

Package ‘aveytoolkit’

March 22, 2016

Title Compilation of functions for data analysis

Version 0.1.0.9017

Description Provides functionality for simple functions like Pause(), and resetPar() as well as some complex plotting functions. Some functions were copied from StackOverflow or other online sites while others were written by Stefan Avey. The author and source is attributed in the documentation of each function.

Depends R (>= 3.0.2)

License CCO

LazyData true

R topics documented:

| | |
|-----------------------------|----|
| annotatePDFs | 2 |
| AverageReplicates | 3 |
| aveytoolkit | 4 |
| barplotCI | 4 |
| browseIndex | 5 |
| cbind.fill | 5 |
| collapseDataset | 6 |
| fishersMethod | 8 |
| FoldChange | 8 |
| geomMean | 9 |
| getBaseTheme | 10 |
| getLoginDetails | 11 |
| ggSmartBoxplot | 12 |
| ggSmoothExprPlot | 13 |
| MakeDF | 15 |
| makeTransparent | 16 |
| Multiplot | 17 |
| Pause | 18 |
| PlotTimeCourse | 19 |
| PrepareExpression | 20 |
| ProcessNames | 21 |
| RepeatBefore | 22 |
| resetPar | 23 |
| runLimma | 24 |
| sigHeatmap | 25 |

VennDiagram 27

writeGMT 28

Index 29

| | |
|--------------|---------------------|
| annotatePDFs | <i>annotatePDFs</i> |
|--------------|---------------------|

Description

Add content on top of existing PDF files and can combine them into one file using command line tools

Usage

```
annotatePDFs(host = "localhost", inFiles, outFiles, titles = NULL,  
  footers = NULL, combineMultiple = TRUE,  
  removeMultiple = combineMultiple)
```

Arguments

| | |
|---------------------|---|
| host | name of the host (e.g. «user»@«IP Address») |
| inFiles | a character vector giving the current PDF names |
| outFiles | a character vector of filenames to save PDFs to. Only the first value is used if combineMultiple is TRUE |
| titles | a character vector of titles for each plot in the same order as cytoscapeWindowList (default is to use the title slot in the cytoscapeWindowList). Use NULL to omit a title. If only 1 title is given, it will be put on all plots. |
| footers | a character vector of footers for each plot in the same order as cytoscapeWindowList (default is NULL for no footers). If only 1 footer is given, it will be put on all plots. |
| combineMultiple | Should multiple windows be combined into a single PDF with one image per page? (Default TRUE) |
| removeMultiple | Should the temporary files be removed (use only if combining PDFs)? |
| cytoscapeWindowList | a list of cytoscapeWindow objects for the windows to save |
| filenames | a character vector of filenames to save PDFs to. Only the first value is used if combineMultiple is TRUE |

Details

This function is a wrapper around some shell commands that allows you to annotate multiple PDF files and combine them into 1. This wrapper requires ghostscript and coherent PDF (cpdf) <http://www.coherentpdf.com/> on the host machine

Value

an (invisible) list of exit codes for each bash operation

Author(s)

Stefan Avey

Examples

```
## Still keeps the original files because
## removeMultiple is FALSE
annotatePDFs(inFiles=paste0("inFile", 1:2)
              outFile=paste0("outFile", 1:2),
              titles=paste0("testTitle", 1:2),
              footers=paste0("testFooter", 1:2),
              combineMultiple=TRUE,
              removeMultiple=FALSE)
```

| | |
|-------------------|--------------------------|
| AverageReplicates | <i>AverageReplicates</i> |
|-------------------|--------------------------|

Description

This function averages replicates in a matrix or data.frame

Usage

```
AverageReplicates(eSubSet, numRep)
```

Arguments

| | |
|---------|--|
| eSubSet | a matrix or data.frame of values with samples as columns |
| numRep | the number of replicates |

Value

a data.frame of averaged values with column names coming from the first of each of the replicates with .avg appended

Note

Assumes that the replicates are all next to each other

Author(s)

Stefan Avey

Examples

```
mat <- matrix(rnorm(1000), ncol=10) ## 10 columns of random uniform numbers
avgMat <- AverageReplicates(mat, numRep=2) ## average adjacent pairs of columns
```

| | |
|-------------|---------------------|
| aveytoolkit | <i>aveytoolkit.</i> |
|-------------|---------------------|

Description

aveytoolkit.

| | |
|-----------|------------------|
| barplotCI | <i>barplotCI</i> |
|-----------|------------------|

Description

Create a barplot from the list x with one bar for each element of x

Usage

```
barplotCI(x, CIs = NULL, compareTo = 1, ...)
```

Arguments

| | |
|-----------|---|
| x | list of values to create a boxplot from |
| CIs | the confidence intervals. Default is NULL and they will be calculated as the 95% confidence interval. |
| compareTo | non-negative integer specifying to which element of x should comparisons be made for significance. If 0, no significance will be added. |
| ... | other arguments passed to barplot2 function |

Value

Same as return from barplot2. A numeric vector (or matrix, when beside = TRUE), say mp, giving the coordinates of `_all_` the bar midpoints drawn, useful for adding to the graph. If beside is true, use `colMeans(mp)` for the midpoints of each `_group_` of bars, see example.

