

Package ‘concordexR’

June 6, 2023

Title Calculate the concordex coefficient

Version 1.1.0

Description Many analysis workflows include approximation of a nearest neighbors graph followed by clustering of the graph structure. The concordex coefficient estimates the concordance between the graph and clustering results. The package 'concordexR' is an R implementation of the original concordex Python-based command line tool.

License Artistic-2.0

URL <https://github.com/pachterlab/concordexR>,
<https://pachterlab.github.io/concordexR/>

BugReports <https://github.com/pachterlab/concordexR/issues>

Imports BiocParallel, cli, DelayedArray, ggplot2, Matrix, methods,
pheatmap, rlang, scales

Suggests BiocNeighbors, BiocStyle, bluster, covr, knitr, patchwork,
rmarkdown, scater, TENxPBMCDData, testthat (>= 3.0.0), vdiff

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Depends R (>= 4.2)

Config/testthat/edition 3

biocViews SingleCell, Clustering, GraphAndNetwork

VignetteBuilder knitr

git_url <https://git.bioconductor.org/packages/concordexR>

git_branch devel

git_last_commit b90a1ca

git_last_commit_date 2023-04-25

Date/Publication 2023-06-06

Author Kayla Jackson [aut, cre] (<<https://orcid.org/0000-0001-6483-0108>>),
A. Sina Boeshaghi [aut] (<<https://orcid.org/0000-0002-6442-4502>>),
Angel Galvez-Merchan [aut] (<<https://orcid.org/0000-0001-7420-8697>>),
Lambda Moses [aut] (<<https://orcid.org/0000-0002-7092-9427>>),
Laura Luebbert [ctb] (<<https://orcid.org/0000-0003-1379-2927>>),
Lior Pachter [aut, rev, ths] (<<https://orcid.org/0000-0002-9164-6231>>)

Maintainer Kayla Jackson <kaylajac@caltech.edu>

R topics documented:

calculateConcordex	2
heatConcordex	4
plotConcordexSim	4
Index	6

calculateConcordex	<i>Compute the concordex coefficient</i>
--------------------	--

Description

Compute the raw and corrected concordex coefficient using a neighborhood graph and observation labels.

Usage

```
calculateConcordex(x, ...)  
  
## S4 method for signature 'ANY'  
calculateConcordex(  
  x,  
  labels,  
  k = 20,  
  n.iter = 15,  
  return.map = TRUE,  
  BPPARAM = SerialParam()  
)
```

Arguments

- x A numeric matrix specifying the neighborhood structure of observations. Typically an adjacency matrix produced by a k-Nearest Neighbor algorithm. It can also be a matrix whose rows correspond to each observation and columns correspond to neighbor indices, i.e. matrix form of an adjacency list which can be a matrix due to fixed number of neighbors.
- ... Arguments passed to methods.

labels	A numeric or character vector containing the label or class corresponding to each observation. For example, a cell type or cluster ID.
k	Number of neighbors to expect for each observation. Defaults to 20.
n.iter	A number specifying the number of permutations for correcting the coefficient.
return.map	Logical, whether to return the matrix of the number of cells of each label in the neighborhood of cells of each label.
BPPARAM	A BiocParallelParam object specifying whether and how computing the metric for numerous observations shall be parallelized.

Value

A named list with the following components:

- concordex The raw concordex coefficient corresponding to the original label assignments.
- mean_random_concordex The average of n.iter concordex coefficients. concordex coefficients are computed after permuting the labels and reassigning them to new observations.
- corrected_concordex Simply the raw concordex coefficient divided by the average of the permuted coefficients.
- simulated Numeric vector of the concordex coefficients from permuted labels, showing the null distribution.
- map Numeric matrix of the number of cells of each label in the neighborhood of cells of each label. Only returned when return.map = TRUE.

Examples

```
# Simplest case where input is a nxn matrix
# Neighbors can be oriented along the rows or columns
nCells <- 10
k <- 3
set.seed(40)
labels <- sample(paste0("1", seq_len(3)), nCells, replace=TRUE)

mtx <- sapply(seq_len(nCells), function(x) {
  out <- rep(0, nCells)
  out[-x] <- sample(c(rep(1, k), rep(0, nCells - k - 1)))
  out
})

res <- calculateConcordex(mtx, labels, k = k)

res

# Also works if input matrix is nxk or kxn
mtx <- sapply(seq_len(nCells), function(x) {
  out <- sample((seq_len(nCells))[-x], k)
  out
})

res <- calculateConcordex(mtx, labels, k = k)
```

res

heatConcordex	<i>Plot the concordex map matrix as a heatmap</i>
---------------	---

Description

The calculateConcordex function returns a matrix showing the number of cells of each label in the neighborhood of cells of each label when argument `return.map = TRUE`. This function plots this matrix as a heatmap, which can be used as a clustering diagnostic.

Usage

```
heatConcordex(concordex, ...)
```

Arguments

concordex	Output from calculateConcordex .
...	Other arguments passed to pheatmap to customize the plot.

Value

A pheatmap object.

Examples

```
library(BiocNeighbors)
g <- findKNN(iris[, seq_len(4)], k = 10)
res <- calculateConcordex(g$index,
  labels = iris$Species, k = 10,
  return.map = TRUE
)
heatConcordex(res)
```

plotConcordexSim	<i>Plot density plot of simulated results</i>
------------------	---

Description

The concordex values from permuted labels represent the null distribution of the statistic. This can be plotted as a density plot and visually compared to the actual value.

Usage

```
plotConcordexSim(concordex, ...)
```

Arguments

<code>concordex</code>	Output from calculateConcordex .
<code>...</code>	Other arguments passed to geom_density .

Value

A `ggplot2` object. The density plot shows the simulated concordex coefficient from permuted labels, while the vertical line shows the actual concordex coefficient.

Examples

```
library(BiocNeighbors)
g <- findKNN(iris[, seq_len(4)], k = 10)
res <- calculateConcordex(g$index, labels = iris$Species, k = 10)
plotConcordexSim(res)
```

Index

BiocParallelParam, [3](#)

calculateConcordex, [2](#), [4](#), [5](#)

calculateConcordex, ANY-method
(calculateConcordex), [2](#)

geom_density, [5](#)

heatConcordex, [4](#)

pheatmap, [4](#)

plotConcordexSim, [4](#)