Package 'caRpools'

December 6, 2015

Type Package

Title CRISPR AnalyzeR for Pooled CRISPR Screens

Version 0.83

Date 2015-12-06

Author Jan Winter, Florian Heigwer

Maintainer Jan Winter <jan.winter@dkfz-heidelberg.de>

Description CRISPR-Analyzer for pooled CRISPR screens (caRpools) provides an end-to-end analysis of CRISPR screens including quality control, hit candidate analysis, visualization and automated report generation using R markdown. Needs MAGeCK (http://sourceforge.net/p/mageck/wiki/Home/), bowtie2 for all functions. CRISPR (clustered regularly interspaced short palindromic repeats) is a method to perform genome editing. See <https://en.wikipedia.org/wiki/CRISPR> for more information on CRISPR.

Depends R (>= 3.1.0)

Imports rmarkdown,VennDiagram,DESeq2,sm,biomaRt,seqinr,scatterplot3d,xlsx

Suggests BiocGenerics,knitr,stringi

SystemRequirements MAGeCK (=0.51, from http://sourceforge.net/p/mageck/wiki/Home/), bowtie2 (http://bowtie-bio.sourceforge.net/bowtie2/index.shtml)

License GPL

URL http://www.crispr-analyzer.de, https://github.com/boutroslab/caRpools

BugReports https://github.com/boutroslab/caRpools

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2015-12-06 13:31:59

R topics documented:

aggregatetogenes *Aggregates pooled CRISPR screen sgRNA data to gene data*

Description

Aggregate all sgRNA data from pooled CRISPR screens to their corresponding gene level.

Usage

```
aggregatetogenes(data.frame, namecolumn = 1, countcolumn = 2,
agg.function = sum, extractpattern = expression("^(.+?)_.+"), type="aggregate")
```

Arguments

data.frame	data.frame with sgRNA readcounts. Must have one column with sgRNA names and one column with readcounts. Please note that the data must be formatted in a way, that gene names are included within the sgRNA name and can be extracted using the extractpattern expression. e.g. GENE_sgRNA1 -> GENE as gene name, _ as the separator and sgRNA1 as the sgRNA identifier.
namecolumn	integer, indicates in which column the names are stored
countcolumn	integer, indicates in which column the readcount are stored
agg.function	expression, the function to be used for aggregating data. Since for sgRNAs, aggregating data to the corresponding gene, sum will be right function in this case. Other possibilities include any other mathematical function R is capable of, e.g. median, mean.
extractpattern	Regular Expression, used to extract the gene name from the sgRNA name. Please make sure that the gene name extracted is accesible by putting its regular expression in brackets (). The default value expression("^(.+?)_.+") will look for the gene name (.+?) in front of the separator _ and any character afterwards .+ e.g. gene1_anything .
type	CaRpools can either aggregate the data frame (‘type = "annotate"‘) or annotate the gene identifiers only as an additional column (‘type = "annotate"‘). *Default* "aggregate" *Values* "aggregate", "annotate"

Details

aggregatetogenes can be used after load.file() to create quality control plots for aggregated gene data instead of single sgRNA data.

Before:

DesignID	fullmatch
AAK1_104_0	0
AAK1_105_0	197
AAK1_106_0	271
AAK1_107_0	1
AAK1_108_0	0

Afterwards:

DesignID	fullmatch
AAK1	880
AATK	2105
ABI1	1610

Value

A data.frame is returned with namecolumn (which no includes only gene names) and all readcount information aggregated by the agg.function.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

CONTROL1.g=aggregatetogenes(data.frame = CONTROL1, agg.function=sum,
                            extractpattern = expression("^(.+?)(_.+)"))
```

`caRpools` *CaRpools - CRISPR-AnalyzeR for pooled Screens*

Description

Analysis of pooled CRISPR screens based on mapped NGS readcount data or raw NGS FASTQ file.

Details

Package:	caRpools
Type:	Package
Version:	0.81
Date:	2015-07-31
License:	GPL V3

Author(s)

Jan Winter (German Cancer Research Center, DKFZ), Florian Heigwer (German Cancer Research Center, DKFZ)

Maintainer: Jan Winter <jan.winter@dkfz-heidelberg.de>

References

~~ Literature or other references for background information ~~

`carpools.hit.overview` *Analysis: Analysis of pooled CRISPR screening data using a Wilcoxon Test*

Description

Candidate genes from all methods can be plotted in an overview to identify overlapping signficant candidate genes using ‘carpools.hit.overview‘.

Usage

```
carpools.hit.overview(wilcox=NULL, deseq=NULL, mageck=NULL, cutoff.deseq = 0.001,
cutoff.wilcox = 0.05, cutoff.mageck = 0.05, cutoff.override=FALSE, cutoff.hits=NULL,
plot.genes="overlapping", type="all")
```

Arguments

`wilcox`	Data output from ‘stat.wilcox‘. *Default* NULL *Values* Data output from ‘stat.wilcox‘.
`deseq`	Data output from ‘stat.deseq‘. *Default* NULL *Values* Data output from ‘stat.deseq‘.
`mageck`	Data output from ‘stat.mageck‘. *Default* NULL *Values* Data output from ‘stat.mageck‘.
`cutoff.deseq`	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
`cutoff.wilcox`	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
`cutoff.mageck`	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
`cutoff.override`	Shall the p-value threshold be ignored? If this is TRUE, the top percentage gene of ‘cutoff.hits‘ is used instead. *Default* FALSE *Values* TRUE, FALSE
`cutoff.hits`	The percentatge of top genes being used if ‘cutoff.override=TRUE‘. *Default** NULL *Values* numeric
`plot.genes`	Defines what kind of data is used. By default, overlapping genes are highlighted in red color. *Default* "overlapping" *Values* "overlapping"
`type`	Defines whether all genes are plotted or only those being enriched or depleted. *Default* "all" *Values* "all", "enriched", "depleted"

Details

none

Value

Returns a generic plot.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)


data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr", filename = "TEST" , fdr.pval = 0.05)

carpools.hit.overview(wilcox=data.wilcox, deseq=data.deseq, mageck=data.mageck,
    cutoff.deseq = 0.001, cutoff.wilcox = 0.05, cutoff.mageck = 0.05,
    cutoff.override=FALSE, cutoff.hits=NULL, plot.genes="overlapping", type="enriched")
```

---

carpools.hit.scatter *Plot: Plotting Scatters for hit candidate genes for all provided sampled*

Description

As described before, scatter plots can be generated for all datasets. ‘carpools.hit.scatter‘ serves as a wrapper for ‘carpools.read.count.vs‘ and allows faster plotting for individual candidate genes or all overlapping candidate genes. It generated a pairs plot with the representation of all provided samples and highlights the candidate gene.

Usage

```
carpools.hit.scatter(wilcox=NULL, deseq=NULL, mageck=NULL, dataset, dataset.names = NULL,
namecolumn=1, fullmatchcolumn=2, title="Read Count", xlab="Readcount Dataset1",
ylab="Readcount Dataset2", labelgenes=NULL, labelcolor="orange",
extractpattern=expression("^(.+?)_.+"),
plotline=TRUE, normalize=TRUE, norm.function=median, offsetplot=1.2,
center=FALSE, aggregated=FALSE, type="enriched",
cutoff.deseq = 0.001, cutoff.wilcox = 0.05,
cutoff.mageck = 0.05, cutoff.override=FALSE, cutoff.hits=NULL,
plot.genes="overlapping", pch=16, col = rgb(0, 0, 0, alpha = 0.65))
```

Arguments

wilcox	Data output from 'stat.wilcox'. *Default* NULL *Values* Data output from 'stat.wilcox'.
deseq	Data output from 'stat.deseq'. *Default* NULL *Values* Data output from 'stat.deseq'.
mageck	Data output from 'stat.mageck'. *Default* NULL *Values* Data output from 'stat.mageck'.
cutoff.deseq	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.wilcox	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.mageck	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
dataset	A list of data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
fullmatchcolumn	In which column are the read counts? *Default* 2 *Values* column number (numeric)
dataset.names	A list of names that must be according to the list of data sets given in *dataset*. *Default* NULL *Value* NULL or list of data names (list)
norm.function	The mathematical function to normalize data. By default, the median is used. *Default* median *Values* Any mathematical function of R (function)
extractpattern	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)
cutoff.override	Shall the p-value threshold be ignored? If this is TRUE, the top percentage gene of 'cutoff.hits' is used instead. *Default* FALSE *Values* TRUE, FALSE

cutoff.hits	The percentatge of top genes being used if ‘cutoff.override=TRUE‘. *Default** NULL *Values* numeric
plot.genes	Defines what kind of data is used. By default, overlapping genes are highlighted in red color. *Default* "overlapping" *Values* "overlapping"
type	Defines whether all genes are plotted or only those being enriched or depleted. *Default* "all" *Values* "all", "enriched", "depleted"
labelgenes	For which gene shall the sgRNA effects being plotted? This expects a gene identifier or a vector of gene identifiers. *Default* NULL *Values* A gene identifier or vector of gene identifiers (character)
xlab	Label of X-Axis, only if ‘pairs=FALSE‘ *Default* "X-Axis" *Values* "Label of X-Axis" (character)
ylab	Label of Y-Axism only if ‘pairs=FALSE‘ *Default* "Y-Axis" *Values* "Label of Y-Axis" (character)
pch	The type of point used in the plot. See ‘?par()‘. *Default* 16 *Values* Any number describing the point, e.g. 16 (numeric)
col	The color of the plotted data. Can be any R color or RGB object. See ?rgb() for further information. *Default* rgb(0, 0, 0, alpha = 0.65) *Values* Any R color name or RGB color object (character OR color object)
plotline	You can draw additional lines indicating a fold change of 0, 2, 4. *Default* TRUE *Values** TRUE, FALSE (boolean)
normalize	Whether you would like to normalize read-counts first. Recommended if not done already. *Default* TRUE *Values* TRUE, FALSE (boolean)
offsetplot	Offetplot is used to stretch the x- and y-axis for nicer graphs. This will extend plotting area by offsetplot. *Default* 1.2 (Plotting area is streched to 1.2 times) *Values* any number (numeric)
center	If you like you can center your data within the plot. *Default* FALSE *Values* TRUE, FALSE (boolean)
aggregated	If you want to highlight genes, set this to true if you provide already aggregated gene read count instead of sgRNA read counts. *Default* FALSE *Values* TRUE, FALSE (boolean)
labelcolor	Color to highlight genes stated in ‘labelgenes‘. *Default* "organge" *Values* Any R color or RGB color object.
title	Title of the plot.

