Package ‘spicyR’

September 24, 2025

Type Package

Title Spatial analysis of in situ cytometry data

Version 1.21.8

Description The spicyR package provides a framework for performing inference on changes in spatial relationships between pairs of cell types for cell-resolution spatial omics technologies. spicyR consists of three primary steps: (i) summarizing the degree of spatial localization between pairs of cell types for each image; (ii) modelling the variability in localization summary statistics as a function of cell counts and (iii) testing for changes in spatial localizations associated with a response variable.

License GPL (>=2)

LazyData true

biocViews SingleCell, CellBasedAssays, Spatial

Encoding UTF-8

Depends R (>= 4.1)

VignetteBuilder knitr

BugReports https://github.com/SydneyBioX/spicyR/issues

URL https://ellispatrick.github.io/spicyR/
https://github.com/SydneyBioX/spicyR,
https://sydneybiox.github.io/spicyR/

Imports BiocParallel, ClassifyR, S4Vectors, SingleCellExperiment, SpatialExperiment, SummarizedExperiment, cli, concaveman, coxme, data.table, dplyr, ggforce, ggh4x, ggnewscale, ggplot2, ggthemes, grDevices, lifecycle, lmerTest, magrittr, methods, pheatmap, rlang, scales, scam, simpleSeg, spatstat.explore, spatstat.geom, stats, survival, tibble, tidyr

Suggests SpatialDatasets, BiocStyle, knitr, rmarkdown, pkgdown, imcRtools, testthat (>= 3.0.0)

RoxygenNote 7.3.3

Config/testthat/edition 3

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

git_branch devel

git_last_commit 21dfb33

git_last_commit_date 2025-09-19

Repository Bioconductor 3.22

Date/Publication 2025-09-23

Author Nicolas Canete [aut],
Ellis Patrick [aut, cre],
Nicholas Robertson [ctb],
Alex Qin [ctb],
Farhan Ameen [ctb],
Shreya Rao [ctb]

Maintainer Ellis Patrick `<ellis.patrick@sydney.edu.au>`

Contents

bind 2
colTest 3
convPairs 4
diabetesData 5
getPairwise 5
getProp 7
imageCrossPlot 7
plotImage 9
signifPlot 10
spicyBoxPlot 11
SpicyResults-class 12
spicyTest 14
survBubble 14
topPairs 15

Index **17**

`bind` *Produces a dataframe showing L-function metric for each imageID entry.*

Description

Produces a dataframe showing L-function metric for each imageID entry.

Usage

```
bind(results, pairName = NULL)
```

Arguments

`results`	Spicy test result obtained from spicy.
`pairName`	A string specifying the pairwise interaction of interest. If NULL, all pairwise interactions are shown.

Value

A data.frame containing the colData related to the results.

Examples

```
data(spicyTest)
df <- bind(spicyTest)
```

---

colTest *Perform a simple wilcoxon-rank-sum test or t-test on the columns of a data frame*

Description

Perform a simple wilcoxon-rank-sum test or t-test on the columns of a data frame

Usage

```
colTest(df, condition, type = NULL, feature = NULL, imageID = "imageID")
```

Arguments

`df`	A data.frame or SingleCellExperiment, SpatialExperiment
`condition`	The condition of interest
`type`	The type of test, "wilcox", "ttest" or "survival".
`feature`	Can be used to calculate the proportions of this feature for each image
`imageID`	The imageID's if presenting a SingleCellExperiment

Value

Proportions

Examples

```
# Test for an association with long-duration diabetes
# This is clearly ignoring the repeated measures...
data("diabetesData")
diabetesData <- spicyR:::.format_data(
  diabetesData, "imageID", "cellType", c("x", "y"), FALSE
)
props <- getProp(diabetesData)
condition <- spicyR:::getImagePheno(diabetesData)$stage
names(condition) <- spicyR:::getImagePheno(diabetesData)$imageID
condition <- condition[condition %in% c("Long-duration", "Onset")]
test <- colTest(props[names(condition), ], condition)
```

`convPairs` *Converts colPairs object into an abundance matrix based on number of nearby interactions for every cell type.*

Description

Converts colPairs object into an abundance matrix based on number of nearby interactions for every cell type.

