Package 'mSigTools'

August 18, 2022

Type Package

Title Mutational Signature Analysis Tools

Version 1.0.0

Description A package containing various utility functions for mutational signature analysis.

License GPL-3

URL https://github.com/Rozen-Lab/mSigTools

BugReports https://github.com/Rozen-Lab/mSigTools/issues

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.2.1

Imports clue, philentropy

Suggests spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

R topics documented:

`match_two_sig_sets` *Find an optimal matching between two sets of signatures subject to a maximum distance*

Description

Find an optimal matching between two sets of signatures subject to a maximum distance

Usage

```
match_two_sig_sets(
  x1,
  x2,
  method = "cosine",
  convert.sim.to.dist = function(x) {
     return(1 - x)
 },
  cutoff = 0.9
)
```

Arguments

`x1`	A numerical-matrix-like object with columns as signatures.
`x2`	A numerical-matrix-like object with columns as signatures. Needs to have the same number of rows as `x1`.
`method`	A character string that specifies a method for `distance`.
`convert.sim.to.dist`	If `method` specifies a similarity rather than a distance, use this function to convert the similarity to a distance.
`cutoff`	A maximum distance or minimum similarity over which to pair signatures between `x1` and `x2`.

Details

Match signatures between `x1` and `x2` using the function `solve_LSAP`, which uses the "Hungarian" (a.k.a "Kuhn–Munkres") algorithm https://en.wikipedia.org/wiki/Hungarian_algorithm, which optimizes the total cost associated with the links between nodes. The functions converts similarities to distances, and generates a distance matrix between the two sets of signatures. It sets distances > `cutoff` to very large values. It then applies `solve_LSAP` to the resulting matrix to compute a matching between `x1` and `x2` that minimizes the sum of the distances.

Examples

```
ex.sigs <- matrix(c(0.2, 0.8, 0.3, 0.7, 0.6, 0.4), nrow = 2)
colnames(ex.sigs) <- c("ex1", "ex2", "ex3")
gt.sigs <- matrix(c(0.21, 0.79, 0.19, 0.81), nrow = 2)
colnames(gt.sigs) <- c("gt1", "gt2")
match_two_sig_sets(ex.sigs, gt.sigs, cutoff = .9)
```

plot_exposure *Plot exposures in multiple plots each with a manageable number of samples*

Description

Plot exposures in multiple plots each with a manageable number of samples

Usage

```
plot_exposure(
  exposure,
  samples.per.line = 30,
  plot.proportion = FALSE,
  xlim = NULL,
  ylim = NULL,
  legend.x = NULL,
  legend.y = NULL,
  cex.legend = 0.9,
  cex.yaxis = 1,
  cex.xaxis = NULL,
  plot.sample.names = TRUE,
  yaxis.labels = NULL,
  ...
)
```

Arguments

exposure	Exposures as a numerical matrix (or data.frame) with signatures in rows and samples in columns. Rownames are taken as the signature names and column names are taken as the sample IDs. If you want exposure sorted from largest to smallest, use sort_exposure. Do not use column names that start with multiple underscores. The exposures will often be mutation counts, but could also be e.g. mutations per megabase.
samples.per.line	Number of samples to show in each plot.
plot.proportion	Plot exposure proportions rather than counts.
xlim, ylim	Limits for the x and y axis. If NULL(default), the function tries to do something reasonable.
legend.x, legend.y	The x and y co-ordinates to be used to position the legend.
cex.legend	A numerical value giving the amount by which legend plotting text and symbols should be magnified relative to the default.
cex.yaxis	A numerical value giving the amount by which y axis values should be magnified relative to the default.
cex.xaxis	A numerical value giving the amount by which x axis values should be magnified relative to the default. If NULL(default), the function tries to do something reasonable.

`plot.sample.names`
: Whether to plot sample names below the x axis. Default is TRUE.

`yaxis.labels`
: User defined y axis labels to be plotted. If `NULL`(default), the function tries to do something reasonable.

`...`
: Other arguments passed to `barplot`. If `ylab` is not included, it defaults to a value depending on `plot.proportion`. If `col` is not supplied the function tries to do something reasonable.

Value

An **invisible** list whose first element is a logic value indicating whether the plot is successful. The second element is a numeric vector giving the coordinates of all the bar midpoints drawn, useful for adding to the graph.

