

Package: statVisual (via r-universe)

September 1, 2024

Type Package

Title Statistical Visualization Tools

Version 1.2.1

Date 2020-02-19

Depends R (>= 3.5.0), Biobase, ggplot2, stats

Imports GGally, dplyr, reshape2, ggdendro, tibble, ggfortify, ggrepel, gbm, forestplot, rpart.plot, grDevices, gridExtra, pROC, factoextra, methods, pvca, limma, randomForest, glmnet, knitr, rmarkdown, pheatmap, RColorBrewer, graphics, tidyverse, magrittr

VignetteBuilder knitr

Description Visualization functions in the applications of translational medicine (TM) and biomarker (BM) development to compare groups by statistically visualizing data and/or results of analyses, such as visualizing data by displaying in one figure different groups' histograms, boxplots, densities, scatter plots, error-bar plots, or trajectory plots, by displaying scatter plots of top principal components or dendrograms with data points colored based on group information, or visualizing volcano plots to check the results of whole genome analyses for gene differential expression.

License GPL (>= 2)

Encoding UTF-8

LazyData true

NeedsCompilation no

Author Wenfei Zhang [aut, cre], Weiliang Qiu [aut, ctb], Xuan Lin [aut, ctb], Donghui Zhang [aut, ctb]

Maintainer Wenfei Zhang <Wenfeizhang@gmail.com>

Date/Publication 2019-12-15 16:30:05 UTC

Repository <https://gefeizhang.r-universe.dev>

RemoteUrl <https://github.com/gefeizhang/statvisual>

RemoteRef HEAD

RemoteSha 6ea489ff5778884e5a5ffcda984ea59e824a9c81

Contents

barPlot	2
BiAxisErrBar	5
Box	6
BoxROC	9
cv_glmnet_plot	11
Den	12
Dendro	14
diffCorDat	17
ErrBar	18
esSim	20
genoSim	21
Heat	22
Hist	23
ImpPlot	26
iprcomp	27
LinePlot	29
longDat	31
PCA_score	32
PVCA	34
stackedBarPlot	36
statVisual	38
Volcano	39
XYscatter	42
Index	44

barPlot

Compare Groups Based on Barplots Across Time

Description

This function is to compare groups using barplots at each time point. In addition, line segments are used to connect the mean/median of each barplot of the same group across time to show the differences between the mean trajectories. Also, for each barplot the barplot of mean \pm standard error will be plot.

Usage

```
barPlot(
  data,
  x = NULL,
  y,
  group = NULL,
  semFlag = TRUE,
  xFlag = FALSE,
  bar.width = 0.5,
  dodge.width = 0.8,
  jitter = FALSE,
  jitter.alpha = 0.7,
  jitter.width = 0.1,
  line = NULL,
  line.color = "black",
  xlab = x,
  ylab = line,
  theme_classic = TRUE,
  group.lab = group,
  title = "bar plots",
  xLevel = NULL,
  addThemeFlag = TRUE,
  ...)
```

Arguments

data	A data frame. Rows are subjects; Columns are variables describing the subjects.
x	character. The column name of data that indicates the first grouping variable
y	character. The column name of data that indicates the variable on y axis
group	character. The column name of data that indicates the subject groups. The barplots will be drawn for each of the subject group within each category of x.
semFlag	logical. Indicate if sem or se should be used to draw error bar
xFlag	logical. Indicate if x should be treated as continuous (xFlag=TRUE)
bar.width	numeric. error bar width
dodge.width	numeric. dodge width for error bar and jitter (prevent overlapping)
jitter	logical, plot jitter or not, default TRUE
jitter.alpha	numeric. jitter transparency
jitter.width	numeric. jitter width in error bar
line	character. line connect error bar, default uses mean, can be set as 'median', NULL (no line)
line.color	character. connection line color, only available when group = NULL
xlab	character. x axis label
ylab	character. y axis label
theme_classic	logical. Use classic background without grids (default: TRUE).

group.lab	character. label of group variable
title	character. title of plot
xLevel	character. A character vector indicating the order of the elements of x to be shown on x-axis if <code>is.null(x)==FALSE</code> .
addThemeFlag	logical. Indicates if light blue background and white grid should be added to the figure.
...	other input parameters for facet & theme

Value

A list of the following 9 elements: “data”, “layers”, “scales”, “mapping”, “theme”, “coordinates”, “facet”, “plot_env”, “labels”.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(longDat)

print(dim(longDat))
print(longDat[1:3,])

print(table(longDat$time, useNA = "ifany"))
print(table(longDat$grp, useNA = "ifany"))
print(table(longDat$sid, useNA = "ifany"))

print(table(longDat$time, longDat$grp))

statVisual(type = 'barPlot',
  data = longDat,
  x = 'time',
  y = 'y',
  group = 'grp',
  title = "Bar plots across time")

barPlot(
  data = longDat,
  x = 'time',
  y = 'y',
  group = 'grp',
  title = "Bar plots across time")
```

Description

Compare patterns of two outcomes with different scales across the range of the common predictor using error bar plots. Each bar plot displays mean \pm standard error.

Usage

```
BiAxisErrBar(dat,
  group,
  y.left,
  y.right,
  title = "Bi-Axis Error Bar Plot",
  xlab = group,
  ylab.left = y.left,
  ylab.right = y.right,
  legendLabel = "y axis variables",
  delta = NULL,
  cvThresh = 0.01,
  Ntick = 5,
  semFlag = TRUE, #semFlag = FALSE if SE is required
  GroupLevel = NULL,
  addThemeFlag = FALSE
)
```

Arguments

<code>dat</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>group</code>	character. A categorical variable in data that indicates the predictor.
<code>y.left</code>	character. The column name of data that indicates the first outcome variable, the error bar plot of which will be drawn on the left side.
<code>y.right</code>	character. The column name of data that indicates the second outcome variable, the error bar plot of which will be drawn on the right side.
<code>title</code>	character. title of the plot.
<code>xlab</code>	character. Label for the x-axis.
<code>ylab.left</code>	character. Label for the left-side y-axis.
<code>ylab.right</code>	character. Label for the right-side y-axis.
<code>legendLabel</code>	character. Legend label.
<code>delta</code>	numeric. A small number so that the second error bar plot will shift delta distance from the first error bar plot.
<code>cvThresh</code>	numeric. A small positive number. If the coefficient of variation (CV) is smaller than <code>cvThresh</code> , then the scaling factor will be set to one.

Ntick	integer. Number of ticks on the two y-axes.
semFlag	logical. Indicating if standard error of the mean (semFlag = TRUE) or standard error (semFlag = FALSE) will be used to construct the error bars.
GroupLevel	A vector of unique values of group indicating the order of group shown in x-axis.
addThemeFlag	logical. Indicates if light blue background and white grid should be added to the figure.

Value

A list with 9 elements. data, layers, scales, mapping, theme, coordinates, facet, plot_env, and labels.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
library(tidyverse)
library(ggplot2)

print(head(mtcars))

print(table(mtcars$gear, useNA="ifany"))

statVisual(type = "BiAxisErrBar",
  dat= mtcars,
  group = "gear",
  y.left = "mpg",
  y.right = "wt")

BiAxisErrBar(
  dat = mtcars,
  group = "gear",
  y.left = "mpg",
  y.right = "wt")
```

Description

This function is to compare groups using boxplots at each time point. In addition, line segments are used to connect the mean/median of each boxplot of the same group across time to show the differences between the mean trajectories.

