Package ‘scHOT’

February 21, 2026

Type Package

Title single-cell higher order testing

Version 1.22.0

Description Single cell Higher Order Testing (scHOT) is an R package that facilitates testing changes in higher order structure of gene expression along either a developmental trajectory or across space. scHOT is general and modular in nature, can be run in multiple data contexts such as along a continuous trajectory, between discrete groups, and over spatial orientations; as well as accommodate any higher order measurement such as variability or correlation. scHOT meaningfully adds to first order effect testing, such as differential expression, and provides a framework for interrogating higher order interactions from single cell data.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.0)

Imports S4Vectors (>= 0.24.3), SingleCellExperiment, Matrix, SummarizedExperiment, IRanges, methods, stats, BiocParallel, reshape, ggplot2, igraph, grDevices, ggforce, graphics

RoxygenNote 7.1.0

Suggests knitr, markdown, rmarkdown, scater, scattermore, scales, matrixStats, deldir

VignetteBuilder knitr

biocViews GeneExpression, RNASeq, Sequencing, SingleCell, Software, Transcriptomics

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

git_branch RELEASE_3_22

git_last_commit a61146b

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2026-02-20

Author Shila Ghazanfar [aut, cre],
Yingxin Lin [aut]

Maintainer Shila Ghazanfar <`shazanfar@gmail.com`>

Contents

`liver` *Liver trajectory example data*

Description

A liver data set contains two branches (hepatocyte and cholangiocyte).

Usage

```
data(liver, package = 'scHOT')
```

Format

A ‘list‘ object.

`MOB_subset` *MOB_subset spatial example data*

Description

A MOB spatial data set.

Usage

```
data(MOB_subset, package = 'scHOT')
```

Format

An object of class `list` of length 1.

`params<-` *Setter functions for scHOT objects*

Description

Setter functions for scHOT objects

Usage

```
params(x) <- value
```

Arguments

`x`	A scHOT object
`value`	The value the slot should take

Value

A scHOT object

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
assayName = "logcounts",
positionType = "spatial",
positionColData = c("x", "y"))

params = list(higherOrderFunction = weightedSpearman,
higherOrderFunctionType = "weighted")

params(scHOT_spatial) <- params
```

`plotColouredExpression` *plotColouredExpression*

Description

the plotColouredExpression function plots an n-panel scatterplot of the gene pairs split by early, mid, and late in the sample ordering.

Usage

```
plotColouredExpression(
  scHOT,
  genes,
  genes_delimeter = "_",
  branches = NULL,
  ranked_by = NULL,
  subsetBranch = NULL,
  n = 3,
  fittedline = TRUE,
  assayName = NULL
)
```

Arguments

`scHOT`	A scHOT object.
`genes`	is either a single character string with a delimeter, or a length two character vector
`genes_delimeter`	is the delimeter to split into two gene names if genes is provided as a single character
`branches`	A character indicates that the colnames stored the branch information in colData
`ranked_by`	A character indicates that the colnames stored the ranking information of the cells in colData, such as trajectory time, if it is NULL, it will be ranked based on the branch information.
`subsetBranch`	subsetBranch is a character vector containing the names of the branches to be plotted. If NULL it will plot all branches
`n`	number of panels to split ranked samples into, default 3.
`fittedline`	logical default TRUE, add a lm straight line to the plot
`assayName`	the name of the assay that are used to plot.

Value

`ggplot` a ggplot object of scatterplots of expression split by sample ordering

Examples

```
data(liver)

scHOT_traj <- scHOT_buildFromMatrix(
mat = liver$liver_branch_hep,
cellData = list(pseudotime = liver$liver_pseudotime_hep),
positionType = "trajectory",
positionColData = "pseudotime")

scHOT_traj

plotColouredExpression(scHOT_traj, c("Cdt1","Top2a"), n = 5)
```

`plotEgoNetwork` *plotEgoNetwork*

Description

the plotEgoNetwork function plots network graphs with edges coloured by weights in the network

Usage

