-based standard errors and p-values are reported in order to test partial independence, i.e., $H_0 : \phi = 0$, where ϕ is a general notation for the measure of partial association, including ϕ_ρ, ϕ_τ and ϕ_σ as defined in (1)-(3). The package also provides additional flexibility to users who may wish to test whether the partial dependence is negligible at certain threshold δ, i.e., $H_0 : |\phi| \leq \delta$ instead of $H_0 : \phi = 0$. Besides quantitative analysis outcomes, the package provides graphical tools including partial regression plots and 3-D probability-to-probability plots (P-P plot). These graphical tools visualize the shape and strength of partial association, enabling further analysis that may help to gain extra insights. It is worth ~~to~~ noting that the package **PAsso** also provides model diagnostic tools (Liu and Zhang, 2018), which is similar to those in the R package **sure** (Greenwell et al., 2018).

Although the focus of this paper is on ordinal data, the package **PAsso** can also deal with binary data. In fact, binary outcome is a special case of ordered categorical data with two categories. The commonly used regression models are binary probit or logit model, for which the definition of surrogate residual remains the same. In the rest of this paper, we first review the framework established by Liu et al. (2021). Then, we provide an overview of the package **PAsso**, following which we demonstrate its utility with a real data analysis of the 2016 American National Election Study (ANES).

Review of Liu et al. (2021)'s methodology

To assess the partial association between ordinal variables, Liu et al. (2021)'s framework uses the surrogate residual (Liu and Zhang, 2018) as a key tool. Consider the cumulative link model

$$G^{-1}(P(Y \leq j)) = \alpha_j - \boldsymbol{X}\boldsymbol{\beta}, \quad j = 1, 2, ..., J, \tag{4}$$

where the intercepts satisfy $-\infty = \alpha_0 < \alpha_1 < \alpha_2 < \cdots < \alpha_J = +\infty$, and $G^{-1}(\cdot)$ is a pre-specified link function. For example, $G^{-1}(u) = \Phi^{-1}(u)$ results in the ordered probit model, and $G^{-1}(u) = \log(\frac{u}{1-u})$ results in the ordered logit model, also known as proportional odds model. Model (4) can be thought of as arising from a latent variable model

$$Z = f(\boldsymbol{X}, \boldsymbol{\beta}) + \epsilon, \tag{5}$$

where $\epsilon \sim G(\cdot)$, and Z is the latent continuous variable such that $Y = j$ if $Z \in (\alpha_{j-1}, \alpha_j]$. Given the observed category of Y, Liu and Zhang (2018) defined a surrogate variable S as

$$S|(Y = j) \sim Z|(\alpha_{j-1} < Z \leq \alpha_j). \tag{6}$$

On the continuous scale of S, Liu and Zhang (2018) defined the surrogate residual R as $R = S - E(S|\boldsymbol{X})$. The statistical properties of the surrogate residual R are similar to the classical residuals defined for linear regression models. The notion also applies to other general ordinal regression models. We refer readers to Liu and Zhang (2018) for more details of the surrogate residuals.

To assess the partial association between ordinal variables Y_1 and Y_2, Liu et al. (2021) proposed to assess the association between their corresponding surrogate residuals R_1 and R_2 from ordinal regression models. This approach mimics the conventional way of assessing the partial association between continuous variables, and has been justified by their key result. Specifically, conditional on a set of covariates $\boldsymbol{X} = \{X_1, ..., X_p\}$, Y_1 and Y_2 are independent if and only if their surrogate residuals R_1 and R_2 are independent. That is,

$$(Y_1 \perp\!\!\!\perp Y_2) \mid \boldsymbol{X} \Leftrightarrow R_1 \perp\!\!\!\perp R_2 \mid \boldsymbol{X}.$$

Based on this result, Liu et al. (2021) developed a new framework for assessing ordinal-ordinal partial association. The framework includes a set of new methods to quantify, visualize and test the association.

As for quantification, Liu et al. (2021) justified the advantages of using their proposed measure ϕ. First, the measure ϕ reflects the association between ordinal variables, rather than that between latent continuous variables. It does not constrain itself to probit models, but broadly applies to models with non-probit link functions. Its variants (1), (2) and (3) can capture linear, monotonic and general associations. These features make the association measure ϕ fundamentally different from the polychoric correlation (Tallis, 1962), a classical association measure that describes the linear association between the latent normal variables. Moreover, their association measures do not require an upfront specification of a joint distribution of Y_1 and Y_2. Instead, their framework only requires the marginal model for each Y to be correctly specified. It thus achieved the so-called "division of labor," where the efforts of specifying marginal models and association structure are divided. This property has been seen in the development of generalized estimating equations (GEEs) and copula models. It enables

us to compute the correlation matrix for a high-dimensional set of ordinal variables with affordable computational cost.

To reduce the variability caused by the simulation of the surrogate variable S in (6), Liu et al. (2021) suggested ~~to~~ using the average

$$\hat{\phi}^{(M)} = \frac{1}{M}\sum_{m=1}^{M}\hat{\phi}^{(m)}, \tag{7}$$

where $\hat{\phi}^{(m)}$ is calculated based on the mth draw of the surrogate residuals $(R_1^{(m)}, R_2^{(m)})$ for $m = 1, ..., M$, and all draws are independent. Specifically, after fitting the two marginal regression models, we draw M independent sets of surrogate residuals for each model, i.e., $(R_1^{(1)}, ..., R_1^{(M)})$ and $(R_2^{(1)}, ..., R_2^{(M)})$. Then $\hat{\phi}^{(m)}$ is obtained based on $(R_1^{(m)}, R_2^{(m)})$, the mth draw of surrogate residuals. They numerically found that $M = 30$ is sufficient to stabilize the variance.

