Package ‘panoramic’

June 5, 2026

Title Meta-Analysis of Spatial Colocalization in Spatial Omics

Version 0.99.3

Description Provides a pipeline for quantifying and meta-analyzing spatial colocalization between cell types in spatial omics experiments. The package prepares SpatialExperiment inputs, computes Loh-bootstrap spatial summary functions (e.g. L- and K-functions) for cell-type pairs across samples, and performs random-effects meta-analysis to assess group-level differences in spatial colocalization.

URL `https://github.com/plevritis-lab/panoramic`

BugReports `https://github.com/plevritis-lab/panoramic/issues`

License MIT + file LICENSE

Depends R (>= 4.6)

Imports dplyr, ggplot2, ggrepel, igraph, tidygraph, ggraph, rlang, S4Vectors, SummarizedExperiment, spatstat.geom, spatstat.explore, BiocParallel, SpatialExperiment, concaveman, metafor, withr, matrixStats, spatstat.utils

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

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

biocViews Software, Spatial, SingleCell

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

git_branch devel

git_last_commit ca3f3ca

git_last_commit_date 2026-04-21

Repository Bioconductor 3.24

Date/Publication 2026-06-04

Author Jacob Chang [aut, cre, fnd] (ORCID: <`https://orcid.org/0000-0002-3719-7949`>)

Maintainer Jacob Chang <`jachang4@stanford.edu`>

Contents

`.local_comp_score` *Internal: local composition enrichment score*

Description

Internal: local composition enrichment score

Usage

```
.local_comp_score(frac, n_neighbors, overlap_area, global_density)
```

Value

Numeric vector of local enrichment scores (percentage-point difference between observed and expected local composition).

`.lohboot_block_weighted`

Internal: custom Loh bootstrap with overlap-weighted blocks

Description

Internal: custom Loh bootstrap with overlap-weighted blocks

Usage

```
.lohboot_block_weighted(
  X,
 fun = c("pcf", "Kest", "Lest", "pcfinhom", "Kinhom", "Linhom", "Kcross", "Lcross",
    "Kdot", "Ldot", "Kcross.inhom", "Lcross.inhom"),
  ...,
  global = FALSE,
  basicboot = FALSE,
  Vcorrection = FALSE,
  confidence = 0.95,
  nx = 4,
  ny = nx,
  nsim = 200,
  type = 7
)
```

Arguments

`X`	A marked ppp object.
`fun`	Character/function passed to spatstat local summary mapping.
`...`	Additional arguments forwarded to the selected local function.
`global`	Logical; if `TRUE`, compute global envelopes.
`basicboot`	Logical; if `TRUE`, use basic bootstrap intervals.
`Vcorrection`	Logical; apply variance correction for K-type functions.
`confidence`	Numeric confidence level in `(0.5, 1)`.
`nx, ny`	Integer numbers of blocks in x/y for block bootstrap.
`nsim`	Integer bootstrap replicates.
`type`	Quantile type.

Value

A data.frame with columns including `r`, theoretical curve, estimate, and lower/upper interval bounds.

`.lohboot_local_comp` *Internal: Loh-style bootstrap for local composition enrichment*

Description

Internal: Loh-style bootstrap for local composition enrichment

Usage

```
.lohboot_local_comp(
  X,
  from,
  to,
  radii_um,
  nsim = 200,
  confidence = 0.95,
  type = 7,
  boot = c("approx", "block"),
  nx = 4,
  ny = nx,
  local_cache = NULL
)
```

Arguments

`X`	A marked ppp object.
`from, to`	Character marks defining anchor and target cell types.
`radii_um`	Numeric radii at which enrichment is evaluated.
`nsim`	Integer bootstrap replicates.
`confidence`	Numeric confidence level.
`type`	Quantile type for bootstrap intervals.
`boot`	One of `"approx"` or `"block"`.
`nx, ny`	Integer block counts (used only for `boot = "block"`).
`local_cache`	Optional cache from `.precompute_local_comp_cache()`.

Value

A data.frame compatible with `.summarize_lohboot()`.

`.lohboot_quiet` *Internal: quiet Loh bootstrap wrapper*

Description

Run `spatstat.explore::lohboot()` while suppressing console output, returning the bootstrap object invisibly for further summarization.

Usage

```
.lohboot_quiet(
  X,
  fun,
  ...,
  verbose = FALSE,
  boot = c("approx", "block"),
  tile_size = NULL,
  nx = NULL,
  ny = NULL
)
```

Arguments

`...`	Arguments passed to `spatstat.explore::lohboot()`.
`boot`	Character. Bootstrap mode: `"approx"` (no tiling) or `"block"` (tiled Loh bootstrap with overlap-weighted blocks).
`tile_size`	Optional numeric scalar giving tile size in the same units as spatial coordinates. Used only when `boot = "block"`; translated into `nx` and `ny` for `lohboot()` based on the sample's window bounding box. Cannot be used with `nx`/`ny`.
`nx, ny`	Optional integers giving the number of tiles in x/y directions for `lohboot()` when `boot = "block"`. Ignored for `boot = "approx"`.

Value

A Loh bootstrap object as returned by `spatstat.explore::lohboot()` or a compatible data.frame for `boot = "block"`.

`.one_pair_one_sample` *Internal: compute spatial stats for one pair in one sample*

Description

Compute a spatial summary curve and bootstrap variance for a single cell-type pair within a single prepared sample, evaluated on a common radius grid.

