Package: edgeicg (via r-universe)

November 24, 2026

Type Package

Title Spillover-Aware Interaction-Changed Genes at Contact, Paracrine and Niche Scale

Version 0.3.0

Description Tests whether a receiver cell type changes a gene's expression with its exposure to a sender cell type, at three spatial scales: touching cells (contact), short-range neighbours (paracrine) and the regional neighbourhood (niche). Transcripts leaked from touching neighbours are modelled from the data and removed per cell before testing. Niche effects are screened with a calibrated marginal test (toroidal-shift null) and attributed to senders by exact fine-mapping. Counts are read in gene blocks through a provider, so whole sections run in bounded memory, in memory or from a disk-backed store, sequentially or on 'mirai' daemons.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.2)

Imports Matrix, data.table, methods, RANN, splines, stats, utils

Suggests edgeicgrs, mirai, testthat (>= 3.0.0), GiottoDisk, arrow, knitr, rmarkdown

Remotes drieslab/edgeicgrs

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

Repository https://drieslab.r-universe.dev

Date/Publication 2026-11-24 21:49:09 UTC

RemoteUrl https://github.com/drieslab/edgeicg

RemoteRef HEAD

RemoteSha 100330cf95edfe739a28d70cdf8095f950e4ea35

Contents

contact_from_polygons . . . 2
counts_provider . . . 3
edgeicg . . . 4
edgeicg_classify . . . 7
edgeicg_exposures . . . 8
edgeicg_local_results . . . 9
edgeicg_patches . . . 10
edgeicg_region_results . . . 10
edgeicg_results . . . 11
simulate_edgeicg . . . 12
spatial_basis . . . 13

Index **14**

`contact_from_polygons` *Touching-cell edges from cell outlines*

Description

For every outline vertex, vertices of other cells within `tau` microns count as touching; the vertex contributes half the length of its two adjacent outline segments. The shared length of (i, j) is the mean of both directions. Used as the spillover graph (which neighbours can leak transcripts).

Usage

```
contact_from_polygons(poly, n, tau = 2, max_k = 6)
```

Arguments

`poly`	data.frame with `cell` (index 1..n into the analysis cells), `x`, `y`: outline vertices in drawing order per cell.
`n`	number of cells.
`tau`	touching distance (microns).
`max_k`	neighbouring vertices searched per vertex.

Value

data.frame of directed edges `recv`, `send`, `w` (shared length, microns).

counts_provider *Counts providers*

Description

EdgeICG reads raw counts only in blocks. A provider is a list with

- genes (character) and n_cells;
- rows(lo, hi, gi): raw counts of cells lo:hi (in the provider's current cell order) for genes gi, as a cells x genes matrix (sparse dgCMatrix preferred; dense is accepted);
- order(ord): a provider whose cell i is this provider's cell ord[i]. The engine visits cells in spatially ordered windows (see window_cells in edgeicg()) and asks the provider for that order once;
- cleanup(): removes anything order() created;
- narrow(gi): a provider for genes gi only (used by the gene prefilter);
- split(list_of_gi): one provider per gene panel (windowed runs with more than panel_genes genes);
- optionally slice(lo, hi, gi): the same block as a compact genes x cells matrix, for providers whose data live in memory (the block is sent to the worker with its task);
- read(gi): all cells for genes gi (cells x genes), kept for direct use.

Usage

```
counts_provider(counts)

counts_provider_store(store, genes, cells)
```

Arguments

counts	genes x cells matrix (dgCMatrix or dense), a GiottoDisk expression store (wrapped with counts_provider_store() for all its genes and cells), or an existing provider (returned as is).
store	a GiottoDisk expression store (genes x cells), e.g. GiottoClass::getExpression(g, values = "raw", output = "exprObj")[].
genes	genes to analyse (store feature IDs).
cells	cell IDs in analysis order (store cell IDs).

Details

counts_provider() wraps an in-memory genes x cells matrix (sparse or dense). counts_provider_store() wraps a GiottoDisk expression store, so whole sections never sit in memory: workers read their own windows from it.