Usage

```
convPairs(cells, colPair, imageID = "imageID", cellType = "cellType")
```

Arguments

`cells`	A SingleCellExperiment that contains objects in the colPairs slot.
`colPair`	The name of the object in the colPairs slot for which the dataframe is constructed from.
`imageID`	The image ID if using SingleCellExperiment.
`cellType`	The cell type if using SingleCellExperiment.

Value

Matrix of abundances

Examples

```
data("diabetesData")
images <- c("A09", "A11", "A16", "A17")
diabetesData <- diabetesData[
  , SummarizedExperiment::colData(diabetesData)$imageID %in% images
]

diabetesData_SPE <- SpatialExperiment::SpatialExperiment(diabetesData,
  colData = SummarizedExperiment::colData(diabetesData)
)
SpatialExperiment::spatialCoords(diabetesData_SPE) <- data.frame(
  SummarizedExperiment::colData(diabetesData_SPE)$x,
  SummarizedExperiment::colData(diabetesData_SPE)$y
) |>
  as.matrix()

SpatialExperiment::spatialCoordsNames(diabetesData_SPE) <- c("x", "y")

diabetesData_SPE <- imcRtools::buildSpatialGraph(diabetesData_SPE,
  img_id = "imageID",
  type = "knn",
  k = 20,
  coords = c("x", "y")
)

pairAbundances <- convPairs(diabetesData_SPE,
```

```
  colPair = "knn_interaction_graph"
)
```

---

diabetesData *Diabetes IMC data in SCE format.*

Description

This is a subset of the Damond et al 2019 imaging mass cytometry dataset. The data contains cells in the pancreatic islets of individuals with early onset diabetes and healthy controls. The object contains single-cell data of 160 images from 8 subjects, with 20 images per subject.

Usage

```
data("diabetesData")
```

Format

diabetesData_SCE a SingleCellExperiment object

Details

Converted into a SingleCellExperiment format.

---

getPairwise *Get statistic from pairwise L curve of a single image.*

Description

Get statistic from pairwise L curve of a single image.

Usage

```
getPairwise(
  cells,
  imageID = "imageID",
  cellType = "cellType",
  spatialCoords = c("x", "y"),
  r = NULL,
  sigma = NULL,
  from = NULL,
  to = NULL,
  cores = 1,
  minLambda = 0.05,
  window = "convex",
  window.length = NULL,
  edgeCorrect = TRUE,
  includeZeroCells = FALSE,
  BPPARAM = NULL,
```

```
    imageIDCol = imageID,
    cellTypeCol = cellType,
    spatialCoordCols = spatialCoords,
    nCores = cores,
    Rs = r
  )
```

Arguments

`cells`	A SummarizedExperiment that contains at least the variables x and y, giving the location coordinates of each cell, and cellType.
`imageID`	The name of the imageID column if using a SingleCellExperiment or SpatialExperiment.
`cellType`	The name of the cellType column if using a SingleCellExperiment or SpatialExperiment.
`spatialCoords`	The names of the spatialCoords column if using a SingleCellExperiment.
`r`	A vector of the radii that the measures of association should be calculated over.
`sigma`	A numeric variable used for scaling when fitting inhomogenous L-curves.
`from`	The 'from' cellType for generating the L curve.
`to`	The 'to' cellType for generating the L curve.
`cores`	Number of cores to use for parallel processing or a BiocParallel MulticoreParam or SerialParam object.
`minLambda`	Minimum value density for scaling when fitting inhomogeneous L-curves.
`window`	Should the window around the regions be 'square', 'convex' or 'concave'.
`window.length`	A tuning parameter for controlling the level of concavity when estimating concave windows.
`edgeCorrect`	A logical indicating whether to perform edge correction.
`includeZeroCells`	A logical indicating whether to include cells with zero counts in the pairwise association.
`BPPARAM`	{DEPRECATED} A BiocParallel MulticoreParam or SerialParam object.
`imageIDCol`	{DEPRECATED} The name of the imageID column if using a SingleCellExperiment or SpatialExperiment.
`cellTypeCol`	{DEPRECATED} The name of the cellType column if using a SingleCellExperiment or SpatialExperiment.
`spatialCoordCols`	{DEPRECATED} The names of the spatialCoords column if using a SingleCellExperiment.
`nCores`	{DEPRECATED} Number of cores to use for parallel processing or a BiocParallel MulticoreParam or SerialParam object.
`Rs`	{DEPRECATED} A vector of the radii that the measures of association should be calculated over. calculation.