Usage

```
.one_pair_one_sample(
  meta,
  ct1,
  ct2,
  stat = "local_comp_enrichment",
  nsim = 100,
  correction = "translate",
  radii_um,
  verbose = FALSE,
  boot = c("approx", "block"),
  tile_size = NULL,
  nx = NULL,
  ny = NULL,
  local_comp_cache = NULL
)
```

Arguments

`meta`	The `metadata$panoramic` list for one sample, containing at least a spatstat ppp object and a `marks_tab` table.
`ct1`, `ct2`	Character. Cell-type labels.
`stat`	Character. Summary statistic. Default is `"local_comp_enrichment"`. Other options are `"Lcross"`, `"Kcross"`, `"Lest"`, and `"Kest"`.
`nsim`	Integer. Number of Loh bootstrap simulations.
`correction`	Character. Edge correction passed to spatstat.
`radii_um`	Numeric vector of radii (microns) on which to summarize.
`boot`	Character. Bootstrap mode: `"approx"` (no tiling) or `"block"` (tiled Loh bootstrap).
`tile_size`	Optional numeric scalar giving tile size in the same units as spatial coordinates. Used only when `boot = "block"`.
`nx`, `ny`	Optional integers giving the number of tiles in x/y directions when `boot = "block"`.
`local_comp_cache`	Optional cache object from `.precompute_local_comp_cache()` to avoid repeated neighbor searches when `stat = "local_comp_enrichment"`.

Value

A data.frame with columns `radius_um`, `yi`, and `vi`, filled with `NA` if insufficient cells of either type are present.

`.panoramic_add_coloc_direction`

Internal: add directional colocalization labels

Description

Internal: add directional colocalization labels

Usage

```
.panoramic_add_coloc_direction(df)
```

`.panoramic_attach_sample_metadata` *Internal: attach sample-level metadata to PANORAMIC colData*

Description

Internal: attach sample-level metadata to PANORAMIC colData

Usage

```
.panoramic_attach_sample_metadata(se_stats, spe_list, fields)
```

Arguments

`se_stats`	SummarizedExperiment from panoramic_spatialstats().
`spe_list`	Named list of SpatialExperiment objects used to produce se_stats.
`fields`	Character vector of colData fields to attach.

Value

`se_stats` with additional columns in `colData(se_stats)`.

`.panoramic_extract_meta_table` *Internal: extract feature-level meta-analysis table from rowData*

Description

Internal: extract feature-level meta-analysis table from rowData

Usage

```
.panoramic_extract_meta_table(
  se,
  feature_cols = c("ct1", "ct2", "radius_um", "stat"),
  drop_na = FALSE
)
```

`.panoramic_extract_spatialstats_table` *Internal: flatten spatialstats assays to a long table*

Description

Internal: flatten spatialstats assays to a long table

Usage

```
.panoramic_extract_spatialstats_table(se, drop_na = FALSE)
```

`.panoramic_one_value_per_sample` *Internal: extract one metadata value per sample from a SpatialExperiment list*

Description

Internal: extract one metadata value per sample from a SpatialExperiment list

Usage

```
.panoramic_one_value_per_sample(spe_list, field)
```

Arguments

`spe_list`	Named list of SpatialExperiment objects.
`field`	Character scalar naming a colData column in each sample.

Value

Character vector aligned to names(spe_list), one value per sample.

`.panoramic_pairs` *Internal: generate cell-type pairs*

Description

Construct the set of cell-type pairs to analyze, optionally including self-pairs and filtering to cell types present in at least a given fraction of samples.

Usage

```
.panoramic_pairs(prep_list, include_self = TRUE, min_presence = 0)
```

Arguments

`prep_list`	List of prepared SpatialExperiment objects, each with `metadata(spe)$panoramic$ppp` containing a marked spatstat point pattern.
`include_self`	Logical. If `TRUE`, include (ct, ct) self-pairs. If `FALSE`, only cross-type pairs are returned.
`min_presence`	Numeric in $[0, 1]$. Minimum fraction of samples in which a cell type must be observed (at least one cell) to be included. `0` keeps all observed types, `0.5` requires observed presence in at least half of samples.

Value

A data.frame with columns `ct1` and `ct2`.

`.precompute_local_comp_cache` *Internal: precompute local composition enrichment cache*

Description

Build per-radius local composition enrichment matrices for a point pattern. For each point and radius, store local neighbor fractions/counts and the overlap area between the radius-disc and tissue window so enrichment can be computed as an edge-corrected proportion difference.

Usage

```
.precompute_local_comp_cache(X, radii_um)
```

Arguments

`X`	A marked ppp object.
`radii_um`	Numeric radii at which to precompute local composition.

Value

A list containing precomputed local composition matrices and indexing structures used by local composition bootstrap routines.

`.reorder_ccw` *Internal: reorder hull vertices counterclockwise Orders a set of 2D coordinates counterclockwise around their centroid. Mainly used for convex spatial window construction to ensure correct polygon orientation for spatstat owin objects.*

Description

Internal: reorder hull vertices counterclockwise Orders a set of 2D coordinates counterclockwise around their centroid. Mainly used for convex spatial window construction to ensure correct polygon orientation for spatstat owin objects.

Usage

```
.reorder_ccw(coords)
```

Arguments

`coords`	Numeric matrix of coordinates (columns: x, y).

Value

Reordered numeric matrix of coordinates

`.safe_approx` *Internal: safe interpolation on a common radius grid*

Description

Interpolate values `y` at target `xout` using `stats::aprox()`, handling missing/non-finite values and low sample sizes.

Usage

```
.safe_approx(x, y, xout)
```

Arguments

`x`	Numeric vector of original inputs x.
`y`	Numeric vector of original y-values.
`xout`	Numeric vector of target x-values.

Value

Numeric vector of interpolated values, length `length(xout)`.

`.spatstat_local_function_info` *Internal: local function info (spatstat helper)*

Description

Internal: local function info (spatstat helper)

Usage

```
.spatstat_local_function_info(key)
```

Arguments

`key`	Character scalar naming a supported spatstat summary function.

Value

A named list describing the global/local function mapping and options for the requested key, or `NULL` if unsupported.

`.summarize_lohboot` *Internal: summarize Loh bootstrap to (r, yi, vi)*

Description

Convert a spatstat Loh bootstrap object to aligned vectors of radii, centered estiamtes, and variance. Variance is taken directly if present, reconstructed from confidence intervale if needed, or computed from simulation replicates as a fallback.

