Package ‘Glimma’

May 29, 2024

Type Package

Title Interactive visualizations for gene expression analysis

Version 2.14.0

Description This package produces interactive visualizations for RNA-seq data analysis, utilizing output from limma, edgeR, or DESeq2. It produces interactive htmlwidgets versions of popular RNA-seq analysis plots to enhance the exploration of analysis results by overlaying interactive features. The plots can be viewed in a web browser or embedded in notebook documents.

Encoding UTF-8

RoxygenNote 7.1.1

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

Depends R (>= 4.0.0)

Imports htmlwidgets, edgeR, DESeq2, limma, SummarizedExperiment, stats, jsonlite, methods, S4Vectors

Suggests testthat, knitr, rmarkdown, BiocStyle, IRanges, GenomicRanges, pryr, AnnotationHub, scRNAseq, scater, scran

License GPL-3

URL https://github.com/hasaru-k/GlimmaV2

BugReports https://github.com/hasaru-k/GlimmaV2/issues

VignetteBuilder knitr

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

git_branch RELEASE_3_19

git_last_commit 498f03d

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-29

Author Shian Su [aut, cre], Hasaru Kariyawasam [aut], Oliver Voogd [aut],
Contents

arraydata ............................... 3
as.hexcol ................................ 3
buildXYData .............................. 4
extractGroups ............................ 5
glBar .................................... 6
glBar.default ............................ 6
glimma .................................... 8
glimmaMA .................................. 8
glimmaMA.DESeqDataSet .................... 9
glimmaMA.DGEExact ......................... 12
glimmaMA.DGELRT .......................... 14
glimmaMA.MArrayLM ....................... 16
glimmaMDS ................................. 19
glimmaMDS.default ........................ 20
glimmaMDS.DESeqDataSet ................. 22
glimmaMDS.DGEList ....................... 24
glimmaVolcano ............................ 25
glimmaVolcano.DESeqDataSet ............. 27
glimmaVolcano.DGEExact ................. 29
glimmaVolcano.DGELRT ................. 31
glimmaVolcano.MArrayLM ............. 34
glimmaXY ................................. 36
glimmaXYWidget .......................... 38
glimma_plot ................................ 39
glink ..................................... 39
glMDPlot .................................. 40
glMDPlot.default ........................ 41
glMDPlot.DESeqDataSet ................. 43
glMDPlot.DESeqResults ................... 45
glMDPlot.DGEExact ....................... 47
glMDPlot.DGELRT ......................... 49
glMDPlot.MArrayLM ....................... 51
glM DRmd .................................. 53
glMDSPlot ................................ 53
glMDSPlot.default ......................... 54
glMDSPlot.DESeqDataSet ................... 55
glMDSPlot.DGEList ....................... 56
glScatter ................................ 58
glScatter.default ......................... 58
**arraydata**

Example microarray for the study of Ezh2.

**Description**

Example microarray for the study of Ezh2.

**Author(s)**


**References**

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

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

**Arguments**

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

**Value**

hex codes for colours
buildXYData  

**XY Data Object Builder**

**Description**

Common processing steps for both MA, XY and volcano plots. Expects a dataframe, `table`, which contains two columns labelled `xlab` and `ylab` as well as a unique identifier column labelled `gene`.

**Usage**

```r
buildXYData(
  table, status, main, display.columns, anno, counts, xlab, ylab, status.cols, sample.cols, groups, transform.counts
)
```

**Arguments**

- `table`: dataframe containing `xlab` and `ylab` columns for plotting.
- `status`: vector of length `nrow(x)` indicating the status of each gene. By default, genes in the summary plot are coloured based on their differential expression status using an adjusted p-value cutoff of 5% by calling the `limma::decideTests` function, where the value of -1 marks down-regulated genes, 0 marks genes with no expression difference, and 1 marks up-regulated genes.
- `main`: character string for the main title of summary plot.
- `display.columns`: character vector containing names of columns from `anno` from which to display in mouseover tooltips and table.
- `anno`: dataframe with `nrow(x)` rows containing gene annotations.
- `counts`: numeric matrix with `nrow(x)` rows containing gene expression values. This can be used to replace the gene counts from `dge$counts`, i.e. you may have log-rpkm values stored in a different object that you wish to use.
- `xlab`: character string for the x-axis label of summary plot.
- `ylab`: character string for the y-axis label of summary plot.
- `status.cols`: vector of length 3 containing valid CSS strings for colours associated with `status` in the order of -1, 0 and 1.
**sample.cols** character vector of length ncol(counts) containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to groups.

**groups** vector of length ncol(dge) representing categorisation of samples in expression plot.

**transform.counts** the type of transformation used on the counts - "logcpm" for using edgeR::cpm(counts, log=TRUE); "cpm" for edgeR::cpm(counts); "rpkm" for edgeR::rpkm(counts); "logrpkm" for edgeR::rpkm(counts, log=TRUE); and "none" for no transformation). Defaults to "logcpm".

**Value**

object for XY plot internal use

---

**extractGroups**

**extractGroups**

**Description**

Extracts the column named group from column data matrix of a SummarizedExperiment object if it is present. Otherwise return a vector of 1s.

**Usage**

extractGroups(cdata)

**Arguments**

 **cdata** SummarizedExperiment column data matrix

**Value**

groups column of data if present, otherwise 1
**glBar**

*Glimma MD Plot*

**Description**

Create an interactive bar plot object.

**Usage**

```r
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`

---

**glBar.default**

*Glimma Bar Plot*

**Description**

Default method for interactive bar plot.

**Usage**

```r
## 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
glimmaMA

Description
The Glimma package provides interactive versions of plots frequently used in the limma package. Currently the MDS and MD plots have been implemented. The functions can be used with both limma, edgeR and DESeq objects.

Main functions
glMDSPlot, glMDPlot, glXYPlot

Description
Generic function for drawing a two-panel interactive MA plot, a special case of the glimmaXY plot. The function invokes the following methods which depend on the class of the first argument:

- `glimmaMA.MArrayLM` for limma analysis
- `glimmaMA.DGEEexact` for edgeR analysis, produced from `exactTest`
- `glimmaMA.DGELRT` for edgeR analysis, produced from `glmLRT`
- `glimmaMA.DESeqDataSet` for DESeq2 analysis

glimmaMD is an alias for glimmaMA.

Usage
glimmaMA(x, ...)
glimmaMD(x, ...)

Arguments

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

The summary plot on the left represents gene-wise log-fold-change (logFC) on the y-axis versus average gene expression calculated as log-counts-per-million (logCPM) values. We call our summary plot an MA plot because this type of plot was originally referred to as an MA plot in the limma package, with the M-value representing logFC and A-value representing average expression - it has since been renamed to MD plot in the limma package. The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Vogd

Examples

methods(glimmaMA) # show methods for glimmaMA

glimmaMA.DESeqDataSet  Glimma MA Plot

Description

Draws a two-panel interactive MA plot from an DESeqDataSet object. This is a special case of the glimmaXY plot.

