Package 'Glimma'

October 16, 2019

Type Package

Title Interactive HTML graphics

Version 1.12.0

Date 2016-12-21

Description This package generates interactive visualisations for analysis of RNA-sequencing data using output from limma, edgeR or DESeq2 packages in an HTML page. The interactions are built on top of the popular static representations of analysis results in order to provide additional information.

biocViews ImmunoOncology, DifferentialExpression, GeneExpression, Microarray, ReportWriting, RNASeq, Sequencing, Visualization

Depends R (>= 3.4.0)

Imports edgeR, grDevices, jsonlite, methods, stats, S4Vectors, utils

Suggests BiocStyle, IRanges, GenomicRanges, SummarizedExperiment, DESeq2, limma, testthat, knitr, rmarkdown, pryr

License GPL-3 | file LICENSE

URL https://github.com/Shians/Glimma

BugReports https://github.com/Shians/Glimma/issues

RoxygenNote 6.0.1

NeedsCompilation no

LazyData true

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/Glimma

git_branch RELEASE_3_9

git_last_commit d021742

git_last_commit_date 2019-05-02

Date/Publication 2019-10-15

Author Shian Su [aut, cre],
Matthew Ritchie [aut],
Charity Law [aut],
Stuart Lee [ctb]

Maintainer Shian Su <su.s@wehi.edu.au>

R topics documented:

`arraydata` *Example microarray for the study of Ezh2.*

Description

Example microarray for the study of Ezh2.

Author(s)

Bhupinder Pal, Toula Bouras, Wei Shi, Francois Vaillant, Julie M. Sheridan, Naiyang Fu, Kelsey Breslin, Kun Jiang, Matthew E. Ritchie, Matthew Young, Geoffrey J. Lindeman, Gordon K. Smyth, Jane E. Visvader

References

http://www.cell.com/cell-reports/abstract/S2211-1247(13)00007-7

as.hexcol *Numeric to hex colour converter*

Description

Convert numbers and R colour strings into corresponding hex codes for colours

Usage

```
as.hexcol(x)
```

Arguments

x	the colour value(s) to be converted to hex values.

Value

hex codes for colours

Examples

```
as.hexcol(c(1, 2, 3))
as.hexcol(c("red", "black", "green"))
```

glBar *Glimma MD Plot*

Description

Create an interactive bar plot object.

Usage

```
glBar(x, ...)
```

Arguments

x	the data.frame containing data to plot.
...	additional arguments depending on input object type.

Value

A chart object containing the information to create an interactive bar plot.

Author(s)

Shian Su

See Also

glBar.default

Examples

```
data(mtcars)
counts <- table(mtcars$gear)
data <- data.frame(nGears=as.numeric(names(counts)), Count=as.numeric(counts))

plot1 <- glBar(data, "Count", "nGears", ylab="Number of Gears")
glimma(plot1, layout=c(1,1), launch=TRUE)
```

---

glBar.default *Glimma MD Plot*

Description

Default method for interactive bar plot.

Usage

```
## Default S3 method:
glBar(x, yval, names.arg = rownames(x), ndigits = NULL,
  signif = 6, xlab = NULL, ylab = yval, main = NULL, height = 400,
  width = 500, colval = NULL, annot = yval, flag = NULL, info = NULL,
  ...)
```

Arguments

x	the data.frame containing data to plot.
yval	the column name for the x-axis values.
names.arg	the column name for the label on each bar.
ndigits	the number of digits after the decimal to round to in the tooltip (overrides signif).
signif	the number of significant figures to display in the tooltip.
xlab	the label on the x-axis.
ylab	the label on the y-axis.
main	the title for the plot.
height	the height of the plot (in pixels).
width	the width of the plot (in pixels).
colval	the colours for each data point.
annot	the columns to display in the tooltip.
flag	the special flag to indicate special plot.
info	additional information for plotting.
...	additional arguments.

Value

A chart object containing the information to create an interactive bar plot.

Author(s)

Shian Su

Examples

```
data(mtcars)
counts <- table(mtcars$gear)
data <- data.frame(nGears=as.numeric(names(counts)), Count=as.numeric(counts))

plot1 <- glBar(data, "Count", "nGears", ylab="Number of Gears")
glimma(plot1, layout=c(1,1), launch=TRUE)
```

---

glimma *Glimma plot manager*

Description

Core glimma plot manager. Generates environment for glimma plots.

Usage

```
glimma(..., layout = c(1, 1), path = getwd(), folder = "glimma-plots",
  html = "index", overwrite = TRUE, launch = TRUE)
```

Arguments

...	the jschart or jslink objects for processing.
layout	the numeric vector representing the number of rows and columns in plot window.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
overwrite	the option to overwrite existing folder if it already exists.
launch	TRUE to launch plot after call.