Usage

```
.summarize_lohboot(loh_obj, center_L = TRUE, conf = 0.95)
```

Arguments

`loh_obj`	Loh bootstrap object returned by spatstat's L/K-summary functions with `global=TRUE`.
`center_L`	Logical. If `TRUE`, center the estimate by the theoretical curve (e.g. L(r)-r) when available.
`conf`	Numeric confidence level used when reconstructing variance from confidence interval half-width.

Value

A data.frame with numeric columns `r`, `yi`, and `vi`, or `NULL` if the object cannot be parsed.

`create_spatial_network` *Construct spatial colocalization network from PANORAMIC results*

Description

Build a cell-type network where edges represent differential spatial colocalization between cell-type pairs, based on the output of `panoramic_meta_mv(...)` (or a compatible function). Edges are filtered by FDR and optional `z_diff` sign, and weighted by `|z_diff|`.

Usage

```
create_spatial_network(
  se_diff,
  fdr_threshold = 0.05,
  leiden_resolution = 1,
  z_sign = c("both", "positive", "negative"),
  include_nonsig = FALSE,
  nonsig_max_fdr = 1,
  directed = FALSE,
  sig_operator = c("lt", "gt")
)
```

Arguments

`se_diff`	A SummarizedExperiment returned by `panoramic_meta_mv(...)`, where rowData(se_diff) contains at least ct1, ct2, z_diff, p_diff, and fdr_diff.
`fdr_threshold`	Numeric scalar. FDR threshold used to define significance for `edge_class` and, when `include_nonsig = FALSE`, to filter edges (default 0.05).
`leiden_resolution`	Numeric. Clustering resolution for the Leiden algorithm.
`z_sign`	Character sign filter for `z_diff`, one of `"both"` (default), `"positive"`, or `"negative"`. This enables group-specific networks by retaining only positive or only negative differential effects.
`include_nonsig`	Logical. If TRUE, include non-significant edges (up to `nonsig_max_fdr`) and mark them as dotted in plotting. If FALSE (default), keep only `fdr_diff < fdr_threshold`.
`nonsig_max_fdr`	Numeric upper bound for retained edges when `include_nonsig = TRUE` (default 1.0).
`directed`	Logical. If `TRUE`, construct a directed network (and directed centrality metrics). If `FALSE`, construct an undirected network.
`sig_operator`	One of `"lt"` (default) or `"gt"` controlling significance threshold direction for `fdr_diff` when `include_nonsig = FALSE`. `"lt"` uses standard `fdr_diff < fdr_threshold`; `"gt"` inverts the threshold (diagnostic use only).

Value

A list with components:

- graph: an igraph object (directed or undirected) with edge attributes `weight`, `fdr`, `pval`, `edge_sig`, `edge_class` and vertex attributes `cluster`, `cluster_id`, `degree`, `betweenness`, `strength`.
- clusters: the community structure object from `igraph::cluster_leiden()`.
- n_clusters: number of detected clusters.
- modularity: modularity score of the clustering.
- z_sign: the applied sign filter.
- fdr_threshold: the applied significance threshold.
- include_nonsig: whether non-significant edges were included.
- nonsig_max_fdr: maximum retained FDR when non-significant edges are included.

Examples

```
se_diff <- SummarizedExperiment::SummarizedExperiment(
  assays = list(dummy = matrix(0, nrow = 3, ncol = 1)),
  rowData = S4Vectors::DataFrame(
    ct1 = c("A", "A", "B"),
    ct2 = c("B", "C", "C"),
    z_diff = c(2.0, -1.5, 1.2),
    p_diff = c(0.01, 0.03, 0.20),
    fdr_diff = c(0.02, 0.05, 0.30)
  )
)
```

```
net <- create_spatial_network(
  se_diff,
  fdr_threshold = 0.2,
  z_sign = "both",
  include_nonsig = TRUE
)

net$graph
```

`null_coalesce` *Pipe-compatible null operator*

Description

Returns b if a is NULL, otherwise a.

Usage

```
a %||% b
```

Arguments

`a`	b Objects to test; if a is NULL b is returned.

Value

The non-NULL object.

`panoramic` *Run PANORAMIC End-To-End*

Description

`panoramic_analyze()` runs preparation, spatial statistics, and multilevel pooling in one call. `panoramic()` is a convenience alias with identical arguments and return structure.

Usage

```
panoramic(
  spe_list,
  design,
  cell_type = "cell_type",
  pairs = "auto",
  radii_um,
  stat = "local_comp_enrichment",
  nsim = 100,
  correction = NULL,
  min_cells = 5L,
  concavity = 50,
  window = c("concave", "convex", "rect"),
```

```
  group_col = "group",
  group_tau2 = c("none", "separate"),
  patient_col = NULL,
  sample_col = NULL,
  tau_structure = c("patient", "patient_sample"),
  method_mv = "REML",
  vi_floor = "group_median",
  seed = 123,
  boot = c("approx", "block"),
  tile_size = NULL,
  nx = NULL,
  ny = NULL,
  BPPARAM = BiocParallel::SerialParam(),
  verbose = FALSE
)
```

