Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *doi: 10.18637/jss.v000.i00*

ggESDA: An R Package for Exploratory Symbolic Data Analysis using ggplot2

Bo-Syue Jiang
National Taipei University

Han-Ming Wu
National Chengchi University

Abstract

Exploratory data analysis (EDA) serves as a preliminary yet essential tool for summarizing the main characteristics of a data set before appropriate statistical modeling can be applied. Quite often, EDA employs the traditional graphical techniques such as the boxplot, histogram and scatterplot and are equipped with various dimension reduction methods and computer-aided interactive functionalities. EDA has been used to explore different data types. Examples were the cases of the survival data, the time series data, the functional data and the longitudinal data. Conventionally, these data set were tabulated by a table with p columns corresponding to p variables. Each subject is measured by a single numerical value for each variable. Nowadays the collected data keeps getting bigger and complex. The description of data was no longer stored by a form of a single value but the intervals, histograms and/or distributions. These are examples of the so-called symbolic data. This study develops an R package, namely **ggESDA**, as an extension of **ggplot2** package to provide a wide variety of plots for exploring symbolic data. SDA supplies various data descriptions and has great capacity for large and complex data. The **ggESDA** acts as an advanced graphical tool that can support the efficient, effective and practical exploration of symbolic data sets.

Keywords: data visualization, exploratory data analysis, interval-valued data, statistical graphics, symbolic data analysis.

1. Introduction

The exploratory data analysis (EDA) was pioneered by Tukey (1977). Data analysis in EDA involves using graphical techniques such as scatterplots, boxplots, and histograms, as well as descriptive statistics such as mean, median, and variance in order to gain a general understanding of the data. Quite often, dimension reduction (DR) methods including Principal Component Analysis (PCA) and Multidimensional Scaling (MDS) are also used to visualize multivariate data sets. There are some advantages of using graphical approaches to data analysis over the traditional formal statistical modeling and inference. As an example, they help users to summarize large, complicated, heterogeneous data sets, and to find patterns, features, trends, anomalies, and relationships among them. EDA is especially useful during the early stages of data mining. Due to the popularity of data science and big data, EDA has regained people's attention in recent years for pattern discovery and statistical visualization. A number of EDA-related packages, including **DataExplorer** (Cui 2020), **GGally** (Schloerke, Cook, Larmarange, Briatte, Marbach, Thoen, Elberg, and Crowley 2021), **SmartEDA** (Dayanand Ubrangala, R, Prasad Kondapalli, and Putatunda 2021) and **tableone** (Yoshida and Bartel 2022), are available on the Comprehensive R Archive Network (CRAN).

In conventional data analysis, the data set is usually presented as a classical data table where each object (or subject) is represented as a single point in p-dimensional space (variables). In contrast, symbolic data with measurements on p random variables are p-dimensional statistical units such as hypercubes or histograms in $\mathbb{R}^p$, or a Cartesian product of p distributions. The symbolic data analysis (SDA) proposed by Billard and Diday (2003, 2007) gave a comprehensive overview of statistical methods for analyzing such data. Symbolic variables make it possible to describe groups of individuals and concepts. In practice, the symbolic data is generated by aggregating the large or enormous data sets into intervals or histograms. An aggregation can produce a data set of more manageable size so as to conduct appropriate analyses; or it can give rise to some scientific questions of interest.

Traditional EDA graphical techniques such as index plots, boxplots, scatterplots, and histograms are not applicable directly to symbolic data, as they were developed for classical data. Some DR methods, on the other hand, have been developed to explore symbolic objects in a lower-dimensional space when the dimensionality of the data is high. Therefore, it is necessary to develop an EDA tool with a more visual approach to symbolic data.

A number of R packages for symbolic data analysis can be found on CRAN, as shown in Table 1. These packages implement statistical and machine learning methods, primarily for interval-valued data, such as dimension reduction, clustering, and regression, among others. Furthermore, some of these packages provided basic and limited graphical tools such as scatterplots, histograms and radar plots using the base R graphics system and **ggplot2** (Wickham (2016)). There has been no R package developed specifically for exploratory symbolic data analysis and visualization. This motivates us to develop a new R package, namely **ggESAD**, based on **ggplot2** in order to take advantage of its advanced graphical features. Our current research focuses on the EDA of interval data, in which variables have interval values.

The article is structured as follows. Section 2 introduces the **ggESDA** package design and interval-valued symbolic data manipulation. In Section 3 we review a few descriptive statistics that are necessary for statistical graphing. Section 4 demonstrates a variety of implemented plots with R codes. Section 5 provides generalizations and customizations of the package, including visualizing the interval data in PCA subspace. We then conclude in Section 6.

2. Package design and data manipulation

2.1. The ggESDA package design

The design of **ggESDA** consists of three parts. The first part is devoted to data manipulation,

Package	SO*	Description	Plots	Data Analysis Methods
GraphPCA	H	Graphical Tools of Histogram PCA	scatterplot	PCA
HistDat	H	Summary Statistics for Histogram/Count Data	-	basic statistics
HistDAWass	H	Histogram-Valued Data Analysis	boxplot, CDF, histogram, matrix of histograms, QF	basic statistics, Batch SOM, Fuzzy c-means, hierarchical clustering, K-means, PCA, regression, time series
HistogramTools	H	Utility Functions for R Histograms	ECDF plots	data manipulation, distance, quantile.
IntervalQuestionStat	I	Tools to Deal with Interval-Valued Responses in Questionnaires	scatterplot	arithmetic operations, Cronbach's α, distance, transformation
intkrige	I	A Numerical Implementation of Interval-Valued Kriging	interval plot, variograms	distance, kriging
iRegression	I	Regression Methods for Interval-Valued Variables	-	regression
MAINT.Data	I	Model and Analyse Interval Data	outliers plot, parallel coordinates plot, scatterplot	clustering, discriminant analysis, LRT, MANOVA, mixture model estimation, MLE
mdsOpt	I	Searching for Optimal MDS Procedure for Metric and Interval-Valued Data	scatterplot	MDS
psda	P	Polygonal Symbolic Data Analysis	scatterplot	basic statistics, regression
RSDA	I	R to Symbolic Data Analysis	histogram, radar, scatterplot	basic statistics, distance, K-means, KNN, MCFA, PCA, regression, RF, SVM.
symbolicDA	I	Analysis of Symbolic Data	3D interval plot, radar	decision tree, distance, dynamical clustering, HINoV, KDA, MDS, PCA, RF, SOM.

*Symbolic object: I for the interval-valued data; H for the histogram-valued data; P for the polygonal-valued data.

Table 1: The main features and the plots provided by the various R packages available on CRAN for symbolic data.

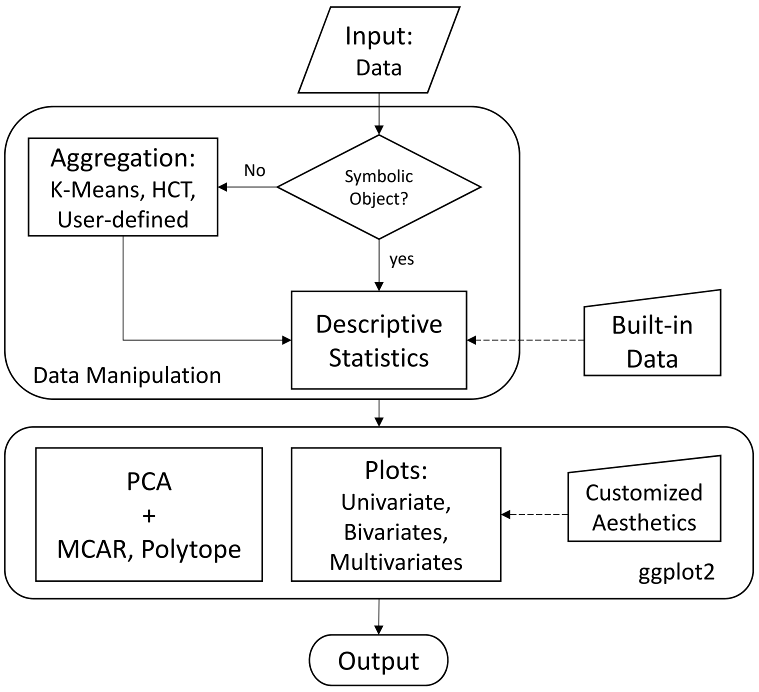


Figure 1: The design structure of the **ggESDA** package.

such as read and write from/to files, and aggregation. The second is to compute descriptive statistics for interval-valued univariate and bivariate variables, such as means, variances, and covariances. The third part covers the implementation of basic and advanced graphical techniques for interval-valued univariate, bivariate, and multivariate symbolic variables such as boxplot, histogram, scatterplot, radar plots, image plot, and so on. Figure 1 illustrates the design structure of **ggESDA**.

As the **ggESDA** package is based on the **ggplot2**, we would like to give a brief description of the **ggplot2**. The concept behind **ggplot2** consists of three different fundamental parts for a plot:

```
Plot = Data + Aesthetics + Geometry.
```

The aesthetics is used for indicating variables x and y of a dataframe `Data`. Additionally, it can be used to customize the color, shape, or size of points, as well as the height of bars. Geometry defines the types of graphics such as histograms, boxplots, lines plots, density plots, dot plots, and so on. The general syntax for constructing a plot with **ggplot2** is as follows.

```
ggplot(data = <DATA> ) +
  <GEOM_FUNCTION> (mapping = aes( <MAPPINGS> ),
                  stat = <STAT> , position = <POSITION> ) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION> +
  <SCALE_FUNCTION> +
  <THEME_FUNCTION>
```

Data	Dimension	Reference	Application
`facedata`	27 × 6	Douzal-Chouakria, Billard, and Diday (2011) Le-Rademacher and Billard (2012)	PCA
`oils`	8 × 4	Lauro and Palumbo (2000) Lauro, Verde, and Irpino (2008)	PCA
`mushroom`	23 × 3	Neto, Cordeiro, and de Carvalho (2011) Domingues, de Souza, and Cysneiros (2010)	Regression
`Cardiological`	11 × 3	Billard and Diday (2000) Xu (2010)	Regression
`AbaloneIdt`	24 × 7	Malerba, Esposito, Gioviale, and Tamma (2001)	Dissimilarity
`Environment`	14 × 17	Umbleja, Ichino, and Yaguchi (2020)	EDA

Table 2: Summaries of the built-in interval-valued data sets in the **ggESDA** package.

Both `ggplot(data = <DATA> )` and `<GEOM_FUNCTION>` are required. Below is an example of a scatterplot of the `Sepal.Length` and `Sepal.Width` of the `iris` data.

```
R> library(ggplot2)
R> ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
+     geom_point() +
+     scale_color_manual(values = c("red", "blue", "yellow"))
```

Note that data used in **ggplot2** should come from the classes of `data.frame` or `tibble`. Classes with interval-valued data, such as `symbolic_interval` and `symbolic_tbl` defined in **RSDA** package (Rodriguez 2022), are not available since they haven't been implemented in **ggplot2**. To resolve the issue, we developed a R function, namely `ggInterval_<GRAPH_TYPE>`, as the principal syntax of **ggESDA** to replace `ggplot()` function. This is also a common design for some **ggplot2** extensions. For instance, the packages of **ggtern** (Hamilton and Ferry 2018), **gganatogram** (Maag 2018), **ggstatsplot** (Patil 2021), and etc. Following is the general syntax of **ggESDA** for producing a plot of interval-valued data.

```
ggInterval_<GRAPH_TYPE>(data = <SYMBOLIC_DATA>,
                        mapping = aes(<MAPPINGS>), ...) +
  geom_<LAYER_TYPE>(...) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION> +
  <SCALE_FUNCTION> +
  <THEME_FUNCTION>
```

The `<LAYER_TYPE>` includes `geom_text`, `geom_line`, `geom_point` and `geom_area`. Section 4 provides some examples of code. In contrast to **ggplot**, which requires data in a specific format, the interval-valued data in the SDA-related R packages have a wide range of formats. To facilitate future maintenance and extension, we adopt the data object of the `symbolic_interval` and `symbolic_tbl` classes from **RSDA** for `data = <SYMBOLIC_DATA>` in our package.

