Package 'statVisual'

December 15, 2019

Type Package

Title Statistical Visualization Tools

Version 1.1.9

Date 2019-12-10

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

Imports GGally, gplots, 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, multigroup

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>

Repository CRAN

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

R topics documented:

`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 $+/-$ 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 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 = '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")
```

BiAxisErrBar *Compare Patterns of Two Outcomes in One Scatter Plot*

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 $+/-$ 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

dat	A data frame. Rows are subjects; Columns are variables describing the subjects.
group	character. A categorical variable in data that indicates the predictor.
y.left	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.
y.right	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.
title	character. title of the plot.
xlab	character. Label for the x-axis.
ylab.left	character. Label for the left-side y-axis.
ylab.right	character. Label for the right-side y-axis.
legendLabel	character. Legend label.
delta	numeric. A small number so that the second error bar plot will shift delta distance from the first error bar plot.
cvThresh	numeric. A small positive number. If the coefficient of variation (CV) is smaller than cvThresh, 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)
library(multigroup)

data(oliveoil)
print(oliveoil[1:2,])

print(table(oliveoil$Group, useNA="ifany"))

statVisual(type="BiAxisErrBar",
  dat= oliveoil,
  group = "Group",
  y.left = "K270",
  y.right = "syrup")


BiAxisErrBar(
  dat= oliveoil,
  group = "Group",
  y.left = "K270",
  y.right = "syrup")
```

Box *Compare Groups Based on Boxplots Across Time*

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

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 (e.g. observation time)
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 (e.g., treatment group). The boxplots will be drawn for each of the subject group within each category of x.
fill	boxplot inside color indicated by the categories of group
theme_classic	logical. Use classic background without grids (default: TRUE).
fill.alpha	boxplot transparency
box.width	boxplot width
dodge.width	dodge width for boxplot and jitter (prevent overlapping)
jitter	logical. plot jitter or not, default TRUE
jitter.alpha	jitter transparency
jitter.width	jitter width in boxplot
point.size	size of a jitter point
xlab	character. x axis label
ylab	character. y axis label
group.lab	label of group variable
fill.lab	label of fill variable

title	character. title of plot
line	line connect boxes, default plot mean, can be set as 'median', or NULL (no line)
line.color	connection line color, only available when group = NULL
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 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

data	A data frame. Rows are subjects; Columns are variables describing the subjects.
group.var	character. The column name of data that indicates the two subject groups. It also indicates the color of the two boxplots.
y	character. The column name of data that indicates the variable, for which the box will be drawn.
box.xlab	character. boxplot x axis label (default: group.var)
box.ylab	character. boxplot y axis label (default: y)
box.group.lab	character. boxplot legend label (default: group.var)
jitter.alpha	numeric. transparency of jitters
jitter.width	numeric. width of jitters
point.size	size of a jitter point
roc.xlab	character. roc curve x axis label (default: Specificities)
roc.ylab	character. roc curve y axis label (default: Sensitivities)
addThemeFlag	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

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

data	A data frame. Rows are subjects; Columns are variables describing the subjects.
y	character. The column name of data that indicates the variable, for which the histogram will be drawn. The string y can also indicate a function of the variable, e.g., $\log(y)$.
group	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.
fill	grouping variable, density inside color
border.color	density border color, only available when group & fill are NULL
inner.color	density inside color, only available when group & fill are NULL
theme_classic	Use classic background without grids (default: FALSE)
xlab	x axis label
ylab	y axis label
group.lab	label of group variable
title	title of plot
alpha	transparency of density inside color
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 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

x	A data frame. Rows are subjects; Columns are variables describing the subjects.
group	character. The column name of data that indicates the subject groups. The nodes of the dendrogram will be colored by info provided by group.
xlab	x axis label
ylab	y axis label
title	title of the plot
cor.use	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".
cor.method	character. Indicate which type of correlation coefficients will be calculated: "pearson", "kendall", "spearman".
distance	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-|cor|" (i.e., using (1−|correlation coefficients|) as distance).
distance.method	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-|cor|" (i.e., using (1−|correlation coefficients|) as distance).
hclust.method	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.
yintercept	numeric. A line indicating the height of the dendrogram, for example, indicating where the dendrogram should be cut to obtain clusters.
theme_classic	logical. Use classic background without grids (default: TRUE).
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 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 $probe2_i = -probe1_i + e_i$, $i = 1, \ldots, nCases$, 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 $+/-$ 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

