Package ‘ggInterval’

February 23, 2026

Type Package

Title Visualizing Interval-Valued Data using 'ggplot2'

Version 0.2.4

Date 2026-02-19

Author Bo-Syue Jiang [aut],
Han-Ming Wu [cre]

Maintainer Han-Ming Wu <wuhm@g.nccu.edu.tw>

Description Implements an extension of 'ggplot2' (formerly 'ggESDA') and visualizes symbolic interval-valued data with various plots, offering more general and flexible input arguments. Additionally, it provides a function to transform classical data into symbolic data using both clustering algorithms and customized methods.

Depends ggplot2, R (>= 4.4.0), tidyverse, RSDA

Suggests testthat (>= 2.1.0), knitr, rmarkdown

Imports rlang, R6, dplyr, tidyr, gridExtra, gtools, stringr, prodlim,
ggforce, ggpubr, ggthemes, tibble, magrittr, vctrs

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

VignetteBuilder knitr

URL https://github.com/hanmingwu1103/ggInterval,
https://CRAN.R-project.org/package=ggInterval

BugReports https://github.com/hanmingwu1103/ggInterval/issues

NeedsCompilation no

Repository CRAN

Date/Publication 2026-02-23 16:50:07 UTC

Contents

`AbaloneIdt` *AbaloneIdt data example*

Description

AbaloneIdt interval data example.

Usage

```
data(AbaloneIdt)
```

Format

An object of class `data.frame` (inherits from `symbolic_tbl`) with 24 rows and 7 columns.

References

Billard L. and Diday E. (2006).Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Examples

```
data(AbaloneIdt)
ggInterval_indexplot(AbaloneIdt, aes(x = Length))
```

`BLOOD` *BLOOD data example*

Description

BLOOD interval data example.

Usage

```
data(BLOOD)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`, `symbolic_tbl`) with 14 rows and 3 columns.

References

Billard L. and Diday E. (2006).Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Examples

```
data(BLOOD)
ggInterval_MMplot(BLOOD, aes(x = Hematocrit))
```

`Cardiological` *Cardiological data example*

Description

Cardiological interval data example.

Usage

```
data(Cardiological)
```

Format

An object of class `symbolic_tbl` (inherits from `tbl_df`, `tbl`, `data.frame`) with 11 rows and 3 columns.

Source

https://CRAN.R-project.org/package=RSDA

References

Billard L. and Diday E. (2006).Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Examples

```
data(Cardiological)
ggInterval_indexplot(Cardiological, aes(x = Syst))
```

`Cardiological2` *Cardiological data example*

Description

Cardiological interval data example.

Usage

```
data(Cardiological2)
```

Format

An object of class `symbolic_tbl` (inherits from `tbl_df`, `tbl`, `data.frame`) with 15 rows and 3 columns.

References

Billard L. and Diday E. (2006).Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Examples

```
data(Cardiological2)
ggInterval_indexplot(Cardiological2, aes(x = Syst))
```

`classic2sym` *Convert classical data frame into a symbolic data.*

Description

A function for converting a classical data, which may present as a data frame or a matrix with one entry one value, into a symbolic data,which is shown as a interval or a set in an entry. Object after converting is ggInterval class containing interval data and raw data(if it exist) and typically statistics.

Usage

```
classic2sym(data=NULL,groupby = "kmeans",k=5,minData=NULL,maxData=NULL,
modalData = NULL)
```

Arguments

`data`	A classical data frame that you want to be converted into a interval data
`groupby`	A way to aggregate. It can be either a clustering method or a variable name which exist in input data (necessary factor type) . Default "kmeans".
`k`	A number of group,which is used by clustering. Default k = 5.
`minData`	if choose groupby parameter as 'customize',user need to define which data is min data or max data.
`maxData`	if choose groupby parameter as 'customize',user need to define which data is min data or max data.
`modalData`	list, each cell of list contain a set of column index of its modal multi-valued data of the input data. the value of it is a proportion presentation, and sum of each row in these column must be equal to 1. ex 0,1,0 or 0.2,0.3,0.5. the input type of modalData for example is modalData[[1]] = c(2, 3), modalData[[2]] = c(7:10), that 2, 3, 7, 8, 9, 10 columns are modal type of the data. Note: the option is only valid when groupby == "customize".

Value

classic2sym returns an object of class "ggInterval",which have a interval data and others as follows.

- intervalData - The Interval data after converting also known as a RSDA object.
- rawData - Classical data that user input.
- clusterResult - Cluster results .If the groupby method is a clustering method then it will exist.
- statisticsDF - A list contains data frame including some typically statistics in each group.

Examples