Usage

## S3 method for class 'DESeqDataSet'
glimmaMA(
  x,
  counts = DESeq2::counts(x),
  groups = extractGroups(colData(x)),
  status = NULL,
  anno = NULL,
  display.columns = NULL,
  status.cols = c("#1052bd", "silver", "#cc212f"),
sample.cols = NULL,
transform.counts = c("logcpm", "cpm", "rpkm", "logrpkm", "none"),
main = "MA Plot",
xlab = "logCPM",
ylab = "logFC",
html = NULL,
width = 920,
height = 920,
...
}

Arguments

x  DESeqDataSet object from which summary statistics are extracted from to create summary (left) plot.

counts  numeric matrix with nrow(x) rows containing gene expression values.

groups  vector/factor representing the experimental group for each sample; see extractGroups for default value.

status  vector of length nrow(x) indicating the status of each gene.

anno  dataframe with nrow(x) rows containing gene annotations.

display.columns  character vector containing names of columns from anno from which to display in mouseover tooltips and table.

status.cols  vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.

sample.cols  character vector of length ncol(counts) containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to groups.

transform.counts  the type of transformation used on the counts - "logcpm" for using edgeR::cpm(counts, log=TRUE); "cpm" for edgeR::cpm(counts); "rpkm" for edgeR::rpkm(counts); "logrpkm" for edgeR::rpkm(counts, log=TRUE); and "none" for no transformation). Defaults to "logcpm".

main  character string for the main title of summary plot.

xlab  character string for the x-axis label of summary plot.

ylab  character string for the y-axis label of summary plot.

html  character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

width  numeric value indicating width of widget in pixels.

height  numeric value indicating width of height in pixels.

additional unused arguments.
Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the y-axis versus average gene expression calculated as log-counts-per-million (logCPM) values. We call our summary plot an MA plot because this type of plot was originally referred to as an MA plot in the limma package, with the M-value representing logFC and A-value representing average expression - it has since been renamed to MD plot in the limma package. The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also

glimmaMA, glimmaMA.MArrayLM, glimmaMA.DGExact, glimmaMA.DGELRT

Examples

dge <- readRDS(
  system.file("RNAsq123/dge.rds", package = "Glimma")
)

dds <- DESeq2::DESeqDataSetFromMatrix(
  countData = dge$counts,
  colData = dge$samples,
  rowData = dge$genes,
  design = ~group
)

dds <- DESeq2::DESeq(dds, quiet=TRUE)
glimmaMA(dds)
glimmaMA.DGEExact  Glimma MA Plot

Description

Draws a two-panel interactive MA plot from an DGEExact object. This is a special case of the glimmaXY plot.

Usage

```r
## S3 method for class 'DGEExact'
glimmaMA(
x,
dge = NULL,
counts = dge$counts,
groups = dge$samples$group,
status = edgeR::decideTestsDGE(x),
anno = x$genes,
display.columns = NULL,
status.cols = c("#1052bd", "silver", "#cc212f"),
sample.cols = NULL,
p.adj.method = "BH",
transform.counts = c("logcpm", "cpm", "rpkm", "logrpkm", "none"),
main = paste(x$comparison[2], "vs", x$comparison[1]),
xlab = "logCPM",
ylab = "logFC",
html = NULL,
width = 920,
height = 920,
...)
```

Arguments

- `x`: DGEExact object from which summary statistics are extracted from to create summary (left) plot.
- `dge`: DGEList object with nrow(x) rows from which expression values are extracted from to create expression (right) plot. Gene counts are taken from dge$counts and sample groups from dge$samples$group. By default raw counts are transformed to log-cpm values (see more in the transform.counts argument).
- `counts`: numeric matrix with nrow(x) rows containing gene expression values. This can be used to replace the gene counts from dge$counts, i.e. you may have log-rpkm values stored in a different object that you wish to use.
- `groups`: vector of length ncol(dge) representing categorisation of samples in expression plot.

## Additional information

- `status`: vector of length ncol(dge) representing categorisation of samples in expression plot.
- `anno`: vector of genes to annotate on the plot.
- `display.columns`: vector of column names to display on the plot.
- `status.cols`: vector of colors for status categories.
- `sample.cols`: vector of colors for sample groups.
- `p.adj.method`: method for adjusting p-values to control the false discovery rate.
- `transform.counts`: vector of transformations to apply to gene expression values.
- `main`: title for the plot.
- `xlab`: title for the x-axis.
- `ylab`: title for the y-axis.
- `html`: HTML code for interactive plot.
- `width`, `height`: dimensions of the plot.
status vector of length nrow(x) indicating the status of each gene. By default genes in the summary plot are coloured based on its differential expression status using an adjusted p-value cutoff of 0.05 by calling the edgeR::decideTestsDGE() function, where the value of -1 marks down-regulated genes, 0 marks genes with no expression difference, and 1 marks up-regulated genes.

anno dataframe with nrow(x) rows containing gene annotations.

display.columns character vector containing names of columns from anno from which to display in mouseover tooltips and table.

status.cols vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.

sample.cols character vector of length ncol(counts) containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to groups.

p.adj.method character string specifying p-value adjustment method.

transform.counts the type of transformation used on the counts - "logcpm" for using edgeR::cpm(counts, log=TRUE); "cpm" for edgeR::cpm(counts); "rpkm" for edgeR::rpkm(counts); "logrpkm" for edgeR::rpkm(counts, log=TRUE); and "none" for no transformation). Defaults to "logcpm".

main character string for the main title of summary plot.

xlab character string for the x-axis label of summary plot.

ylab character string for the y-axis label of summary plot.

html character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

width numeric value indicating width of widget in pixels.

height numeric value indicating width of height in pixels.

Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the y-axis versus average gene expression calculated as log-counts-per-million (logCPM) values. We call our summary plot an MA plot because this type of plot was originally referred to as an MA plot in the limma package, with the M-value representing logFC and A-value representing average expression - it has since been renamed to MD plot in the limma package. The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.
Value
htmlwidget object or NULL if html argument is specified.