Author(s)

Christopher Bolen (creator) ; Stefan Avey (modified)

browseIndex

browseIndex

Description

Simple function to open HTML page of index in the default browser

Usage

```
browseIndex(package = NULL, lib.loc = NULL)
```

Arguments

| | |
|---------|--|
| package | a string with the name of package to use. If no name is supplied, the most recently loaded package is used. |
| lib.loc | a string of the directory name of the R library, or <code><e2><80><98>NULL<e2><80><99></code> . The default value of <code><e2><80><98>NULL<e2><80><99></code> corresponds to all libraries currently known. |

Details

Only works for a single package. Could improve to list an HTML page with multiple packages that you could then choose from. Not currently implemented. Borrows the concepts from `utils::browseVignettes`

Value

invisibly, the URL passed to `browseURL` (i.e. "file:" + `<index_filename>`)

Author(s)

Stefan Avey

Examples

```
browseIndex("utils")
```

cbind.fill

resetPar

Description

Simple function to combine multiple objects by column while filling in NAs into extra rows created from differing lengths

Usage

```
cbind.fill(...)
```

Arguments

... the objects, that will be converted to a list, to bind column-wise

Value

the cbind'ed objects passed in

Author(s)

Dimitris Rizopoulos and Tyler Rinker

References

<http://stackoverflow.com/questions/7962267/cbind-a-df-with-an-empty-df-cbind-fill>

See Also

[cbind](#)

Examples

```
x<-matrix(1:10,5,2)
y<-matrix(1:16, 4,4)
z<-matrix(1:12, 2,6)

cbind.fill(x,y)
cbind.fill(x,y,z)
cbind.fill(mtcars, mtcars[1:10,])
```

collapseDataset

collapseDataset

Description

Collapses a dataset from probes to gene symbols.

Usage

```
collapseDataset(exprsVals, platform = NULL, mapVector = NULL, oper = max,
  prefer = c("none", "up", "down"), singleProbeset = FALSE,
  returnProbes = FALSE, deProbes = NULL, debug = FALSE)
```

Arguments

| | |
|-----------|---|
| exprsVals | a matrix or data.frame of numeric values with rownames denoting the identifiers. |
| platform | the microarray platform the data comes from for extracting the gene symbols |
| mapVector | a named character vector with names specifying the current identifiers (probes matching the rownames of exprsVals) and the values of the vector specifying the gene symbols (or other identifier to collapse to). |
| oper | the operation used to choose which probe when multiple probes map to the same gene. Default is max which will calculate the maximum of the average. |

| | |
|----------------|--|
| prefer | one of "none", "up", or "down", can be abbreviated. |
| singleProbeset | If TRUE, the operation applies to the average over all conditions and all values for a gene will come from one probeset. Otherwise, if FALSE, the operation applies to the probesets over all conditions and the values for a gene may come from different probe sets. Default is FALSE for compatability reasons but TRUE is recommended. |
| returnProbes | if TRUE, a list of the collapsed expression matrix and the probes are both returned (see return). |
| deProbes | a list with named vectors "up" and "down" giving the names of up and downregulated probes |
| debug | When TRUE, things will be printed out to help debug errors |

Details

This function is designed to work for microarray data but can work for any sort of numeric matrix for which multiple rows need to be collapsed. The aggregate function would probably work better and speed this up but this code is the slow brute force way to do it.

If singleProbeset is set to FALSE, the default for compatability reasons but untested and not recommended, the values for each sample will be taken from the maximum across any probe that maps to that gene. This means that a gene's expression values may be a composition of values from different probes rather than a single probe. Most users will not need to use the 'prefer' argument. If prefer is "up", when multiple deProbes match the same gene, the upregulated will be chosen. Similary for "down". Default is "none" and the probe with the 'oper' (default max) will be chosen.

Note that it is possible for multiple probes to have the same operation (oper) over all conditions and, in this case, I've decided arbitrarily to choose the first one.

Value

If returnProbes is TRUE, a list containing the collapsed dataset in \$exprsVals and the probes chosen in \$probeSets. Otherwise, if returnProbes is FALSE, only the expression matrix is returned.

Author(s)

Christopher Bolen, Modified by Stefan Avey

Examples

```
## Trivial Example showing basic functionality
fakeExpr <- matrix(rnorm(50, mean=8, sd=1), ncol=5, nrow=10,
                  dimnames=list(probes=paste("probe", 1:10, sep='_'),
                                samples=paste("sample", LETTERS[1:5], sep='_')))
mv <- rep(paste("Gene", LETTERS[1:5], sep='_'), each=2) # mapVector
names(mv) <- rownames(fakeExpr)
res <- collapseDataset(fakeExpr, mapVector=mv, oper=max,
                      singleProbeset=TRUE, # recommend setting singleProbeset to TRUE
                      returnProbes=TRUE)

res$probes
## between probe_1 and probe_2, probe_2 was chosen for Gene_A
## between probe_3 and probe_4, probe_4 was chosen for Gene_B
## etc.

res$exprsVals # collapsed expression values
```

```
## only difference is in rownames, numbers are identical
all.equal(res$exprsVals, fakeExpr[res$probes,])
```

| | |
|---------------|----------------------|
| fishersMethod | <i>fishersMethod</i> |
|---------------|----------------------|

Description

This function combines multiple p-values according to Fisher's Method

Usage

```
fishersMethod(x)
```

Value

a single combined p-value

Author(s)

Mike Love

References

<http://mikelove.wordpress.com/2012/03/12/combining-p-values-fishers-method-sum-of-p-values-bino>

Examples

```
x <- c(runif(1000, 0, 1), runif(100, .1, .2))
fishersMethod(x)
```

| | |
|------------|-------------------|
| FoldChange | <i>FoldChange</i> |
|------------|-------------------|