```
#classical data to symbolic data
classic2sym(iris)
classic2sym(mtcars, groupby = "kmeans", k = 10)
classic2sym(iris, groupby = "hclust", k = 7)
classic2sym(iris, groupby = "Species")

x1<-runif(10, -30, -10)
y1<-runif(10, -10, 30)
x2<-runif(10, -5, 5)
y2<-runif(10, 10, 50)
x3<-runif(10, -50, 30)
y3<-runif(10, 31, 60)

d<-data.frame(min1=x1,max1=y1,min2=x2,max2=y2,min3=x3,max3=y3)
classic2sym(d,groupby="customize",minData=d[,c(1,3,5)],maxData=d[,c(2,4,6)])
classic2sym(d,groupby="customize",minData=d$min1,maxData=d$min2)


#example for build modal data
#for the first modal data proportion
a1 <- runif(10, 0,0.4) %>% round(digits = 1)
a2 <- runif(10, 0,0.4) %>% round(digits = 1)

#for the second modal data proportion
b1 <- runif(10, 0,0.4) %>% round(digits = 1)
b2 <- runif(10, 0,0.4) %>% round(digits = 1)

#for interval-valued data
c1 <- runif(10, 10, 20) %>% round(digits = 0)
c2 <- runif(10, -50, -10) %>% round(digits = 0)

#build simulated data
d <- data.frame(a1 = a1, a2 = a2, a3 = 1-(a1+a2),
                c1 = c1, c2 = c2,
                b1 = b1, b2 = b2, b3 = 1-(b1+b2))

#transformation
classic2sym(d, groupby = "customize",
            minData = d$c2,
            maxData = d$c1,
            modalData = list(1:3, 6:8))#two modal data
```

```
#extract the data
symObj<-classic2sym(iris)
symObj$intervalData      #interval data
symObj$rawData           #raw data
symObj$clusterResult     #cluster result
symObj$statisticsDF      #statistics
```

`cor` *Generic function for the correlation*

Description

This function compute the symbolic correlation

Usage

```
cor(x, ...)

## Default S3 method:
cor(
  x,
  y = NULL,
  use = "everything",
  method = c("pearson", "kendall", "spearman"),
  ...
)

## S3 method for class 'symbolic_tbl'
cor(x, ...)

## S3 method for class 'symbolic_interval'
cor(x, y, method = c("centers", "B", "BD", "BG"), ...)
```

Arguments

`x`	First symbolic variables.
`...`	As in R cor function.
`y`	Second symbolic variables.
`use`	an optional character string giving a method for computing correlation in the presence of missing values. This must be (an abbreviation of) one of the strings 'everything', 'all.obs', 'complete.obs', 'na.or.complete', or 'pairwise.complete.obs'.
`method`	The method to be use.

Value

Return a real number.

Author(s)

Oldemar Rodriguez Rojas

References

Billard L. and Diday E. (2006). Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Rodriguez, O. (2000). Classification et Modeles Lineaires en Analyse des Donnees Symboliques. Ph.D. Thesis, Paris IX-Dauphine University.

cov *Generic function for the covariance*

Description

This function compute the symbolic covariance.

Usage

```
cov(x, ...)