Details

none

Value

Return generic plots. See ?plot and ?pairs.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)

#Single Gene
plothitsscatter.enriched = carpools.hit.scatter(wilcox=data.wilcox,
deseq=data.deseq, mageck=data.mageck, dataset=list(TREAT1, TREAT2, CONTROL1, CONTROL2),
dataset.names = c(d.TREAT1, d.TREAT2, d.CONTROL1, d.CONTROL2),
namecolumn=1, fullmatchcolumn=2, title="Title", labelgenes="CASP8",
labelcolor="orange", extractpattern=expression("^(.+?)(_.+)"),
normalize=TRUE, norm.function=median, offsetplot=1.2, center=FALSE,
aggregated=FALSE, type="enriched", cutoff.deseq = 0.001,
cutoff.wilcox = 0.05, cutoff.mageck = 0.05, cutoff.override=FALSE,
cutoff.hits=NULL,  pch=16)

#Overlapping candidate genes

plothitsscatter.enriched = carpools.hit.scatter(wilcox=data.wilcox,
deseq=data.deseq, mageck=data.mageck, dataset=list(TREAT1, TREAT2, CONTROL1, CONTROL2),
dataset.names = c(d.TREAT1, d.TREAT2, d.CONTROL1, d.CONTROL2), namecolumn=1,
fullmatchcolumn=2, title="Title", labelgenes=NULL, labelcolor="orange",
extractpattern=expression("^(.+?)(_.+)"), normalize=TRUE, norm.function=median,
offsetplot=1.2, center=FALSE, aggregated=FALSE, type="enriched",
cutoff.deseq = 0.001, cutoff.wilcox = 0.05, cutoff.mageck = 0.05,
cutoff.override=FALSE, cutoff.hits=NULL,  pch=16)
```

---

carpools.hit.sgrna *Plotting sgRNA effects for all candidate genes or single genes*

Description

Since there is more than just one single sgRNA targeting your gene of interest, you can user caRpools to plot different sgRNA phenotype effects, e.g. the fold change or z-ratio, as desribed before in ‘carpools.raw.genes‘. A set of plots can be generated with ‘carpools.hit.sgrna‘, which serves as a wrapper for ‘carpools.raw.genes‘. By default, a foldchange plot as well as a violine plot are generated.

Usage

```
carpools.hit.sgrna(wilcox=NULL, deseq=NULL, mageck=NULL, dataset=NULL,
dataset.names = NULL, namecolumn=1, fullmatchcolumn=2,
norm.function=median, extractpattern=expression("^(.+?)_.+"),
put.names=TRUE, type="enriched", labelgenes=NULL, cutoff.deseq = 0.05,
cutoff.wilcox = 0.05, cutoff.mageck = 0.05, cutoff.override=FALSE,
plot.genes="overlapping", cutoff.hits=NULL,
plot.type=NULL, controls.target=NULL, controls.nontarget=NULL)
```

Arguments

wilcox	Data output from ‘stat.wilcox‘. *Default* NULL *Values* Data output from ‘stat.wilcox‘.
deseq	Data output from ‘stat.deseq‘. *Default* NULL *Values* Data output from ‘stat.deseq‘.
mageck	Data output from ‘stat.mageck‘. *Default* NULL *Values* Data output from ‘stat.mageck‘.
cutoff.deseq	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.wilcox	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.mageck	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
dataset	A list of data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
fullmatchcolumn	In which column are the read counts? *Default* 2 *Values* column number (numeric)
dataset.names	A list of names that must be according to the list of data sets given in *dataset*. *Default* NULL *Value* NULL or list of data names (list)
norm.function	The mathematical function to normalize data. By default, the median is used. *Default* median *Values* Any mathematical function of R (function)
extractpattern	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please

see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)

`cutoff.override`
: Shall the p-value threshold be ignored? If this is TRUE, the top percentage gene of 'cutoff.hits' is used instead. *Default* FALSE *Values* TRUE, FALSE

`cutoff.hits`
: The percentatge of top genes being used if 'cutoff.override=TRUE'. *Default** NULL *Values* numeric

`plot.genes`
: Defines what kind of data is used. By default, overlapping genes are highlighted in red color. *Default* "overlapping" *Values* "overlapping"

`type`
: Defines whether all genes are plotted or only those being enriched or depleted. *Default* "all" *Values* "all", "enriched", "depleted"

`labelgenes`
: For which gene shall the sgRNA effects being plotted? This expects a gene identifier or a vector of gene identifiers. If NULL, plots will be generated for all overlapping hit candidate genes. *Default* NULL *Values* A gene identifier or vector of gene identifiers (character)

`controls.target`
: If 'type="controls"', this is the gene identifier of the positive control. *Default* NULL *Value* Gene Identifier (character)

`controls.nontarget`
: If 'type="controls"', this is the gene identifier of the non-targeting control. *Default* "random" *Value* Gene Identifier (character)

`put.names`
: Do you want the sgRNA identifiers to be plotted? *Default* FALSE *Values* TRUE, FALSE

`plot.type`
: WHich kind of plot is to be drawn? If NULL, foldchange and violine plots are generated. *Default* NULL *Values* NULL, "foldchange", "z-score", "z-ratio", "vioplot"

Details

none

Value

Return generic plots according to 'type'.

By default, a foldchange plot as well as a violine plot are generated representing log2 fold changes of single sgRNAs.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)


sgrnas.en = carpools.hit.sgrna(wilcox=data.wilcox, deseq=data.deseq,
    mageck=data.mageck, dataset=list(CONTROL1, CONTROL2, TREAT1, TREAT2),
    dataset.names = c(d.CONTROL1, d.CONTROL2, d.TREAT1, d.TREAT2), namecolumn=1,
    fullmatchcolumn=2, norm.function=median, extractpattern=expression("^(.+?)(_.+)"),
    put.names=TRUE, type="enriched", labelgenes="CASP8", plot.type=NULL,
    cutoff.deseq = 0.001, cutoff.wilcox=0.05, cutoff.mageck = 0.05,
   cutoff.override=FALSE, cutoff.hits=NULL, controls.target="CASP8", controls.nontarget="random")
```

`carpools.hitident` *Visualization of hit analysis performed by Wilcox, DESeq2 and MAGeCK*

Description

The output from ‘stat.wilcox‘, ‘stat.DEseq‘ and ‘stat.mageck‘ can be visualized with ‘carpools.hitident‘. In this case, log2 fold changes are plotted against the gene names for all methods as well as the number of significant sgRNAs for data analyzed with DESeq2 or MAGeCK.

Usage

```
carpools.hitident(data, type="deseq2", title="DESeq2 plot", print.names=FALSE,
cutoff=c(0,0,0,0), inches=0.1, offsetplot=1.2, plot.p=0.01, sgRNA.top=1, separate=FALSE)
```

Arguments

`data`	Output data from either ‘stat.wilcox‘, ‘stat.DEseq‘ or ‘stat.mageck‘. *Default* empty *Values* Output from either ‘stat.wilcox‘, ‘stat.DEseq‘ or ‘stat.mageck‘.

type	Which type of analysis method was used? *Default* deseq2 *Values* "wilcox", "deseq2", "mageck"
title	Title of the plot. *Default* "DESeq2 plot" *Values* (character)
print.names	Shall the names of significant or top candidates being plotted? *Default* FALSE *Values* TRUE, FALSE (boolean)
cutoff	A vector containing plotting cutoffs if 'print.names=TRUE'. c("top enriched", "top depleted", "most sgRNA enriched", "most sgRNA depleted"). *Default* c(0,0,0,0) *Values* Vector of length 4 (numeric)
inches	see '?par'. *Default* 0.1 *Values* (numeric)
offsetplot	Multiplication factor for stretching the plotting area to get a better plot experience. *Default* 1.2 *Values* > 1 (numeric)
plot.p	Which p-value shall be plotted and used for visualization? *Default* 0.05 *Values* (numeric)
sgRNA.top	For sgRNA plots, this indicates how many genes will be labeled (the top X genes). *Default* 1 *Values* (numeric, integer)
separate	Gene that showed enrichment can be plotted separately from those that have shown a depletion for better overview, works only for wilcox. *Default* FALSE *Values* TRUE, FALSE

Details

none

Value

carpools.hitident returns a generic plot, which can be passed on to any device.

Note

see ?plot for detailed plotting information.

Author(s)

Jan Winter

Examples

```
data(caRpools)


data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
```

```
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)

mageck.result = carpools.hitident(data.mageck, type="mageck",
title="MAGeCK", inches=0.1, print.names=TRUE, plot.p=0.05, offsetplot=1.2, sgRNA.top=1)

wilcox.result = carpools.hitident(data.wilcox, type="wilcox",
title="Wilcox", inches=0.1, print.names=TRUE, plot.p=0.05, offsetplot=1.2, sgRNA.top=1)
```

`carpools.raw.genes` *Plotting sgRNA phenotype effects of a given gene*

Description

CaRpools also allows you to visualize the phenotypic effects of sgRNA belonging to the same gene via ‘carpools.raw.genes‘. This includes plotting of sgRNA foldchanges, z-score, z-ratios or read-counts. Moreover, ‘type="vioplot"‘ will present fold change data in comparison to the whole dataset and controls.