Usage

```
Box(
  data,
  x = NULL,
  y,
  group = NULL,
  fill = NULL,
  theme_classic = TRUE,
  fill.alpha = 0.7,
  box.width = 0.5,
  dodge.width = 0.8,
  jitter = TRUE,
  jitter.alpha = 0.7,
  jitter.width = 0.2,
  point.size = 1,
  xlab = x,
  ylab = y,
  group.lab = group,
  fill.lab = group,
  title = "Boxplot",
  line = "mean",
  line.color = "black",
  xLevel = NULL,
  addThemeFlag = TRUE,
  ...)
```

Arguments

<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>x</code>	character. The column name of data that indicates the first grouping variable (e.g. observation time)
<code>y</code>	character. The column name of data that indicates the variable on y axis
<code>group</code>	character. The column name of data that indicates the subject groups (e.g., treatment group). The boxplots will be drawn for each of the subject group within each category of x.
<code>fill</code>	boxplot inside color indicated by the categories of group
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>fill.alpha</code>	boxplot transparency
<code>box.width</code>	boxplot width
<code>dodge.width</code>	dodge width for boxplot and jitter (prevent overlapping)
<code>jitter</code>	logical. plot jitter or not, default TRUE
<code>jitter.alpha</code>	jitter transparency
<code>jitter.width</code>	jitter width in boxplot
<code>point.size</code>	size of a jitter point

<code>xlab</code>	character. x axis label
<code>ylab</code>	character. y axis label
<code>group.lab</code>	label of group variable
<code>fill.lab</code>	label of fill variable
<code>title</code>	character. title of plot
<code>line</code>	line connect boxes, default plot mean, can be set as 'median', or NULL (no line)
<code>line.color</code>	connection line color, only available when group = NULL
<code>xLevel</code>	character. A character vector indicating the order of the elements of x to be shown on x-axis if <code>is.null(x)==FALSE</code> .
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with the following 9 elements: data, layers, scales, mapping, theme, coordinates, facet, plot_env, and labels.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
library(dplyr)

data(longDat)

print(dim(longDat))
print(longDat[1:3,])

print(table(longDat$time, useNA = "ifany"))
print(table(longDat$grp, useNA = "ifany"))
print(table(longDat$sid, useNA = "ifany"))

print(table(longDat$time, longDat$grp))

statVisual(type = 'Box',
           data = longDat,
           x = 'time',
           y = 'y',
           group = 'grp',
           title = "Boxplots across time")

Box(
  data = longDat,
  x = 'time',
```



```

y = 'y',
group = 'grp',
title = "Boxplots across time")

```

BoxROC

Compare Boxplots with ROC Curve

Description

Compare boxplots with ROC curve. The value of the variable `y` will be jittered shown on the boxplots. The area under ROC curve will also be calculated and shown in the plot of ROC curve.

Usage

```

BoxROC(
  data,
  group.var,
  y,
  box.xlab = group.var,
  box.ylab = y,
  box.group.lab = group.var,
  jitter.alpha = 0.8,
  jitter.width = 0.1,
  point.size = 3,
  roc.xlab = "Specificity",
  roc.ylab = "Sensitivity",
  addThemeFlag = TRUE)

```

Arguments

<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>group.var</code>	character. The column name of data that indicates the two subject groups. It also indicates the color of the two boxplots.
<code>y</code>	character. The column name of data that indicates the variable, for which the box will be drawn.
<code>box.xlab</code>	character. boxplot x axis label (default: <code>group.var</code>)
<code>box.ylab</code>	character. boxplot y axis label (default: <code>y</code>)
<code>box.group.lab</code>	character. boxplot legend label (default: <code>group.var</code>)
<code>jitter.alpha</code>	numeric. transparency of jitters
<code>jitter.width</code>	numeric. width of jitters
<code>point.size</code>	size of a jitter point
<code>roc.xlab</code>	character. roc curve x axis label (default: Specificities)
<code>roc.ylab</code>	character. roc curve y axis label (default: Sensitivities)
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.

Value

A list with the following 12 elements: grobs, layout, widths, heights, respect, rownames, colnames, name, gp, vp, children, childrenOrder.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
library(dplyr)
library(gridExtra)

data(esSim)
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first probe which is over-expressed in cases
pDat$probe1 = dat[1,]

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

statVisual(type = 'BoxROC',
           data = pDat,
           group = 'grp',
           y = 'probe1',
           point.size = 1)

BoxROC(
  data = pDat,
  group = 'grp',
  y = 'probe1',
  point.size = 1)
```

`cv_glmnet_plot`*Plot the Cross-Validation Curve Produced by cv.glmnet*

Description

Plots the cross-validation curve, and upper and lower standard error curves, as a function of the values of the tuning parameter lambda.

Usage

```
cv_glmnet_plot(x,  
               y,  
               family = "binomial",  
               addThemeFlag = TRUE,  
               ...)
```

Arguments

<code>x</code>	a matrix with rows are subjects and columns are numeric variables (predictors). No missing values are allowed.
<code>y</code>	a vector of response. The number of elements of <code>y</code> is the same as the number of rows of <code>x</code> .
<code>family</code>	character. Indicating response type. see the description in glmnet .
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for glmnet function.

