Package ‘mSigTools’

January 13, 2023

Type Package

Title Mutational Signature Analysis Tools

Version 1.0.7

Description Utility functions for mutational signature analysis as described in Alexandrov, L. B. (2020) <doi:10.1038/s41586-020-1943-3>

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 or comparing sets of mutational signatures. `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 7.2.3

Imports clue, philentropy, quadprog, sets

Suggests cosmicsig, ICAMS, 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>

Repository CRAN

Date/Publication 2023-01-13 13:30:05 UTC
R topics documented:

find_best_reconstruction_QP ........................................ 2
match_two_sig_sets .................................................. 4
optimize_exposure_QP ............................................... 5
plot_exposure ........................................................... 6
plot_exposure_to_pdf ................................................. 7
read_exposure ............................................................ 9
sig_dist_matrix .......................................................... 10
sort_exposure ............................................................ 11
TP_FP_FN_avg_sim ..................................................... 12
write_exposure ........................................................... 13

Index 14

find_best_reconstruction_QP

Find "best" reconstruction of a target signature or spectrum from a set of signatures.

Description

Find "best" reconstruction of a target signature or spectrum from a set of signatures.

Usage

find_best_reconstruction_QP(
  target.sig,
  sig.universe,
  max.subset.size = NULL,
  method = "cosine",
  trim.less.than = 1e-10
)

Arguments

target.sig The signature or spectrum to reconstruct; a non-negative numeric vector or 1-column matrix-like object.

sig.universe The universe of signatures from which to reconstruct target.sig; a non-negative numeric matrix-like object with nrow(sig.universe) == length(target.sig). The sums of each column must be 1. Must not contain duplicate columns or have other non-unique quadratic programming solutions (not checked, but will generate an error from solve.QP in package quadprog).

max.subset.size Maximum number of signatures to use to reconstruct target.sig.
As in dist_one_one in package philentropy, and used only to find the final "best" reconstruction. The optimized exposures from which to selected the "best" reconstruction are calculated using optimize_exposure_QP, which uses solve.QP in package quadprog.

After optimizing exposures with optimize_exposure_QP, remove exposures less than trim.less.than and then re-optimize.

This function should be fast if you do not specify max.subset.size, but it will be combinatorially slow if max.subset.size is large and trim.less.than is small or negative. So do not do that. If max.subset.size is NULL, then the function just uses optimize_exposure_QP. and then excludes exposures < trim.less.than, and then re-runs optimize_exposure_QP. Otherwise, after excluding exposures < trim.less.than, then the function runs optimize_exposure_QP on subsets of signatures of size <= max.subset.size, removes exposures < trim.less.than, reruns optimize_exposure_QP, calculates the reconstruction and similarity between the reconstruction and the target.sig and returns the information for the exposures that have the greatest similarity.

A list with elements:

- optimized.exposure A numerical vector of the exposures that give the "best" reconstruction. This vector is empty if there is an error.
- similarity The similarity between the reconstruction (see below) and target.sig according to the distance or similarity provided by the method argument.
- method The value specified for the method argument, or an error message if optimize.exposure is empty.
- reconstruction The reconstruction of target.sig according to optimized.exposure.

Examples

```r
set.seed(888)
sig.u <-
do.call(
cbind,
lapply(1:6, function(x) {
col <- runif(n = 96)
  col / sum(col)
})
)
rr <- find_best_reconstruction_QP(
  target.sig = sig.u[, 1, drop = FALSE],
  sig.universe = sig.u[, 2:6]
)
names(rr)
rr$optimized.exposure
rr$similarity
rr <- find_best_reconstruction_QP(
  target.sig = sig.u[, 1, drop = FALSE],
```
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.
optimize_exposure_QP

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 \( x_1 \) and \( x_2 \).
- modified.matrix The argument orig.matrix with distances > cutoff changed to very large values.

Examples

```r
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)
```

---

**optimize_exposure_QP**  
*Quadratic programming optimization of signature activities*

Description

Quadratic programming optimization of signature activities

Usage

```r
optimize_exposure_QP(spectrum, signatures)
```

Arguments

- `spectrum` Mutational signature or mutational spectrum as a numeric vector or single column data frame or matrix.
- `signatures` Matrix or data frame of signatures from which to reconstruct `spectrum`. Rows are mutation types and columns are signatures. Should have column names for interpretable results. Cannot be a vector because the column names are needed.

Details

Code adapted from `SignatureEstimation::decomposeQP` and uses `solve.QP` in package `quadprog`.

Value

A vector of exposures with names being the colnames from signatures.
Examples

```r
usigs <- matrix(c(0.2, 0.7, 0.1,
                  0.3, 0.6, 0.1,
                  0.1, 0.1, 0.8), nrow = 3)
colnames(usigs) <- c("u1", "u2", "u3")
tsig <- matrix(c(0.25, 0.65, 0.1), nrow = 3)
optimize_exposure_QP(tsig, usigs)
```

plot_exposure

**Description**

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

**Usage**

```r
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. Ignored if there are no column names on exposure.

**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**

```r
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.
read_exposure

**Description**

Read an exposure matrix from a file.

**Usage**

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

**Examples**

```r
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)
```
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 and 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

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

Description

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

Usage

```r
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.
sort_exposure

Examples

```r
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**

```r
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**

```r
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.
write_exposure

- 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

```r
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

```r
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

```r
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

barplot, 7, 9

dist_one_one, 3
distance, 4, 10

find_best_reconstruction_QP, 2

match_two_sig_sets, 4, 12

optimize_exposure_QP, 3, 5

plot_exposure, 6
plot_exposure_to_pdf, 7

read_exposure, 9

sig_dist_matrix, 10, 12

solve.QP, 2, 3, 5

solve_LSAP, 4

sort_exposure, 6, 8, 11

TP_FP_FN_avg_sim, 12

write_exposure, 13