Usage

```
carpools.raw.genes(untreated.list,treated.list, genes=NULL, namecolumn=1,
fullmatchcolumn=2, norm.function=median, extractpattern=expression("^(.+?)_.+"),
do.plot=TRUE, log=FALSE, put.names=FALSE, type="foldchange", controls.target= NULL,
controls.nontarget=NULL, sort=TRUE)
```

Arguments

`untreated.list`	A list of untreated sample data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames of the untreated samples
`treated.list`	A list of treated sample data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames of the treated samples
`namecolumn`	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
`fullmatchcolumn`	In which column are the read counts? *Default* 2 *Values* column number (numeric)
`norm.function`	The mathematical function to normalize data if ‘normalize=TRUE‘. By default, the median is used. *Default* median *Values* Any mathematical function of R (function)

`extractpattern` PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)

`do.plot` Whether a plot is drawn or only tabular output is returned. *Default* TRUE *Values* TRUE, FALSE (boolean)

`log` Plot in log-scale? *Default* FALSE *Values* TRUE, FALSE (boolean)

`put.names` Do you want the sgRNA identifiers to be plotted? *Default* FALSE *Values* TRUE, FALSE

`type` Provides different types. "foldchange" for log2 foldchange, "readcount" for read-count, "z-score" for Z-scores, "z-ratio" for a Z-ratio or "vioplot" for a log2 FC of sgRNA effects. *Default* "foldchange" *Values* "foldchange", "read-count", "z-score", "z-ratio", "vioplot"

`controls.target` Highlights the positive control in red color. *Default* NULL *Value* Gene Identifier (character)

`controls.nontarget` Highlights the non-targeting control in blue color. *Default* "random" *Value* Gene Identifier (character)

`sort` This leads to output sorted by foldchange or z-ratio instead of names. *Default* TRUE *Values* TRUE, FALSE

`genes` For which gene shall the sgRNA effect plots being generated?

Details

none

Value

Return either generic plots or tables.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

# Foldchange
p1 = carpools.raw.genes(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1, TREAT2), genes="CASP8", namecolumn=1,
```

```
  fullmatchcolumn=2, norm.function=median, extractpattern=expression("^(.+?)_.+"),
  do.plot=TRUE, log=FALSE, put.names=TRUE, type="foldchange" )

# Z-Ratio
p2 = carpools.raw.genes(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1, TREAT2), genes="CASP8", namecolumn=1,
  fullmatchcolumn=2, norm.function=median, extractpattern=expression("^(.+?)_.+"),
  do.plot=TRUE, log=FALSE, put.names=TRUE, type="z-ratio" )

# Read Count
p3 = carpools.raw.genes(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1, TREAT2), genes="CASP8", namecolumn=1,
  fullmatchcolumn=2, norm.function=median, extractpattern=expression("^(.+?)_.+"),
  do.plot=TRUE, log=FALSE, put.names=TRUE, type="readcount" )

# Violine plot
p4 = carpools.raw.genes(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1, TREAT2), genes="CASP8", namecolumn=1,
  fullmatchcolumn=2, norm.function=median, extractpattern=expression("^(.+?)_.+"),
  do.plot=TRUE, log=FALSE, put.names=TRUE, type="vioplot" )
```

`carpools.read.count.vs`

QC: Scatterplots of Read-Counts

Description

CaRpools also allows you to compare the readcount for different samples using 'carpools.read.count.vs'. By this, you can easily compare the screen and replicate performance as well as highlighting your non-targeting or positive controls. Moreover, you can highlight any gene as well. For details regarding all arguments and option see '?carpools.read.count.vs'.

Usage

```
carpools.read.count.vs(dataset, namecolumn=1, fullmatchcolumn=2, title="Read Count",
dataset.names = NULL, xlab="Readcount Dataset1", ylab="Readcount Dataset2", xlim=NULL,
ylim=NULL, pch=16, col = rgb(0, 0, 0, alpha = 0.65), labelgenes=NULL, labelcolor="red",
extractpattern=expression("^(.+?)_.+"), plotline=TRUE, normalize=TRUE,
norm.function=median, offsetplot=1.2, center=FALSE, aggregated=FALSE,
pairs=FALSE, type=NULL, plot.identify=FALSE, plot.log=TRUE)
```

Arguments

`dataset`	A list of data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames
`namecolumn`	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)

`fullmatchcolumn`
: In which column are the read counts? *Default* 2 *Values* column number (numeric)

`title`
: The title of the plot. *Default* "Read Count" *Values* "Any title" (character)

`dataset.names`
: A list of names that must be according to the list of data sets given in *dataset*. *Default* NULL *Value* NULL or list of data names (list)

`xlab`
: Label of X-Axis, only if 'pairs=FALSE' *Default* "X-Axis" *Values* "Label of X-Axis" (character)

`ylab`
: Label of Y-Axism only if 'pairs=FALSE' *Default* "Y-Axis" *Values* "Label of Y-Axis" (character)

`xlim`
: You can define the x-axis range being plotted, e.g. 'c(0,1)'. *Default* empty *Values* empty or a vector with the lower and upper limit.

`ylim`
: You can define the y-axis range being plotted, e.g. 'c(0,1)'. *Default* empty *Values* empty or a vector with the lower and upper limit.

`pch`
: The type of point used in the plot. See '?par()'. *Default* 16 *Values* Any number describing the point, e.g. 16 (numeric)

`col`
: The color of the plotted data. Can be any R color or RGB object. See ?rgb() for further information. *Default* rgb(0, 0, 0, alpha = 0.65) *Values* Any R color name or RGB color object (character OR color object)

`labelgenes`
: You can highlight certain genes within the plot. This expects a gene identifier or a fector of gene identifiers. *Default* NULL *Values* A gene identifier or vector of gene identifiers (character)

`labelcolor`
: Color to highlight genes stated in 'labelgenes'. *Default* "organge" *Values* Any R color or RGB color object.

`extractpattern`
: PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)

`plotline`
: You can draw additional lines indicating a fold change of 0, 2, 4. *Default* TRUE *Values** TRUE, FALSE (boolean)

`normalize`
: Whether you would like to normalize read-counts first. Recommended if not done already. *Default* TRUE *Values* TRUE, FALSE (boolean)

`norm.function`
: The mathematical function to normalize data if 'normalize=TRUE'. By default, the median is used. *Default* median *Values* Any mathematical function of R (function)

`offsetplot`
: Offetplot is used to stretch the x- and y-axis for nicer graphs. This will extend plotting area by offsetplot. *Default* 1.2 (Plotting area is streched to 1.2 times) *Values* any number (numeric)

`center`
: If you like you can center your data within the plot. *Default* FALSE *Values* TRUE, FALSE (boolean)

`aggregated`
: If you want to highlight genes, set this to true if you provide already aggregated gene read count instead of sgRNA read counts. *Default* FALSE *Values* TRUE, FALSE (boolean)

pairs	In the case of plotting all four data sets at once, you can use a pairs plot for easier overview (see '?pairs()'). *Default* FALSE *Values* TRUE, FALSE (boolean)
type	This indicates whether you would like to color all highlighted genes in either red ("enriched") or blue ("depleted") color according to the standrds in caRpools for plotting enriched or depleted genes after analysis. *Default* NULL *Values* NULL, "enriched", "depleted"
plot.identify	You can ask R to let you identify genes by clikcing on the dots in the graph. This only works if 'pairs=FALSE'. *Default* FALSE *Values* TRUE, FALSE (boolean)
plot.log	If all plots are created using log-transformed data. *Default* TRUE *Values* TRUE, FALSE (boolean)

Details

For generic plot arguments, see ?plot.

Value

plot.read.count.vs returns a basic plot.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

carpools.read.count.vs(dataset=list(TREAT1,CONTROL1),
dataset.names = c(d.TREAT1, d.CONTROL1),
  pairs=FALSE, namecolumn=1, fullmatchcolumn=2, title="", pch=16,
  normalize=TRUE, norm.function=median, labelgenes="random", labelcolor="blue",
  center=FALSE, aggregated=FALSE)

carpools.read.count.vs(dataset=list(TREAT1, TREAT2, CONTROL1, CONTROL2),
  dataset.names = c(d.TREAT1, d.TREAT2, d.CONTROL1, d.CONTROL2),
  pairs=TRUE, namecolumn=1, fullmatchcolumn=2, title="", pch=16,
  normalize=TRUE, norm.function=median,
  labelgenes="random", labelcolor="blue", center=FALSE, aggregated=FALSE)
```

carpools.read.depth *QC: Plot Sequencing Read Depth*

Description

You can also visualize the read depth of genes per sgRNA in order to check for sufficient sequencing depth using ‘carpools.read.depth‘. For further details see ‘?carpools.read.depth‘. You can either plot single dat samples or all four data samples at once.

Usage

```
carpools.read.depth(datasets, namecolumn=1, fullmatchcolumn=2, dataset.names=NULL,
extractpattern=expression("^(.+?)_.+"), col=rgb(0, 0, 0, alpha = 0.65), xlab="Genes",
ylab="Read Count per sgRNA", statistics=TRUE, labelgenes = NULL,
controls.target = controls.target,
controls.nontarget=controls.nontarget, labelcolor="orange", waterfall=FALSE)
```

Arguments

datasets	A list of data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
fullmatchcolumn	In which column are the read counts? *Default* 2 *Values* column number (numeric)
dataset.names	A list of names that must be according to the list of data sets given in *dataset*. *Default* NULL *Value* NULL or list of data names (list)
extractpattern	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)
col	The color of the plotted data. Can be any R color or RGB object. See ?rgb() for further information. *Default* rgb(0, 0, 0, alpha = 0.65) *Values* Any R color name or RGB color object (character OR color object)
xlab	Label of X-Axis *Default* "X-Axis" *Values* "Label of X-Axis" (character)
ylab	Label of Y-Axis *Default* "Y-Axis" *Values* "Label of Y-Axis" (character)
statistics	Whether basic stattistics will be shown in the plot. *Default* TRUE *Values* TRUE, FALSE (boolean)
labelgenes	You can highlight certain genes within the plot. This expects a gene identifier or a fector of gene identifiers. *Default* NULL *Values* A gene identifier or vector of gene identifiers (character)

`labelcolor` Color to highlight genes stated in ‘labelgenes‘. *Default* "organge" *Values* Any R color or RGB color object.