Value

A list with 9 elements. `data`, `layers`, `scales`, `mapping`, `theme`, `coordinates`, `facet plot_env`, and `labels`.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
library(dplyr)  
library(tibble)  
library(glmnet)  
  
data(esSim)  
print(esSim)  
  
# expression data
```

```
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first 6 probes (3 OE probes, 2 UE probes, and 1 NE probe)
pDat$probe1 = dat[1,]
pDat$probe2 = dat[2,]
pDat$probe3 = dat[3,]
pDat$probe4 = dat[4,]
pDat$probe5 = dat[5,]
pDat$probe6 = dat[6,]

print(pDat[1:2, ])

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

statVisual(type = "cv_glmnet_plot",
           x = as.matrix(pDat[, c(3:8)]),
           y = pDat$grp,
           family = "binomial")

cv_glmnet_plot(x = as.matrix(pDat[, c(3:8)]),
              y = pDat$grp,
              family = "binomial")
```

Den

Compare Groups Based on Density Plots

Description

Compare groups based on density plots.

Usage

```
Den(
  data,
  y,
```

```

group = NULL,
fill = group,
border.color = NULL,
inner.color = NULL,
theme_classic = TRUE,
xlab = y,
ylab = "density",
group.lab = group,
title = "Density plot",
alpha = 0.3,
addThemeFlag = TRUE,
...)
```

Arguments

<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>y</code>	character. The column name of data that indicates the variable, for which the histogram will be drawn. The string <code>y</code> can also indicate a function of the variable, e.g., $\log(y)$.
<code>group</code>	character. The column name of data that indicates the subject groups. The density will be drawn for each of the subject group. It also indicates the border colors of the densities.
<code>fill</code>	grouping variable, density inside color
<code>border.color</code>	density border color, only available when <code>group</code> & <code>fill</code> are NULL
<code>inner.color</code>	density inside color, only available when <code>group</code> & <code>fill</code> are NULL
<code>theme_classic</code>	Use classic background without grids (default: FALSE)
<code>xlab</code>	x axis label
<code>ylab</code>	y axis label
<code>group.lab</code>	label of group variable
<code>title</code>	title of plot
<code>alpha</code>	transparency of density inside color
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with 9 elements. `data`, `layers`, `scales`, `mapping`, `theme`, `coordinates`, `facet`, `plot_env`, and `labels`.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(esSim)
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first probe which is over-expressed in cases
pDat$probe1 = dat[1,]

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

statVisual(type = 'Den',
           data = pDat,
           y = 'probe1',
           group = 'grp')

Den(
  data = pDat,
  y = 'probe1',
  group = 'grp')
```

Dendro

Compare Groups Based on Dendrogram

Description

Compare groups based on dendrogram. The nodes of the dendrogram will be colored by group.

Usage

```
Dendro(
  x,
  group = NULL,
  xlab = NULL,
```

```

ylab = NULL,
title = NULL,
cor.use = "pairwise.complete.obs",
cor.method = "pearson",
distance = "rawdata",
distance.method = "euclidean",
hclust.method = "complete",
yintercept = NULL,
theme_classic = TRUE,
addThemeFlag = TRUE,
...)
```

Arguments

<code>x</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>group</code>	character. The column name of data that indicates the subject groups. The nodes of the dendrogram will be colored by info provided by group.
<code>xlab</code>	x axis label
<code>ylab</code>	y axis label
<code>title</code>	title of the plot
<code>cor.use</code>	character. Indicate which data will be used to compute correlation coefficients. It can take values “everything”, “all.obs”, “complete.obs”, “na.or.complete”, “pairwise.complete.obs”.
<code>cor.method</code>	character. Indicate which type of correlation coefficients will be calculated: “pearson”, “kendall”, “spearman”.
<code>distance</code>	character. Indicate which type of data will be used to calculate distance: “rawdata” (i.e., using raw data to calculate distance), “cor” (i.e., using correlation coefficients as distance), “1-cor” (i.e., using (1–correlation coefficients) as distance), “1-lcor” (i.e., using (1– correlation coefficients) as distance).
<code>distance.method</code>	character. Available when ‘distance = “rawdata”’. Indicate the definition of distance: distance used in calculate dist “rawdata” (i.e., using raw data to calculate distance), “cor” (i.e., using correlation coefficients as distance), “1-cor” (i.e., using (1–correlation coefficients) as distance), “1-lcor” (i.e., using (1– correlation coefficients) as distance).
<code>hclust.method</code>	character. Indicate which agglomeration method will be used to perform hierarchical clustering. This should be (an unambiguous abbreviation of) one of “ward.D”, “ward.D2”, “single”, “complete”, “average”, “mcquitty”, “median”, or “centroid”. Please refer to hclust .
<code>yintercept</code>	numeric. A line indicating the height of the dendrogram, for example, indicating where the dendrogram should be cut to obtain clusters.
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with 9 elements. data, layers, scales, mapping, theme, coordinates, facet plot_env, and labels.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(esSim)
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first 6 probes (3 OE probes, 2 UE probes, and 1 NE probe)
pDat$probe1 = dat[1,]
pDat$probe2 = dat[2,]
pDat$probe3 = dat[3,]
pDat$probe4 = dat[4,]
pDat$probe5 = dat[5,]
pDat$probe6 = dat[6,]

print(pDat[1:2, ])

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

pDat$grp = factor(pDat$grp)

statVisual(type = 'Dendro',
           x = pDat[, c(3:8)],
           group = pDat$grp)

Dendro(
  x = pDat[, c(3:8)],
  group = pDat$grp)
```

`diffCorDat`*A Dataset for Differential Correlation Analysis*

Description

A dataset for differential correlation analysis.

Usage

```
data("diffCorDat")
```

Format

A data frame with 100 observations on the following 3 variables.

`probe1` numeric. expression level for probe1

`probe2` numeric. expression level for probe2

`grp` character. a factor with levels `cases` `controls`

Details

The simulated data set contains expression levels of 2 gene probes for 50 cases and 50 controls. The expression levels of probe1 are generated from $N(0, 1)$. The expression levels of probe2 for controls are also generated from $N(0, 1)$. The expression levels of probe 2 for cases are generated from the formula $probe2_i = -probe1_i + e_i$, $i = 1, \dots, nCases$, where $e_i \sim N(0, 0.3^2)$.

That is, the expression levels of probe 1 and probe 2 are negatively correlated in cases, but not correlated in controls.

Examples

```
data(diffCorDat)

print(dim(diffCorDat))
print(diffCorDat[1:2,])
```

 ErrBar

Compare Groups Based on dotplots Across Time

Description

This function is to compare groups using dotplots at each time point. In addition, line segments are used to connect the mean/median of each dotplot of the same group across time to show the differences between the mean trajectories. Also, for each dotplot the barplot of mean \pm standard error will be plot.

Usage