Value

Statistic from pairwise L-curve of a single image.

Examples

```
data("diabetesData")
# Subset by imageID for fast example
selected_cells <- diabetesData[
  , SummarizedExperiment::colData(diabetesData)$imageID == "A09"
]
pairAssoc <- getPairwise(selected_cells)
```

`getProp` *Get proportions from a SummarizedExperiment.*

Description

Get proportions from a SummarizedExperiment.

Usage

```
getProp(cells, feature = "cellType", imageID = "imageID")
```

Arguments

`cells`	A SingleCellExperiment, SpatialExperiment or data.frame.
`feature`	The feature of interest
`imageID`	The imageID’s

Value

Proportions

Examples

```
data("diabetesData")
prop <- getProp(diabetesData)
```

`imageCrossPlot` *Pairwise cross-plot of spatial associations*

Description

Builds a pairwise scatterplot of marker-marker associations for a single image using the output of ‘getPairwise()‘.

Usage

```
imageCrossPlot(
  result,
  image = NULL,
  colourGradient = c("#4575B4", "white", "#D73027"),
  marksToPlot = NULL,
  limits = NULL
)
```

Arguments

`result`	A matrix or data frame produced by ‘getPairwise()‘ where rownames are image IDs and colnames are concatenated marker pairs in the form ‘"from__to"‘. Must contain at least one row with name matching ‘image‘.
`image`	Character scalar. Which image (row in ‘result‘) to plot. If ‘NULL‘, defaults to the first rowname of ‘result‘.
`colourGradient`	Character vector of length 3 giving the low, mid, and high colours for a diverging palette used by ‘scale_colour_gradient2()‘. Default is ‘c("#4575B4", "white", "#D73027")‘.
`marksToPlot`	Optional character vector of marker names. If supplied, the plot is restricted to rows and columns where both ‘from‘ and ‘to‘ are in this set.
`limits`	Numeric length-2 vector giving the lower and upper caps applied to association values for colour mapping. Use ‘NULL‘ to avoid clamping. Point sizes are clamped to ‘c(0, max(abs(limits)))‘.

Details

The function expects ‘colnames(result)‘ to contain ‘"__"‘ separating marker names; otherwise an error is thrown. Values are transformed into two aesthetics:

- `value`: the (possibly clamped) signed association used for colour.
- `size`: the (possibly clamped) absolute association used for point size.

The colourbar shows three ticks at `lower`, `0`, and `upper` with labels `"<lower"`, `"0"`, and `">upper"`.

Value

A **ggplot2** object.

See Also

`getPairwise`

Examples

```
## Not run:
# Minimal example (toy data)
set.seed(1)
mks <- c("A","B","C")
pairs <- as.vector(outer(mks, mks, paste, sep="__"))
res <- matrix(rnorm(length(pairs)*2), nrow = 2,
              dimnames = list(c("img1","img2"), pairs))

p <- imageCrossPlot(res, image = "img1",
                      colourGradient = c("#4575B4","white","#D73027"),
                      marksToPlot = c("A","B","C"),
                      limits = c(-3, 3))
print(p)

## End(Not run)
```

`plotImage` *Plots an image with specified from and to cell types.*

Description

Plots an image with specified from and to cell types.

Usage

```
plotImage(
  cells,
  imageToPlot,
  from,
  to,
  imageID = "imageID",
  cellType = "cellType",
  spatialCoords = c("x", "y")
)
```

Arguments

`cells`	A SummarizedExperiment object.
`imageToPlot`	The ID of the image to be plotted.
`from`	The "from" cell type.
`to`	The "to" cell type.
`imageID`	The name of the imageID column in the SummarizedExperiment object.
`cellType`	The name of the cellType column in the SummarizedExperiment object.
`spatialCoords`	The names of the spatialCoords column if using a SingleCellExperiment.

Value

A ggplot object.

Examples

```
data("diabetesData")
plotImage(diabetesData, "A09", from = "acinar", to = "alpha")
```

`signifPlot` *Plots result of signifPlot.*

Description

Plots result of signifPlot.