`controls.target` Highlights the positive control in red color. *Default* NULL *Value* Gene Identifier (character)

`controls.nontarget` Highlights the non-targeting control in blue color. *Default* "random" *Value* Gene Identifier (character)

`waterfall` You can either plot the read depth sorted by gene identifier (FALSE, default) or according to the read depth. *Default* FALSE *Values* TRUE, FALSE (boolean) s

Details

notes

Value

plot.read.depth returns a generic plot.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

carpools.read.depth(datasets = list(CONTROL1), namecolumn=1 ,fullmatchcolumn=2,
  dataset.names=list(d.CONTROL1), extractpattern=expression("^(.+?)_.+"),
  xlab="Genes", ylab="Read Count per sgRNA",statistics=TRUE, labelgenes = NULL,
  controls.target = "CASP8", controls.nontarget="random", waterfall=FALSE)
```

---

carpools.read.distribution

QC: Plot Readcount Distribution

Description

A distribution for NGS data readcount can be created by ‘carpools.read.distribution‘ to visualize how the data set is distributed. This allows to check for data skewness and to estimate the overall assay quality. For further details see ‘?carpools.read.distribution‘.

Usage

```
carpools.read.distribution(dataset,namecolumn=1, fullmatchcolumn=2, breaks="",
title="Title", xlab="X-Axis", ylab="Y-Axis",statistics=TRUE,
col=rgb(0, 0, 0, alpha = 0.65), extractpattern=expression("^(.+?)_.+"),
plotgene=NULL, type="distribution", logscale=TRUE)
```

Arguments

dataset	Data frame of read-count data as created by load.file(). *Default* none *Values* A data frame
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
fullmatchcolumn	In which column are the read counts? *Default* 2 *Values* column number (numeric)
breaks	Histogramm breaks see '?hist'. By default, will be calculated according to the dataset length. *Default* NULL *Values* (numeric)
title	Main title of plot *Default* "Title" *Values* "The title you want" (character)
xlab	Label of X-Axis *Default* "X-Axis" *Values* "Label of X-Axis" (character)
ylab	Label of Y-Axis *Default* "Y-Axis" *Values* "Label of Y-Axis" (character)
statistics	Whether basic stattistics will be shown in the plot. *Default* TRUE *Values* TRUE, FALSE (boolean)
col	The color of the plotted data. Can be any R color or RGB object. See ?rgb() for further information. *Default* rgb(0, 0, 0, alpha = 0.65) *Values* Any R color name or RGB color object (character OR color object)
extractpattern	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)
plotgene	You can only plot the read count distribution of sgRNAs belonging to a certain gene, which is given to the function via plotgene. *Default* NULL *Value* NULL or gene identifier (character)
type	You can plot either the read count distribution either as a normal histogram, or a box-and-whisker plot. *Default* "distribution" *Values* "distribution" to plot a histogram, or "whisker" to plot a whisker plot (character)
logscale	Indicates whether the read-count is plotted in a logarithmic scale. *Default* TRUE *Values* TRUE, FALSE (boolean)

Details

none

Value

plot.read.distribution return a generic plot, that can be passed on to any device.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

carpools.read.distribution(CONTROL1, fullmatchcolumn=2,breaks=200,
  title=d.CONTROL1, xlab="log2 Readcount", ylab="# sgRNAs",statistics=TRUE)

carpools.read.distribution(CONTROL1, fullmatchcolumn=2,breaks=200,
  title=d.CONTROL1, xlab="log2 Readcount", ylab="# sgRNAs",statistics=TRUE,
  type="whisker")
```

---

carpools.reads.genedesigns

QC: Plot representation of sgRNAs per gene

Description

Since in most cases several sgRNAs are used to target a gene, the information how many sgRNAs are present in the data for each gene is of interest to make sure the number of sgRNAs present is still sufficient. Typically, only few sgRNAs should get "lost" during the screening procedure, so that the full sgRNA coverage is maintained throughout the assay. The only exception would be drop-out screens with a stringent setup. The representation of sgRNAs per gene can be plotted using ‘carpools.reads.genedesigns‘. For further details see ‘?carpools.reads.genedesigns‘.

Usage

```
carpools.reads.genedesigns(dataset, namecolumn=1, fullmatchcolumn=2, title="Read Count",
xlab="Percentage of sgRNAs present", ylab="Number of Genes", agg.function=sum,
extractpattern=expression("^(.+?)_.+"), col = rgb(0, 0, 0, alpha = 0.65))
```

Arguments

dataset	A data frame of read-count data as created by load.file(). *Default* none *Values* Adata frame
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)

`fullmatchcolumn`
: In which column are the read counts? *Default* 2 *Values* column number (numeric)

`title`
: The title of the plot. *Default* "Read Count" *Values* "Any title" (character)

`xlab`
: Label of X-Axis *Default* "X-Axis" *Values* "Label of X-Axis" (character)

`ylab`
: Label of Y-Axis *Default* "Y-Axis" *Values* "Label of Y-Axis" (character)

`agg.function`
: The function to aggregate sgRNA read-count. *Default* sum *Values* any mathematical function (function)

`extractpattern`
: PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)

`col`
: The color of the plotted data. Can be any R color or RGB object. See ?rgb() for further information. *Default* rgb(0, 0, 0, alpha = 0.65) *Values* Any R color name or RGB color object (character OR color object)

Details

none

Value

carpools.reads.genedesigns returns a generic plot.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

control1.readspergene = carpools.reads.genedesigns(CONTROL1, namecolumn=1, fullmatchcolumn=2,
title=paste("sgRNA Represenation:", d.CONTROL1, sep=" "),
xlab="Percentage of sgRNAs present", ylab="# of Genes")
```

carpools.sgrna.table *Table Output of sgRNA effect and Target Sequence*

Description

Since there is more than just one single sgRNA targeting your gene of interest, you can user caRpools to plot different sgRNA phenotype effects, e.g. the fold change or z-ratio, as desribed before in ‘carpools.raw.genes‘. In addition to that, caRpools also generated a tabular view which includes the log2 fold change as well as the target sequence, so that the user can directly pick the target sequence of the sgRNA he or shew wants.

This function, ‘carpools.sgrna.table‘ is best combined with ‘carpools.raw.genes‘ to give a fast overview of the sgRNA performance.

Usage

```
carpools.sgrna.table (wilcox=NULL, deseq=NULL, mageck=NULL, dataset=NULL,
dataset.names = NULL, namecolumn=1, fullmatchcolumn=2, norm.function=median,
extractpattern=expression("^(.+?)_.+"), type="enriched", cutoff.deseq = 0.05,
cutoff.wilcox = 0.05, cutoff.mageck = 0.05,
cutoff.override=FALSE, plot.genes="overlapping", cutoff.hits=NULL, sgrna.file=NULL,
labelgenes=NULL, write=FALSE, datapath=getwd(), analysis.name="Screen")
```

Arguments

wilcox	Data output from ‘stat.wilcox‘. *Default* NULL *Values* Data output from ‘stat.wilcox‘.
deseq	Data output from ‘stat.deseq‘. *Default* NULL *Values* Data output from ‘stat.deseq‘.
mageck	Data output from ‘stat.mageck‘. *Default* NULL *Values* Data output from ‘stat.mageck‘.
cutoff.deseq	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.wilcox	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.mageck	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
dataset	A list of data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
fullmatchcolumn	In which column are the read counts? *Default* 2 *Values* column number (numeric)

`dataset.names` A list of names that must be according to the list of data sets given in *dataset*. *Default* NULL *Value* NULL or list of data names (list)

`norm.function` The mathematical function to normalize data. By default, the median is used. *Default* median *Values* Any mathematical function of R (function)

`extractpattern` PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)

`cutoff.override` Shall the p-value threshold be ignored? If this is TRUE, the top percentage gene of 'cutoff.hits' is used instead. *Default* FALSE *Values* TRUE, FALSE

`cutoff.hits` The percentatge of top genes being used if 'cutoff.override=TRUE'. *Default** NULL *Values* numeric

`plot.genes` Defines what kind of data is used. By default, overlapping genes are highlighted in red color. *Default* "overlapping" *Values* "overlapping"

`type` Defines whether all genes are plotted or only those being enriched or depleted. *Default* "all" *Values* "all", "enriched", "depleted"

`sgrna.file` This is the library reference file loaded via 'load.file' providing the sgRNA target sequence. *Default* NULL *Values* object from 'load.file'

`labelgenes` For which gene shal the sgRNA effects being generated? This expects a gene identifier or a fector of gene identifiers. *Default* NULL *Values* A gene identifier or vector of gene identifiers (character)

`write` If you want to write directly to a file, this must be TRUE. Leave FALSE if you want the function to return a table. *Default* FALSE *Values* TRUE, FALSE

`datapath` If 'write=TRUE', this is the directory the file is written. *Default* getwd() *Values* absolute path

`analysis.name` The name of the file if 'write=TRUE' *Default* "Screen" *Values* any file name (character)

Details

Output is a table or file (if write=TRUE).

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)

sgrnas.en.table = carpools.sgrna.table(wilcox=data.wilcox, deseq=data.deseq,
    mageck=data.mageck, dataset=list(CONTROL1, CONTROL2, TREAT1, TREAT2),
    dataset.names = c(d.CONTROL1, d.CONTROL2, d.TREAT1, d.TREAT2), namecolumn=1,
    fullmatchcolumn=2, norm.function=median, extractpattern=expression("^(.+?)(_.+)"),
    type="enriched", labelgenes="CASP8", cutoff.deseq = 0.001, cutoff.wilcox=0.05,
    cutoff.mageck = 0.05, cutoff.override=FALSE, cutoff.hits=NULL, sgrna.file = libFILE,
    write=FALSE)

knitr::kable(sgrnas.en.table)
```