```
ErrBar(
  data,
  x = NULL,
  y,
  group = NULL,
  semFlag = TRUE,
  xFlag = FALSE,
  bar.width = 0.5,
  dodge.width = 0.8,
  jitter = TRUE,
  jitter.alpha = 0.7,
  jitter.width = 0.1,
  line = "mean",
  line.color = "black",
  xlab = x,
  ylab = line,
  theme_classic = TRUE,
  group.lab = group,
  title = "Dot plots",
  xLevel = NULL,
  addThemeFlag = TRUE,
  ...)
```

Arguments

<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>x</code>	character. The column name of data that indicates the first grouping variable
<code>y</code>	character. The column name of data that indicates the variable on y axis
<code>group</code>	character. The column name of data that indicates the subject groups. The dotplots will be drawn for each of the subject group within each category of x.
<code>semFlag</code>	logical. Indicate if sem or se should be used to draw error bar
<code>xFlag</code>	logical. Indicate if x should be treated as continuous (<code>xFlag=TRUE</code>)
<code>bar.width</code>	numeric. error bar width

dodge.width	numeric. dodge width for error bar and jitter (prevent overlapping)
jitter	logical, plot jitter or not, default TRUE
jitter.alpha	numeric. jitter transparency
jitter.width	numeric. jitter width in error bar
line	character. line connect error bar, default uses mean, can be set as 'median', NULL (no line)
line.color	character. connection line color, only available when group = NULL
xlab	character. x axis label
ylab	character. y axis label
theme_classic	logical. Use classic background without grids (default: TRUE).
group.lab	character. label of group variable
title	character. title of plot
xLevel	character. A character vector indicating the order of the elements of x to be shown on x-axis if is.null(x)==FALSE.
addThemeFlag	logical. Indicates if light blue background and white grid should be added to the figure.
...	other input parameters for facet & theme

Value

A list of the following 9 elements: “data”, “layers”, “scales”, “mapping”, “theme”, “coordinates”, “facet”, “plot_env”, “labels”.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(longDat)

print(dim(longDat))
print(longDat[1:3,])

print(table(longDat$time, useNA = "ifany"))
print(table(longDat$grp, useNA = "ifany"))
print(table(longDat$sid, useNA = "ifany"))

print(table(longDat$time, longDat$grp))

statVisual(type = 'ErrBar',
  data = longDat,
  x = 'time',
  y = 'y',
```

```
group = 'grp',
title = "Dot plots across time")

ErrBar(
  data = longDat,
  x = 'time',
  y = 'y',
  group = 'grp',
  title = "Dot plots across time")
```

esSim

A Simulated Gene Expression Dataset

Description

A simulated gene expression dataset for differential expression analysis.

Usage

```
data("esSim")
```

Format

The format is: Formal class 'ExpressionSet' [package "Biobase"] with expression levels of 100 probes for 20 samples.

The phenotype data contain 2 phenotype variables: sid (subject id) and grp (group indicator: 1 stands for case; 0 stands for control).

The feature data contain 4 feature variables: probeid (probe id), gene (fake gene symbol), chr (fake chromosome number), and memProbes (probe significance indicator: 1 stands for probes over-expressed (OE) in cases; -1 stands for probes under-expressed (UE) in cases; and 0 stands for non-differentially expressed (NE) probes). There are 3 OE probes, 2 UE probes, and 95 NE probes.

Details

The dataset was generated based on the R code in the manual of the function `lmFit` of the R Bioconductor package `limma`. There are 100 probes and 20 samples (10 controls and 10 cases). The first 3 probes are over-expressed in cases. The 4-th and 5-th probes are under-expressed in cases. The remaining 95 probes are non-differentially expressed between cases and controls. Expression levels for 100 probes were first generated from normal distribution with mean 0 and standard deviation varying between probes ($sd = 0.3\sqrt{4/\chi_4^2}$). For the 3 OE probes, we add 2 to the expression levels of the 10 cases. For the 2 UE probes, we subtract 2 from the expression levels of the 10 cases.

References

Please see the example in the manual for the function `lmFit` in the R Bioconductor package `limma`.

Examples

```
data(esSim)

print(esSim)

###
dat=exprs(esSim)
print(dim(dat))
print(dat[1:2,])

###
pDat=pData(esSim)
print(dim(pDat))
print(pDat)

# subject group status
print(table(esSim$grp))

###
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2, ])

# probe's status of differential expression
print(table(fDat$memProbes))
```

genoSim

An ExpressionSet Object Storing Simulated Genotype Data

Description

An ExpressionSet object storing simulated genotype data with 10 SNPs and 100 subjects.

Usage

```
data("genoSim")
```

Details

The simulated genotype data contain 50 cases and 50 controls. Each subject has genotype data for 10 SNPs. The first 2 SNPs have different minor allele frequencies (MAFs) between cases and controls (MAF for cases is 0.4 and MAF for controls is 0.2). We assume Hardy Weinberg Equilibrium. The remaining 8 SNPs have the same MAF ($MAF = 0.2$) in both cases and controls.

Examples

```
data(genoSim)

print(genoSim)
```

Heat

*Heatmap with Row Names Colored by Group***Description**

Heatmap with row names colored by group.

Usage

```
Heat(data,
      group = NULL,
      fontsize_row=10,
      fontsize_col=10,
      scale = "none",
      cluster_rows = TRUE,
      cluster_cols = TRUE,
      color = colorRampPalette(rev(brewer.pal(n = 7, name = "RdYlBu")))(100),
      angle_col = c("270", "0", "45", "90", "315"),
      ...)
```

Arguments

<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects. Except the column indicating subject group, all columns of data should be numeric.
<code>group</code>	character. The column name of data that indicates the subject groups. The row names of the heatmap will be colored based on group.
<code>fontsize_col</code>	x axis label font size
<code>fontsize_row</code>	y axis label font size
<code>scale</code>	character. Indicate how data will be scaled: "none" (i.e., no scaling), "row" (i.e., row scaled), "column" (i.e., column scaled).
<code>cluster_rows</code>	logic. Indicates if rows should be clustered.
<code>cluster_cols</code>	logic. Indicates if columns should be clustered.
<code>color</code>	vector indicating colors used in heatmap
<code>angle_col</code>	angle of the column labels. Please refer to the manual in pheatmap
<code>...</code>	other input parameters for facet & theme.

Value

A list with 10 elements: "rowInd", "colInd", "call", "carpet", "rowDendrogram", "colDendrogram", "breaks", "col", "colorTable", "layout".