Value

Generates interactive plots based on filling layout row by row from left to right.

Examples

```
data(iris)

plot1 <- glScatter(iris, xval="Sepal.Length", yval="Sepal.Width", colval="Species")
glimma(plot1, c(1,1))
```

gllink *Plot linkages*

Description

Helper function for writing the link properties in interactive Glimma plots

Usage

```
gllink(from, to, src = "none", dest = "none", flag = "none",
  both = FALSE, info = "none")
```

Arguments

from	the index of the plot from which the event is dispatched.
to	the index of the plot which receives the event and performs an action.
src	the action that is performed in the "from" plot.
dest	the action that is performed in the "to" plot.
flag	indicates special links for particular chart types.
both	creates symmetric links whereby the "dest" action in "to" also triggers the "src" action in "from".
info	additional info for creating the link.

Value

a link object containing the plot linking information.

Examples

```
data(iris)
data <- data.frame(Name=paste("Flower", 1:nrow(iris), sep="-"), iris)

plot1 <- glScatter(data, xval="Sepal.Length", yval="Sepal.Width", colval="Species")
plot2 <- glScatter(data, xval="Species", yval="Petal.Length", colval="Species")
link1 <- gllink(1, 2, src="hover", dest="hover", both=TRUE)
glimma(plot1, plot2, link1, layout=c(1,2))
```

glMDPlot *Glimma MD Plot*

Description

Draw an interactive MD plot

Usage

```
glMDPlot(x, ...)
```

Arguments

x	the DE object to plot.
...	additional arguments affecting the plots produced. See specific methods for detailed arguments.

Value

Draws a two-panel interactive MD plot in an html page. The left plot shows the log-fold-change vs average expression. The right plot shows the expression levels of a particular gene of each sample. Hovering over points on left plot will plot expression level for corresponding gene, clicking on points will fix the expression plot to gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot.

Author(s)

Shian Su

See Also

glMDPlot.default, glMDPlot.DGELRT, glMDPlot.DGEExact, glMDPlot.MArrayLM, glMDPlot.DESeqDataSet

Examples

```
library(limma)
library(edgeR)

data(lymphomaRNAseq)
x <- lymphomaRNAseq

sel <- rowSums(cpm(x$counts)>0.5)>=3
x <- x[sel,]

genotype <- relevel(x$samples$group, "Smchd1-null")
x <- calcNormFactors(x, method="TMM")
des <- model.matrix(~genotype)

## Apply voom with sample quality weights and fit linear model
v <- voomWithQualityWeights(x, design=des, normalization="none", plot=FALSE)
vfit <- lmFit(v,des)

## Apply treat relative to a fold-change of 1.5
vtfit <- treat(vfit,lfc=log2(1.5))
vfit <- eBayes(vfit)
results <- decideTests(vfit,p.value=0.01)


glMDPlot(vfit, counts=x$counts, anno=x$genes, groups=genotype, samples=1:7,
         status=results[,2], main="MD plot: Wild-type vs Smchd1",
         display.columns=c("Symbols", "GeneID", "GeneName"),
         folder="Smchd1-Lymphoma")
```

`glMDPlot.default` *Glimma MD Plot*

Description

Draw an interactive MD plot from a data.frame

Usage

```
## Default S3 method:
glMDPlot(x, xval, yval, counts = NULL, anno = NULL,
  groups = NULL, samples = NULL, status = rep(0, nrow(x)),
  transform = FALSE, main = "", xlab = xval, ylab = yval,
  side.main = "GeneID", side.xlab = "Group", side.ylab = "Expression",
  side.log = FALSE, side.gridstep = ifelse(!transform || side.log, FALSE,
  0.5), jitter = 30, display.columns = side.main, cols = c("#00bfff",
  "#858585", "#ff3030"), sample.cols = rep("#1f77b4", ncol(counts)),
  path = getwd(), folder = "glimma-plots", html = "MD-Plot",
  launch = TRUE, ...)
```

Arguments

`x`	the data.frame object containing expression and fold change values.
`xval`	the column to plot on x axis of left plot.
`yval`	the column to plot on y axis of left plot.
`counts`	the matrix of expression values, with samples in columns.
`anno`	the data.frame containing gene annotations.
`groups`	the factor containing experimental groups of the samples.
`samples`	the names of the samples.
`status`	vector giving the control status of data point, of same length as the number of rows of object. If NULL, then all points are plotted in the default colour.
`transform`	TRUE if counts should be log-cpm transformed.
`main`	the title for the left plot.
`xlab`	the label on the x axis for the left plot.
`ylab`	the label on the y axis for the left plot.
`side.main`	the column containing mains for right plot.
`side.xlab`	label for x axis on right plot.
`side.ylab`	label for y axis on right plot.
`side.log`	TRUE to plot expression on the right plot on log scale.
`side.gridstep`	intervals along which to place grid lines on y axis. Currently only available for linear scale.
`jitter`	the amount of jitter to apply to the samples in the expressions plot.
`display.columns`	character vector containing names of columns to display in mouseover tooltips and table.