2.2. Built-in interval-valued data sets

Several widely used interval-valued data sets are included in this package as built-in data sets, as shown in Table 2. Two of them will be used for illustration in this study: `facedata` and `Environment`. The face recognition data set (Leroy, Chouakria, Herlin, and Diday 1996;

Douzal-Chouakria *et al.* 2011; Le-Rademacher and Billard 2012) provides six face measurements of nine men (denoted by `Subjects`), all with three replicates, for a total of 27 observations. The six measurements designed to identify each face are expressed as the number of pixels in an image. Specically, they are the distance spanned by the eyes (denoted by $X_1 =$ AD), the distance between the eyes ($X_2 =$ BC), the distance from the outer right eye to the upper middle lip between the nose and mouth ($X_3 =$ AH), the corresponding distance for the left eye ($X_4 =$ DH), the distance from the upper middle lip to the outside of the mouth on the right side ($X_5 =$ EH), as well as the distance to the left side of the mouth ($X_6 =$ GH). These measurements were taken from a sequence of more than 1,000 images and cover a wide range of values which make them interval variables.

```
R> library(ggESDA)
R> data(facedata)
R> facedata
```

```
# A tibble: 27 x 6
                     AD                BC                  AH                  DH
 *         <symblc_n>        <symblc_n>          <symblc_n>          <symblc_n>
 1 [155.00 : 157.00] [58.00 : 61.01] [100.45 : 103.28] [105.00 : 107.30]
 2 [154.00 : 160.01] [57.00 : 64.00] [101.98 : 105.55] [104.35 : 107.30]
 3 [154.01 : 161.00] [57.00 : 63.00]  [99.36 : 105.65] [101.04 : 109.04]
 4 [168.86 : 172.84] [58.55 : 63.39] [102.83 : 106.53] [122.38 : 124.52]
 5 [169.85 : 175.03] [60.21 : 64.38] [102.94 : 108.71] [120.24 : 124.52]
 6 [168.76 : 175.15] [61.40 : 63.51] [104.35 : 107.45] [120.93 : 125.18]
 7 [155.26 : 160.45] [53.15 : 60.21]   [95.88 : 98.49]    [91.68 : 94.37]
 8 [156.26 : 161.31] [51.09 : 60.07]   [95.77 : 99.36]    [91.21 : 96.83]
 9 [154.47 : 160.31] [55.08 : 59.03]   [93.54 : 98.98]    [90.43 : 96.43]
10 [164.00 : 168.00] [55.01 : 60.03] [120.28 : 123.04] [117.52 : 121.02]
# ... with 17 more rows, and 2 more variables: EH <symblc_n>, GH <symblc_n>
```

```
R> class(facedata)
```

```
[1] "symbolic_tbl" "tbl_df"       "tbl"          "data.frame"
```

```
R> Subjects <- substr(rownames(facedata), 1, 3)
R> Subjects
```

```
 [1] "FRA" "FRA" "FRA" "HUS" "HUS" "HUS" "INC" "INC" "INC" "ISA" "ISA" "ISA"
[13] "JPL" "JPL" "JPL" "KHA" "KHA" "KHA" "LOT" "LOT" "LOT" "PHI" "PHI" "PHI"
[25] "ROM" "ROM" "ROM"
```

When applied to an interval variable, `summary()` function returns the summary statistics of the lower and upper bounds of the intervals, similar to what `summary()` does for a numerical variable.

```
R> summary(facedata)
```

```
$symbolic_interval
                       AD                BC                  AH
Min.    [149.34 : 155.32] [50.36 : 55.23]   [93.54 : 98.49]
1st Qu. [154.56 : 158.91] [53.60 : 59.08] [102.88 : 106.99]
Median  [163.00 : 167.07] [57.00 : 63.00] [115.26 : 119.60]
Mean    [162.90 : 162.90] [60.00 : 60.00] [113.17 : 113.17]
3rd Qu. [167.13 : 171.19] [61.22 : 65.04] [117.91 : 121.60]
Max.    [169.85 : 175.15] [66.03 : 69.01] [123.75 : 127.29]
Std.        [6.82 : 6.82]   [4.42 : 4.42]     [9.08 : 9.08]
                       DH                EH                GH
Min.      [90.43 : 94.37] [49.41 : 54.64] [48.27 : 50.61]
1st Qu. [105.18 : 111.07] [54.65 : 58.49] [51.60 : 56.03]
Median  [114.28 : 117.41] [56.73 : 61.72] [55.32 : 60.46]
Mean    [113.20 : 113.20] [59.85 : 59.85] [57.69 : 57.69]
3rd Qu. [117.10 : 121.72] [60.96 : 65.80] [58.52 : 63.84]
Max.    [124.08 : 127.78] [63.89 : 69.07] [64.20 : 67.80]
Std.        [9.48 : 9.48]   [4.04 : 4.04]   [4.63 : 4.63]
```

The Environment questionnaire data (Diday and Noirhomme-Fraiture 2008) consists of 14 objects with 17 variables, four of which are modal multi-valued and the rest interval-valued. Modal multi-valued variables will be used to illustrate the radar plot.

```
R> data(Environment)
R> dim(Environment)

[1] 14 17

R> summary(Environment[1:3])

$symbolic_interval
                    CONTROL                SATISFY            INDIVIDUAL
Min.    [-723.25 : -339.65] [-570.57 : -259.45] [-826.25 : -484.48]
1st Qu.   [-270.31 : 83.66]  [-355.04 : -60.17]  [-508.97 : -88.07]
Median   [-189.18 : 188.39]  [-176.68 : 189.71]  [-202.13 : 132.72]
Mean        [-9.75 : -9.75]      [10.97 : 10.97]    [-31.16 : -31.16]
3rd Qu.   [-28.26 : 307.38]    [20.92 : 328.60]    [120.46 : 394.34]
Max.      [133.40 : 478.04]   [342.58 : 705.03]    [484.35 : 817.59]
Std.      [209.53 : 209.53]   [287.41 : 287.41]    [377.18 : 377.18]

$symbolic_modal
     URBANICITY INCOMELEVEL EDUCATION REGIONDEVELOPME
[1,] 6: 0.15    25: 0.32    1: 0.37   4: 0.54
[2,] 4: 0.44    50: 0.20    3: 0.34   3: 0.23
[3,] 5: 0.10    75: 0.25    5: 0.16   2: 0.15
[4,] 1: 0.19    90: 0.13    6: 0.13   1: 0.07
[5,] 3: 0.07    100: 0.10
[6,] 2: 0.04
```

2.3. Aggregate conventional data table to interval-valued data table

The aggregation of a numerical data set is a possible source of interval data. By this approach, the data is categorized into groups according to some criteria, and the intervals are constructed by summarizing the values of the minimum value and maximum value within each group. The advantage of using interval-valued symbolic data is that a large data set can be reduced to a manageable size, retaining as much information as possible from the original data set. We implement an R function, `classic2sym()`, that aggregates a data object of the `data.frame` class into an object of the `symbolic_tbl` class. We demonstrate the proposed aggregation approaches below using the Breast Cancer Wisconsin (Diagnostic) data set obtained from kaggle at https://www.kaggle.com/uciml/breast-cancer-wisconsin-data?select=data.csv. Data consists of 569 patients and 32 variables (ID, diagnosis, and 30 real-valued features) that describe characteristics of the nuclei present in a digitized image of a fine needle aspirate (FNA) of a breast mass.

```
R> breastData <- read.csv("data.csv")
R> colnames(breastData)
```

```
 [1] "diagnosis"               "radius_mean"             "texture_mean"
 [4] "perimeter_mean"          "area_mean"               "smoothness_mean"
...
[28] "concavity_worst"         "concave.points_worst"    "symmetry_worst"
[31] "fractal_dimension_worst"
```

Aggregation by clustering algorithms

Clustering algorithms can be seen as a natural way of aggregating data. Among these, `classic2sym()` adopts K-means and hierarchical clustering through the argument `groupby = c(kmeans, hclust, customize)` and returns a list of components, such as clustering results and interval-valued data. An interval-valued object is formed by each cluster. The interval-valued data can be extracted using `$intervalData`. Note that only numerical variables are used in the clustering algorithm. As a result of clustering, categorical variables are summarized by their relative frequencies within each cluster.

```
R> breastData <- dplyr::select(breastData, -id)
R> breastData.sym <- classic2sym(breastData, groupby = "kmeans", k = 5)
R> breastData.sym.i <- breastData.sym$intervalData
R> as.data.frame(head(breastData.sym.i[, 1:4], 5))
```

```
      diagnosis     radius_mean    texture_mean    perimeter_mean
1 B:0.00 M:1.00 [20.73 : 28.11] [17.25 : 31.12] [135.70 : 188.50]
2 B:0.00 M:1.00 [15.50 : 24.25] [10.38 : 32.47] [102.90 : 166.20]
3 B:0.68 M:0.32 [11.84 : 16.30] [10.89 : 30.72]  [77.93 : 109.80]
4 B:0.98 M:0.02  [6.98 : 13.05]  [9.71 : 33.81]   [43.79 : 85.09]
5 B:0.04 M:0.96 [13.81 : 19.59] [11.89 : 39.28]  [91.56 : 132.40]
```

Aggregation by a categorical variable

A categorical variable in the data may also be used to conduct the aggregation. The number of observations (objects) for the converted interval-valued data is the number of categories for that variable. One can also possible aggregate observations based on user-defined categories.

```
R> breastData.sym <- classic2sym(breastData, groupby = "diagnosis")
R> breastData.sym.i <- breastData.sym$intervalData
R> head(breastData.sym.i[, 1:4])
```

```
      radius_mean    texture_mean   perimeter_mean           area_mean
1   [6.98 : 17.85]  [9.71 : 33.81] [43.79 : 114.60]   [143.50 : 992.10]
2 [10.95 : 28.11] [10.38 : 39.28] [71.90 : 188.50] [361.60 : 2,501.00]
```

Alternatively, users can create interval-valued data tables using the function `classic.to.sym()` in the **RSDA** package.

2.4. Reading and writing interval-valued data to and from R

To deal with the input and output of the interval-valued data, **ggESDA** utilizes R functions, `read.sym.table()` and `write.sym.table()`, which are provided by **RSDA**. Symbolic data can be written and read as CSV files with a pre-defined format. With this design, users of SDA-related packages can become familiar with **ggESDA** quickly. Please refer to the help manual of `read.sym.table()` and `write.sym.table()` in the **RSDA** package for details.

3. Descriptive statistics for interval-valued data

EDA consists primarily of summaries of numerical data (e.g., the measures of central tendency and of variation or variability) as well as graphical summaries. For example, the five-number summary of numerical data (minimum, 25% quartiles, median, 70% quartiles, and maximum) is used to construct a boxplot. As a result, the basic descriptive statistics of symbolic data should be provided before (some) graphing techniques are applied. Additionally, if we want to standardize data or conduct multivariate analysis such as PCA on interval-valued data, we need to calculate the mean, variance, and covariance first. The SDA literature contains some algorithms for calculating descriptive statistics for interval-valued data. Denote the ith observation of a univariate interval-valued variable X by $[a_i, b_i], i = 1, \cdots, n$ and the ith observation of a bivariate interval-valued variables (X_1, X_2) by $([a_{i1}, b_{i1}], [a_{i2}, b_{i2}]), i = 1, \cdots, n$. We have implemented basic descriptive statistics for univariate and bivariate interval-valued variables according to Irpino and Verde (2015). Formulas can be found in Table 3. The following code illustrates how to obtain these descriptive statistics.

```
R> mean(facedata)
```

```
          AD    BC    AH    DH    EH    GH
       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1       163.  60.0  113.  113.  59.8  57.7
```

(a) Univariate statistics

According to Bertrand and Goupil (2000), Billard and Diday (2007) and Billard (2008)	
Mean	$\bar{X} = \frac{1}{n}\sum_{i=1}^{n}\frac{a_i + b_i}{2}$
Variance	$S^2 = \frac{1}{3n}\sum_{i=1}^{n}(a_i^2 + a_i b_i + b_i^2) - \frac{1}{4n^2}\left[\sum_{i=1}^{n}(a_i + b_i)\right]^2$

(b) Bivariate statistics

According to Bertrand and Goupil (2000)	
Covariance	$C_{BG}(X_1, X_2) = \frac{1}{4n}\sum_{i=1}^{n}(b_{i1} + a_{i1})(b_{i2} + a_{i2}) - \frac{1}{4n^2}\left[\sum_{i=1}^{n}(b_{i1} + a_{i1})\right]\left[\sum_{i=1}^{n}(b_{i2} + a_{i2})\right]$
Correlation	$R_{BG}(X_1, X_2) = \frac{C_{BG}(X_1, X_2)}{S(X_1)S(X_2)}$
According to Billard and Diday (2007)	
Covariance	$\begin{aligned} C_{BD}(X_1, X_2) &= \frac{1}{3n}\sum_{i=1}^{n} G_1 G_2 \left[Q_1 Q_2\right]^{1/2}, \text{where for } \; j = 1, 2, \\ Q_j &= (a_{ij} - \bar{X}_j)^2 + (a_{ij} - \bar{X}_j)(b_{ij} - \bar{X}_j) + (b_{ij} - \bar{X}_j)^2, \\ G_j &= \begin{cases} -1, & \text{if } \xi_{ij}^c \le \bar{X}_j, \\ 1, & \text{if } \xi_{ij}^c > \bar{X}_j, \end{cases} \\ & \text{where } \xi_{ij}^c = (a_{ij} + b_{ij})/2. \end{aligned}$
Correlation	$R_{BD}(X_1, X_2) = \frac{C_{BD}(X_1, X_2)}{S(X_1)S(X_2)}$
According to Billard (2008)	
Covariance	$\begin{aligned} C_B(X_1, X_2) = \frac{1}{6n}\sum_{i=1}^{n}[2(a_{i1} - \bar{X}_1)(a_{i2} - \bar{X}_2) + (a_{i1} - \bar{X}_1)(b_{i2} - \bar{X}_2) + \\ (b_{i1} - \bar{X}_1)(a_{i2} - \bar{X}_2) + 2(b_{i1} - \bar{X}_1)(b_{i2} - \bar{X}_2)]. \end{aligned}$
Correlation	$R_B(X_1, X_2) = \frac{C_B(X_1, X_2)}{S(X_1)S(X_2)}$