---

carpools.waterfall.pval

Visualization of p-value distribution

Description

Each of the analysis methods returns an adjusted p-value (corrected for multiple testing) as well as a fold change (Wilcox, DESeq2) or gene rank (MAGeCK). Therefore the -log10 p-value can be plotted against the gene names with ‘carpools.waterfall.pval‘:

Usage

```
carpools.waterfall.pval (type=NULL,dataset=NULL, pval=0.05, mageck.type="pos", log=TRUE)
```

Arguments

type	This indicates which kind of analysis method was used for p-value calculation. *Default* NULL *Values* "mageck", "deseq2", "wilcox"

dataset	Result from either ‘stat.wilcox‘, ‘stat.DEseq‘ or ‘stat.mageck‘. *Default* NULL *Values* Result from either ‘stat.wilcox‘, ‘stat.DEseq‘ or ‘stat.mageck‘
pval	The significance value set for the analysis which is to be plotted. *Default* 0.05 *Values* numeric
mageck.type	Only for plotting p-value calculate by MAGeCK. Indicates whether enriched ("pos") or depleted ("neg") genes are used. *Default* "pos" *Values* "pos", "neg"
log	-log10 of the p-valiues is plotted if set to TRUE. *Default* TRUE *Values* TRRUE, FALSE (boolean)

Value

Return a generic plot.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)

carpools.waterfall.pval(type="mageck", dataset=data.mageck, pval=0.05, log=TRUE)
```

---

check.caRpools *Test caRpools installation and dependent software*

Description

You can verify that the MIACCS.xls file as well as the used template file and all necessary scripts are found by calling ‘check.caRpools()‘. CaRpools also uses MAGeCK to look for enriched or depleted genes within your screening data. Please note that MAGeCK needs to be installed correctly, this can be tested by ‘check.caRpools‘.

Usage

```
check.caRpools(packages=TRUE, files=TRUE, mageck=TRUE, bowtie2=TRUE,
pandoc=TRUE, skip.updates=TRUE, template=NULL, scripts=TRUE, miaccs="MIACCS.xls")
```

Arguments

packages	if TRUE, packages will be checked using load.packages()
files	If TRUE, MIACCS as well as data and scripts folder will be checked in addition to CRISPR-mapping.pl and CRISPR-extract.pl.
mageck	If TRUE, mageck installation is checked.
bowtie2	if TRUE, bowtie2 installation is checked.
pandoc	if TRUE, pandoc installation is checked.
skip.updates	if TRUE, updates are skipped during package check.
template	Rmd template file name to use.
scripts	if TRUE, checks for perl scripts CRISPR-mapping and CRISPR-extract.pl.
miaccs	Filename of MIACCS file. Will be checked for proper loading.

Details

none

Value

This function does not return any value.

Note

none

Author(s)

Jan Winter

Examples

```
#check.caRpools()
```

---

compare.analysis *Exporting Hit Candidate Gene Information*

Description

Although the candidate lists of each analysis method can be saved separately, caRpools offer a comparative approach, which creates tables that include the information from all analysis methods at once for a faster overview.

This is done using the function ‘compare.analysis‘, which offers not only ouptu for Venn Diagrams, but also for tables.

Usage

```
compare.analysis(wilcox=NULL, deseq=NULL, mageck=NULL, type="enriched",
cutoff.deseq = NULL, cutoff.wilcox = NULL, cutoff.mageck = NULL,
cutoff.override=TRUE, cutoff.hits=5, output="list",
sort.by=c("mageck","pval","fdr"), plot.method=c("wilcox","mageck", "deseq"),
plot.feature=c("pval","fdr","pval"), pch=16)
```

Arguments

wilcox	Data output from 'stat.wilcox'. *Default* NULL *Values* Data output from 'stat.wilcox'.
deseq	Data output from 'stat.deseq'. *Default* NULL *Values* Data output from 'stat.deseq'.
mageck	Data output from 'stat.mageck'. *Default* NULL *Values* Data output from 'stat.mageck'.
type	Either enriched or depleted.
cutoff.deseq	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.wilcox	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.mageck	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.override	Shall the p-value threshold be ignored? If this is TRUE, the top percentage gene of 'cutoff.hits' is used instead. *Default* FALSE *Values* TRUE, FALSE
cutoff.hits	The percentatge of top genes being used if 'cutoff.override=TRUE'. *Default** NULL *Values* numeric
output	Three different types of output can be generated: A list with all genes including the information from 'stat.wilcox', 'stat.DEseq' and 'stat.mageck', a sorted ranked output, a venn diagram compatible output and 3D scatterplot. *Default* "list" *Values* "list", "rank", "venn", "3dplot"
sort.by	This indicates the sorting for 'type="rank" and type="list"' and is a vector. By default, data is sorted by the FDR of MAGeCK. needs to be a vector. *Default* c("mageck","fdr","fdr") *Values* c("mageck","fdr","fdr"), c("mageck","fdr","rank"), c("mageck","fdr","rank"), c("wilcox","pval","pval"), c("wilcox","pval","genes"), c("deseq","pval","pval"), c("deseq","pval","genes")
plot.method	Used only if 'type="3dplot"'. This indicates what is plotted at the x, y and z-axis and thus needs to be a vector of length 3. *Default* c("wilcox","mageck", "deseq"), plots wilcox on X-axis, mageck on y-axis and deseq on z-axis *Values* c("wilcox","mageck", "deseq") or any other combination
plot.feature	If 'type="3dplot"', this indicates the type of data plotted on each axis of the 3d plot. This can only be set according to the features available of the method used to be plotted as indicated in 'plot.method'. *Default* c("pval","fdr", "pval") which uses the p-value of wilcox, the fdr or MAGeCK and p-value of DESeq2. *Values* c("pval","fdr", "pval"), or ANY combination according to 'plot.method'

pch The type of point used in the plot. See '?par()'. *Default* 16 *Values* Any number describing the point, e.g. 16 (numeric)

Details

none

Value

Returns a table with information.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)

# Perform the comparison
data.analysis.enriched = compare.analysis(wilcox=data.wilcox,
    deseq=data.deseq, mageck=data.mageck, type="enriched",
    cutoff.override = FALSE, cutoff.hits=NULL, output="list",
    sort.by=c("mageck","fdr","rank"))
## Write to a file
xlsx::write.xlsx(data.analysis.enriched,
    file="COMPARE-HITS.xls",
    sheetName="Enriched")
# Print to console
knitr::kable(data.analysis.enriched[1:10,c(2:7)])
```

CONTROL1 *Read-count data for untreated sample, replicate 1*

Description

Replciate 1 of untreated sample

Usage

```
CONTROL1
```

Format

data frame

CONTROL1.g *Read-count data for untreated sample, replicate 1*

Description

Replciate 1 of untreated sample. Aggregated by sum to gene level.

Usage

```
CONTROL1.g
```

Format

data frame

CONTROL2 *Read-count data for untreated sample, replicate 2*

Description

Replciate 2 of untreated sample

Usage

```
CONTROL2
```

Format

data frame

CONTROL2.g — *Read-count data for untreated sample, replicate 2*

Description

Replciate 2 of untreated sample. Aggregated by sum to gene level.

Usage

```
CONTROL2.g
```

Format

data frame

d.CONTROL1 — *Name of Read-count data for untreated sample, replicate 1*

Description

Name of Replciate 1 of untreated sample

Usage

```
d.CONTROL1
```

Format

data frame

d.CONTROL2 — *Name of Read-count data for untreated sample, replicate 2*

Description

name of Replciate 2 of untreated sample

Usage

```
d.CONTROL2
```

Format

character

d.TREAT1 *Name of Read-count data for treated sample, replicate 1*

Description

Name of Replicate 1 of treated sample

Usage

```
d.TREAT1
```

Format

character

d.TREAT2 *Name of Read-count data for treated sample, replicate 2*

Description

Name of Replciate 2 of treated sample

Usage

```
d.TREAT2
```

Format

character

data.extract *Extracting sgRNA information from NGS FASTQ files to create read-count files for caRpools Analysis*

Description

CaRpools offers two ways of providing CRISPR/Cas9 screening data. Either raw **read-count files** are directly used as described before, or read-count files are generated from NGS FASTQ files by extracting the 20 nt target sequence, mapping it against a reference library and extracting the read-count information for each sgRNA identifier.

In a first step, NGS FASTQ data is extracted and mapped against a reference library file using bowtie2.