Liu et al. (2021) showed how to use (7) to make inferences, such as calculating standard errors, confidence intervals and p-values. They established a bootstrap scheme to approximate the empirical distribution of $\hat{\phi}^{(M)}$ in (7). Specifically, it generates B bootstrap samples from the original data and obtains a set of bootstrap estimates $\{\hat{\phi}_1^{(M)}, \hat{\phi}_2^{(M)}, ..., \hat{\phi}_B^{(M)}\}$. Denote the empirical distribution of $\{\hat{\phi}_1^{(M)}, \hat{\phi}_2^{(M)}, ..., \hat{\phi}_B^{(M)}\}$ by $\hat{F}_B(\phi)$. This distribution approximates the distribution of $\hat{\phi}^{(M)}$. Therefore, the standard deviation of the bootstrap distribution $\hat{F}_B(\phi)$ is an estimate of standard error of $\hat{\phi}^{(M)}$. The interval $(\hat{F}_B^{-1}(\alpha/2), \hat{F}_B^{-1}(1-\alpha/2))$ can be used as a $100(1-\alpha)\%$ confidence interval. For the hypothesis testing of partial independence, i.e., $H_0 : \phi = 0$, the p-value is calculated as

$$2 \times \min(\hat{F}_B(0), 1 - \hat{F}_B(0)).$$

For example, $\hat{F}_B(0)$ is computed as $\mathbb{I}(\hat{\phi}_b^{(M)} \leq 0)/B$, where $\mathbb{I}(x)$ is the indicator function that takes value 1 if x is true and 0 otherwise. Furthermore, this approach allows testing the hypothesis whether the partial association is smaller than a threshold δ, i.e., $H_0 : |\phi| < \delta$. In this case, the p-value is calculated as $2 \times \min(\hat{F}_B(\delta), 1 - \hat{F}_B(-\delta))$. A useful application is to test the hypothesis of a negligible association, where δ usually takes a small value that can be determined by domain expert based on specific questions.

In addition to quantitative analysis, Liu et al. (2021) developed graphical tools to visualize the shape of the partial association. One of the most intuitive plots is the partial regression plot, which is a scatter plot between the surrogate residuals R_1 and R_2. Such a partial regression plot reflects the relationship between Y_1 and Y_2 after adjusting for $\boldsymbol{X}$. Another graphical tool is a 3-D probability-to-probability (P-P) plot. It is developed based on Theorem 2 and Corollary 2 in Liu et al. (2021). Specifically, if R_1 and R_2 are independent, then

$$C(u, v) = uv, \tag{8}$$

where $C(u, v)$ is a copula function for the joint distribution of the surrogate residuals R_1 and R_2, i.e., $C(u, v) = \Pr(R_1 \leq G_1^{-1}(u), R_2 \leq G_2^{-1}(v))$ where G_1 and G_2 are the assumed link functions in model (4). The 3-D P-P plot draws $C(u, v) - uv$ against (u, v), where the deviation $C(u, v) - uv$ reflects the degree of dependence between R_1 and R_2, hence the partial dependence between Y_1 and Y_2.

In addition to the cumulative link model (4), Liu et al. (2021)'s framework applies to general ordinal regression models such as the adjacent-category logit and stereotype models. In the following sections, we introduce the **PAsso** package and discuss how to use it to implement Liu et al. (2021)'s framework with sample code and examples.

Partial association analysis with R package PAsso

An overview

Among the exported functions from the **PAsso** package, `PAsso()`, `test()` and `plot()` are the three main functions for estimating, testing and visualizing partial associations. In practice, users should first apply the function `PAsso()`, which generates a `PAsso` object. The components in this object include estimates of partial and marginal associations, fitted regression models for each variable and other components to be used for relevant analysis. The generated `PAsso` object plays an instrumental role, as it is a necessary input for other functions in the package. This design provides convenience as the details of data, regression models and type of association measures need only to be specified once in the function `PAsso()`, and they will be passed to other functions once the `PAsso` object is called.

In addition to the main functions, the **PAsso** package also provides several other functions that can further assist the analysis of partial association. Specifically, the function `plot3D()` produces the copula-based 3-D P-P plot, which helps to visualize the strength of a general partial association. The function `residuals()` can extract the surrogate residuals from a `PAsso` object or a model object. The function `diagnostic.plot()` generates residual-based model diagnostic plots to help validate the model specifications.

We provide in Figure 1 a flow chart to illustrate how to use the **PAsso** package for assessing partial association under a bivariate setting. On the far left, the grey block contains elements to input: the data and the models. By calling the function `PAsso()`, we generate a `PAsso` object as shown in the orange block in the center. This `PAsso` object can yield results for estimation, testing and visualization (the three orange blocks on the far right). The `PAsso` object can also give us residuals, which can be used for model diagnostics (blue blocks). In fact, the model diagnostic plots are byproducts of the package. To make it more convenient, we provide the function `diagnostic.plot()` that can be directly applied to the `PAsso` object.