Usage

```
signifPlot(
  results,
  fdr = FALSE,
  type = "bubble",
  breaks = NULL,
  comparisonGroup = NULL,
  colours = c("#4575B4", "white", "#D73027"),
  marksToPlot = NULL,
  cutoff = 0.05,
  contextColours = NULL,
  contextLabels = waiver()
)
```

Arguments

`results`	A spicy results object
`fdr`	TRUE if FDR correction is used.
`type`	Whether to make a bubble plot or heatmap. Note: For survival results a bubble plot will be used.
`breaks`	Vector of 3 numbers giving breaks used in legend. The first number is the minimum, the second is the maximum, the third is the number of breaks.
`comparisonGroup`	A string specifying the name of the outcome group to compare with the base group.
`colours`	Vector of colours to use to colour legend.
`marksToPlot`	Vector of marks to include in plot.
`cutoff`	significance threshold for circles in bubble plot.
`contextColours`	Used for `Kontextual` results. A named list specifying the colours for each context. By default the Tableau colour palette is used.
`contextLabels`	Used for `Kontextual` results. A named list to change the default labels for each context.

Value

a ggplot or pheatmap object

Examples

```
data(spicyTest)

p <- signifPlot(spicyTest, breaks = c(-3, 3, 0.5))
# plot includes unicode characters, do not use default pdf device
ggplot2::ggsave(p, filename = tempfile(), device = cairo_pdf)
```

`spicyBoxPlot` *Plots boxplot for a specified cell-cell relationship*

Description

Plots boxplot for a specified cell-cell relationship

Usage

```
spicyBoxPlot(results, from = NULL, to = NULL, rank = NULL)
```

Arguments

`results`	Data frame obtained from spicy.
`from`	Cell type which you would like to compare to the to cell type.
`to`	Cell type which you would like to compare to the from cell type.
`rank`	Ranking of cell type in terms of p-value, the smaller the p-value the higher the rank.

Value

a ggplot2 boxplot

Examples

```
data(spicyTest)

spicyBoxPlot(spicyTest,
             rank = 1)
```

`SpicyResults-class` *Performs spatial tests on spatial cytometry data.*

Description

Performs spatial tests on spatial cytometry data.

Usage

```
spicy(
  cells,
  condition,
  subject = NULL,
  covariates = NULL,
  imageID = "imageID",
  cellType = "cellType",
  spatialCoords = c("x", "y"),
  r = NULL,
  sigma = NULL,
  from = NULL,
  to = NULL,
  alternateResult = NULL,
  cores = 1,
  minLambda = 0.05,
  weights = TRUE,
  weightsByPair = FALSE,
  weightFactor = 1,
  window = "convex",
  window.length = NULL,
  edgeCorrect = TRUE,
  includeZeroCells = FALSE,
  verbose = FALSE,
  BPPARAM = NULL,
  imageIDCol = imageID,
  cellTypeCol = cellType,
  spatialCoordCols = spatialCoords,
  nCores = cores,
  Rs = r,
  ...
)
```

Arguments

`cells`	A SummarizedExperiment or data frame that contains at least the variables x and y, giving the location coordinates of each cell, and cellType.
`condition`	A character specifying which column which contains the condition or ‘Surv‘ objects.
`subject`	Vector of subject IDs corresponding to each image if cells is a data frame.
`covariates`	Vector of covariate names that should be included in the mixed effects model as fixed effects.

