Package ‘treekoR’

April 16, 2024

Type Package

Title Cytometry Cluster Hierarchy and Cellular-to-phenotype Associations

Version 1.10.0


Description treekoR is a novel framework that aims to utilise the hierarchical nature of single cell cytometry data to find robust and interpretable associations between cell subsets and patient clinical end points. These associations are aimed to recapitulate the nested proportions prevalent in workflows involving manual gating, which are often overlooked in workflows using automatic clustering to identify cell populations. We developed treekoR to: Derive a hierarchical tree structure of cell clusters; quantify a cell type as a proportion relative to all cells in a sample (%total), and, as the proportion relative to a parent population (%parent); perform significance testing using the calculated proportions; and provide an interactive html visualisation to help highlight key results.

Depends R (>= 4.1)

Imports stats, utils, tidyr, dplyr, data.table, ggiraph, ggplot2, hopach, ape, ggtree, patchwork, SingleCellExperiment, diffcyt, edgeR, lme4, multcomp

License GPL-3

Encoding UTF-8

LazyData false

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, BiocStyle, CATALYST, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

git_url https://git.bioconductor.org/packages/treekoR

git_branch RELEASE_3_18
addFreqBars

**addFreqBars**

**Description**

a function to add the frequency bars for each cluster
addHeatMap

Usage

addFreqBars(
  p,
  clusters,
  offset = 0.75,
  bar_length = 3,
  bar_width = 0.4,
  freq_labels = FALSE
)

Arguments

p a phylogenetic tree plot created from the ggtree() function
clusters a vector representing the cell type or cluster of each cell (can be character or numeric)
offset distance between the heatmap and frequency bars
bar_length length of bar with max frequency
bar_width width of each frequency bar
freq_labels boolean indicated whether or not to show frequency bar labels

Value

an interactive ggplot graph object with frequency bars of clusters alongside heatmap of cluster median expression

addHeatMap  Title

Description

a function to add a heatmap of cluster medians alongside the phylogenetic tree

Usage

addHeatMap(
  p,
  cluster_medians,
  offset = 0.5,
  width = 1,
  expand_y_lim = 20,
  low = "#313695",
  mid = "ivory",
  high = "#A50026",
  colnames_angle = 90,
  metric_name = "Column z-score"
)
Arguments

- **p**: a phylogenetic tree plot created from the `ggtree()` function
- **cluster_medians**: a dataframe with the cluster medians. The row numbers of the clusters median dataframe should correspond to the nodes in the phylo tree. The column names should also correspond to the labels you want to use
- **offset**: the distance between the tree plot and heatmap
- **width**: width of each tile in the heatmap
- **expand_y_lim**: white space below heatmap
- **low**: colour used for low values on heatmap
- **mid**: colour used for medium values on heatmap
- **high**: colour used for large values on heatmap
- **colnames_angle**: angle for x-axis label
- **metric_name**: legend title

Value

an interactive ggplot graph object with heatmap of median cluster expressions plotted alongside hierarchical tree

Description

a function to create a skeleton tree diagram to display significance testing results on each node

Usage

```r
addTree(p, offset = 0.3, font_size = 2.5, hjust = 0)
```

Arguments

- **p**: a phylogenetic tree plot created from the `ggtree()` function
- **offset**: distance between leaf nodes on the tree and their labels
- **font_size**: font size of leaf labels
- **hjust**: horizontal justification as defined in ggplot2

Value

a ggtree graph object with the hierarchical tree of clusters and corresponding labels
**Description**

Adding statistical test results onto the tree by using colourful nodes and branches. Takes a `ggtree` object with test results for each node and returns a `ggtree` graph object.

**Usage**

```r
colourTree(
    tree,
    point_size = 1.5,
    high = "#00c434",
    low = "purple",
    mid = "ivory2"
)
```

**Arguments**

- **tree**: a tree plot created from the `ggtree()` function with `p$data` containing test statistic and `p-val`
- **point_size**: size of nodes in the tree
- **high**: colour for large values
- **low**: colour for low values
- **mid**: colour for middle values

**Value**

An interactive `ggplot` graph object, plotting the hierarchical tree of clusters with nodes and branches coloured by the significance testing results.

**Examples**

```r
library(SingleCellExperiment)
data(COVIDSampleData)

sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_tree <- getClusterTree(exprs,
    clusters,
    hierarchy_method="hopach")
```
```r
tested_tree <- testTree(clust_tree$clust_tree, 
                        clusters=clusters, 
                        samples=samples, 
                        classes=classes)

colourTree(tested_tree)
```

---

**DeBiasi_COVID_CD8_samp**

*COVID-19 Sample data*

---

**Description**

Data from a an experiment investigating T cell compositions between COVID-19 patients and healthy control. This data has been transformed using a arcsinh transform using a co-factor of 5 and randomly subsetted.