Note

This function is based on the function [pheatmap](#) in pheatmap R package.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(esSim)
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first 6 probes (3 OE probes, 2 UE probes, and 1 NE probe)
pDat$probe1 = dat[1,]
pDat$probe2 = dat[2,]
pDat$probe3 = dat[3,]
pDat$probe4 = dat[4,]
pDat$probe5 = dat[5,]
pDat$probe6 = dat[6,]

print(pDat[1:2, ])

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

statVisual(type = 'Heat',
           data = pDat[, c(2:8)],
           group = 'grp')

Heat(
  data = pDat[, c(2:8)],
  group = 'grp')
```

Description

Compare groups based on histograms.

Usage

```
Hist(
  data,
  y,
  group = NULL,
  fill = group,
  border.color = NULL,
  inner.color = NULL,
  theme_classic = TRUE,
  bins = NULL,
  binwidth = NULL,
  alpha = 0.8,
  xlab = y,
  ylab = "count",
  group.lab = group,
  title = "Histogram",
  addThemeFlag = TRUE,
  ...)
```

Arguments

<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>y</code>	character. The column name of data that indicates the variable, for which the histogram will be drawn. The string <code>y</code> can also indicate a function of the variable, e.g., $\log(y)$.
<code>group</code>	character. The column name of data that indicates the subject groups. The histogram will be drawn for each of the subject group. It also indicates the border colors of the histograms.
<code>fill</code>	character. The column name of data that indicates the subject groups. It indicates the inside colors of the histograms.
<code>border.color</code>	Histogram border color, only available when <code>group</code> & <code>fill</code> are NULL.
<code>inner.color</code>	Histogram inside color, only available when <code>group</code> & <code>fill</code> are NULL.
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>bins</code>	integer. number of bins of histogram (default: 30).
<code>binwidth</code>	Bin width of histogram.
<code>alpha</code>	Transparency of histogram inside color.
<code>xlab</code>	x axis label
<code>ylab</code>	y axis label
<code>group.lab</code>	label of group variable
<code>title</code>	title of the plot

addThemeFlag logical. Indicates if light blue background and white grid should be added to the figure.

... other input parameters for facet & theme

Value

A list with the following 9 elements. data, layers, scales, mapping, theme, coordinates, facet, plot_env, and labels.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(esSim)
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first probe which is over-expressed in cases
pDat$probe1 = dat[1,]

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

statVisual(type = 'Hist',
           data = pDat,
           y = 'probe1',
           group = 'grp')

Hist(
  data = pDat,
  y = 'probe1',
  group = 'grp')
```

ImpPlot

Plot of Variable Importance

Description

Plot of variable importance based on results from [randomForest](#) or [gbm](#).

Usage

```
ImpPlot(model,  
  theme_classic = TRUE,  
  n.trees = NULL,  
  addThemeFlag = TRUE,  
  ...)
```

Arguments

<code>model</code>	An object returned by randomForest or gbm
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>n.trees</code>	integer. The number of trees used to generate the plot used in the function <code>summary.gbm</code> in the R library gbm . Only the first <code>n.trees</code> trees will be used.
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with 9 elements. `data`, `layers`, `scales`, `mapping`, `theme`, `coordinates`, `facet plot_env`, and `labels`.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
library(dplyr)  
library(randomForest)  
library(tibble)  
  
data(esSim)  
print(esSim)  
  
# expression data  
dat = exprs(esSim)
```

```
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first 6 probes (3 OE probes, 2 UE probes, and 1 NE probe)
pDat$probe1 = dat[1,]
pDat$probe2 = dat[2,]
pDat$probe3 = dat[3,]
pDat$probe4 = dat[4,]
pDat$probe5 = dat[5,]
pDat$probe6 = dat[6,]

print(pDat[1:2, ])

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

pDat$grp = factor(pDat$grp)

rf_m = randomForest(
  x = pDat[, c(3:8)],
  y = pDat$grp,
  importance = TRUE, proximity = TRUE
)

statVisual(type = 'ImpPlot', model = rf_m)

ImpPlot(model = rf_m)
```

Description

Calculate principal components when data contains missing values.

Usage

```
iprcomp(dat, center = TRUE, scale. = FALSE)
```

Arguments

dat	n by p matrix. rows are subjects and columns are variables
center	logical. Indicates if each row of dat needs to be mean-centered
scale.	logical. Indicates if each row of dat needs to be scaled to have variance one

Details

We first set missing values as median of the corresponding variable, then call the function `prcomp`. This is a very simple solution. The user can use their own imputation methods before calling `prcomp`.

Value

A list of 3 elements

sdev	square root of the eigen values
rotation	a matrix with columns are eigen vectors, i.e., projection direction
x	a matrix with columns are principal components

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
# generate simulated data
set.seed(1234567)
dat.x = matrix(rnorm(500), nrow = 100, ncol = 5)
dat.y = matrix(rnorm(500, mean = 2), nrow = 100, ncol = 5)
dat = rbind(dat.x, dat.y)
grp = c(rep(0, 100), rep(1, 100))
print(dim(dat))

res = iprcomp(dat, center = TRUE, scale. = FALSE)

# for each row, set one artificial missing value
dat.na=dat
nr=nrow(dat.na)
nc=ncol(dat.na)
for(i in 1:nr)
{
  posi=sample(x=1:nc, size=1)
  dat.na[i,posi]=NA
}

res.na = iprcomp(dat.na, center = TRUE, scale. = FALSE)

##
# pca plot
```

```
##
par(mfrow = c(3,1))
# original data without missing values
plot(x = res$x[,1], y = res$x[,2], xlab = "PC1", ylab = "PC2")
# perturbed data with one NA per probe
# the pattern of original data is captured
plot(x = res.na$x[,1], y = res.na$x[,2], xlab = "PC1", ylab = "PC2", main = "with missing values")
par(mfrow = c(1,1))
```

LinePlot

Compare Groups Based on Trajectory Plots

Description

Compare groups based on trajectory plots. Trajectories belonging to different groups will have different colors.

