

Package: SegStability (via r-universe)

October 20, 2026

Type Package

Title Descriptive Segmentation Sensitivity in Spatial Transcriptomics

Version 0.99.0

Description Quantifies correspondence between alternative cell segmentations of the same molecule-resolved spatial transcriptomics experiment. Preserves an immutable molecule universe, exact molecule identities, complete cell censuses, and explicit assignment statuses when constructing weighted shared-molecule graphs. Describes categorical cell-state stability, directional support sensitivity, optional spatial topology and polygon overlap, and agreement among stochastic segmentation runs. Results describe sensitivity to segmentation choices and do not estimate segmentation accuracy or establish biological ground truth.

License Artistic-2.0

Encoding UTF-8

Depends R (>= 4.5.0)

biocViews Spatial, Transcriptomics, SingleCell, DataRepresentation

Imports methods, S4Vectors, digest

Suggests testthat (>= 3.0.0), BiocStyle, knitr, rmarkdown, bit64, sf, MoleculeExperiment, SpatialFeatureExperiment

Config/testthat/edition 3

VignetteBuilder knitr

URL <https://github.com/ramakrishna-p21/SegStability>

BugReports <https://github.com/ramakrishna-p21/SegStability/issues>

Config/roxygen2/version 8.1.0

Repository <https://biocstaging.r-universe.dev>

Date/Publication 2026-10-19 22:55:25 UTC

RemoteUrl <https://github.com/BiocStaging/SegStability>

RemoteRef HEAD

RemoteSha 0f5a1bad6deba6addf4e7d341330d23252ad1d5a

Contents

SegStability-package	2
anchorTopology	3
compareGeometry	5
compareSegmentations	6
fromMoleculeExperiment	8
geometryFromSpatialFeatureExperiment	10
ovrlpy_validation_summary	11
replicateAgreement	12
SegmentationComparison-class	14
segmentationExample	16
SegmentationPartition-class	17
SegmentationSet-class	21
stateStability	24
supportSensitivity	26
validateCoordinates	28
validateGeometry	29
validateProvenance	30
validateSegmentation	31
validateSegmentationSet	32
Index	34

SegStability-package	<i>Descriptive segmentation sensitivity in spatial transcriptomics</i>
----------------------	--

Description

SegStability compares alternative cell segmentations of one immutable molecule universe. [SegmentationPartition\(\)](#) records each full cell census and the assigned, unassigned or ambiguous status of every molecule. [SegmentationSet\(\)](#) binds named partitions to the shared universe; retrieve one with `ss[["name"]]` and read keyed assignments with [assignmentData\(\)](#).

Details

[compareSegmentations\(\)](#) builds the complete shared-molecule graph with independent assigned denominators. Its edge, cell, component and status summaries retain empty cells and exact dominance ties. [stateStability\(\)](#) uses separately supplied categorical labels. [supportSensitivity\(\)](#) describes directional edge retention without changing the primary graph. Optional [anchorTopology\(\)](#) and [compareGeometry\(\)](#) use qualified spatial evidence; [replicateAgreement\(\)](#) compares runs through molecule identity.

[fromMoleculeExperiment\(\)](#) and [geometryFromSpatialFeatureExperiment\(\)](#) accept explicit IDs and mappings from optional Bioconductor containers. Validation functions report missing evidence without inferring provenance or repairing inputs. Results are descriptive and do not rank segmentation accuracy or biological truth.

Author(s)

Maintainer: Rama Krishna Pudota <ramakrishnapudota36@gmail.com> ([ORCID](#))

Authors:

- Rama Krishna Pudota <ramakrishnapudota36@gmail.com> ([ORCID](#))

See Also

[segmentationExample](#)

Examples

```
data("segmentationExample", package = "SegStability")
names(segmentationExample)
head(segmentationExample$assignments$first, 3)
```

anchorTopology

Describe secondary reciprocal-anchor topology

Description

Select unique reciprocal dominant edges of a complete molecule correspondence graph whose end-points have eligible geometry strictly inside a rectangular observation window. Classes describe dominance relationships, not biological truth, segmentation accuracy or method superiority.

Usage

```
anchorTopology(comparison, geometry, window, coordinate_info = NULL)
```

Arguments

comparison	A SegmentationComparison with complete cell censuses.
geometry	A named list with exactly both segmentation names. Each entry is an sf object with explicit typed cell_id, or NULL for unavailable geometry.
window	Required finite unclassed numeric rectangle with exactly one each of xmin, xmax, ymin, ymax; xmin < xmax and ymin < ymax.
coordinate_info	Optional common coordinate declaration SimpleList. NULL uses comparison declarations. Known compatible units, origin, axis_directions, transforms, registration and specimen_id are required. An empty structured transforms value explicitly declares no transforms. Local planar coordinates may have undeclared CRS; known geometry CRS must match a common declaration. Registration/source evidence remains unchecked.

Details

Geometry validation is reused without repair. Missing geometry is unavailable; invalid, empty, nonfinite or nonpositive-area geometry is ineligible; UNKNOWN backend prerequisites remain unchecked. These cells and all edges are retained. Geometry properties not checked remain NA, and only proven eligible interior cells contribute to anchors or fans. Unsupported types, malformed or unknown IDs and known declaration conflicts stop analysis.

Strict interior compares the complete polygon bbox to all four bounds using strict inequalities. Boundary-touching, crossing and outside polygons are non-interior. Holes and all multipolygon parts are retained. Interior status does not certify independence from unavailable exterior molecule context.

Fan-out counts eligible interior side-2 cells whose unique complete-graph dominant side-1 partner is the anchor's side-1 cell. Fan-in reverses these roles. Both endpoints of each contributing relation must be interior, but contributing relations need not be reciprocal. Classes are exactly one_to_one_like (1,1), split_like (>1,1), merge_like (1,>1), complex (>1,>1). Non-anchor topology is undefined. Selection strata distinguish isolates, ties, nonreciprocity, reciprocal unavailable/ineligible/unchecked geometry, reciprocal non-interior pairs and interior anchors. No adjacency or distance predicate, support filter or state input changes the complete primary graph.

Value

A SimpleList with anchors (DataFrame), cells (named SimpleList of complete cell-summary DataFrames), edges (complete Hits with augmented metadata), validation (comparison/coordinates/named geometry reports) and window (the validated input). Anchors contain edge_index, typed cell_id_1 and cell_id_2, integer fan_out and fan_in, and character anchor_topology. Cells add geometry_present, geometry_status, geometry_reason, geometry_eligible, strict_interior, selection_stratum, interior_anchor, anchor_topology and side-1 fan_out or side-2 fan_in. Edges add geometry_eligible, interior_edge, interior_anchor, fan_out, fan_in and anchor_topology. Original census and canonical edge order are preserved.

Examples

```
if (requireNamespace("sf", quietly = TRUE)) {
  m <- S4Vectors::DataFrame(molecule_id = "m", feature_id = "g")
  p <- SegmentationPartition(m, "first",
    S4Vectors::DataFrame(cell_id = "x"), "m", "assigned", "x")
  q <- SegmentationPartition(m, "second",
    S4Vectors::DataFrame(cell_id = "a"), "m", "assigned", "a")
  ss <- SegmentationSet(m, list(first = p, second = q), "specimen")
  cmp <- compareSegmentations(ss, "first", "second")
  ring <- rbind(c(1, 1), c(2, 1), c(2, 2), c(1, 2), c(1, 1))
  shape <- sf::st_sfc(sf::st_polygon(list(ring)))
  geometry <- list(first = sf::st_sf(cell_id = "x", geometry = shape),
    second = sf::st_sf(cell_id = "a", geometry = shape))
  info <- S4Vectors::SimpleList(units = "micrometres", origin = c(0, 0),
    axis_directions = c("right", "up"),
    transforms = S4Vectors::SimpleList(), registration = "common XY",
    specimen_id = "specimen")
  anchorTopology(cmp, geometry,
    c(xmin = 0, xmax = 10, ymin = 0, ymax = 10), info)$anchors
```

```
}

```

compareGeometry

Describe polygon agreement and derived maximum-IoU matching

Description

Compare supplied polygons across the full molecule correspondence and all positive-area geometric pairs. Geometry matching is a descriptive annotation; it does not replace molecule correspondence or rank segmentation accuracy.