```
plotEgoNetwork(
  scHOT,
  hubnode,
  network,
  weight = "higherOrderStatistic",
  subset = FALSE,
  thresh = NULL
)
```

Arguments

`scHOT`	a scHOT object
`hubnode`	is a character vector of node(s) to include as hub nodes
`network`	is an igraph network
`weight`	A string indicates the column name stored in scHOT_output slot that are used as the weights of the network
`subset`	is a logical asking if you should subset based on the weight (default FALSE)
`thresh`	is the subset weight threshold

Value

`igraph` object containing the network graphed. Produces an igraph plot

---

`plotHigherOrderSequence`

plotHigherOrderSequence

Description

the plotHigherOrderSequence function plots weighted higher order statistic vectors (stored in higherOrderSequence) as line plots

Usage

```
plotHigherOrderSequence(
  scHOT,
  gene,
  positionType = NULL,
  branches = NULL,
  positionColData = NULL
)
```

Arguments

`scHOT`	A scHOT object with higherOrderSequence in scHOT_output slot
`gene`	is either a logical vector matching rows of entries in wcorsList, or a character of a gene
`positionType`	A string indicates the position type, either trajectory or spatial
`branches`	A character indicates that the colnames stored the branch information in colData (for trajectory type of data)
`positionColData`	A vector indicates column names of colData that stored the postion informaton (for spatial type of data)

Value

`ggplot` object with line plots

Examples

```
data(liver)

scHOT_traj <- scHOT_buildFromMatrix(
mat = liver$liver_branch_hep,
cellData = list(pseudotime = liver$liver_pseudotime_hep),
positionType = "trajectory",
positionColData = "pseudotime")
scHOT_traj
plotColouredExpression(scHOT_traj, c("Cdt1","Top2a"), n = 5)

scHOT_traj <- scHOT_addTestingScaffold(scHOT_traj,
t(as.matrix(c("Cdt1", "Top2a"))))
scHOT_traj <- scHOT_setWeightMatrix(scHOT_traj,
positionColData = c("pseudotime"),
```

```
positionType = "trajectory",
nrow.out = NULL,
span = 0.25)
scHOT_traj <- scHOT_calculateGlobalHigherOrderFunction(scHOT_traj,
                                                       higherOrderFunction =
                                                       weightedSpearman,
                                                      higherOrderFunctionType =
                                                      "weighted")

scHOT_traj <- scHOT_calculateHigherOrderTestStatistics(scHOT_traj,
                                                       higherOrderSummaryFunction =
                                                       sd)

slot(scHOT_traj, "scHOT_output")
plotHigherOrderSequence(scHOT_traj, c("Cdt1_Top2a"))
```

`plotOrderedExpression` *plotOrderedExpression*

Description

the plotOrderedExpression function plots expression vectors along branches and genes as ribbon plots

Usage

```
plotOrderedExpression(
  scHOT,
  genes,
  positionType = NULL,
  branches = NULL,
  ranked_by = NULL,
  xvals = NULL,
  subsetBranch = NULL,
  facet = FALSE,
  positionColData = NULL,
  assayName = NULL
)
```

Arguments

`scHOT`	A scHOT object, where the expression data is stored in the assay slot, with assay name "expression".
`genes`	is a character vector for gene names
`positionType`	A string indicates the position type, either trajectory or spatial
`branches`	A character indicates that the colnames stored the branch information in colData
`ranked_by`	A character indicates that the colnames stored the ranking information of the cells in colData, such as trajectory time
`xvals`	A character indicates that the colnames stored in colData of the x-values associated with the samples in branchData

`subsetBranch`	subsetBranch is a character vector containing the names of the branches to be plotted. If NULL it will plot all branches
`facet`	can either be FALSE, "branch", "gene", or "both"
`positionColData`	A vector indicates column names of colData that stored the postion informaton (for spatial type of data)
`assayName`	the name of the assay that are used to plot.

Value

`ggplot` a ggplot object for ribbon plot with points

`positionColData<-` *Setter functions for scHOT objects*

Description

Setter functions for scHOT objects

Usage

```
positionColData(x) <- value
```

Arguments

`x`	A scHOT object
`value`	The value the slot should take

Value

A scHOT object

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
assayName = "logcounts",
positionType = "spatial",
positionColData = c("x", "y"))

positionColData(scHOT_spatial) <- c("x", "y")
```