Usage

```
LinePlot(
  data,
  x,
  y,
  sid,
  group = NULL,
  xFlag = FALSE,
  points = TRUE,
  point.size = 1,
  theme_classic = TRUE,
  xlab = x,
  ylab = y,
  title = "Trajectory plot",
  xLevel = NULL,
  addThemeFlag = TRUE,
  ...)
```

Arguments

data	A data frame. Rows are subjects; Columns are variables describing the subjects.
x	character. The column name of data that indicates the time.
y	character. The column name of data that indicates the variable on y axis
sid	character. The column name of data that indicates the subject id.
group	character. The column name of data that indicates the subject groups. The trajectories of subjects in the same group will have the same color.
xFlag	logical. Indicate if x should be treated as continuous (xFlag=TRUE)

<code>points</code>	logical. Indicates if points will be added to the trajectories on the coordinate (x, y).
<code>point.size</code>	numeric. size of the data points on the trajectories
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>xlab</code>	character. x axis label
<code>ylab</code>	character. y axis label
<code>title</code>	character. title of plot
<code>xLevel</code>	character. A character vector indicating the order of the elements of x to be shown on x-axis if <code>is.null(x)==FALSE</code> .
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with the following 9 elements: data, layers, scales, mapping, theme, coordinates, facet, plot_env, and labels.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(longDat)

print(dim(longDat))
print(longDat[1:3,])

print(table(longDat$time, useNA = "ifany"))
print(table(longDat$grp, useNA = "ifany"))
print(table(longDat$sid, useNA = "ifany"))

print(table(longDat$time, longDat$grp))

statVisual(type = "LinePlot",
  data = longDat,
  x = 'time',
  y = 'y',
  sid = 'sid',
  group = 'grp')

LinePlot(
  data = longDat,
  x = 'time',
  y = 'y',
  sid = 'sid',
  group = 'grp')
```

 longDat

A Simulated Dataset for Longitudinal Data Analysis

Description

A simulated dataset for longitudinal data analysis.

Usage

```
data("longDat")
```

Format

A data frame with 540 observations on the following 4 variables.

sid subject id

time time points. A factor with levels time1 time2 time3 time4 time5 time6

y numeric. outcome variable

grp subject group. A factor with levels grp1 grp2 grp3

Details

The dataset is generated from the following mixed effects model for repeated measures:

$$y_{ij} = \beta_{0i} + \beta_1 t_j + \beta_2 \text{grp}_{2i} + \beta_3 \text{grp}_{3i} + \beta_4 \times (t_j \times \text{grp}_{2i}) + \beta_5 \times (t_j \times \text{grp}_{3i}) + \epsilon_{ij},$$

where y_{ij} is the outcome value for the i -th subject measured at j -th time point t_j , grp_{2i} is a dummy variable indicating if the i -th subject is from group 2, grp_{3i} is a dummy variable indicating if the i -th subject is from group 3, $\beta_{0i} \sim N(\beta_0, \sigma_b^2)$, $\epsilon_{ij} \sim N(0, \sigma_e^2)$, $i = 1, \dots, n$, $j = 1, \dots, m$, n is the number of subjects, and m is the number of time points.

When $t_j = 0$, the expected outcome value is

$$E(y_{ij}) = \beta_0 + \beta_2 \text{dose}_{2i} + \beta_3 \text{dose}_{3i}.$$

Hence, we have at baseline

$$E(y_{ij}) = \beta_0, \text{ for dose 1 group.}$$

$$E(y_{ij}) = \beta_0 + \beta_2, \text{ for dose 2 group.}$$

$$E(y_{ij}) = \beta_0 + \beta_3, \text{ for dose 3 group.}$$

For dose 1 group, the expected outcome values across time is

$$E(y_{ij}) = \beta_0 + \beta_1 t_j.$$

We also can get the expected difference of outcome values between dose 2 group and dose 1 group, between dose 3 group and dose 1 group, and between dose 3 group and dose 2 group:

$$E(y_{ij} - y_{i'j}) = \beta_2 + \beta_4 t_j, \text{ for subject } i \text{ in dose 2 group and subject } i' \text{ in dose 1 group,}$$

$$E(y_{kj} - y_{i'j}) = \beta_3 + \beta_5 t_j, \text{ for subject } k \text{ in dose 3 group and subject } i' \text{ in dose 1 group,}$$

$$E(y_{kj} - y_{ij}) = (\beta_3 - \beta_2) + (\beta_5 - \beta_4) t_j, \text{ for subject } i \text{ in dose 3 group and subject } i \text{ in dose 2 group.}$$

We set $n = 90$, $m = 6$, $\beta_0 = 5$, $\beta_1 = 0$, $\beta_2 = 0$, $\beta_3 = 0$, $\beta_4 = 2$, $\beta_5 = -2$, $\sigma_e = 1$, $\sigma_b = 0.5$, and $t_{ij} = j, j = 1, \dots, m$.

That is, the trajectories for dose 1 group are horizontal with mean intercept at 5, the trajectories for dose 2 group are linearly increasing with slope 2 and mean intercept 5, and the trajectories for dose 3 group are linearly decreasing with slope -2 and mean intercept 5.

Examples

```
data(longDat)

print(dim(longDat))
print(longDat[1:3,])

print(table(longDat$time, useNA = "ifany"))
print(table(longDat$grp, useNA = "ifany"))
print(table(longDat$sid, useNA = "ifany"))

print(table(longDat$time, longDat$grp))
```

PCA_score

Scatter Plot of 2 Specified Principal Components

Description

Scatter plot of 2 specified principal components. The size of the data points on the PCA plot indicates the Mahalanobis distance (distance between each point and mean value).

Usage

```
PCA_score(
  prcomp_obj,
  data,
  dims = c(1, 2),
  color = NULL,
  MD = TRUE,
```



```

loadings = FALSE,
loadings.color = "black",
loadings.label = FALSE,
title = "pca plot",
addThemeFlag = TRUE)

```

Arguments

<code>prcomp_obj</code>	the object returned by the function <code>prcomp</code> .
<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects. The object <code>prcomp_obj</code> is based on <code>data</code>
<code>dims</code>	a numeric vector with 2 elements indicating which two principal components will be used to draw scatter plot.
<code>color</code>	character. The column name of <code>data</code> that indicates the subject groups. The data points on the PCA plot will be colored by the group info.
<code>MD</code>	logical. Indicate if the Mahalanobis distance (distance between each point and mean value) would be used to indicate the size of data points on the PCA plot
<code>loadings</code>	logical. Indicate if loading plot would be superimposed on the PCA plot. (default: FALSE)
<code>loadings.color</code>	character. Indicate the color of the loading axis.
<code>loadings.label</code>	logical. Indicating if loading labels should be added to the plot. (default: FALSE)
<code>title</code>	character. Figure title.
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.

Value

A list with 9 elements. `data`, `layers`, `scales`, `mapping`, `theme`, `coordinates`, `facet`, `plot_env`, and `labels`.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```

library(factoextra)

data(esSim)
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

```

```

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first 6 probes (3 OE probes, 2 UE probes, and 1 NE probe)
pDat$probe1 = dat[1,]
pDat$probe2 = dat[2,]
pDat$probe3 = dat[3,]
pDat$probe4 = dat[4,]
pDat$probe5 = dat[5,]
pDat$probe6 = dat[6,]

print(pDat[1:2, ])

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

pDat$grp = factor(pDat$grp)

###

pca.obj = iprcomp(pDat[, c(3:8)], scale. = TRUE)

# scree plot
factoextra::fviz_eig(pca.obj, addlabels = TRUE)

# scatter plot of PC1 vs PC2
statVisual(type = 'PCA_score',
           prcomp_obj = pca.obj,
           dims = c(1, 2),
           data = pDat,
           color = 'grp',
           loadings = FALSE)

PCA_score(prcomp_obj = pca.obj,
          dims = c(1, 3),
          data = pDat,
          color = 'grp',
          loadings = FALSE)

```

Description

Plot of weighted average proportion variance versus effects in principal variance component analysis (PVCA).