Arguments

`spe_list`	Named list of SpatialExperiment objects (one per sample).
`design`	data.frame with at least columns `sample` and `group`.
`cell_type`	Character colData column containing cell-type labels.
`pairs`	Either `"auto"` or a data.frame with columns `ct1`, `ct2`.
`radii_um`	Numeric vector of radii (microns).
`stat`	Spatial statistic passed to `panoramic_spatialstats()`.
`nsim`	Integer bootstrap replicates per sample/pair.
`correction`	Optional edge-correction method for spatstat-based statistics. Ignored when `stat = "local_comp_enrichment"`.
`min_cells`	Minimum cells per type per sample during preparation.
`concavity`	Concavity for concave-hull sample windows (larger values approach convex hulls).
`window`	One of `"concave"`, `"convex"`, or `"rect"`.
`group_col`	Group column in `colData` used for multilevel pooling and contrasts.
`group_tau2`	If `"separate"`, fit additional per-group multilevel models to report group-specific heterogeneity components.
`patient_col`	Optional patient-id column for multilevel pooling. If `NULL`, PANORAMIC uses `"patient"` from `design` when available, otherwise falls back to `sample_col`.
`sample_col`	Optional sample-id column used in `panoramic_meta_mv()`. If `NULL`, PANORAMIC uses `"sample"` (aligned to `design$sample`).
`tau_structure`	Random-effects structure in `panoramic_meta_mv()`: `"patient"` or `"patient_sample"`.
`method_mv`	Estimator passed to `metafor::rma.mv()` (for example `"REML"`).
`vi_floor`	Variance-flooring mode passed to `panoramic_meta_mv()`. Default is `"group_median"` (`"median"`, `"group_median"`, or `"none"`).
`seed`	Optional random seed for reproducibility.
`boot`	Bootstrap mode in `panoramic_spatialstats()`: `"approx"` or `"block"`.
`tile_size`	Optional tile size for block bootstrap (same units as coordinates).
`nx, ny`	Optional tile counts for block bootstrap (used when `boot = "block"`).
`BPPARAM`	BiocParallel backend.
`verbose`	Logical verbosity passed to bootstrap routines.

Details

`panoramic()` delegates directly to `panoramic_analyze()`.

Value

A list with:

- `prep`: output of `panoramic_prepare()`.
- `stats`: output of `panoramic_spatialstats()`.
- `pooled`: output of `panoramic_meta_mv()`.
- `tables`: pre-flattened data.frames (spatialstats/meta/contrast).

Examples

```
toy <- panoramic_simulate_dataset(seed = 1)
out <- panoramic(
  spe_list = toy$spe_list,
  design = toy$design,
  cell_type = "cell_type",
  radii_um = c(10, 20),
  nsim = 5,
  min_cells = 2,
  window = "rect",
  BPPARAM = BiocParallel::SerialParam()
)
names(out)
```

`panoramic_analyze` *Run PANORAMIC end-to-end, including pooling/meta-analysis*

Description

`panoramic_analyze()` provides a streamlined API for the most common workflow: prepare data, compute spatial statistics, and run pooling/meta-analysis in one call.

Usage

```
panoramic_analyze(
  spe_list,
  design,
  cell_type = "cell_type",
  pairs = "auto",
  radii_um,
  stat = "local_comp_enrichment",
  nsim = 100,
  correction = NULL,
  min_cells = 5L,
  concavity = 50,
  window = c("concave", "convex", "rect"),
  group_col = "group",
  group_tau2 = c("none", "separate"),
```

```
  patient_col = NULL,
  sample_col = NULL,
  tau_structure = c("patient", "patient_sample"),
  method_mv = "REML",
  vi_floor = "group_median",
  seed = 123,
  boot = c("approx", "block"),
  tile_size = NULL,
  nx = NULL,
  ny = NULL,
  BPPARAM = BiocParallel::SerialParam(),
  verbose = FALSE
)
```

Arguments

`spe_list`	Named list of SpatialExperiment objects (one per sample).
`design`	data.frame with at least columns `sample` and `group`.
`cell_type`	Character; colData column containing cell-type labels.
`pairs`	Either `"auto"` or a data.frame with columns `ct1`, `ct2`.
`radii_um`	Numeric vector of radii in microns.
`stat`	Character spatial statistic passed to `panoramic_spatialstats()`.
`nsim`	Integer bootstrap replicates.
`correction`	Optional edge-correction method for spatstat-based statistics. Ignored when `stat = "local_comp_enrichment"`.
`min_cells`	Minimum cells per type per sample.
`concavity`	Concavity for concave hull windows.
`window`	One of `"concave"`, `"convex"`, or `"rect"`.
`group_col`	Character group column for meta-analysis (defaults to `"group"`).
`group_tau2`	Controls whether PANORAMIC additionally computes group-specific heterogeneity (`tau2`) by fitting per-group multilevel models. Use `"none"` for faster fitting or `"separate"` for more detailed heterogeneity summaries.
`patient_col`	Optional patient-id column for multilevel pooling. If `NULL`, PANORAMIC uses `"patient"` from `design` when available, otherwise falls back to `sample_col`.
`sample_col`	Optional sample-id column used in `panoramic_meta_mv()`. If `NULL`, PANORAMIC uses `"sample"` (aligned to `design$sample`).
`tau_structure`	Random-effects structure passed to `panoramic_meta_mv()`.
`method_mv`	Method passed to `metafor::rma.mv()` in `panoramic_meta_mv()`.
`vi_floor`	Variance-flooring mode in `panoramic_meta_mv()`. Default is `"group_median"`. Use `"median"` to replace non-positive `vi` with per-feature median positive `vi`, or `"none"` to disable flooring.
`seed`	Optional random seed.
`boot`	Bootstrap mode `"approx"` or `"block"`.
`tile_size`	Optional tile size for block bootstrap.
`nx, ny`	Optional tile counts for block bootstrap.
`BPPARAM`	BiocParallel backend.
`verbose`	Logical verbosity passed to `panoramic_spatialstats()`.

Value

A list with:

- `prep`: output of `panoramic_prepare()`
- `stats`: output of `panoramic_spatialstats()`
- `pooled`: output of `panoramic_meta_mv()`
- `tables`: pre-flattened data.frames for convenient result extraction

Examples

```
toy <- panoramic_simulate_dataset(seed = 1)
out <- panoramic_analyze(
  spe_list = toy$spe_list,
  design = toy$design,
  cell_type = "cell_type",
  radii_um = c(10, 20),
  stat = "local_comp_enrichment",
  nsim = 5,
  min_cells = 2,
  window = "rect",
  group_col = "group",
  BPPARAM = BiocParallel::SerialParam()
)
names(out)
```

`panoramic_extract_contrast` *Extract PANORAMIC Contrast, Meta, or Spatialstats Tables*

Description

Convenience helper for flattening PANORAMIC outputs into analysis-ready tables.

