Package 'mSigTools'

August 27, 2022

Type Package

Title Mutational Signature Analysis Tools

Version 1.0.3

Description Utility functions for mutational signature analysis. This package provides two groups of functions. One is for dealing with mutational signature ``exposures" (i.e. the counts of mutations in a sample that are due to each mutational signature). The other group of functions is for matching two sets of mutational signatures. The match minimizes the total distance between paired signatures by using the ``Hungarian algorithm" described in : Kuhn, H. W. (1955) <doi:10.1002/nav.3800020109>. 'mSigTools' stands for mutational Signature analysis Tools.

License GPL-3

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

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

Encoding UTF-8

Language en-US

RoxygenNote 7.2.1

Imports clue, philentropy

Suggests spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Steven Rozen [aut, cre] (<https://orcid.org/0000-0002-4288-0056>), Nanhai Jiang [aut] (<https://orcid.org/0000-0003-4974-2753>)

Maintainer Steven Rozen <steverozen@pm.me>

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`	As for the `distance` function in package `philenropy`.
`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. This function generates a distance matrix between the two sets of signatures using `method` and, if necessary, `convert.sim.to.dist`. It then sets distances > `cutoff` to very large values and then applies `solve_LSAP` to the resulting matrix to compute a matching between `x1` and `x2` that minimizes the sum of the distances.

Value

A list with the elements

- `table` Table of extracted signatures that matched a reference signature. Each row contains the extracted signature name, the reference signature name, and the distance of the match.
- `orig.matrix` The matrix of numeric distances between `x1` and `x2`.
- `modified.matrix` The argument `orig.matrix` with distances > `cutoff` changed to very large values.

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")
match_two_sig_sets(ex.sigs, ref.sigs, cutoff = .9)
```

`plot_exposure` *Plot exposures in multiple plots, with each plot showing exposures for a manageable number of samples.*

Description

Plot exposures in multiple plots, with each plot showing exposures for 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. The first element is a logical value indicating whether the plot is successful. The second element is a numeric vector giving the coordinates of the bar x-axis 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 to a single PDF file, with each plot showing exposures for a manageable number of samples.*

Description

Plot exposures in multiple plots to a single PDF file, with each plot showing exposures for a manageable number of samples.

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. The first element is a logical value indicating whether the plot is successful. The second element is a numeric vector giving the coordinates of the bar x-axis 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	File path to a CSV file containing an exposure matrix, i.e. the numbers of mutations due to each mutational signature. Each row corresponds to a mutational signature an each column corresponds to a tumor or other biological sample.
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

Numerical matrix of exposures, with the same shape as the contents of file.

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 numerical matrix-like object in which each column is a signature).
x2	The second set of signatures, similar data type to x1, and must have the same number of rows as x1.
method	As for the distance function in package philenropy.

Value

A numeric matrix with dimensions ncol(x1) X ncol(x2). Each element represents the distance or similarity (depending on method) between a column in x1 and a column in 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")
ref.sigs <- matrix(c(0.21, 0.79, 0.19, 0.81), nrow = 2)
colnames(ref.sigs) <- c("ref1", "ref2")
sig_dist_matrix(ex.sigs, ref.sigs)
```

`sort_exposure` *Sort columns of an exposure matrix based on the number of mutations in each sample (column).*

Description

Sort columns of an exposure matrix based on the number of mutations in each sample (column).

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 `match_two_sig_sets` based on cosine similarity.

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` The average cosine similarity of true positives to their matching reference signatures.
- `table` A data.frame 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 numeric distance or similarity matrix between `extracted.sigs` and `reference.sigs` as returned from `sig_dist_matrix`.
- `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.

Value

No return value, called for side effects.

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