Usage

```
data.extract(scriptpath=NULL, datapath=NULL, fastqfile=NULL, extract = FALSE,
pattern = "default", machinepattern = "default", createindex = FALSE,
referencefile = NULL, mapping = FALSE, reversecomplement = FALSE,
threads = 1, bowtieparams = "", sensitivity = "very-sensitive-local", match = "perfect")
```

Arguments

scriptpath	Absolute path of the folder that contains 'CRISPR-extract.pl' and 'CRISPR-mapping.pl' *Default* NULL *Values* absolute path (character)
datapath	Absolute path of the folder that contains the data files (e.g. file.FASTQ) *Default* NULL *Values* absolute path (character)
fastqfile	Filename of FASTQ file WITHOUT .fastq extension *Default* NULL *Values* filename (character)
extract	Whether CRISPR-extract.pl is used to extract the 20 nt target sequence from the NGS reads using 'pattern' *Default* FALSE *Values* TRUE, FALSE (boolean)
pattern	PERL regular Expression to extract 20 nt target sequence from NGS reads. Please see *extract pattern* in this manual for more information. *Default* Regular Expression (character)
machinepattern	Maschine ID of your Sequencing maschine. Used ot identify the read id.
createindex	Do you want caRpools to generate a bowtie2 index? Only necessary if 'mapping=TRUE'. *Default* FALSE *Values* TRUE, FALSE
referencefile	Filename of the library reference FASTA file, without extension. Is the same as bowtie2 file, if 'createindex=TRUE'.
mapping	Indicates whether FASTQ files need to be mapped against 'referencefile'/'bowtie2file'. FALSE by default. *Default* FALSE *Values* TRUE, FALSE
reversecomplement	Is the NGS sequence in reverse complement order? *Default* FALSE *Values* TRUE, FALSE
threads	How many threads can bowtie2 use for mapping? Only used if 'mapping=TRUE'. Usually cores of CPU. *Default* 2 *Values* any integer
bowtieparams	If you want to pass additional parameters to bowtie2.
sensitivity	You can djust the sensitivity of bowtie2 using this parameter. By default, bowtie2 is used in a very-sensitive-local setting. More information about different sensitivy parameters can be found at the [bowtie2 options](http://bowtie-bio.sourceforge.net/bowtie2/manual.sl *Default* "very-sensitive-local" *Other options: very-fast, fast, sensitive, very-fast-local, fast-local, sensitive-local*
match	After bowtie2 mapping, the aligment is converted into read count files *filename_extracted-design.txt* and *filename_extracted-genes.txt*. You can indiciate how well the alignment must be in order to be used for generating the read count for each sgRNA. By default, this is set to *perfect*, which only employs a mapped read if the full 20 nt from the sequencing match perfectly to the sgRNA found in your library reference. The following options can be used: * __perfect__ - Read is used of all 20 nt from the sequencing are matching the target sequence given in the library reference * __high__ - Read is used if at least

18 nt (starting from the PAM) are matching the target sequence in the reference * __seed__ - Read is used if at least 14 nt (starting from the PAM) are a perfect match against the target sequence in the reference

Details

none

Value

Returns file name for load.file(). Generated additional read-count files.

Note

Needs bowtie2 and PERL working. use check.caRpools() first.

Author(s)

Jan Winter

Examples

```
data(caRpools)
# fileCONTROL1 = data.extract(scriptpath="path.to.scripts",
# datapath="path.to.FASTQ", fastqfile="filename1", extract=TRUE,
# seq.pattern, maschine.pattern, createindex=TRUE,
# bowtie2file=filename.lib.reference, referencefile="filename.lib.reference",
# mapping=TRUE, reversecomplement=FALSE, threads, bowtieparams,
#sensitivity="very-sensitive-local",match="perfect")
# Now we can load the generated Read-Count file directly!
#CONTROL1 = load.file(paste(datapath, fileCONTROL1, sep="/")) # Untreated sample 1 loaded

# Don't forget the library reference
# libFILE = load.file( paste(datapath, paste(referencefile,".fasta",sep=""), sep="/"),
# header = FALSE, type="fastalib")
```

`final.table` *CaRpools: Generating Table with Analysis Information from all Methods*

Description

CaRpools also provides you with a final gene table, which includes p-values, fold changes and ranks by all methods in a single tabular output. This output is **unbiased** and can thus be used for further analysis and data visualization. It takes the output generated by each analysis method, ‘stat.wilcox‘, ‘stat.DEseq‘ and ‘stat.mageck‘ and combines it into a single tabular representation.

Usage

```
final.table(wilcox=NULL, deseq=NULL, mageck=NULL, dataset, namecolumn=1,
norm.function=median, type="genes", extractpattern = expression("^(.+?)_.+"))
```

Arguments

wilcox	Data output from 'stat.wilcox'. *Default* NULL *Values* Data output from 'stat.wilcox'.
deseq	Data output from 'stat.deseq'. *Default* NULL *Values* Data output from 'stat.deseq'.
mageck	Data output from 'stat.mageck'. *Default* NULL *Values* Data output from 'stat.mageck'.
dataset	data.frame as created by 'load.file' *Default* empty *Values* data frame
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
extractpattern	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)
norm.function	The mathematical function to normalize data if 'normalize=TRUE'. By default, the median is used. *Default* median *Values* Any mathematical function of R (function)
type	Output generated. *Default* "genes" *Values* "genes"

Details

none

Value

Returns a data.frame of gene names and all information generated by stat.wilcox, stat.DEseq and stat.mageck.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"), mageckfolder=NULL,
sort.criteria="neg", adjust.method="fdr", filename = "TEST" , fdr.pval = 0.05)

final.tab = final.table(wilcox=data.wilcox, deseq=data.deseq,
mageck=data.mageck, dataset=CONTROL1.g, namecolumn=1, type="genes")
knitr::kable(final.tab[1:20,])
```

`gene.remove` *Remove gene information from sgRNA data.frame*

Description

This function is used to remove genes/gene information from a data.frame containing pooled CRISPR screen data. It is meant to exclude genes from the analysis and removes all entries belonging to a gene from the sgRNA data.frame.

Usage

```
gene.remove(data, namecolumn = 1, toremove = NULL,
extractpattern = expression("^(.+?)_.+") )
```

Arguments

`data`	data.frame with sgRNA readcounts. Must have one column with sgRNA names and one column with readcounts. Please note that the data must be formatted in a way, that gene names are included within the sgRNA name and can be extracted using the extractpattern expression. e.g. GENE_sgRNA1 -> GENE as gene name, _ as the separator and sgRNA1 as the sgRNA identifier.
`namecolumn`	integer, indicates in which column the names are stored
`toremove`	Vector of gene names that will be removed from sgRNA dataset. The gene name must be included in the sgRNA names in order to be extracted using the pattern defined in extractpattern. e.g. c=("gene1","gene2")

`extractpattern` Regular Expression, used to extract the gene name from the sgRNA name. Please make sure that the gene name extracted is accesible by putting its regular expression in brackets (). The default value expression("^(.+?)_.+") will look for the gene name (.+?) in front of the separator _ and any character afterwards .+ e.g. gene1_anything .

Details

In a table with

DesignID	fullmatch
AAK1_104_0	0
AAK1_105_0	197
AAK1_106_0	271
AAK1_107_0	1
AAK1_108_0	0

calling gene.remove(data.frame, toremove="AAK1", extractpattern = expression("^(.+?)_.+")) will remove all entries shown above, since AAK1 is the gene name, separated by an undescore _ from the sgRNA identifier.

Value

gene.remove returns a data.frame that has the same column dimensions as the input data.frame, however all rows in which toremove=gene is present, are deleted.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
gene.remove(CONTROL1, toremove="AAK1", extractpattern = expression("^(.+?)_.+"))
```

`generate.hits` *Retrieving overlapping hits from caRpools analysis*

Description

CaRpools can also calculate which genes overlapped in all hit analysis methods using 'generate.hits'.

Usage

```
generate.hits(wilcox=NULL, deseq=NULL, mageck=NULL,  type="enriched",
cutoff.deseq = 0.001, cutoff.wilcox = 0.05, cutoff.mageck = 0.05,
cutoff.override=FALSE, cutoff.hits=NULL, plot.genes="overlapping")
```

Arguments

wilcox	Data output from 'stat.wilcox'. *Default* NULL *Values* Data output from 'stat.wilcox'.
deseq	Data output from 'stat.deseq'. *Default* NULL *Values* Data output from 'stat.deseq'.
mageck	Data output from 'stat.mageck'. *Default* NULL *Values* Data output from 'stat.mageck'.
cutoff.deseq	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.wilcox	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.mageck	P-Value threshold used to determine significance. *Default* 0.001 *Values* numeric
cutoff.override	Shall the p-value threshold be ignored? If this is TRUE, the top percentage gene of 'cutoff.hits' is used instead. *Default* FALSE *Values* TRUE, FALSE
cutoff.hits	The percentatge of top genes being used if 'cutoff.override=TRUE'. *Default** NULL *Values* numeric
plot.genes	Defines what kind of data is returned, by default only overlapping genes or MAGeCK. *Default* "overlapping" *Values* "overlapping"
type	Defines whether all genes are plotted or only those being enriched or depleted. *Default* "all" *Values* "all", "enriched", "depleted"

Details

none

Value

generate.hits return a vector with overlapping candidate genes from all analysis methods.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)

overlap.enriched = generate.hits(wilcox=data.wilcox, deseq=data.deseq,
mageck=data.mageck, type="enriched", cutoff.deseq = 0.001, cutoff.wilcox = 0.05,
cutoff.mageck = 0.05, cutoff.override=FALSE, cutoff.hits=NULL, plot.genes="overlapping")
print(overlap.enriched)
```

---

get.gene.info *Retrieving Gene Annotation and Gene Identifier Conversion from BiomaRt*

Description

It is also possible to either enrich the screening dataset file with additional information provided by the biomaRt interface. For example, gene identifiers can be changed from EnsemblIDs to official gene symbols are Gene Ontology terms can be added to the dataset. This can be done using ‘get.gene.info‘, which serves as a wrapper for the **biomaRt** package with its load of options and possibilities (more information see ‘?biomaRt‘).

You can convert any gene identifier which is included in your sgRNA identifer to e.g. EnsemblID or HGNC Gene Symbol using caRpools. **Please note that Internet Access is required for biomaRt.** For further information about biomaRt conversion, please see the [biomaRt Manual](www.bioconductor.org/packages/release/bioc/vignettes/biomaRt/inst/doc/biomaRt.pdf).

Usage

