

# Package ‘treekoR’

May 6, 2026

**Type** Package

**Title** Cytometry Cluster Hierarchy and Cellular-to-phenotype Associations

**Version** 1.21.0

**biocViews** Clustering, DifferentialExpression, FlowCytometry, ImmunoOncology, MassSpectrometry, SingleCell, Software, StatisticalMethod, Visualization

**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 types 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** devel

**git\_last\_commit** e1ad33c

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-05-05

**Author** Adam Chan [aut, cre],  
Ellis Patrick [ctb]

**Maintainer** Adam Chan <adam.s.chan@sydney.edu.au>

## Contents

addFreqBars . . . . .	2
addHeatMap . . . . .	3
addTree . . . . .	4
colourTree . . . . .	4
DeBiasi_COVID_CD8_samp . . . . .	5
findChildren . . . . .	6
geometricMean . . . . .	6
getCellGMeans . . . . .	7
getCellProp . . . . .	8
getClusterTree . . . . .	9
getParentProp . . . . .	10
getTotalProp . . . . .	10
getTreeResults . . . . .	11
hopachToPhylo . . . . .	12
plotInteractiveHeatmap . . . . .	12
plotSigScatter . . . . .	14
runEdgeRTests . . . . .	15
runGLMMTests . . . . .	15
runHOPACH . . . . .	16
testTree . . . . .	17
<b>Index</b>	<b>19</b>

---

addFreqBars	<i>Title</i>
-------------	--------------

---

## Description

a function to add the frequency bars for each cluster

## 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 <code>ggtree()</code> 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	<i>Title</i>
------------	--------------

---

**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 <code>ggtree()</code> function
cluster_medians	a dataframe with the cluster medians. The rownumbers 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
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

---

addTree	<i>Title</i>
---------	--------------

---

**Description**

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

**Usage**

```
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

---

colourTree	<i>colourTree</i>
------------	-------------------

---

**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**

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

**Arguments**

tree	a tree plot created from the <code>ggtree()</code> function with <code>p\$data</code> containing test statistic and p-
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**

```
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)

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**

```
data(COVIDSampleData)
```

**Format**

An object of class "SingleCellExperiment"

**Source**

[FlowRepository](#)

**References**

De Biasi et al. (2020) Nat Commun 11, 3434 ([Nature](#))

**Examples**

```
data(COVIDSampleData)
```

---

findChildren	<i>findChildren</i>
--------------	---------------------

---

**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	<i>geometricMean</i>
---------------	----------------------

---

**Description**

getCellGMeans helper function

**Usage**

```
geometricMean(x, na.rm = TRUE)
```

**Arguments**

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

**Value**

geomtric mean of vector x

---

getCellGMeans                      *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	<i>getCellProp</i>
-------------	--------------------

---

### 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)
```



---

getParentProp	<i>getParentProp</i>
---------------	----------------------

---

**Description**

getCellProp helper function

**Usage**

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

**Arguments**

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

---

getTotalProp	<i>getTotalProp</i>
--------------	---------------------

---

**Description**

getCellProp helper function

**Usage**

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

**Arguments**

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	<i>getTreeResults</i>
----------------	-----------------------

---

**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(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,
                       pos_class_name=NULL)

res_df <- getTreeResults(tested_tree)

head(res_df, 10)
```

---

hopachToPhylo	<i>hopachToPhylo</i>
---------------	----------------------

---

**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

*Title*

---

**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 rownumbers 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 svf canvas
tr_offset	distance between leaf nodes on the tree and their labels
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 whne 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

*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	<i>runEdgeRTests</i>
---------------	----------------------

---

**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  $-\log_{10}(p)$ ), and FDR

---

runGLMMTests	<i>runGLMMTests</i>
--------------	---------------------

---

**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	<i>runHOPACH</i>
-----------	------------------

---

**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**

```
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)))
```

---

<code>testTree</code>	<i>testTree</i>
-----------------------	-----------------

---

## 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

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

## Arguments

<code>phylo</code>	a <code>ggtree</code> object
<code>clusters</code>	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.
<code>samples</code>	a vector identifying the patient each cell belongs to
<code>classes</code>	a vector containing the patient outcome/class each cell belongs to
<code>sig_test</code>	a character, either "ttest" or "wilcox" indicating the significance test to be used
<code>p_adjust</code>	a character, indicating whether p-value adjustment should be performed. Valid values are in <code>stats::p.adjust.methods</code>
<code>pos_class_name</code>	a character indicating which class is positive

## Value

a `ggtree` object with significance testing results in embedded data

## 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,  
                        sig_test="ttest",  
                        pos_class_name=NULL)
```

# Index

## \* datasets

DeBiasi\_COVID\_CD8\_samp, [5](#)

addFreqBars, [2](#)

addHeatMap, [3](#)

addTree, [4](#)

colourTree, [4](#)

DeBiasi\_COVID\_CD8\_samp, [5](#)

findChildren, [6](#)

geometricMean, [6](#)

getCellMeans, [7](#)

getCellProp, [8](#)

getClusterTree, [9](#)

getParentProp, [10](#)

getTotalProp, [10](#)

getTreeResults, [11](#)

hopachToPhylo, [12](#)

plotInteractiveHeatmap, [12](#)

plotSigScatter, [14](#)

runEdgeRTests, [15](#)

runGLMMTests, [15](#)

runHOPACH, [16](#)

testTree, [17](#)