For windowed runs, order() bakes the analysis genes into a temporary store in the engine's cell order (GiottoDisk::storeWrite() of the view). The expression store is sorted by cell, so a window and its halo are then a contiguous row range that prunes parquet row groups, and only the analysis genes are scanned. The temporary store is removed by cleanup().

Value

a provider list.

`edgeicg` *Spillover-aware interaction-changed genes at three scales*

Description

Tests, for every receiver type, sender type and gene, whether the receiver's expression changes with its exposure to the sender at contact, paracrine and niche scale, after removing transcripts leaked from touching neighbours. See the package description for the model.

Usage

```
edgeicg(
  counts,
  coords,
  types,
  lib = NULL,
  spill_edges = NULL,
  ex = NULL,
  block_genes = 500L,
  window_cells = NULL,
  panel_genes = 3000L,
  covariates = NULL,
  covariate_scope = c("all", "niche"),
  workers = 0L,
  backend = c("auto", "rust", "r"),
  threads = NULL,
  calibrate = TRUE,
  calib_cond = TRUE,
  attribute = TRUE,
  B = 30L,
  min_shift = 450,
  q_thr = 0.05,
  min_cells = 60L,
  min_det = 0.01,
  min_exposed = 30L,
  exposed_at = 0.1,
  drop_degenerate = TRUE,
  prefilter = TRUE,
  seed = 1L,
  verbose = TRUE,
  region = NULL,
  local = FALSE,
  patch_size = 1000,
```

```
  min_cells_region = 50L,
  min_exposed_region = 20L,
  genes = NULL,
  leak_genes = c("all", "tested"),
  leak = NULL,
  ...
)
```

Arguments

counts	raw counts, genes x cells: a dgCMatrix or dense matrix, a GiottoDisk expression store, or a provider from counts_provider() / counts_provider_store(). Give all genes (or a broad set) and choose the genes to test with genes: the leak coefficients are fitted across the genes of counts (see leak_genes). The library size comes from lib.
coords	n x 2 cell centroids (microns), in the provider's cell order.
types	cell-type labels (character or factor), one per cell.
lib	library size per cell over **all** genes (default: column sums of counts if it is a matrix).
spill_edges	optional touching-cell edge list (recv, send, w), e.g. from contact_from_polygons(); the spillover graph. Default: the kNN contact graph.
ex	optional exposures from edgeicg_exposures() (computed if NULL).
block_genes	genes per block (memory: about cells of a type x block_genes x 8 bytes per copy).
window_cells	cells per window. With more cells than this, cells are visited in spatially ordered windows (each with the halo of touching neighbours the leak correction needs) and every statistic is accumulated as a sum over windows, so memory no longer grows with the section. NULL (default): one window for in-memory counts; for a store, one window per worker, at most getOption("edgeicg.window_cells", 50000) cells (the store is then read once per sweep, in contiguous cell ranges). Results equal the one-window run up to rounding.
panel_genes	with several windows, genes are processed in panels of at most this many (each a separately baked store when counts are on disk), which bounds the memory of the partial sums; every count is still read once per pass. Default 3000.
covariates	optional numeric matrix (cells x q) of covariates added to every receiver type's design and adjusted for but not tested, e.g. a zonation score or spatial_basis() to absorb large-scale spatial trends (zonation, sub-state gradients) that otherwise correlate with exposure. Columns without variance within a receiver type are dropped (with drop_degenerate).
covariate_scope	"all" (default): covariates enter every test's design; "niche": only the niche screen's base design (and its calibration). Large-scale spatial trends confound mainly the niche scale; on the Atera kidney section spatial_basis() brought the niche pseudo-null from 4.7% to 0.5-0.7% with either scope, at about 1.4x the run time (df 4 or 6, either scope; the niche screen and its calibration dominate the extra cost). "niche" leaves the contact and paracrine results exactly as without covariates.

`workers` number of mirai daemons (0 = sequential). One window: tasks are gene blocks; several windows: tasks are bands of windows. Results are identical.

`backend` `"auto"` (Rust kernels from the edgeicgrs package when installed, else R), `"rust"` or `"r"`. The Rust kernels are the matrix cross-products of the sweeps, the hotspot of the method.