## Default S3 method:
cov(
  x,
  y = NULL,
  use = "everything",
  method = c("pearson", "kendall", "spearman"),
  ...
)

## S3 method for class 'symbolic_tbl'
cov(x, ...)

## S3 method for class 'symbolic_interval'
cov(x, y = NULL, method = c("centers", "B", "BD", "BG"), na.rm = FALSE, ...)
```

Arguments

`x`	First symbolic variables.
`...`	As in R cov function.
`y`	Second symbolic variables.
`use`	an optional character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings 'everything', 'all.obs', 'complete.obs', 'na.or.complete', or 'pairwise.complete.obs'.
`method`	The method to be use.
`na.rm`	As in R cov function.

Value

Return a real number.

Author(s)

Oldemar Rodriguez Rojas

References

Billard L. and Diday E. (2006). Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Rodriguez, O. (2000). Classification et Modeles Lineaires en Analyse des Donnees Symboliques. Ph.D. Thesis, Paris IX-Dauphine University.

`Environment` *Environment data example*

Description

Environment interval and modal data example.

Usage

```
data(Environment)
```

Format

An object of class `symbolic_tbl` (inherits from `tbl_df`, `tbl`, `data.frame`) with 14 rows and 17 columns.

Examples

```
data(Environment)
ggInterval_radarplot(Environment,
 plotPartial = 2,
 showLegend = FALSE,
 base_circle = TRUE,
 base_lty = 2,
 addText = FALSE)
```

`facedata` *Face Data Example*

Description

Symbolic data matrix with all the variables of interval type.

Usage

```
data('facedata')
```

Format

$I;AD;AD;$I;BC;BC;.........

HUS1;$I;168.86;172.84;$I;58.55;63.39;.........
HUS2;$I;169.85;175.03;$I;60.21;64.38;.........
HUS3;$I;168.76;175.15;$I;61.4;63.51;.........
INC1;$I;155.26;160.45;$I;53.15;60.21;.........
INC2;$I;156.26;161.31;$I;51.09;60.07;.........
INC3;$I;154.47;160.31;$I;55.08;59.03;.........
ISA1;$I;164;168;$I;55.01;60.03;.........
ISA2;$I;163;170;$I;54.04;59;.........
ISA3;$I;164.01;169.01;$I;55;59.01;.........
JPL1;$I;167.11;171.19;$I;61.03;65.01;.........
JPL2;$I;169.14;173.18;$I;60.07;65.07;.........
JPL3;$I;169.03;170.11;$I;59.01;65.01;.........
KHA1;$I;149.34;155.54;$I;54.15;59.14;.........
KHA2;$I;149.34;155.32;$I;52.04;58.22;.........
KHA3;$I;150.33;157.26;$I;52.09;60.21;.........
LOT1;$I;152.64;157.62;$I;51.35;56.22;.........
LOT2;$I;154.64;157.62;$I;52.24;56.32;.........
LOT3;$I;154.83;157.81;$I;50.36;55.23;.........
PHI1;$I;163.08;167.07;$I;66.03;68.07;.........
PHI2;$I;164;168.03;$I;65.03;68.12;.........
PHI3;$I;161.01;167;$I;64.07;69.01;.........
ROM1;$I;167.15;171.24;$I;64.07;68.07;.........
ROM2;$I;168.15;172.14;$I;63.13;68.07;.........
ROM3;$I;167.11;171.19;$I;63.13;68.03;.........

Source

https://CRAN.R-project.org/package=RSDA

References

Billard L. and Diday E. (2006). Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Examples

```
data(facedata)
ggInterval_hist(facedata, aes(x = AD))
```

`ggInterval` *A symbolic object by R6 class for interval analysis and ggplot*

Description

This is an object that will be used to make a ggplot object.A ggInterval object contains both classic data that user have and interval data which we transform.More over,some basic statistics from row data will also be recorded in this object,and the interval data which is from RSDA transformation will still contain RSDA properties.

Public fields

`rawData` the data from user.

`statisticsDF` contains min max mean median dataframe for each group of symbolic data

`intervalData` interval data from RSDA type

`clusterResult` clustering result

Methods

Public methods:

- `ggInterval$new()`
- `ggInterval$clone()`

Method `new()`**:** initialize all data, check whether satisfy theirs form

Usage:

```
ggInterval$new(
  rawData = NULL,
  statisticsDF = NULL,
  intervalData = NULL,
  clusterResult = NULL
)
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
ggInterval$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`ggInterval_2Dhist` *visualize a 2-dimension histogram by symbolic data with ggplot package.*

Description

Visualize the two continuous variable distribution by dividing both the x axis and y axis into bins,and calculating the frequency of observation interval in each bin.