**Usage**

```r
data(COVIDSampleData)
```

**Format**

An object of class "SingeCellExperiment"

**Source**

FlowRepository

**References**


**Examples**

```r
data(COVIDSampleData)
```
**findChildren**

**Description**

findChildren

**Usage**

findChildren(tree)

**Arguments**

- `tree` a ggtree object

**Value**

a ggtree object with the data containing a column with the clusters contained in each node

---

**geometricMean**

**Description**

ggetCellGMeans helper function

**Usage**

geometricMean(x, na.rm = TRUE)

**Arguments**

- `x` vector containing numeric values
- `na.rm` whether or not to ignore NA values

**Value**

g geomtric mean of vector x
getCellGMeans

description
getCellGMeans

Usage
getCellGMeans(phylo, exprs, clusters, samples, classes)

Arguments
phylo a phylogram with tip.labels corresponding to cell types/cluster contained in 'clusters' vector
exprs a dataframe containing single cell expression data
clusters a vector representing the cell type or cluster of each cell (can be character or numeric). If numeric, cluster names need to be consecutive starting from 1.
samples a vector identifying the patient each cell belongs to
classes a vector containing the patient outcome/class each cell belongs to

Value
a dataframe containing proportions calculated for each sample

Examples
library(SingleCellExperiment)
data(COVIDSampleData)
sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_tree <- getClusterTree(exprs,
    clusters,
    hierarchy_method="hopach")

means_df <- getCellGMeans(clust_tree$clust_tree,
    exprs=exprs,
    clusters=clusters,
    samples=samples,
    classes=classes)
getCellProp

Description
getCellProp

Usage
getCellProp(phylo, clusters, samples, classes, excl_top_node_parent = TRUE)

Arguments
phylo  
a phylogram with tip.labels corresponding to cell types/cluster contained in 'clusters' vector
clusters  
a vector representing the cell type or cluster of each cell (can be character or numeric). If numeric, cluster names need to be consecutive starting from 1.
samples  
a vector identifying the patient each cell belongs to
classes  
a vector containing the patient outcome/class each cell belongs to
excl_top_node_parent  
a boolean indicating whether the for cell types with the highest node as their parent

Value
a dataframe containing proportions calculated for each sample

Examples
library(SingleCellExperiment)
data(COVIDSampleData)
sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_tree <- getClusterTree(exprs,
                           clusters,
                           hierarchy_method="hopach")

prop_df <- getCellProp(clust_tree$clust_tree,
                       clusters=clusters,
                       samples=samples,
                       classes=classes)
getClusterTree

This function takes a CATALYST sce with clusters and creates a hierarchical tree.

**Description**

getClusterTree

**Usage**

```r
getClusterTree(
  exprs,
  clusters,
  hierarchy_method = "hopach",
  hopach_kmax = 5,
  hopach_K = 10,
  scale_exprs = TRUE
)
```

**Arguments**

- `exprs`: a dataframe containing single cell expression data
- `clusters`: a vector representing the cell type or cluster of each cell (can be character or numeric). If numeric, cluster names need to be consecutive starting from 1.
- `hierarchy_method`: a string indicating the hierarchical tree construction method to be used
- `hopach_kmax`: integer between 1 and 9 specifying the maximum number of children at each node in the tree
- `hopach_K`: positive integer specifying the maximum number of levels in the tree. Must be 15 or less, due to computational limitations (overflow)
- `scale_exprs`: boolean indicating whether to scale median cluster expression data before constructing hierarchical tree

**Value**

a list containing the cluster median frequencies and a phylogram of the hierarchical tree

**Examples**

```r
library(SingleCellExperiment)
data(COVIDSampleData)
sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id
```
**getParentProp**

```r
clust_tree <- getClusterTree(exprs, 
    clusters, 
    hierarchy_method="hopach")
```

---

**getTotalProp**

---

**Description**
getCellProp helper function

**Usage**

```
getParentProp(vars1, vars2, n_cells)
```

**Arguments**

```r
vars1  
name of cell type, matching to column in n_cells
vars2  
name of parent cell type, matching to column in n_cells
n_cells  
matrix of counts of each cell type per sample
```

**Value**

a vector containing the proportions of cell type vars1 as a percent of parent vars2 per sample

---

**Description**
getCellProp helper function

**Usage**

```
getTotalProp(vars1, n_cells, n_cells_pat)
```

**Arguments**

```r
vars1  
name of cell type, matching to column in n_cells
n_cells  
matrix of counts of each cell type per sample
n_cells_pat  
vector containing number of cells per sample
```

**Value**

a vector containing the proportions of cell type vars1 as a percent of total per sample
getTreeResults

Description

getTreeResults

Usage

getTreeResults(testedTree, sort_by = "parent")