Usage

```
PVCA(  
  clin_data,  
  clin_subjid,  
  gene_data,  
  pct_threshold = 0.8,  
  batch.factors,  
  theme_classic = FALSE,  
  addThemeFlag = TRUE,  
  ...)
```

Arguments

<code>clin_data</code>	A data frame containing clinical information, including an id variable that corresponds to rownames of <code>gene_data</code>
<code>clin_subjid</code>	character. The column name of <code>clin_data</code> that indicates subject id. It corresponds to the rowname of <code>gene_data</code> .
<code>gene_data</code>	A data frame with genes as rows and subjects as columns.
<code>pct_threshold</code>	numeric. The percentile value of the minimum amount of the variabilities that the selected principal components need to explain
<code>batch.factors</code>	character. A vector of factors that the mixed linear model will be fit on.
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with 9 elements. `data`, `layers`, `scales`, `mapping`, `theme`, `coordinates`, `facet`, `plot_env`, and `labels`.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
library(pvca)  
  
data(esSim)
```

```
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# create a fake Batch variable
esSim$Batch=c(rep("A", 4), rep("B", 6), rep("C", 10))
# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

statVisual(type = 'PVCA',
           clin_data = pData(esSim),
           clin_subjid = "sid",
           gene_data = exprs(esSim),
           batch.factors = c("grp", "Batch"))

PVCA(
  clin_data = pData(esSim),
  clin_subjid = "sid",
  gene_data = exprs(esSim),
  batch.factors = c("grp", "Batch"))
```

stackedBarPlot

Draw Stacked Bar Plots

Description

Draw stacked bar plots.

Usage

```
stackedBarPlot(dat,
               catVar,
               group,
               xlab = catVar,
               ylab = "Count",
               group.lab = group,
               title = "Stacked barplots of counts",
```

```

catVarLevel = NULL,
groupLevel = NULL,
addThemeFlag = TRUE)

```

Arguments

<code>dat</code>	A data frame object. Rows are subjects and columns are variables.
<code>catVar</code>	character. The name of the categorical variable to be shown in x-axis.
<code>group</code>	character. The name of variable indicating groups of subjects.
<code>xlab</code>	character. Label for x-axis.
<code>ylab</code>	character. Label for y-axis.
<code>group.lab</code>	character. Label for group in legend.
<code>title</code>	character. Figure title.
<code>catVarLevel</code>	character. A vector indicating the order of the unique elements of <code>catVar</code> should be shown in x-axis.
<code>groupLevel</code>	character. A vector indicating the order of the unique elements of <code>group</code> should be shown in figure and in legend.
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.

Value

A list of the following 9 elements: “data”, “layers”, “scales”, “mapping”, “theme”, “coordinates”, “facet”, “plot_env”, “labels”.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```

data(genoSim)

pDat = pData(genoSim)
geno = exprs(genoSim)

pDat$snp1 = geno[1,]

print(table(pDat$snp1, pDat$grp, useNA="ifany"))

stackedBarPlot(dat = pDat,
  catVar = "snp1",
  group = "grp",
  xlab = "snp1",
  ylab = "Count",
  group.lab = "grp",
  title = "Stacked barplots of counts",

```

```
catVarLevel = NULL)
```

statVisual	<i>The Wrapper Function Incorporating All Wrapper Functions in statVisual</i>
------------	---

Description

The wrapper function incorporating all wrapper functions in statVisual.

Usage

```
statVisual(type, ...)
```

Arguments

type	character. Indicate the functions to be called. It can take the following values: “BiAxisErrBar”, “Box”, “BoxROC”, “cv_glmnet_plot”, “Den”, “Dendro”, “ErrBar”, “Heat”, “Hist”, “ImpPlot”, “iprcomp”, “LinePlot”, “PCA_score”, “PVCA”, “statVisual”, “Volcano”, “XYscatter”.
...	input parameters for the functions specified by type.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

See Also

[BiAxisErrBar](#), [Box](#), [BoxROC](#), [cv_glmnet_plot](#), [Den](#), [Dendro](#), [ErrBar](#), [Heat](#), [Hist](#), [ImpPlot](#), [iprcomp](#), [LinePlot](#), [PCA_score](#), [PVCA](#), [statVisual](#), [Volcano](#), [XYscatter](#).

Examples

```
data(esSim)
print(esSim)

# expression data
dat = exprs(esSim)
print(dim(dat))
print(dat[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat[1:2,])

# feature data
```

```
fDat = fData(esSim)
print(dim(fDat))
print(fDat[1:2,])

# choose the first probe which is over-expressed in cases
pDat$probe1 = dat[1,]

# check histograms of probe 1 expression in cases and controls
print(table(pDat$grp, useNA = "ifany"))

statVisual(type = 'Hist',
           data = pDat,
           y = 'probe1',
           group = 'grp')
```

Volcano

Volcano Plot

Description

Volcano plot with the option to label the significant results.

