Package 'blisa'

October 30, 2026

Type Package

Title Infer Cell-Cell Communication from Spatial Transcriptomics

Version 1.0.3

Description Identifies cell-cell communication hotspots in spatial transcriptomics data using bivariate Local Moran's I statistics on hexagonally binned cells. Provides functions for spatial weighting, ligand-receptor pair filtering, hotspot detection, and visualisation of sender-receiver cell-type interactions.

License GPL (>= 3)

URL https://github.com/ChenLaboratory/blisa, https://chenlaboratory.github.io/blisa/

BugReports https://github.com/ChenLaboratory/blisa/issues

Encoding UTF-8

Imports sf, spdep, fastLISA, Matrix, SpatialExperiment, SummarizedExperiment, ComplexHeatmap, ggplot2, viridisLite, grid

VignetteBuilder knitr

Suggests knitr, rmarkdown

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

NeedsCompilation no

Author Yunshun Chen [aut, cre], Lei Qin [aut], Lizhong Chen [aut]

Maintainer Yunshun Chen <yuchen@wehi.edu.au>

Repository CRAN

Date/Publication 2026-10-30 03:40:02 UTC

Contents

blisa-package . . . 2
blisa . . . 3
blisaPathway . . . 6
computeSpatialWeights . . . 8
filterLRpairs . . . 9
hexBinCells . . . 10
is.blisa . . . 11
plotCCI . . . 12
plotCCILR . . . 14
plotCCIspatial . . . 15
plotCCIsummary . . . 17
plotHotspots . . . 18
plotLRrank . . . 20
runCCI . . . 22
visiumSpotBins . . . 24

Index **26**

`blisa-package` *blisa: Bivariate Local Indicator of Spatial Association for spatial transcriptomics*

Description

Identifies cell-cell communication hotspots in spatial transcriptomics data using bivariate Local Moran's I statistics on hexagonally binned cells. Provides functions for spatial weighting, ligand-receptor pair filtering, hotspot detection, and visualisation of sender-receiver cell-type interactions.

Author(s)

Maintainer: Yunshun Chen <yuchen@wehi.edu.au>

Authors:

- Lei Qin <qin.l@wehi.edu.au>
- Lizhong Chen

See Also

Useful links:

- https://github.com/ChenLaboratory/blisa
- https://chenlaboratory.github.io/blisa/
- Report bugs at https://github.com/ChenLaboratory/blisa/issues

`blisa` *Run BLISA spatial cell-cell communication analysis*

Description

Generic function for running BLISA (Bivariate Local Indicator of Spatial Association). Dispatches on the class of `x`:

- `blisa.default` accepts a pre-binned gene-by-bin count matrix and a matching `bins` polygon object.
- `blisa.SpatialExperiment` accepts a cell-level `SpatialExperiment` object and bins cells into hexagonal tiles internally via `hexBinCells` before running the analysis.

Usage

```
blisa(x, ...)

## Default S3 method:
blisa(
  x,
  bins,
  LR_df = NULL,
  bin_size = 50,
  dmax = 250,
  nsim = 999,
  p_cutoff = 0.05,
  min_ligand = 10,
  min_receptor = 10,
  min_cells = 1,
  n_cells_col = NA,
  annotation_col = "annotation",
  default_mode = "diffuse",
  diffuse_category = c("Secreted Signaling", "Non-protein Signaling"),
  species = c("human", "mouse"),
  genes = NULL,
  counts_by_group = NULL,
  fast = TRUE,
  cpu_threads = 4L,
  verbose = FALSE,
  ...
)

## S3 method for class 'SpatialExperiment'
blisa(
  x,
  bin_size = 50,
  LR_df = NULL,
```

```
  group = "cell_type",
  genes = NULL,
  min_cells = 1,
  min_total_counts = 10,
  platform = c("auto", "xenium", "visium"),
  verbose = FALSE,
  ...
)
```

Arguments

x	A gene-by-bin count matrix (for `blisa.default`) or a cell-level `SpatialExperiment` object (for `blisa.SpatialExperiment`).
...	Additional arguments passed to the relevant method.
bins	An `sf` object of bin polygons. Row order must match the column order of `x`.
LR_df	Data frame of ligand-receptor pairs with columns `ligand.symbol` and `receptor.symbol`. When `NULL`, CellChatDB for the chosen `species` is downloaded automatically.
bin_size	Numeric. Width of each hexagonal bin in coordinate units (e.g. microns). Passed to `hexBinCells` and `computeSpatialWeights`. Default `50`; for the Visium path it defaults to `100` (spot center-to-center spacing) when left unset.
dmax	Numeric. Maximum distance for diffuse-mode neighbours. Default `250`.
nsim	Integer. Number of permutations for Moran's I significance. Default `999`.
p_cutoff	Numeric. P-value threshold for High-High hotspots. Default `0.05`.
min_ligand	Numeric. Minimum ligand count threshold. Default `10`.
min_receptor	Numeric. Minimum receptor count threshold. Default `10`.
min_cells	Integer. Minimum number of cells per bin. For a `SpatialExperiment`, bins below this threshold are dropped during binning by `hexBinCells`. For pre-binned input supplied directly to `blisa.default`, they are dropped from `bins`, `x` and `counts_by_group` before the spatial weights are built; this requires `n_cells_col` and is ignored when `n_cells_col = NA`. Default `1`.
n_cells_col	Character or `NA`. Column in `bins` holding per-bin cell counts used for `min_cells` filtering. Set to `NA` to skip (default).
annotation_col	Character. Column in `LR_df` specifying interaction category used for communication-mode assignment. Default `"annotation"`.
default_mode	Character. CCC mode assigned to LR pairs whose annotation does not match `diffuse_category`. Default `"diffuse"`.
diffuse_category	Character vector of annotation categories treated as diffuse signalling.
species	Character. Which CellChatDB to download when `LR_df = NULL`. One of `"human"` (default) or `"mouse"`.
genes	Character vector of gene names to consider when matching ligand-receptor pairs. Defaults to `rownames(x)` (all genes in the `SpatialExperiment` object).