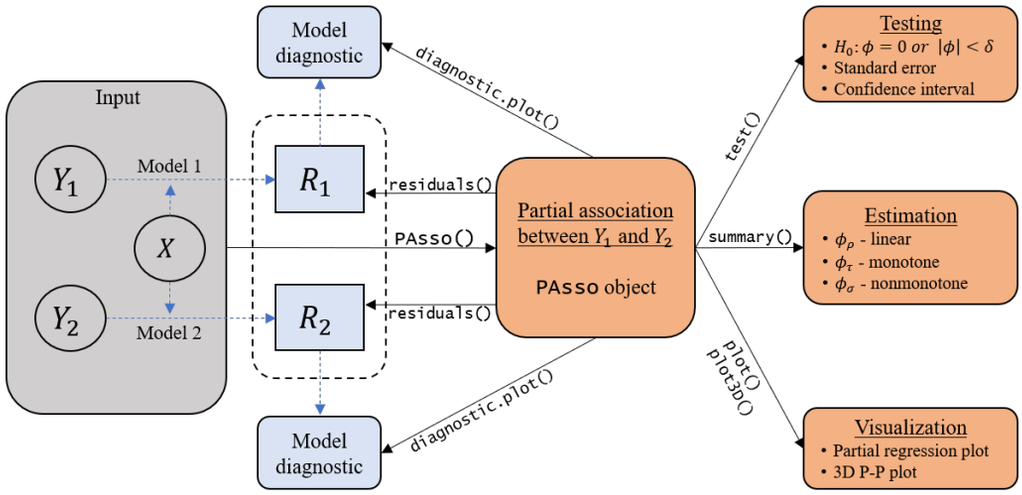

Figure 1: Illustration of functions in the **PAsso** package for partial association analysis.

Figure 1 clearly shows that the `PAsso` object plays a central role in the entire analysis, where all the other functions take it as an input. Therefore, understanding the `PAsso()` function and specifying appropriate inputs are important, which ensures valid results produced by other functions in the subsequent analysis. Next, we describe the detailed implementation and the key inputs of the function `PAsso()`, as well as other functions in the package.

Implementation of main functions

`PAsso()` for estimation

To apply the function `PAsso()`, users need to specify the dataframe, the names of Y variables (e.g., Y_1 and Y_2) and covariates, and the type of association measure. Other inputs, including the type of model and residual method, have default values. The first step performed by `PAsso()` is to estimate the regression model for each Y variable. Based on the data type, by default the function `PAsso()` applies the cumulative probit model as in (4) (`polr()` from **MASS** (Venables and Ripley, 2002)) if Y is ordinal and the binary probit model (`glm()` from **stats**) if Y is binary. Users can also specify other link functions supported by `polr()` and `glm()`.

Once the regression models for Y_1 and Y_2 are estimated, the surrogate residuals, R_1 and R_2, can be obtained by using the function `residuals()` in the package. Here the function `residuals()` calls the fitted model based on which the surrogate residuals are computed. The implementation of `residuals()` is similar to the `resids()` function from package **sure**. Then, by applying the specified measure to the surrogate residuals, one can obtain the estimated partial association defined in (7). In the current package version, the three types of association measures, (1), (2) and (3), are implemented with `cor()` from the **stats** package, `cor.fk()` from the **pcaPP** package (Filzmoser et al., 2018) and `wolfCOP()` from the **copBasic** package (Asquith, 2020), respectively.

It is worth ~~to~~ mention that the function `PAsso()` provides additional flexibility by allowing pre-fitted model objects to be the inputs. It permits users to use specific models based on domain knowledge and experiences. Similarly, the pre-fitted model objects can also be the inputs for the

functions `residuals()` and `diagnostic.plot()`. The supported models and corresponding R functions along with the packages (**rms** (Harrell Jr, 2019), **ordinal** (Christensen, 2019), **VGAM** (Yee et al., 2010)) are listed in Table 1.

Table 1: Models and packages supported by **PAsso**.

		stats	MASS	rms		ordinal	VGAM	
Response	Model	`glm`	`polr`	`lrm`	`orm`	`clm`	`vglm`	`vgam`
Ordinal	Ordered probit		✓		✓	✓	✓	✓
	Ordered logit (proportional odds)		✓		✓	✓	✓	✓
	Proportional hazard (clog-log)		✓		✓	✓	✓	✓
	Adjacent-category logit						✓	✓
Binary	Logit	✓		✓			✓	✓
	Probit	✓					✓	✓
	Complementary log-log	✓					✓	✓

Table's caption should be placed below the table.

`test()` for hypothesis testing

To carry out more detailed inferences on the estimated partial association, one can apply the function `test()`, which is implemented based on a bootstrap scheme discussed earlier. To be specific, it first generates bootstrap samples based on the original data, and then repeats the calculations performed in the `PAsso()` function for each bootstrap sample. As a result, the bootstrap distribution of $\hat{\phi}^{(M)}$ defined in (7) can be obtained.

The components from `test()` include standard errors, confidence intervals and p-values for partial independence test. To apply `test()`, in addition to the key input `PAsso` object, user can specify the number of bootstrap samples, the type of null hypothesis and whether or not the parallel computing should be employed. If the Schweizer-Wolff-sigma-based measure ϕ_σ is used, the computational cost of `test()` may be high. This is because the estimation of ϕ_σ based on the current version of the package **copBasic** is slow, due to the double integration in the formula (3).

`plot()` and `plot3D()` for visualization

There are two types of plots being covered in the **PAsso** package: the pairwise partial regression plot and the 3-D P-P plot. The pairwise partial regression plot can be obtained by applying the function `plot()`. It is implemented based on the function `ggpairs()` in the package **GGally** (Schloerke et al., 2020), which is a nice extension of the widely used graphical tool **ggplot2** (Wickham, 2016). For users who are familiar with **GGally**, the produced figure can be fully customized using the arguments for the function `ggpairs()`.

The 3-D P-P plot is obtained by using the function `plot3D()`. This function is implemented based on the **plotly** (Sievert, 2020) package, which offers interactive graphs. It plots $C(u, v) - uv$ against u and v as discussed in (8). The height of the surface reflects the degree of partial dependence between Y_1 and Y_2.

Table 2 summarizes the main input and output of the functions in the **PAsso** package. Users should refer to the package vignette for detailed arguments and output values.