Table 3: Univariate and bivariate statistics for interval-valued data.

```
R> sd(facedata)

# A tibble: 1 x 6
     AD    BC    AH    DH    EH    GH
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  6.82  4.42  9.08  9.48  4.04  4.63

R> > cov(facedata$AD, facedata$BC, method = "BD")

[1] 21.47015
```

Standardization of interval-valued data

The standardization of each variable of the data to have a zero mean and unit variance is often

No. variates	Plots	Syntax
Univariate	index plot	`ggInterval_index`
	boxplot	`ggInterval_boxplot`
	histogram	`ggInterval_hist`
	line plot	`ggInterval_index`
	center-range plot	`ggInterval_centerRange`
	min-max plot	`ggInterval_minmax`
Bivariate	2D scatterplot	`ggInterval_scatter`
	2D histogram	`ggInterval_2Dhist`
Multivariate	image plot	`ggInterval_indexImage`
	radar plot	`ggInterval_radar`
	scatterplot matrices	`ggInterval_scaMatrix`
	2D histogram matrices	`ggInterval_2DhistMatrix`

Table 4: The graphing techniques provided by **ggESDA**.

vital for data analysis. Standardization prevents variables with larger measurement scales from dominating those with smaller scales. In this package, the general rule for standardizing an interval-valued variable, consists of applying the same transformation separately to both the lower and upper bounds of all intervals, which standardizes all point values between the lower and upper bounds in the same linear way.

```
R> scale(facedata)$intervalData

                  AD               BC               AH               DH
FRA1 [-1.10 : -0.82]  [-0.39 : 0.20] [-1.38 : -1.07] [-0.85 : -0.61]
FRA2 [-1.24 : -0.40]  [-0.59 : 0.79] [-1.21 : -0.83] [-0.92 : -0.61]
FRA3 [-1.23 : -0.26]  [-0.59 : 0.59] [-1.50 : -0.82] [-1.26 : -0.43]
HUS1   [0.83 : 1.38]  [-0.28 : 0.67] [-1.12 : -0.72]   [0.95 : 1.17]
HUS2   [0.96 : 1.68]   [0.04 : 0.86] [-1.11 : -0.48]   [0.73 : 1.17]
HUS3   [0.81 : 1.70]   [0.28 : 0.69] [-0.96 : -0.62]   [0.80 : 1.24]
...
```

4. Graphics for interval-valued symbolic data

ggESDA extends and implements some traditional graphical techniques for interval-valued data that can be classified as univariate, bivariate, and multivariate plots as shown in Table 4. It should be noted that there have been many other variants of traditional graphical techniques proposed in the literature. We aim to introduce basic statistical graphics in this study for displaying symbolic data accurately and effectively for data exploration. It is possible to add more advanced graphics in the future. According to the characteristics of interval-valued data, we also develop the min-max plot and the center-range plot.

4.1. Univariate plots

The univariate plot describes or characterizes the location, dispersion, and distribution of

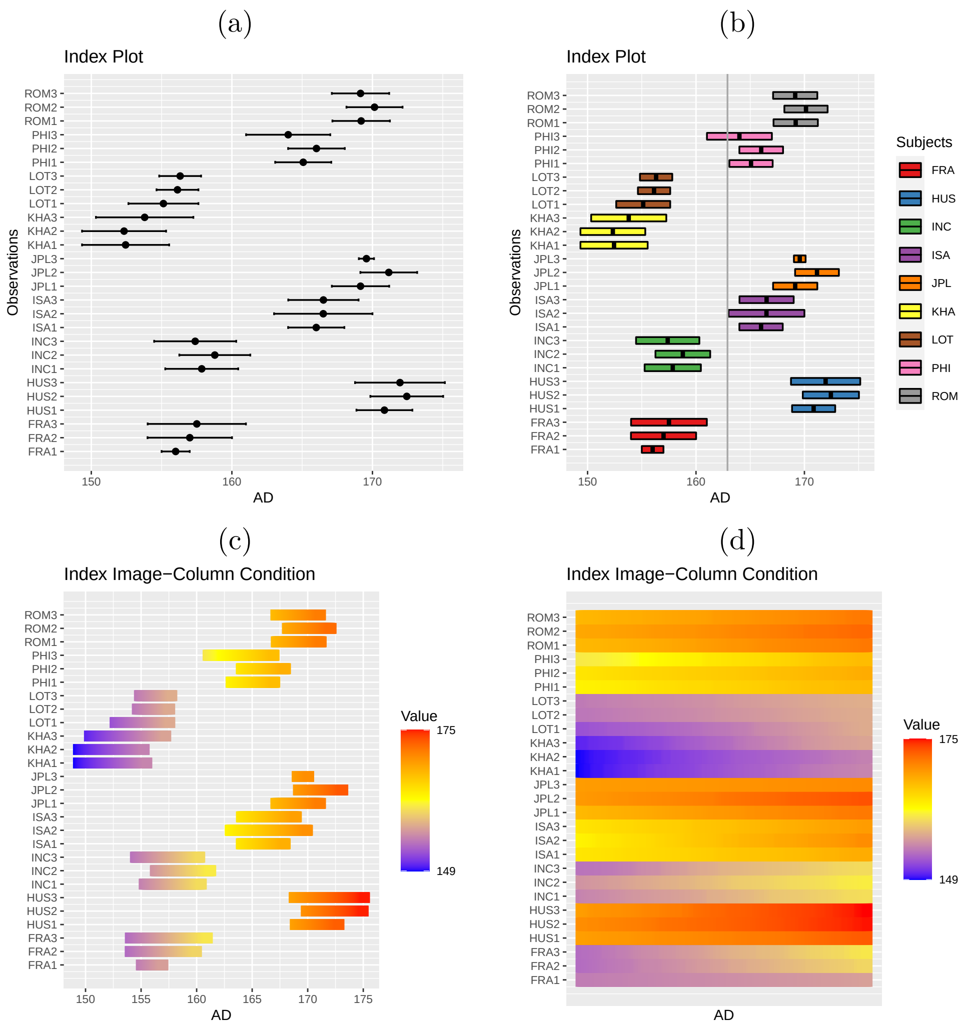


Figure 2: Four types of index plot for the variable AD of the face recognition data set. (a) Index plot. (b) Index plot with color bars. (c) Index image. (d) Index image with equi-width strips.

observations for an individual variable. For interval-valued observations, we have implemented the index plot, boxplot, histogram, line plot, min-max plot, and center-range plot.

The index plot An index plot (also called the run sequence plot (x_i vs i)) shows the values of a variable against the corresponding observation number (row or index) in the data set. With it, you can quickly scan large amounts of data in order to seek out anomalies and unusual observations. The index plot of an interval variable is composed of a segment or a bar with lower and upper bounds against the corresponding observation numbers. This plot makes it easy to observe the distributions of the ranges and centers of the intervals. The following example code using `ggInterval_index()` illustrates four types of index plots for the variable AD in the face data set. The Figure 2(a) shows an index plot with segment representation, where the x-axis represents the range of AD measurements and the y-axis represents the index of the observations. The index plot with colored bars representation is shown in Figure 2(b). The Figure 2(c) shows a variant of index plot with an image strip that displays the intervals' range through sequential colors. Hence, we call it an index image. Figure 2(d) shows equi-width image strips of the index image. As we can see, the set of three faces within each subject is cohesive within that subject. Obviously, subjects LOT, KHA, INC, and FRA have lower AD values than average.

```
R> # Figure 2(a)
R> f.nr <- nrow(facedata)
R> f.nc <- ncol(facedata)
R> ggInterval_index(facedata, aes(x = AD)) +
+    scale_y_continuous(breaks = 1:f.nr, labels = rownames(facedata))
R>
R> # Figure 2(b)
R> ggInterval_index(facedata, aes(x = AD, fill = Subjects)) +
+    scale_fill_brewer(palette = "Set1") +
+    geom_vline(xintercept = mean(facedata$AD), color="darkgray") +
+    scale_y_continuous(breaks = 1:f.nr, labels = rownames(facedata)) +
+    labs(fill = "Subjects")
R>
R> # Figure 2(c)
R> ggInterval_indexImage(facedata, aes(x = AD)) +
+    coord_flip()
R>
R> # Figure 2(d)
R> ggInterval_indexImage(facedata, aes(x = AD), full_strip = T) +
+    coord_flip()
```

Figure 3(a) shows the index plot of all six variables in the face recognition data set. In accordance with a normal person's symmetry of facial characteristics, we observed that two people, HUS and KHA, have different quantities of AH and DH measurements. As shown in Figure 3(b) and (c), it is possible to reorder the observations in accordance with some specific quantities such as centers or ranges of intervals to reveal the structural information. The following code chunk creates these graphs. Note that with the argument `plotAll = T`, `ggInterval_index()` will produce an index plot of all variables in a data set. Other graphing functions follow the same design, including `ggInterval_boxplot()`, `ggInterval_centerRange()`, `ggInterval_hist()`, `ggInterval_indexImage()`, and `ggInterval_minmax()`.

```
R> # Figure 3(a)
R> face.ip <- ggInterval_index(facedata, aes(fill = Subjects), plotAll = T) +
+    scale_fill_brewer(palette = "Set1") +
+    labs(x = "", y = "", fill = "Subjects")
R> face.ip
R>
R> # Figure 3(b)
R> library(dplyr)
R> dd <- 0
R> for(i in 1:dim(facedata)[2]){
```

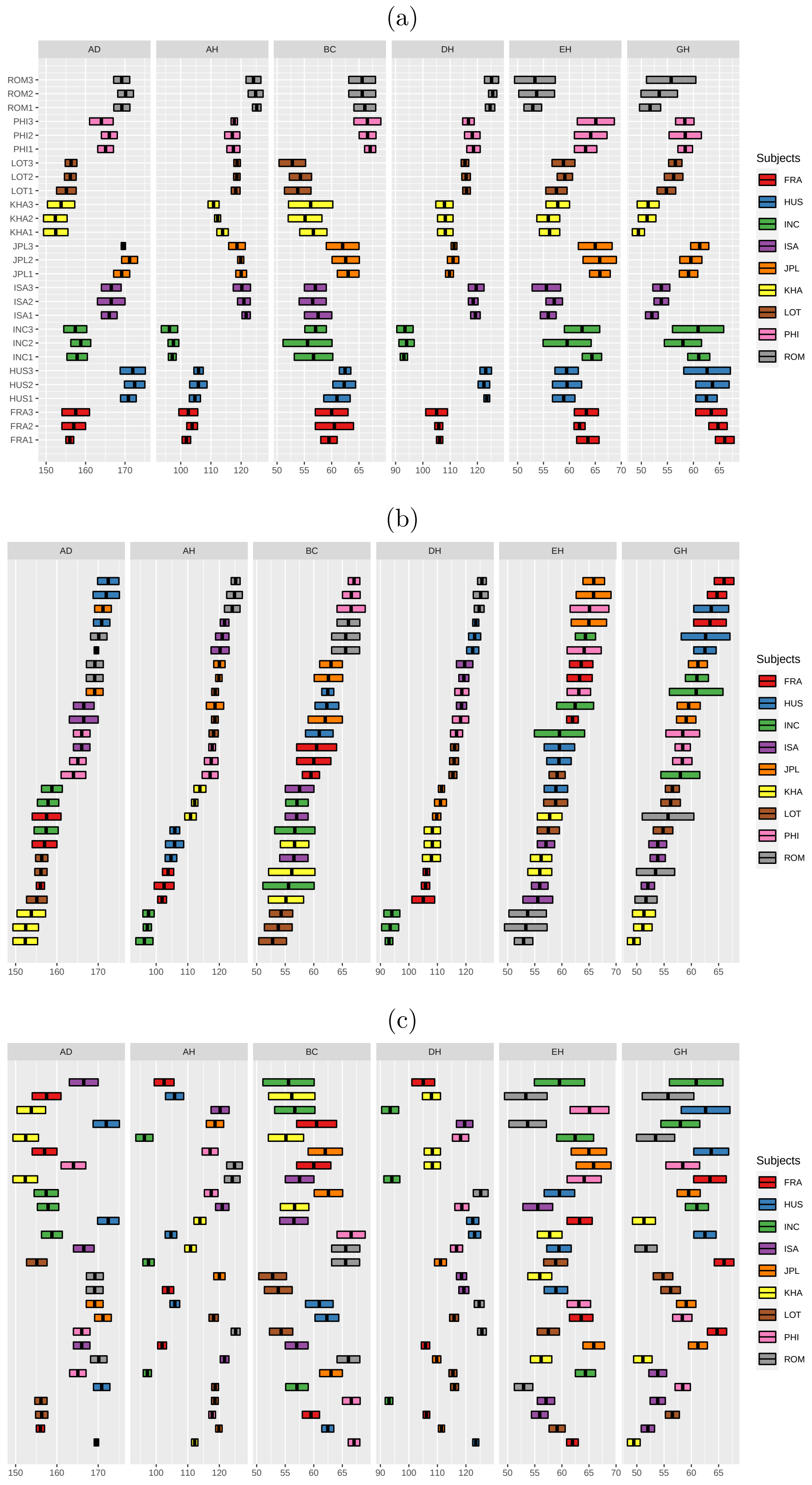


Figure 3: Index plots for face recognition data with different ordering of observations.

```
+    d <- data.frame(min = facedata[[i]]$min,
+                    max = facedata[[i]]$max,
+                    myname = Subjects,
+                    mysort = (facedata[[i]]$min + facedata[[i]]$max) / 2)
+    temp <- dplyr::arrange(d, mysort)
+    dd <- cbind(dd, temp)
+  }
R>
R> dd <- dd[,-1]
R> newd <- classic2sym(dd, groupby = "customize",
+                      minData = cbind(dd$min, dd$min.1, dd$min.2,
+                                      dd$min.3, dd$min.4, dd$min.5),
+                      maxData = cbind(dd$max, dd$max.1, dd$max.2,
+                                      dd$max.3, dd$max.4, dd$max.5))
R> face.sort <- newd$intervalData
R> colnames(face.sort) <- colnames(facedata)
R> rownames(face.sort) <- rownames(facedata)
R> myname <- dd[,c(3,7,11,15,19,23)]
R> ggInterval_index(face.sort, aes(fill = unlist(myname)), plotAll = T) +
+    scale_fill_brewer(palette="Set1") +
+    labs(x = "", y = "", fill = "Subjects") +
+    scale_y_continuous(breaks = NULL, labels = NULL)
```