counts_by_group
: Named list of gene-by-bin count matrices, one per group level (e.g. cell type), as returned by hexBinCells when group is supplied. When provided, runCCI is called automatically after the BLISA loop and its output is included in the result as CCI_scores. Default NULL.

fast
: Logical. When TRUE (default), uses fastLISA::local_moran_bv (a fast C/OpenMP backend) for the bivariate local Moran's I computation. When FALSE, uses the original spdep::localmoran_bv + spdep::hotspot pipeline.

cpu_threads
: Integer. Number of OpenMP threads used by fastLISA::local_moran_bv. Only used when fast = TRUE. Ignored on platforms without OpenMP. Default 4L.

verbose
: Logical. If TRUE, print progress messages. Default FALSE (silent).

group
: Character. Column name in colData(x) to use as the grouping variable (e.g. cell type) for per-group bin aggregation and downstream CCI analysis via runCCI. If the column is not found in colData(x), a message is issued and CCI is skipped. Default "cell_type".

min_total_counts
: Numeric. Bins whose total counts (summed over all genes) fall below this threshold are dropped during binning by hexBinCells. Set to 0 to disable. Default 10.

platform
: Character. "auto" (default) inspects colData(x): if array_col/array_row are present the object is treated as "visium", otherwise "xenium". Set explicitly to override the detection.

Value

A list; see individual method documentation for details.

An object of class blisa with four components:

LR_results Data frame of BLISA results for each LR pair, including ccc_mode, sig_numbers, sig_index, sig_pval, all_pval, all_lisa, all_quadrant, and original columns from LR_df. all_quadrant is a character vector of hotspot quadrant labels ("High-High", "Low-Low", etc.) for every bin; non-tested bins are NA.

bins Bin-level sf object of hexagonal polygons.

spatial_weights Spatial weights list from computeSpatialWeights.

CCI_scores NULL unless counts_by_group is supplied, in which case a wide data frame of interaction scores from runCCI: rows are "Sender->Receiver" group pairs, columns are LR pairs.

Methods (by class)

- blisa(default): Method for a gene-by-bin count matrix.
- blisa(SpatialExperiment): Method for a cell-level SpatialExperiment object. For cell-level platforms (e.g. Xenium) it bins cells into hexagonal tiles via hexBinCells; for Visium it treats each spot as a bin via visiumSpotBins. Either way it then delegates to blisa.default. The path is chosen by platform.

Examples

```
## Not run:
# Download and cache the example Xenium breast cancer dataset (~21 MB)
data_url  <- paste0(
  "https://github.com/ChenLaboratory/example_data/releases/",
  "download/v1.0.0/spe_xenium_bc_s1rep1.rds"
)
cache_dir <- tools::R_user_dir("blisa", "cache")
data_file <- file.path(cache_dir, "spe_xenium_bc_s1rep1.rds")
if (!file.exists(data_file)) {
  dir.create(cache_dir, recursive = TRUE, showWarnings = FALSE)
  download.file(data_url, data_file, mode = "wb")
}
spe <- readRDS(data_file)

# blisa.SpatialExperiment method: bins cells, runs LISA, scores CCI
result <- blisa(spe, bin_size = 50, group = "cell_type")
result

## End(Not run)
```

`blisaPathway` *Aggregate BLISA hotspots to the pathway level*

Description

Rolls up per-ligand-receptor (LR) pair BLISA hotspot results to the signalling *pathway* level, grouping LR pairs by a pathway annotation column (e.g. CellChatDB's `pathway_name`). For each pathway it returns the spots that are hotspots for the pathway, a per-spot pathway p-value, and the LR pair driving each spot.

Usage

```
blisaPathway(
  x,
  method = c("minP", "simes", "fisher"),
  pathway_col = "pathway_name",
  p_cutoff = 0.05,
  p_adjust = c("none", "BH")
)
```

Arguments

`x`	A `blisa` object (or its `LR_results` data frame).
`method`	Character. Per-spot combination: `"minP"` (default, the original behaviour), `"simes"`, or `"fisher"`.
`pathway_col`	Character. Column of `LR_results` holding the pathway label. Default `"pathway_name"` (CellChatDB). Rows with `NA` are dropped.

p_cutoff	Numeric. Significance threshold for "simes"/"fisher". Default 0.05. Ignored for "minP".
p_adjust	Character. Multiple-testing adjustment across spots for "simes"/"fisher": "none" (default) or "BH". Ignored for "minP".