cols	vector of strings denoting colours corresponding to control status -1, 0 and 1. (may be R named colours or Hex values)
sample.cols	vector of strings denoting colours for each sample point on the expression plot.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
launch	TRUE to launch plot after call.
...	additional arguments to be passed onto the MD plot. (main, xlab, ylab can be set for the left plot)

Value

Draws a two-panel interactive MD plot in an html page. The left plot shows the log-fold-change vs average expression. The right plot shows the expression levels of a particular gene of each sample. Hovering over points on left plot will plot expression level for corresponding gene, clicking on points will fix the expression plot to gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot.

Author(s)

Shian Su

---

glMDPlot.DESeqDataSet *Glimma MD Plot*

Description

Draw an interactive MD plot from a DESeqDataSet object

Usage

```
## S3 method for class 'DESeqDataSet'
glMDPlot(x, counts = NULL, anno, groups,
  samples = NULL, status = rep(0, nrow(x)), transform = FALSE,
  main = "", xlab = "Mean Expression", ylab = "log-fold-change",
  side.xlab = "Group", side.ylab = "logMean", side.log = FALSE,
  side.gridstep = ifelse(!transform || side.log, FALSE, 0.5), jitter = 30,
  side.main = "GeneID", display.columns = NULL, cols = c("#00bfff",
  "#858585", "#ff3030"), sample.cols = rep("#1f77b4", ncol(x)),
  path = getwd(), folder = "glimma-plots", html = "MD-Plot",
  launch = TRUE, ...)
```

Arguments

x	the DESeqDataSet object.
counts	the matrix of expression values, with samples in columns.
anno	the data.frame containing gene annotations.
groups	the factor containing experimental groups of the samples.
samples	the names of the samples.

status	vector giving the control status of data point, of same length as the number of rows of object. If NULL, then all points are plotted in the default colour.
transform	TRUE if counts should be log-cpm transformed.
main	the title for the left plot.
xlab	label for x axis on left plot.
ylab	label for y axis on left plot.
side.xlab	label for x axis on right plot.
side.ylab	label for y axis on right plot.
side.log	TRUE to plot expression on the right plot on log scale.
side.gridstep	intervals along which to place grid lines on y axis. Currently only available for linear scale.
jitter	the amount of jitter to apply to the samples in the expressions plot.
side.main	the column containing mains for right plot.
display.columns	character vector containing names of columns to display in mouseover tooltips and table.
cols	vector of strings denoting colours corresponding to control status -1, 0 and 1. (may be R named colours or Hex values)
sample.cols	vector of strings denoting colours for each sample point on the expression plot.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
launch	TRUE to launch plot after call.
...	additional arguments to be passed onto the MD plot. (main, xlab, ylab can be set for the left plot)

Value

Draws a two-panel interactive MD plot in an html page. The left plot shows the log-fold-change vs average expression. The right plot shows the expression levels of a particular gene of each sample. Hovering over points on left plot will plot expression level for corresponding gene, clicking on points will fix the expression plot to gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot.

Author(s)

Shian Su

`glMDPlot.DESeqResults` *Glimma MD Plot*

Description

Draw an interactive MD plot from a DESeqResults object

Usage

```
## S3 method for class 'DESeqResults'
glMDPlot(x, counts = NULL, anno, groups,
  samples = NULL, status = rep(0, nrow(x)), transform = FALSE,
  main = "", xlab = "Mean Expression", ylab = "log-fold-change",
  side.xlab = "Group", side.ylab = "Expression", side.log = FALSE,
  side.gridstep = ifelse(!transform || side.log, FALSE, 0.5), jitter = 30,
  side.main = "GeneID", display.columns = NULL, cols = c("#00bfff",
  "#858585", "#ff3030"), sample.cols = rep("#1f77b4", ncol(counts)),
  path = getwd(), folder = "glimma-plots", html = "MD-Plot",
  launch = TRUE, ...)
```