```
get.gene.info(data, namecolumn=1, extractpattern=expression("^(.+?)(_.+)"),
host="www.ensembl.org", database="ENSEMBL_MART_ENSEMBL", dataset="hsapiens_gene_ensembl",
filters="ensembl_gene_id", attributes = c("hgnc_symbol"),
return.val = "dataset", controls=FALSE)
```

Arguments

data	Data frame that contains read-count data. *Default* none *Values* data.frame containing read-count data (data.frame)
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
extractpattern	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)
host	Host used to retrieve biomaRt information. By default, host is set to www.ensembl.org.
database	BiomaRt database to be used. See '?listMarts()' or biomaRt documentation. *Default* "ENSEMBL_MART_ENSEMBL", is using the ensembl database *Values* Any biomaRt database (character)
dataset	The biomaRt dataset to be used. For *homo sapiens*, *hsapiens_gene_ensembl* is recommended. See '?listDatasets' or biomaRt documentation. *Default* "hsapiens_gene_ensembl" *Values* Any biomaRt dataset (character)
filters	The input filter information to retrieve biomaRt annotation, usually is the type of gene identifier used in the read-count files, e.g. "ensemble_gene_id". see '?listFilters' *Default* "ensembl_gene_id" *Values* Any biomaRt filter (character)
attributes	The output attribute to retrieve from biomaRt, usually the annotations that need to be fetched, e.g. "hgnc_symbol". see '?listAttributes' *Default* "hgnc_symbol" *Values* Any biomaRt attribute (character)
return.val	The type of object that is returned. For whole dataset, e.g. conversion of gene identifiers, use "dataset". *Default* "dataset" *Values* "dataset" (will give back the same data frame, but with exchanged gene identifiers), "info" (will return a data frame with all attributes fetched for genes, is used to annotate gene with additional information)
controls	Is set to TRUE if 'data' is not a data frame, but a vector. *Default* FALSE *Values* TRUE, FALSE (boolean)

Details

none

Value

Return either a data.frame with converted gene identifier or a data frame with annotations.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
#CONTROL1.replaced = get.gene.info(CONTROL1, namecolumn=1,
#extractpattern=expression("^(.+?)(_.+)"), host="www.ensembl.org",
#database="ensembl", #dataset="hsapiens_gene_ensembl",
#filters="hgnc_symbol",attributes = c("ensembl_gene_id"),
#return.val = "dataset")

#knitr::kable(CONTROL1.replaced[1:10,])

#CONTROL1.replaced.info = get.gene.info(CONTROL1, namecolumn=1,
#extractpattern=expression("^(.+?)(_.+)"), database="ensembl",
#dataset="hsapiens_gene_ensembl", filters="hgnc_symbol",
#attributes = c("ensembl_gene_id","description"), return.val = "info")

#knitr::kable(CONTROL1.replaced.info[1:10,])
```

---

libFILE *FASTA file containing als sgRNA target sequences and identifiers. USed for mapping and sgRNA table.*

Description

FASTA file containing als sgRNA target sequences and identifiers. USed for mapping and sgRNA table.

Usage

```
libFILE
```

Format

data frame

---

load.file *Load sgRNA NGS Data especially for caRpools*

Description

This function is a parser of read.table to load sgRNA NGS data into a data.frame

Usage

```
load.file(filename, header = TRUE, sep = "\t", comment.char="", type=NULL)
```

Arguments

filename	The filename of the NGS dataset file.
header	Specifies whether a header is present in the file or not.
sep	Specifies how data is separated column-wise. See ?read.table for further information.
comment.char	comment.char see ?read.table
type	Type of data being loaded. Bu default NULL, which loads tabular data. Other values: xlsx for MIACCS file and fastalib to read the library reference fasta file

Details

See ?read.table for further information.

Value

load.file returns a data.frame.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
#data.frame = load.file("sgRNA.txt", header= TRUE, sep="\t")
```

---

load.packages *Loading and Installing packages used for caRpools*

Description

This function is used to check for presence of all packages and install them if not.

Usage

```
load.packages(noupdate=TRUE)
```

Arguments

noupdate	Indicates whether packages will NOT be updated, by default TRUE.

Details

Is only used to check R packages

Value

load.packages does not give any return value, however it will give you errors if something is wrong.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
load.packages()
```

`referencefile` *Name of fasta reference file without extension.*

Description

Name of fasta reference file without extension.

Usage

```
referencefile
```

Format

character

stat.DESeq *Analysis: DESeq2 Analysis of pooled CRISPR NGS data*

Description

For the DESeq2 analysis implementation, the read counts of all sgRNAs for a given gene are first summed up to increase the available read count. Then, DESeq2 analysis is perfomed, which includes the estimation of size-factors, the variance stabilization using a parametric fit and a Wald-Test for differnece in log2 fold changes between the untreated and treated data. More information about this can be found in _Love et al._ [Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2](http://www.ncbi.nlm.nih.gov/pubmed/25516281) _Genome Biology_ 2014

Usage

```
stat.DESeq(untreated.list,treated.list,namecolumn=1, fullmatchcolumn=2,
agg.function=sum, extractpattern=expression("^(.+?)_.+"), sorting=FALSE,
sgRNA.pval = 0.01, filename.deseq="data", fitType="parametric", p.adjust="holm")
```

Arguments

untreated.list	A list of data.frames of untreated, control samples. e.g. list(df.control1, df.control2)
treated.list	A list of data.frames of treated samples. e.g. list(df.treated1, df.treated2)
namecolumn	In which the target names are located, e.g. namecolumn=1 for the first columns.
fullmatchcolumn	Column, in which readcounts are located, e.g. fullmatchcolumn=2 for the second column.
agg.function	Function used to aggregate gene data from individual sgRNA data. By default, agg.function=mean, but it can be any other function e.g. sum or median.
extractpattern	Regular Expression, used to extract the gene name from the sgRNA name. Please make sure that the gene name extracted is accesible by putting its regular expression in brackets (). The default value expression("^(.+?)_.+") will look for the gene name (.+?) in front of the separator _ and any character afterwards .+ e.g. gene1_anything .
sorting	Defines whether the final output is sorted by the calculated p-value. By default, sorting=FALSE will return a table sorted by gene name.
sgRNA.pval	p-value threshold to count significant sgRNAs for each gene. *Default* 0.001 *Value* (numeric)
filename.deseq	Filename of raw DESeq2 data output. *Default* "data" *Values* (character)
fitType	See '?DESeq2'. *Default* "parametric" *Values* "parametric", "local" "mean"
p.adjust	Method to adjust p-value for multiple testing. See '?DEseq2'. *Default* "holm" *Values* see '?DESeq2'

Details

none

Value

stat.DESeq returns a formal class that contains gene names including the calculated p-value. The returned class can be visualized using carpools.hitident (see ?carpools.hitident). The output is formatted as follows:

log2 fold change (MAP): condition untreated vs treated
Wald test p-value: condition untreated vs treated
DataFrame with 813 rows and 6 columns

	baseMean	log2FoldChange	lfcSE	stat	pvalue	padj
AAK1	73.90565	-0.23319491	0.2927459	-0.7965779	0.42569619	0.7018234
AATK	159.43350	-0.11312924	0.2740927	-0.4127408	0.67979655	0.8514905
ABI1	131.03013	-0.09915855	0.2693971	-0.3680758	0.71281670	0.8691949
ABL1	77.51711	0.07837768	0.3155477	0.2483862	0.80383562	0.9114121
ABL2	119.22621	-0.49412039	0.2846396	-1.7359507	0.08257254	0.3128525
...	...	...	...	...	...	...

Note

none

Author(s)

Jan Winter, DESEq2 was developed by the Wolfgang Huber lab (EMBL, Heidelberg)

Examples

```
data(caRpools)
data.deseq = stat.DESeq(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1,
  fullmatchcolumn=2, extractpattern=expression("^(.+?)(_.+)"),
  sorting=FALSE, filename.deseq = "ANALYSIS-DESeq2-sgRNA.tab",
  fitType="parametric")

knitr::kable(data.deseq$genes[1:10,])
```

`stat.mageck` *Analysis: Analysis of pooled CRISPR screening data using a MAGeCK*

Description

CaRpools also uses MAGeCK to look for enriched or depleted genes within your screening data. Please note that MAGeCK needs to be installed correctly, this can be tested by 'check.caRpools'.

Within this approach, the read counts of all sgRNAs in one dataset are first normalized by the function set in the MIACCS file. By default, normalization is done by read count division with the dataset median. Then, the fold change of each population of sgRNAs for a gene is tested against the population of either the non-targeting controls or randomly picked sgRNAs, as defined by the random picks option within the MIACCS file, using a two-sided Mann-Whitney-U test. P-values are corrected for multiple testing using FDR.

Usage

```
stat.mageck(untreated.list, treated.list, namecolumn=1, fullmatchcolumn=2,
norm.fun=median, extractpattern=expression("^(.+?)_.+"), mageckfolder=NULL,
sort.criteria="neg", adjust.method="fdr", filename=NULL, fdr.pval=0.05)
```

Arguments

`untreated.list`	A list of untreated sample data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames of the untreated samples
`treated.list`	A list of treated sample data frames of read-count data as created by load.file(). *Default* none *Values* A list of data frames of the treated samples
`namecolumn`	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
`fullmatchcolumn`	In which column are the read counts? *Default* 2 *Values* column number (numeric)
`extractpattern`	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)
`sort.criteria`	MAGeCK argument *–sort-criteria* *Default* "neg" *Values* see MAGeCK documentation
`mageckfolder`	Folder for MAGeCK raw data output (internally used). *Default* NULL *Value* (character)
`filename`	Filename of raw MAGeCK data output. *Default* "data" *Values* (character)
`adjust.method`	Method to adjust p-value for multiple testing. See MAGeCK documentation. *Default* "fdr" *Values* see MAGeCK documentation
`fdr.pval`	FDR used for correction. *Default* 0.05 *Values* (numeric)
`norm.fun`	The mathematical function to normalize data. By default, the median is used. *Default* median *Values* Any mathematical function of R (function)