Details

The per-spot pathway p-value is computed by one of three methods:

"minP" The union of each LR pair's significant hotspot spots, with the minimum p-value per spot. p_cutoff and p_adjust are ignored.

"simes" Per-spot Simes combination across *all* LR pairs in the pathway, using the full per-spot p-values (all_pval) restricted to the High-High co-expression direction. Non-High-High/untested contributions are set to 1.

"fisher" As "simes" but Fisher's method.

For "simes"/"fisher", a spot is a pathway hotspot when its combined p-value is $\leq$ p_cutoff (after optional p_adjust).

Value

A blisa object whose LR_results slot is the pathway-level table (one row per pathway, row names = pathway, ordered by sig_numbers descending), so the standard blisa methods work directly (plotHotspots, plotLRrank, print). The bins and spatial_weights slots are carried over from x (when it is a blisa object); CCI_scores is NULL. The LR_results columns are:

sig_numbers Number of pathway hotspot spots.

n_LR_pairs Number of LR pairs annotated to the pathway.

sig_index List column: the hotspot spot indices.

sig_pval List column: the per-spot pathway p-values.

LR_pairs List column: all LR pair IDs in the pathway.

top_LR_pair List column: per spot, the LR pair(s) with the smallest contributing p-value.

See Also

blisa

Examples

```
## Not run:
res <- blisa(spe, platform = "visium")
blisaPathway(res)                          # min-P (original behaviour)
blisaPathway(res, method = "simes", p_adjust = "BH")

## End(Not run)
```

`computeSpatialWeights` *Compute Spatial Weights for BLISA*

Description

Builds queen (nearby) and distance-decay (diffuse) spatial weight matrices from a bin-level `sf` object, excluding bins that have no neighbours.

Usage

```
computeSpatialWeights(bins, bin_size = 50, dmax = 250)
```

Arguments

`bins`	An `sf` object of spatial bins.
`bin_size`	Numeric. Bin spacing used to define queen adjacency (`1.2 * bin_size` radius).
`dmax`	Numeric. Maximum distance for diffuse-mode neighbours.

Value

A list with:

queen_wt Spatial weights list for nearby (queen) mode.

dist_wt Spatial weights list for diffuse (distance-decay) mode.

keep_idx_queen Integer indices of bins used in queen-mode Moran.

keep_idx_dist Integer indices of bins used in diffuse-mode Moran.

isolate_idx_queen Integer indices of original queen-mode isolates.

isolate_idx_dist Integer indices of original diffuse-mode isolates.

queen_nb_full Full (unsubset) neighbour list for nearby mode, indexed over all bins.

dist_nb_full Full (unsubset) neighbour list for diffuse mode, indexed over all bins.

Examples

```
## Not run:
set.seed(42)
pts  <- sf::st_as_sf(
  data.frame(x = runif(300, 0, 1000), y = runif(300, 0, 1000)),
  coords = c("x", "y"), crs = NA
)
bins <- sf::st_sf(
  geometry = sf::st_make_grid(pts, cellsize = 100,
                              what = "polygons", square = FALSE)
)
sw <- computeSpatialWeights(bins, bin_size = 100, dmax = 300)
names(sw)

## End(Not run)
```

`filterLRpairs` *Filter ligand-receptor pairs by expression threshold*

Description

Retains only LR pairs where at least one bin/spot has counts at or above `min_ligand` for every ligand subunit and `min_receptor` for every receptor subunit.

Usage

```
filterLRpairs(
  counts,
  min_ligand = 10,
  min_receptor = 10,
  LR_df = NULL,
  species = c("human", "mouse")
)
```

Arguments

`counts`	Gene-by-bin count matrix (dense or sparse). Row names must be gene symbols.
`min_ligand`	Numeric. Minimum count threshold for ligand genes. At least one bin must meet or exceed this value. Default 10.
`min_receptor`	Numeric. Minimum count threshold for receptor genes. At least one bin must meet or exceed this value. Default 10.
`LR_df`	Data frame of ligand-receptor pairs with columns `ligand.symbol` and `receptor.symbol` (comma-separated gene symbols for multi-subunit complexes). When `NULL`, the CellChatDB for the chosen `species` is downloaded automatically.
`species`	Character. Which CellChatDB to download when `LR_df` is `NULL`. One of `"human"` (default) or `"mouse"`.

Value

A subset of `LR_df` containing only pairs that pass the expression thresholds for both ligand and receptor.

Examples

```
## Not run:
# Supply a small custom LR_df to avoid a network download
LR_df <- data.frame(
  ligand.symbol   = c("GENE1", "GENE3"),
  receptor.symbol = c("GENE2", "GENE4"),
  annotation      = c("Secreted Signaling", "ECM-Receptor"),
  row.names       = c("LR1", "LR2")
)
set.seed(1)
```

```
counts <- matrix(
  rpois(4 * 50, lambda = c(20, 1, 5, 20)), nrow = 4, ncol = 50,
  dimnames = list(c("GENE1", "GENE2", "GENE3", "GENE4"),
                  paste0("bin_", 1:50))
)
filterLRpairs(counts, min_ligand = 10, min_receptor = 10, LR_df = LR_df)

## End(Not run)
```

`hexBinCells` *Bin cells into hexagonal spatial bins*

Description

Aggregates single-cell spatial data into hexagonal bins and returns a bin-level count matrix together with a matching `sf` polygon object, ready to pass directly to `blisa.default`.