`positionType<-` *Setter functions for scHOT objects*

Description

Setter functions for scHOT objects

Usage

```
positionType(x) <- value
```

Arguments

`x`	A scHOT object
`value`	The value the slot should take

Value

A scHOT object

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
assayName = "logcounts",
positionType = "spatial",
positionColData = c("x", "y"))

positionType(scHOT_spatial) <- "spatial"
```

`scHOT` *scHOT*

Description

A wrapper function to perform scHOT

Usage

```
scHOT(
  scHOT,
  testingScaffold = NULL,
  weightMatrix = NULL,
  positionType = NULL,
  positionColData = NULL,
  nrow.out = NULL,
  averageAcrossTrajectoryTies = FALSE,
  higherOrderFunction = NULL,
```

```
  higherOrderFunctionType = NULL,
  numberPermutations = 1000,
  numberScaffold = 100,
  storePermutations = TRUE,
  higherOrderSummaryFunction = sd,
  parallel = FALSE,
  BPPARAM = BiocParallel::SerialParam(),
  usenperm_estimate = FALSE,
  nperm_estimate = 10000,
  maxDist = 0.1,
  plot = FALSE,
  verbose = TRUE,
  ...
)
```

Arguments

`scHOT`	A scHOT object
`testingScaffold`	A matrix with rows for each testing combination
`weightMatrix`	A matrix indicates the weight matrix for scHOT analysis
`positionType`	A string indicating the position type, either "trajectory" or "spatial"
`positionColData`	Either trajectory or spatial information for each sample. If positionType is "trajectory" then positionColData should be a character or numeric indicating the subset of colData of the scHOT object. If positionType is "spatial" then positionColData should be a character or numeric vector indicating the subset of colData that give the full spatial coordinates.
`nrow.out`	The number of weightings to include for testing, a smaller value is faster for computation
`averageAcrossTrajectoryTies`	Logical indicating whether ties in the trajectory should be given the same local weights
`higherOrderFunction`	A function object indicates the higher order function
`higherOrderFunctionType`	is "weighted" or "unweighted", determines if there is a weighting argument in the higher order function
`numberPermutations`	The number of permutations, set as 1000 by default
`numberScaffold`	The number of testing scaffolds to perform permutations, set as 100 by default
`storePermutations`	a logical flag on whether permutation values should be saved
`higherOrderSummaryFunction`	A functon indicating the higher order summary function (default is standard deviation ‘sd‘)
`parallel`	A logical input indicating whether to run the permutation test using multiple cores in parallel.
`BPPARAM`	A `BiocParallelParam` class object from the `BiocParallel` package is used. Default is SerialParam().

`usenperm_estimate`	Logical (default FALSE) if number of neighbouring permutations should be used to estimate P-values, or if difference of global higher order statistic should be used
`nperm_estimate`	Number of neighbouring permutations to use for p-value estimation
`maxDist`	max difference of global higher order statistic to use for p-value estimation (default 0.1)
`plot`	A logical input indicating whether the results are plotted
`verbose`	A logical input indicating whether the intermediate steps will be printed
`...`	parameters for function trajectoryWeightMatrix or spatialWeightMatrix

Value

A scHOT object

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                    assayName = "logcounts",
                                    positionType = "spatial",
                                    positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)
rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")

scHOT_spatial <- scHOT(scHOT_spatial,
                       testingScaffold = pairs,
                       positionType = "spatial",
                       positionColData = c("x", "y"),
                       nrow.out = NULL,
                       higherOrderFunction = weightedSpearman,
                       higherOrderFunctionType = "weighted",
                       numberPermutations = 100,
                       higherOrderSummaryFunction = sd,
                       parallel = FALSE,
                       verbose = TRUE,
                       span = 0.05)
```

`scHOT-class` *scHOT class*

Description

scHOT class

Slots

`testingScaffold` A matrix with rows for each testing combination

`weightMatrix` A matrix or dgCMatrix indicates the weight matrix

`scHOT_output` A data.frame or DtatFrame to store output from scHOT

`params`	A list of parameters
`positionType`	A character indicates the type of the position, either trajectory or spatial
`positionColData`	A vector indicates column names of colData that stored the postion informaton

---

`scHOT_addTestingScaffold` *scHOT_addTestingScaffold*

Description

Add a testing scaffold to a scHOT object

Usage

```
scHOT_addTestingScaffold(scHOT, testingScaffold)
```

Arguments

`scHOT`	A scHOT object
`testingScaffold`	A matrix with rows for each testing combination, and columns for level of dimensionality (1 for single gene etc.)