Analysis of the 2016 American National Election Study (ANES)

We demonstrate the utility of the **PAsso** package with a real data example, a sample of the survey data from the 2016 American National Election Studies (the data of Time Series study). The American National Election Studies (ANES) is a joint project between the University of Michigan and Stanford University since 1948 (DeBell, 2010), aiming to provide researchers, policy makers and citizens with high-quality survey data pertinent to political science. The original data include over a thousand survey questions for pre- and post-election surveys based on the same population (4,271 respondents). For our analysis, we select eight survey questions in the pre-election study and remove the respondents who have missing values on any of these eight variables. Here we assume that the missing values are missing completely at random. As a result, our sample consists of 2,188 respondents. Table 3 describes the variables in our sample.

To demonstrate the utility of the **PAsso** package, we conduct an illustrative analysis, which focuses on the four variables: voter's party identification (PID), his/her own left-right placement (selfLR) and the left-right placement for Donald Trump (TrumpLR) and Hilary Clinton (ClinLR). These four

Table 2: Summary of inputs and outputs of the functions in **PAsso**.

Function name	Functionality	Input	Output
`PAsso()`	Estimation	• dataframe to be analyzed; • names of the Y variables; • names of the covariates; • association measure; • other inputs such as marginal model specifications have default values (see the package vignette)	• partial and marginal association matrix; • summary of model coefficient estimates for each marginal model; • results also include surrogate residuals and detailed model summaries
`test()`	Testing	• the `PAsso` object; • number of bootstrap samples; • type of null hypothesis	• Partial association matrix; • standard errors; • p-values; • confidence intervals
`plot()`	Visualization	• the `PAsso` object; • other arguments follow the **GGally** (see the package vignette)	• Pairwise partial regression plot
`plot3D()`	Visualization	• the `PAsso` object; • two variable names; • other arguments follow the **plotly** (see the package vignette)	• 3-D P-P plot based on Copula
`residuals()`	Residuals	• the `PAsso` object or a single model object; • number of draws; • residual method	• surrogate residuals
`diagnostic.plot()`	Diagnostics	• the `PAsso` object or a single model object	• model diagnostic plots

Place the caption here.

Table 3: Variable description for the selected sample of the 2016 ANES data.

Variable	Explanation
age	respondent's age
education	respondent's highest level of education. We create another variable 'edu.year', which recodes the original education to a continuous scale as the approximate number of years of education. (This conversion is for convenience in model fitting.)
income	respondent's annual income in categories, e.g., \$40k-\$60k. We create another variable 'income.num', which recodes original income to continuous scale (the median of the recorded range)
PID	respondent's party identification with 7 ordinal levels from strong democrat (=1) to strong republication (=7)
selfLR	respondent's self-placement about own left-right placement in 7 ordinal levels from extremely liberal (=1) to extremely conservative (=7)
TrumpLR	respondent's opinion about Donald Trump's left-right placement in 7 ordinal levels (same scale as selfLR)
ClinLR	respondent's opinion about Hilary Clinton's left-right placement in 7 ordinal levels (same scale as selfLR)
PreVote	respondent's voting preference between Donald Trump and Hilary Clinton

Place the caption here.

variables of interest have the same ordinal scale, and they all represent respondents' political views for themselves as well as for the two presidential candidates. We attempt to answer the following question. Are these four variables still correlated after adjusting for their age, education and income? In what follows, we use the **PAsso** package to answer this question step-by-step from three aspects:

estimation, hypothesis testing and graphical visualization.

Estimation

First, we use the function `PAsso()` to obtain the estimates of the association measures in (1)-(3). We specify the names of these four variables as `responses` and the covariate names as the `adjustments`. If the Kendall-tau-based measure (2) is desired, we can specify the option `method="kendall"`, as in the sample code below.

```
library(PAsso)
data(ANES2016)   # load the dataset
set.seed(2020)  # for reproducibility
phi <- PAsso(responses = c("PID", "selfLR", "TrumpLR", "ClinLR"),
             adjustments = c("age", "edu.year", "income.num"),
             data = ANES2016,
             method = "kendall")
print(phi, digits = 3)

#> -----------------------------------------
#> The partial correlation coefficient matrix:
#>          PID     selfLR  TrumpLR  ClinLR
#> PID       1.000   0.516  -0.061   -0.323
#> selfLR            1.000  -0.103   -0.294
#> TrumpLR                   1.000   -0.072
#> ClinLR                             1.000
```