Details

none

Value

stat.mageck retrieves a list of two data frames. One with gene information, the other with sgRNA information.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
data.mageck = stat.mageck(untreated.list = list(CONTROL1, CONTROL2),
treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
norm.fun="median", extractpattern=expression("^(.+?)(_.+)"),
mageckfolder=NULL, sort.criteria="neg", adjust.method="fdr",
filename = "TEST" , fdr.pval = 0.05)

knitr::kable(data.mageck$genes[1:10,])
```

---

stat.wilcox *Analysis: Analysis of pooled CRISPR screening data using a Wilcoxon Test*

Description

__Wilcox__

Within this approach, the read counts of all sgRNAs in one dataset are first normalized by the function set in the MIACCS file. By default, normalization is done by read count division with the dataset median. Then, the fold change of each population of sgRNAs for a gene is tested against the population of either the non-targeting controls or randomly picked sgRNAs, as defined by the random picks option within the MIACCS file, using a two-sided Mann-Whitney-U test. P-values are corrected for multiple testing using FDR.

Usage

```
stat.wilcox(untreated.list=list(NULL, NULL),treated.list=list(NULL, NULL),
namecolumn=1, fullmatchcolumn=2,normalize=TRUE,norm.fun=median,
extractpattern=expression("^(.+?)_.+"), controls=NULL, control.picks=300, sorting=TRUE)
```

Arguments

`untreated.list`	A list of data.frames of untreated, control samples. e.g. list(df.control1, df.control2)
`treated.list`	A list of data.frames of treated samples. e.g. list(df.treated1, df.treated2)
`namecolumn`	In which the target names are located, e.g. namecolumn=1 for the first columns.
`fullmatchcolumn`	Column, in which readcounts are located, e.g. fullmatchcolumn=2 for the second column.
`normalize`	Datasets can be normalized by norm.fun if normalize=TRUE.
`norm.fun`	The function used to normalize the datasets if normalize=TRUE. By default, normalization is done using the dataset median, but any other function e.g. mean, can be used in principle.
`extractpattern`	Regular Expression, used to extract the gene name from the sgRNA name. Please make sure that the gene name extracted is accesible by putting its regular expression in brackets (). The default value expression("^(.+?)_.+") will look for the gene name (.+?) in front of the separator _ and any character afterwards .+ e.g. gene1_anything .
`controls`	DSS requires a set of non-targeting sgRNAs (negative controls) within the datasets. You can specify the arbitrary gene name for these controls using controls="arbitrary.gene.name.of.controls
`sorting`	Analysis output is by default sorted by gene name (sorting=FALSE). If desired, the output table can be sorted according to the p-value of the genes (sorting=TRUE).
`control.picks`	If no non-targeting controls are present or set, wilcox will pick a randum number of sgRNAs from the data set as the alternative population. This is only used if 'controls=NULL'. *Default* 300 *Values* numeric

Value

stat.wilcox return a data.frame, which can be visualized by plot.hitident. The data.frame has the following format:

	untreated	treated	foldchange	p.value
AAK1	2.061346	3.007924	1.351672	0.2966311
AATK	3.413357	5.129985	1.398695	0.1146190
ABI1	2.997385	4.384881	1.418959	0.1437962
ABL1	2.269906	2.874087	1.211499	0.3681327
ABL2	2.519391	4.539583	1.732575	0.6335575

For each gene, the foldchange as well as the p-value, derived by the Mann-Whitney U test against the non-targeting controls, are listed.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)

data.wilcox = stat.wilcox(untreated.list = list(CONTROL1, CONTROL2),
  treated.list = list(TREAT1,TREAT2), namecolumn=1, fullmatchcolumn=2,
  normalize=TRUE, norm.fun=median, sorting=FALSE, controls="random",
  control.picks=NULL)

knitr::kable(data.wilcox[1:10,])
```

---

stats.data *Calculating data set statistics*

Description

General statistics for a given dataset can be obtained by 'stats.data'.

Usage

```
stats.data(dataset, namecolumn = 1, fullmatchcolumn = 2,
extractpattern=expression("^(.+?)_.+"), readcount.unmapped.total = NA,
controls.target = NULL, controls.nontarget = "random", type="stats")
```

Arguments

`dataset`	Data frame of read-count object. *Default* none *Values* data frame as created by 'load.file()'
`namecolumn`	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
`fullmatchcolumn`	In which column are the read counts? *Default* 2 *Values* column number (numeric)
`extractpattern`	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)
`readcount.unmapped.total`	Number of raw NGS reads, only used if 'type="mapping'. *Default* NA *Values* Number of raw reads (integer)

controls.target
: If 'type="controls"', this is the gene identifier of the positive control. *Default* NULL *Value* Gene Identifier (character)

controls.nontarget
: If 'type="controls"', this is the gene identifier of the non-targeting control. *Default* "random" *Value* Gene Identifier (character)

type
: Which type os statistic will be generated. *Default* "stats" *Values* "stats" will generate short statistics like median and mean for the data set, "mapping" will generate an overview of how many reads are present, "datatset" is used to generate in-depth statistics for each gene of a dataset, "controls" is used for in-depth statistics of the controls.

Details

none

Value

Returns different tabular outputs.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
U1.stats = stats.data(dataset=CONTROL1, namecolumn = 1, fullmatchcolumn = 2,
                      extractpattern=expression("^(.+?)_.+"), type="stats")

knitr::kable(stats.data(dataset=CONTROL1, namecolumn = 1, fullmatchcolumn = 2,
  extractpattern=expression("^(.+?)_.+"), readcount.unmapped.total = 1786217, type="mapping"))

knitr::kable(stats.data(dataset=CONTROL1, namecolumn = 1, fullmatchcolumn = 2,
  extractpattern=expression("^(.+?)_.+"), readcount.unmapped.total = 1786217,
  type="stats"))

knitr::kable(stats.data(dataset=CONTROL1, namecolumn = 1, fullmatchcolumn = 2,
  extractpattern=expression("^(.+?)_.+"), readcount.unmapped.total = 1786217,
  type="dataset")[1:10,1:5])
```

TREAT1 *Read-count data for treated sample, replicate 1*

Description

Replciate 1 of treated sample

Usage

```
TREAT1
```

Format

data frame

TREAT1.g *Read-count data for treated sample, replicate 1*

Description

Replciate 1 of treated sample. Aggregated by sum to gene level.

Usage

```
TREAT1.g
```

Format

data frame

TREAT2 *Read-count data for treated sample, replicate 2*

Description

Replciate 2 of treated sample

Usage

```
TREAT2
```

Format

data frame

TREAT2.g *Read-count data for treated sample, replicate 2*

Description

Replicate 2 of treated sample. aggregated by sum to gene level.

Usage

```
TREAT2.g
```

Format

data frame

unmapped.genes *sgRNAs without reads*

Description

CaRpools also provides you with the number of missing sgRNA, that means sgRNAs without a single read during NGS. If you want to know WHICH sgRNAs dropped out for a given gene, please consider using ‘genes‘ as an optional argument with the gene identifier of interest.

Usage

```
unmapped.genes(data, namecolumn=1, fullmatchcolumn=2,
genes=NULL, extractpattern=expression("^(.+?)_.+"))
```

Arguments

data	A data.frame as created by ‘load.file‘. *Default* empty *Values* read-count data.frame
namecolumn	In which column are the sgRNA identifiers? *Default* 1 *Values* column number (numeric)
fullmatchcolumn	In which column are the read counts? *Default* 2 *Values* column number (numeric)
genes	If you want to know how many sgRNAs are not present for a single gene, set ‘genes‘ to your gene identifier of interest. *Default* NULL *Values* gene identifier (character)
extractpattern	PERL regular expression that is used to retrieve the gene identifier from the overall sgRNA identifier. e.g. in **AAK1_107_0** it will extract **AAK1**, since this is the gene identifier beloning to this sgRNA identifier. **Please see: Read-Count Data Files** *Default* expression("^(.+?)(_.+)"), will work for most available libraries. *Values* PERL regular expression with parenthesis indicating the gene identifier (expression)

Value

Tabular output with number of missing sgRNAs for each gene or the name of the missing sgRNA if genes!=NULL.

Author(s)

Jan Winter

Examples

```
data(caRpools)
U1.unmapped = unmapped.genes(data=CONTROL1, namecolumn=1,
fullmatchcolumn=2, genes=NULL, extractpattern=expression("^(.+?)_.+"))

knitr::kable(U1.unmapped)

U1.unmapped = unmapped.genes(data=CONTROL1, namecolumn=1,
fullmatchcolumn=2, genes="random", extractpattern=expression("^(.+?)_.+"))

knitr::kable(U1.unmapped)
```

---

use.caRpools *Starting caRpools eport generation from R console*

Description

Moreover, caRpools report generation can also be initiated without R-studio installation, so that this can be done via R command line even on remote computers. In this case, caRpools report generation can be started via 'use.caRpools' with additional parameters, which are described below.

Usage

```
use.caRpools(type=NULL, file="CaRpools-extended-PDF.Rmd",
miaccs="MIACCS.xls", check=TRUE, work.dir=NULL)
```

Arguments

type	*Description* If you provide a custom Rmd template that can generate both, PDF and HTML reports you can indicate which version you want to generate. *Default* NULL *Values* "PDF", "HTML"
file	*Description* The file name of your custom Rmd template file (with extension). *Default* "CaRpools-extended-PDF.Rmd" *Values* filename as character
miaccs	*Description* The filename of your MIACCS file. *Default* "MIACCS.xls" *Values* filename as character
check	*Description* Indicates whether caRpools will check for correct installation and file access. *Default* TRUE *Values* TRUE or FALSE (boolean)

`work.dir` *Description* You can provide the absolute path to the working directory in which all files are placed (e.g. the MIACCS.xls and Rmd template). *Default* NULL *Values* absolute path (character) or NULL if standard R working directory is used

Details

none

Value

Start caRpools report generation, so no direct return value is generated.

Note

none

Author(s)

Jan Winter

Examples

```
data(caRpools)
#use.caRpools(check=FALSE)
```

Index