Usage

```
compareGeometry(comparison, geometry, coordinate_info = NULL)
```

Arguments

comparison	A SegmentationComparison with complete cell censuses.
geometry	A base named list with exactly both segmentation names, each containing an sf object with explicit typed cell_id, or NULL.
coordinate_info	Common coordinate declaration SimpleList. NULL uses comparison declarations. Known compatible units, origin, axis_directions, transforms, registration and specimen_id are required. Local planar CRS may be undeclared; known CRS must match a common declaration. Geographic coordinates are ineligible for these planar measurements.

Details

area_1 and area_2 are polygon areas; intersection_area is their intersection; union_area is their area union. Areas use declared units squared. iou (also Jaccard) is intersection/union; dice is twice intersection/summed areas; overlap_fraction_1/2 divide intersection by the respective endpoint area. Ratios are dimensionless. Eligible disjoint molecule edges have zero overlap. Nonpositive areas never enter metrics. Missing, invalid, empty, nonfinite and unchecked geometry remains explicit; unresolved pair metrics are NA.

Pairs retain every molecule edge and all eligible positive-area geometry pairs, including those without shared molecules. Failed measurements remain explicit. No window or interior filter applies. Split, merge and complex correspondence stays pairwise; holes and all multipolygon parts are retained.

Matching maximizes total unmodified IoU on eligible positive geometry pairs. Error-free sums distinguish exact ties from near ties in supplied doubles. Equal optimal objectives select the lexicographically first sorted list of census-index pairs. No epsilon, weight perturbation or zero-weight match is used. Missing/ineligible geometry gives matching_status eligible_subset; unchecked operations give unchecked with no certified matching. Empty known graphs have objective zero. Source registration/authenticity stays UNKNOWN.

Summary reports coverage_complete, matching_status, n_molecule_edges, n_positive_geometry_pairs, n_geometry_only_pairs, n_matches, matching_objective, n_geometry_only_matches, n_molecule_edges_omitted and named per-side cells status counts. The displayed objective is rounded to a double; optimization and equality decisions use error-free sums internally. Enumeration failure stops rather than claiming all positive pairs were found. No repair, buffering, clipping, simplification, transformation or inferred geometry/registration is performed. Optional sf is required only here.

Value

A SimpleList with pairs and matching DataFrames, named full-census cells SimpleList, original edges Hits with derived annotations, unchanged components, summary SimpleList, validation reports and coordinate_info. Pairs contain pair_index, cell_index_1/2, typed cell_id_1/2, original component_id_1/2, nullable edge_index, molecule_edge, positive_overlap, geometry_matched, geometry_status/reason and the eight metric fields below. Matching contains the selected pair rows. Cells add geometry_present, geometry_status/reason, geometry_eligible, area and geometry_matched. Edges add pair_index, geometry_status/reason, overlap/matching flags and metrics, preserving every original field and graph relation.

Examples

```
if (requireNamespace("sf", quietly = TRUE)) {
  m <- S4Vectors::DataFrame(molecule_id = "m", feature_id = "g")
  p <- SegmentationPartition(m, "first",
    S4Vectors::DataFrame(cell_id = "x"), "m", "assigned", "x")
  q <- SegmentationPartition(m, "second",
    S4Vectors::DataFrame(cell_id = "a"), "m", "assigned", "a")
  cmp <- compareSegmentations(
    SegmentationSet(m, list(first = p, second = q), "specimen"),
    "first", "second")
  ring <- rbind(c(0, 0), c(1, 0), c(1, 1), c(0, 1), c(0, 0))
  shape <- sf::st_sfc(sf::st_polygon(list(ring)))
  g <- list(first = sf::st_sf(cell_id = "x", geometry = shape),
    second = sf::st_sf(cell_id = "a", geometry = shape))
  info <- S4Vectors::SimpleList(units = "micrometres", origin = c(0, 0),
    axis_directions = c("right", "up"),
    transforms = S4Vectors::SimpleList(), registration = "common XY",
    specimen_id = "specimen")
  compareGeometry(cmp, g, info)$pairs
}
```

compareSegmentations	<i>Compare two segmentations of one immutable molecule universe</i>
----------------------	---

Description

Construct the complete positive shared-molecule correspondence graph and descriptive structural summaries before any selection.

Usage

```
compareSegmentations(object, segmentation1, segmentation2)
```

Arguments

object A valid SegmentationSet owning the shared molecule universe.

segmentation1, segmentation2
Distinct existing segmentation names, each one nonmissing, nonempty character string. Order determines query and subject sides and split/merge direction.

Details

Both source assignments must retain valid shared-universe bindings and assignment fingerprints. The universe digest is checked once and both selected assignment digests are checked without repeating source-ID matching. Only uniquely assigned molecules on both sides contribute to edges; each supplies one unit to one pair before aggregation. Each own-side denominator counts all its uniquely assigned molecules, including those unresolved on the opposite side. Full censuses and isolates are retained.

Empty censuses/universes and empty positive graphs are supported. Counts must be exact whole values at most 2^{53} ; scientific identifiers are never coerced to ordinary double. No cell-by-cell dense matrix is constructed. Combined census sizes beyond the integer positional-index range fail. No support, state, geometry or anchor selection is performed.

Value

A validated SegmentationComparison. See [SegmentationComparison](#) for graph columns, side summaries, status accounting and component classes.

See Also

[correspondenceEdges\(\)](#), [cellSummary\(\)](#), [componentSummary\(\)](#), [assignmentSummary\(\)](#)

Examples

```
m <- S4Vectors::DataFrame(molecule_id = "m", feature_id = "g")
p <- SegmentationPartition(m, "first",
  S4Vectors::DataFrame(cell_id = "x"), "m", "assigned", "x")
q <- SegmentationPartition(m, "second",
  S4Vectors::DataFrame(cell_id = "a"), "m", "unassigned", NA_character_)
ss <- SegmentationSet(m, list(first = p, second = q), "specimen")
cmp <- compareSegmentations(ss, "first", "second")
cellSummary(cmp, "first")
assignmentSummary(cmp)
```

fromMoleculeExperiment

Import retained molecules with explicit segmentation evidence

Description

Translate one MoleculeExperiment assay/sample into constructor arguments for descriptive segmentation sensitivity. Original identity must survive upstream processing; coordinates or row order cannot recover lost identity.

Usage

```
fromMoleculeExperiment(
  object,
  assay,
  sample_id,
  name,
  specimen_id,
  cells,
  assignments,
  molecule_id_col = "molecule_id",
  coordinate_cols = c(x = "x_location", y = "y_location"),
  universe = NULL,
  ambiguity = NULL,
  coordinate_info = S4Vectors::SimpleList(),
  run_metadata = S4Vectors::SimpleList(),
  provenance = S4Vectors::SimpleList(),
  metadata = S4Vectors::SimpleList()
)
```

Arguments

object	A MoleculeExperiment with retained original molecule IDs.
assay	Explicit character assay name.
sample_id	Explicit character source sample key; not an inferred specimen identifier.
name	Segmentation name.
specimen_id	Explicit specimen/ROI identity.
cells	Complete data.frame or DataFrame cell census with cell_id, including zero-molecule cells.
assignments	data.frame or DataFrame containing molecule_id, status and cell_id. Status is explicitly assigned, unassigned or ambiguous.
molecule_id_col	Name of the retained original molecule ID column.

coordinate_cols	Named character vector mapping x/y or x/y/z to source columns, or NULL to disable coordinate translation.
universe	Optional canonical molecule universe data.frame/DataFrame. Exact typed membership, feature identity and shared columns must agree.
ambiguity	External-ID ambiguity evidence as documented by [SegmentationPartition()]. NULL retains unknown candidate evidence.
coordinate_info	Coordinate declarations as a SimpleList. Unknown registration stays unknown; coordinates are never transformed.
run_metadata	Explicit run metadata SimpleList.
provenance	Explicit provenance SimpleList; no values are inferred.
metadata	Ancillary metadata SimpleList, retained in partition and set.