As shown above, the default output is pairwise correlations in a matrix form. The `PAsso` object contains other hidden components, including the marginal association matrix, details of each regression model and all draws of the surrogate residuals. For users' convenience, we also make the generic function `summary()` available for the `PAsso` object.

```
summary(phi, digits = 4)

#> -----------------------------------------
#> The partial correlation coefficient matrix:
#>
#>          PID      selfLR   TrumpLR  ClinLR
#> PID       1.0000   0.5161  -0.0610  -0.3232
#> selfLR             1.0000  -0.1034  -0.2938
#> TrumpLR                     1.0000  -0.0715
#> ClinLR                              1.0000
#> -----------------------------------------
#> The marginal correlation coefficient matrix:
#>
#>          PID      selfLR   TrumpLR  ClinLR
#> PID       1.0000   0.6331  -0.0956  -0.4081
#> selfLR             1.0000  -0.1592  -0.3724
#> TrumpLR                     1.0000  -0.0721
#> ClinLR                              1.0000
#>
#> -----------------------------------------
#> -----------------------------------------
#>
#> The coefficients of fitted models are:
#>
#>             PID         selfLR      TrumpLR     ClinLR
#> age          0.0048***   0.0098***  -0.0056***  -0.0069***
#> Std. Error   0.0013      0.0013      0.0013      0.0013
#> ---
#> edu.year    -0.0459***  -0.0737***   0.0540***  -0.0127
#> Std. Error   0.0098      0.0095      0.0096      0.0097
#> ---
#> income.num   0.0009*     0.0004      0.0005     -0.0009*
#> Std. Error   0.0004      0.0004      0.0004      0.0004
```

```
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The function `summary()` gives the partial and marginal association in two separate matrices and summary of coefficient estimates of each regression model. The comparison between the two association matrices shows the extent to which the strength of association has been weakened due to the adjusted covariates. Clearly, the associations between all pairs of the four variables have been weakened by as much as 36% (PID vs. TrumpLR) and as less as 0.8% (TrumpLR vs. ClinLR). The last table is the model summary table, where each column displays the coefficient estimates and their standard errors for each regression model, and the column names are the names of response variables (e.g., PID, selfLR, TrumpLR and ClinLR). The stars represent the range of p-values as noted at the bottom of the model summary table. For instance, `'***'` indicates that the corresponding p-value is ~~in~~ between 0 and 0.001.

Hypothesis testing

Next, we use the function `test()` to obtain the p-values and bootstrap standard errors of the partial associations. This function is directly applied to the `PAsso` object. The number of bootstrap replicates can be specified with the argument `bootstrap_rep`, whose default value is 300. In general, we recommend the minimum number of bootstrap replicates to be 1000 in order to carry out more accurate p-values. To speed up the computational time, we can further adopt ~~the~~ parallel computing by specifying `parallel=TRUE`. However, this option requires users to pre-set the number of cores and the cluster as demonstrated below.

```
library(doParallel) # load packages for parallel computing
library(progress)
numCores <- detectCores()  # Number of CPU cores. Do Not be too aggressive!
# Set up parallel backend (multicore for unix and snow for windows)
cl <- if (.Platform$OS.type == "unix") numCores else makeCluster(numCores)
registerDoParallel(cl)
test(phi, bootstrap_rep = 1000, H0 = 0, parallel = TRUE)

# The partial association analysis:
#
#          PID      selfLR    TrumpLR   ClinLR
# PID       1.0000   0.5161   -0.0610   -0.3232
# S.E.               0.0094    0.0134    0.0102
# Pr                 0.001***  0.001***  0.001***
# ---
# selfLR             1.0000   -0.1034   -0.2938
# S.E.                        0.0129    0.0108
# Pr                          0.001***  0.001***
# ---
# TrumpLR                      1.0000   -0.0715
# S.E.                                  0.0154
# Pr                                    0.001***
# ---
# ClinLR                                1.0000
# S.E.
# Pr
# ---
# Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

The obtained p-values indicate that the partial associations between each pair of variables are statistically significant. We can also test for negligible dependence with a certain threshold δ. In this case, the null hypothesis is $H_0 : |\phi| \leq \delta$. The value of δ can be specified using the argument `H0`. The default value of `H0` is 0, representing $H_0 : \phi = 0$. The example below illustrates a negligible dependence test with threshold $\delta = 0.05$.

```
test(phi, bootstrap_rep = 1000, H0 = 0.05, parallel = TRUE)

# The partial association analysis:
#
#          PID      selfLR    TrumpLR   ClinLR
# PID       1.0000   0.5161   -0.0610   -0.3232
```

```
# S.E.              0.0093    0.0135    0.0104
# Pr                0.001***  0.450     0.001***
# ---
# selfLR             1.0000   -0.1034   -0.2938
# S.E.                        0.0133    0.0109
# Pr                          0.001***  0.001***
# ---
# TrumpLR                      1.0000   -0.0715
# S.E.                                  0.0153
# Pr                                    0.128
# ---
# ClinLR                                1.0000
# S.E.
# Pr
# ---
# Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

From the output, we can see that the partial associations between PID and TrumpLR, and TrumpLR and ClinLR are not significant. This result implies that the partial dependencies can be negligible under the threshold $\delta = 0.05$.

Visualization

To further understand how these variables are associated after the adjustment, we utilize ~~the~~ graphical tools for a quick visualization. First, we apply the function `plot()` to the `PAsso` object `phi`. It produces the pairwise partial regression plot.

```
plot(phi, color = "red", alpha = 0.1) # alpha specifies opacity
```

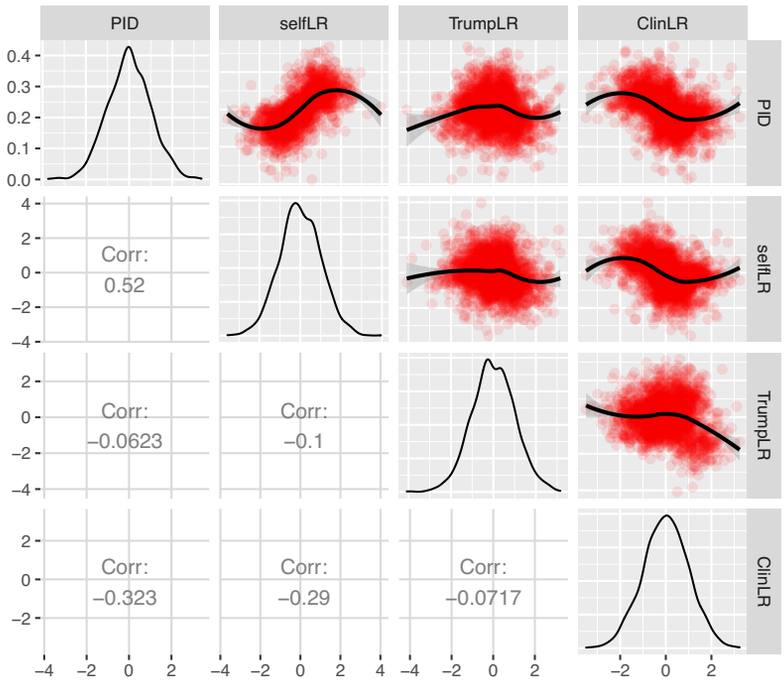

Figure 2: Pairwise partial regression plot for PID, selfLR, TrumpLR, and ClinLR.

Figure 2 contains the partial regression plots displayed on the upper triangle, the estimated partial associations shown on the lower triangle and the density plots of the surrogate residuals on the diagonal line. The partial regression plot between each pair of variables is essentially the scatter plot of the corresponding pair of surrogate residuals. The fitted smooth curve on top of each scatter plot is through the local weighted smoothing splines (LOWESS) that are available in the **GGally** package.

An interesting finding from Figure 2 is that, the weakest partial association (between PID and TrumpLR) shown previously may not necessarily be weak, because a quadratic relationship is clearly

shown. To confirm this relationship, it is worth to draw the 3-D P-P plot, which can indicate the strength of a non-monotonic relationship. This can be done by applying the function `plot3D()` on the `PAsso` object and specifying the pair of variables of interest.