Usage

```
hexBinCells(
  coords_df,
  counts_matrix,
  bin_size = 50,
  min_cells = 1,
  min_total_counts = 10,
  group = NULL,
  verbose = FALSE
)
```

Arguments

`coords_df`	Data frame or matrix with columns `x_centroid` and `y_centroid` (e.g. the output of `SpatialExperiment::spatialCoords()`). Row names must be cell IDs matching the column names of `counts_matrix`.
`counts_matrix`	Gene-by-cell count matrix (dense or sparse). Row names must be gene symbols; column names must be cell IDs present in `coords_df`.
`bin_size`	Numeric. Approximate width of each hexagonal bin in coordinate units (e.g. microns). Analogous to `grid.length.x` in `sciderHex::gridDensity`. Default `50`.
`min_cells`	Integer. Bins containing fewer than `min_cells` cells are dropped from the output. Default `1`.
`min_total_counts`	Numeric. Bins whose total counts (summed over all genes) fall below this threshold are dropped from the output, alongside the `min_cells` filter. Set to `0` to disable. Default `10`.

group	Factor or character vector of length ncol(counts_matrix) giving the cell-type label of each cell. When supplied, a named list of per-cell-type gene-by-bin matrices is included in the output as counts_by_group. Default NULL (not computed).
verbose	Logical. If TRUE, print progress messages. Default FALSE (silent).

Value

A list with:

counts_matrix Gene-by-bin sparse count matrix (all cells combined). Column *i* corresponds to row *i* of bins.

bins An sf object of hexagonal bin polygons with an n_cells column recording how many cells fall in each bin and a total_counts column recording the summed counts per bin. Row order matches the columns of counts_matrix.

counts_by_group (Only when group is supplied.) A named list of gene-by-bin sparse matrices, one per cell-type level, with the same bin order as counts_matrix.

Examples

```
## Not run:
set.seed(42)
n <- 500
coords <- data.frame(
  x_centroid = runif(n, 0, 1000),
  y_centroid = runif(n, 0, 1000),
  row.names  = paste0("cell_", seq_len(n))
)
counts <- Matrix::Matrix(
  matrix(rpois(20L * n, lambda = 5), nrow = 20L, ncol = n,
         dimnames = list(paste0("gene_", 1:20), paste0("cell_", seq_len(n)))),
  sparse = TRUE
)
group  <- sample(c("TypeA", "TypeB"), n, replace = TRUE)
binned <- hexBinCells(coords, counts, bin_size = 100, group = group)
str(binned, max.level = 1)

## End(Not run)
```

---

is.blisa *Test if an object is a blisa object*

Description

Test if an object is a blisa object

Usage

```
is.blisa(x)
```

Arguments

x	Any R object.

Value

Logical.

Examples

```
is.blisa(list())          # FALSE
is.blisa("not a blisa")   # FALSE
```

---

plotCCI *Heatmap of CCI scores across all ligand-receptor pairs*

Description

Generic function. Draws a clustered heatmap (via `ComplexHeatmap`) with rows as Sender → Receiver cell-type pairs and columns as LR pairs. Row annotations colour-code the sender and receiver cell types.

Usage

```
plotCCI(x, ...)

## S3 method for class 'blisa'
plotCCI(
  x,
  top_lr = 20,
  top_pairs = 30,
  lr_pairs = NULL,
  sender = NULL,
  receiver = NULL,
  colors = NULL,
  colours = NULL,
  main = NULL,
  ...
)

## Default S3 method:
plotCCI(
  x,
  top_lr = 20,
  top_pairs = 30,
  lr_pairs = NULL,
  sender = NULL,
  receiver = NULL,
```

```
  colors = NULL,
  colours = NULL,
  main = NULL,
  ...
)
```

Arguments

x	A blisa object or a CCI scores data frame (the CCI_scores slot of a blisa object). The data frame must contain columns Sender, Receiver, and one column per LR pair.
...	Additional arguments passed to the relevant method.
top_lr	Integer or NULL. Number of top-ranked LR pairs (by sig_numbers) to display as columns. LR pairs in CCI_scores are already ordered by rank, so this simply takes the first top_lr columns. Ignored when lr_pairs is supplied. Default 20.
top_pairs	Integer or NULL. Number of top sender-receiver pairs to display as rows, ranked by their maximum interaction score across the displayed LR pairs (after top_lr / lr_pairs is applied). When NULL all rows are shown. Default 30.
lr_pairs	Character vector or NULL. When supplied, only these LR pair column names are shown, in the order given, overriding top_lr. Names not found in CCI_scores are dropped with a warning. Default NULL (use top_lr).
sender	Character vector or NULL. If provided, only rows where Sender is in this vector are kept. Applied independently of receiver (AND logic when both are supplied). Default NULL (all senders).
receiver	Character vector or NULL. If provided, only rows where Receiver is in this vector are kept. Applied independently of sender (AND logic when both are supplied). Default NULL (all receivers).
colors	Named character vector mapping cell-type names to colours, used for the sender/receiver row annotations. When NULL (default), colours are assigned automatically from the package palette. The British spelling colours is accepted as an alias.
colours	Alias for colors (British spelling). Ignored when colors is supplied.
main	Character or NULL. Title drawn above the heatmap (mapped to the column_title of ComplexHeatmap::Heatmap). Default NULL (no title).