Author(s)
Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also
glimmaMA, glimmaMA.MArrayLM, glimmaMA.DGELRT, glimmaMA.DESeqDataSet

Examples
dge <- readRDS(
  system.file("RNAseq123/dge.rds", package = "Glimma"))
design <- readRDS(
  system.file("RNAseq123/design.rds", package = "Glimma"))
contr.matrix <- readRDS(
  system.file("RNAseq123/contr.matrix.rds", package = "Glimma"))

dge <- edgeR::estimateDisp(dge, design)
gfit <- edgeR::glmFit(dge, design)
glrt <- edgeR::glmLRT(gfit, design, contrast = contr.matrix)

glimmaMA(glrt, dge = dge)

glimmaMA.DGELRT

Glimma MA Plot

Description
Draws a two-panel interactive MA plot from a DGELRT object. This is a special case of the
glimmaXY plot.

Usage
### S3 method for class 'DGELRT'
glimmaMA(
  x,
  dge = NULL,
  counts = dge$counts,
  groups = dge$samples$group,
  status = edgeR::decideTestsDGE(x),
  anno = x$genes,
  display.columns = NULL,
  status.cols = c("#1052bd", "silver", "#cc212f"),
  sample.cols = NULL,
p.adj.method = "BH",
transform.counts = c("logcpm", "cpm", "rpkm", "logrpkm", "none"),
main = paste(x$comparison[2], "vs", x$comparison[1]),
xlab = "logCPM",
ylab = "logFC",
html = NULL,
width = 920,
height = 920,

Arguments

x  
DGELRT object from which summary statistics are extracted from to create summary (left) plot.
dge  
DGEList object with nrow(x) rows from which expression values are extracted from to create expression (right) plot. Gene counts are taken from dge$counts and sample groups from dge$samples$group. By default raw counts are transformed to log-cpm values (see more in the transform.counts argument).
counts  
numeric matrix with nrow(x) rows containing gene expression values. This can be used to replace the gene counts from dge$counts, i.e. you may have log-rpkm values stored in a different object that you wish to use.
groups  
vector of length ncol(dge) representing categorisation of samples in expression plot.
status  
vector of length nrow(x) indicating the status of each gene. By default genes in the summary plot are coloured based on its differential expression status using an adjusted p-value cutoff of 0.05 by calling the edgeR::decideTestsDGE() function, where the value of -1 marks down-regulated genes, 0 marks genes with no expression difference, and 1 marks up-regulated genes.
anno  
dataframe with nrow(x) rows containing gene annotations.
display.columns  
character vector containing names of columns from anno from which to display in mouseover tool tips and table.
status.cols  
vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.
sample.cols  
character vector of length ncol(counts) containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to groups.
p.adj.method  
character string specifying p-value adjustment method.
transform.counts  
the type of transformation used on the counts - "logcpm" for using edgeR::cpm(counts, log=TRUE); "cpm" for edgeR::cpm(counts); "rpkm" for edgeR::rpkm(counts); "logrpkm" for edgeR::rpkm(counts, log=TRUE); and "none" for no transformation). Defaults to "logcpm".
main  
character string for the main title of summary plot.
glimmaMA.MArrayLM

xlab character string for the x-axis label of summary plot.
ylab character string for the y-axis label of summary plot.
html character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. “file.html”.
width numeric value indicating width of widget in pixels.
height numeric value indicating width of height in pixels.
... additional unused arguments.

Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the y-axis versus average gene expression calculated as log-counts-per-million (logCPM) values. We call our summary plot an MA plot because this type of plot was originally referred to as an MA plot in the limma package, with the M-value representing logFC and A-value representing average expression - it has since been renamed to MD plot in the limma package. The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also

glimmaMA, glimpseMA.MArrayLM, glimpseMA.DGEEst, glimpseMA.DESeqDataSet

glimmaMA.MArrayLM  Glimma MA Plot

Description

Draws a two-panel interactive MA plot from an MArrayLM object. This is a special case of the glimpsaXY plot.
Usage

```r
## S3 method for class 'MArrayLM'
glimmaMA(
x,  
dge = NULL,
  counts = dge$counts,
  groups = dge$samples$group,
  coef = ncol(x$coefficients),
  status = limma::decideTests(x),
  anno = x$genes,
  display.columns = NULL,
  status.cols = c("#1052bd", "silver", "#cc212f"),
  sample.cols = NULL,
  p.adj.method = "BH",
  transform.counts = c("logcpm", "cpm", "rpkm", "logrpkm", "none"),
  main = colnames(x)[coef],
  xlab = "logCPM",
  ylab = "logFC",
  html = NULL,
  width = 920,
  height = 920,
  ...)
```

Arguments

- **x** `MArrayLM` object from which summary statistics are extracted from to create summary (left) plot.
- **dge** `DGEList` object with `nrow(x)` rows from which expression values are extracted from to create expression (right) plot. Gene counts are taken from `dge$counts` and sample groups from `dge$samples$group`. By default raw counts are transformed to log-cpm values (see more in the `transform.counts` argument).
- **counts** numeric matrix with `nrow(x)` rows containing gene expression values. This can be used to replace the gene counts from `dge$counts`, i.e. you may have log-rpkm values stored in a different object that you wish to use.
- **groups** vector of length `ncol(dge)` representing categorisation of samples in expression plot.
- **coef** integer indicating the column in `x` from the summary plot is created.
- **status** vector of length `nrow(x)` indicating the status of each gene. By default genes in the summary plot are coloured based on its differential expression status using an adjusted p-value cutoff of 5% by calling the `limma::decideTests` function, where the value of -1 marks down-regulated genes, 0 marks genes with no expression difference, and 1 marks up-regulated genes.
- **anno** dataframe with `nrow(x)` rows containing gene annotations.
- **display.columns** character vector containing names of columns from `anno` from which to display in mouseover tooltips and table.
status.cols vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.

sample.cols character vector of length ncol(counts) containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to groups.

p.adj.method character string specifying p-value adjustment method.

transform.counts the type of transformation used on the counts - "logcpm" for using edgeR::cpm(counts, log=TRUE); "cpm" for edgeR::cpm(counts); "rpkm" for edgeR::rpkm(counts); "logrpkm" for edgeR::rpkm(counts, log=TRUE); and "none" for no transformation). Defaults to "logcpm".

main character string for the main title of summary plot.
xlab character string for the x-axis label of summary plot.

ylab character string for the y-axis label of summary plot.

html character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

width numeric value indicating width of widget in pixels.

height numeric value indicating width of height in pixels.

... additional unused arguments.

Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the y-axis versus average gene expression calculated as log-counts-per-million (logCPM) values. We call our summary plot an MA plot because this type of plot was originally referred to as an MA plot in the limma package, with the M-value representing logFC and A-value representing average expression - it has since been renamed to MD plot in the limma package. The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also

glimmaMA,glimmaMA.DGEEexact,glimmaMA.DGELRT,glimmaMA.DESeqDataSet
Examples

dge <- readRDS(
  system.file("RNAseq123/dge.rds", package = "Glimma"))
design <- readRDS(
  system.file("RNAseq123/design.rds", package = "Glimma"))
contr.matrix <- readRDS(
  system.file("RNAseq123/contr.matrix.rds", package = "Glimma"))

v <- limma::voom(dge, design)
vfit <- limma::lmFit(v, design)
vfit <- limma::contrasts.fit(vfit, contrasts = contr.matrix)
efit <- limma::eBayes(vfit)

glimmaMA(efit, dge = dge)

Description

Generic function for drawing a two-panel interactive multidimensional scaling (MDS) plot. The
function invokes the following methods which depend on the class of the first argument:

- glimmaMDS.DGEList for edgeR analysis
- glimmaMDS.DESeqDataSet for DESeq2 analysis
- glimmaMDS.default for all other object types

Usage

glimmaMDS(x, ...)

Arguments

x the matrix containing the gene expressions.
...
the additional arguments affecting the plot produced. See specific methods for
detailed arguments.

Details

The left plot shows two MDS dimensions, with sample annotations displayed on hover. The right
panel contains a bar plot of the eigenvalues of each dimension. The controls beneath the plots can be
used to change the dimensions being displayed, and the scale, colour and shape of points. The inter-
active MDS plot allows users to adjust sample points by scale, colour and shape for multiple vectors
associated with sample information. This is carried out most effectively when x$samples includes
an abundance of sample information, or when a data frame object is supplied to groups. If a sim-
ple character or factor vector is given to groups (with the default of continuous.colour=FALSE),
then sample points will have no scaling options, but can only be adjusted in colour and shape
by groups and labels. Instead, if groups is a numeric vector (e.g. library size or expression level of a specific gene), then the plot can be scaled and coloured by the numeric values with continuous.colour=TRUE. For more details, refer to limma::plotMDS.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

Examples

dge <- readRDS(system.file("RNAseq123/dge.rds", package = "Glimma"))
glimmaMDS(dge)

# using DESeqDataSet
dds <- DESeq2::DESeqDataSetFromMatrix(
  countData = dge$counts,
  colData = dge$samples,
  rowData = dge$genes,
  design = ~group
)
glimmaMDS(dds)

# using matrix object
expr <- edgeR::cpm(dge, log = TRUE)
glimmaMDS(expr)

glimmaMDS.default

Description

Draws a two-panel interactive MDS plot.

Usage

## Default S3 method:
glimmaMDS(
  x,
  groups = as.character(rep(1, ncol(x))),
  labels = as.character(seq_len(ncol(x))),
  continuous.colour = FALSE,
  top = 500,
  gene.selection = c("pairwise", "common"),
html = NULL,
width = 900,
height = 500,

Arguments

x
the matrix containing the gene expressions.
groups
vector or data frame object with associated sample information such as experimental groups. The information is displayed in mouseover tooltips, and appropriate vector(s) can be used to adjust the plot using scale_by, colour_by and shape_by drop-down boxes of the widget.
labels
character vector of sample names or labels.
continuous.colour
TRUE if continuous colour schemes should be used. Defaults to FALSE where distinct colour schemes are used.
top
integer indicating number of top genes used to calculate pairwise distances.
gene.selection
character string specifying how genes are selected from the plot - "pairwise" if most variable genes are to be chosen for each pair of samples, or "common" to select the same genes for all comparisons.
html
character string for naming HTML file or exportation of widget. The extension should be included in the file name e.g. "file.html".
width
numeric value indicating width of widget in pixels.
height
numeric value indicating width of widget in pixels.

Details

The left plot shows two MDS dimensions, with sample annotations displayed on hover. The right panel contains a bar plot of the eigenvalues of each dimension. The controls beneath the plots can be used to change the dimensions being displayed, and the scale, colour and shape of points. The interactive MDS plot allows users to adjust sample points by scale, colour and shape for multiple vectors associated with sample information. This is carried out most effectively when x$samples includes an abundance of sample information, or when a data frame object is supplied to groups. If a simple character or factor vector is given to groups (with the default of continuous.colour=FALSE), then sample points will have no scaling options, but can only be adjusted in colour and shape by groups and labels. Instead, if groups is a numeric vector (e.g. library size or expression level of a specific gene), then the plot can be scaled and coloured by the numeric values with continuous.colour=TRUE. For more details, refer to limma::plotMDS.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd
See Also
glimmaMDS, glimmaMDS.DGEList, glimmaMDS.DESeqDataSet

Examples

dge <- readRDS(system.file("RNAseq123/dge.rds", package = "Glimma"))
expr <- edgeR::cpm(dge, log = TRUE)
glimmaMDS(expr)

glimmaMDS.DESeqDataSet

Glimma MDS Plot

Description

Draws a two-panel interactive MDS plot using a DESeqDataset x. Transforms counts using edgeR::cpm(DESeq2::counts(x), log = TRUE, prior.count = prior.count).

Usage

## S3 method for class 'DESeqData'glimmaMDS(
x, 
groups = as.data.frame(SummarizedExperiment::colData(x)), 
labels = rownames(SummarizedExperiment::colData(x)), 
continuous.colour = FALSE, 
top = 500, 
gene.selection = c("pairwise", "common"), 
prior.count = 2, 
html = NULL, 
width = 900, 
height = 500, 
... 
)

Arguments

x DESeqDataSet object containing gene counts.
groups vector or data frame object with associated sample information such as experimental groups. The information is displayed in mouseover tooltips, and appropriate vector(s) can be used to adjust the plot using scale_by, colour_by and shape_by drop-down boxes of the widget.
labels character vector of sample names or labels.
continuous.colour TRUE if continuous colour schemes should be used. Defaults to FALSE where distinct colour schemes are used.
top integer indicating number of top genes used to calculate pairwise distances.
gene.selection character string specifying how genes are selected from the plot - "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 integer indicating the average count to be added to each observation to avoid taking log of zero when raw counts are transformed to log-counts-per-million values (using `edgeR::cpm` function).
html character string for naming HTML file or exportation of widget. The extension should be included in the file name e.g. "file.html".
width numeric value indicating width of widget in pixels.
height numeric value indicating width of widget in pixels.
... additional unused arguments.

Details
The left plot shows two MDS dimensions, with sample annotations displayed on hover. The right panel contains a bar plot of the eigenvalues of each dimension. The controls beneath the plots can be used to change the dimensions being displayed, and the scale, colour and shape of points. The interactive MDS plot allows users to adjust sample points by scale, colour and shape for multiple vectors associated with sample information. This is carried out most effectively when `x$samples` includes an abundance of sample information, or when a data frame object is supplied to `groups`. If a simple character or factor vector is given to `groups` (with the default of `continuous.colour=FALSE`), then sample points will have no scaling options, but can only be adjusted in colour and shape by `groups` and `labels`. Instead, if `groups` is a numeric vector (e.g. library size or expression level of a specific gene), then the plot can be scaled and coloured by the numeric values with `continuous.colour=TRUE`. For more details, refer to `limma::plotMDS`.