```
plot3D(phi, y1 = "PID", y2 = "TrumpLR")
```

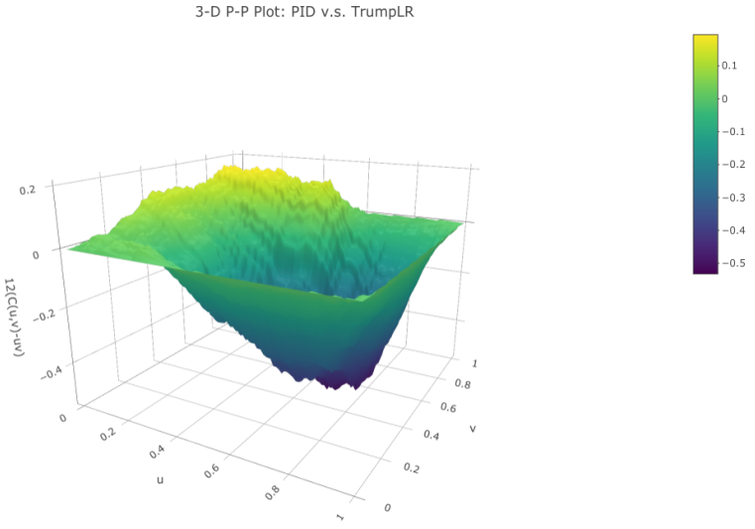

Figure 3: 3D probability-to-probability plot based on a bivariate copula.

From Figure 3, the surface reaches to both positive and negative sides (vertical axis). This is consistent with the quadratic pattern displayed in the partial regression plot in Figure 2. Furthermore, the altitude of the surface on both sides are moderate, indicating that the strength of the partial association between PID and TrumpLR may not be negligible. Additionally, the function `plot3D()` also provides an option to convert the 3-D plot to a contour plot by specifying `type = "contour"`. The following line of code creates Figure 4, showing the contour plot corresponding to Figure 3.

```
plot3D(phi, y1 = "PID", y2 = "TrumpLR", type = "contour")
```

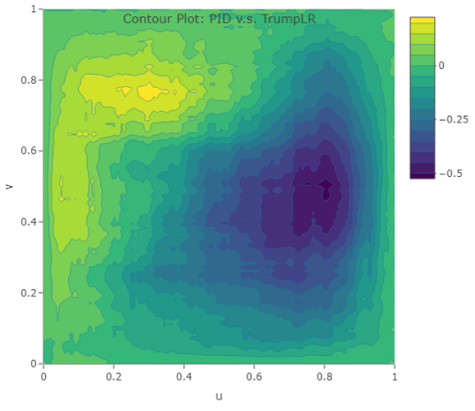

Figure 4: Colored contour plot based on a bivariate copula.

As mentioned earlier, the function `PAsso()` allows users to use pre-fitted regression models as inputs. Below we provide a sample code for this alternative use of `PAsso()`.

```
library(MASS)
# convert numeric response to factor
Yname <- c("PID", "selfLR", "TrumpLR", "ClinLR")
ANES2016[Yname] <- lapply(ANES2016[Yname], factor)
# fit each model separately
fit.PID <- polr(PID ~ age + edu.year + income.num, data = ANES2016, method = "probit")
```

```
fit.selfLR <- polr(selfLR ~ age + edu.year + income.num, data = ANES2016, method = "probit")
fit.TrumpLR <- polr(TrumpLR ~ age + edu.year + income.num, data = ANES2016, method = "probit")
fit.ClinLR <- polr(ClinLR ~ age + edu.year + income.num, data = ANES2016, method = "probit")
# assess partial association
set.seed(2020)
phi1 <- PAsso(fitted.models = list(fit.PID, fit.selfLR, fit.TrumpLR, fit.ClinLR),
              method = "kendall")
print(phi1, digits = 3)

#> -------------------------------------------
#> The partial correlation coefficient matrix:
#>          PID     selfLR  TrumpLR  ClinLR
#> PID       1.000   0.516  -0.061   -0.323
#> selfLR            1.000  -0.103   -0.294
#> TrumpLR                   1.000   -0.072
#> ClinLR                             1.000
```

The above output is exactly the same as the other way we showed earlier. Such flexibility allows user to specify different models and link functions for different response variables based on domain expertise and convention, while it does not affect other functions used in following analyses. Each of these model objects (`fit.PID`, `fit.selfLR`, `fit.TrumpLR` and `fit.ClinLR`) can also be the input for the functions `residuals()` and `diagnostic.plot()`.

Model diagnostics

In addition to assessing partial association, the package also provides the function `diagnostic.plot()` for model diagnostics. The output of this function contains three types of diagnostic plots: (1) residual Q-Q plot; (2) residual vs. fitted value; and (3) residual vs. covariates. We can specify one of these types of plot by using the argument `output`. Since there are at least two regression models being estimated, the function `diagnostic.plot()` allows user to specify a particular model (using the argument `model_id`) for which the diagnostic plots are produced. We provide two examples below to illustrate the two different outputs.