The boxplot Figure 4 shows the side-by-side boxplots for all six variables of the face data. For each quantile box, the top (bottom) end represents the quantile of the upper (lower) bounds of the interval. In addition to providing a quick visual representation of data distributions, the boxplot can be particularly useful when comparing distributions among groups of variables. Below is the `ggInterval_boxplot()` function that generates the interval boxplot shown in Figure 4. Compared to other measurements, AD (BC) has the longest (shortest) length. It is expected that AH and DH (EH and GH) have similar facial characteristics since they are both expected to be symmetrical in this regard. Furthermore, the overlap of quantile boxes for variables BC, EH, and GH could result in a bell-shaped distribution.

```
R> # Figure 4
R> ggInterval_boxplot(facedata, plotAll = T) +
+    theme(legend.position = "bottom", axis.text.x = element_blank())
```

The histogram The histogram is a graphical representation of data in which data are grouped into continuous number ranges with each range corresponding to a vertical bar. Numbers or frequencies within each range are represented by the height of each bar. It is best for visualizing the distribution of numerical data. An univariate histogram of the interval data is constructed as follows.

1. Let $I = [\min_i a_i, \max_i b_i], i = 1, 2, \cdots, n$ be the interval that spans all the observed values of X.

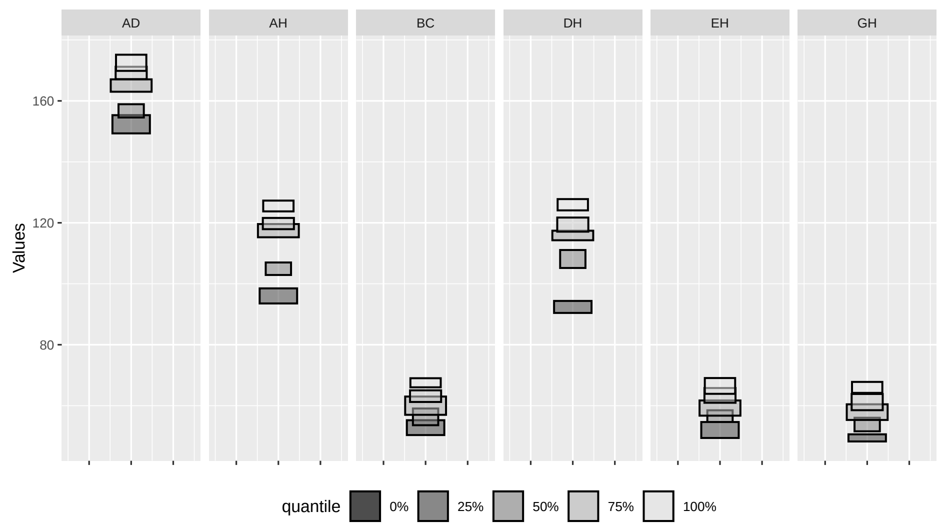


Figure 4: The side-by-side boxplot for all six variables of the face recognition data.

2. Partition I into r subintervals $I_g = [\xi_{g-1}, \xi_g)$ of equal width $\|I_g\|, g = 1, 2, \cdots, r$.
3. The observed frequency for the histogram subinterval $I_g = [\xi_{g-1}, \xi_g)$ is defined as:

$$f_g = \sum_i \frac{\| [a_i, b_i] \cap I_g \|}{\| [a_i, b_i] \|}, \; g = 1, \cdots, r. \tag{1}$$

4. The histogram of an interval-valued variable X is then obtained by $\{(I_g, f_g), g = 1, \cdots, r\}$.

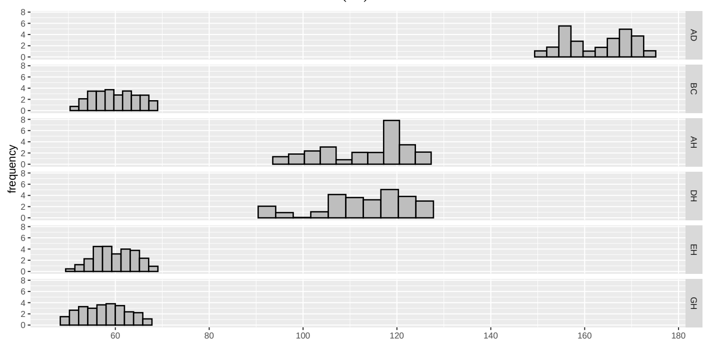
For the above steps, the subinterval I_g may not be necessary to have the same width. All boundaries of the intervals can be used to compute the breaks ξ_g. To be more precise, we pool a_i and b_i into a new vector and sort it as $(x^{(1)}, x^{(2)}, \cdots, x^{(2n)})$ where $x^{(j)} \leq x^{(j+1)}$, $j = 1, 2, \cdots, 2n-1$. The frequency for the subinterval $I'_g = [x^{(j)}, x^{(j+1)})$ can be obtained from the Equation (1). This histogram is known as the non-equidistant-bin histogram.

The function `ggInterval_hist()` implements the histogram for interval variables. By using `method = "unequal-bin"`, we can obtain the non-equidistant-bin histogram. Figure 5 (a) and (b) show the histograms of all six variables of the face data with equidistant bins and non-equidistant bins. They both show the location and scale of distribution of each variable, whereas the non-equidistant-bin histogram shows more details about data characteristics. According to these histograms, the distribution information is similar to that expressed in Figure 4. Variables AD, AH, and DH exhibit bimodal distributions.

```
R> # Figure 5(a)
R> hist.obj.equal <- ggInterval_hist(facedata, plotAll = T, bins = 10)
R> hist.obj.equal$plot
R> hist.obj.equal$`Table AD`

           Interval Observed.Frequency Relative.Frequency
1  [149.34:151.92]           1.077477         0.03990656
```

(a)



(b)

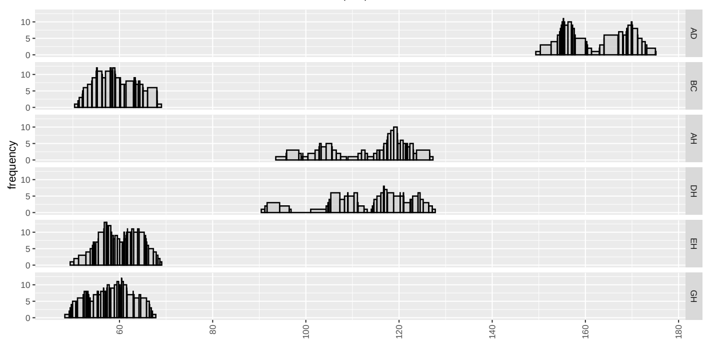


Figure 5: Histograms of the face recognition data. (a) Equidistant-bin. (b) Non-equidistant-bin.

```
2   [151.92:154.5]         1.753623        0.06494900
3   [154.5:157.08]         5.525627        0.20465284
4  [157.08:159.66]         2.806570        0.10394703
5  [159.66:162.24]         1.042880        0.03862518
6  [162.24:164.83]         1.703999        0.06311106
7  [164.83:167.41]         3.304380        0.12238443
8  [167.41:169.99]         4.948225        0.18326760
9  [169.99:172.57]         3.738883        0.13847713
10 [172.57:175.15]         1.098337        0.04067915
```

```
R> # Figure 5(b)
R> ggInterval_hist(facedata, plotAll = T, method = "unequal-bin")$plot
```

The line plot In a line plot, data is displayed along an ordered sequence or number line (usually on x-axis), showing how the data changes over time and if the data points are random or exhibit any patterns. A time plot is essentially a line plot. For illustration, we use stock price data. A time series plot can provide valuable information about changes in a financial instrument's price over time before a statistical model is applied. The following code reads the stock prices of three companies from Yahoo Finance, International Business Machines Corporation (IBM), Microsoft Corporation (MSFT) and Apple Inc. (APPL). This historical

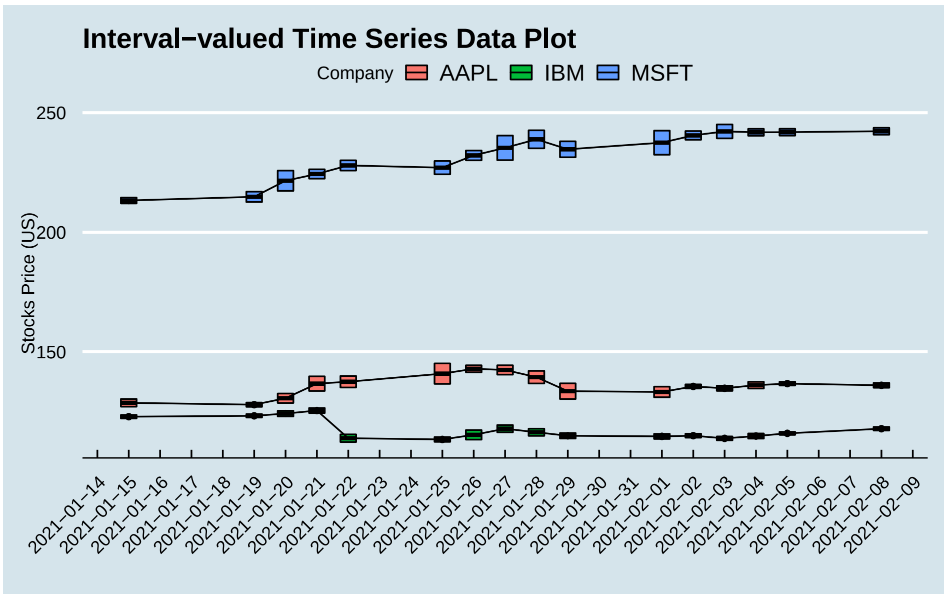


Figure 6: Interval-valued time series.

stock data set includes open, high, low, close, adjusted close prices, and trading volume values. To illustrate the interval time series plot, we select the low and high prices from 2021-01-15 to 2021-02-08.

```
R> library(data.table)
R> download.url <- "https://query1.finance.yahoo.com/v7/finance/download/"
R> url.option <- "?period1=1609459200&period2=1640995200&interval=1d&
+    events=history&includeAdjustedClose=true"
R>
R> ibm <- fread(paste0(download.url, "IBM", url.option))

 Downloaded 18703 bytes...

R> msft <- fread(paste0(download.url, "MSFT", url.option))

 Downloaded 18941 bytes...

R> aapl <- fread(paste0(download.url, "AAPL", url.option))

 Downloaded 19020 bytes...

R> from <- 10
R> to <- 25
R> Company <- rep(c("IBM", "MSFT", "AAPL"), each = to - from + 1)
R> DLH <- c("Date", "Low", "High")
R> stock.data <- rbind(ibm[from:to, ..DLH], msft[from:to, ..DLH],
+                        aapl[from:to, ..DLH])
R> head(stock.data)
```

```
        Date      Low     High
1 2021-01-15 122.0554 123.5564
2 2021-01-19 122.4570 123.8910
3 2021-01-20 122.9063 125.2964
4 2021-01-21 124.3308 126.4245
5 2021-01-22 112.1989 115.3920
6 2021-01-25 112.2849 114.2830
```

The interval time series plot shown in Figure 6 is accomplished by the functions `ggInterval_index()` and `geom_line()`, with an economist theme. A line connects the centers of high and low prices at each time point for three companies. During this period of time, MSFT's high/low price is higher than the two other stock companies and it is increasing in general, indicating an upward trend. A clear downturn in IBM's stock price was also observed between the days of 2021-01-21 and 2021-01-21, and there was no obvious pattern in the fluctuation afterward. Between 2021-01-21 and 2021-02-1, the stock price variability for all three companies seems to have increased slightly.