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 dotplots 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 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 ($sd = 0.3\sqrt{4/\chi^2_4}$). 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

data	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.
group	character. The column name of data that indicates the subject groups. The row names of the heatmap will be colored based on group.
fontsize_col	x axis label font size
fontsize_row	y axis label font size
scale	character. Indicate how data will be scaled: "none" (i.e., no scaling), "row" (i.e., row scaled), "column" (i.e., column scaled).
cluster_rows	logic. Indicates if rows should be clustered.
cluster_cols	logic. Indicates if columns should be clustered.
color	vector indicating colors used in heatmap
angle_col	angle of the column labels. Please refer to the manual in pheatmap
...	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')
```

`Hist` *Compare Groups Based on Histograms*

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

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

model	An object returned by randomForest or gbm
theme_classic	logical. Use classic background without grids (default: TRUE).
n.trees	integer. The number of trees used to generate the plot used in the function summary.gbm in the R library gbm. Only the first n.trees trees will be used.
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 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)
```

---

iprcomp *Improved Function for Obtaining Principal Components*

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)
points	logical. Indicates if points will be added to the trajectories on the coordinate (x, y).
point.size	numeric. size of the data points on the trajectories
theme_classic	logical. Use classic background without grids (default: TRUE).
xlab	character. x axis label
ylab	character. y axis label
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 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:

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

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, $\beta_{0i} \sim N\left(\beta_0, \sigma_b^2\right)$, $\epsilon_{ij} \sim N\left(0, \sigma_e^2\right)$, $i = 1, \ldots, n, j = 1, \ldots, m$, n is the number of subjects, and m is the number of time points.

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

$$E\left(y_{ij}\right) = \beta_0 + \beta_2 dose_{2i} + \beta_3 dose_{3i}.$$

Hence, we have at baseline

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

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

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

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

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

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:

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

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

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

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 $t_{ij} = j, j = 1, \ldots, m$.

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

prcomp_obj	the object returned by the function prcomp.
data	A data frame. Rows are subjects; Columns are variables describing the subjects. The object prcomp_obj is based on data
dims	a numeric vector with 2 elements indicating which two principal components will be used to draw scatter plot.
color	character. The column name of data that indicates the subject groups. The data points on the PCA plot will be colored by the group info.
MD	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
loadings	logical. Indicate if loading plot would be superimposed on the PCA plot. (default: FALSE)
loadings.color	character. Indicate the color of the loading axis.
loadings.label	logical. Indicating if loading labels should be added to the plot. (default: FALSE)
title	character. Figure title.
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(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)
```

PVCA *Principal Variance Component Analysis (PVCA)*

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

clin_data	A data frame containing clinical information, including an id variable that corresponds to rownames of gene_data
clin_subjid	character. The column name of clin_data that indicates subject id. It corresponds to the rowname of gene_data.
gene_data	A data frame with genes as rows and subjects as columns.
pct_threshold	numeric. The percentile value of the minimum amount of the variabilities that the selected principal components need to explain
batch.factors	character. A vector of factors that the mixed linear model will be fit on.
theme_classic	logical. Use classic background without grids (default: TRUE).
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 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

dat	A data frame object. Rows are subjects and columns are variables.
catVar	character. The name of the cateogrical variable to be shown in x-axis.
group	character. The name of variable indicating groups of subjects.
xlab	character. Label for x-axis.
ylab	character. Label for y-axis.
group.lab	character. Label for group in legend.
title	character. Figure title.
catVarLevel	character. A vector indicating the order of the unique elements of catVar should be shown in x-axis.
groupLevel	character. A vector indicating the order of the unique elements of group should be shown in figure and in legend.
addThemeFlag	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` *The Wrapper Function Incorporating All Wrapper Functions in statVisual*

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

resFrame	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.
stats	character. The column name of resFrame that indicates the effect of a gene.
p.value	character. The column name of resFrame that indicates the p-value.
group	character. The column name of resFrame that indicates the significance of a gene.
xlab	x axis label
ylab	y axis label
title	title of the plot
vline.col	color of the vertical lines (default: "orange")
hline.col	color of the horizontal lines (default: "dodgerblue")
vline	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".
hline	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".
rowname.var	character. The column name of resFrame 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.
point.size	size of data points in the plot.
theme_classic	logical. Use classic background without grids (default: TRUE).
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 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

data	A data frame. Rows are subjects; Columns are variables describing the subjects.
x	character. The column name of data that indicates the variable on the x axis of the scatter plot
y	character. The column name of data that indicates the variable on the y axis of the scatter plot
group	character. The column name of data 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.
alpha	Transparency of histogram inside color.
point.size	numeric. Indicate the size of the data points
xlab	x axis label
ylab	y axis label

group.lab	label of group variable
title	title of the plot
theme_classic	logical. Use classic background without grids (default: TRUE).
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 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