Usage

```
ggInterval_2Dhist(data = NULL,mapping = aes(NULL)
,xBins = 14,yBins=16,removeZero = FALSE,
addFreq = TRUE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame,which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`xBins`	x axis bins, which mean how many partials x variable will be separate into.
`yBins`	y axis bins.It is the same as xBins.
`removeZero`	whether remove data whose frequency is equal to zero
`addFreq`	where add frequency text in each cells.

Value

Return a ggplot2 object.

Examples

```
ggInterval_2Dhist(oils, aes(x = GRA, y = FRE),
  xBins = 5, yBins = 5)
```

`ggInterval_2DhistMatrix` *2-Dimension histogram matrix*

Description

Visualize the all continuous variable distribution by dividing both the x axis and y axis into bins,and calculating the frequency of observation interval in each bin.Eventually show it by a matrix plot. Note: this function will automatically filter out the discrete variables,and plot all continuous in input data, so it can not be necessary that give the particularly variables in aes such like (aes(x = x, y = y)). It isn't also recommended to deal with too many variables because the big O in calculating full matrix will be too large.

Usage

```
ggInterval_2DhistMatrix(data = NULL,mapping = aes(NULL)
,xBins = 8,yBins=8,removeZero = FALSE,
addFreq = TRUE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame, which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`xBins`	x axis bins,which mean how many bins x variable will be separate into
`yBins`	y axis bins. It is the same as xBins
`removeZero`	whether remove data whose frequency is equal to zero
`addFreq`	where add frequency text in each cells.

Value

Return a plot with ggplot2 object

Examples

```
ggInterval_2DhistMatrix(oils, xBins = 5, yBins = 5)
```

`ggInterval_3Dscatterplot` *3D scatter plot for interval data*

Description

Visualize the three continuous variable distribution by collecting all vertices in each interval to form a shape of cube.Also show the difference between each group.

Usage

```
ggInterval_3Dscatterplot(data = NULL,mapping = aes(NULL),scale=FALSE)
```

Arguments

`data`	A ggSDA object. It can also be either RSDA object or classical data frame, which will be automatically convert to ggSDA data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`scale`	A boolean variable,TRUE, standardlize data. FALSE, not standardlize. If variance is too large(or small) or the difference between two variables are too large,it will be distortion or unseeable,which may happen in different units or others. So, a standardlize way is necessary.

Value

Return a ggplot2 object (It will still be 2-Dimension).

Examples

```
ggInterval_3Dscatterplot(facedata[1:5, ], aes(x = BC, y = EH, z = GH))
```

`ggInterval_boxplot` *A interval Box plot*

Description

Visualize the one continuous variable distribution by box represented by multiple rectangles.

Usage

```
ggInterval_boxplot(data = NULL,mapping = aes(NULL),plotAll=FALSE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame, which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`plotAll`	booleans, if TRUE, plot all variable together

Value

Return a ggplot2 object.

Examples

```
p<-ggInterval_boxplot(iris,aes(iris$Petal.Length))
p
p+scale_fill_manual(values = c("red","yellow",
    "green","blue","black"),
    labels=c("0%","25%","50%","75%","100%"),
    name="quantile")

mydata<-ggInterval::facedata
ggInterval_boxplot(mydata,aes(AD,col="black",alpha=0.5))

myMtcars<-classic2sym(mtcars)
myMtcars<-myMtcars$intervalData
ggInterval_boxplot(myMtcars,aes(disp))
```

`ggInterval_CRplot` *Figure with x-axis = center y-axis = range*

Description

Visualize the relation between center and range.

Usage

```
ggInterval_CRplot(data = NULL,mapping = aes(NULL),plotAll=FALSE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame,which will be automatically convert to ggInterval data.

`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`plotAll`	booleans, if TRUE, plot all variable together

Value

Return a ggplot2 object.