Usage

```
Volcano(
  resFrame,
  stats,
  p.value,
  group = NULL,
  xlab = "logFC",
  ylab = "-log10(p value)",
  title = NULL,
  vline.col = "orange",
  hline.col = "dodgerblue",
  vline = list(xintercept = c(-1, 1), label = c(-1, 1)),
  hline = list(
    yintercept = c(-log10(0.05),
                  -log10(0.05/nrow(resFrame)),
                  -log10(max(resFrame[p.adjust(resFrame[, p.value],
                                                method = "fdr") <= 0.05, p.value])))),
    label = c("p value: 0.05", "Bonferroni: 0.05", "FDR: 0.05")),
  rowname.var = NULL,
  point.size = 3,
  theme_classic = TRUE,
  addThemeFlag = TRUE,
  ...)
```

Arguments

<code>resFrame</code>	A data frame stored information about the results, including gene id, statistic (e.g., log fold change, odds ratio), p-value, and significance of a gene.
<code>stats</code>	character. The column name of <code>resFrame</code> that indicates the effect of a gene.
<code>p.value</code>	character. The column name of <code>resFrame</code> that indicates the p-value.
<code>group</code>	character. The column name of <code>resFrame</code> that indicates the significance of a gene.
<code>xlab</code>	x axis label
<code>ylab</code>	y axis label
<code>title</code>	title of the plot
<code>vline.col</code>	color of the vertical lines (default: "orange")
<code>hline.col</code>	color of the horizontal lines (default: "dodgerblue")
<code>vline</code>	A list with two elements: "xintercept" and "label", where the former element is a numeric vector indicating the x-axis location to draw vertical color lines and the latter element is list of labels for the elements in "xintercept".
<code>hline</code>	A list with two elements: "yintercept" and "label", where the former element is a numeric vector indicating the y-axis location to draw horizontal color lines and the latter element is list of labels for the elements in "xintercept".
<code>rowname.var</code>	character. The column name of <code>resFrame</code> that indicates which variable will be used to label the significant results in the volcano plot. The elements of this column for non-significant results should be set to be NA.
<code>point.size</code>	size of data points in the plot.
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with 9 elements. `data`, `layers`, `scales`, `mapping`, `theme`, `coordinates`, `facet`, `plot_env`, and `labels`.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
library(ggrepel)
library(limma)

# load the simulated dataset
data(esSim)
```



```
print(esSim)

# expression levels
y = exprs(esSim)
print(dim(y))
print(y[1:2,])

# phenotype data
pDat = pData(esSim)
print(dim(pDat))
print(pDat)

# design matrix
design = model.matrix(~grp, data = pDat)
print(design)

options(digits = 3)

# Ordinary fit
fit <- lmFit(y, design)
fit2 <- eBayes(fit)

# get result data frame
resFrame = topTable(fit2,coef = 2, number = nrow(esSim))
print(dim(resFrame))
print(resFrame[1:2,])
resFrame$sigFlag = resFrame$adj.P.Val < 0.05

resFrame$probe = rownames(resFrame)
# make sure set NA to genes non-differentially expressed
resFrame$probe[which(resFrame$sigFlag == FALSE)] = NA

print(resFrame[1:2,])
print(table(resFrame$sigFlag, useNA = "ifany"))

statVisual(type = 'Volcano',
           resFrame = resFrame,
           stats = 'logFC',
           p.value = 'P.Value',
           group = 'sigFlag',
           rowname.var = 'probe',
           point.size = 1)

Volcano(
  resFrame = resFrame,
  stats = 'logFC',
  p.value = 'P.Value',
  group = 'sigFlag',
  rowname.var = 'probe',
  point.size = 1)
```

XYscatter

*Compare Groups Based on Scatter Plots***Description**

Compare groups based on scatter plots.

Usage

```
XYscatter(
  data,
  x,
  y,
  group = NULL,
  alpha = 1,
  point.size = 3,
  xlab = x,
  ylab = y,
  group.lab = group,
  title = "Scatter plot",
  theme_classic = TRUE,
  addThemeFlag = TRUE,
  ...)
```

Arguments

<code>data</code>	A data frame. Rows are subjects; Columns are variables describing the subjects.
<code>x</code>	character. The column name of <code>data</code> that indicates the variable on the x axis of the scatter plot
<code>y</code>	character. The column name of <code>data</code> that indicates the variable on the y axis of the scatter plot
<code>group</code>	character. The column name of <code>data</code> that indicates the subject groups. The scatter plot will be drawn for each of the subject group. It also indicates the colors of the data points in the scatter plots.
<code>alpha</code>	Transparency of histogram inside color.
<code>point.size</code>	numeric. Indicate the size of the data points
<code>xlab</code>	x axis label
<code>ylab</code>	y axis label
<code>group.lab</code>	label of group variable
<code>title</code>	title of the plot
<code>theme_classic</code>	logical. Use classic background without grids (default: TRUE).
<code>addThemeFlag</code>	logical. Indicates if light blue background and white grid should be added to the figure.
<code>...</code>	other input parameters for facet & theme

Value

A list with 9 elements. data, layers, scales, mapping, theme, coordinates, facet plot_env, and labels.

Author(s)

Wenfei Zhang <Wenfei.Zhang@sanofi.com>, Weiliang Qiu <Weiliang.Qiu@sanofi.com>, Xuan Lin <Xuan.Lin@sanofi.com>, Donghui Zhang <Donghui.Zhang@sanofi.com>

Examples

```
data(diffCorDat)

print(dim(diffCorDat))
print(diffCorDat[1:2,])

statVisual(type = 'XYscatter',
  data = diffCorDat,
  x = 'probe1',
  y = 'probe2',
  group = 'grp',
  title = 'Scatter Plot: probe1 vs probe2')

XYscatter(
  data = diffCorDat,
  x = 'probe1',
  y = 'probe2',
  group = 'grp',
  title = 'Scatter Plot: probe1 vs probe2')
```

Index

* datasets

diffCorDat, 17
esSim, 20
genoSim, 21
longDat, 31

* method

barPlot, 2
BiAxisErrBar, 5
Box, 6
BoxROC, 9
cv_glmnet_plot, 11
Den, 12
Dendro, 14
ErrBar, 18
Heat, 22
Hist, 23
ImpPlot, 26
iprcomp, 27
LinePlot, 29
PCA_score, 32
PVCA, 34
stackedBarPlot, 36
statVisual, 38
Volcano, 39
XYscatter, 42

barPlot, 2
BiAxisErrBar, 5, 38
Box, 6, 38
BoxROC, 9, 38

cv_glmnet_plot, 11, 38

Den, 12, 38
Dendro, 14, 38
diffCorDat, 17

ErrBar, 18, 38
esSim, 20

gbm, 26

genoSim, 21
glmnet, 11

hclust, 15
Heat, 22, 38
Hist, 23, 38

ImpPlot, 26, 38
iprcomp, 27, 38

LinePlot, 29, 38
lmFit, 20
longDat, 31

PCA_score, 32, 38
pheatmap, 22
prcomp, 33
PVCA, 34, 38

randomForest, 26

stackedBarPlot, 36
statVisual, 38, 38

Volcano, 38, 39

XYscatter, 38, 42