Usage

```
panoramic_extract_contrast(
  se,
  feature_cols = c("ct1", "ct2", "radius_um"),
  what = c("contrast", "meta", "spatialstats"),
  drop_na = FALSE
)
```

Arguments

`se`	A `SummarizedExperiment` from PANORAMIC workflow steps.
`feature_cols`	Feature columns to include when available (for example `ct1`, `ct2`, `radius_um`).
`what`	Which table to extract: `"contrast"`, `"meta"`, or `"spatialstats"`.
`drop_na`	If `TRUE`, drop rows with missing extracted statistics.

Value

A `data.frame`. For `what = "contrast"`, the table includes `beta_diff`, `se_diff`, `z_diff`, `p_diff`, and `fdr_diff`, plus available feature columns.

Examples

```
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(dummy = matrix(0, nrow = 1, ncol = 1)),
  rowData = S4Vectors::DataFrame(
    ct1 = "A",
    ct2 = "B",
    radius_um = 10,
    beta_diff = 0.25,
    se_diff = 0.10,
    z_diff = 2.5,
    p_diff = 0.012,
    fdr_diff = 0.02
  )
)

panoramic_extract_contrast(se)
```

`panoramic_meta_mv` *Multilevel Random-Effects Meta-Analysis of PANORAMIC Features*

Description

Feature-wise multilevel random-effects pooling for PANORAMIC spatial statistics using `metafor::rma.mv()`, accounting for clustering of samples within patients.

Usage

```
panoramic_meta_mv(
  se,
  patient_col,
  group_col = NULL,
  sample_col = "sample",
  tau_structure = c("patient", "patient_sample"),
  method = "REML",
  group_tau2 = c("none", "separate"),
  warn_sigma2 = TRUE,
  vi_floor = NULL,
  sigma2_tol = 1e-06,
  sigma2_rel = 1e-04,
  control = NULL,
  sparse = FALSE,
  BPPARAM = BiocParallel::SerialParam()
)
```

Arguments

`se`	A `SummarizedExperiment` with assays `yi` and `vi`.
`patient_col`	Column in `colData(se)` identifying patients.
`group_col`	Optional grouping column. When provided, pooled means are estimated with `mods = ~ 0 + group`. For two groups, contrast is `beta_diff = group2 - group1`; group order is factor levels (if factor) or lexicographic order of labels.
`sample_col`	Optional sample-id column; if `NULL`, `colnames(se)` are used.
`tau_structure`	Random-effects structure: `"patient"` or `"patient_sample"`.
`method`	Estimator passed to `metafor::rma.mv()` (for example `"REML"`).
`group_tau2`	If `"separate"`, additionally fit per-group multilevel models to estimate group-specific heterogeneity components.
`warn_sigma2`	If `TRUE`, warn when many features have near-zero variance components.
`vi_floor`	Optional handling for non-positive `vi`: `"median"`, `"group_median"`, or `"none"`.
`sigma2_tol`	Absolute tolerance for near-zero `sigma2` diagnostics.
`sigma2_rel`	Relative tolerance (fraction of median `vi`) for near-zero `sigma2` diagnostics.
`control`	Optional named optimizer control list passed to `metafor::rma.mv()`.
`sparse`	Logical forwarded to `metafor::rma.mv(sparse = ...)`.
`BPPARAM`	BiocParallel backend used across features.

Value

The input `SummarizedExperiment` with meta-analysis columns appended to `rowData(se)`. For grouped analyses, columns are group-prefixed (`*_mu_hat`, `*_se_mu`, `*_p_mu`, `*_k`). For exactly two groups, contrast columns `beta_diff`, `se_diff`, `z_diff`, `p_diff`, `fdr_diff` are added. Metadata are stored in `metadata(se)$panoramic$meta_mv`.

Examples

```
yi <- matrix(c(0.20, 0.15, 0.10, 0.05), nrow = 1)
vi <- matrix(c(0.04, 0.05, 0.06, 0.05), nrow = 1)
colnames(yi) <- colnames(vi) <- paste0("s", seq_len(4L))

se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(yi = yi, vi = vi),
  rowData = S4Vectors::DataFrame(ct1 = "A", ct2 = "B", radius_um = 10),
  colData = S4Vectors::DataFrame(
    sample = paste0("s", seq_len(4L)),
    patient = paste0("p", seq_len(4L)),
    group = c("control", "control", "case", "case")
  )
)

se_mv <- panoramic_meta_mv(
  se = se,
  patient_col = "patient",
  group_col = "group",
  sample_col = "sample",
  BPPARAM = BiocParallel::SerialParam()
)

SummarizedExperiment::rowData(se_mv)$beta_diff
```

`panoramic_prepare` *Prepare PANORAMIC inputs from a list of SpatialExperiment objects*

Description

Creates SpatialExperiment objects ready for PANORAMIC spatial analyses. Cell type labels are harmonized, rare cell types (fewer than `min_cells`) are dropped per sample, and a spatial window is computed. Cached spatstat objects are stored within each SpatialExperiment's metadata.

Usage

```
panoramic_prepare(
  spe_list,
  design,
  cell_type = "cell_type",
  min_cells = 5,
  concavity = 50,
  window = c("concave", "convex", "rect"),
  BPPARAM = BiocParallel::SerialParam()
)
```

Arguments

`spe_list`	Named or unnamed list of SpatialExperiment (one per sample)
`design`	data.frame with at least columns `sample`, `group` to map samples for meta-analysis. If only one group is used, give all the same group label.
`cell_type`	Character; name of SpatialExperiment colData column holding cell type labels
`min_cells`	Integer. Cell types with fewer than this count (per sample) are dropped.
`concavity`	Numeric passed to concaveman::concaveman(). Controls level of hull detail. 1 is highly detailed, `Inf` is a convex hull.
`window`	one of "concave","convex","rect". Typically use concave.
`BPPARAM`	BiocParallel param for optional parallel processing.