Description

Calculate the fold change between pairs of conditions in a matrix or data frame

Usage

```
FoldChange(x, condNum, condDen, conditions, grouping, preserveOrder = FALSE,
  log2Transform = FALSE, oper = c("subtract", "divide"))
```


Arguments

| | |
|---------------|---|
| x | matrix or data.frame from which to calculate fold changes with samples in columns. |
| condNum | a vector of condition(s) to be used as the numerator in the fold change calculation |
| condDen | a vector of condition(s) to be used as the denominator in the fold change calculation |
| conditions | a vector with length equal to the number of columns of x containing the condition labels between which to find the fold changes. |
| grouping | a vector with length equal to the number of columns of x containing a grouping of the samples (e.g. subjects, cell lines, strains). |
| preserveOrder | if TRUE, the same ordering of the columns in x will be kept after columns in condDen are removed. If FALSE (the default for backwards compatability), the ordering is changed to sort by group, then by condNum in order passed in. |
| log2Transform | when 'TRUE', log2 transformation will be applied to x before taking the FC. If 'FALSE' (default) no transformation is applied and x is ASSUMED to be already log transformed. |
| oper | the operation to be performed between numerator and denominator. Default is 'subtract' because this is appropriate for log-transformed data. Set oper to 'divide' if data is on linear scale. |

Details

FoldChange takes the fold change of log2 transformed data by subtracting columns of the x dataframe or matrix depending on the conditions passed in.

Value

a data.frame of the fold changes with one column for each fold change

Author(s)

Stefan Avey

geomMean

geomMean

Description

Calculate the geometric mean

Usage

```
geomMean(x, na.rm = TRUE)
```

Arguments

| | |
|-------|--|
| x | a vector of positive numeric values. |
| na.rm | (optional) whether to remove NA values before calculation. Default is TRUE |

Details

This function handles negative or 0 values by warning that they are ignored and calculating the geometric mean without them

Value

the geometric mean of x

Author(s)

Paul McMurdie

References

<http://stackoverflow.com/questions/2602583/geometric-mean-is-there-a-built-in>

See Also

[exp](#) [sum](#) [log](#)

Examples

```
x <- 1:10
x2 <- x^2
x3 <- -5:5

geomMean(x)
mean(x)

geomMean(x2)
mean(x2)

## Warning because x3 contains negative values ##
geomMean(x3)
mean(x3)
```

getBaseTheme

getBaseTheme

Description

getBaseTheme Defines universal plotting settings to use with ggplot2

Usage

```
getBaseTheme()
```

Author(s)

Jason Vander Heiden <jason.vanderheiden@yale.edu>

Examples

```
p <- ggplot(mtcars, aes(wt, mpg))  
p <- p + geom_point() + getBaseTheme()  
plot(p)
```

`getLoginDetails`*getLoginDetails*

Description

Uses tcltk to display a prompt for a loginID and password

Usage

```
getLoginDetails()
```

Details

This function displays a window for a user to enter a loginID and password without showing the password. It may be bad package etiquette that tcltk is required for this function but not imported in the NAMESPACE file. This is done because loading tcltk will not work without a display and I want to be able to use this package (even if not this function) without a display

Value

an invisible named vector of loginID and password

Author(s)

Markus Gesmann, Barry Rowlingson

References

<http://www.r-bloggers.com/simple-user-interface-in-r-to-get-login-details/> <http://r.789695.n4.nabble.com/tkentry-that-exits-after-RETURN-tt854721.html#none>

See Also

tcltk

Examples

```
credentials <- getLoginDetails()  
## Do what needs to be done with loginID and password  
rm(credentials) # Delete credentials
```

ggSmartBoxplot

*ggSmartBoxplot***Description**

Boxplot wrapper for ggplot

Usage

```
ggSmartBoxplot(x, mat, splitRowBy = NA, splitColBy = NA, colorBy = NULL,
  rows, cols = NA, whichCols = NA, sep = ".", outlier.shape = 17,
  ylab = NULL, space = "fixed", scales = "fixed", fileName = NA,
  plot = TRUE, ...)
```

Arguments

| | |
|---------------|--|
| x | the variable to group by for boxplots |
| mat | data.frame or matrix of values to plot with samples in columns |
| splitRowBy | a factor used to split the data by row in facet_grid |
| splitColBy | a factor used to split the data by col in facet_grid |
| colorBy | a factor used for coloring. No coloring will be done if NULL (default) |
| rows | row names or row indices of the items to be plotted |
| cols | substring to search for with "grep" in column names to be plotted |
| whichCols | the column indices or full column names |
| sep | a separator used in searching for cols in the column names |
| outlier.shape | shape of outliers (default is 17, filled triangle) |
| ylab | if NULL, default is to use rownames. Can specify a string instead to use |
| space | If "fixed", the default, all panels have the same size. If "free_y" their height will be proportional to the length of the y scale; if "free_x" their width will be proportional to the length of the x scale; or if "free" both height and width will vary. This setting has no effect unless the appropriate scales also vary. |
| scales | Are scales shared across all facets (the default, "fixed"), or do they vary across rows ("free_x"), columns ("free_y"), or both rows and columns ("free") |
| fileName | |
| plot | logical specifying whether or not to plot the plot(s). Default is TRUE. |
| ... | other arguments that are passed to qplot |
| filename | the name of a file to write a PDF to or NA to plot in standard graphics device. |

Value

invisibly returns a list with 2 elements: ggplot: the ggplot object to be plotted (this can be added to dat: a named list of the data frame(s) passed to data in ggplot. The names come from converting the rows argument to a character vector.