Details

Feature identity comes from retained character feature keys. Extraction follows table order; IDs never come from that order. Explicit assignments and a full census are required because upstream boundaries and assigned molecule rows cannot establish completeness or ambiguity. Default upstream essential imports may discard molecule IDs and are then unsupported. All certified ID families are preserved, including optional integer64. Source tables require compatible schemas; extra columns survive. Canonical ID/feature/coordinate columns precede ancillary columns. With a supplied universe, its column order is retained instead. Source object, geometry and assays are not modified or recomputed.

Value

A SimpleList of exactly the six [SegmentationSet()] arguments, containing a single validated partition. Use 'do.call(SegmentationSet, as.list(result))' to construct the set.

Examples

```
if (requireNamespace("MoleculeExperiment", quietly = TRUE)) {
  tab <- data.frame(molecule_id = c("001", "002"),
    x_location = c(1, 2), y_location = c(3, 4))
  me <- MoleculeExperiment::MoleculeExperiment(
    molecules = list(detected = list(roi = list(g = tab))))
  args <- fromMoleculeExperiment(me, "detected", "roi", "native", "roi1",
    cells = data.frame(cell_id = c("cell1", "empty")),
    assignments = data.frame(molecule_id = c("002", "001"),
      status = c("unassigned", "assigned"),
      cell_id = c(NA_character_, "cell1")))
  ss <- do.call(SegmentationSet, as.list(args))
  moleculeData(ss)
}
```

```
geometryFromSpatialFeatureExperiment
```

Extract an explicitly keyed SpatialFeatureExperiment geometry side-car

Description

Preserve existing polygons and CRS for optional spatial analysis. Geometry supplies no evidence of molecule assignment identity or registration.

Usage

```
geometryFromSpatialFeatureExperiment(
  object,
  type,
  sample_id,
  geometry_kind,
  cell_map,
  cells,
  coordinate_info,
  reference = NULL
)
```

Arguments

object	A SpatialFeatureExperiment.
type	Explicit character geometry name.
sample_id	Explicit character source sample key.
geometry_kind	Either "col" or "annot", without automatic fallback.
cell_map	data.frame/DataFrame with unique character geometry_id source keys and loss-less cell_id values. Every selected geometry must be mapped.
cells	Complete census data.frame/DataFrame or SegmentationPartition.
coordinate_info	Explicit coordinate declarations SimpleList; unknown fields remain unknown. No metadata or registration is inferred.
reference	Optional SegmentationSet reference for existing validation.

Details

Source keys must be explicitly retained names, not automatic row numbers. No coordinate repair, CRS stripping, transformation, union, clipping or geometry dropping occurs. Census cells lacking geometry remain unresolved. Structural, identity and declaration errors stop extraction; per-cell eligibility failures are retained in the validation report. Only geometry interoperability is supported; this helper constructs no molecule universe. Both SpatialFeatureExperiment and sf are optional.

Value

SimpleList(geometry, coordinate_info, validation). Geometry is sf with exact cell_id; validation is the existing [validateGeometry()] report.

Examples

```
if (requireNamespace("SpatialFeatureExperiment", quietly = TRUE) &&
    requireNamespace("sf", quietly = TRUE)) {
  ring <- rbind(c(0, 0), c(1, 0), c(1, 1), c(0, 1), c(0, 0))
  g <- sf::st_sf(geometry = sf::st_sfc(sf::st_polygon(list(ring))))
  rownames(g) <- "source-cell"
  sfe <- SpatialFeatureExperiment::SpatialFeatureExperiment(
    assays = list(counts = matrix(0L, 1L, 1L,
      dimnames = list("g", "source-cell"))),
    colData = S4Vectors::DataFrame(sample_id = "roi"),
    spatialCoords = matrix(c(0.5, 0.5), 1L,
      dimnames = list(NULL, c("x", "y"))),
    sample_id = "roi", unit = "micron")
  SpatialFeatureExperiment::colGeometry(sfe, "cellSeg", sample_id = "roi",
    translate = FALSE) <- g
  sidecar <- geometryFromSpatialFeatureExperiment(sfe, "cellSeg", "roi",
    "col", data.frame(geometry_id = "source-cell", cell_id = "cell1"),
    data.frame(cell_id = c("cell1", "empty")),
    S4Vectors::SimpleList(units = "micron", specimen_id = "roi1"))
  sidecar$validation
}
```

 ovrlpy_validation_summary

Public summary of the qualified ovrlpy/Xenium application

Description

The small CSV at inst/extdata/validation/ovrlpy_validation_summary.csv records source identifiers, exact file hashes, the source-defined molecule universe and ROI, and checked vendor/Baysor correspondence and directional support counts. Its key, value, and evidence columns distinguish facts checked in the public-source replay from retained qualification evidence.

Format

A CSV with one record per key and columns key, value, and evidence. Read values as text to preserve identifiers and hashes.

Details

The evaluated region is tile (9,9) of replicate 1 of Xenium_V1_FF_Mouse_Brain_MultiSection_1. Original source molecule IDs and a complete, independent ROI cell census were preserved on each side. Native cells can cross the ROI boundary; their reported denominators here count their own

assigned ROI molecules. Vendor and saved Baysor v0.7.1 outputs differed in upstream preparation. The record describes correspondence and support, not an isolated boundary effect, accuracy ranking, or biological ground truth.

The optional development-time replay uses `inst/scripts/prepare_ovrlpy_validation.py` followed by `inst/scripts/reproduce_ovrlpy_validation.R`. They require explicitly acquired, hash-checked public source files and run only when invoked by a user. No source molecules, large saved segmentations, or benchmark work files are bundled. The source dataset and Zenodo deposit are credited to their respective creators under their declared CC BY 4.0 terms.

Value

This documentation topic returns no object. Read the installed CSV with `system.file()` and `read.csv()`, as shown in the example.

Source

Tiesmeyer et al., *Identifying 3D signal overlaps in spatial transcriptomics data with ovrlpy*, Nature Biotechnology (2026), doi:10.1038/s41587-026-03004-8. Saved Baysor data: doi:10.5281/zenodo.16736589 (record version 2). Original Xenium output: Xenium_V1_FF_Mouse_Brain_MultiSection_1, 10x Genomics Xenium 1.0.2.

Examples

```
path <- system.file("extdata", "validation",
  "ovrlpy_validation_summary.csv", package = "SegStability")
record <- read.csv(path, colClasses = "character")
record[record$key %in% c("roi_universe_size", "shared_mass"), ]
```

replicateAgreement	<i>Label-invariant agreement among segmentation runs</i>
--------------------	--

Description

Describe computational repeatability through full-universe assignment-status accounting and conditional adjusted Rand index (ARI). These results do not certify biological truth, segmentation accuracy or independent replication.

Usage

```
replicateAgreement(object, segmentations)
```

Arguments

object	A valid SegmentationSet owning one immutable molecule universe.
segmentations	A base character vector naming at least two distinct existing segmentations. Names must be nonmissing and nonempty. Supplied order determines pair order; there is no implicit selection or reference.

Details

Every unordered selected-run pair is reported, with the earlier supplied run first: (1,2), (1,3), ..., (2,3), ... Construction has aligned assignments through lossless molecule identities. Analysis verifies the universe digest once and selected assignment digests once each, then uses those existing indices. Equal cell IDs in different runs have no matching semantics. Cell labels and census permutations do not change agreement.

Assignment-status agreement counts identical assigned, unassigned or ambiguous statuses over the complete universe. All nine status combinations are retained, including zeros. Ambiguous and unassigned molecules are never forced into cells or combined into a pseudo-cell. The jointly assigned count includes only molecules uniquely assigned in both runs. Both reported fractions use the full molecule universe, with NA for an empty universe.