`imageID`	The name of the imageID column if using a SingleCellExperiment or SpatialExperiment.
`cellType`	The name of the cellType column if using a SingleCellExperiment or SpatialExperiment.
`spatialCoords`	The names of the spatialCoords column if using a SingleCellExperiment.
`r`	A vector of the radii that the measures of association should be calculated over.
`sigma`	A numeric variable used for scaling when fitting inhomogenous L-curves.
`from`	vector of cell types which you would like to compare to the to vector.
`to`	vector of cell types which you would like to compare to the from vector.
`alternateResult`	A pairwise association statistic between each combination of celltypes in each image.
`cores`	Number of cores to use for parallel processing or a BiocParallel MulticoreParam or SerialParam object.
`minLambda`	Minimum value density for scaling when fitting inhomogeneous L-curves.
`weights`	logical indicating whether to include weights based on cell counts.
`weightsByPair`	logical indicating whether weights should be calculated for each cell type pair.
`weightFactor`	numeric that controls the convexity of the weight function.
`window`	Should the window around the regions be 'square', 'convex' or 'concave'.
`window.length`	A tuning parameter for controlling the level of concavity when estimating concave windows.
`edgeCorrect`	A logical indicating whether to perform edge correction.
`includeZeroCells`	A logical indicating whether to include cells with zero counts in the pairwise association calculation.
`verbose`	logical indicating whether to output messages.
`BPPARAM`	{DEPRECATED} A BiocParallel MulticoreParam or SerialParam object.
`imageIDCol`	{DEPRECATED} The name of the imageID column if using a SingleCellExperiment or SpatialExperiment.
`cellTypeCol`	{DEPRECATED} The name of the cellType column if using a SingleCellExperiment or SpatialExperiment.
`spatialCoordCols`	{DEPRECATED} The names of the spatialCoords column if using a SingleCellExperiment.
`nCores`	{DEPRECATED} Number of cores to use for parallel processing or a BiocParallel MulticoreParam or SerialParam object.
`Rs`	{DEPRECATED} A vector of the radii that the measures of association should be calculated over.
`...`	Other options

Value

Data frame of p-values.

Examples

```
data("diabetesData")

# Test with random effect for patient on a pairwise combination of cell
# types.
spicy(diabetesData,
  condition = "stage", subject = "case",
  from = "Tc", to = "Th"
)

# Test all pairwise combinations of cell types without random effect of
# patient.
## Not run:
spicyTest <- spicy(diabetesData, condition = "stage", subject = "case")

## End(Not run)

# Test all pairwise combination of cell types with random effect of patient.
## Not run:
spicy(diabetesData, condition = "condition", subject = "subject")

## End(Not run)
```

`spicyTest` *Results from spicy for diabetesData*

Description

Results from the call: spicyTest <- spicy(diabetesData, condition = "condition", subject = "subject")

Usage

```
data("spicyTest")
```

Format

spicyTest a spicy object

`survBubble` *Plots survival results from spicy.*

Description

Plots survival results from spicy.

Usage

```
survBubble(
  result,
  fdr = FALSE,
  cutoff = 0.05,
  colourGradient = c("#4575B4", "white", "#D73027"),
  marksToPlot = NULL,
  contextColours = NULL,
  contextLabels = waiver()
)
```

Arguments

`result`	A spicyResults object that contains survival results.
`fdr`	TRUE if FDR correction is used.
`cutoff`	Significance threshold for circles in bubble plot.
`colourGradient`	A vector of colours, used to define the low, medium, and high values for the colour scale.
`marksToPlot`	Vector of marks to include in bubble plot.
`contextColours`	Used for `Kontextual` results. A named list specifying the colours for each context. By default the Tableau colour palette is used.
`contextLabels`	Used for `Kontextual` results. A named list to change the default labels for each context.

Value

A ggplot object.

`topPairs` *A table of the significant results from spicy tests*

Description

A table of the significant results from spicy tests

Usage

```
topPairs(x, coef = NULL, n = 10, adj = "fdr", cutoff = NULL, figures = NULL)
```

Arguments

`x`	The output from spicy.
`coef`	Which coefficient to list.
`n`	Extract the top n most significant pairs.
`adj`	Which p-value adjustment method to use, argument for p.adjust().
`cutoff`	A p-value threshold to extract significant pairs.
`figures`	Round to ‘figures‘ significant figures.

Value

A data.frame

Examples

```
data(spicyTest)
topPairs(spicyTest)
```

Index

∗ **datasets**
  diabetesData, 5
  spicyTest, 14

bind, 2

colTest, 3
convPairs, 4

diabetesData, 5

getPairwise, 5, *8*
getProp, 7

imageCrossPlot, 7

Kontextual, *10*, *15*

plotImage, 9

signifPlot, 10
spicy *(*SpicyResults-class*)*, 12
spicy,spicy-method
  *(*SpicyResults-class*)*, 12
spicyBoxPlot, 11
SpicyResults,list,ANY-method
  *(*SpicyResults-class*)*, 12
SpicyResults-class, 12
spicyTest, 14
survBubble, 14

topPairs, 15
topPairs,SpicyResults-method
  *(*topPairs*)*, 15