Value
htmlwidget object or NULL if `html` argument is specified.

Author(s)
Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also
glimmaMDS, glimmaMDS.default, glimmaMDS.DGEList

Examples
dge <- readRDS(system.file("RNAseq123/dge.rds", package = "Glimma"))
dds <- DESeq2::DESeqDataSetFromMatrix(
  countData = dge$counts,
  colData = dge$samples,
  rowData = dge$genes,
  design = ~group
)
glimmaMDS(dds)

glimmaMDS.DGEList  Glimma MDS Plot

Description

Draws a two-panel interactive MDS plot using a DGEList x. Transforms counts using edgeR::cpm(x, log=TRUE, prior.count = prior.count).

Usage

```r
## S3 method for class 'DGEList'
glimmaMDS(
  x,
  groups = x$samples,
  labels = rownames(x$samples),
  continuous.colour = FALSE,
  top = 500,
  gene.selection = c("pairwise", "common"),
  prior.count = 2,
  html = NULL,
  width = 900,
  height = 500,
  ...
)
```

Arguments

- **x**: DGEList object containing gene counts in x$counts.
- **groups**: vector or data frame object with associated sample information such as experimental groups. The information is displayed in mouseover tooltips, and appropriate vector(s) can be used to adjust the plot using scale_by, colour_by and shape_by drop-down boxes of the widget.
- **labels**: character vector of sample names or labels.
- **continuous.colour**: TRUE if continuous colour schemes should be used. Defaults to FALSE where distinct colour schemes are used.
- **top**: integer indicating number of top genes used to calculate pairwise distances.
- **gene.selection**: character string specifying how genes are selected from the plot - "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**: integer indicating the average count to be added to each observation to avoid taking log of zero when raw counts are transformed to log-counts-per-million values (using edgeR::cpm function).
character string for naming HTML file or exportation of widget. The extension should be included in the file name e.g. "file.html".

width numeric value indicating width of widget in pixels.

height numeric value indicating width of widget in pixels.

... additional unused arguments.

Details

The left plot shows two MDS dimensions, with sample annotations displayed on hover. The right panel contains a bar plot of the eigenvalues of each dimension. The controls beneath the plots can be used to change the dimensions being displayed, and the scale, colour and shape of points. The interactive MDS plot allows users to adjust sample points by scale, colour and shape for multiple vectors associated with sample information. This is carried out most effectively when x$samples includes an abundance of sample information, or when a data frame object is supplied to groups. If a simple character or factor vector is given to groups (with the default of continous.colour=FALSE), then sample points will have no scaling options, but can only be adjusted in colour and shape by groups and labels. Instead, if groups is a numeric vector (e.g. library size or expression level of a specific gene), then the plot can be scaled and coloured by the numeric values with continous.colour=TRUE. For more details, refer to limma::plotMDS.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also

glimmaMDS, limmaMDS.default, limmaMDS.DESeqDataSet

Examples

dge <- readRDS(system.file("RNAseq123/dge.rds", package = "Glimma"))
glimmaMDS(dge)
Description

Generic function for drawing a two-panel interactive volcano plot, a special case of the glimmaXY plot. The function invokes the following methods which depend on the class of the first argument:

- `glimmaVolcano.MArrayLM` for limma analysis
- `glimmaVolcano.DGEExact` for edgeR analysis, produced from `exactTest`
- `glimmaVolcano.DGELRT` for edgeR analysis, produced from `glmLRT`
- `glimmaVolcano.DESeqDataSet` for DESeq2 analysis

Usage

`glimmaVolcano(x, ...)`

Arguments

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

Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the x-axis versus -log10(pvalue). The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

Examples

```r
dge <- readRDS(
  system.file("RNAseq123/dge.rds", package = "Glimma"))
design <- readRDS(
  system.file("RNAseq123/design.rds", package = "Glimma"))
contr.matrix <- readRDS(
  system.file("RNAseq123/contr.matrix.rds", package = "Glimma"))

v <- limma::voom(dge, design)
```
```r
vfit <- limma::lmFit(v, design)
vfit <- limma::contrasts.fit(vfit, contrasts = contr.matrix)
efit <- limma::eBayes(vfit)

glimmaVolcano(efit, dge = dge)
```

---

### Description

Draws a two-panel interactive volcano plot from an DESeqDataSet object. This is a special case of the `glimmaXY` plot.

### Usage

```r
## S3 method for class 'DESeqDataSet'
glimmaVolcano(
x, 
counts = DESeq2::counts(x),
groups = extractGroups(colData(x)),
status = NULL,
anno = NULL,
display.columns = NULL,
status.cols = c("#1052bd", "silver", "#cc212f"),
sample.cols = NULL,
transform.counts = c("logcpm", "cpm", "rpkm", "none"),
main = "Volcano Plot",
xlab = "logFC",
ylab = "negLog10PValue",
html = NULL,
width = 920,
height = 920,
...
)
```

### Arguments

- **x**: DESeqDataSet object from which summary statistics are extracted from to create summary (left) plot.
- **counts**: numeric matrix with nrow(x) rows containing gene expression values.
- **groups**: vector/factor representing the experimental group for each sample; see `extractGroups` for default value.
- **status**: vector of length nrow(x) indicating the status of each gene.
- **anno**: dataframe with nrow(x) rows containing gene annotations.
display.columns
character vector containing names of columns from anno from which to display in mouseover tooltips and table.

status.cols
vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.

class.cols
character vector of length ncol(counts) containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to groups.

transform.counts
the type of transformation used on the counts - "logcpm" for using edgeR::cpm(counts, log=TRUE); "cpm" for edgeR::cpm(counts); "rpkm" for edgeR::rpkm(counts); "logrpkm" for edgeR::rpkm(counts, log=TRUE); and "none" for no transformation). Defaults to "logcpm".

main
character string for the main title of summary plot.

xlab
character string for the x-axis label of summary plot.

ylab
character string for the y-axis label of summary plot.

html
character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

width
numeric value indicating width of widget in pixels.

height
numeric value indicating width of height in pixels.

... additional unused arguments.

Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the x-axis versus -log10(pvalue). The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also

glimmaVolcano,limmaVolcano.MArrayLM,limmaVolcano.DGEEexact,limmaVolcano.DGELRT
Examples

dge <- readRDS(
  system.file("RNAseq123/dge.rds", package = "Glimma"))

dds <- DESeq2::DESeqDataSetFromMatrix(
  countData = dge$counts,
  colData = dge$samples,
  rowData = dge$genes,
  design = ~group
)

dds <- DESeq2::DESeq(dds, quiet=TRUE)
glimmaVolcano(dds)

---

glimmaVolcano.DGEExact

Glimma Volcano Plot

Description

Draws a two-panel interactive volcano plot from an DGEExact object. This is a special case of the
glimmaXY plot.