`threads` threads per process for the Rust kernels (default: 1 with workers, all cores without).

`calibrate` run the toroidal-shift calibration of the niche scale (recommended).

`calib_cond` also calibrate the conditional (all-sender) niche z (secondary output).

`attribute` attribute niche screen hits to senders.

`B`, `min_shift` number of shifts and minimum shift (microns) of the calibration.

`q_thr` FDR threshold for the niche screen hits that are attributed.

`min_cells`, `min_det`, `min_exposed`, `exposed_at` testability: receiver types with at least `min_cells` cells; genes detected in at least `min_det` of receivers; at least `min_exposed` receivers with exposure above `exposed_at`.

`drop_degenerate` drop exposure columns without variance within a receiver type (their tests are NA). FALSE keeps them in the design.

`prefilter` drop genes detected in fewer than `min_det` of the cells of every receiver type before the fit (one cheap detection pass). Such genes have no testable entry; without the prefilter they still enter the regression of the leak coefficients across genes.

`seed` seed of the calibration shifts.

`verbose` progress messages.

`region` optional region per cell (character or factor; NA = not in a region). The effect of every entry is then also estimated within each region, with a test of whether it differs between regions; see `edgeicg_region_results()`. The leak model and the other terms of the model stay global.

`local` experimental. TRUE: the same within square patches of `patch_size` microns (`edgeicg_patches()`), tested with Cochran's Q over patches; see `edgeicg_local_results()`. The Q test is calibrated but has little power for smooth or half-plane patterns; prefer `region` with labels.

`patch_size` patch side for `local` (microns).

`min_cells_region`, `min_exposed_region` a region or patch is used for a receiver type with at least `min_cells_region` of its cells, and for an entry with at least `min_exposed_region` exposed receivers.

`genes` genes to test (default: all genes of `counts`).

`leak_genes` genes the leak coefficients (lambda, mu per receiver-sender pair) are fitted on. `"all"` (default): all genes of `counts` that pass the prefilter, so the leak, and with it the contact results, do not change with the genes chosen for testing. This costs one extra pass over all genes (stage 1 only). `"tested"`: only the genes in `genes`; faster, but the leak then depends on that selection.

leak	optional leak coefficients to reuse instead of fitting them: an EdgeICG fit or its pairs table (columns receiver, sender, lambda, mu). Pairs without a value get no leak correction.
...	further engine settings: huber, scale, max_size, prior_sd, cs_level, k, sigma_p, sigma_n, bin.

Value

an EdgeICG object; see edgeicg_results().

`edgeicg_classify` *Classify hits by leakage risk*

Description

Flags hits whose gene marks another cell type, so leaked or mis-segmented transcripts can be told apart from candidate biology. Classes (first that applies):

Usage

```
edgeicg_classify(
  x,
  hits,
  markers = NULL,
  types_considered = NULL,
  spec_thr = 0.8,
  regional_n = 3
)
```

Arguments

x	an EdgeICG object.
hits	a table from edgeicg_results().
markers	optional named list (type -> character vector of marker genes).
types_considered	types used as "other" types (default: all fitted types).
spec_thr, regional_n	thresholds.

Details

- sender_marker: the gene marks the tested sender (sender specificity > spec_thr, or in markers).
- other_marker: the gene marks a third type (not the receiver).

- own_down: a gene of the receiver goes down next to the sender, where the receiver expresses it far more than the sender (receiver specificity against the sender > spec_thr, or in markers of the receiver). Next to that sender, mixed or mis-segmented cells and dilution by the sender's transcripts lower such genes, so these hits need checking before they are read as regulation.
- clean: none of the above.

Niche hits where the same sender and sign recur in at least regional_n receiver types are flagged regional (shared tissue zone or a regional technical artefact).

Specificity of gene g for type k against type r is m_kg / (m_kg + m_rg) with m the mean composition per type from the fit (x$type_mean).

Value

hits with columns class and regional.

---

edgeicg_exposures *Neighbourhood exposures at three scales*

Description

Every exposure is a *share*: for each cell, the fraction of its neighbourhood that belongs to each cell type (rows sum to 1), so local cell density does not enter.