```
# residual vs. fitted value (the linear predictor) for all models
diagnostic.plot(phi, output = "fitted")
```

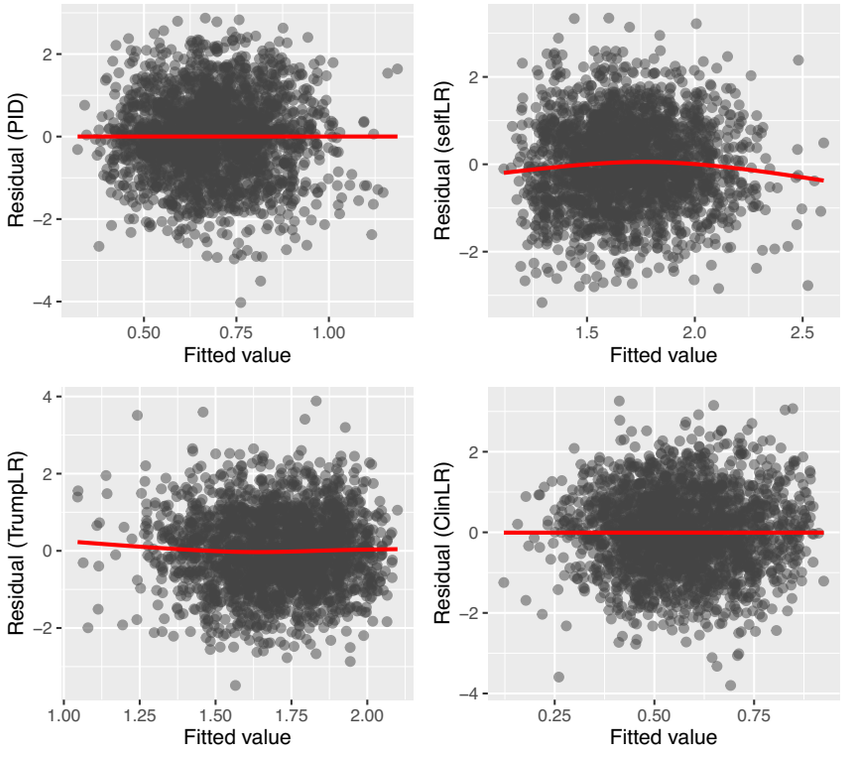

Figure 5: Model diagnostic plot (Residual vs. fitted) for all four models.

```
# residual Q-Q plot for the second model (selfLR)
diagnostic.plot(phi, output = "qq", model_id = 2)
```

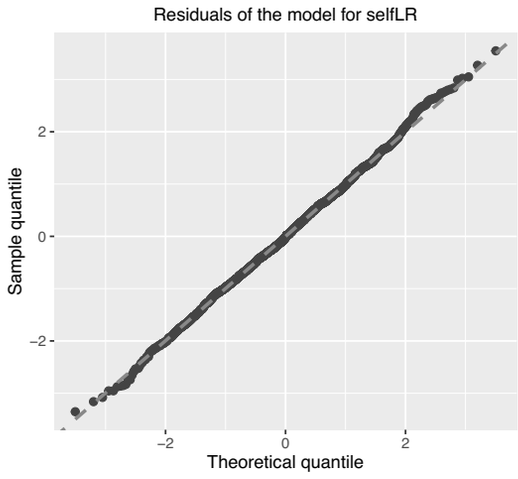

Figure 6: Model diagnostic plot (Residual Q-Q plot) for the second model.

Figure 5 shows the scatter plot between surrogate residuals and the fitted values (the linear predictor in the model (5)) for all four models. Figure 6 shows the Q-Q plot for the second model whose response variable is "selfLR." Neither of the two figures provides evidence of model misspecification or inadequate fitting.

Adjacent-category logit model

Finally, we show an example where the adjacent-category logit model is employed for the covariates adjustment. In addition, we also include a binary variable "PreVote" in order to demonstrate the utility of **PAsso** for different types of data. The variable "PreVote" takes two values: Hilary Clinton and Donald Trump. For convenience, we recode "PreVote" to numeric value 0 (Hilary Clinton) and 1 (Donald Trump), and the binary logit model is used. For the demonstration purpose, we only show the estimation results from `PAsso()`.

```
ANES2016$PID <- as.numeric(ANES2016$PID)
phi2 <- PAsso(responses = c("PreVote.num", "PID", "selfLR", "TrumpLR", "ClinLR"),
              adjustments = c("age", "edu.year", "income.num"),
              data = ANES2016,
              method = "kendall",
              model = c("logit", "acat", "acat", "acat", "acat"))
summary(phi2, digits = 3)
```

```
#> ------------------------------------------
#> The partial correlation coefficient matrix:
#>
#>              PreVote.num  PID     selfLR  TrumpLR  ClinLR
#> PreVote.num   1.000        0.445   0.390  -0.091   -0.300
#> PID                        1.000   0.516  -0.062   -0.319
#> selfLR                             1.000  -0.105   -0.293
#> TrumpLR                                    1.000   -0.071
#> ClinLR                                             1.000
#> ------------------------------------------
#> The marginal correlation coefficient matrix:
#>
#>              PreVote.num  PID     selfLR  TrumpLR  ClinLR
#> PreVote.num   1.000        0.706   0.637  -0.182   -0.494
#> PID                        1.000   0.633  -0.096   -0.408
#> selfLR                             1.000  -0.159   -0.372
#> TrumpLR                                    1.000   -0.072
#> ClinLR                                             1.000
#>
```