Note that if interval-valued time series data are obtained by aggregating observed values within some fixed time interval (i.e. days or months) sequentially over time , then the above code for producing the line plot is applicable as well.

```
R> # Figure 6
R> stock.data.LH.i <- classic2sym(stock.data, groupby = "customize",
+                                  minData = stock.data$Low,
+                                  maxData = stock.data$High)$intervalData
R>
R> ggInterval_index(stock.data.LH.i, aes(y = V1, x = stock.data$Date,
+                                        group = Company, fill = Company)) +
+    geom_line() +
+    scale_x_date(date_breaks = "1 day") +
+    labs(title = "Interval-valued Time Series Data Plot",
+         x = "", y = "Stocks Price (US)", fill = "Company") +
+    theme_economist() +
+    theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1))
```

The min-max plot The min-max plot is a 2D sactterplot in which the x-axis represents the minima and the y-axis presents the maxima of the intervals. The minimum and maximum points are connected by a segment. A diagonal 45-degree reference line is also superimposed. Variations within intervals can be easily visualized. In the code chunk below, `ggInterval_minmax()` is used to produce the min-max plots for the face data as shown in Figure 7. We can see that the variables EH, BC, and GH exhibit relatively large variability with intervals. Additionally, we design the argument `scaleXY = "global"` to force the limits of the x-axis and y-axis of all min-max plots to be the same.

```
R> # Figure 7
R> mm.plot <- function(x){
+    plot.var <<- x
```

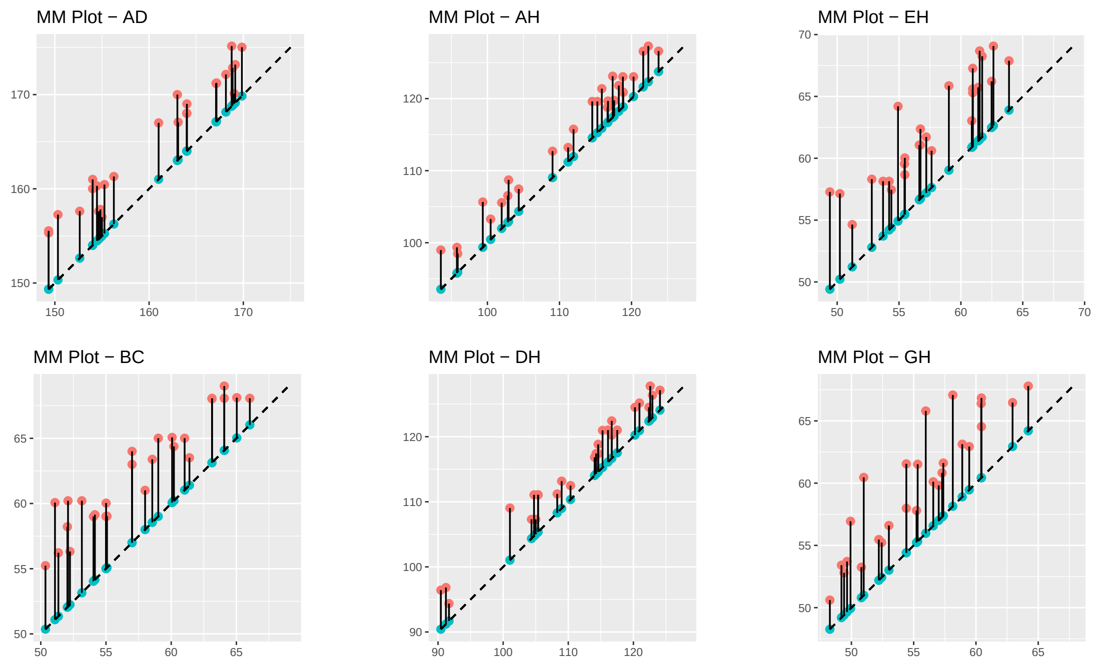


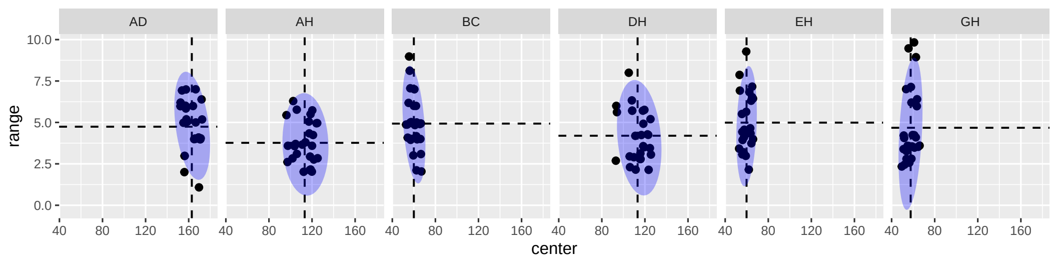
Figure 7: The min-max plot for the six variables of the face recognition data.

```
+    ggInterval_minmax(facedata, aes(facedata[[plot.var]], size = 2)) +
+    theme(legend.position = "none") +
+    coord_fixed(ratio = 1)
+ }
R> mm.plot.list <- lapply(1:6, mm.plot)
R> library(gridExtra)
R> marrangeGrob(mm.plot.list, nrow = 2, ncol = 3, top="")
```

The center-range plot As an alternative, the literature commonly uses center and range to represent intervals. A center-range plot is a 2D scatterplot in which the x-axis represents the centers and the y-axis represents the interval ranges. Figure 8(a) and (b) present a center-range plot for the six variables of the face recognition data with original and standardized scale using `ggInterval_centerRange()` function. The `stat_ellipse()` layer adds bivariate normal data ellipses. Researchers can more easily grasp the relationship between centers and ranges with the help of this plot. After standardization, AH and DH exhibit relative weak correlations between center and range, while in variable BC, there is a significant negative correlation between center and range.

```
R> # Figure 8(a)
R> ggInterval_centerRange(facedata, aes(size = 1.5), plotAll = T) +
+    stat_ellipse(geom = "polygon", fill = "blue", alpha = 0.3)
R>
R> # Figure 8(b)
```

(a)



(b)

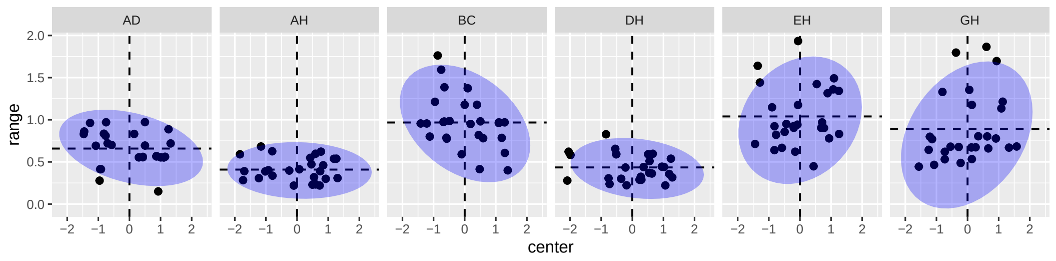


Figure 8: The center-range plot for the six variables of the face recognition data. (a) Original scale data. (b) The standardized data.

```
R> facedata.scale <- scale(facedata)$intervalData
R> ggInterval_centerRange(facedata.scale, aes(size = 1.5), plotAll = T) +
+    stat_ellipse(geom = "polygon", fill = "blue", alpha = 0.3)
```

4.2. Bivariate plots

The bivariate plot provides the means to determine how variables relate to each other when measured on the same sample of subjects. Several basic characteristics of this relationship are of interest, such as its form, its strength, and its dependence on external factors. We have implemented 2D scatterplots, 2D histograms, and time series plots.

The 2D scatterplot The function `ggInterval_scatter()` implements the 2D scatterplot for interval observations. The following code chunk produces the 2D scatterplot for AD and BC of the face recognition data, as shown in Figure 9(a). Using the function `scale_fill_brewer()`, the rectangles are filled with the same color for identical subjects. Variables AD and BC show a positive correlation in the scatterplot. Two main groups of observations can be distinguished. The AD and BC measurements of one group are larger while those of the other group are smaller.

```
R> # Figure 9(a)
R> ggInterval_scatter(facedata, aes(x = BC, y = AD, fill = Subjects),
+                    col = "black") +
+    scale_fill_brewer(palette = "Set1") +
+    labs(fill = "Subjects")
```

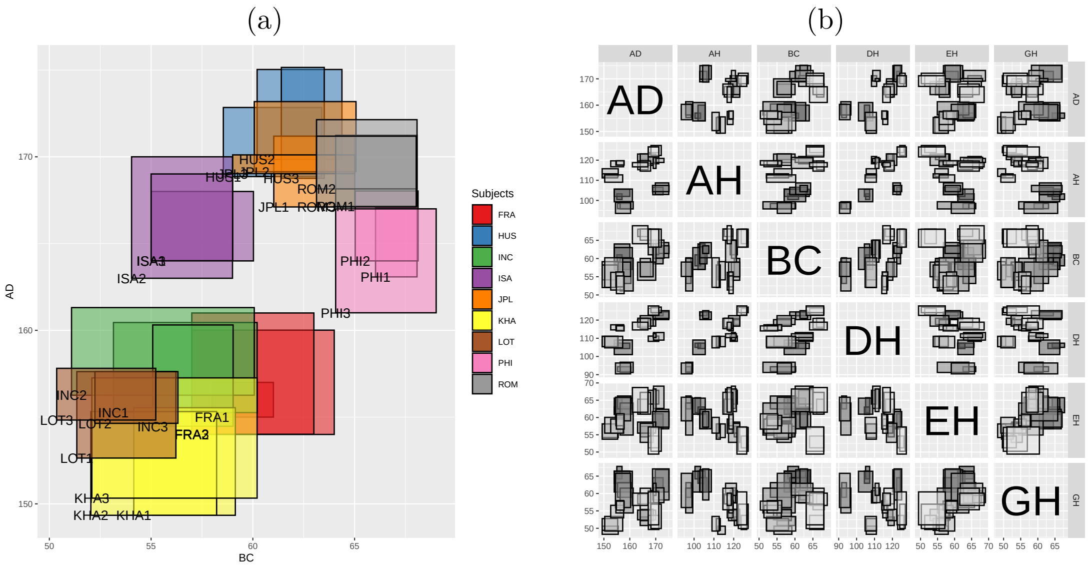


Figure 9: (a) The scatterplot of the variables AD against BC and (b) the scatterplot matrix of the face recognition data.

The 2D histogram In a similar manner to Equation (1), we can get the joint histogram of (X_1, X_2) as follows.

1. Let $I_j = [\min_i a_{ij}, \max_i b_{ij}], i = 1, 2, \cdots, n$ be the interval that spans all the observed values of $X_j,\ j = 1, 2$.

2. Partition I_j into r_j subintervals $I_{j,g_j} = [\xi_{j,g_j-1}, \xi_{j,g_j})$ of equal width $\|I_{j,g_j}\|, g_j = 1, 2, \cdots, r_j,\ j = 1, 2$.

3. Define the joint histogram for X_1 and X_2 by plotting $\{R_{g_1g_2}, p_{g_1g_2}\}$ over the rectangles $R_{g_1g_2} = \{[\xi_{1,g_1-1}, \xi_{1,g_1}) \times [\xi_{2,g_2-1}, \xi_{2,g_2})\}$, $g_1 = 1, \cdots, r_1, g_2 = 1, \cdots, r_2$, where

$$p_{g_1g_2} = \frac{1}{n}\sum_{i=1}^{n} \frac{\|Z_i \cap R_{g_1g_2}\|}{\|Z_i\|}$$

is the probability an arbitrary individual description vector lies in the rectangle $R_{g_1g_2}$ and $Z_i = [a_{i1}, b_{i1}) \times [a_{i2}, b_{i2})$.