Examples

```
ggInterval_CRplot(iris,aes(iris$Sepal.Length))

mydata<-ggInterval::facedata
ggInterval_CRplot(mydata,aes(AD,col="blue",pch=2))
```

`ggInterval_hist` *Histogram for symbolic data with equal-bin or unequal-bin.*

Description

Visualize the continuous variable distribution by dividing the x axis into bins,and calculating the frequency of observation interval in each bin.

Usage

```
ggInterval_hist(data = NULL,mapping = aes(NULL),method="equal-bin",bins=10,
 plotAll = FALSE, position = "identity", alpha = 0.5)
```

Arguments

`data`	A ggInterval object.It can also be either RSDA object or classical data frame,which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`method`	It can be equal-bin(default) or unequal-bin.Enqual-bin means the width in histogram is equal, which represent all intervals divided have the same range. unequal-bin means the range of intervals are not the same,and it can be more general on data. Thus, the bins of unequal-bin method depends on the data, and the argument "bins" will be unused.
`bins`	x axis bins,which mean how many partials the variable
`plotAll`	boolean, whether plot all variables, default FALSE. will be separate into.
`position`	"stack" or "identity"
`alpha`	fill alpha

Value

Return a ggplot2 object.

Examples

```
ggInterval_hist(mtcars,aes(x=wt))

ggInterval_hist(iris,aes(iris$Petal.Length,col="blue",alpha=0.2,
   fill="red"),bins=30)


d<-data.frame(x=rnorm(1000,0,1))
p<-ggInterval_hist(d,aes(x=x),bins=40,method="equal-bin")$plot
p

p+scale_fill_manual(values=rainbow(40))+labs(title="myNorm")
```

`ggInterval_indexImage` *An index plot presented by color image for interval data.*

Description

Visualize the range of the variables of each observations by using color image.The index image replace margin bar by color,thus it will be more visible for data.

Usage

```
ggInterval_indexImage(data = NULL,mapping = aes(NULL),
column_condition=TRUE,full_strip=FALSE, plotAll = FALSE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame,which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
`column_condition`	Boolean variables, which mean the color present by column condition (if TRUE) or matrix condition (if FALSE)
`full_strip`	Boolean variables, which mean the strip present in full figure-width (if TRUE) or only in its variable values(if FALSE).
`plotAll`	Boolean, which determine if the heatmap type for visualizing full variables is used. default FALSE.

Value

Return a ggplot2 object.

Examples

```
d<-data.frame(qq=rnorm(1000,0,1))
ggInterval_indexImage(d,aes(qq))

mydata<-ggInterval::facedata
p<-ggInterval_indexImage(mydata,aes(AD),full_strip=TRUE,column_condition = TRUE)
#Recommend to add coord_flip() to make the plot more visible
p+coord_flip()

myIris<-classic2sym(iris,groupby="Species")
myIris<-myIris$intervalData
p<-ggInterval_indexImage(myIris,aes(myIris$Petal.Length),full_strip=FALSE,column_condition=TRUE)
p

ggInterval_indexImage(mtcars,aes(disp))+labs(x="anything")
```

`ggInterval_indexplot` *Plot the range of each observations*

Description

Visualize the range of the variables of each observations by using a kind of margin bar that indicate the minimal and maximal of observations.

Usage

```
ggInterval_indexplot(data = NULL,mapping = aes(NULL),
plotAll = FALSE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame,which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`plotAll`	plot all variables

Value

Return a ggplot2 object.

Examples

```
#the observations show on the y-axis .values on x-axis
ggInterval_indexplot(iris,aes(x=iris$Sepal.Length))

#change above axis
ggInterval_indexplot(mtcars,aes(y=disp,col="red",fill="grey"))

#symbolic data
mydata <- ggInterval::facedata
ggInterval_indexplot(mydata,aes(x=3:13,y=AD))
```

`ggInterval_MMplot` *A min-max plot for interval data*

Description

Visualize the range of the variables of each observations by marking minimal and maximal point.