Author(s)

Stefan Avey

See Also[ggplot2](#), [qplot](#)**Examples**

```
data(OrchardSprays)
## Example of functionality
ggSmartBoxplot(x=OrchardSprays$treatment,
               mat=t(OrchardSprays[,1]),
               rows=1, whichCols=1:ncol(t(OrchardSprays)),
               colorBy=factor(OrchardSprays$rowpos+OrchardSprays$colpos > 9),
               xlab="Treatment")

## Not run:
cellType <- "PBMC"
## expr would be an expression matrix with genes in rows and samples in columns
geneSub <- grep("HLA-A29.1", rownames(expr))
age <- "Young"
ages <- c("Young", "Old")
responses <- c("NR", "R")
subset <- targetFClst[[cellType]]$Age %in% ages &
  targetFClst[[cellType]]$Response %in% responses
ggSmartBoxplot(x=targetFClst[[cellType]][subset, "Time"],
               mat=exprFClst[[cellType]], ylim=c(-1,1),
               rows=geneSub, whichCols=which(subset),
               colorBy=targetFClst[[cellType]][subset, "Response"],
               splitRowBy=targetFClst[[cellType]][subset, "Age"],
               xlab="Days (Post Vaccination)",
               fileName=NA)

## End(Not run)
```

ggSmoothExprPlot

*ggSmoothExprPlot***Description**

Wrapper around ggplot to transform data and plot profiles (e.g. expression or activity) over time

Usage

```
ggSmoothExprPlot(x, mat, rows, method = "auto", formula = formula("y ~ x"),
                 splitRowBy = NA, splitColBy = NA, colorBy = NULL, cols = NA,
                 whichCols = NA, sep = ".", colorByLabel = "Response", ggtitle = TRUE,
                 xlab = "Time (Days Post-Vaccination)", ylab = "Expression",
                 colors = c("#1B9E77", "#D95F02", "#7570B3", "#E7298A", "#66A61E", "#E6AB02",
                             "#A6761D", "#666666"), space = "fixed", scales = "fixed", fileName = NA,
                 plot = TRUE)
```

Arguments

| | |
|---------------------------|---|
| <code>x</code> | the numeric x-axis variable for the plot (usually time) |
| <code>mat</code> | data.frame or matrix of values to plot with samples in columns |
| <code>rows</code> | row names or row indices of the items to be plotted |
| <code>method</code> | smoothing method (function). See <code>stat_smooth</code> |
| <code>formula</code> | a formula to use for smoothing in <code>stat_smooth</code> (e.g. the default <code>"y ~ x"</code> or <code>"y ~ ns(x, 3)"</code>). |
| <code>splitRowBy</code> | a factor used to split the data by row in <code>facet_grid</code> |
| <code>splitColBy</code> | a factor used to split the data by col in <code>facet_grid</code> |
| <code>colorBy</code> | a factor used for coloring. No coloring will be done if NULL (default) |
| <code>cols</code> | substring to search for with <code>"grep"</code> in column names to be plotted |
| <code>whichCols</code> | the column indices or full column names |
| <code>sep</code> | a separator used in searching for cols in the column names |
| <code>colorByLabel</code> | the labels used for the color legend |
| <code>ggtitle</code> | logical. If TRUE, rows is coerced to character and passed to <code>ggtitle</code> |
| <code>xlab</code> | passed to <code>xlab</code> . Defaults to <code>"Time (Post-Vaccination)"</code> |
| <code>ylab</code> | passed to <code>ylab</code> . Defaults to <code>"Expression"</code> |
| <code>colors</code> | The colors to use. Defaults to the colors given by using RColorBrewer's "Dark2" palette (but RColorBrewer is not called directly so is not required). |
| <code>space</code> | If <code>"fixed"</code> , the default, all panels have the same size. If <code>"free_y"</code> their height will be proportional to the length of the y scale; if <code>"free_x"</code> their width will be proportional to the length of the x scale; or if <code>"free"</code> both height and width will vary. This setting has no effect unless the appropriate scales also vary. |
| <code>scales</code> | Are scales shared across all facets (the default, <code>"fixed"</code>), or do they vary across rows (<code>"free_x"</code>), columns (<code>"free_y"</code>), or both rows and columns (<code>"free"</code>) |
| <code>fileName</code> | the name of a file to write a PDF to or NA to plot in standard graphics device. |
| <code>plot</code> | logical specifying whether or not to plot the plot(s). Default is TRUE. |

Value

invisibly returns a list with 2 elements: `ggplot`: the `ggplot` object to be plotted (this can be added to `dat`: a named list of the data frame(s) passed to `data` in `ggplot`. The names come from converting the `rows` argument to a character vector.

Author(s)

Stefan Avey

See Also

[ggplot2](#)

Examples

```

data(OrchardSprays)
## Example of functionality
library(Biobase)
data(sample.ExpressionSet, package="Biobase")
dat <- sample.ExpressionSet
## Normally x-axis is time but in this dataset there is no time
## so we will use the `score` as the x-axis
genderF <- dat$sex == "Female"
ggSmoothExprPlot(x=dat$score[genderF],
                  mat=exprs(dat),
                  rows="31345_at",
                  whichCols=which(genderF), # females only
                  colorBy=as.factor(dat$type)[genderF],
                  colorByLabel="Condition",
                  xlab="score")
## Not run: tmp <- ggSmoothExprPlot(x=times[subset], mat=expr, rows=gene,
                                   formula=formula("y ~ ns(x,3)"),
                                   whichCols=subset, colorBy=target[subset,respType],
                                   splitColBy=splitby,
                                   splitRowBy=as.factor(target[subset,"Study"]),
                                   ggtitle=TRUE, colorByLabel=respType, plot=TRUE)

## End(Not run)

```

MakeDF

MakeDF

Description

Creates a data frame from a list. Useful for when the list elements have unequal lengths and `as.data.frame` fails.