If the volume on the rectangle $R_{g_1g_2}$ of the joint histogram represents the probability, its height would be

$$f_{g_1g_2} = [(\xi_{1,g_1} - \xi_{1,g_1-1}) \times (\xi_{2,g_2} - \xi_{2,g_2-1}]^{-1} p_{g_1g_2}.$$

That is, $f_{g_1g_2}$ is the number of observations that falls within the rectangle $R_{g_1g_2}$ and is not necessarily an integer value (except in the special case of classical data). The relative frequency that an arbitrary individual description vector lies in the rectangle $R_{g_1g_2}$ is therefore $p_{g_1g_2} = f_{g_1g_2}/n$. The function `ggInterval_2Dhist()` in the following code chunk produces the 2D histogram of variables AD and BC with bins 10×10 as shown in Figure 10(a). Each cell in the rectangle is colored with a number that represents its frequency. Results were

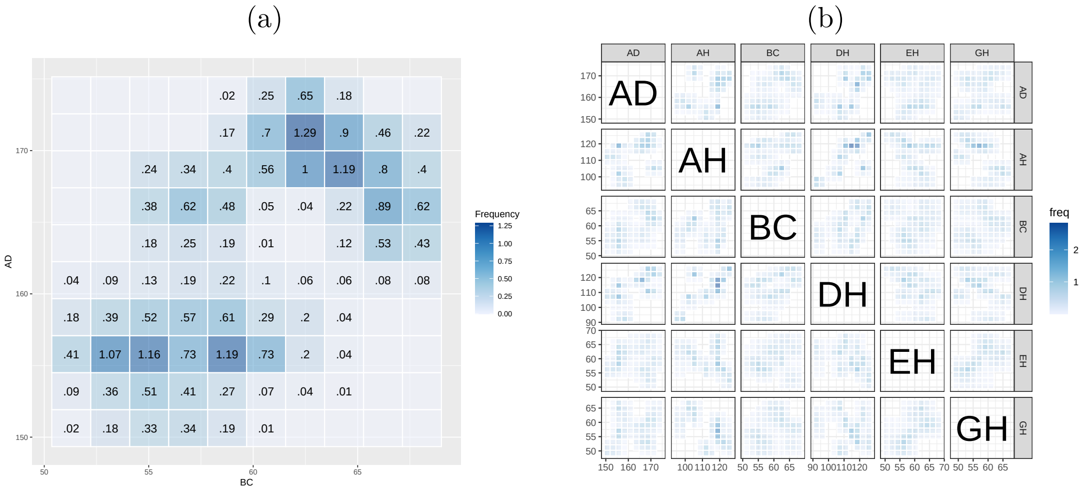


Figure 10: (a) The 2D histogram of the variables AD against BC of the face recognition data. (b) The 2D histogram matrix of the face recognition data.

obtained using the **ggplot2** layer functions `geom_text()` and `scale_fill_distiller()`. The 2D histogram in Figure 10(a) exhibits the same phenomenon as 2D scatterplots of variables AD and BC.

```
R> # Figure 10(a)
R> face.2dh <- ggInterval_2Dhist(facedata, aes(x = BC, y = AD, col = "white"),
+                                  xBins = 10, yBins = 10, addFreq = F)
R>
R> face.2dh$plot +
+    labs(fill = 'Frequency', title = "") +
+    coord_fixed(ratio = 1) +
+    scale_fill_distiller(palette = "Blues", direction = 1) +
+    geom_text(data = . %>% dplyr::filter(round(.data$freq, 2) != 0),
+              aes(x = (x1 + x2) / 2, y = (y1 + y2) / 2,
+                  label = round(freq, 2) %>% as.character %>%
+                    gsub(pattern = "^0", replacement = "")))
```

4.3. Multivariate plots

In order to visualize the multivariate data sets, we have implemented scatterplot matrices, 2D histogram matrices, radar plots, and image plots.

The scatterplot matrix The scatterplot matrix is commonly used to display multivariate data on two dimensions. It provides the bivariate relationships between combinations of variables using a grid (or matrix) of scatterplots. An individual scatterplot in the matrix displays a relationship between a pair of two variables, allowing the user to track a variety of relationships in one chart. As depicted in Figure 9(b), the scatterplot matrix for the six variables of the face data is obtained by the following code.

```
R> # Figure 9(b)
R> ggInterval_scaMatrix(data = facedata)
```

The interval-valued observations are represented by grayscale rectangles. Due to facial symmetry, the pairs {AH, DH} and {EH, GH} are strongly positively correlated. In addition, the nine subjects seem to be easily discriminated by variables AD and AH. By contrast, a negative correlation is observed between the pairs {AH, GH} and {DH, EH}, which are measures of two complementary distances on the opposite sides of a person's face.

The 2D histogram matrix A 2D histogram matrix displays bivariate relationships between variables using a grid of 2D histograms similar to a scatterplot matrix. As shown in Figure 10(b), the 2D histogram matrix for the six variables of the face data is generated by the function `ggInterval_2DhistMatrix()`, which yields similar results to those obtained by scatterplot matrices. `addFreq` and `removeZero` are logical arguments that control whether or not the frequencies display on rectangles with or without zeros.

```
R> # Figure 10(b)
> ggInterval_2DhistMatrix(facedata, aes(col = "white"),
+                          xBins = 10, yBins = 10,
+                          removeZero = T, addFreq = F) +
+   scale_fill_distiller(palette = "Blues", direction = 1)
```

The image plot The image plot, also known as the data image or the heatmap, is a dimensional-free method of visualizing data tables. Each column of the map represents the index image for the corresponding variable. Figures 11(a) and (b) depict the image plot for the face data using the function `ggInterval_indexImage()` with a predefined sequential color spectrum. Two heatmaps differ based on their display condition, namely the column and matrix display conditions, which is controlled by the logical argument `column_condition`.

```
R> # Figure 11(a)
R> ggInterval_indexImage(facedata, plotAll = T, full_strip = T,
+                         column_condition = F) +
+    scale_colour_distiller(palette = "Blues", direction = 1) +
+    labs(x = "Subjects") +
+    theme(axis.title.x=element_blank(), axis.text.x=element_blank(),
+        axis.ticks.x=element_blank())
R>
R> # Figure 11(b)
R> ggInterval_indexImage(facedata, plotAll = T, full_strip = T) +
+    scale_colour_distiller(palette = "Spectral") +
+    labs(x = "Subjects")
```

Using the column display condition, the entire color spectrum represents the full range of values for each variable. When variables in the input data set have large range differences, or when they are measured in different units, this display condition is considered to be adopted. When variables in the input data set have large range differences, or when they are measured in different units, this display condition is used. Visual investigation makes comparing

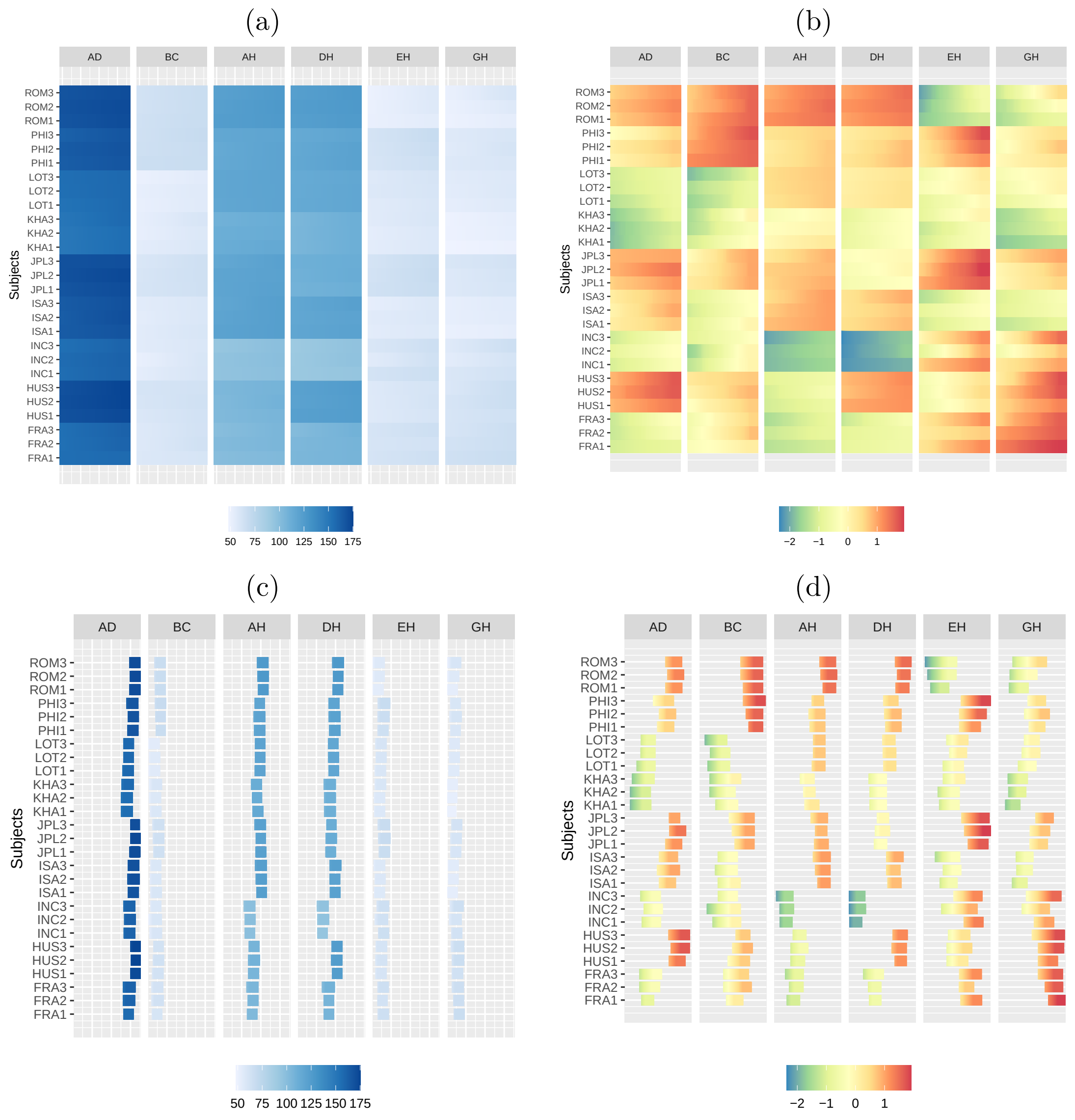


Figure 11: The image plot for the face recognition data with (a) matrix and (b) column display conditions. The non-equi-width image strips representation of the image plot with (c) matrix and (d) column display conditions.

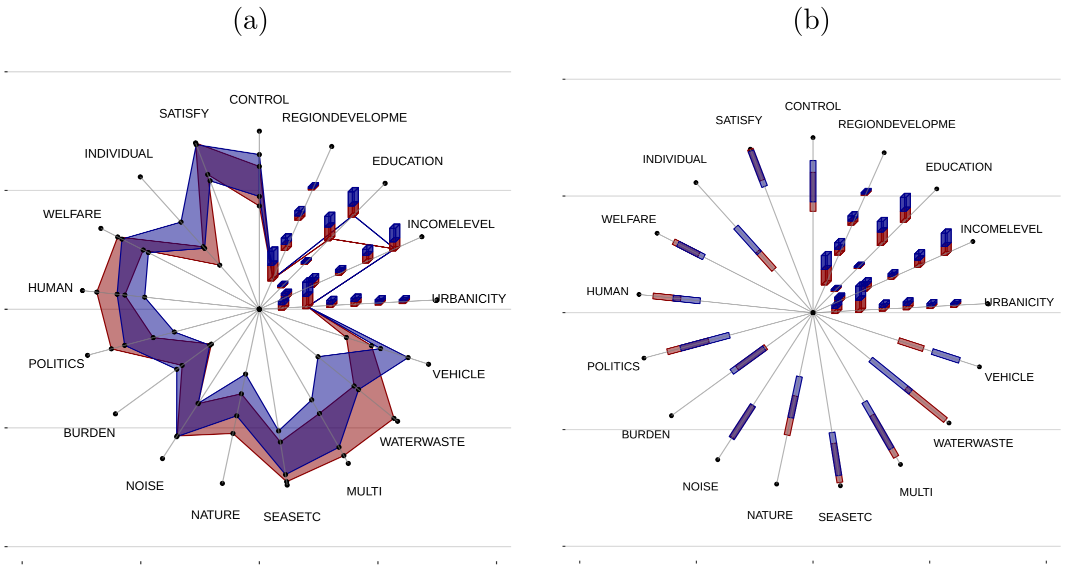


Figure 12: The radar plots of the Environmental questionnaire data. (a) The polygon representation. (b) The rectangle representation.

variable distributions easier. In the matrix display condition, the entire color spectrum represents the raw data matrix's full range of values. Display conditions exhibit the same effects as data transformations. For example, the column display condition is equivalent to the standardization of each variable.

Figure 11(c) and (d) illustrate alternative representations of the image plots in Figure 11(a) and (b), in which the color strips correspond to ranges of the intervals of different widths. This preference is controlled by the logical argument `full_strip = F`. Based on these image plots, we have observed similar results to those of the index plots in Figure 3. Heatmap heavily utilizes the colors to represent the numerical quantities and employs the effective ordering of rows and columns to discover patterns of data structure. It is therefore suggested that before using the image plot, suitable seriation and/or clustering algorithms for rows and columns of the table should be applied (Wu, Tien, and Chen 2010).

The radar plot A radar plot is also known as a spider plot or a star plot. It displays multivariate data as a two-dimensional chart with radial axes originating from the center for quantitative and qualitative variables. As a result, they can be used to identify which variables are scoring high, low, or similar, or whether any variables consist of outliers. It is straightforward to extend the radar plot to interval-valued and/or modal multi-valued data. For the interval-valued data, the minima and the maxima of each variable are plotted along their respective axes and all the minima, as well as all the maxima, are connected to form a filled area polygon. To represent modal multi-valued variables, the stacked bar is used. Bar segments represent percentages associated with each variable category.