ARI compares the partition co-membership of jointly assigned molecules, using a sparse shared-molecule contingency and its conditional margins. These margins do not replace any primary graph denominators. If A is the sum of choose(edge mass,2), R and C the corresponding row/column pair sums, and T=choose(jointly assigned count,2), the usual ARI is $(A - R * C / T) / ((R + C) / 2 - R * C / T)$. Pair counts and degeneracy checks remain exact through signed LLint arithmetic at the supported integer-index universe limit. Zero denominator is detected through exact nonnegative factors, before double conversion for the final numeric ratio. ARI is NA_real_ for fewer than two jointly assigned molecules or an exactly zero adjusted denominator, including all-singleton and one-cell equivalent partitions.

All input censuses, assignment evidence, provenance and run metadata remain unchanged. Complete selected run metadata, including unknown or missing values, is retained without simplification. Integrity fingerprints do not establish provenance or authenticity. There is no consensus, averaged score, cell matching, geometry or biological-state integration. Graph, state, support and topology variation should be analyzed separately for each run.

Value

A SimpleList containing exactly two DataFrames:

- pairwise_summary: segmentation_1, segmentation_2, n_molecules, n_status_agree, n_status_disagree, status_agreement_fraction, n_jointly_assigned, jointly_assigned_fraction and ari_jointly_assigned.
- assignment_status_contingency: segmentation_1, segmentation_2, status_1, status_2 and n_molecules. Each pair has nine rows in assigned, unassigned, ambiguous order, with side 1 varying slowest.

Names/statuses are character; molecule counts are exact whole-valued numeric values; fractions and ARI are doubles. Result metadata contains segmentations in supplied order and run_metadata, a named SimpleList retaining each selected partition's complete run metadata.

See Also

[SegmentationSet\(\)](#), [compareSegmentations\(\)](#), [assignmentSummary\(\)](#)

Examples

```
m <- S4Vectors::DataFrame(molecule_id = paste0("m", 1:4),
                          feature_id = rep("g", 4))
```

```

a <- SegmentationPartition(m, "run1",
  S4Vectors::DataFrame(cell_id = c("x", "y", "empty")),
  m$molecule_id, rep("assigned", 4), c("x", "x", "y", "y"),
  run_metadata = S4Vectors::SimpleList(seed = 1L))
b <- SegmentationPartition(m, "run2",
  S4Vectors::DataFrame(cell_id = c("unrelated", "labels")),
  m$molecule_id, rep("assigned", 4),
  c("unrelated", "unrelated", "labels", "labels"),
  run_metadata = S4Vectors::SimpleList(seed = 2L))
ss <- SegmentationSet(m, list(run1 = a, run2 = b), "specimen")
result <- replicateAgreement(ss, c("run1", "run2"))
result$pairwise_summary
result$assignment_status_contingency
S4Vectors::metadata(result)$run_metadata

```

SegmentationComparison-class

Complete descriptive correspondence between two segmentations

Description

Represent a complete positive shared-molecule graph, independent assigned denominators, directional dominance and structural components. Results do not establish biological truth or rank segmentation accuracy.

Usage

```

## S4 method for signature 'SegmentationComparison'
show(object)

## S4 method for signature 'SegmentationComparison'
summary(object, ...)

correspondenceEdges(x)

## S4 method for signature 'SegmentationComparison'
correspondenceEdges(x)

cellSummary(x, segmentation)

## S4 method for signature 'SegmentationComparison'
cellSummary(x, segmentation)

componentSummary(x)

## S4 method for signature 'SegmentationComparison'
componentSummary(x)

```

```

assignmentSummary(x)

## S4 method for signature 'SegmentationComparison'
assignmentSummary(x)

## S4 method for signature 'SegmentationComparison'
segmentationNames(x)

## S4 method for signature 'SegmentationComparison'
provenance(x)

```

Arguments

<code>object</code>	A SegmentationComparison to display or summarize.
<code>...</code>	Additional summary arguments; currently unused.
<code>x</code>	A SegmentationComparison.
<code>segmentation</code>	Exactly one of the two compared segmentation names.

Details

Create results with `compareSegmentations()`; there is no public comparison constructor or replacement API. Query and subject indices in the Hits returned by `correspondenceEdges()` refer to the full side-1 and side-2 census rows. Edges are unique positive pairs sorted by query then subject. External cell IDs remain lossless in the side tables.

Edge metadata columns are `weight`, `fraction_1`, `fraction_2`, `dominant_1`, `dominant_2`, `unique_dominant_1`, `unique_dominant_2` and `reciprocal_dominant`. Directional fractions divide shared weight by the endpoint's complete assigned mass. Dominance maximizes those fractions. Comparing integer weights is exactly equivalent because the denominator is constant across an endpoint's edges, and avoids rounding-induced ties. Every exact tie is retained. Reciprocity requires a unique maximum on both endpoints.

Cell tables contain `cell_id`, `assigned_mass`, `matched_mass`, `unmatched_mass`, `partner_count`, `dominance_count`, `component_id` and `matched_fraction`, in original census order. Matched mass sums incident weights; unmatched mass is assigned minus matched. Zero-molecule and zero-correspondence cells remain explicit. No edges means zero `dominance_count`; one maximum means unique dominance; more than one means a tie. `matched_fraction` is NA for zero assigned mass, and otherwise `matched_mass/assigned_mass`.

Components use the complete positive graph including isolates. The table contains `component_id`, `n_segmentation_1`, `n_segmentation_2` and `component_class`. Cardinalities (1,1), (1,>1), (>1,1), (>1,>1) and isolated (1,0)/(0,1) give `one_to_one`, `split`, `merge`, `many_to_many` and `unmatched`. These are structural summaries, not biological split/merge events or anchor topology. IDs follow first encountered node in side-1 then side-2 census order; census permutations may change component numbering.

Assignment accounting has nine rows with `status_1`, `status_2` and `n_molecules`. Each side uses assigned, unassigned, ambiguous order; side 1 varies slowest. Every molecule contributes once. The assigned/assigned count equals total edge weight and matched mass on either side. Unresolved statuses are not forced onto edges or conflated with unmatched assigned mass.

Counts are exact whole-valued numeric values at most 2^{53} . Comparisons retain internal source bindings, coordinate declarations and nested set/ segmentation provenance with unknown values preserved. They contain no molecule universe, molecule assignment vectors, geometry or states. Structural validity does not recompute fingerprints. Standard R copy and serialization semantics apply, including reference-valued ancillary data.

Value

`correspondenceEdges()` returns Hits. `cellSummary()`, `componentSummary()` and `assignmentSummary()` return DataFrames. `segmentationNames()` returns the two names in comparison order; `provenance()` returns a SimpleList with set and named segmentations entries. `summary()` returns a SimpleList with `segmentation_names`, `universe_size`, `cell_summary_1`, `cell_summary_2`, `components`, `assignment_summary` and `coordinate_info`. The coordinate record is returned unchanged, including unknowns, without validation, completion or spatial prerequisites. `show()` displays structural counts.

See Also

[compareSegmentations\(\)](#), [SegmentationSet\(\)](#)

Examples

```
m <- S4Vectors::DataFrame(molecule_id = c("m1", "m2"),
                          feature_id = c("g", "g"))
p <- SegmentationPartition(m, "left",
  S4Vectors::DataFrame(cell_id = c("x", "empty")),
  m$molecule_id, c("assigned", "assigned"), c("x", "x"))
q <- SegmentationPartition(m, "right",
  S4Vectors::DataFrame(cell_id = c("a", "b")),
  m$molecule_id, c("assigned", "assigned"), c("a", "b"))
cmp <- compareSegmentations(
  SegmentationSet(m, list(left = p, right = q), "specimen"),
  "left", "right")
cmp
correspondenceEdges(cmp)
cellSummary(cmp, "left")
componentSummary(cmp)
assignmentSummary(cmp)
segmentationNames(cmp)
provenance(cmp)
summary(cmp)
```

Description

A 24-molecule example for learning the input contract. The data are entirely synthetic and carry no biological interpretation. Both segmentations use the same molecule IDs, with complete cell censuses that include an empty cell. The first segmentation has one ambiguous molecule with two known candidate cells. The tables also contain assigned and unassigned molecules, a split, a merge, and cells with no correspondence.