Usage

```
MakeDF(list, names)
```

Arguments

| | |
|--------------------|-----------------------|
| <code>list</code> | the list to convert |
| <code>names</code> | the names of the list |

Value

a data frame of the converted list.

Author(s)

Josh O'Brien

References

<http://stackoverflow.com/questions/15753091/convert-mixed-length-named-list-to-data-frame>

See Also[gsub](#)**Examples**

```
## Test timing with a 50k-item list
ll <- createList(50000)
nms <- c("a", "b", "c")

system.time(makeDF(ll, nms))
# user  system elapsed
# 0.47    0.00    0.47
```

| | |
|-----------------|------------------------|
| makeTransparent | <i>makeTransparent</i> |
|-----------------|------------------------|

Description

Simple function to make some colors transparent

Usage

```
makeTransparent(alpha = 0.5, ...)
```

Arguments

| | |
|-------|------------------------------------|
| alpha | transparency factor in range [0,1] |
| ... | vector or list of colors |

Value

a vector of new colors made transparent

Author(s)

Ricardo Oliveros-Ramos

References

<http://stackoverflow.com/questions/8047668/transparent-equivalent-of-given-color>

See Also

[rgb](#), [col2rgb](#)

Examples

```
makeTransparent("red", "blue")
##[1] "#FF00007F" "#0000FF7F"
makeTransparent("red", "blue", alpha=0.8)
## [1] "#FF0000CC" "#0000FFCC"
```

Multiplot

Multiplot

Description

Multiple Plot Function for ggplot

Usage

```
Multiplot(..., plotlist = NULL, file, cols = 1, layout = NULL)
```

Arguments

| | |
|----------|---|
| ... | ggplot objects |
| plotlist | a list of ggplot objects |
| cols | Number of columns in layout |
| layout | A matrix specifying the layout. If present, 'cols' is ignored |

Details

If the layout is something like `matrix(c(1,2,3,3), nrow=2, byrow=TRUE)`, then plot 1 will go in the upper left, 2 will go in the upper right, and 3 will go all the way across the bottom.

Author(s)

R Cookbook

References

http://www.cookbook-r.com/Graphs/Multiple_graphs_on_one_page_%28ggplot2%29/

Examples

```
library(ggplot2)

## This example uses the ChickWeight dataset, which comes with ggplot2
# First plot
p1 <- ggplot(ChickWeight, aes(x=Time, y=weight, colour=Diet, group=Chick)) +
  geom_line() +
  ggtitle("Growth curve for individual chicks")

# Second plot
p2 <- ggplot(ChickWeight, aes(x=Time, y=weight, colour=Diet)) +
  geom_point(alpha=.3) +
  geom_smooth(alpha=.2, size=1) +
  ggtitle("Fitted growth curve per diet")

# Third plot
p3 <- ggplot(subset(ChickWeight, Time==21), aes(x=weight, colour=Diet)) +
  geom_density() +
  ggtitle("Final weight, by diet")
```

```

# Fourth plot
p4 <- ggplot(subset(ChickWeight, Time==21), aes(x=weight, fill=Diet)) +
  geom_histogram(colour="black", binwidth=50) +
  facet_grid(Diet ~ .) +
  ggtitle("Final weight, by diet") +
  theme(legend.position="none")      # No legend (redundant in this graph)

Multiplot(p1, p2, p3, p4, cols=2)

```

 Pause

Pause

Description

This function prompts for return key and waits until the return is pushed to continue execution. It is used often to view plots coded in a loop one at a time allowing the user to control when the next plot should be displayed

Usage

```
Pause(str = "continue", quiet = FALSE)
```

Arguments

| | |
|--------------------|---|
| <code>str</code> | optional string to display. Defaults to "continue". |
| <code>quiet</code> | if TRUE, no prompt is displayed. Default is FALSE |

Details

The Pause function uses `readline` to wait until a newline character (produced by the Enter key) is given. Instead of pressing Enter, a newline character can be used to automate this waiting time.

Value

NULL is returned by invisible

Author(s)

Stefan Avey

See Also

[readline](#), [invisible](#)

Examples

```

for(p in 1:10) {
  plot(-10:10, (-10:10)^p, type='b')
  Pause(paste0('see plot of x^',p+1))
}

```

PlotTimeCourse

PlotTimeCourse

Description

Plot helper function for PlotPCATimeCourse

Usage

```
PlotTimeCourse(x, y, colors, groups, sampleNames, pch = 19, plotTitle = "",
  legend.loc = "topleft", plotType = c("times", "points"), alpha = 0.15,
  cex.pt = 1, cex.time = 2, time.adj = c(-0.3, -0.3), arrLen = 0.1,
  lwd = 3, numRep = 3, plotFont = NULL, ctrl = TRUE, hourMarks = TRUE,
  legend.cex = 2, ...)
```