Value

Invisibly returns the Heatmap object.

Methods (by class)

- plotCCI(blisa): Method for a blisa object. Extracts CCI_scores and delegates to plotCCI.default. Stops with an informative error if CCI_scores is NULL.
- plotCCI(default): Method for a CCI scores data frame (e.g. the CCI_scores slot of a blisa object).

See Also

plotCCILR for a sender-by-receiver heatmap of a single LR pair; plotCCIsummary for an aggregated sender-by-receiver heatmap across all LR pairs.

Examples

```
## Not run:
# Continuing from the blisa() example:
# result <- blisa(spe, bin_size = 50, group = "cell_type")
plotCCI(result, top_lr = 20, top_pairs = 30)

## End(Not run)
```

---

plotCCILR *Sender-by-receiver heatmap of CCI scores for one ligand-receptor pair*

Description

Generic function. Reshapes the CCI data frame into a receiver-by-sender cell-type matrix for one selected LR pair and draws a clustered heatmap.

Usage

```
plotCCILR(x, ...)

## S3 method for class 'blisa'
plotCCILR(x, index = 1, ligand = NULL, receptor = NULL, main = NULL, ...)

## Default S3 method:
plotCCILR(x, lr_pair, main = NULL, ...)
```

Arguments

x	A blisa object or a CCI scores data frame (the CCI_scores slot of a blisa object).
...	Additional arguments passed to the relevant method.
index	Integer. Row index into LR_results selecting the ligand-receptor pair to visualise. Ignored when both ligand and receptor are supplied. Default 1 (top-ranked pair).
ligand	Character. Ligand gene symbol. When both ligand and receptor are provided the matching LR pair is located automatically and index is ignored. Must be supplied together with receptor.
receptor	Character. Receptor gene symbol. Must be supplied together with ligand.

main	Character or NULL. Title drawn above the heatmap. When supplied, the heatmap is drawn with this overall title (via ComplexHeatmap::draw); the Heatmap object is returned invisibly. Default NULL (no title; object returned for the caller to print).
lr_pair	Character. Column name in the CCI scores data frame corresponding to the ligand-receptor pair to visualise (e.g. "CXCL12_CXCR4").

Value

A Heatmap object.

Methods (by class)

- plotCCILR(blisa): Method for a blisa object. The LR pair is selected by index (default 1, the top-ranked pair) unless both ligand and receptor are supplied, in which case the matching row is located automatically and index is ignored. Stops with an informative error if CCI_scores is NULL or the selected LR pair has no significant hotspots.
- plotCCILR(default): Method for a CCI scores data frame (e.g. the CCI_scores slot of a blisa object). The LR pair is selected by column name via lr_pair.

See Also

plotCCI for an overview heatmap across all LR pairs; plotCCIsummary for an aggregated sender-by-receiver heatmap.

Examples

```
## Not run:
# Continuing from the blisa() example:
# result <- blisa(spe, bin_size = 50, group = "cell_type")
plotCCILR(result, index = 1)

## End(Not run)
```

---

plotCCIspatial *Spatial map of dominant sender-receiver cell-type pairs at BLISA hotspots*

Description

For a selected ligand-receptor pair, identifies the dominant interacting cell-type pair at each significant hotspot bin and draws a spatial map of the tissue coloured by those pairs. Receiver cells are those inside hotspot bins; sender cells are drawn from the neighbouring bins defined by the LR pair's signalling mode.

Usage

```
plotCCIspatial(
  x,
  counts_by_group,
  index = 1,
  ligand = NULL,
  receptor = NULL,
  top_pairs = 30,
  weighted = TRUE
)
```

Arguments

x	A blisa object as returned by blisa.
counts_by_group	Named list of gene-by-bin count matrices, one per cell type. Typically the counts_by_group element returned by hexBinCells. Names must match the cell-type levels.
index	Integer. Row index into x$LR_results selecting the ligand-receptor pair to visualise. Ignored when both ligand and receptor are supplied. Default 1 (top-ranked pair).
ligand	Character. Ligand gene symbol. When both ligand and receptor are provided the matching LR pair is located automatically and index is ignored. Must be supplied together with receptor.
receptor	Character. Receptor gene symbol. Must be supplied together with ligand.
top_pairs	Integer. Maximum number of distinct cell-type pairs to show in the legend; remaining pairs are grouped as "rare pairs". Default 30.
weighted	Logical. If TRUE (default), the ligand contribution of each sender cell type at hotspot h is $\sum_{j \in N(h)} w_{hj} L_j$, using the same neighbours and row-standardised spatial weights as the bivariate Moran's I test (the hotspot bin itself is excluded). If FALSE, ligand counts are summed without weights over the hotspot bin and its neighbours.