Value

A scHOT object with slot testingScaffold saved

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
assayName = "logcounts",
positionType = "spatial",
positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)
scHOT_spatial <- scHOT_addTestingScaffold(scHOT_spatial, pairs)
```

`scHOT_buildFromMatrix` *scHOT_buildFromMatrix*

Description

Create scHOT object from a matrix

Usage

```
scHOT_buildFromMatrix(
  mat,
  cellData = NULL,
  positionType = NULL,
  positionColData = NULL
)
```

Arguments

`mat`	A matrix with rows for genes and columns for cells
`cellData`	A dataframe or DataFrame object with rows for cells
`positionType`	A string indicating the position type, either "trajectory" or "spatial"
`positionColData`	Strings indicate the position information stored in colData. If positionType is "trajectory" then positionColData should be a sortable vector if positionType is "spatial" then positionColData should be a matrix type object.

Value

A scHOT object

Examples

```
dat <- rbind(rnorm(50), rnorm(50), rnorm(50))
colnames(dat) <- paste0("cell_", 1:ncol(dat))
rownames(dat) <- c("gene_1","gene_2", "gene_2")

scHOT <- scHOT_buildFromMatrix(dat, cellData = data.frame(1:ncol(dat)))
```

`scHOT_buildFromSCE` *scHOT_buildFromSCE*

Description

Create scHOT object from a SingleCellExperiment object

Usage

```
scHOT_buildFromSCE(
  sce,
  assayName = "counts",
  positionType = NULL,
  positionColData = NULL
)
```

Arguments

`sce`	A SingleCellExperiment object
`assayName`	is a single assay to pull out from sce as the expression matrix input of scHOT
`positionType`	A string indicates the position type, either trajectory or spatial
`positionColData`	Strings indicate the position information stored in colData. If positionType is "trajectory" then positionColData should be a sortable vector if positionType is "spatial" then positionColData should be a matrix type object.

Value

A scHOT object

Examples

```
library(SingleCellExperiment)
dat <- rbind(rnorm(50), rnorm(50), rnorm(50))
colnames(dat) <- paste0("cell_", 1:ncol(dat))
rownames(dat) <- c("gene_1","gene_2", "gene_2")

sce <- SingleCellExperiment::SingleCellExperiment(assays =
S4Vectors::SimpleList(counts = dat))
scHOT <- scHOT_buildFromSCE(sce)
```

`scHOT_calculateGlobalHigherOrderFunction` *scHOT_calculateGlobalHigherOrderFunction*

Description

this calculates the global higher order function and stores it in the output if these aren't found in the params slot then they need to be specified here

Usage

```
scHOT_calculateGlobalHigherOrderFunction(
  scHOT,
  higherOrderFunction = NULL,
  higherOrderFunctionType = NULL
)
```

Arguments

`scHOT`	A scHOT object
`higherOrderFunction`	A function object indicating the higher order function
`higherOrderFunctionType`	is "weighted" or "unweighted", determines if there is a weighting argument in the higher order function

Details

Calculates the global higher order function

Value

A scHOT object with scHOT_output$globalHigherOrderFunction in slot scHOT_output saved

Examples

```
 data(MOB_subset)
 sce_MOB_subset <- MOB_subset$sce_MOB_subset
 scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                     assayName = "logcounts",
                                    positionType = "spatial",
                                     positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)
rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")
scHOT_spatial <- scHOT_addTestingScaffold(scHOT_spatial, pairs)
 scHOT_spatial <- scHOT_setWeightMatrix(scHOT_spatial,
                                       positionColData = c("x","y"),
                                        positionType = "spatial",
                                        nrow.out = NULL,
                                        span = 0.05)
scHOT_spatial <- scHOT_calculateGlobalHigherOrderFunction(
  scHOT_spatial,
  higherOrderFunction = weightedSpearman,
  higherOrderFunctionType = "weighted")
```

`scHOT_calculateHigherOrderTestStatistics` *scHOT_calculateHigherOrderTestStatistics*

Description

Calculate and store the higherOrderSequence and higherOrderTestStatistic

Usage