Usage

```
data(segmentationExample)
```

Format

A named list with five entries:

molecules A data.frame with molecule_id, feature_id and toy x/y coordinates. Row order defines the shared universe.

cells A named list of full cell-census data.frames, each with cell_id.

assignments A named list of data.frames with molecule_id, status and cell_id. Every molecule appears once on each side; unresolved cell_id values are missing.

ambiguity A named list with first-side evidence and candidates tables keyed by molecule_id; second-side evidence is NULL.

states A named list of categorical cell-state vectors keyed by census cell_id. Missing values mean an unresolved label.

See Also

[SegmentationPartition\(\)](#), [SegmentationSet\(\)](#)

Examples

```
data(segmentationExample)
names(segmentationExample)
table(segmentationExample$assignments$first$status)
```

SegmentationPartition-class

One segmentation bound to an immutable molecule universe

Description

Represent a segmentation with an explicit cell census, exactly aligned molecule assignments, and retained ambiguity evidence. These objects support descriptive sensitivity analysis; they do not score segmentation accuracy.

Usage

```

SegmentationPartition(
  molecules,
  name,
  cells,
  molecule_id,
  status,
  cell_id = NULL,
  ambiguity = NULL,
  run_metadata = S4Vectors::SimpleList(),
  provenance = S4Vectors::SimpleList(),
  metadata = S4Vectors::SimpleList()
)

## S4 method for signature 'SegmentationPartition'
show(object)

cellCensus(x)

## S4 method for signature 'SegmentationPartition'
cellCensus(x)

assignmentStatus(x)

## S4 method for signature 'SegmentationPartition'
assignmentStatus(x)

ambiguityData(x)

## S4 method for signature 'SegmentationPartition'
ambiguityData(x)

provenance(x)

## S4 method for signature 'SegmentationPartition'
provenance(x)

runMetadata(x)

## S4 method for signature 'SegmentationPartition'
runMetadata(x)

```

Arguments

molecules	A data.frame or S4Vectors DataFrame with lossless molecule_id and base character feature_id columns. Row order defines the immutable universe. Additional columns are retained internally without transformation. Internal molecule_index and feature_index columns are reserved.
-----------	---

name	Nonempty, nonmissing character scalar naming the segmentation.
cells	A data.frame or S4Vectors DataFrame with unique lossless cell_id for the full census, including zero-molecule cells.
molecule_id	External assignment molecule IDs, exactly the universe members in any order, using the universe ID family.
status	Character vector parallel to molecule_id, containing only assigned, unassigned or ambiguous.
cell_id	Cell IDs parallel to molecule_id. Assigned entries must name census cells; unresolved entries must be missing in the census ID family. NULL is permitted only when there are no assigned molecules.
ambiguity	NULL for unknown candidate evidence, or a S4Vectors SimpleList containing external-ID-keyed evidence and candidates DataFrames.
run_metadata	S4Vectors SimpleList of run metadata; empty by default.
provenance	S4Vectors SimpleList of provenance. Missing standardized concepts receive explicit unknown placeholders; supplied values are retained.
metadata	S4Vectors SimpleList of ancillary metadata; empty by default.
object	A SegmentationPartition to display.
x	A SegmentationPartition.

Details

Molecule and cell identifiers support base integer and character, and certified exact paths for S4Vectors LLint and optional bit64 integer64. Ordinary double, logical, factor, complex, raw, missing and empty identifiers are rejected. Matching requires the same family and never converts IDs through ordinary double. Original cell IDs and census order are preserved, including cells with no molecules. No cell ID is an implicit sentinel.

For v0.99, feature IDs support base character only. Missing and empty values fail; whitespace, spelling, case and Unicode sequences are preserved without normalization. There is no automatic conversion from other types. Repeated features across molecules are valid; internal dictionary entries are unique in first-occurrence order and map deterministically to integer feature indices.

Construction verifies exact molecule membership and reorders assignments once to the supplied universe order. Assignment-side molecule IDs are then discarded. Internal status codes and positional indices are implementation details; use the accessors. Do not mutate scientific core slots.

Ambiguity input uses external IDs. Its evidence table has molecule_id, candidate_count (nonnegative whole count or NA), and candidate_ids_known (nonmissing logical). Its candidates table has molecule_id and cell_id. Both tables use the same ID families as their universe/census. Candidate pairs must be unique. Known sets are complete and counts must agree; omitted known counts are derived from the supplied set. Count-only evidence has a known count and FALSE known flag; entirely unknown evidence has NA count and FALSE flag. Omitted evidence for ambiguous molecules remains entirely unknown. No candidate identities are fabricated. Counts do not impose an additional biological threshold on unresolved assignments.

Internal version 1 SHA-256 fingerprints bind ordered molecule/feature identity and ordered census/assignment/ambiguity content. Canonical encoding uses framed fields, exact decimal integer keys, ID-family tags and missingness tags. Character text is encoded in UTF-8 without normalization; byte-marked strings retain their bytes. Census and universe order affect the digest; candidate

input order does not. Optional molecule/cell metadata, run metadata and provenance are outside this identity/assignment digest scope. Fingerprints establish neither source authenticity nor provenance. Structural validity checks fingerprint fields without repeating expensive source-ID matching.

Value

SegmentationPartition() returns a validated SegmentationPartition. cellCensus() returns the full cell DataFrame; assignmentStatus() returns semantic character statuses in universe order; ambiguityData() returns a SimpleList with index-based evidence and molecule-to-cell Hits. provenance() and runMetadata() return their respective SimpleLists.

Slots

name Segmentation name.
 cells Explicit lossless cell census DataFrame.
 status Internal integer status enum in universe order.
 cell_index Integer census row for assigned molecules, NA otherwise.
 ambiguity SimpleList containing evidence DataFrame and candidate Hits.
 run_metadata Structured run metadata SimpleList.
 provenance Structured provenance SimpleList, retaining unknowns.
 universe_fingerprint Universe identity digest as character.
 assignment_fingerprint Assignment identity digest as character.
 fingerprint_version Canonical serialization version as character.
 fingerprint_algorithm Digest algorithm as character.
 schema_version Object schema version as character.
 metadata Ancillary metadata SimpleList.

Examples

```
molecules <- S4Vectors::DataFrame(
  molecule_id = c("m1", "m2", "m3"), feature_id = c("g", "g", "h"))
cells <- S4Vectors::DataFrame(cell_id = c("c1", "unused"))
p <- SegmentationPartition(molecules, "native", cells,
  molecule_id = c("m3", "m1", "m2"),
  status = c("ambiguous", "assigned", "unassigned"),
  cell_id = c(NA_character_, "c1", NA_character_))
p
cellCensus(p)
assignmentStatus(p)
ambiguityData(p)
provenance(p)
runMetadata(p)
```

SegmentationSet-class *A named collection sharing one immutable molecule universe*

Description

Own the molecule universe and feature dictionary once and bind independently censused segmentations to it. This is descriptive structural infrastructure, not a segmentation quality score or ranking.

Usage

```
SegmentationSet(  
  molecules,  
  segmentations,  
  specimen_id,  
  coordinate_info = S4Vectors::SimpleList(),  
  provenance = S4Vectors::SimpleList(),  
  metadata = S4Vectors::SimpleList()  
)  
  
## S4 method for signature 'SegmentationSet'  
show(object)  
  
## S4 method for signature 'SegmentationSet'  
summary(object, ...)  
  
## S4 method for signature 'SegmentationSet'  
x[[i, j, ...]]  
  
assignmentData(x, segmentation)  
  
## S4 method for signature 'SegmentationSet'  
assignmentData(x, segmentation)  
  
segmentationNames(x)  
  
## S4 method for signature 'SegmentationSet'  
segmentationNames(x)  
  
segmentationInfo(x)  
  
## S4 method for signature 'SegmentationSet'  
segmentationInfo(x)  
  
moleculeData(x)  
  
## S4 method for signature 'SegmentationSet'  
moleculeData(x)
```

```
featureData(x)

## S4 method for signature 'SegmentationSet'
featureData(x)

## S4 method for signature 'SegmentationSet'
provenance(x)
```

Arguments

molecules	A data.frame or DataFrame with unique lossless molecule_id and character feature_id, in immutable universe order. Other columns are retained.
segmentations	A uniquely named base list or S4Vectors SimpleList of valid SegmentationPartition objects, with matching names and universe binding.
specimen_id	Exactly one nonmissing, nonempty character string.
coordinate_info	S4Vectors SimpleList of coordinate declarations.
provenance	S4Vectors SimpleList of provenance. Omitted standardized concepts receive explicit unknown placeholders, as for partitions.
metadata	S4Vectors SimpleList of ancillary set metadata.
object	A SegmentationSet to display or summarize.
...	Additional arguments to summary (unused), or only exact = TRUE for retrieval.
x	A SegmentationSet.
i	One exact nonempty segmentation name for retrieval.
j	Not supported.
segmentation	One exact existing segmentation name.