Value

A ggplot object.

See Also

plotHotspots for a significance-based spatial map of hotspot bins.

Examples

```
## Not run:
# Continuing from the blisa() example:
# result <- blisa(spe, bin_size = 50, group = "cell_type")
binned <- hexBinCells(
```

```
  as.data.frame(SpatialExperiment::spatialCoords(spe)),
  SummarizedExperiment::assay(spe, "counts"),
  bin_size = 50, group = spe$cell_type
)
plotCCIspatial(result, binned$counts_by_group, index = 1)

## End(Not run)
```

`plotCCIsummary` *Sender-by-receiver heatmap of aggregated CCI scores across LR pairs*

Description

Generic function. Aggregates CCI scores across all (or the top-ranked) ligand-receptor pairs and draws a clustered receiver-by-sender heatmap, one cell per Sender → Receiver combination.

Usage

```
plotCCIsummary(x, ...)

## S3 method for class 'blisa'
plotCCIsummary(
  x,
  top_lr = NULL,
  sender = NULL,
  receiver = NULL,
  agg_fun = sum,
  main = NULL,
  ...
)

## Default S3 method:
plotCCIsummary(
  x,
  top_lr = NULL,
  sender = NULL,
  receiver = NULL,
  agg_fun = sum,
  main = NULL,
  ...
)
```

Arguments

`x`	A `blisa` object or a CCI scores data frame (the `CCI_scores` slot of a `blisa` object).
`...`	Additional arguments passed to the relevant method.

top_lr	Integer or NULL. Number of top-ranked LR pairs (by sig_numbers) to include before aggregating. LR pairs in CCI_scores are already ordered by rank, so this takes the first top_lr columns. NULL (default) uses all pairs.
sender	Character vector or NULL. If provided, only rows where Sender is in this vector are kept (AND logic with receiver). Default NULL (all senders).
receiver	Character vector or NULL. If provided, only rows where Receiver is in this vector are kept (AND logic with sender). Default NULL (all receivers).
agg_fun	Function used to aggregate scores across LR pairs for each Sender → Receiver combination. Receives a numeric vector with NAs already removed. Default sum.
main	Character or NULL. Title drawn above the heatmap. When supplied, the heatmap is drawn with this overall title (via ComplexHeatmap::draw); the Heatmap object is returned invisibly. Default NULL (no title; object returned for the caller to print).

Value

A Heatmap object.

Methods (by class)

- plotCCIsummary(blisa): Method for a blisa object. Stops with an informative error if CCI_scores is NULL.
- plotCCIsummary(default): Method for a CCI scores data frame (e.g. the CCI_scores slot of a blisa object).

See Also

plotCCILR for a per-LR-pair version of this plot; plotCCI for a heatmap with LR pairs as columns.

Examples

```
## Not run:
# Continuing from the blisa() example:
# result <- blisa(spe, bin_size = 50, group = "cell_type")
plotCCIsummary(result)
plotCCIsummary(result, top_lr = 10, agg_fun = mean)

## End(Not run)
```

---

plotHotspots *Spatial hotspot map for one ligand-receptor pair*

Description

Generic function. Plots each bin coloured by significance status: empty, non-significant, or significant hotspot (continuous gradient of -log10 p-value or 1 - p-value).

Usage

```
plotHotspots(x, ...)

## S3 method for class 'blisa'
plotHotspots(
  x,
  index = 1,
  ligand = NULL,
  receptor = NULL,
  as_points = FALSE,
  size = 1.5,
  background = NULL,
  log_pval = TRUE,
  p_cutoff = NULL,
  spots = NULL,
  spot_pval = NULL,
  title = NULL,
  ...
)
```

Arguments

x	A blisa object.
...	Additional arguments passed to the method.
index	Integer. Row index into LR_results selecting the ligand-receptor pair to visualise. Ignored when both ligand and receptor are supplied. Default 1 (top-ranked pair).
ligand	Character. Ligand gene symbol. When both ligand and receptor are provided the matching LR pair is located automatically and index is ignored. Must be supplied together with receptor.
receptor	Character. Receptor gene symbol. Must be supplied together with ligand.
as_points	Logical. If TRUE, draw each bin as a dot at its centroid instead of its polygon. Useful for Visium, where each bin is a single spot. Default FALSE (draw polygons).
size	Numeric. Point size when as_points = TRUE (or when background is supplied). Default 1.5.
background	A ggplot object to draw the hotspots on top of, e.g. scider::plotImage(spe) to place the H&E image behind the spots. When supplied, the hotspots are always rendered as dots (the raster background is incompatible with geom_sf coordinates), and the bins must share the background plot's coordinate frame (true for objects read by scider::readVisium()). Default NULL.
log_pval	Logical. If TRUE (default), colour significant bins by -log10(p-value). If FALSE, use 1 - p-value.
p_cutoff	Numeric or NULL. When NULL (default), the pre-computed hotspot bins stored in the blisa object are used, reflecting the p_cutoff and High-High quadrant

	classification applied during blisa. When a numeric value is supplied, bins are re-defined on the fly as those with all_pval <= p_cutoff and quadrant label "High-High" (from the stored all_quadrant), giving an exact re-threshold consistent with the original classification.
spots	Integer vector or NULL. Significant spot indices to plot directly, bypassing the LR-pair lookup – e.g. a sig_spot_index entry from blisaPathway. When supplied, index/ ligand/receptor/p_cutoff are ignored and every bin is drawn as grey background with these spots coloured. Default NULL.
spot_pval	Numeric vector. P-values for spots (same length and order); required when spots is supplied.
title	Character or NULL. Plot title. Defaults to the LR pair id (or "hotspots" in spots mode). Useful to label a pathway, e.g. title = "MDK pathway".