Usage

```r
## S3 method for class 'DGEExact'
glimmaVolcano(
  x,
  dge = NULL,
  counts = dge$counts,
  groups = dge$samples$group,
  status = edgeR::decideTestsDGE(x),
  anno = x$genes,
  display.columns = NULL,
  status.cols = c("#1052bd", "silver", "#cc212f"),
  sample.cols = NULL,
  p.adj.method = "BH",
  transform.counts = c("logcpm", "cpm", "rpkm", "none"),
  main = paste(x$comparison[2], "vs", x$comparison[1]),
  xlab = "logFC",
  ylab = "negLog10PValue",
  html = NULL,
  width = 920,
  height = 920,
  ...)
```

**Arguments**

- **x**
  DGEExct object from which summary statistics are extracted from to create summary (left) plot.

- **dge**
  DGEList object with `nrow(x)` rows from which expression values are extracted from to create expression (right) plot. Gene counts are taken from `dge$counts` and sample groups from `dge$samples$group`. By default raw counts are transformed to log-cpm values (see more in the `transform.counts` argument).

- **counts**
  numeric matrix with `nrow(x)` rows containing gene expression values. This can be used to replace the gene counts from `dge$counts`, i.e. you may have log-rpkm values stored in a different object that you wish to use.

- **groups**
  vector of length `ncol(dge)` representing categorisation of samples in expression plot.

- **status**
  vector of length `nrow(x)` indicating the status of each gene. By default genes in the summary plot are coloured based on its differential expression status using an adjusted p-value cutoff of 0.05 by calling the `edgeR::decideTestsDGE()` function, where the value of -1 marks down-regulated genes, 0 marks genes with no expression difference, and 1 marks up-regulated genes.

- **anno**
  dataframe with `nrow(x)` rows containing gene annotations.

- **display.columns**
  character vector containing names of columns from `anno` from which to display in mouseover tooltips and table.

- **status.cols**
  vector of length 3 containing valid CSS strings for colours associated with `status` in the order of -1, 0 and 1.

- **sample.cols**
  character vector of length `ncol(counts)` containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to `groups`.

- **p.adj.method**
  character string specifying p-value adjustment method.

- **transform.counts**
  the type of transformation used on the counts - "logcpm" for using `edgeR::cpm(counts, log=TRUE)"; "cpm" for `edgeR::cpm(counts)"; "rpkm" for `edgeR::rpkm(counts)"; "logrpkm" for `edgeR::rpkm(counts, log=TRUE)"; and "none" for no transformation). Defaults to "logcpm".

- **main**
  character string for the main title of summary plot.

- **xlab**
  character string for the x-axis label of summary plot.

- **ylab**
  character string for the y-axis label of summary plot.

- **html**
  character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

- **width**
  numeric value indicating width of widget in pixels.

- **height**
  numeric value indicating width of height in pixels.

- **...**
  additional unused arguments.
Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the x-axis versus \(-\log_{10}(pvalue)\). The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also

glimmaVolcano, glimmaVolcano.MArrayLM, glimmaVolcano.DGELRT, glimmaVolcano.DESeqDataSet

Examples

dge <- readRDS(
  system.file("RNAseq123/dge.rds", package = "Glimma"))
design <- readRDS(
  system.file("RNAseq123/design.rds", package = "Glimma"))
contr.matrix <- readRDS(
  system.file("RNAseq123/contr.matrix.rds", package = "Glimma"))

dge <- edgeR::estimateDisp(dge, design)
gfit <- edgeR::glmFit(dge, design)
glrt <- edgeR::glmLRT(gfit, design, contrast = contr.matrix)

glimmaVolcano(glrt, dge = dge)

Description

Draws a two-panel interactive volcano plot from an DGELRT object. This is a special case of the glimmaXY plot.
Usage

```r
# S3 method for class 'DGELRT'
glimmaVolcano(
  x,
  dge = NULL,
  counts = dge$counts,
  groups = dge$samples$group,
  status = edgeR::decideTestsDGE(x),
  anno = x$genes,
  display.columns = NULL,
  status.cols = c("#1052bd", "silver", "#cc212f"),
  sample.cols = NULL,
  p.adj.method = "BH",
  transform.counts = c("logcpm", "cpm", "rpkm", "none"),
  main = paste(x$comparison[2], "vs", x$comparison[1]),
  xlab = "logFC",
  ylab = "negLog10PValue",
  html = NULL,
  width = 920,
  height = 920,
  ...
)
```

Arguments

- **x**: DGELRT object from which summary statistics are extracted from to create summary (left) plot.
- **dge**: DGEList object with nrow(x) rows from which expression values are extracted from to create expression (right) plot. Gene counts are taken from dge$counts and sample groups from dge$samples$group. By default raw counts are transformed to log-cpm values (see more in the transform.counts argument).
- **counts**: numeric matrix with nrow(x) rows containing gene expression values. This can be used to replace the gene counts from dge$counts, i.e. you may have log-rpkm values stored in a different object that you wish to use.
- **groups**: vector of length ncol(dge) representing categorisation of samples in expression plot.
- **status**: vector of length nrow(x) indicating the status of each gene. By default genes in the summary plot are coloured based on its differential expression status using an adjusted p-value cutoff of 0.05 by calling the edgeR::decideTestsDGE() function, where the value of -1 marks down-regulated genes, 0 marks genes with no expression difference, and 1 marks up-regulated genes.
- **anno**: dataframe with nrow(x) rows containing gene annotations.
- **display.columns**: character vector containing names of columns from anno from which to display in mouseover tooltips and table.
- **status.cols**: vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.
sample.cols character vector of length `nrow(counts)` containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to `groups`.

`p.adj.method` character string specifying p-value adjustment method.

`transform.counts` the type of transformation used on the counts - "logcpm" for using `edgeR::cpm(counts, log=TRUE)"; "cpm" for `edgeR::cpm(counts)"; "rpkm" for `edgeR::rpkm(counts)"; "logrpkm" for `edgeR::rpkm(counts, log=TRUE)"; and "none" for no transformation). Defaults to "logcpm".

`main` character string for the main title of summary plot.

`xlab` character string for the x-axis label of summary plot.

`ylab` character string for the y-axis label of summary plot.

`html` character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

`width` numeric value indicating width of widget in pixels.

`height` numeric value indicating width of height in pixels.

... additional unused arguments.