```
scHOT_calculateHigherOrderTestStatistics(
  scHOT,
  higherOrderSummaryFunction = stats::sd,
  ...
)
```

Arguments

`scHOT`	A scHOT object
`higherOrderSummaryFunction`	A function which indicates how the higher order sequence is summarised, default is sd
`...`	parameters for higherOrderSummaryFunction

Value

scHOT A scHOT object with results stored in scHOT_output slot

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                     assayName = "logcounts",
                                    positionType = "spatial",
                                     positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)

rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")
scHOT_spatial <- scHOT_addTestingScaffold(scHOT_spatial, pairs)

scHOT_spatial <- scHOT_setWeightMatrix(scHOT_spatial,
                                       positionColData = c("x","y"),
                                        positionType = "spatial",
                                        nrow.out = NULL,
                                        span = 0.05)
scHOT_spatial <- scHOT_calculateGlobalHigherOrderFunction(
  scHOT_spatial,
  higherOrderFunction = weightedSpearman,
  higherOrderFunctionType = "weighted")
scHOT_spatial <- scHOT_setPermutationScaffold(scHOT_spatial,
                                              numberPermutations = 100)
scHOT_spatial <- scHOT_calculateHigherOrderTestStatistics(
  scHOT_spatial,
  higherOrderSummaryFunction = sd)
```

`scHOT_estimatePvalues` *scHOT_estimatePvalues*

Description

Estimate p-values based on already run permutation tests

Usage

```
scHOT_estimatePvalues(
  scHOT,
  usenperm_estimate = FALSE,
  nperm_estimate = 10000,
```

```
  maxDist = 0.1,
  plot = FALSE,
  verbose = FALSE
)
```

Arguments

`scHOT`	A scHOT object
`usenperm_estimate`	Logical (default FALSE) if number of neighbouring permutations should be used, or if difference of global higher order statistic should be used
`nperm_estimate`	Number of neighbouring permutations to use for p-value estimation
`maxDist`	max difference of global higher order statistic to use for p-value estimation (default 0.1)
`plot`	A logical input indicating whether the results are plotted
`verbose`	A logical input indicating whether the intermediate steps will be printed

Value

scHOT A scHOT object with results stored in scHOT_output slot

Examples

```
 data(MOB_subset)
 sce_MOB_subset <- MOB_subset$sce_MOB_subset
 scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                     assayName = "logcounts",
                                    positionType = "spatial",
                                     positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)
rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")

scHOT_spatial <- scHOT_addTestingScaffold(scHOT_spatial, pairs)

scHOT_spatial <- scHOT_setWeightMatrix(scHOT_spatial,
                                       positionColData = c("x","y"),
                                        positionType = "spatial",
                                        nrow.out = NULL,
                                        span = 0.05)
scHOT_spatial <- scHOT_calculateGlobalHigherOrderFunction(
  scHOT_spatial,
  higherOrderFunction = weightedSpearman,
  higherOrderFunctionType = "weighted")
scHOT_spatial <- scHOT_setPermutationScaffold(scHOT_spatial,
                                              numberPermutations = 100)
scHOT_spatial <- scHOT_calculateHigherOrderTestStatistics(
  scHOT_spatial,
  higherOrderSummaryFunction = sd)

scHOT_spatial <- scHOT_performPermutationTest(
  scHOT_spatial,
  verbose = TRUE,
  parallel = FALSE)
```

```
scHOT_spatial <- scHOT_estimatePvalues(scHOT_spatial)
```

---

scHOT_output<- *Setter functions for scHOT objects*

Description

Setter functions for scHOT objects

Usage

```
scHOT_output(x) <- value
```

Arguments

`x`	A scHOT object
`value`	The value the slot should take

Value

A scHOT object

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
assayName = "logcounts",
positionType = "spatial",
positionColData = c("x", "y"))

scHOT_output(scHOT_spatial) <- data.frame()
```

---

scHOT_performPermutationTest *scHOT_performPermutationTest*

Description

Perform permutation test

Usage