```
#> ------------------------------------------
#> ------------------------------------------
#>
#> The coefficients of fitted models are:
#>
#>             PreVote.num  PID        selfLR     TrumpLR    ClinLR
#> age          0.015***    -0.002***  -0.006***   0.002**   0.004***
#> Std. Error   0.003        0.001      0.001      0.001     0.001
#> ---
#> edu.year    -0.129***     0.020***   0.045***  -0.036***  0.022**
#> Std. Error   0.019        0.004      0.006      0.006     0.007
#> ---
#> income.num   0.001        0.000**    0.000     -0.001**   0.001**
#> Std. Error   0.001        0.000      0.000      0.000     0.000
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The same elements as for the cumulative link model are printed: the estimated partial and marginal associations, and the coefficient estimates of the five models. As specified in the argument `model`, the binary logit model is used for `PreVote`, and the adjacent-category logit model ("acat") is used for the other four variables. From the output, we found that the strength of association between "PreVote" and "PID" reduces from 0.706 to 0.444 ($\approx 37\%$) after adjusting for the three covariates. This result is qualitatively consistent with that in Liu et al. (2021), which has conducted the same analysis based on the 1996 ANES data.

Summary

We have developed the R package **PAsso** (Zhu et al., 2021) for assessing the partial association between ordinal variables. The development follows the statistical framework established by Liu et al. (2021). The package provides several functions, allowing estimation, hypothesis testing and visualization of partial associations. By using this single package, users can quickly obtain a full picture of the partial associations between multiple variables.

Bibliography

A. Agresti. *Analysis of Ordinal Categorical Data*. John Wiley & Sons: Hoboken, New Jersey, 2 edition, 2010. [p]

W. Asquith. *copBasic—General Bivariate Copula Theory and Many Utility Functions*, 2020. R package version 2.1.5. [p]

R. H. B. Christensen. ordinal—regression models for ordinal data, 2019. R package version 2019.12-10. https://CRAN.R-project.org/package=ordinal. [p]

M. DeBell. How to analyze anes survey data. *Am. Nat'l Election Stud.(May 2010), http://electionstudies.org/resources/papers/neso12492. pdf [http://perma. ee/Y4VG-X47F]*, 2010. [p]

P. Filzmoser, H. Fritz, and K. Kalcher. *pcaPP: Robust PCA by Projection Pursuit*, 2018. URL https://CRAN.R-project.org/package=pcaPP. R package version 1.9-73. [p]

B. M. Greenwell, A. J. McCarthy, B. C. Boehmke, and D. Liu. Residuals and diagnostics for binary and ordinal regression models: An introduction to the sure package. *The R Journal*, 10(1):381–394, 2018. [p]

F. E. Harrell Jr. *rms: Regression Modeling Strategies*, 2019. URL https://CRAN.R-project.org/package=rms. R package version 5.1-4. [p]

M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938. [p]

D. Liu and H. Zhang. Residuals and diagnostics for ordinal regression models: A surrogate approach. *Journal of the American Statistical Association*, 113(522):845–854, 2018. [p]

D. Liu, S. Li, Y. Yu, and I. Moustaki. Assessing partial association between ordinal variables: quantification, visualization, and hypothesis testing. *Journal of the American Statistical Association*, 116(534): 955–968, 2021. doi: 10.1080/01621459.2020.1796394. [p]

P. McCullagh. Regression models for ordinal data (with discussion). *Journal of the Royal Statistical Society. Series B (Methodological)*, 42:109–142, 1980. [p]

B. Schloerke, J. Crowley, D. Cook, F. Briatte, M. Marbach, E. Thoen, A. Elberg, and J. Larmarange. *GGally: Extension to 'ggplot2'*, 2020. URL https://CRAN.R-project.org/package=GGally. R package version 1.5.0. [p]

B. Schweizer and E. F. Wolff. On nonparametric measures of dependence for random variables. *The Annals of Statistics*, 9(4):879–885, 1981. [p]

C. Sievert. *Interactive Web-Based Data Visualization with R, plotly, and shiny*. Chapman and Hall/CRC, 2020. ISBN 9781138331457. URL https://plotly-r.com. [p]

G. Tallis. The maximum likelihood estimation of correlation from contingency tables. *Biometrics*, 18(3): 342–353, 1962. [p]

W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. URL https://www.stats.ox.ac.uk/pub/MASS4/. ISBN 0-387-95457-0. [p]

H. Wickham. *ggplot2: elegant graphics for data analysis*. Springer, 2016. [p]

T. W. Yee et al. The vgam package for categorical data analysis. *Journal of Statistical Software*, 32(10): 1–34, 2010. [p]

X. Zhu, S. Li, Y. Chen, and D. Liu. *PAsso: Assessing the Partial Association Between Ordinal Variables*, 2021. URL https://CRAN.R-project.org/package=PAsso. R package version 0.1.10. [p]

Shaobo Li
University of Kansas
1654 Naismith Drive
Lawrence, KS 66045
shaobo.li@ku.edu

Xiaorui Zhu
University of Cincinnati
2906 Woodside Drive
Cincinnati, OH 45221
zhuxr@mail.uc.edu

Yuejie Chen
University of Cincinnati
2906 Woodside Drive
Cincinnati, OH 45221
chen4y9@mail.uc.edu

Dungang Liu
University of Cincinnati
2906 Woodside Drive
Cincinnati, OH 45221
dungang.liu@uc.edu