Details

Construction preserves molecule order and lossless identity using the certified ID paths documented in [SegmentationPartition\(\)](#). Feature IDs are base character; the dictionary follows first occurrence. Partitions contain aligned assignment indices and binding fingerprints, without copies of molecule IDs or the feature dictionary. Full independent cell censuses, including zero-molecule cells, remain unchanged.

At least one partition is required. Collection names must be unique, nonmissing, nonempty and identical to partition names. Input order is preserved. Empty molecule universes and empty cell censuses are supported.

The universe is prepared and fingerprinted once. Each partition's stored molecule count, universe fingerprint, version and algorithm must match; its assignment fingerprint is checked for integrity. Universe order, ID family and feature identity/mapping are integrity-significant. Version 1 SHA-256 fingerprints do not establish provenance or source authenticity. Ancillary metadata is outside the identity/assignment digest scope.

S4 validity checks structure and binding metadata without recomputing digests. The universe is scientifically immutable; do not modify core slots. There are no public replacement methods.

Ordinary R copy-on-modify applies to returned DataFrames and SimpleLists. Reference-valued ancillary metadata retains its normal R semantics; the package does not deep-copy environments.

Coordinate information records declarations only. Construction does not certify spatial compatibility or transform coordinates. Unknown provenance is retained explicitly. Serialization uses standard R objects; no cross-R binary compatibility guarantee is added.

Value

`SegmentationSet()` returns a validated `SegmentationSet`. `segmentationNames()` returns names in collection order. `segmentationInfo()` returns a `DataFrame` with `name`, `n_cells`, `n_molecules` and a `run_metadata` list column retaining each partition's complete `SimpleList` without simplification. `moleculeData()` returns a `DataFrame` with external `molecule_id`, reconstructed `feature_id` and ancillary columns, without internal indices. `featureData()` returns the feature dictionary `DataFrame`. `provenance()` returns set provenance. `x[[name]]` returns the existing universe-bound partition after integrity checking. Only one exact character name is supported; partial or positional selection and replacement are not supported. `assignmentData(x, segmentation)` returns exactly `molecule_id`, semantic status and native typed `cell_id` in owning-universe order, one row per molecule. Unassigned and ambiguous cell IDs are typed missing; candidate evidence remains in `ambiguityData()`. Integrity is checked on read. `summary()` returns a `SimpleList` with `specimen_id`, `n_molecules`, `n_features`, `n_segmentations`, `segmentations` (the `segmentationInfo()` `DataFrame`), and `coordinate_info` (the exact stored declaration, including unknowns, without validation, completion or spatial prerequisites). `show()` displays structural counts and names. Neither method compares cells. `runMetadata()`, `cellCensus()`, `assignmentStatus()` and `ambiguityData()` remain partition accessors.

Slots

`molecules` Immutable molecule `DataFrame` with internal positional indices.

`features` First-occurrence feature dictionary `DataFrame`.

`segmentations` Named `SimpleList` of universe-bound partitions.

`specimen_id` Specimen identity.

`coordinate_info` Coordinate declarations `SimpleList`.

`provenance` Set provenance `SimpleList`.

`universe_fingerprint` Shared universe identity digest.

`fingerprint_version` Canonical fingerprint serialization version.

`fingerprint_algorithm` Digest algorithm.

`schema_version` Object schema version.

`metadata` Ancillary metadata `SimpleList`.

See Also

[SegmentationPartition\(\)](#)

Examples

```

molecules <- S4Vectors::DataFrame(
  molecule_id = c("m1", "m2"), feature_id = c("g", "h"))
p <- SegmentationPartition(molecules, "native",
  S4Vectors::DataFrame(cell_id = c("c1", "unused")),
  molecule_id = c("m2", "m1"),
  status = c("unassigned", "assigned"),
  cell_id = c(NA_character_, "c1"))
ss <- SegmentationSet(molecules, list(native = p), "specimen1")
ss
segmentationNames(ss)
ss[["native"]]
assignmentData(ss, "native")
segmentationInfo(ss)
moleculeData(ss)
featureData(ss)
provenance(ss)
summary(ss)

```

stateStability

Describe categorical state stability across complete correspondence

Description

Quantify same-state, different-state and unresolved shared-molecule mass without ranking segmentation accuracy or claiming biological truth.

Usage

```
stateStability(comparison, states)
```

Arguments

comparison	A SegmentationComparison from compareSegmentations().
states	A base named list keyed by exactly both compared segmentation names. Each element is a named character, factor or integer categorical vector, or an S4Vectors DataFrame with cell_id and state columns. Each mapping must cover its full census exactly. Additional DataFrame columns are not used. Named vectors require character census IDs; other ID families require a DataFrame with the same lossless cell-ID family.

Details

Missing mapping cells, unknown or duplicate IDs and unsupported state types are errors. A present cell with typed NA state or an NA factor label is unresolved. Factors compare by labels; integer labels compare by exact decimal text internally, so 1L equals "1" but differs from "01". Original values, types and factor levels are retained in public cell and edge tables. No normalization or synonym mapping is performed; an empty character label is an available category.

Let M be all edge weight, R weight with both endpoint labels available, and S resolved same-state weight. Different-state mass is R-S and unresolved label mass is M-R. Missing labels are neither agreements nor disagreements. Conditional stability is S/R; the same-state matched fraction is S/M. Directional total-mass stability is S/N using each complete independently assigned population, including assignments unresolved on the other side. Each cell uses all its incident edges. Zero denominators return NA; a zero numerator with positive denominator returns zero. Equal-cell total means use $N_i > 0$; conditional means use $R_i > 0$, with population counts reported.

State-analysis prerequisites are explicitly checked and must PASS, including structure, complete-graph accounting, dominance, components and state alignment. Unavailable source fingerprint recomputation, external census completeness and authenticity remain UNKNOWN and do not block an otherwise valid state calculation. Returned diagnostics do not attest independently reverified source integrity. Provenance diagnostics are contextual evidence, separate from required state checks. No source data or validation cache is inferred, and no molecule rematching or support/topology selection occurs. Ambiguity is reported from the molecule-status contingency: side-specific counts and the distinct-molecule union, counting both-ambiguous molecules once. Ambiguous assignments never acquire forced correspondence edges.

Value

A SimpleList with elements in this order:

global One-row DataFrame with matched_mass (M), resolved_mass (R), same_state_mass (S), different_state_mass, unresolved_label_mass, conditional_stability, same_state_fraction_of_matched_mass, segmentation_1/2, assigned_mass_1/2, unmatched_mass_1/2, total_mass_stability_1/2, ambiguous_mass_1/2 and ambiguous_mass_union.

directional Two-row DataFrame in comparison order, with segmentation, the unsuffixed masses/ratios above except ambiguity, n_cells, zero_molecule_cells, zero_correspondence_cells, endpoint_unavailable_cells, total_estimable_cells, total_nonestimable_cells, conditional_estimable_cells, conditional_nonestimable_cells, mean_cell_total_mass_stability and mean_cell_conditional_stability.

cells SimpleList keyed by segmentation name, retaining original census tables/order and appending state, resolved_mass, same_state_mass, different_state_mass, unresolved_label_mass, total_mass_stability and conditional_stability.

edges Hits retaining original indices and metadata, appending state_1, state_2, component_id, resolved, same_state (NA if unresolved), resolved_mass, same_state_mass, different_state_mass and unresolved_label_mass.

components DataFrame retaining original component rows/classes and appending pooled M/R/S masses and ratios. Assigned/unmatched mass, total-mass stability and directional cell-count/equal-cell-mean fields are appended with _1/_2 suffixes. No biological component class is inferred.

validation DataFrame(check,status,severity,message). Required checks have state. prefixes and PASS; unavailable evidence remains UNKNOWN.