Usage

```
ggInterval_MMplot(data = NULL,mapping = aes(NULL),
          scaleXY = "local",plotAll=FALSE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame,which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
`scaleXY`	default "local", which means limits of x-axis and y-axis depend on their own variable. "global" means limits of them depend on all variables that user input.
`plotAll`	booleans, if TRUE, plot all variable together

Value

Return a ggplot2 object.

Examples

```
ggInterval_MMplot(mtcars,aes(disp))

mydata2<-ggInterval::Cardiological
ggInterval_MMplot(mydata2,aes(mydata2$Pulse,size=3))

d<-mapply(c(10,20,40,80,160),c(20,40,80,160,320),FUN=runif,n=1000)
```

```
d<-data.frame(qq=matrix(d,ncol=1))
ggInterval_MMplot(d,aes(qq))

myIris<-classic2sym(iris,groupby="Species")
myIris<-myIris$intervalData
ggInterval_MMplot(myIris,aes(myIris$Petal.Length))+
   theme_classic()
```

`ggInterval_PCA` *Vertice-PCA for interval data*

Description

ggInterval_PCA performs a principal components analysis on the given numeric interval data and returns the results like princomp, ggplot object and a interval scores.

Usage

```
ggInterval_PCA(data = NULL,mapping = aes(NULL),plot=TRUE,
                        concepts_group=NULL, poly = FALSE, adjust = TRUE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame, which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping. It is the same as the mapping of ggplot2.
`plot`	Boolean variable,Auto plot (if TRUE).It can also plot by its inner object
`concepts_group`	color with each group of concept
`poly`	if plot a poly result
`adjust`	adjust sign of the principal component

Value

A ggplot object for PC1,PC2,and a interval scores and others.

- scores_interval - The interval scores after PCA.
- ggplotPCA - a ggplot object with x-axis and y-axis are PC1 and PC2.
- others - others are the returns values of princomp.

Examples

```
ggInterval_PCA(iris)

mydata2<-ggInterval::Cardiological
ggInterval_PCA(mydata2,aes(col="red",alpha=0.2))

d<-mapply(c(10,20,40,80,160),c(20,40,80,160,320),FUN=runif,n=1000)
d<-data.frame(qq=matrix(d,ncol=4))
ggInterval_PCA(d)

myIris<-classic2sym(iris,"Species")
p<-ggInterval_PCA(myIris,plot=FALSE)
p$ggplotPCA
p$scores_interval
```

`ggInterval_radarplot` *A interval Radar plot*

Description

Using ggplot2 package to make a radar plot with multiple variables.Each variables contains min values and max values as a symbolic data.

Usage

```
ggInterval_radarplot(data=NULL,layerNumber=3,
inOneFig=TRUE,showLegend=TRUE,showXYLabs=FALSE,
plotPartial=NULL,
alpha=0.5,
base_circle=TRUE,
base_lty=2,
addText=TRUE,
type="default",
quantileNum=4,
Drift=0.5,
addText_modal=TRUE,
addText_modal.p=FALSE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame(not recommended),which will be automatically convert to ggInterval data.
`layerNumber`	number of layer of a concentric circle,usually to visuallize the reach of a observation in particularly variable.
`inOneFig`	whether plot all observations in one figure.if not, it will generate a new windows containing distinct observations.

`showLegend`	whether show the legend.
`showXYLabs`	whether show the x,y axis labels.
`plotPartial`	a numeric vector,which is the row index from the data.if it is not null, it will extract the row user deciding to draw a radar plot from original data.Notes : the data must be an interval data if the plotPartial is not null.
`alpha`	aesthetic alpha of fill color
`base_circle`	boolean, if true, it will generate inner circle.
`base_lty`	line type in base figure
`addText`	add the value of interval-valued variables in figure
`type`	different type of radar,it can be "default","rect","quantile"
`quantileNum`	if type == "quantile", it will provide the number of percentage
`Drift`	The drift term, which determines the radar values beginning.
`addText_modal`	add the factor of modal multi-valued variables in figure..
`addText_modal.p`	add the value of modal multi-valued variables in figure..

Examples