```
scHOT_performPermutationTest(
  scHOT,
  verbose = FALSE,
  parallel = FALSE,
  BPPARAM = BiocParallel::SerialParam()
)
```

Arguments

`scHOT`	A scHOT object
`verbose`	A logical input indicates whether the intermediate steps will be printed
`parallel`	A logical input indicates whether run the permutation test using multiple cores in parallel.
`BPPARAM`	A `BiocParallelParam` class object from the `BiocParallel` package is used. Default is SerialParam().

Value

scHOT A scHOT object with results stored in scHOT_output slot

Examples

```
 data(MOB_subset)
 sce_MOB_subset <- MOB_subset$sce_MOB_subset
 scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                    assayName = "logcounts",
                                   positionType = "spatial",
                                    positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)
rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")
scHOT_spatial <- scHOT_addTestingScaffold(scHOT_spatial, pairs)
 scHOT_spatial <- scHOT_setWeightMatrix(scHOT_spatial,
                                      positionColData = c("x","y"),
                                       positionType = "spatial",
                                       nrow.out = NULL,
                                       span = 0.05)
scHOT_spatial <- scHOT_calculateGlobalHigherOrderFunction(
  scHOT_spatial,
  higherOrderFunction = weightedSpearman,
  higherOrderFunctionType = "weighted")
scHOT_spatial <- scHOT_setPermutationScaffold(scHOT_spatial,
                                             numberPermutations = 100)
scHOT_spatial <- scHOT_calculateHigherOrderTestStatistics(
  scHOT_spatial,
  higherOrderSummaryFunction = sd)

scHOT_spatial <- scHOT_performPermutationTest(
  scHOT_spatial,
  verbose = TRUE,
  parallel = FALSE)
```

`scHOT_plotPermutationDistributions`

scHOT_plotPermutationDistributions

Description

the scHOT_plotPermutationDistributions function plots the permutation test statistics as a diagnostic plot for estimating p-values

Usage

```
scHOT_plotPermutationDistributions(scHOT)
```

Arguments

`scHOT`	a scHOT object

Value

`ggplot` graph of global higher order function and the permutation scHOT test statistics. This should have a continuous pattern to be reliably used for p-value estimation

---

`scHOT_setPermutationScaffold` *scHOT_setPermutationScaffold*

Description

Set permutation scaffold

Usage

```
scHOT_setPermutationScaffold(
  scHOT,
  numberPermutations = 1000,
  numberScaffold = 100,
  storePermutations = TRUE
)
```

Arguments

`scHOT`	A scHOT object
`numberPermutations`	The number of permutations, set as 1000 by default
`numberScaffold`	The number of scaffold, set as 100 by default, minimum 6. if you want all combinations to do permutations then set, numberScaffold much higher than the testingScaffold
`storePermutations`	a logical flag on whether Permutations should be stored, or discarded once used

Value

A scHOT object with storePermutations in slot scHOT_output saved

Examples

```
 data(MOB_subset)
 sce_MOB_subset <- MOB_subset$sce_MOB_subset
 scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                    assayName = "logcounts",
                                   positionType = "spatial",
                                    positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)
rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")
scHOT_spatial <- scHOT_addTestingScaffold(scHOT_spatial, pairs)
scHOT_spatial <- scHOT_setWeightMatrix(scHOT_spatial,
                                       positionColData = c("x","y"),
                                        positionType = "spatial",
                                        nrow.out = NULL,
                                        span = 0.05)
scHOT_spatial <- scHOT_calculateGlobalHigherOrderFunction(
  scHOT_spatial,
  higherOrderFunction = weightedSpearman,
  higherOrderFunctionType = "weighted")
scHOT_spatial <- scHOT_setPermutationScaffold(scHOT_spatial,
                                              numberPermutations = 100)
```

`scHOT_setWeightMatrix` *scHOT_setWeightMatrix*

Description

Create scHOT object from a SingleCellExperiment object

Usage