Details

This step computes per-sample spatial windows to exclude background, filters rare cell types separately per sample, builds consistent cell-type factor levels, and caches spatstat objects and type tables for PANORAMIC's spatial statistics.

Value

List of SpatialExperiment objects with metadata slot `panoramic` containing `ppp`, cell-type table, spatial window, group/sample info.

Examples

```
spe_list <- list(
  sample1 = panoramic_simulate_spe(
    n_cells = 60,
    sample_id = "sample1",
    scenario = "random",
    seed = 1
  )
)

# Design with a single group ----------------------------------------
design <- data.frame(
  sample = "sample1",
  group  = "group1",
  stringsAsFactors = FALSE
)

# Run panoramic_prepare ---------------------------------------------
prepped <- panoramic_prepare(
  spe_list,
  design      = design,
  cell_type   = "cell_type",
  min_cells   = 3,
  concavity   = 50,
  window      = "concave",
  BPPARAM     = BiocParallel::SerialParam()
)

# Inspect cached spatstat objects in metadata
names(S4Vectors::metadata(prepped[[1]])$panoramic)
```

`panoramic_simulate_dataset`

Simulate a two-group PANORAMIC example dataset

Description

Create a list of `SpatialExperiment` objects and matching design table for differential colocalization tutorials.

Usage

```
panoramic_simulate_dataset(
  n_group1 = 3L,
  n_group2 = 3L,
  n_cells_group1 = 200L,
  n_cells_group2 = 350L,
  group_labels = c("group1", "group2"),
  scenario_group1 = "random",
  scenario_group2 = "colocalized",
  seed = NULL
)
```

Arguments

`n_group1`	Integer number of samples in group 1.
`n_group2`	Integer number of samples in group 2.
`n_cells_group1`	Integer number of cells per group-1 sample.
`n_cells_group2`	Integer number of cells per group-2 sample.
`group_labels`	Character length-2 vector of group names.
`scenario_group1`	Scenario passed to `panoramic_simulate_spe()` for group 1.
`scenario_group2`	Scenario passed to `panoramic_simulate_spe()` for group 2.
`seed`	Optional integer seed for reproducibility.

Value

A list with entries `spe_list` and `design`.

Examples

```
toy <- panoramic_simulate_dataset(seed = 1)
names(toy)
head(toy$design)
```

`panoramic_simulate_spe` *Simulate one SpatialExperiment object for PANORAMIC examples*

Description

Create a toy `SpatialExperiment` with simple spatial patterns that can be used in package examples, vignettes, and tests.

Usage

```
panoramic_simulate_spe(
  n_cells = 200L,
  sample_id = "sample_1",
  cell_types = c("A", "B", "C"),
  scenario = c("random", "colocalized"),
  bounds = c(0, 100),
  center = c(50, 50),
  cluster_sd = 18,
  n_genes = 10L,
  seed = NULL
)
```

Arguments

`n_cells`	Integer number of cells to simulate.
`sample_id`	Character sample identifier stored in `colData`.
`cell_types`	Character vector of cell-type labels.
`scenario`	Character string, either `"random"` or `"colocalized"`. In `"colocalized"`, the first two cell types are enriched near the center.
`bounds`	Numeric length-2 vector giving minimum and maximum coordinate.
`center`	Numeric length-2 vector giving center for colocalized pattern.
`cluster_sd`	Numeric standard deviation for central clustering.
`n_genes`	Integer number of toy genes in the counts matrix.
`seed`	Optional integer for reproducibility.

Value

A `SpatialExperiment` object with simulated coordinates and cell types.

Examples

```
spe <- panoramic_simulate_spe(
  n_cells = 120,
  sample_id = "sample_1",
  scenario = "colocalized",
  seed = 1
)
spe
```

---

panoramic_spatialstats

Compute Pairwise Spatial Statistics for PANORAMIC

Description

Compute pairwise spatial summary curves and bootstrap variances across samples for all requested cell-type pairs and radii.

Usage

```
panoramic_spatialstats(
  prep,
  pairs = "auto",
  radii_um,
  stat = "local_comp_enrichment",
  nsim = 100,
  correction = "translate",
  seed = 123,
  boot = c("approx", "block"),
  tile_size = NULL,
  nx = NULL,
```

```
  ny = NULL,
  BPPARAM = BiocParallel::SerialParam(),
  verbose = FALSE
)
```

Arguments

`prep`	List of prepared SpatialExperiment objects from `panoramic_prepare()`.
`pairs`	Either `"auto"` or a data.frame with columns `ct1`, `ct2`.
`radii_um`	Numeric vector of radii (microns).
`stat`	Summary statistic. Default is `"local_comp_enrichment"`. Other supported values are `"Lcross"`, `"Kcross"`, `"Lest"`, and `"Kest"`.
`nsim`	Integer number of Loh bootstrap replicates per sample/pair.
`correction`	Edge correction method passed to spatstat summaries.
`seed`	Optional seed for reproducible bootstrap sampling.
`boot`	Bootstrap mode: `"approx"` or `"block"`.
`tile_size`	Optional tile size for block bootstrap in coordinate units.
`nx`, `ny`	Optional tile counts for block bootstrap when `boot = "block"`.
`BPPARAM`	BiocParallel backend used across samples.
`verbose`	If `TRUE`, show bootstrap console output.

Details

For L-function statistics, PANORAMIC computes variances via the delta method from corresponding K-function bootstrap estimates and centers by `r`.

For `stat = "local_comp_enrichment"`, PANORAMIC reports edge-corrected observed-minus-expected local target composition (percentage points) and uses Loh bootstrap for uncertainty.