Arguments

`x`	the DESeqResults object.
`counts`	the matrix of expression values, with samples in columns.
`anno`	the data.frame containing gene annotations.
`groups`	the factor containing experimental groups of the samples.
`samples`	the names of the samples.
`status`	vector giving the control status of data point, of same length as the number of rows of object. If NULL, then all points are plotted in the default colour.
`transform`	TRUE if counts should be log-cpm transformed.
`main`	the title for the left plot.
`xlab`	label for x axis on left plot.
`ylab`	label for y axis on left plot.
`side.xlab`	label for x axis on right plot.
`side.ylab`	label for y axis on right plot.
`side.log`	TRUE to plot expression on the right plot on log scale.
`side.gridstep`	intervals along which to place grid lines on y axis. Currently only available for linear scale.
`jitter`	the amount of jitter to apply to the samples in the expressions plot.
`side.main`	the column containing mains for right plot.
`display.columns`	character vector containing names of columns to display in mouseover tooltips and table.
`cols`	vector of strings denoting colours corresponding to control status -1, 0 and 1. (may be R named colours or Hex values)

sample.cols	vector of strings denoting colours for each sample point on the expression plot.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
launch	TRUE to launch plot after call.
...	additional arguments to be passed onto the MD plot. (main, xlab, ylab can be set for the left plot)

Value

Draws a two-panel interactive MD plot in an html page. The left plot shows the log-fold-change vs average expression. The right plot shows the expression levels of a particular gene of each sample. Hovering over points on left plot will plot expression level for corresponding gene, clicking on points will fix the expression plot to gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot.

Author(s)

Shian Su

---

glMDPlot.DGEExact *Glimma MD Plot*

Description

Draw an interactive MD plot from a DGELRT objet

Usage

```
## S3 method for class 'DGEExact'
glMDPlot(x, counts = NULL, anno = NULL, groups = NULL,
  samples = NULL, status = rep(0, nrow(x)), transform = FALSE,
  main = "", xlab = "Average log CPM", ylab = "log-fold-change",
  side.xlab = "Group", side.ylab = "Expression", side.log = FALSE,
  side.gridstep = ifelse(!transform || side.log, FALSE, 0.5),
  p.adj.method = "BH", jitter = 30, side.main = "GeneID",
  display.columns = NULL, cols = c("#00bfff", "#858585", "#ff3030"),
  sample.cols = rep("#1f77b4", ncol(counts)), path = getwd(),
  folder = "glimma-plots", html = "MD-Plot", launch = TRUE, ...)
```

Arguments

x	the DGEExact object.
counts	the matrix of expression values, with samples in columns.
anno	the data.frame containing gene annotations.
groups	the factor containing experimental groups of the samples.
samples	the names of the samples.
status	vector giving the control status of data point, of same length as the number of rows of object. If NULL, then all points are plotted in the default colour.

transform	TRUE if counts should be log-cpm transformed.
main	the title for the left plot.
xlab	label for x axis on left plot.
ylab	label for y axis on left plot.
side.xlab	label for x axis on right plot.
side.ylab	label for y axis on right plot.
side.log	TRUE to plot expression on the right plot on log scale.
side.gridstep	intervals along which to place grid lines on y axis. Currently only available for linear scale.
p.adj.method	character vector indicating multiple testing correction method. See p.adjust for available methods. (defaults to "BH")
jitter	the amount of jitter to apply to the samples in the expressions plot.
side.main	the column containing mains for right plot.
display.columns	character vector containing names of columns to display in mouseover tooltips and table.
cols	vector of strings denoting colours corresponding to control status -1, 0 and 1. (may be R named colours or Hex values)
sample.cols	vector of strings denoting colours for each sample point on the expression plot.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
launch	TRUE to launch plot after call.
...	additional arguments to be passed onto the MD plot. (main, xlab, ylab can be set for the left plot)

Value

Draws a two-panel interactive MD plot in an html page. The left plot shows the log-fold-change vs average expression. The right plot shows the expression levels of a particular gene of each sample. Hovering over points on left plot will plot expression level for corresponding gene, clicking on points will fix the expression plot to gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot.

Author(s)

Shian Su

`glMDPlot.DGELRT` *Glimma MD Plot*

Description

Draw an interactive MD plot from a DGELRT object

Usage

```
## S3 method for class 'DGELRT'
glMDPlot(x, counts = NULL, anno = NULL, groups = NULL,
  samples = NULL, status = rep(0, nrow(x)), transform = FALSE,
  main = "", xlab = "Average log CPM", ylab = "log-fold-change",
  side.xlab = "Group", side.ylab = "Expression", side.log = FALSE,
  side.gridstep = ifelse(!transform || side.log, FALSE, 0.5),
  p.adj.method = "BH", jitter = 30, side.main = "GeneID",
  display.columns = NULL, cols = c("#00bfff", "#858585", "#ff3030"),
  sample.cols = rep("#1f77b4", ncol(counts)), path = getwd(),
  folder = "glimma-plots", html = "MD-Plot", launch = TRUE, ...)
```