Arguments

| | |
|-------------|--|
| x | x-values for plotting |
| y | y-values for plotting |
| colors | named vector specifying colors for each sample |
| groups | the virus strain names for the conditions of interest |
| sampleNames | names of the samples |
| pch | the plotting character. Default is 19 (a closed circle). |
| plotTitle | a string used for the plotting title |
| legend.loc | location of the legend. Default is topleft. |
| plotType | one of "times" or "points". See Details |
| alpha | transparency factor passed to the alpha function (scales library) |
| cex.pt | size of points. Default is 1 |
| cex.time | size of time labels. Default is 2 |
| time.adj | the ammount to adjust the time labels. Default is c(-.3, -.3) which moves them to the lower left |
| arrLen | length of the arrows plotted at the average of each time point. Default is 0.1 |
| lwd | line width. Default is 3 |
| numRep | the number of replicates. Default is 3 |
| plotFont | which font to use for plotting text |
| ctrl | should the control time points be included? |
| hourMarks | should the 4 and 8 hour time points be marked on the plot? |
| legend.cex | size expansion for the legened. Default is 2. |
| ... | other arguments passed to heatmap.2 |

Details

If plotType is "times", ??? Also used to plot 2 genes expression against each other over time. If legend.loc is "none", no legend is plotted. ctrl flag indicates whether or not first numRep values in x and y are from a control measurement

Value

Nothing is returned

Note

Colors are assumed to have as the names attribute some part of the sampleName which can uniquely identify it.

Author(s)

Stefan Avey

| | |
|-------------------|--------------------------|
| PrepareExpression | <i>PrepareExpression</i> |
|-------------------|--------------------------|

Description

Takes in an expression matrix or data frame and prepares it for further analysis

Usage

```
PrepareExpression(eset, target, returnProbes = TRUE, labelColumn = "Label",
  select = colnames(target), collapse = ".")
```

Arguments

| | |
|--------------|---|
| eset | expression information and (potentially) other columns |
| target | target file where the column names of eset can be matched to 'Label' |
| returnProbes | whether probe mapping should be returned along with expression values in a list. This will only be returned correctly if there is a column of eset matching SYMBOL in any case. |
| labelColumn | the column name in the target file to use for matching the column names of eset. Default is "Label" |
| select | the column names of target to select and merge as the new column names of eset |
| collapse | |

Details

Wrote this to automate the few lines I always perform to "prepare" an expression set for further processing. I always want to remove the symbols column, rename the column names based on the target file, and (usually) change the rownames to be gene symbols. This function takes in the matrix format that I use to store processed expression files (in a package or file).

Value

if returnProbes is FALSE: a list of the prepared expression data frame (exprDat) and the (potentially modified) target data frame (target) . if returnProbes is TRUE (default): a list of three elements including the two above and probeMap (a vector mapping from gene symbols to probe names).

Author(s)

Stefan Avey

Examples

```
## Creating fake expression matrix
dat <- matrix(rnorm(1000, mean=8, sd=1), nrow=100, ncol=10)
colnames(dat) <- sample(letters[1:10], size=10)
fakeGenes <- as.vector(outer(LETTERS[1:26], LETTERS[1:26], paste0))
x <- data.frame(symbol=fakeGenes[1:nrow(dat)], dat, row.names=paste0("Probe_", 1:nrow(dat)))
head(x) # look at first 6 rows of toy data set
target <- data.frame(Label=letters[1:26], Class=rep(1:3, length.out=26))
head(target)

preplist <- PrepareExpression(x, target, select="Class")
head(preplist$exprDat)
head(preplist$target)
head(preplist$probeMap)

## Not run:
## Load expression data from HIPC package
library(HIPC)
data(y3ExprPBMC, y3Target)
preplist <- PrepareExpression(y3ExprPBMC, y3Target,
                             select=c("Response", "SubjectID", "Age", "Time"))

## End(Not run)
```

ProcessNames

*ProcessNames***Description**

Cleans up strings to make them pretty names by removing punctuation, whitespace, and specified substrings

Usage

```
ProcessNames(strs, stringsToRm = NULL, rmPunct = TRUE, sep = "_")
```

Arguments

| | |
|-------------|--|
| strs | vector or strings to process |
| stringsToRm | a vector or list of strings to search for and remove from strs |
| rmPunct | should punctuation be removed? Default is TRUE. |
| sep | character to replace whitespace |

Details

stringsToRm are replaced by " " in the order they are given using gsub. After this, punctuation is removed if rmPunct is TRUE. Then, leading and/or trailing whitespace will be removed and the sep will be used to separate words. This function is useful when reading in other people's data and you want to change the row or column names to legal R names or just shorten the names.

Value

a vector of modified strings from strs

Author(s)

Stefan Avey

See Also

[gsub](#)

Examples

```
badNames <- c("Who's Birthday?", "[Date]", "gift Received")
## Remove the string "Who's", remove punctuation, and separate words by '_'
goodNames <- ProcessNames(badNames, stringsToRm="Who's", rmPunct=TRUE, sep='_')
goodNames
## Remove the string "Who's", don't remove punctuation, and put no separation between words
goodNames <- ProcessNames(badNames, stringsToRm="Who's", rmPunct=FALSE, sep='')
goodNames
```

| | |
|--------------|---------------------|
| RepeatBefore | <i>RepeatBefore</i> |
|--------------|---------------------|

Description

Replaces NAs with the latest non-NA value

Usage

```
RepeatBefore(x)
```

Arguments

x a vector of values

Details

NA values will be replaced by the most recent value with a lower index. If there is no non-NA value before the NA appears, it will remain NA.