Value

A `SummarizedExperiment` with assays:

- `yi`: centered estimates per `(ct1, ct2, radius_um)` feature and sample.
- `vi`: variance estimates aligned to `yi`.

`rowData` contains `ct1`, `ct2`, `radius_um`, `stat`; `colData` contains `sample` and `group`.

Examples

```
toy <- panoramic_simulate_dataset(seed = 1)
prep <- panoramic_prepare(
  spe_list = toy$spe_list,
  design = toy$design,
  cell_type = "cell_type",
  min_cells = 2,
  window = "rect",
  BPPARAM = BiocParallel::SerialParam()
)
se_stats <- panoramic_spatialstats(
  prep = prep,
  radii_um = c(10, 20),
```

```
  nsim = 5,
  correction = "translate",
  seed = 1,
  BPPARAM = BiocParallel::SerialParam()
)
se_stats
```

---

plot_forest *Forest Plot for PANORAMIC Spatial Colocalization*

Description

Draw individual-sample estimates and pooled group estimates for one cell-type pair at one radius.

Usage

```
plot_forest(
  se_meta,
  ct1,
  ct2,
  radius_um = NULL,
  group_col = "group",
  group_colors = NULL,
  text_size = 2,
  header_text_size = 2,
  title_text_size = 8,
  base_size = 5,
  show_est_se = TRUE,
  show_ci = TRUE
)
```

Arguments

`se_meta`	SummarizedExperiment from `panoramic_meta_mv()` containing `yi`/`vi` assays and pooled columns in `rowData`.
`ct1, ct2`	Cell-type labels selecting one feature.
`radius_um`	Radius (microns). If `NULL`, the first available radius for the pair is used.
`group_col`	Group column in `colData(se_meta)`.
`group_colors`	Optional named color vector for groups.
`text_size`	Text size for row annotations.
`header_text_size`	Text size for table-like column headers.
`title_text_size`	Title text size.
`base_size`	Base size passed to `theme_classic()`.
`show_est_se`	Show the `Est (SE)` text column.
`show_ci`	Show the `95% CI` text column.

Details

This helper currently supports exactly two groups in `group_col`.

Value

A `ggplot` forest-plot object.

Examples

```
yi <- matrix(c(0.12, 0.18, 0.35, 0.30), nrow = 1)
vi <- matrix(c(0.03, 0.04, 0.05, 0.05), nrow = 1)
colnames(yi) <- colnames(vi) <- paste0("s", seq_len(4L))

se_meta <- SummarizedExperiment::SummarizedExperiment(
  assays = list(yi = yi, vi = vi),
  rowData = S4Vectors::DataFrame(
    ct1 = "A",
    ct2 = "B",
    radius_um = 10,
    control_mu_hat = 0.15,
    control_se_mu = 0.10,
    case_mu_hat = 0.32,
    case_se_mu = 0.11
  ),
  colData = S4Vectors::DataFrame(
    group = c("control", "control", "case", "case")
  )
)

p <- plot_forest(
  se_meta,
  ct1 = "A",
  ct2 = "B",
  radius_um = 10,
  group_col = "group"
)

p
```

`plot_representative_samples` *Plot Representative Samples for Significant Differential Hits*

Description

Select representative samples per group for significant differential colocalization features, then build side-by-side spatial panels highlighting source and target cell types.

Usage

```
plot_representative_samples(
  se_stats,
  se_meta,
```

```
  spe_list,
  top_hits = NULL,
  sig_col = c("fdr_diff", "p_diff"),
  alpha = 0.05,
  top_n = 10L,
  group_col = "group",
  cell_type_col = "cell_type",
  sample_col = "sample",
  out_prefix = NULL
)
```

Arguments

`se_stats`	A `SummarizedExperiment` from `panoramic_spatialstats()` containing sample-level `yi`.
`se_meta`	A `SummarizedExperiment` from `panoramic_meta_mv()` containing contrast columns.
`spe_list`	Named list of `SpatialExperiment` objects.
`top_hits`	Optional data.frame of selected features. If `NULL`, features are selected from `se_meta` using `sig_col`, `alpha`, and `top_n`.
`sig_col`	One of `"fdr_diff"` or `"p_diff"` used when selecting `top_hits` = `NULL`.
`alpha`	Numeric threshold for feature selection when `top_hits` = `NULL`.
`top_n`	Integer number of selected hits when `top_hits` = `NULL`.
`group_col`	Character group column in `colData(se_stats)`.
`cell_type_col`	Character cell-type column in `colData(spe)`.
`sample_col`	Character sample-id column in `colData(se_stats)`.
`out_prefix`	Optional output prefix. If provided, each panel is saved as PNG/PDF and the returned index includes file paths.

Details

This helper currently supports exactly two groups in `group_col`.

Value

A list with `plots` (named list of ggplot objects) and `index` (data.frame summarizing selected hits and samples).

Examples

```
sample_ids <- paste0("sample_", seq_len(4L))
spe_list <- stats::setNames(
  lapply(sample_ids, function(sid) {
    panoramic_simulate_spe(
      n_cells = 40,
      sample_id = sid,
      cell_types = c("A", "B"),
      scenario = "random",
      seed = 1
    )
  }),
  sample_ids
```

```
)

yi <- matrix(c(0.10, 0.15, 0.25, 0.30), nrow = 1)
vi <- matrix(rep(0.05, 4), nrow = 1)
colnames(yi) <- colnames(vi) <- sample_ids
se_stats <- SummarizedExperiment::SummarizedExperiment(
  assays = list(yi = yi, vi = vi),
  rowData = S4Vectors::DataFrame(ct1 = "A", ct2 = "B", radius_um = 10),
  colData = S4Vectors::DataFrame(
    sample = sample_ids,
    group = c("control", "control", "case", "case")
  )
)

se_meta <- SummarizedExperiment::SummarizedExperiment(
  assays = list(dummy = matrix(0, nrow = 1, ncol = 1)),
  rowData = S4Vectors::DataFrame(
    ct1 = "A",
    ct2 = "B",
    radius_um = 10,
    beta_diff = 0.25,
    se_diff = 0.10,
    z_diff = 2.5,
    p_diff = 0.012,
    fdr_diff = 0.02,
    control_mu_hat = 0.125,
    case_mu_hat = 0.275
  )
)

out <- plot_representative_samples(
  se_stats = se_stats,
  se_meta = se_meta,
  spe_list = spe_list,
  sig_col = "p_diff",
  alpha = 0.05,
  top_n = 1
)
names(out)
```