Details

The summary plot on the left represents gene-wise log-fold-change (logFC) on the x-axis versus -log10(pvalue). The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also

glimmaVolcano, glimmaVolcano.MArrayLM, glimmaVolcano.DGExa...


glimmaVolcano.MArrayLM

Glimma Volcano Plot

Description

Draws a two-panel interactive volcano plot from an MArrayLM object. This is a special case of the glimmaXY plot.

Usage

## S3 method for class 'MArrayLM'
glimmaVolcano(
x,  
dge = NULL,  
counts = dge$count,  
groups = dge$samples$group,  
coef = ncol(x$coefficients),  
status = limma::decideTests(x),  
anno = x$genes,  
display.columns = NULL,  
status.cols = c("#1052bd", "silver", "#cc212f"),  
sample.cols = NULL,  
p.adj.method = "BH",  
transform.counts = c("logcpm", "cpm", "rpkm", "none"),  
main = colnames(x)[coef],  
xlab = "logFC",  
ylab = "negLog10PValue",  
html = NULL,  
width = 920,  
height = 920,  
...  
)

Arguments

x MArrayLM object from which summary statistics are extracted from to create summary (left) plot.

dge DGEList object with nrow(x) rows from which expression values are extracted from to create expression (right) plot. Gene counts are taken from dge$count and sample groups from dge$samples$group. By default raw counts are transformed to log-cpm values (see more in the transform.counts argument).

counts numeric matrix with nrow(x) rows containing gene expression values. This can be used to replace the gene counts from dge$count, i.e. you may have log-rpkm values stored in a different object that you wish to use.

groups vector of length ncol(dge) representing categorisation of samples in expression plot.
coef
integer indicating the column in *x* from the summary plot is created.

status
vector of length *nrow(x)* indicating the status of each gene. By default genes in the summary plot are coloured based on its differential expression status using an adjusted p-value cutoff of 5% by calling the `limma::decideTests` function, where the value of -1 marks down-regulated genes, 0 marks genes with no expression difference, and 1 marks up-regulated genes.

anno
dataframe with *nrow(x)* rows containing gene annotations.

display.columns
character vector containing names of columns from *anno* from which to display in mouseover tooltips and table.

status.cols
vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.

sample.cols
character vector of length *ncol(counts)* containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to *groups*.

p.adj.method
character string specifying p-value adjustment method.

transform.counts
the type of transformation used on the counts - "logcpm" for using `edgeR::cpm(counts, log=TRUE)"; "cpm" for `edgeR::cpm(counts)"; "rpkm" for `edgeR::rpkm(counts)"; "logrpkm" for `edgeR::rpkm(counts, log=TRUE)"; and "none" for no transformation). Defaults to "logcpm".

main
character string for the main title of summary plot.

xlab
character string for the x-axis label of summary plot.

ylab
character string for the y-axis label of summary plot.

html
character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

width
numeric value indicating width of widget in pixels.

height
numeric value indicating width of height in pixels.

...
additional unused arguments.

Details
The summary plot on the left represents gene-wise log-fold-change (logFC) on the x-axis versus -\log_{10}(pvalue). The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value
htmlwidget object or NULL if *html* argument is specified.
Author(s)
Hasaru Kariyawasam, Shian Su and Oliver Voogd

See Also
glimmaVolcano, glimmaVolcano.DGEEexact, glimmaVolcano.DGELRT, glimmaVolcano.DESeqDataSet

glimmaXY

Glimma XY Plot

Description
Draws a two-panel interactive XY scatter plot.

Usage
glimmaXY(
x, 
y, 
xlab = "x", 
ylab = "y", 
dge = NULL, 
counts = dge$counts, 
groups = dge$samples$group, 
status = rep(0, length(x)), 
anno = NULL, 
display.columns = NULL, 
status.cols = c("#1052bd", "silver", "#cc212f"), 
sample.cols = NULL, 
transform.counts = c("logcpm", "cpm", "rpkm", "none"), 
main = "XY Plot", 
html = NULL, 
width = 920, 
height = 920
)

Arguments

x numeric vector of values to plot on the x-axis of the summary plot.
y numeric vector of values to plot on the y-axis of the summary plot.
xlab character string for the x-axis label of summary plot.
ylab character string for the y-axis label of summary plot.
dge DGEList object with length(x) rows from which expression values are extracted from to create expression (right) plot. Gene counts are taken from dge$counts and sample groups from dge$samples$group.
counts numeric matrix with length(x) rows containing gene expression values. This can be used to replace raw gene counts from dge$counts with transformed counts e.g. logCPM or logRPKM values.

groups vector of length ncol(counts) representing categorisation of samples in expression plot.

status vector of length length(x) indicating the status of each gene. A value of -1 marks a down-regulated gene, 0 marks a gene with no expression difference, and 1 marks an up-regulated gene.

anno dataframe with length(x) rows containing gene annotations.

display.columns character vector containing names of columns from anno from which to display in mouseover tooltips and table.

status.cols vector of length 3 containing valid CSS strings for colours associated with status in the order of -1, 0 and 1.

sample.cols character vector of length ncol(counts) containing valid CSS strings for colours associated with each sample to be displayed on the expression plot. If left unspecified, samples will be coloured according to groups.

transform.counts the type of transformation used on the counts - "logcpm" for using edgeR::cpm(counts, log=TRUE); "cpm" for edgeR::cpm(counts); "rpkm" for edgeR::rpkm(counts); "logrpkm" for edgeR::rpkm(counts, log=TRUE); and "none" for no transformation). Defaults to "logcpm".

main character string for the main title of summary plot.

html character string for naming HTML file for exportation of widget. The extension should be included in the file name e.g. "file.html".

width numeric value indicating width of widget in pixels.

height numeric value indicating width of height in pixels.

Details

The summary plot on the left displays the x and y values specified. The expression plot on the right displays sample expression values for a single gene. Interactions with the htmlwidget include clicking on genes (points) in the summary plot to bring up associated sample expression values in the expression plot, as well as the summary statistics in the table below. Alternatively, users can interact with the table by clicking on genes (rows) to highlight genes in the summary plot, as well as bring up associated sample expression values in the expression plot. Briefly, other interactive features include a search box for the table, buttons to save plots and data (summary statistics and expression values), additional pop-up information when hovering on points in plots, and rescaling of the y-axis in the expression plot.

Value

htmlwidget object or NULL if html argument is specified.

Author(s)

Hasaru Kariyawasam, Shian Su and Oliver Voogd
Examples

dge <- readRDS(
  system.file("RNAseq123/dge.rds", package = "Glimma"))
design <- readRDS(
  system.file("RNAseq123/design.rds", package = "Glimma"))
contr.matrix <- readRDS(
  system.file("RNAseq123/contr.matrix.rds", package = "Glimma"))

v <- limma::voom(dge, design)
vfit <- limma::lmFit(v, design)
vfit <- limma::contrasts.fit(vfit, contrasts = contr.matrix)
efit <- limma::eBayes(vfit)

glimmaXY(efit$Amean, efit$coefficients)

glimmaXYWidget

GlimmaXY HTMLWidget Wrapper

Description

Passes packaged data to JS interface for rendering.