Value

A ggplot object.

Methods (by class)

- plotHotspots(blisa): Method for a blisa object.

Examples

```
## Not run:
# Continuing from the blisa() example:
# result <- blisa(spe, bin_size = 50, group = "cell_type")
plotHotspots(result, index = 1)
plotHotspots(result, index = 1, log_pval = FALSE)

## End(Not run)
```

---

plotLRrank *Dot plot ranking LR pairs by number of significant hotspot bins*

Description

Generic function for ranking LR pairs. Dispatches on the class of x:

- plotLRrank.blisa accepts a blisa object and uses its LR_results slot directly.
- plotLRrank.data.frame accepts the LR_results data frame directly.

Usage

```
plotLRrank(x, ...)

## S3 method for class 'blisa'
plotLRrank(x, top = 30, pt_size = 4, flip = FALSE, size_by = "auto", ...)

## S3 method for class 'data.frame'
plotLRrank(
  x,
  top = 30,
  pt_size = 4,
  flip = FALSE,
  item_label = "Ligand-Receptor Pair",
  size_by = NULL,
  ...
)
```

Arguments

x	A blisa object or a data frame of LR results. The data frame must contain columns sig_numbers and annotation.
...	Additional arguments passed to the relevant method.
top	Integer or NULL. Number of top LR pairs (by sig_numbers) to display. Default 30.
pt_size	Numeric. Point size passed to geom_point. Default 4.
flip	Logical. When TRUE, LR pairs are placed on the x-axis and the hotspot count on the y-axis (vertical orientation). Default FALSE (LR pairs on y-axis, horizontal orientation).
size_by	Character or NULL. Name of a numeric column to map to point size. When NULL, all points use the fixed pt_size. For plotLRrank.blisa the default is "auto", which sizes pathway-level results (from blisaPathway) by "n_LR_pairs" and leaves LR-level results unsized; pass a column name, or NULL to disable, to override. Relabel the legend with + ggplot2::labs(size = ...).
item_label	Character. Axis label for the ranked items. Defaults to "Ligand-Receptor Pair"; plotLRrank.blisa sets it to "Pathway" for pathway-level objects.

Value

A ggplot object.

Methods (by class)

- plotLRrank(blisa): Method for a blisa object. Extracts LR_results and delegates to plotLRrank.data.frame.
- plotLRrank(data.frame): Method for a data frame of LR results (e.g. the LR_results slot of a blisa object).

Examples

```
## Not run:
# Continuing from the blisa() example:
# result <- blisa(spe, bin_size = 50, group = "cell_type")
plotLRrank(result, top = 30)
plotLRrank(result, top = 20, flip = TRUE)

## End(Not run)
```

---

runCCI *Score cell-cell interactions from BLISA hotspots*

Description

Generic function for scoring cell-cell interactions. Dispatches on the class of x:

- runCCI.blisa accepts a blisa object. If CCI_scores are already present and overwrite = FALSE (the default), the object is returned unchanged. Set overwrite = TRUE with a counts_by_group to recompute and replace existing scores. If no scores exist, counts_by_group must be supplied and scores are computed and attached.
- runCCI.default performs the raw computation given a blisa object and a counts_by_group list, returning only the scores data frame. Used internally by runCCI.blisa and blisa.default.

Usage

```
runCCI(x, ...)

## S3 method for class 'blisa'
runCCI(x, counts_by_group = NULL, overwrite = FALSE, weighted = TRUE, ...)

## Default S3 method:
runCCI(x, counts_by_group, weighted = TRUE, ...)
```

Arguments

x	A blisa object.
...	Additional arguments passed to the relevant method.
counts_by_group	Named list of gene-by-bin sparse count matrices, one per group level (e.g. cell type). Typically the counts_by_group element of the list returned by hexBinCells when group is supplied. Names must match the group levels. Required when x$CCI_scores is NULL or when overwrite = TRUE.
overwrite	Logical. If FALSE (default) and x$CCI_scores is already populated, the object is returned unchanged. If TRUE and counts_by_group is supplied, existing scores are recomputed and replaced.

`weighted` Logical. If `TRUE` (default), the ligand count of each sender group is the sum over hotspots h of its spatially weighted ligand counts around h, $\sum_h \sum_j w_{hj} L_j$, using the same neighbours and row-standardised spatial weights as the bivariate Moran's I test. This is the sum of the local ligand terms used by `plotCCIspatial`. If `FALSE`, ligand counts are summed without weights over the hotspot bins and their neighbours, each bin counted once.

Value

See individual method documentation.

`runCCI.blisa`: the input `blisa` object with `CCI_scores` populated (a wide data frame – rows are `"Sender->Receiver"` group pairs, columns are LR pairs).