Arguments

`x`	the DGELRT object.
`counts`	the matrix of expression values, with samples in columns.
`anno`	the data.frame containing gene annotations.
`groups`	the factor containing experimental groups of the samples.
`samples`	the names of the samples.
`status`	vector giving the control status of data point, of same length as the number of rows of object. If NULL, then all points are plotted in the default colour.
`transform`	TRUE if counts should be log-cpm transformed.
`main`	the title for the left plot.
`xlab`	label for x axis on left plot.
`ylab`	label for y axis on left plot.
`side.xlab`	label for x axis on right plot.
`side.ylab`	label for y axis on right plot.
`side.log`	TRUE to plot expression on the right plot on log scale.
`side.gridstep`	intervals along which to place grid lines on y axis. Currently only available for linear scale.
`p.adj.method`	character vector indicating multiple testing correction method. See `p.adjust` for available methods. (defaults to "BH")
`jitter`	the amount of jitter to apply to the samples in the expressions plot.
`side.main`	the column containing mains for right plot.
`display.columns`	character vector containing names of columns to display in mouseover tooltips and table.

cols	vector of strings denoting colours corresponding to control status -1, 0 and 1. (may be R named colours or Hex values)
sample.cols	vector of strings denoting colours for each sample point on the expression plot.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
launch	TRUE to launch plot after call.
...	additional arguments to be passed onto the MD plot. (main, xlab, ylab can be set for the left plot)

Value

Draws a two-panel interactive MD plot in an html page. The left plot shows the log-fold-change vs average expression. The right plot shows the expression levels of a particular gene of each sample. Hovering over points on left plot will plot expression level for corresponding gene, clicking on points will fix the expression plot to gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot.

Author(s)

Shian Su

---

glMDPlot.MArrayLM *Glimma MD Plot*

Description

Draw an interactive MD plot from a MArrayLM object

Usage

```
## S3 method for class 'MArrayLM'
glMDPlot(x, counts = NULL, anno = NULL, groups = NULL,
  samples = NULL, status = rep(0, nrow(x)), transform = FALSE,
  main = "", xlab = "Average log CPM", ylab = "log-fold-change",
  side.main = "GeneID", side.xlab = "Group", side.ylab = "Expression",
  side.log = FALSE, side.gridstep = ifelse(!transform || side.log, FALSE,
  0.5), coef = ncol(x$coefficients), p.adj.method = "BH", jitter = 30,
  display.columns = NULL, cols = c("#00bfff", "#858585", "#ff3030"),
  sample.cols = rep("#1f77b4", ncol(counts)), path = getwd(),
  folder = "glimma-plots", html = "MD-Plot", launch = TRUE, ...)
```

Arguments

x	the MArrayLM object.
counts	the matrix of expression values, with samples in columns.
anno	the data.frame containing gene annotations.
groups	the factor containing experimental groups of the samples.
samples	the names of the samples.

status	vector giving the control status of data point, of same length as the number of rows of object. If NULL, then all points are plotted in the default colour.
transform	TRUE if counts should be log-cpm transformed.
main	the title for the left plot.
xlab	label for x axis on left plot.
ylab	label for y axis on left plot.
side.main	the column containing mains for right plot.
side.xlab	label for x axis on right plot.
side.ylab	label for y axis on right plot.
side.log	TRUE to plot expression on the right plot on log scale.
side.gridstep	intervals along which to place grid lines on y axis. Currently only available for linear scale.
coef	integer or character index vector indicating which column of object to plot.
p.adj.method	character vector indicating multiple testing correction method. See p.adjust for available methods. (defaults to "BH")
jitter	the amount of jitter to apply to the samples in the expressions plot.
display.columns	character vector containing names of columns to display in mouseover tooltips and table.
cols	vector of strings denoting colours corresponding to control status -1, 0 and 1. (may be R named colours or Hex values)
sample.cols	vector of strings denoting colours for each sample point on the expression plot.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
launch	TRUE to launch plot after call.
...	additional arguments to be passed onto the MD plot. (main, xlab, ylab can be set for the left plot)

Value

Draws a two-panel interactive MD plot in an html page. The left plot shows the log-fold-change vs average expression. The right plot shows the expression levels of a particular gene of each sample. Hovering over points on left plot will plot expression level for corresponding gene, clicking on points will fix the expression plot to gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot.