```
scHOT_setWeightMatrix(
  scHOT,
  weightMatrix = NULL,
  positionType = NULL,
  positionColData = NULL,
  nrow.out = NULL,
  averageAcrossTrajectoryTies = FALSE,
  ...
)
```

Arguments

`scHOT`	A scHOT object
`weightMatrix`	A matrix indicating the weight matrix for scHOT analysis, such as the output from ‘trajectoryWeightMatrix‘ or ‘spatialWeightMatrix‘. If this is not NULL then other parameters are ignored.
`positionType`	A string indicating the position type, either "trajectory" or "spatial"

`positionColData`	Either trajectory or spatial information for each sample. If positionType is "trajectory" then positionColData should be a character or numeric indicating the subset of colData of the scHOT object. If positionType is "spatial" then positionColData should be a character or numeric vector indicating the subset of colData that give the full spatial coordinates.
`nrow.out`	The number of weightings to include for testing, a smaller value is faster for computation
`averageAcrossTrajectoryTies`	Logical indicating whether ties in the trajectory should be given the same local weights
`...`	parameters for function trajectoryWeightMatrix or spatialWeightMatrix

Value

A scHOT object with slot weightMatrix saved

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                    assayName = "logcounts",
                                   positionType = "spatial",
                                    positionColData = c("x", "y"))
pairs <- matrix(c("Arrb1", "Mtor", "Dnm1l", "Gucy1b3"), ncol = 2, byrow = TRUE)
rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")
scHOT_spatial <- scHOT_addTestingScaffold(scHOT_spatial, pairs)
scHOT_spatial <- scHOT_setWeightMatrix(scHOT_spatial,
                                       positionColData = c("x","y"),
                                        positionType = "spatial",
                                        nrow.out = NULL,
                                        span = 0.05)
```

`scHOT_stripOutput` *scHOT_stripOutput*

Description

Strip the scHOT output

Usage

```
scHOT_stripOutput(scHOT, force = TRUE, store = FALSE, file_name = NULL)
```

Arguments

`scHOT`	A scHOT object
`force`	A logical indicates whther forcing stripping the scHOT output
`store`	A logical flag on whether the scHOT should be stored as .rds file
`file_name`	A string indicates the file name of the scHOT will be stored

Value

A scHOT object with scHOT_output striped

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
                                     assayName = "logcounts",
                                    positionType = "spatial",
                                     positionColData = c("x", "y"))

scHOT_spatial <- scHOT_stripOutput(scHOT_spatial)
```

`spatialWeightMatrix` *spatialWeightMatrix*

Description

Create weight matrix for spatial data

Usage

```
spatialWeightMatrix(x, span = NULL)
```

Arguments

`x`	a matrix with rows corresponding to cells and columns corresponding to dimensions to calculate Euclidean distance
`span`	proportion of samples to include on either side, default is 13/(number of rows in ‘x‘), corresponding roughly to points within a diamond shape distance away

Value

A weighted matrix

Examples

```
spat_x <- rnorm(50)
spat_y <- rnorm(50)
spat_coord <- cbind(spat_x, spat_y)
W <- spatialWeightMatrix(spat_coord)
```

`testingScaffold<-` *Setter functions for scHOT objects*

Description

Setter functions for scHOT objects

Usage

```
testingScaffold(x) <- value
```

Arguments

`x`	A scHOT object
`value`	The value the slot should take

Value

A scHOT object

Examples

```
data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
assayName = "logcounts",
positionType = "spatial",
positionColData = c("x", "y"))

pairs <- t(combn(rownames(sce_MOB_subset),2))
rownames(pairs) <- apply(pairs,1,paste0,collapse = "_")

testingScaffold(scHOT_spatial) <- pairs
```

`thin` *thin*

Description

The thin function extracts the rows of a matrix evenly so that roughly n number of rows remain. Used for thinning down the weight matrix to speed up overall computation.

Usage

```
thin(W, n = 100)
```

Arguments

`W`	matrix
`n`	rough number of rows to keep

Value

`matrix` of thinned matrix keeping only roughly n rows.

Examples

```
W = trajectoryWeightMatrix(500)
W_small = thin(W, n = 100)
```

`trajectoryWeightMatrix` *trajectoryWeightMatrix*

Description

Create weight matrix for trajectory data

Usage