Figure 12(a) shows the radar plot for the 4th and 6th symbolic objects in Environmental questionnaire data by the function `ggInterval_radar()` with the argument `plotPartial = c(4, 6)`. There are four probabilistic variables which are displayed as stacked bars in the chart. As a default, the [type] argument specifies the polygon representation for interval variables. When the `type = "rect"` is specified, intervals are represented as rectangles as in Figure 12(b). This is also known as the three-dimensional zoom star in Diday and Noirhomme-Fraiture (2008). Grid lines connecting axis-to-axis are often used as guides and can be controlled using the logical argument `base_circle`. The `inOneFig = TRUE` argument displays all observations on one chart, otherwise, each observation will be displayed on a separate radar plot.

```
R> # Figure 12(a)
R> ggInterval_radar(Environment, plotPartial = c(4, 6),
+                  showLegend = F, base_circle = F, base_lty = 1,
+                  addText = F, addText_modal = F) +
+    scale_fill_manual(values = c("darkred", "darkblue")) +
+    scale_color_manual(values = c("darkred", "darkblue")) +
+    labs(title = "") +
+    theme_hc()
```

The quantiles plot The quantiles plot in our implementation is simply derived from the radar plot by calling the function `ggInterval_radar()` with argument `type = "quantile"`, and specifying the number of quantiles using `quantileNum`. Figure 13(a) and (b) show the quantile plots for the face recognition data and the Environmental questionnaire data with interval-valued variables only. Each variable's distribution can be easily visualized and compared. As an example, AH in the face data exhibits a left-skewed distribution, while POLITICS in the Environmental questionnaire data suggests a left-skewed distribution.

```
R> # Figure 13(a)
R> ggInterval_radar(facedata, plotPartial = 1, base_circle = F,
+                  base_lty = 1, type = "quantile", quantileNum = 5,
+                  showLegend = T, Drift = 0) +
+    scale_fill_brewer(palette = "Greys") +
+    labs(title = "", fill = "Quantiles") +
+    theme_hc()
```

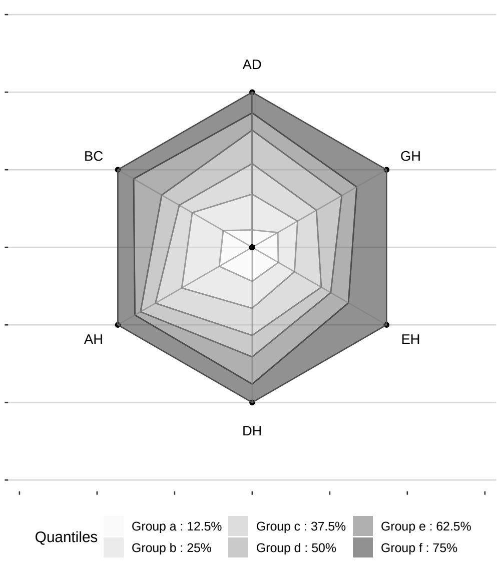
5. Generalization and other functions

5.1. Working with ggplot2 for customized plots

Package **ggESDA** is an extension of **ggplot2**. Most of the features available in **ggplot2** can be adopted by **ggESDA** to create more advanced graphs. Below is an example code showing the customized index plot for the variable AD of face data based on the K-means results produced by `ggInterval_index()`. Figure 14 shows the resulting plot that uses the functions `geom_text()`, `labs()`, `scale_fill_brewer()`, `facet_grid()`, and `theme()`.

```
R> # Figure 14
R> set.seed(1234567890)
```

(a)



(b)

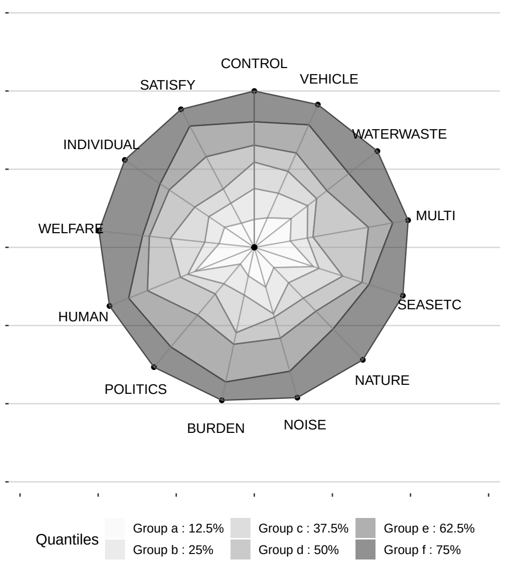


Figure 13: The quantile plots for the face recognition data and the Environmental questionnaire data (interval-valued variables only).

```
R> facedata.tmp <- facedata
R> facedata.tmp$cluster <- sym.kmeans(facedata.tmp, k = 3)$cluster
R> ggInterval_index(facedata.tmp, aes(y = AD, fill = Subjects)) +
+    geom_text(aes(x = 1:f.nr, y = .data$AD$min,
+                  label = rownames(facedata), vjust = 1.5), size = 2) +
+    scale_fill_brewer(palette = "Set1") +
+    labs(title = "Customize Index Plot using K-Menas Results",
+         fill = "Subjects") +
+    facet_grid(cols = vars(facedata.tmp$cluster),
+               scales = "free_x", space = "free_x") +
+    theme_economist_white() +
+    theme(axis.title.x=element_blank(), axis.text.x=element_blank(),
+          axis.ticks.x=element_blank())
```

As we can see that Section 4 has give some examples that illustrate the more advanced aesthetically pleasing features of **ggplot2**. These features are beneficial to **ggESDA**. Below are some remarks.

Geometries The common visual representations of **ggplot2** geometries can easily be applied to the existing graphic created by **ggESDA** such as `geom_area()`, `geom_line()`, `geom_point()`, `geom_segment()` and `geom_text()`.

Scale The **ggESDA** package allows natural use of `scale_*` functions in **ggplot2**, such as `scale_(x|y)_*` and `scale_(fill|colour)_*`.

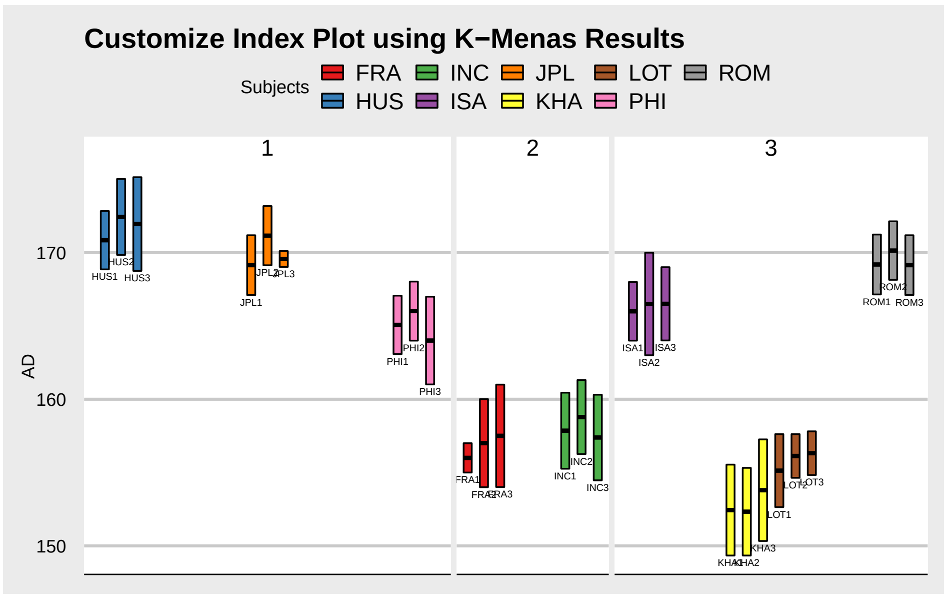


Figure 14: A customized index plot for the variable AD of face recognition data based on the K-means results.

Theme It is possible to apply themes from **ggplot2**, **ggthemes** (Arnold 2021) or **ggpubr** (Kassambara 2020) to the graph created by **ggESDA**.

Facets The developed plot functions such as `ggInterval_index()`, `ggInterval_centerRange()`, `ggInterval_minmax()`, and `ggInterval_scatter()` are compatible with the faceting function of **ggplot2**. Faceting by either `facet_grid()` or `facet_wrap()` creates multiple subplots with different subsets of the data. `facet_grid()` creates a matrix of panels based on both row and column faceting variables, while `facet_wrap()` uses only one variable with a number of levels.

While most additional layers or scales can be added to the existing **ggESDA** graph, there are still some cases where they may conflict with the native **ggplot2**. The complexity of the data type may cause some problems and are difficult to combine with the default aesthetic. Examples include `geom_bar()`, `geom_count()`, `geom_density()`, `geom_dotplot()`, and so on. Most of the implemented graphics suffer from this issue are resolved by producing a new plot rather than adding an additional layer.

5.2. Working with other SDA packages

The symbolic objects resulting from the SDA-related packages may exist in a variety of formats. To increase the usability of those packages with **ggESDA**, we provide a transformation function `classic2sym()` so that the symbolic objects obtained from them could be converted and visualized by **ggESDA**. To illustrate this, we take a look at two SDA packages: **HistDAWass** (Irpino 2021) and **MAINT.Data** (Silva and Brito 2021).

HistDAWass The built-in data set, `BLOOD` data, in **HistDAWass** package is a histogram-valued data set. It consists of 14 groups of patients described by 3 variables. Using `get.MatH.stats()`

function we can obtain the minimum and maximum of each histogram. We then convert them into intervals by using the `classic2sym()` function to make them ready for use in **ggESDA**.

```
R> library(HistDAWass)
R> BLOOD <- HistDAWass::BLOOD
R> blood.min <- get.MatH.stats(BLOOD, stat = "min")
R> blood.max <- get.MatH.stats(BLOOD, stat = "max")
R> blood <- data.frame(blood.min, blood.max)
R> blood.sym <- classic2sym(blood, groupby = "customize",
+    minData = blood[, 2:4], maxData = blood[, 6:8])
R> blood.i <- blood.sym$intervalData
R> colnames(blood.i) <- get.MatH.main.info(BLOOD)$varnames
R> head(as.data.frame(blood.i), 5)
```

```
  Cholesterol      Hemoglobin      Hematocrit
1 [80.00 : 240.00] [12.00 : 15.00] [35.00 : 47.00]
2 [80.00 : 240.00] [10.50 : 14.00] [31.00 : 44.00]
3 [95.00 : 245.00] [10.50 : 14.00] [31.00 : 43.50]
4 [105.00 : 260.00] [10.50 : 14.00] [31.00 : 42.50]
5 [115.00 : 260.00] [10.80 : 13.60] [31.00 : 42.50]
```

MAINT.Data Interval-valued data can also be represented as midpoints and ranges. An example of this is the built-in data set `Abalone` in the package **MAINT.Data**. This dataset consists of 24 observations and 7 interval-valued variables including midpoints and log-ranges. The intervals were obtained by aggregating the Abalone dataset (UCI Machine Learning Repository) based on sex and age factors. With `classic2sym()`, we convert them back to intervals of lower and upper bounds.

```
R> library(MAINT.Data)
R> AbaloneIdt <- MAINT.Data::AbaloneIdt
R> a.range <- exp(AbaloneIdt@LogR)
R> a.midp <- AbaloneIdt@MidP
R> abalone <- data.frame(a.midp - a.range / 2, a.midp + a.range / 2)
R> abalone.sym <- classic2sym(abalone, groupby = "customize",
+    minData = abalone[, 1:7], maxData = abalone[, 8:14])
R> colnames(abalone.sym$intervalData) <- AbaloneIdt@VarNames
R> abalone.i <- abalone.sym$intervalData %>%
+    cbind(Aba.obs = AbaloneIdt@ObsNames) %>%
+    column_to_rownames(var = "Aba.obs")
R> head(abalone.i[, 1:4], 5)
```

```
        Length          Diameter       Height         Whole_weight
F-10-12 [0.34 : 0.78] [0.26 : 0.63] [0.06 : 0.23] [0.21 : 2.66]
F-13-15 [0.39 : 0.82] [0.30 : 0.65] [0.10 : 0.25] [0.27 : 2.51]
F-16-18 [0.40 : 0.74] [0.32 : 0.60] [0.10 : 0.24] [0.35 : 2.20]
F-19-21 [0.49 : 0.72] [0.36 : 0.58] [0.12 : 0.21] [0.68 : 2.12]
F-23-24 [0.45 : 0.80] [0.38 : 0.63] [0.14 : 0.22] [0.64 : 2.53]
```

5.3. Visualizing intervals in 2D Principal Component space

If the result of a statistical or machine learning method is interval-valued, the **ggESDA** package provides a convenient way to visualize it. The extension of PCA to interval-valued data is one of such examples (Lauro *et al.* 2008; Douzal-Chouakria *et al.* 2011). Using PCA, users can explore the data structure by projecting interval-valued data into a two- or three-dimensional DR subspace. There have been several approaches proposed to visualize interval objects in lower-dimensional subspaces. Our study implements two types of them, the maximum covering area rectangle (MCAR) and the polytopes representation. In the following, we show the projected intervals of face data in the first DR plane. The interval PCA is performed by the R function `sym.pca()` of the **RSDA** package.