Author(s)

Shian Su

Examples

```
library(limma)
library(edgeR)
```

```
data(lymphomaRNAseq)
x <- lymphomaRNAseq

sel <- rowSums(cpm(x$counts)>0.5)>=3
x <- x[sel,]

genotype <- relevel(x$samples$group, "Smchd1-null")
x <- calcNormFactors(x, method="TMM")
des <- model.matrix(~genotype)

## Apply voom with sample quality weights and fit linear model
v <- voomWithQualityWeights(x, design=des, normalization="none", plot=FALSE)
vfit <- lmFit(v,des)

## Apply treat relative to a fold-change of 1.5
vtfit <- treat(vfit,lfc=log2(1.5))
vfit <- eBayes(vfit)
results <- decideTests(vfit,p.value=0.01)

glMDPlot(vfit, counts=x$counts, anno=x$genes, groups=genotype, samples=1:7,
         status=results[,2], main="MD plot: Wild-type vs Smchd1",
         display.columns=c("Symbols", "GeneID", "GeneName"),
         folder="Smchd1-Lymphoma")
```

---

glMDRmd *glMDPlot Rmarkdown link and instructions*

Description

When run inside of a text-block of Rmarkdown document using 'r ...' this produces a link and instructions about the usage of the interactive plots.

Usage

```
glMDRmd(html = "MD-Plot")
```

Arguments

html	name of the HTML page containing plots from glMDPlot.

Value

None

See Also

glMDPlot

Examples

```
glMDRmd()
```

`glMDSPlot` *Glimma MDS Plot*

Description

Glimma MDS Plot

Draw an interactive MD plot from a DGEList object with distances calculated from most variable genes.

Usage

```
glMDSPlot(x, ...)
```

Arguments

`x`	the matrix containing the gene expressions.
`...`	additional arguments.

Value

Draws a two-panel interactive MDS plot in an html page. The left panel contains the plot between two MDS dimensions, with annotations displayed on hover. The right panel contains a bar plot of the eigenvalues of each dimension, clicking on any of the bars will plot the corresponding dimension against the next dimension.

Author(s)

Shian Su, Gordon Smyth

See Also

`glMDSPlot.default`, `glMDSPlot.DGEList`

Examples

```
data(lymphomaRNAseq)
genotype <- relevel(lymphomaRNAseq$samples$group, "Smchd1-null")

glMDSPlot(lymphomaRNAseq, labels=1:7, groups=genotype)
```

`glMDSPlot.default` *Glimma MDS Plot*

Description

Glimma MDS Plot

Draw an interactive MD plot from a DGEList object with distances calculated from most variable genes.

Usage

```
## Default S3 method:
glMDSPlot(x, top = 500, labels = seq_cols(x),
  groups = rep(1, ncol(x)), gene.selection = c("pairwise", "common"),
  main = "MDS Plot", path = getwd(), folder = "glimma-plots",
  html = "MDS-Plot", launch = TRUE, ...)
```

Arguments

`x`	the matrix containing the gene expressions.
`top`	the number of top most variable genes to use.
`labels`	the labels for each sample.
`groups`	the experimental group to which samples belong.
`gene.selection`	"pairwise" if most variable genes are to be chosen for each pair of samples or "common" to select the same genes for all comparisons.
`main`	the title of the plot.
`path`	the path in which the folder will be created.
`folder`	the name of the fold to save html file to.
`html`	the name of the html file to save plots to.
`launch`	TRUE to launch plot after call.
`...`	additional arguments.

Value

Draws a two-panel interactive MDS plot in an html page. The left panel contains the plot between two MDS dimensions, with annotations displayed on hover. The right panel contains a bar plot of the eigenvalues of each dimension, clicking on any of the bars will plot the corresponding dimension against the next dimension.

Author(s)

Shian Su, Gordon Smyth

`glMDSPlot.DESeqDataSet` *Glimma MDS Plot*

Description

Glimma MDS Plot

Draw an interactive MD plot from a DGEList object with distances calculated from most variable genes.

Usage

```
## S3 method for class 'DESeqDataSet'
glMDSPlot(x, top = 500, labels = NULL,
  groups = NULL, gene.selection = c("pairwise", "common"),
  prior.count = 0.25, main = "MDS Plot", path = getwd(),
  folder = "glimma-plots", html = "MDS-Plot", launch = TRUE, ...)
```

Arguments

`x`	the DESeqDataSet containing the gene expressions.
`top`	the number of top most variable genes to use.
`labels`	the labels for each sample.
`groups`	the experimental group to which samples belong.
`gene.selection`	"pairwise" if most variable genes are to be chosen for each pair of samples or "common" to select the same genes for all comparisons.
`prior.count`	average count to be added to each observation to avoid taking log of zero. Used only if log=TRUE.
`main`	the title of the plot.
`path`	the path in which the folder will be created.
`folder`	the name of the fold to save html file to.
`html`	the name of the html file to save plots to.
`launch`	TRUE to launch plot after call.
`...`	additional arguments.