See Also

compareSegmentations(), cellSummary(), assignmentSummary(), validateProvenance()

Examples

```

m <- S4Vectors::DataFrame(molecule_id = c("m1", "m2"),
                          feature_id = c("g", "g"))
p <- SegmentationPartition(m, "left",
                          S4Vectors::DataFrame(cell_id = "x"), m$molecule_id,
                          c("assigned", "assigned"), c("x", "x"))
q <- SegmentationPartition(m, "right",
                          S4Vectors::DataFrame(cell_id = c("a", "b")), m$molecule_id,
                          c("assigned", "assigned"), c("a", "b"))
cmp <- compareSegmentations(
  SegmentationSet(m, list(left = p, right = q), "specimen"),
  "left", "right")
result <- stateStability(cmp,
  list(left = c(x = "s"), right = c(a = "s", b = NA_character_)))
result$global
result$cells$left
result$validation

```

supportSensitivity	<i>Describe directional support sensitivity of complete correspondence</i>
--------------------	--

Description

Measure support retention and optional categorical agreement using the immutable complete comparison and its original independent denominators.

Usage

```

supportSensitivity(
  comparison,
  thresholds = c(0, 0.01, 0.025, 0.05, 0.1, 0.2),
  direction = NULL,
  states = NULL
)

```

Arguments

comparison	A SegmentationComparison from compareSegmentations().
thresholds	Nonempty plain integer or numeric vector of unique finite fractions in [0,1]. Supplied order is retained.
direction	NULL for both segmentation names in comparison order, or one exact segmentation name.
states	NULL, or a named list following the stateStability() mapping contract. States do not affect structural support selection or accounting.

Details

For each direction select edges with weight/original assigned_mass greater than or equal to the threshold. The directions are separate estimands, not an AND/OR selection. No molecule rematching or stateStability() call occurs. Original graph metadata, dominance/ties, reciprocity, component identities, assignment accounting and cell censuses remain unchanged. Selected support is summarized within original components; no component reclassification, geometry selection or anchor topology is performed. Selected edges are internal views, never a new primary comparison.

Retained mass fraction uses all original matched mass. Retained assigned fraction and total_mass_stability use all originally assigned mass on the selected side. Original unmatched mass excludes only original nonmatches; excluded_matched_mass reports the separate cost of support filtering. Supplied states yield selected resolved mass R, same-state mass S, different-state mass R-S and unresolved-label mass retained_mass-R. Conditional stability is S/R; total-mass stability is S/N_original. Missing labels are unresolved, never disagreements. With states=NULL all state-dependent fields are NA. Zero denominators yield NA; positive denominators with zero numerator yield zero. Equal-cell total means use original assigned_mass>0; conditional means use selected resolved_mass>0.

Required support checks must PASS. Unavailable source-level fingerprint recomputation, external census completeness and authenticity evidence stays UNKNOWN and does not block otherwise valid support analysis. Validation metadata does not certify source authenticity or independently reverified source integrity. These results are descriptive, not accuracy rankings.

Value

A DataFrame, with direction-major rows then supplied threshold order. Scalar fields are direction, threshold, states_supplied, original_edges, retained_edges, removed_edges, original_matched_mass, retained_mass, excluded_matched_mass, retained_mass_fraction, assigned_mass, unmatched_mass, retained_assigned_fraction, n_cells, zero_molecule_cells, zero_correspondence_cells (original), retained_cells and selected_zero_correspondence_cells. State fields are resolved_mass, same_state_mass, different_state_mass, unresolved_label_mass, conditional_stability, same_state_fraction_of_matched_mass, total_mass_stability, endpoint_unavailable_cells, total_estimable_cells, total_nonestimable_cells, conditional_estimable_cells, conditional_nonestimable_cells, mean_cell_total_mass_stability and mean_cell_conditional_stability.

The cells list column holds a SimpleList keyed by both segmentation names, retaining original census fields and appending selected_retained_mass, selected_removed_mass, selected_edge_count, selected_removed_edge_count, selected_retained_mass_fraction, selected_retained_assigned_fraction and selected_state fields above (counts become per-cell indicators; means become per-cell ratios). Supplied state vectors retain their types in state. The components list column retains original component rows/classes, with original_edges, original_matched_mass, pooled selected support/state fields and directional scalar summary fields prefixed selected_ and suffixed _1/_2. Pooled selected counts are selected_edge_count/selected_removed_edge_count; pooled removed mass is selected_removed_mass. Original assigned_mass and unmatched_mass are also included with _1/_2 suffixes. Both nested tables preserve zero-support census members. metadata(result)\$validation holds DataFrame(check,status,severity,message); support_checks are required, source/provenance diagnostics are contextual and can remain UNKNOWN.

See Also

compareSegmentations(), stateStability(), componentSummary()

Examples

```
m <- S4Vectors::DataFrame(molecule_id = c("m1", "m2"),
                          feature_id = c("g", "g"))
p <- SegmentationPartition(m, "left",
                          S4Vectors::DataFrame(cell_id = "x"), m$molecule_id,
                          c("assigned", "assigned"), c("x", "x"))
q <- SegmentationPartition(m, "right",
                          S4Vectors::DataFrame(cell_id = c("a", "b")), m$molecule_id,
                          c("assigned", "assigned"), c("a", "b"))
cmp <- compareSegmentations(
  SegmentationSet(m, list(left = p, right = q), "specimen"),
  "left", "right")
result <- supportSensitivity(cmp, thresholds = c(0, .5, 1))
result[, c("direction", "threshold", "retained_mass")]
result$cells[[1]]$left
result$components[[1]]
S4Vectors::metadata(result)$validation
```

validateCoordinates	<i>Validate molecule coordinates without transformation</i>
---------------------	---

Description

Finite base numeric/double or integer spatial measurements are accepted. The no-double identifier rule applies to molecule/cell IDs, not coordinates. Exact ID alignment may index temporary diagnostics, but never reorders input.

Usage

```
validateCoordinates(
  object,
  coordinate_info = NULL,
  reference = NULL,
  strict = FALSE
)
```

Arguments

object	A SegmentationSet or DataFrame with molecule_id, x, y and optional z columns. NULL denotes absent optional coordinates.
coordinate_info	Optional uniquely named SimpleList declaring units, origin, axis_directions, transforms, registration, specimen_id and crs. Missing declarations remain unknown. An empty structured transforms value explicitly means none. Known conflicting declarations fail; units and declarations are compared exactly, without aliases or inferred transforms.
reference	Optional SegmentationSet for exact molecule coverage, coordinate values and declaration consistency checks.
strict	Logical scalar. Stop only for FAIL/error findings.

Details

Units, specimen_id and crs are character scalars. Origin and axis_directions accept character or base numeric vectors. Transforms/registration accept character or structured records; their contents are not executed or attested. Explicit declarations must agree with known stored/reference declarations. Finite measurements are checked without inferred range limits or imputation. Source identity linkage, registration verification and authenticity remain unchecked even when values and declarations agree.

Value

A DataFrame with character check, status, severity and message columns. Absent coordinates are NOT_APPLICABLE/note. Malformed supplied coordinates are FAIL/error. Matching values/declarations do not attest registration.

Examples

```
d <- S4Vectors::DataFrame(molecule_id = c("m1", "m2"),
  x = c(0.25, 1.5), y = c(2.5, 3.75))
validateCoordinates(d, strict = TRUE)
```

validateGeometry	<i>Validate optional cell geometry without repair</i>
------------------	---

Description

Geometry is external to core objects. Polygons, multipolygons and holes are supported; overlaps are allowed. Cell IDs are matched exactly to the complete census. No geometry is repaired, buffered, clipped, simplified or transformed.