Examples

```
file <- system.file("extdata",
  "Liver-HCC.exposure.csv",
  package = "mSigTools"
)
exposure <- read_exposure(file)
old.par <- par(mar = c(8, 5, 1, 1))
plot_exposure(exposure[, 1:30],
  main = "Liver-HCC exposure", cex.yaxis = 0.8,
  plot.proportion = TRUE
)
par(old.par)
```

---

plot_exposure_to_pdf *Plot exposures in multiple plots each with a manageable number of samples to PDF*

Description

Plot exposures in multiple plots each with a manageable number of samples to PDF

Usage

```
plot_exposure_to_pdf(
  exposure,
  file,
  mfrow = c(2, 1),
  mar = c(6, 4, 3, 2),
  oma = c(3, 2, 0, 2),
  samples.per.line = 30,
  plot.proportion = FALSE,
  xlim = NULL,
  ylim = NULL,
  legend.x = NULL,
  legend.y = NULL,
  cex.legend = 0.9,
  cex.yaxis = 1,
```

```
  cex.xaxis = NULL,
  plot.sample.names = TRUE,
  yaxis.labels = NULL,
  width = 8.2677,
  height = 11.6929,
  ...
)
```

Arguments

exposure	Exposures as a numerical matrix (or data.frame) with signatures in rows and samples in columns. Rownames are taken as the signature names and column names are taken as the sample IDs. If you want exposure sorted from largest to smallest, use sort_exposure. Do not use column names that start with multiple underscores. The exposures will often be mutation counts, but could also be e.g. mutations per megabase.
file	The name of the PDF file to be produced.
mfrow	A vector of the form c(nr,nc). Subsequent figures will be drawn in an nr-by-nc array on the device by rows.
mar	A numerical vector of the form c(bottom,left,top,right) which gives the number of lines of margin to be specified on the four sides of the plot.
oma	A vector of the form c(bottom,left,top,right) giving the size of the outer margins in lines of text.
samples.per.line	Number of samples to show in each plot.
plot.proportion	Plot exposure proportions rather than counts.
xlim, ylim	Limits for the x and y axis. If NULL(default), the function tries to do something reasonable.
legend.x, legend.y	The x and y co-ordinates to be used to position the legend.
cex.legend	A numerical value giving the amount by which legend plotting text and symbols should be magnified relative to the default.
cex.yaxis	A numerical value giving the amount by which y axis values should be magnified relative to the default.
cex.xaxis	A numerical value giving the amount by which x axis values should be magnified relative to the default. If NULL(default), the function tries to do something reasonable.
plot.sample.names	Whether to plot sample names below the x axis. Default is TRUE.
yaxis.labels	User defined y axis labels to be plotted. If NULL(default), the function tries to do something reasonable.
width, height	The width and height of the graphics region in inches.
...	Other arguments passed to barplot. If ylab is not included, it defaults to a value depending on plot.proportion. If col is not supplied the function tries to do something reasonable.

Value

An **invisible** list whose first element is a logic value indicating whether the plot is successful. The second element is a numeric vector giving the coordinates of all the bar midpoints drawn, useful for adding to the graph.

Examples

```
file <- system.file("extdata",
  "Liver-HCC.exposure.csv",
  package = "mSigTools"
)
exposure <- read_exposure(file)
plot_exposure_to_pdf(exposure,
  file = file.path(tempdir(), "Liver-HCC.exposure.pdf"),
  cex.yaxis = 0.8, plot.proportion = TRUE
)
```

---

read_exposure *Read an exposure matrix from a file*

Description

Read an exposure matrix from a file

Usage

```
read_exposure(file, check.names = FALSE)
```

Arguments

file	CSV file containing an exposure matrix.
check.names	Passed to read.csv. **IMPORTANT**: If TRUE this will replace the double colon in identifiers of the form <tumor_type>::<sample_id> with two periods (i.e. <tumor_type>..<sample_id>. If check.names is true, generate a warning if double colons were present.

Value

Matrix of exposures.

Examples

```
file <- system.file("extdata",
  "Liver-HCC.exposure.csv",
  package = "mSigTools"
)
exposure <- read_exposure(file)
```

sig_dist_matrix — *Compute a matrix of distances / similarities between two sets of signatures*

Description

Compute a matrix of distances / similarities between two sets of signatures

Usage

```
sig_dist_matrix(x1, x2, method = "cosine")
```

Arguments

x1	The first set of signatures (a positive matrix in which each column is a signature). The elements of x1 will be the rows of the output matrix
x2	The second set of signatures, similar data type to x1. The elements of x2 will be the columns of the output matrix
method	(as for the philentropy::distance) function.