Value

Draws a two-panel interactive MDS plot in an html page. The left panel contains the plot between two MDS dimensions, with annotations displayed on hover. The right panel contains a bar plot of the eigenvalues of each dimension, clicking on any of the bars will plot the corresponding dimension against the next dimension.

Author(s)

Shian Su, Gordon Smyth

`glMDSPlot.DGEList` *Glimma MDS Plot*

Description

Glimma MDS Plot

Draw an interactive MD plot from a DGEList object with distances calculated from most variable genes.

Usage

```
## S3 method for class 'DGEList'
glMDSPlot(x, top = 500, labels = NULL, groups = rep(1,
  ncol(x)), gene.selection = c("pairwise", "common"), prior.count = 0.25,
  main = "MDS Plot", path = getwd(), folder = "glimma-plots",
  html = "MDS-Plot", launch = TRUE, ...)
```

Arguments

`x`	the DGEList containing the gene expressions.
`top`	the number of top most variable genes to use.
`labels`	the labels for each sample.
`groups`	the experimental group to which samples belong.
`gene.selection`	"pairwise" if most variable genes are to be chosen for each pair of samples or "common" to select the same genes for all comparisons.
`prior.count`	average count to be added to each observation to avoid taking log of zero. Used only if log=TRUE.
`main`	the title of the plot.
`path`	the path in which the folder will be created.
`folder`	the name of the fold to save html file to.
`html`	the name of the html file to save plots to.
`launch`	TRUE to launch plot after call.
`...`	additional arguments.

Value

Draws a two-panel interactive MDS plot in an html page. The left panel contains the plot between two MDS dimensions, with annotations displayed on hover. The right panel contains a bar plot of the eigenvalues of each dimension, clicking on any of the bars will plot the corresponding dimension against the next dimension.

Author(s)

Shian Su, Gordon Smyth

glScatter *Glimma Scatter Plot*

Description

Create an interactive scatter plot object

Usage

```
glScatter(x, ...)
```

Arguments

x	the data.frame containing data to plot.
...	additional arguments depending on input object type.

Value

A chart object containing the information to create an interactive scatter plot.

Author(s)

Shian Su

Examples

```
data(iris)

plot1 <- glScatter(iris, xval="Sepal.Length", yval="Sepal.Width", colval="Species")
glimma(plot1, c(1,1))
```

glScatter.default *Glimma Scatter Plot*

Description

Default method for creating an interactive scatter plot

Usage

```
## Default S3 method:
glScatter(x, xval = "x", yval = "y", idval = NULL,
  point.size = 2, x.jitter = 0, y.jitter = 0, ndigits = NULL,
  signif = 6, log = "", xgrid = FALSE, ygrid = FALSE, xstep = FALSE,
  ystep = FALSE, xlab = xval, ylab = yval, main = NULL, height = 400,
  width = 500, colval = NULL, annot = c(xval, yval), annot.lab = NULL,
  flag = NULL, info = NULL, hide = FALSE, disable = NULL, ...)
```

Arguments

x	the data.frame containing data to plot.
xval	the column name for the x-axis values.
yval	the column name for the y-axis values.
idval	the column name for unique identifiers.
point.size	the size of the data points.
x.jitter	the amount of jittering to add to values along the x axis.
y.jitter	the amount of jittering to add to values along the y axis.
ndigits	the number of digits after the decimal to round to in the tooltip (overrides signif).
signif	the number of significant figures to display in the tooltip.
log	a character string which contains "x" if the x axis is to be logarithmic, "y" if the y axis is to be logarithmic and "xy" or "yx" if both axes are to be logarithmic.
xgrid	TRUE if grid lines should be placed along x axis.
ygrid	TRUE if grid lines should be placed y axis.
xstep	the interval at which to set grid lines along the x axis.
ystep	the interval at which to set grid lines along the y axis.
xlab	the label on the x-axis.
ylab	the label on the y-axis.
main	the title for the plot.
height	the height of the plot (in pixels).
width	the width of the plot (in pixels).
colval	the colours for each data point.
annot	the columns to display in the tooltip.
annot.lab	alternative labels for the values displayed in the tooltip.
flag	the special flag to indicate special plot.
info	additional information for plotting.
hide	TRUE to hide the plot when page starts.
disable	the events to disable, options are "click", "hover", "zoom".
...	additional arguments.

Value

A chart object containing the information to create an interactive scatter plot.

Author(s)

Shian Su

Examples

```
data(iris)

plot1 <- glScatter(iris, xval="Sepal.Length", yval="Sepal.Width", colval="Species")
glimma(plot1, c(1,1))
```

glTable *Glimma Table*

Description

Create a table using the data from a chart.

Usage

```
glTable(target, columns)
```

Arguments

target	the index of the plot from which data is drawn.
columns	the columns of data to plot.