Usage

```
edgeicg_exposures(
  coords,
  types,
  K = max(types),
  k = 8,
  sigma_p = 25,
  sigma_n = 150,
  bin = 10,
  max_k = 400
)
```

Arguments

coords	n x 2 matrix of cell centroids in microns.
types	integer cell-type codes 1..K.
K	number of types.
k	contact neighbours.
sigma_p, sigma_n	paracrine and niche kernel widths (microns).
bin	niche grid bin (microns).
max_k	cap on paracrine neighbours per cell.

Details

- **contact** (C): share of the k nearest neighbours' weight, w = exp(-d / median(d)).
- **paracrine** (P): Gaussian (sigma_p) share among cells within 3 * sigma_p that are not contact neighbours.
- **niche** (N): Gaussian (sigma_n) share from a per-type cell-count grid (bin bin) blurred with a separable filter, read back at each cell's bin. The grids are kept for the niche calibration.

Value

list(contact = edge list (recv, send, w), C, P, N (n x K), fields (niche grids), settings).

---

edgeicg_local_results *Local effects (experimental)*

Description

Experimental: the Q test over patches is calibrated but has little power for smooth or half-plane patterns. For a fit with local = TRUE: for every testable entry, whether its effect varies across patches of the tissue (Cochran's Q over patches, BH within scale) and, for entries with q_het < q_thr, the effect per patch with empirical-Bayes shrinkage towards the mean (effect_shrunk, sd_shrunk), to map where it is strongest.

Usage

```
edgeicg_local_results(
  x,
  scale = c("contact", "paracrine", "niche"),
  q_thr = 0.05
)
```

Arguments

x	an EdgeICG object fitted with local = TRUE.
scale	scales to report.
q_thr	threshold on q_het.

Value

list(entries = data.table(receiver, sender, gene, scale, global_effect, global_z, Q, df, tau2, q_het, local_frac), patches = data.table(receiver, sender, gene, scale, patch, x, y, n_exposed, effect, se, effect_shrunk, sd_shrunk)). local_frac: share of the exposed receivers that sit in patches whose shrunken effect is more than 2 posterior sd from zero, with the sign of the strongest patch.

edgeicg_patches *Square patches for local effects*

Description

Assigns each cell to a square patch of a grid laid over the tissue. Used by `edgeicg(local = TRUE)`.

Usage

```
edgeicg_patches(coords, size = 1000)
```

Arguments

`coords`	n x 2 cell centroids (microns).
`size`	patch side (microns).

Value

list(code = patch index per cell, labels, centers = data.frame(patch, x, y)).

edgeicg_region_results *Region-specific effects*

Description

For a fit with `region =`, the effect of every testable entry within each region, and a test of whether the effect differs between regions.

Usage

```
edgeicg_region_results(
  x,
  scale = c("contact", "paracrine", "niche"),
  q_thr = 0.05,
  min_lfc = 0.25,
  all = FALSE
)
```

Arguments

`x`	an `EdgeICG` object fitted with `region =`.
`scale`	scales to report.
`q_thr`	threshold on the region q and on `q_het`.
`min_lfc`	effect-size floor for a region to count as significant.
`all`	FALSE (default): only entries that are heterogeneous or significant in at least one region.

Details

Within a region the effect is the slope of the receiver's expression (leak-corrected, adjusted for the other senders and scales of the global model) on its exposure to the sender, with a region intercept. Between regions, Cochran's Q (df = regions - 1) tests heterogeneity; `q_het` is its BH-adjusted p within scale.

`pattern`, for entries with `q_het` < `q_thr`: `opposite` (significant in a region, q < `q_thr` and |lfc| >= `min_lfc`, and another region goes the other way with |z| >= 2), `specific` (otherwise significant in one region only), `variable` (heterogeneous otherwise); `shared` when `q_het` >= `q_thr`.