Value

a vector of values

Author(s)

Ruben

References

<http://stackoverflow.com/questions/7735647/replacing-nas-with-latest-non-na-value>

See Also[rep](#)**Examples**

```
x = c(NA, NA, 'a', NA, NA, NA, NA, NA, NA, NA, NA, 'b', 'c', 'd', NA, NA, NA, NA, NA, 'e')
newX <- RepeatBefore(x)
show(newX)
```

`resetPar`*resetPar*

Description

Simple function to reset plotting parameters for when things get wonky

Usage

```
resetPar()
```

Details

This function resets the graphical parameters from the par function. It flashes a new device on the screen but works to reset parameters. Meant to be used when things get hairy and not coded in scripts

Value

an invisible named list of parameters returned by calling par

Author(s)

Gavin Simpson

References

<http://stackoverflow.com/questions/5789982/reset-par-to-the-default-values-at-startup>

See Also

`par`

Examples

```
par(oma=c(4,10,2,1))
plot(1,1)
## paramter settings weren't saved so do a reset
resetPar()
plot(1,1)
```

runLimma

*runLimma***Description**

A wrapper around limma functions to perform a basic analysis on the given expression matrix

Usage

```
runLimma(eset, labels, contrasts, block = NULL, covariates = NULL,
  min.fold.change = 1, min.intensity = 4, p.cutoff = 0.05,
  fitOnly = FALSE)
```

Arguments

| | |
|-----------------|--|
| eset | the expression matrix (not expression set object) |
| labels | the labels for each column of the eset |
| contrasts | Vector of contrasts to make |
| block | vector or factor specifying a blocking variable on the arrays. Has length equal to the number of arrays. Must be <code>NULL</code> if <code>ndups > 2</code> (Not extensively tested, use with caution) |
| covariates | data frame of covariates (of same length as labels) to include in the model. Use this if there are paired samples, etc. |
| min.fold.change | Minimum log2 fold change to be differentially expressed. Default is 1. |
| min.intensity | Minimum log2 intensity (at any time) to be differentially expressed. Default is 4. |
| p.cutoff | FDR corrected cutoff for significant differential expression. Default is 0.05. |
| fitOnly | If true, will return fit2, rather than the matrix of significant genes. Default is FALSE. |

Details

Generally, an expression matrix is made up of rows of genes (or any other features) and columns of samples. The matrix has data for multiple classes (which are denoted with the 'labels' parameter) and the classes are compared using the vector of contrasts. Block can be used for biological (or technical) replicates or for separate subjects (in which case it will determine the inter-subject correlation). See `?duplicateCorrelation` for more information. ## Example: If you have a m X 10 matrix 'eset', with 5 samples of class A and 5 of class B, you could compare class A to class B using the following code:

```
results = runLimma(eset, c('A','A','A','A','A','B','B','B','B','B'), "B-A")
```

This will return to you a matrix with columns for each comparison and rows for each gene. The value in each cells will either be -1, 0, or 1, depending on whether the gene is significantly higher in B, not significant, or significantly higher in A, respectively. If you want information on p-values and fold changes, set "fitOnly=T", and you can access the fit object to get the information.

For other comparisons, you can look at the LIMMA user guide: `limmaUsersGuide()`

Value

depends on fitOnly

Author(s)

Christopher Bolen, Stefan Avey

See Also

[limma](#)

Examples

```
## Not run:
## Load in example data from colonCA package (install if necessary)
## source("http://bioconductor.org/biocLite.R")
## biocLite("colonCA")
library(colonCA)
## Look at head of data
head(pData(colonCA))
labels <- pData(colonCA)$class      # t and n for tumor and normal
## Data are paired (-1 and 1 come from same subject)
pair <- factor(abs(pData(colonCA)$samp))
covars <- data.frame(Pairing=as.character(pair))
deRes <- runLimma(eset=exprs(colonCA), labels=as.character(labels), contrasts="t-n",
                  covariates=covars, fitOnly=TRUE)

topTable(deRes)
## Or just do tests in the function to get -1, 0, 1 for DE status of each probe
testRes <- runLimma(eset=exprs(colonCA), labels=as.character(labels), contrasts="t-n",
                  covariates=data.frame(Pairing=as.character(pair)), fitOnly=FALSE)

head(testRes)

## End(Not run)
```

sigHeatmap

sigHeatmap

Description

Draw heatmap with significance indicated on boxes

Usage

```
sigHeatmap(hm, pvals, pvalDisplayName = "P-value", cutoff = 0.05,
  showOnly = c("both", "positive", "negative", "all"), main = "",
  mainNewlines = 0, sigChar = "*", Rowv = T, hclustMethod = "ward.D",
  ...)
```

Arguments

| | |
|-----------------|--|
| hm | a matrix of values used for drawing the heatmap |
| pvals | a list or data frame of (possibly FDR corrected but this is not handled by the function) positive p-values |
| pvalDisplayName | is printed on the heatmap as a legend. Default is "P-value" but might want to change to "Q-value", "FDR", etc. |
| cutoff | is threshold for significance of pvals. Default is 0.05 |
| showOnly | one of "both", "positive", "negative", or "all" can be abbreviated. |
| main | a string giving the plot main title. Default is "" (i.e. no title is plotted). |
| mainNewlines | a non-negative integer specifying the number of newline characters to plot before the main title. Used to make the title appear lower on the page. Default is 0 |
| sigChar | the character used for plotting on top of significant boxes |
| Rowv | should the rows be reordered, passed into heatmap.2 |
| hclustMethod | passed to the function stats::hclust. The agglomeration method to be used. This should be (an unambiguous abbreviation of) one of "ward.D", "ward.D2", "single", "complete", "average" (= UPGMA), "mcquitty" (= WPGMA), "median" (= WPGMC) or "centroid" (= UPGMC). Default is "ward.D". |
| ... | other arguments passed to heatmap.2 |

Details

Only rows with at least one significant column are plotted. If showOnly is "both", plots both positive and negative significant changes. If showOnly is "positive" or "negative", plots only rows of hm with significant positive or negative values respectively. If showOnly is "all", all rows of hm are shown.