```
# must specify plotPartial to some certain rows you want to plot
Environment.n <- Environment[, 5:17]
ggInterval_radarplot(Environment.n,
                plotPartial = 2,
                showLegend = FALSE,
                base_circle = TRUE,
                base_lty = 2,
                addText = FALSE
) +
 labs(title = "") +
 scale_fill_manual(values = c("gray50")) +
 scale_color_manual(values = c("red"))

ggInterval_radarplot(Environment,
                plotPartial = 2,
                showLegend = FALSE,
                base_circle = FALSE,
                base_lty = 1,
                addText = TRUE
) +
 labs(title = "") +
 scale_fill_manual(values = c("gray50")) +
 scale_color_manual(values = c("gray50"))
```

ggInterval_scatterMatrix

scatter plot for all variable by interval data.

Description

Visualize the all continuous variable distribution by rectangle for both x-axis and y-axis with a matrix grid. Note: this function will automatically filter out the discrete variables,and plot all continuous in input data,so it can not be necessary that give the particularly variables in aes such like (aes(x = x, y = y)). It isn't also recommended to deal with too many variables because the big O in calculating full matrix will be too large.

Usage

```
ggInterval_scatterMatrix(data = NULL,mapping = aes(NULL), showLegend=FALSE)
```

Arguments

`data`	A ggInterval object. It can also be either RSDA object or classical data frame,which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
`showLegend`	whether show the legend.

Value

Return a plot with no longer a ggplot2 object,instead of a marrangeGrob object.

Examples

```
a<-rnorm(1000,0,5)
b<-runif(1000,-20,-10)
c<-rgamma(1000,10,5)
d<-as.data.frame(cbind(norm=a,unif=b,gamma_10_5=c))
ggInterval_scatterMatrix(d)


ggInterval_scatterMatrix(mtcars[,c("mpg","wt","qsec")],
    aes(col="red",lty=2,fill="blue",alpha=0.3))


myIris <- classic2sym(iris,groupby = "Species")$intervalData
ggInterval_scatterMatrix(myIris[,1:3])


mydata <- ggInterval::Cardiological
ggInterval_scatterMatrix(mydata[,1:3],aes(fill="black",alpha=0.2))
```

ggInterval_scatterplot

scatter plot for two continuous interval data

Description

Visualize the twwo continuous variable distribution by rectangle and each of its width and heigth represents a interval of the data.

Usage

```
ggInterval_scatterplot(data = NULL,mapping = aes(NULL), ...)
```

Arguments

`data`	A ggInterval object.It can also be either RSDA object or classical data frame, which will be automatically convert to ggInterval data.
`mapping`	Set of aesthetic mappings created by aes() or aes_(). If specified and inherit. aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.
`...`	Others in ggplot2.

Value

Return a ggplot2 object.

Examples

```
a<-rnorm(1000,0,5)
b<-runif(1000,-20,-10)
d<-as.data.frame(cbind(norm=a,unif=b))
ggInterval_scatterplot(d,aes(a,b))


ggInterval_scatterplot(mtcars[,c("mpg","wt","qsec")],
    aes(x=mpg,y=wt,
    col="red",lty=2,fill="blue",alpha=0.3))


myIris <- classic2sym(iris,groupby = "Species")$intervalData
p<-ggInterval_scatterplot(myIris,aes(myIris$Petal.Length,myIris$Petal.Width))
p
p+scale_fill_manual(labels=rownames(myIris),
                  values=c("red","blue","green"),
                  name="Group")


mydata <- ggInterval::facedata
```

```
p<-ggInterval_scatterplot(mydata[1:10,],aes(AD,BC,alpha=0.2))
p+scale_fill_manual(labels=rownames(mydata)[1:10],
                    values=rainbow(10),
                    name="Group")
```

`iris.i` *iris.i data example*

Description

iris.i interval data example.

Usage

```
data(iris.i)
```

Format

An object of class `data.frame` (inherits from `symbolic_tbl`) with 3 rows and 4 columns.

Examples

```
data(iris.i)
ggInterval_indexplot(iris.i, aes(x = Sepal.Length))
```

`mtcars.i` *mtcars.i data example*

Description

mtcars.i interval and modal data example.

Usage

```
data(mtcars.i)
```

Format

An object of class `symbolic_tbl` (inherits from `tbl_df`, `tbl`, `data.frame`, `symbolic_tbl`) with 5 rows and 11 columns.

Examples

```
data(mtcars.i)
ggInterval_indexplot(mtcars.i, aes(x = mpg))
```

`mushroom` *mushroom data example*

Description

mushroom interval data example.

Usage

```
data(mushroom)
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`, `symbolic_tbl`) with 23 rows and 3 columns.