Value

data.table with one row per entry and region: receiver, sender, gene, scale, region, n_cells, n_exposed, effect, lfc, se, z, q, global_z, global_q, Q, df, q_het, pattern, class. `lfc` uses the pair's global exposure contrast (10th to 90th percentile), so it is comparable between regions. `class` is the entry's hit class (`edgeicg_classify()`): a region-dependent `sender_marker` hit often reflects a sender whose own expression differs between regions, not the receiver.

`edgeicg_results` *Results table*

Description

Results table

Usage

```
edgeicg_results(
  x,
  scale = c("contact", "paracrine", "niche"),
  q_thr = 0.05,
  min_lfc = 0.25
)
```

Arguments

`x`	an `EdgeICG` object.
`scale`	one or more of `"contact"`, `"paracrine"`, `"niche"` (niche = calibrated screen).
`q_thr`	keep entries with q below this (1 = all testable entries).
`min_lfc`	keep entries whose effect size, `lfc` (log2 fold change between receivers at the 10th and 90th percentile of their exposure to the sender), is at least this in absolute value. Default 0.25: at whole-section scale significance alone also picks up small confounded associations (zonation, sub-states); on the Atera kidney section the floor cut null-gene calls from 27% to 6% without losing planted effects of at least 0.5. Use 0 for significance only.

Value

data.table: receiver, sender, gene, scale, effect, lfc, se, z, q, plus for niche the top attributed sender (`top_sender`, `alpha`, `cs_size`).

`simulate_edgeicg` *Simulate a small tissue with known neighbour effects and leakage*

Description

Three cell types in smooth domains (cells about 10 microns apart). Ground truth the model never sees: directed contact effects (multiplicative, receiver 1 <- sender 2 up on `S1`, receiver 2 <- sender 3 down on `S2`), optional niche effects (receiver 1 genes `S3` respond to the regional density of type 2), and mass-conserving spillover between neighbours over a different graph than the model uses.

Usage

```
simulate_edgeicg(
  n = 4000,
  G = 200,
  seed = 1,
  kappa = 0.1,
  gamma = 1,
  niche_gamma = 0,
  spacing = 10,
  region_gamma = 0
)
```

Arguments

`n`	cells.
`G`	genes.
`seed`	seed.
`kappa`	mean leaked fraction.
`gamma`	contact log effect at full exposure.
`niche_gamma`	niche log effect.
`spacing`	cell spacing (microns).
`region_gamma`	contact log effect of region-dependent genes (0 = none). Regions are the left and right halves of the tissue (`region`). Receiver 1 <- sender 2: genes `S4` respond only in the right half, genes `S5` respond in opposite directions (up on the left, down on the right). Needs G >= 150.

Value

list(counts (genes x cells), coords, types, truth (data.table), lib, region (per cell); `truth` gains a `region` column ("all", "right" or "opposite")).

`spatial_basis` *Spatial basis for large-scale trends*

Description

A tensor-product natural-spline basis of the cell positions, to pass as `covariates` to `edgeicg()`: it absorbs smooth spatial trends (tissue zonation, gradients of cell states within a type) so that they are not mistaken for exposure effects. `df` per axis sets the scale of the trends removed: with a section of width W, trends longer than about W / df are absorbed.

Usage

```
spatial_basis(coords, df = 4L)
```

Arguments

`coords`	n x 2 cell centroids.
`df`	degrees of freedom per axis (default 4: 4 + 4 + 16 = 24 columns).

Value

a numeric matrix (cells x (2 df + df^2)), columns centred.

Index

contact_from_polygons, 2
contact_from_polygons(), *5*
counts_provider, 3
counts_provider(), *5*
counts_provider_store
        *(*counts_provider*)*, 3
counts_provider_store(), *5*

edgeicg, 4
edgeicg(), *3*, *13*
edgeicg_classify, 7
edgeicg_classify(), *11*
edgeicg_exposures, 8
edgeicg_exposures(), *5*
edgeicg_local_results, 9
edgeicg_local_results(), *6*
edgeicg_patches, 10
edgeicg_patches(), *6*
edgeicg_region_results, 10
edgeicg_region_results(), *6*
edgeicg_results, 11
edgeicg_results(), *7*

simulate_edgeicg, 12
spatial_basis, 13
spatial_basis(), *5*