Arguments

testedTree  a ggtree object outputed from testTree()
sort_by     whether to sort by p-values testing via proportions to parent or p-values testing via absolute proportions. Values can be c(NA, "parent", "all")

Value

a dataframe with hierarchical tree nodes, corresponding clusters and corresponding significance testing results

Examples

library(SingleCellExperiment)
data(COVISSampleData)
sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_tree <- getClusterTree(exprs,
      clusters,
      hierarchy_method="hopach")

tested_tree <- testTree(clust_tree$clust_tree,
      clusters=clusters,
      samples=samples,
      classes=classes,
      pos_class_name=NULL)

res_df <- getTreeResults(tested_tree)

head(res_df, 10)
Description

hopachToPhylo

Usage

hopachToPhylo(res)

Arguments

res an object returned from the runHOPACH() function

Value

a phylogram converted from the outputted list from the runHOPACH function

Examples

library(SingleCellExperiment)
library(data.table)
data(COVIDSampleData)
sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_med_dt <- as.data.table(exprs)
clust_med_dt[, cluster_id := clusters]
res <- clust_med_dt[, lapply(.SD, median, na.rm=TRUE), by=cluster_id]
res2 <- res[, .SD, .SDcols = !c('cluster_id')]

hopach_res <- runHOPACH(as.data.frame(scale(res2)))
phylo <- hopachToPhylo(hopach_res)
plotInteractiveHeatmap

Description

This function takes a hierarchical tree which has been tested for proportion to all and proportion to parent cluster.

Usage

plotInteractiveHeatmap(
  testedTree, clust_med_df, clusters,
  svg_width = 13, svg_height = 9,
  tr_offset = 0.3, tr_font_size = 2,
  tr_point_size = 1.5,
  tr_col_high = "#00c434", tr_col_low = "purple",
  tr_col_mid = "ivory2",
  hm_offset = 1,
  hm_tile_width = 1,
  hm_expand_y_lim = 20,
  hm_col_high = "#cc2010",
  hm_col_mid = "#fff8de",
  hm_col_low = "#66a6cc",
  fb_offset = 0.75,
  fb_bar_length = 3,
  fb_bar_width = 0.4,
  fb_freq_labels = FALSE
)

Arguments

testedTree: a ggtree object that has been run through the testTree
clust_med_df: a dataframe with the cluster medians. The row numbers of the clusters median data frame should correspond to the nodes in the phylo tree. The column names should also correspond to the labels you want to use.
clusters: a vector representing the cell type or cluster of each cell (can be character or numeric)
svg_width: width of svg canvas
svg_height: height of svg canvas
tr_offset: distance between leaf nodes on the tree and their labels
plotInteractiveHeatmap

tr_font_size    font size of leaf labels
tr_point_size   size of each node in the tree
tr_col_high     colour used for high test statistics, coloured on the nodes and branches of the tree
tr_col_low      colour used for low test statistics, coloured on the nodes and branches of the tree
tr_col_mid      colour used for medium test statistics, coloured on the nodes and branches of the tree
hm_offset       distance between the tree and the heatmap
hm_tile_width   width of each tile in the heatmap
hm_expand_y_lim white space below heatmap
hm_col_high     colour used for large values on heatmap
hm_col_mid      colour used for medium values on heatmap
hm_col_low      colour used for low values on heatmap
fb_offset       distance between the heatmap and frequency bars
fb_bar_length   length of bar with max frequency
fb_bar_width    width of each frequency bar
fb_freq_labels  boolean indicated whether or not to show frequency bar labels

Value

an interactive ggplot object containing the hierarchical tree of clusters coloured by significance testing results, with corresponding heatmap and a scatterplot comparing significance when testing using proportions to parent vs. absolute proportions

Examples

library(SingleCellExperiment)
data(COVIDSampleData)
sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_tree <- getClusterTree(exprs,
                           clusters,
                           hierarchy_method="hopach")

tested_tree <- testTree(clust_tree$clust_tree,
                         clusters=clusters,
                         samples=samples,
                         classes=classes)

plotInteractiveHeatmap(tested_tree,
                         clust_med_df = clust_tree$median_freq,
                         clusters=clusters)
**plotSigScatter**

**Description**

plotSigScatter

**Usage**

plotSigScatter(testedTree, scatter_tooltip, max_val)

**Arguments**

- **testedTree**: an output from the function testTree()
- **scatter_tooltip**: vector containing tooltips for interactive plot
- **max_val**: maximum value to set as x/y axis limits

**Value**

a ggplot object, containing test statistics from testing proportions relative to parent against the test statistics from testing absolute proportions.