Value

a vector indicating which of the rows of hm were determined to be significant and subsequently plotted

Author(s)

Stefan Avey

Examples

```
data(mtcars)
x <- as.matrix(mtcars)
alpha <- 10^-7 # significance threshold
## Caculate whether difference from mean is significant
## This is not done correctly but just to have some sort of significance
diffMean <- mtcars-matrix(colMeans(mtcars),
                          ncol=ncol(mtcars), nrow=nrow(mtcars), byrow=TRUE)
stdErr <- matrix(sapply(mtcars, sd)/sqrt(nrow(mtcars)),
                 ncol=ncol(mtcars), nrow=nrow(mtcars), byrow=TRUE)
tstats <- diffMean/stdErr
pvals <- pt(as.matrix(tstats), nrow(mtcars)-2, lower=FALSE)
op <- par(oma=c(4,0,0,20))
```

```

sel <- sigHeatmap(x, pvals=pvals, cutoff=alpha, showOnly="b",
                 main="mtcars Example Heatmap", sigChar="x", notecol='black',
                 notecex=2, Colv=T, Rowv=T, dendrogram="row", trace="none")
par(op)
## Which cars weren't selected
rownames(mtcars)[setdiff(1:nrow(mtcars), sel)]

```

VennDiagram

VennDiagram

Description

Draw a venn diagram of 2 or 3 sets

Usage

```
VennDiagram(setList, mar = c(0, 0, 1, 0), ...)
```

Arguments

setList a (named) list of the sets to be plotted. The names will be used on the plot. If the list is unnamed, the default names in [vennDiagram](#)

Details

Wrapper around the [limma vennDiagram](#) function to make it simpler.

Value

a data frame of binary values indicating membership in each set with rownames giving the set entries.

Author(s)

Stefan Avey

References

Code modified from <http://research.stowers-institute.org/mcm/venn.R>

See Also

[vennDiagram](#)

| | |
|----------|-------------------------------|
| writeGMT | <i>aveytoolkit_writeGMT.R</i> |
|----------|-------------------------------|

Description

Write out a GMT (Gene Matrix Transposed) file.

Usage

```
writeGMT(filename, sets, setNames = names(sets), setDescription = rep(NA,
length(sets)))
```

Arguments

| | |
|----------------|--|
| filename | the file to write to (should include '.gmt' extension) |
| sets | a list of character vectors containing the sets to write |
| setNames | a character vector of set names corresponding to sets. Defaults to 1, 2, 3, ..., length(sets) if none specified. |
| setDescription | a character vector of set descriptions corresponding to sets. Defaults to NA values if none specified. |

Details

Take in a vector of set names, descriptions, and gene identifiers and write them to a GMT file format.

http://www.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data_formats#GMT:_Gene_Matrix_Transposed_file

Author(s)

Stefan Avey

Index

*Topic **aveytoolkit**

AverageReplicates, [3](#)
browseIndex, [5](#)
cbind.fill, [5](#)
collapseDataset, [6](#)
fishersMethod, [8](#)
FoldChange, [8](#)
geomMean, [9](#)
getBaseTheme, [10](#)
getLoginDetails, [11](#)
ggSmartBoxplot, [12](#)
ggSmoothExprPlot, [13](#)
MakeDF, [15](#)
makeTransparent, [16](#)
Pause, [18](#)
PlotTimeCourse, [19](#)
PrepareExpression, [20](#)
ProcessNames, [21](#)
RepeatBefore, [22](#)
resetPar, [23](#)
runLimma, [24](#)
sigHeatmap, [25](#)
VennDiagram, [27](#)

annotatePDFs, [2](#)
as.data.frame, [15](#)
AverageReplicates, [3](#)
aveytoolkit, [4](#)
aveytoolkit-package (aveytoolkit), [4](#)

barplotCI, [4](#)
browseIndex, [5](#)

cbind, [6](#)
cbind.fill, [5](#)
col2rgb, [16](#)
collapseDataset, [6](#)

exp, [10](#)

fishersMethod, [8](#)
FoldChange, [8](#)

geomMean, [9](#)
getBaseTheme, [10](#)
getLoginDetails, [11](#)
ggplot2, [13](#), [14](#)
ggSmartBoxplot, [12](#)
ggSmoothExprPlot, [13](#)
gsub, [16](#), [22](#)

invisible, [18](#)

limma, [25](#), [27](#)
log, [10](#)

MakeDF, [15](#)
makeTransparent, [16](#)
Multiplot, [17](#)

Pause, [18](#)
PlotTimeCourse, [19](#)
PrepareExpression, [20](#)
ProcessNames, [21](#)

qplot, [13](#)

readline, [18](#)
rep, [23](#)
RepeatBefore, [22](#)
resetPar, [23](#)
rgb, [16](#)
runLimma, [24](#)

sigHeatmap, [25](#)
sum, [10](#)

VennDiagram, [27](#)
vennDiagram, [27](#)

writeGMT, [28](#)