`runCCI.default`: a data frame with `"Sender->Receiver"` row names and one column per significant LR pair containing the interaction score `0.5 * log2(receiver * sender + 1)`.

Methods (by class)

- `runCCI(blisa)`: Method for a `blisa` object. If `CCI_scores` are already present and `overwrite = FALSE` (the default), the object is returned unchanged. Set `overwrite = TRUE` with a `counts_by_group` to recompute and replace existing scores. If no scores exist, `counts_by_group` must be supplied and scores are computed and attached to `x$CCI_scores`.
- `runCCI(default)`: Default method. Performs the raw CCI computation and returns only the scores data frame. Typically called internally; use `runCCI.blisa` to compute and attach scores to a `blisa` object in one step.

Examples

```
## Not run:
# Continuing from the blisa() example:
# result <- blisa(spe, bin_size = 50, group = "cell_type")

# CCI is computed automatically when group is supplied to blisa();
# access the scores directly:
head(result$CCI_scores)

# Or compute / recompute scores explicitly:
binned <- hexBinCells(
  as.data.frame(SpatialExperiment::spatialCoords(spe)),
  SummarizedExperiment::assay(spe, "counts"),
  bin_size = 50, group = spe$cell_type
)
result2 <- runCCI(result, counts_by_group = binned$counts_by_group,
                  overwrite = TRUE)

## End(Not run)
```

visiumSpotBins *Build spot-level hexagonal bins from a Visium SpatialExperiment*

Description

Turns a 10x Visium SpatialExperiment (as produced by scider::readVisium()) into the pre-binned inputs expected by blisa.default, treating each Visium spot as one bin. The spot coordinates are rebuilt from the integer array_col/array_row indices onto an exact, de-tilted hexagonal lattice with 100 um (spot_pitch) spacing – the standalone equivalent of scider::realignVisium().

Usage

```
visiumSpotBins(
  spe,
  spot_pitch = NULL,
  in_tissue_only = TRUE,
  hexagons = TRUE,
  min_total_counts = 10,
  verbose = FALSE
)
```

Arguments

spe	A Visium SpatialExperiment with a "counts" assay whose row names are gene symbols. array_col/array_row in colData are used to rebuild the exact lattice when present; otherwise spatialCoords(spe) is used as-is.
spot_pitch	Numeric or NULL. Spot center-to-center spacing used as bin_size. NULL (default) means: use 100 um for the array-index lattice, or measure it from the coordinates in the fallback. Supply a number to force it (interpreted in coordinate units).
in_tissue_only	Logical. Keep only spots with in_tissue == 1. Default TRUE.
hexagons	Logical. If TRUE (default), build hexagonal bin polygons; if FALSE, use point geometry (faster, but plotHotspots then draws points).
min_total_counts	Numeric. Drop spots whose total counts fall below this threshold. Default 10. Set to 0 to keep all spots.
verbose	Logical. Print progress messages. Default FALSE.

Details

On Visium, each spot is already a spatial bin, so no cell-to-bin aggregation is needed (unlike hexBinCells). Because array_col increments by 2 within a row and alternate rows shift array_col by 1, the half-row stagger is already encoded in the indices; the lattice therefore needs no odd/even phase correction. Only the bin centroids are used by computeSpatialWeights; the hexagon polygons are for plotting (they do tessellate the lattice exactly).

When array_col/array_row are absent (e.g. a SpatialExperiment not built by scider::readVisium()), the function falls back to using spatialCoords(spe) directly: spots are still one bin each, but the de-tilt/exact-lattice step is skipped. In that case, if spot_pitch is NULL, the pitch is measured from the coordinates as the median nearest-neighbour distance. Since the physical Visium pitch is 100 um, 100 / pitch is microns-per-unit; the coordinates are *rescaled to microns* by that factor (near-identity when they are already microns), so the returned pitch is 100 and downstream bin_size/dmax are in microns. If you instead supply spot_pitch, no rescaling is done and everything stays in the coordinates' native units.

Value

A list with:

counts_matrix Gene-by-spot count matrix (columns match bins).

bins An sf object of per-spot bins with bin_id, n_cells (always 1), total_counts, and img_x/img_y (the raw image-registered spot coordinates, for overlaying plots on the H&E image) columns. Row order matches the columns of counts_matrix.

pitch The spot spacing used (= the recommended bin_size).

coord_unit "micron", "pixel/other", or NA (when spot_pitch was supplied) – the inferred coordinate unit.

See Also

blisa, hexBinCells

Index

blisa, 3, *7*, *16*, *20*, *25*
blisa-package, 2
blisa.default, *10*, *22*, *24*
blisaPathway, 6, *20, 21*

computeSpatialWeights, *4, 5*, 8, *24*

filterLRpairs, 9

hexBinCells, *3–5*, 10, *16*, *22*, *24, 25*

is.blisa, 11

plotCCI, 12, *15*, *18*
plotCCILR, *14*, 14, *18*
plotCCIspatial, 15, *23*
plotCCIsummary, *14, 15*, 17
plotHotspots, *7*, *16*, 18
plotLRrank, *7*, 20

runCCI, *5*, 22

visiumSpotBins, *5*, 24