---

**runEdgeRTests**

**Description**

This function runs edgeR using the treekoR inputs across all nodes of the hierarchical tree of clusters, adapted from the diffcyt workflow

**Usage**

runEdgeRTests(td, clusters, samples, classes, pos_class_name)

**Arguments**

- **td**: a dataframe of data from ggtree object
- **clusters**: a vector representing the cell type or cluster of each cell (can be character or numeric). If numeric, cluster names need to be consecutive starting from 1.
- **samples**: a vector identifying the patient each cell belongs to
- **classes**: a vector containing the patient outcome/class each cell belongs to
- **pos_class_name**: a character indicating which class should be treated as positive

**Value**

a dataframe with pvalues, test statistic (signed -log10(p)), and FDR
runGLMMTests

Description
This function runs GLMM using the treekoR inputs across all nodes of the hierarchical tree of clusters, adapted from the diffcyt workflow. (Unable to get direction of test statistic currently)

Usage
runGLMMTests(td, clusters, samples, classes, pos_class_name, neg_class_name)

Arguments
- **td**: a dataframe of data from ggtree object
- **clusters**: a vector representing the cell type or cluster of each cell (can be character or numeric). If numeric, cluster names need to be consecutive starting from 1.
- **samples**: a vector identifying the patient each cell belongs to
- **classes**: a vector containing the patient outcome/class each cell belongs to
- **pos_class_name**: a character indicating which class should be treated as positive
- **neg_class_name**: a character indicating which class should be treated as negative

Value
- a dataframe with pvalues and test statistics

runHOPACH

Description
runHOPACH

Usage
runHOPACH(data, K = 10, kmax = 5, dissimilarity_metric = "cor")

Arguments
- **data**: dataframe containing the median expression of the clusters/cell types
- **K**: positive integer specifying the maximum number of levels in the tree. Must be 15 or less, due to computational limitations (overflow)
- **kmax**: integer between 1 and 9 specifying the maximum number of children at each node in the tree
- **dissimilarity_metric**: metric used to calculate dissimilarities between clusters/cell types
Value

a list containing the groups each cluster belongs to at each level of the hopach tree

Examples

```r
library(SingleCellExperiment)
library(data.table)
data(COVIDSampleData)

sce <- DeBiasi_COVID_CD8_samp
eprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_med_dt <- as.data.table(eprs)
clust_med_dt[, cluster_id := clusters]
res <- clust_med_dt[, lapply(.SD, median, na.rm=TRUE), by=cluster_id]
res2 <- res[, , SDcols = !c('cluster_id')]

hopach_res <- runHOPACH(as.data.frame(scale(res2)))
```

Description

This function takes a hierarchical tree of the cluster medians of a cytometry dataset, and then uses this structure to perform t-tests between conditions of patients testing for difference using the proportion of cluster relative to sample’s n and proportion of cluster relative to sample’s n of hierarchical parent cluster. Takes a ggtree object and returns a ggtree object with testing results appended in the data

Usage

```r
testTree(  
  phylo,  
  clusters,  
  samples,  
  classes,  
  sig_test = "ttest",  
  p_adjust = NULL,  
  pos_class_name = NULL  
)
```
testTree

Arguments

- **phylo**: a ggtree object
- **clusters**: a vector representing the cell type or cluster of each cell (can be character or numeric). If numeric, cluster names need to be consecutive starting from 1.
- **samples**: a vector identifying the patient each cell belongs to
- **classes**: a vector containing the patient outcome/class each cell belongs to
- **sig_test**: a character, either "ttest" or "wilcox" indicating the significance test to be used
- **p_adjust**: a character, indicating whether p-value adjustment should be performed. Valid values are in stats::p.adjust.methods
- **pos_class_name**: a character indicating which class is positive

Value

a ggtree object with significance testing results in embedded data

Examples

```r
library(SingleCellExperiment)
data(COVIDSampleData)
sce <- DeBiasi_COVID_CD8_samp
exprs <- t(assay(sce, "exprs"))
clusters <- colData(sce)$cluster_id
classes <- colData(sce)$condition
samples <- colData(sce)$sample_id

clust_tree <- getClusterTree(exprs,
                           clusters,
                           hierarchy_method="hopach")

tested_tree <- testTree(clust_tree$clust_tree,
                        clusters=clusters,
                        samples=samples,
                        classes=classes,
                        sig_test="ttest",
                        pos_class_name=NULL)
```
Index

* datasets
  DeBiasi_COVID_CD8_samp, 6

addFreqBars, 2
addHeatMap, 3
addTree, 4

colourTree, 5
DeBiasi_COVID_CD8_samp, 6
findChildren, 7
geometricMean, 7
getCellGMeans, 8
getCellProp, 9
getClusterTree, 10
getParentProp, 11
getTotalProp, 11
getTreeResults, 12

hopachToPhylo, 13
plotInteractiveHeatmap, 14
plotSigScatter, 16

runEdgeRTests, 16
runGLMMTests, 17
runHOPACH, 17
testTree, 18