The maximum covering area rectangle The maximum covering area rectangle (Cazes, Chouakria, Diday, and Schektman (1997)) is widely used for graphical representation of interval objects on a DR subspace owing to its simplicity. It constructs observations as k-dimensional hyperrectangles (usually $k = 2$) on a DR subspace, where the sides of hyperrectangles are parallel to the axes. MCAR's main disadvantage is that interval objects are over-sized in DR space compared to real objects in $\mathbb{R}^p$. Hence, a hyperrectangle in DR space may include data points that were not included in the original observations. Figure 15(a) shows the 2D projections of the vertices method of PCA to the face data based on the MCAR representation.

```
R> # Figure 15(a)
R> pca.results <- RSDA::sym.pca(facedata, method = "tops")
R> rownames(pca.results$Sym.Components) <- rownames(facedata)
R> ggInterval_scatter(pca.results$Sym.Components, aes(Dim.1, Dim.2,
+                      fill = as.factor(Subjects)), color = "black") +
+    coord_fixed(ratio = 1) +
+    labs(x = "PCA-1", y = "PCA-2", fill = "Subjects")
```

The polytopes representation The symbolic covariance method of interval PCA with the polytopes representation for interval data was proposed by Le-Rademacher and Billard (2012). The vertices of interval-valued data are projected onto lower-dimensional subspace, and the edges of points are formed in the DR subspace by connecting the vertices, as in the original input space $\mathbb{R}^p$. The polytopes representation improves MCAR and provides a true projection of the observed interval data. Figure 15(b) presents the results of the 2D projection of the symbolic covariance method of PCA to the face data based on the polytope representation, which is a reproduction of Figure 4(a) in Le-Rademacher and Billard (2012).

```
R> # Figure 15(b)
R> ggInterval_PCA(facedata, poly = T, concepts_group = as.factor(Subjects)) +
+   coord_fixed(ratio = 1) +
+   labs(x = "PCA-1", y = "PCA-2")
```

6. Conclusion and future development

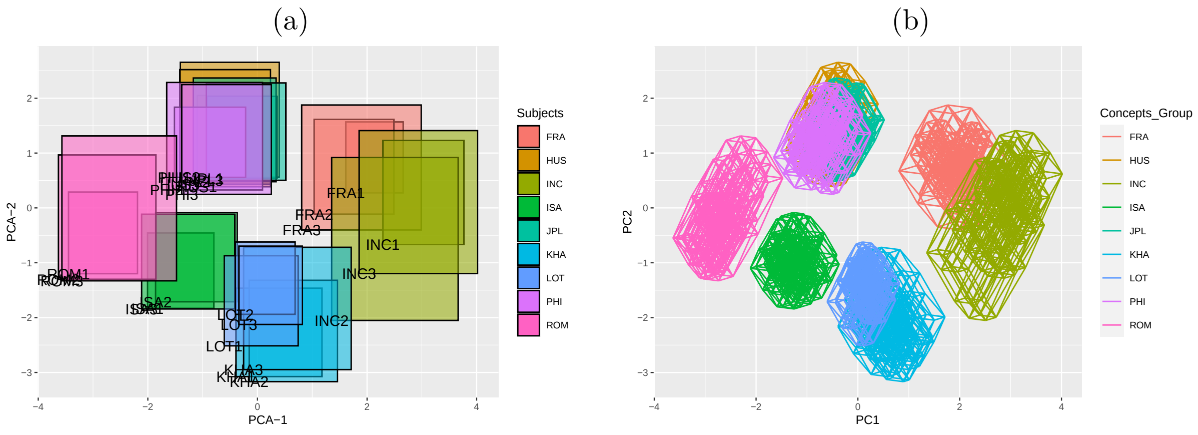


Figure 15: A 2D projection of interval PCA to the recognition face: (a) the vertices method with MCAR representation, and (b) the symbolic covariance method with the polytopes representation.

The EDA and data visualization processes help users discover hidden patterns within data. Without graphical exploration, analyzing data solely with numerical statistics may be misleading. As noted in Ripley (2005) "finding ways to visualize data sets can be as important as ways to analyze them" and Borcard, Gillet, and Legendre (2011) "EDA is often neglected by people who are eager to jump to more sophisticated analyses. It should have an important place." the development of EDA and visualization tools for investigating symbolic data is essential. In this study, we developed the R package **ggESDA** to complement the **ggplot2** package for exploratory symbolic data analysis, focusing on the interval-valued data. Among SDA-related R packages, its development stands out as unique. We have demonstrated its ability and functions through the graphing of several commonly used interval-valued data sets. Several multivariate graphical techniques and methodological improvements will be incorporated into the package in the next version, including the revision of 3D scatter plots, interactive tools, etc. We believe **ggESDA** will bring a valuable contribution to the SDA community and broaden participation in data science fields.

Big data, on the other hand, failed almost all of the traditional graphical techniques of EDA due to the volume of data. The statisticians are challenged with analyzing big data gathered rapidly from diverse resources with complex types. As a result, it is imperative to develop new, more efficient statistical and graphical methods for analyzing these data. In this regard, the SDA has great capabilities by aggregating them into interval-valued, histogram-valued, or distribution-valued data of a more manageable size. **ggESDA** can then be applied to them for a quick and initial exploration.

Computational details

The results in this paper were obtained using R 4.1.2 (R Core Team 2021). R itself and all packages used are available from the CRAN at `https://CRAN.R-project.org`. The **ggESDA** package is now available from the CRAN at `https://cran.r-project.org/web/packages/ggESDA/index.html`.

References

Arnold JB (2021). *ggthemes: Extra Themes, Scales and Geoms for ggplot2*. R package version 4.2.4, URL `https://github.com/jrnold/ggthemes`.

Bertrand P, Goupil F (2000). "Descriptive Statistics for Symbolic Data." In *In: Bock, HH., Diday, E. (eds) Analysis of Symbolic Data. Studies in Classification, Data Analysis, and Knowledge Organization*, pp. 106–124. Springer, Berlin, Heidelberg.

Billard L (2008). "Sample Covariance Functions for Complex Quantitative Data." In *Proceedings of World IASC Conference, Yokohama, Japan*, pp. 157–163.

Billard L, Diday E (2000). "Regression Analysis for Interval-valued Data." In *Data Analysis, Classification, and Related Methods*, pp. 369–374. Springer.

Billard L, Diday E (2003). "From The Statistics of Data to The Statistics of Knowledge: Symbolic Data Analysis." *Journal of the American Statistical Association*, **98**(462), 470–487.

Billard L, Diday E (2007). *Symbolic Data Analysis: Conceptual Statistics and Data Mining*. Wiley, New Jersey.

Borcard D, Gillet F, Legendre P (2011). *Numerical Ecology with R*. Springer, New York, NY.

Cazes P, Chouakria A, Diday E, Schektman Y (1997). "Extension de l'analyse en composantes principales à des données de type intervalle." *Revue de Statistique appliquée*, **45**(3), 5–24.

Cui B (2020). *DataExplorer: Automate Data Exploration and Treatment*. R package version 0.8.2, URL `http://boxuancui.github.io/DataExplorer/`.

Dayanand Ubrangala, R K, Prasad Kondapalli R, Putatunda S (2021). *SmartEDA: Summarize and Explore the Data*. R package version 0.3.8, URL `https://daya6489.github.io/SmartEDA/`.

Diday E, Noirhomme-Fraiture M (2008). *Symbolic Data Analysis and The Sodas Software*. John Wiley & Sons.

Domingues MA, de Souza RM, Cysneiros FJA (2010). "A Robust Method for Linear Regression of Symbolic Interval Data." *Pattern Recognition Letters*, **31**(13), 1991–1996.

Douzal-Chouakria A, Billard L, Diday E (2011). "Principal Component Analysis for Interval-valued Observations." *Statistical Analysis and Data Mining*, **4**(2), 229–246.

Hamilton NE, Ferry M (2018). "**ggtern**: Ternary Diagrams Using ggplot2." *Journal of Statistical Software, Code Snippets*, **87**(3), 1–17. `doi:10.18637/jss.v087.c03`.

Irpino A (2021). *HistDAWass: Histogram-Valued Data Analysis*. R package version 1.0.6, URL `https://CRAN.R-project.org/package=HistDAWass`.

Irpino A, Verde R (2015). "Basic Statistics for Distributional Symbolic Variables: A New Metric-based Approach." *Advances in Data Analysis and Classification volume*, **9**, 143–175.

Kassambara A (2020). *ggpubr: ggplot2 Based Publication Ready Plots.* R package version 0.4.0, URL `https://rpkgs.datanovia.com/ggpubr/`.

Lauro CN, Palumbo F (2000). "Principal Component Analysis of Interval Data: A Symbolic Data Analysis Approach." *Computational statistics*, **15**(1), 73–87.

Lauro NC, Verde R, Irpino A (2008). "Principal Component Analysis of Symbolic Data Described by Intervals." *Symbolic Data Analysis and the SODAS Software, ed. E. Diday and M. Noirhomme-Fraiture*, pp. 279–312.

Le-Rademacher J, Billard L (2012). "Symbolic Covariance Principal Component Analysis and Visualization for Interval-valued Data." *Journal of Computational and Graphical Statistics*, **21**(2), 413–432.

Leroy B, Chouakria A, Herlin I, Diday E (1996). "Approche géométrique et classification pour la reconnaissance de visage." In *Congrès de Reconnaissance des formes et Intelligence Artificielle*, pp. 548–557.

Maag JL (2018). "**gganatogram**: An R Package for Modular Visualisation of Anatograms and Tissues Based on ggplot2." *F1000Research*, **7**, 1576.

Malerba D, Esposito F, Gioviale V, Tamma V (2001). "Comparing Dissimilarity Measures for Symbolic Data Analysis." *Proceedings of Exchange of Technology and Know-how and New Techniques and Technologies for Statistics*, **1**, 473–481.

Neto EdAL, Cordeiro GM, de Carvalho FdA (2011). "Bivariate Symbolic Regression Models for Interval - valued Variables." *Journal of Statistical Computation and Simulation*, **81**(11), 1727–1744.

Patil I (2021). "Visualizations with statistical details: The 'ggstatsplot' approach." *Journal of Open Source Software*, **6**(61), 3167. `doi:10.21105/joss.03167`. URL `https://doi.org/10.21105/joss.03167`.

R Core Team (2021). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. URL `https://www.R-project.org/`.

Ripley BD (2005). "How Computing has Changed Statistics." *in Celebrating Statistics: Papers in Honour of Sir David Cox on His 80th Birthday, eds A. Davison, Y. Dodge, and N. Wermuth*, pp. 197–212.

Rodriguez O (2022). *RSDA: R to Symbolic Data Analysis.* R package version 3.0.12, URL `http://www.oldemarrodriguez.com`.

Schloerke B, Cook D, Larmarange J, Briatte F, Marbach M, Thoen E, Elberg A, Crowley J (2021). *GGally: Extension to ggplot2.* R package version 2.1.2, URL `https://CRAN.R-project.org/package=GGally`.

Silva PD, Brito P (2021). *MAINT.Data: Model and Analyse Interval Data.* R package version 2.6.2, URL `https://CRAN.R-project.org/package=MAINT.Data`.

Tukey JW (1977). *Exploratory Data Analysis.* Reading, Massachusetts: Addison-Wesley.

Umbleja K, Ichino M, Yaguchi H (2020). "Improving Symbolic Data Visualization for Pattern Recognition and Knowledge Discovery." *Visual Informatics*, **4**(1), 23–31.

Wickham H (2016). ***ggplot2****: Elegant Graphics for Data Analysis*. Springer-Verlag New York. ISBN 978-3-319-24277-4. URL `https://ggplot2.tidyverse.org`.

Wu HM, Tien YJ, Chen CH (2010). "GAP: A Graphical Environment for Matrix Visualization and Cluster Analysis." *Computational Statistics and Data Analysis*, **54**(2), 767–778.

Xu W (2010). *Symbolic Data Analysis: Interval-valued Data Regression*. Ph.D. thesis, University of Georgia Athens, GA.

Yoshida K, Bartel A (2022). *tableone: Create Table 1 to Describe Baseline Characteristics with or without Propensity Score Weights*. R package version 0.13.2, URL `https://github.com/kaz-yos/tableone`.

Affiliation:

Wu, Han-Ming
Faculty of Department of Statistics
National Chengchi University
No. 64, Sec. 2, ZhiNan Rd., Wenshan District,
Taipei City 11605, Taiwan
E-mail: `wuhm@g.nccu.edu.tw`
URL: `http://www.hmwu.idv.tw`

Journal of Statistical Software `http://www.jstatsoft.org/`
published by the Foundation for Open Access Statistics `http://www.foastat.org/`
MMMMMM YYYY, Volume VV, Issue II *Submitted:* yyyy-mm-dd
`doi:10.18637/jss.v000.i00` *Accepted:* yyyy-mm-dd