References

Billard L. and Diday E. (2006).Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Examples

```
data(mushroom)

ggInterval_scatterplot(mushroom, aes(x = Cap.Widths, y = Stipe.Lengths))
```

`oils` *oils data example*

Description

oils interval data example.

Usage

```
data(oils)
```

Format

An object of class `symbolic_tbl` (inherits from `tbl_df`, `tbl`, `data.frame`) with 8 rows and 4 columns.

References

Billard L. and Diday E. (2006).Symbolic data analysis: Conceptual statistics and data mining. Wiley, Chichester.

Examples

```
data(oils)

ggInterval_scatterplot(oils, aes(x = GRA, y = IOD))
```

`RSDA2sym` *RSDA object to symbolic object for ggplot*

Description

It will be a good way to unify all symbolic data object in R that collects all useful symbolic analysis tools such like RSDA into the same class for management. In this way, user who wants to do some study in symbolic data will be more convenient for searching packages.Thus,RSDA2sym collecting RSDA object into ggInterval object will do for plot(ggplot) and RSDA's analysis.

Usage

```
RSDA2sym(data=NULL,rawData=NULL)
```

Arguments

`data`	an interval data, which may transfrom by RSDA::classic.to.sym .Note:data is a necessary parameter,and must have symbolic_tbl class.
`rawData`	rawData, which can be transformed to interval data, must be a data frame and match to data.

Value

Return an object of class "ggInterval", which have a interval data and others as follows.

- intervalData - The Interval data after converting also known as a RSDA object.
- rawData - Classical data that user input.
- clusterResult - Cluster results .If the groupby method is a clustering method then it will exist.
- statisticsDF - A list contains data frame including some typically statistics in each group.

#'

Examples

```
r<-ggInterval::Cardiological
mySym<-RSDA2sym(r)
mySym$intervalData
```

`scale` *scale for symbolic data table*

Description

scale for symbolic data table

Usage

```
scale(x, ...)

## Default S3 method:
scale(x, center = TRUE, scale = TRUE, ...)

## S3 method for class 'symbolic_tbl'
scale(x, ...)

## S3 method for class 'symbolic_interval'
scale(x, ...)
```

Arguments

`x`	A ggInterval object. It can also be either RSDA object or classical data frame, which will be automatically convert to ggInterval data.
`...`	Used by other R function.
`center`	same as base::scale, either a logical value or numeric-alike vector of length equal to the number of columns of x, where nmeric-alike means that as.numeric(.) will be applied successfully if is.numeric(.) is not true.
`scale`	same as base::scale, either a logical value or a numeric-alike vector of length equal to the number of columns of x.

Value

Return a scale ggInterval object.

Examples

```
#For all interval-valued
scale(facedata)

#For both interval-valued and modal multi-valued
scale(mtcars.i)
```

`summary` *summary for symbolic data table*

Description

summary for symbolic data table

Usage

```
summary(object, ...)

## Default S3 method:
summary(object, ...)

## S3 method for class 'symbolic_tbl'
summary(object, ...)

## S3 method for class 'symbolic_interval'
summary(object, ...)

## S3 method for class 'symbolic_modal'
summary(object, summary_fun = "mean", ...)
```

Arguments

`object`	an object for which a summary is desired.
`...`	additional arguments affecting the summary produced.
`summary_fun`	only works when the symbolic_modal class input, it determine which summary function be applied for each modal.

Value

Return a summary table.

Examples

```
#For all interval-valued
summary(facedata)

#For both interval-valued and modal multi-valued
summary(Environment)

summary(Environment$URBANICITY, summary_fun = "quantile")
```

Index