Value

A matrix with dimensions ncol(x1) X ncol(x2) with each element representing the distance or similarity (depending on method) between the corresponding elements of x1 and x2

Examples

```
ex.sigs <- matrix(c(0.2, 0.8, 0.3, 0.7, 0.4, 0.6), nrow = 2)
colnames(ex.sigs) <- c("ex1", "ex2", "ex3")
gt.sigs <- matrix(c(0.21, 0.79, 0.19, 0.81), nrow = 2)
colnames(gt.sigs) <- c("gt1", "gt2")
sig_dist_matrix(ex.sigs, gt.sigs)
```

sort_exposure — *Sort columns of an exposure matrix from largest to smallest (or vice versa)*

Description

Sort columns of an exposure matrix from largest to smallest (or vice versa)

Usage

```
sort_exposure(exposure, decreasing = TRUE)
```

Arguments

exposure	Exposures as a numerical matrix (or data.frame) with signatures in rows and samples in columns. Rownames are taken as the signature names and column names are taken as the sample IDs.
decreasing	If TRUE, sort from largest to smallest.

Value

The original exposure with columns sorted.

Examples

```
file <- system.file("extdata",
  "Liver-HCC.exposure.csv",
  package = "mSigTools"
)
exposure <- read_exposure(file)
exposure.sorted <- sort_exposure(exposure)
```

TP_FP_FN_avg_sim *Find best matches (by cosine similarity) of a set of mutational signatures to a set of reference mutational signatures*

Description

Find best matches (by cosine similarity) of a set of mutational signatures to a set of reference mutational signatures

Usage

```
TP_FP_FN_avg_sim(extracted.sigs, reference.sigs, similarity.cutoff = 0.9)
```

Arguments

extracted.sigs	Mutational signatures discovered by some analysis. A numerical-matrix-like object with columns as signatures.
reference.sigs	A numerical-matrix-like object with columns as signatures. This matrix should contain the reference mutational signatures. For example, these might be from a synthetic data set or they could be from reference set of signatures, such as the signatures at the COSMIC mutational signatures web site. See CRAN package cosmicsig.
similarity.cutoff	A signature in reference.sigs must be matched by >= similarity.cutoff by a signature in extracted.sigs to be considered detected.

Details

Match signatures in `extracted.sigs` to signatures in `reference.sigs` using the function `solve_LSAP`, which uses the "Hungarian" (a.k.a "Kuhn–Munkres") algorithm https://en.wikipedia.org/wiki/Hungarian_algorithm, which optimizes the total cost associated with the links between nodes. The function first computes the all-pairs cosine similarity matrix between the two sets of signatures, then converts cosine similarities to cosine distances (including `similarity.cutoff`) by subtracting from 1, then sets distances > the converted cutoff to very large values. It then applies `solve_LSAP` to the resulting matrix to compute an optimal matching between `extracted.sigs` and `reference.sigs`.

Value

A list with the elements

- `TP` The number of true positive extracted signatures.
- `FP` The number of false positive extracted signatures.
- `FN` The number of false negative reference signatures.
- `avg.cos.sim` Average cosine similarity of true positives to their matching reference signatures.
- `table` Table of extracted signatures that matched a reference signature. Each row contains the extracted signature name, the reference signature name, and the cosine similarity of the match.
- `sim.matrix` The similarity matrix corresponding to the input signatures.
- `unmatched.ex.sigs` The identifiers of the extracted signatures that did not match a reference signature.
- `unmatched.ref.sigs` The identifiers of the reference signatures that did not match an extracted signature.

Examples

```
ex.sigs <- matrix(c(0.2, 0.8, 0.3, 0.7, 0.6, 0.4), nrow = 2)
colnames(ex.sigs) <- c("ex1", "ex2", "ex3")
ref.sigs <- matrix(c(0.21, 0.79, 0.19, 0.81), nrow = 2)
colnames(ref.sigs) <- c("ref1", "ref2")
TP_FP_FN_avg_sim(
  extracted.sigs = ex.sigs,
  reference.sigs = ref.sigs,
  similarity.cutoff = .9
)
```

---

write_exposure *Write an exposure matrix to a file*

Description

Write an exposure matrix to a file

Usage

```
write_exposure(exposure, file, row.names = TRUE)
```

Arguments

exposure	Exposures as a numerical matrix (or data.frame) with signatures in rows and samples in columns. Rownames are taken as the signature names and column names are taken as the sample IDs.
file	File to which to write the exposure matrix (as a CSV file).
row.names	Either a logical value indicating whether the row names of exposure are to be written along with exposure, or a character vector of row names to be written.

Examples

```
file <- system.file("extdata",
  "Liver-HCC.exposure.csv",
  package = "mSigTools"
)
exposure <- read_exposure(file)
write_exposure(exposure, file = file.path(tempdir(), "Liver-HCC.exposure.csv"))
```

Index