`plot_spatial_network` *Plot PANORAMIC spatial colocalization network*

Description

Wrapper to construct and visualize a PANORAMIC spatial colocalization network. It builds the directed network via `create_spatial_network()` and then renders it with ggraph/tidygraph.

Usage

```
plot_spatial_network(
  se_diff,
  fdr_threshold = 0.05,
  leiden_resolution = 1,
```

```
    z_sign = c("both", "positive", "negative"),
    include_nonsig = FALSE,
    nonsig_max_fdr = 1,
    directed = FALSE,
    sig_operator = c("lt", "gt"),
    layout = "fr",
    node_size_by = "degree",
    label_repel = TRUE,
    label_box_padding = 0.2,
    label_point_padding = 0.1,
    label_force = 0.8,
    return_net = FALSE
)
```

Arguments

`se_diff`	A SummarizedExperiment returned by `panoramic_meta_mv()`.
`fdr_threshold`	Numeric threshold used for significance and edge classification.
`leiden_resolution`	Numeric Leiden resolution passed to `create_spatial_network()`.
`z_sign`	Character sign filter for `z_diff` (`"both"`, `"positive"`, `"negative"`).
`include_nonsig`	Logical; include non-significant edges.
`nonsig_max_fdr`	Numeric max FDR retained when `include_nonsig` = `TRUE`.
`directed`	Logical; pass through to `create_spatial_network()`.
`sig_operator`	One of `"lt"` or `"gt"` passed through to `create_spatial_network()` for threshold direction.
`layout`	Character string specifying the graph layout passed to ggraph (e.g. "fr", "kk", "stress"). Default "fr".
`node_size_by`	Character name of a vertex attribute used to scale node sizes (e.g. "degree", "strength", "betweenness"). Default "degree".
`label_repel`	Logical; if TRUE use repelled node labels.
`label_box_padding`	Numeric box padding for repelled labels.
`label_point_padding`	Numeric point padding for repelled labels.
`label_force`	Numeric repulsion force for labels.
`return_net`	Logical; if TRUE return a list with `plot` and `net`.

Value

A ggplot object by default, or a list with `plot` and `net` when `return_net` = `TRUE`.

Examples

```
se_diff <- SummarizedExperiment::SummarizedExperiment(
  assays = list(dummy = matrix(0, nrow = 3, ncol = 1)),
  rowData = S4Vectors::DataFrame(
    ct1 = c("A", "A", "B"),
    ct2 = c("B", "C", "C"),
    z_diff = c(2.0, -1.5, 1.2),
```

```
    p_diff = c(0.01, 0.03, 0.20),
    fdr_diff = c(0.02, 0.05, 0.30)
  )
)

if (requireNamespace("ggraph", quietly = TRUE) &&
    requireNamespace("tidygraph", quietly = TRUE)) {
  p_net <- plot_spatial_network(
    se_diff,
    fdr_threshold = 0.2,
    z_sign = "both",
    include_nonsig = TRUE,
    layout = "fr",
    node_size_by = "degree"
  )
  print(p_net)
}
```

`plot_volcano` *Volcano plot for differential spatial colocalization*

Description

Updated volcano helper based on the CRC TMA manuscript workflow. It uses `beta_diff` for the x-axis and `-log10(p_diff)` for the y-axis, and colors points by significance direction.

Usage

```
plot_volcano(
  se_diff,
  sig_col = c("fdr_diff", "p_diff"),
  alpha = 0.05,
  x_scale = c("beta_diff", "log2fc"),
  label_top = 12L,
  label_text_pt = 4,
  p_floor = 1e-300,
  title = NULL
)
```

Arguments

`se_diff`	A `SummarizedExperiment` with contrast columns in `rowData(se_diff)` (at least `ct1`, `ct2`, `beta_diff`, `p_diff`; and `fdr_diff` when `sig_col = "fdr_diff"`).
`sig_col`	Character, one of `"fdr_diff"` or `"p_diff"`. Controls which column determines significance coloring.
`alpha`	Numeric threshold for significance based on `sig_col`.
`x_scale`	Character, one of `"beta_diff"` or `"log2fc"`. `"log2fc"` uses `sign(beta_diff) * log2(1 + abs(beta_diff))`.
`label_top`	Integer number of significant hits to label. Use `NULL` or `0` to disable labels.
`label_text_pt`	Numeric font size (points) for significant-hit labels.
`p_floor`	Numeric floor applied to `p_diff` before `-log10()`.
`title`	Optional character title.

Value

A `ggplot` object.

Examples

```
se_diff <- SummarizedExperiment::SummarizedExperiment(
  assays = list(dummy = matrix(0, nrow = 2, ncol = 1)),
  rowData = S4Vectors::DataFrame(
    ct1 = c("A", "A"),
    ct2 = c("B", "C"),
    radius_um = c(10, 10),
    beta_diff = c(0.4, -0.3),
    p_diff = c(0.01, 0.20),
    fdr_diff = c(0.02, 0.30)
  )
)
m <- S4Vectors::metadata(se_diff)
m$panoramic <- list(contrast = list(control = "control", case = "case"))
S4Vectors::metadata(se_diff) <- m
plot_volcano(se_diff, sig_col = "fdr_diff")
```

Index