Value

a input object containing the input field information.

gltablink *Plot linkages*

Description

Helper function for writing the link properties in interactive Glimma plots

Usage

```
gltablink(from, to, action = "none", info = "none")
```

Arguments

from	the index of the source table.
to	the index of the plot which receives the event and performs an action.
action	the action that is performed in the plot.
info	additional info for creating the link.

Value

a link object containing the plot linking information.

glXYPlot *Glimma XY Plot*

Description

Draw an interactive XY plot with multiple panels

Usage

```
glXYPlot(x, y, counts = NULL, groups = NULL, samples = NULL,
  status = rep(0, nrow(data)), anno = NULL, display.columns = NULL,
  xlab = "x", ylab = "y", side.main = "GeneID", side.xlab = "Group",
  side.ylab = "Expression", sample.cols = rep("#1f77b4", length(groups)),
  cols = c("#00bfff", "#858585", "#ff3030"), jitter = 30, path = getwd(),
  folder = "glimma-plots", html = "XY-Plot", launch = TRUE, ...)
```

Arguments

x	a numeric vector of values to plot on the x-axis of the summary plot.
y	a numeric vector of values to plot on the y-axis of the summary plot.
counts	the matrix containing all counts, the column order should correspond to the order of the x and y vectors.
groups	the factor containing experimental groups of the samples.
samples	the names of the samples.
status	vector giving the control status of data point, of same length as the number of rows of object. If NULL, then all points are plotted in the default colour
anno	the data.frame containing gene annotations.
display.columns	character vector containing names of columns to display in mouseover tooltips and table.
xlab	the label on the x axis for the left plot.
ylab	the label on the y axis for the left plot.
side.main	the column containing mains for right plot.
side.xlab	the label on the x axis for the right plot.
side.ylab	the label on the y axis for the right plot.
sample.cols	vector of strings denoting colours for each sample point on the expression plot.
cols	vector of strings denoting colours corresponding to control status -1, 0 and 1. (may be R named colours or Hex values)
jitter	the amount of jitter to apply to the samples in the expressions plot.
path	the path in which the folder will be created.
folder	the name of the fold to save html file to.
html	the name of the html file to save plots to.
launch	TRUE to launch plot after call.
...	additional arguments to be passed onto the MD plot. (main, etc. can be set for the left plot)

Value

Draws a two-panel interactive XY scatter plot in an html page. The left plot shows the x and y values specified. The right plot shows the expression levels of a particular gene in each sample. Hovering over points on left plot will plot expression level for the corresponding gene, clicking on points will fix the expression plot to that gene. Clicking on rows on the table has the same effect as clicking on the corresponding gene in the plot. This function generates a display that is similar in style to glMDPlot, except that it provides more flexibility in what the user can provide.

Author(s)

Charity Law and Shian Su

Examples

```
data(iris)

glXYPlot(iris$Sepal.Width, iris$Sepal.Length,
         xlab="Sepal.Width", ylab="Sepal.Length", side.main="PlantID")
```

---

is.hex *Hexcode colours*

Description

Check if string(s) are valid hex colour representation

Usage

```
is.hex(x)
```

Arguments

x	the colour value(s) to check.

Value

Logical vector indicating if strings(s) are valid hex representations

lymphomaRNAseq *Mouse based RNAseq data for study of smchd1 gene.*

Description

Mouse based RNAseq data for study of smchd1 gene.

Author(s)

Ruijie Liu, Kelan Chen, Natasha Jansz, Marnie E. Blewitt, Matthew E. Ritchie

References

http://www.sciencedirect.com/science/article/pii/S2213596015301306

makeJson *JSON converter for R objects*

Description

Function to generate json strings from

Usage

```
makeJson(x, ...)
```

Arguments

x	the object to be converted into JSON
...	additional arguments

Value

a stringified JSON object.

makeJson.data.frame *JSON converter for data frames*

Description

Function to create a JSON from a data.frame

Usage

```
## S3 method for class 'data.frame'
makeJson(df, convert.logical = TRUE,
  dataframe = c("rows", "columns"))
```

Arguments

df	the data.frame to be converted into JSON
convert.logical	whether to convert logicals into strings "TRUE" and "FALSE"
dataframe	how to encode data.frame objects: must be one of 'rows', 'columns'

Value

a stringified JSON, the data.frame is encoded as a vector of objects, with each column being one object with keys corresponding to column names.

makeJson.jschart *JSON converter for chart objects*

Description

Function to make json object from a chart, ignoring the json property

Usage

```
## S3 method for class 'jschart'
makeJson(chart)
```

Arguments

chart	the chart object to be converted into JSON

Value

a stringified JSON object containing the chart data.

Index