```
trajectoryWeightMatrix(n, type = NULL, span = NULL)
```

Arguments

`n`	indicates the number of cels
`type`	Type of weight matrix, one of "triangular" (default), "block", and "harmonic"
`span`	proportion of samples to include on either side, default is 0.25

Value

A weighted matrix

Examples

```
W <- trajectoryWeightMatrix(100)
W <- trajectoryWeightMatrix(100, type = "triangular")
W <- trajectoryWeightMatrix(100, type = "block")
W <- trajectoryWeightMatrix(100, type = "harmonic")
```

`weightedPearson` *weightedPearson*

Description

the weightedPearson function

Usage

```
weightedPearson(x, y, w = 1)
```

Arguments

`x`	x and y are data vectors
`y`	x and y are data vectors
`w`	weight vector, values should be between 0 and 1

Value

`numeric` weighted correlation value between x and y

Examples

```
x = rnorm(100)
y = rnorm(100)
w = runif(100)
weightedPearson(x,y,w)
```

`weightedSpearman` *weightedSpearman*

Description

the weightedSpearman function

Usage

```
weightedSpearman(x, y, w = 1)
```

Arguments

`x`	x and y are data vectors
`y`	x and y are data vectors
`w`	weight vector, values should be between 0 and 1

Value

`numeric` weighted correlation value between x and y

Examples

```
x = rnorm(100)
y = rnorm(100)
w = runif(100)
weightedSpearman(x,y,w)
```

`weightedVariance` *weightedVariance*

Description

the weightedVariance function

Usage

```
weightedVariance(x, y = NULL, w)
```

Arguments

`x`	x is a data vector
`y`	default to NULL, if given it is ignored
`w`	weight vector, values should be between 0 and 1

Value

`numeric` weighted variance value for x

Examples

```
x = rnorm(100)
w = runif(100)
weightedVariance(x,w = w)
```

`weightedZIKendall` *weightedZIKendall*

Description

the weightedZIKendall function calculates weighted Tau*, where Tau* is described in Pimentel et al (2015) doi:10.1016/j.spl.2014.09.002. This association measure is defined for zero-inflated, non-negative random variables.

Usage

```
weightedZIKendall(x, y, w = 1)
```

Arguments

`x`	x and y are non-negative data vectors
`y`	x and y are non-negative data vectors
`w`	weight vector, values should be between 0 and 1

Value

`numeric` weighted Tau* association value between x and y

Examples

```
x = pmax(0,rnorm(100))
y = pmax(0,rnorm(100))
w = runif(100)
weightedZIKendall(x,y,w)
```

`weightedZISpearman` *weightedZISpearman*

Description

the weightedZISpearman function calculates weighted rho*, where rho* is described in Pimentel et al (2009). This association measure is defined for zero-inflated, non-negative random variables.

Usage

```
weightedZISpearman(x, y, w = 1)
```

Arguments

`x`	x and y are non-negative data vectors
`y`	x and y are non-negative data vectors
`w`	weight vector, values should be between 0 and 1

Value

`numeric` weighted rho* association value between x and y

Pimentel, Ronald Silva, "Kendall's Tau and Spearman's Rho for Zero-Inflated Data" (2009). Dissertations. 721. https://scholarworks.wmich.edu/dissertations/721

Examples

```
x = pmax(0,rnorm(100))
y = pmax(0,rnorm(100))
w = runif(100)
weightedZISpearman(x,y,w)
```

`weightMatrix<-` *Setter functions for scHOT objects*

Description

Setter functions for scHOT objects

Usage

```
weightMatrix(x) <- value
```

Arguments

`x`	A scHOT object
`value`	The value the slot should take

Value

A scHOT object

Examples

```
library(SingleCellExperiment)

data(MOB_subset)
sce_MOB_subset <- MOB_subset$sce_MOB_subset
scHOT_spatial <- scHOT_buildFromSCE(sce_MOB_subset,
assayName = "logcounts",
positionType = "spatial",
positionColData = c("x", "y"))

W <- spatialWeightMatrix(colData(scHOT_spatial)[,slot(scHOT_spatial, "positionColData")])

weightMatrix(scHOT_spatial) <- W
```

Index