Usage

glimmaXYWidget(xData, width, height, html)

Arguments

    xData       packaged data object returned from buildXYData()
    width       htmlwidget element width in pixels
    height      htmlwidget element height in pixels
    html        name of HTML file (including extension) to export widget into rather than displaying the widget; NULL by default.

Value

htmlwidget object for XY plot internal use
glimma_plot  Glimma plot manager

Description

Core glimma plot manager. Generates environment for glimma plots.

Usage

glimma_plot(
  ..., 
  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.

gllink  Plot linkages

Description

Helper function for writing the link properties in interactive Glimma plots.
Usage

```r
glink(
  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.

glMDPlot

**Glimma MD Plot**

Description

Draw an interactive MD plot

Usage

```r
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`

---

**glMDPlot.default**

**Glimma MD Plot**

**Description**

Draw an interactive MD plot from a data.frame

**Usage**

```r
## 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 tool tips 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)
**glMDPlot.DESeqDataSet**

**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**

```r
## 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**

```r
## 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

**Description**

Draw an interactive MD plot from a DGELRT object.

**Usage**

```r
## 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("#ff77b4", 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
transform
main
xlab
ylab
side.xlab
side.ylab
side.log
side.gridstep
p.adj.method
jitter
side.main
display.columns
cols
sample.cols
path
folder
html
launch
...

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

```r
## 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
Description

Draw an interactive MD plot from a MArrayLM object

Usage

```r
## 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
transform
main
xlab
ylab
side.main
side.xlab
side.ylab
side.log
side.gridstep
doef
p.adj.method
jitter
display.columns
cols
sample.cols
path
folder
html
launch
...

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
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

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

Arguments

- `html`: name of the HTML page containing plots from `glMDPlot`.

Value

None

See Also

`glMDPlot`

Examples

```r
glMDRmd()
```

glMDSPlot

Glimma MDS Plot

Description

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

Usage

```r
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

description

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.
glMDSPlot.DESeqDataSet

**Description**

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

**Usage**

```r
## 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,
  ...
)
```

**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
Arguments

\texttt{x} \hspace{1cm} \text{the DESeqData\texttt{Set} containing the gene expressions.}
\texttt{top} \hspace{1cm} \text{the number of top most variable genes to use.}
\texttt{labels} \hspace{1cm} \text{the labels for each sample.}
\texttt{groups} \hspace{1cm} \text{the experimental group to which samples belong.}
\texttt{gene.selection} \hspace{1cm} "pairwise" if most variable genes are to be chosen for each pair of samples or "common" to select the same genes for all comparisons.
\texttt{prior.count} \hspace{1cm} \text{average count to be added to each observation to avoid taking log of zero. Used only if log=TRUE.}
\texttt{main} \hspace{1cm} \text{the title of the plot.}
\texttt{path} \hspace{1cm} \text{the path in which the folder will be created.}
\texttt{folder} \hspace{1cm} \text{the name of the fold to save html file to.}
\texttt{html} \hspace{1cm} \text{the name of the html file to save plots to.}
\texttt{launch} \hspace{1cm} \text{TRUE to launch plot after call.}
\texttt{...} \hspace{1cm} \text{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 \hspace{1cm} \text{Glimma MDS Plot}

Description

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

Usage

```r
## S3 method for class 'DGEList'

glMDSPlot(
  x,
  top = 500,
  labels = NULL,
  groups = rep(1, ncol(x)),
  ...)
```

---

**Arguments**

- **x**: the DESeqData\texttt{Set} 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 \hspace{1cm} \text{Glimma MDS Plot}

**Description**

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

**Usage**

```r
## S3 method for class 'DGEList'


glMDSPlot(
  x,
  top = 500,
  labels = NULL,
  groups = rep(1, ncol(x)),
  ...)
```
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
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

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 = "",
)
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

glTable

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**  
```r
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**  
```r
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 glMDPPlot, 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,

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**


---

**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

* **RNAseq**
  lymphomaRNAseq, 64

* **internal**
  buildXYData, 4
effectGroups, 5
glBar, 6
glBar.default, 6
glimma_plot, 39
glimmaXYWidget, 38
glLink, 39
glScatter, 58
glScatter.default, 58
gltable, 60
gltablelink, 61
makeJson, 64
makeJson.data.frame, 65
makeJson.jschart, 65

* **microarray**
  arraydata, 3
as.hexcol, 3
buildXYData, 4

effectTest, 8, 26
effectGroups, 5, 10, 27
glBar, 6
glBar.default, 6, 6
glimma, 8
glimma_plot, 39
glimmaMA, 8, 11, 14, 16, 18
glimmaMA.DESeqDataSet, 8, 9, 14, 16, 18
glimmaMA.DGEExact, 8, 11, 12, 16, 18
glimmaMA.DGELRT, 8, 11, 14, 18
glimmaMA.MArrayLM, 8, 11, 14, 16, 18
glimmaMD (glimmaMA), 8
glimmaMDS, 19, 22, 23, 25
glimmaMDS.default, 19, 20, 23, 25
glimmaMDS.DESeqDataSet, 19, 22, 25
glimmaMDS.DGEList, 19, 22, 23, 24
glimmaVolcano, 25, 28, 31, 33, 36
glimmaVolcano.DESeqDataSet, 26, 27, 31, 33, 36
glimmaVolcano.DGEExact, 26, 28, 29, 33, 36
glimmaVolcano.DGELRT, 26, 28, 31, 31, 36
glimmaVolcano.MArrayLM, 26, 28, 31, 33, 34
glimmaXY, 36
glimmaXYWidget, 38
glLink, 39
glMDPlot, 8, 40, 53
glMDPlot.default, 41, 41
glMDPlot.DESeqDataSet, 41, 43
glMDPlot.DESeqResults, 45
glMDPlot.DGEExact, 41, 47
glMDPlot.DGELRT, 41, 49
glMDPlot.MArrayLM, 41, 51
glMDRmd, 53
glMDPlot, 8, 53
glMDPlot.default, 54, 54
glMDPlot.DESeqDataSet, 55
glMDPlot.DGEList, 54, 56
g timp, 8, 26
glScatter, 58
glScatter.default, 58
gltable, 60
gltablelink, 61
glxyplot, 8, 61
is.hex, 63
lymphomaRNAseq, 64
makeJson, 64
makeJson.data.frame, 65
makeJson.jschart, 65
p.adjust, 48, 50, 52

66