Usage

```
validateGeometry(
  geometry,
  cells,
  coordinate_info = NULL,
  reference = NULL,
  strict = FALSE
)
```

Arguments

geometry	An sf object with explicit cell_id, or NULL when absent.
cells	A DataFrame containing cell_id or a SegmentationPartition census.
coordinate_info	Optional uniquely named coordinate declaration SimpleList with the fields described in validateCoordinates().

reference	Optional SegmentationSet for specimen/declaration consistency.
strict	Logical scalar. Stop ONLY for FAIL/error findings. Invalid, empty, nonpositive-area and nonfinite supplied geometry is FAIL/warning and ineligible; these findings alone never cause strict mode to throw.

Details

Per-cell check labels `geometry.cell[i]` refer to original census row `i`. Geometry rows are aligned by exact IDs, including cells without geometry. These diagnostics do not return altered geometry or certify spatial sources.

Value

A `DataFrame` with character check, status, severity and message columns. Missing census geometry is UNKNOWN/note; absent geometry NOT_APPLICABLE/note. Supplied geometry without optional `sf` is unchecked UNKNOWN/warning. Unknown/duplicate IDs, unsupported types and known conflicts are FAIL/error. Geographic planar area is unchecked. Declaration consistency alone cannot attest source linkage or registration.

Examples

```
validateGeometry(NULL, S4Vectors::DataFrame(cell_id = "c1"), strict = TRUE)
if (requireNamespace("sf", quietly = TRUE)) {
  ring <- rbind(c(0, 0), c(1, 0), c(1, 1), c(0, 1), c(0, 0))
  g <- sf::st_sf(cell_id = "c1",
    geometry = sf::st_sfc(sf::st_polygon(list(ring))))
  validateGeometry(g, S4Vectors::DataFrame(cell_id = "c1"), strict = TRUE)
}
```

validateProvenance	<i>Validate provenance shape and completeness</i>
--------------------	---

Description

Checks standardized fields without completing or changing supplied metadata. Unknown evidence remains unknown; completeness does not attest authenticity.

Usage

```
validateProvenance(object, strict = FALSE)
```

Arguments

object	A SimpleList or a SegmentationPartition, SegmentationSet or Segmentation-Comparison, whose provenance accessor supplies the metadata.
strict	Logical scalar. Stop only if a report contains FAIL/error. UNKNOWN and FAIL/warning findings do not themselves cause an error.

Details

Required keys are source, source_version, retrieval_date, source_hashes, software, software_version, software_hashes, command, parameters, model, threads, runtime, observation_window and output_hashes. Text fields are character scalars; hashes are character vectors or uniquely named nested list/SimpleList records. Parameters/model accept character or structured values. Threads is a positive base integer scalar; runtime is a finite nonnegative base numeric scalar. Typed NA values remain unknown. A window is a finite named numeric xmin/xmax/ymin/ymax bbox with increasing bounds, or an empty SimpleList unknown placeholder. Extra keys are retained. Reports are deterministic and side-effect free. Rows describe checks rather than an aggregate validity or authenticity flag. FAIL/error closes the relevant gate; UNKNOWN indicates unavailable evidence. Strict mode throws only for FAIL/error, never solely for FAIL/warning or UNKNOWN. Backend warnings are captured in diagnostics; a warned check without a failure is UNKNOWN/warning rather than an unsupported PASS.

Value

A DataFrame with character columns check, status, severity and message. Status is PASS, FAIL, UNKNOWN or NOT_APPLICABLE; severity is error, warning or note. Missing required keys and malformed values are FAIL/error; typed unknowns and appropriate empty structured values are UNKNOWN/note.

Examples

```
p <- SegmentationPartition(
  S4Vectors::DataFrame(molecule_id = character(),
    feature_id = character()),
  "empty", S4Vectors::DataFrame(cell_id = character()),
  character(), character())
validateProvenance(p, strict = TRUE)
```

validateSegmentation	<i>Validate a segmentation partition explicitly</i>
----------------------	---

Description

Checks census identity, assignment conservation, ambiguity evidence and assignment integrity independently of provenance completeness. Empty censuses and zero-molecule cells are retained. External census completeness remains unchecked; assignment mass cannot certify missing zero-molecule cells. A partition does not own its universe; standalone universe verification is UNKNOWN. No input is repaired or changed.

Usage

```
validateSegmentation(object, check_integrity = TRUE, strict = FALSE)
```

Arguments

<code>object</code>	A SegmentationPartition.
<code>check_integrity</code>	Logical scalar. Recompute the existing assignment fingerprint by default. FALSE reports integrity as UNKNOWN. Fingerprints bind identity/assignments, not provenance, authenticity or spatial metadata.
<code>strict</code>	Logical scalar. Stop only for FAIL/error findings, after checks.

Value

A DataFrame with character check, status, severity and message columns. Statuses are PASS, FAIL, UNKNOWN and NOT_APPLICABLE; severities are error, warning and note. Unknown candidate evidence is reported without fabrication.

Examples

```
m <- S4Vectors::DataFrame(molecule_id = "m1", feature_id = "g1")
p <- SegmentationPartition(m, "s", S4Vectors::DataFrame(cell_id = "c1"),
  "m1", "assigned", "c1")
validateSegmentation(p, strict = TRUE)
```

validateSegmentationSet

Validate a shared-universe segmentation set explicitly

Description

Checks exact identity, feature mapping, full partition binding and conservation. Recomputes the shared universe once and each assignment fingerprint once when enabled. Coordinates are optional; this function checks declaration structure, without running spatial operations. No comparisons or repairs are performed.

Usage

```
validateSegmentationSet(object, check_integrity = TRUE, strict = FALSE)
```

Arguments

<code>object</code>	A SegmentationSet.
<code>check_integrity</code>	Logical scalar. Recompute existing fingerprints; FALSE reports integrity as UNKNOWN rather than claiming it was checked.
<code>strict</code>	Logical scalar. Stop only for FAIL/error findings.

Value

A `DataFrame` with character check, status, severity and message columns. Nested findings follow partition order. UNKNOWN evidence cannot certify a gate. Integrity fingerprints do not attest source authenticity or provenance.

Examples

```
m <- S4Vectors::DataFrame(molecule_id = "m1", feature_id = "g1")
p <- SegmentationPartition(m, "s", S4Vectors::DataFrame(cell_id = "c1"),
  "m1", "assigned", "c1")
x <- SegmentationSet(m, list(s = p), "specimen")
validateSegmentationSet(x, strict = TRUE)
```

Index

* **datasets**
 segmentationExample, 16
 [[, SegmentationSet-method
 (SegmentationSet-class), 21

 ambiguityData
 (SegmentationPartition-class),
 17
 ambiguityData, SegmentationPartition-method
 (SegmentationPartition-class),
 17
 anchorTopology, 3
 anchorTopology(), 2
 assignmentData (SegmentationSet-class),
 21
 assignmentData(), 2
 assignmentData, SegmentationSet-method
 (SegmentationSet-class), 21
 assignmentStatus
 (SegmentationPartition-class),
 17
 assignmentStatus, SegmentationPartition-method
 (SegmentationPartition-class),
 17
 assignmentSummary
 (SegmentationComparison-class),
 14
 assignmentSummary(), 7, 13
 assignmentSummary, SegmentationComparison-method
 (SegmentationComparison-class),
 14

 cellCensus
 (SegmentationPartition-class),
 17
 cellCensus, SegmentationPartition-method
 (SegmentationPartition-class),
 17
 cellSummary
 (SegmentationComparison-class),
 14
 cellSummary(), 7
 cellSummary, SegmentationComparison-method
 (SegmentationComparison-class),
 14
 compareGeometry, 5
 compareGeometry(), 2
 compareSegmentations, 6
 compareSegmentations(), 2, 13, 15, 16
 componentSummary
 (SegmentationComparison-class),
 14
 componentSummary(), 7
 componentSummary, SegmentationComparison-method
 (SegmentationComparison-class),
 14
 correspondenceEdges
 (SegmentationComparison-class),
 14
 correspondenceEdges(), 7
 correspondenceEdges, SegmentationComparison-method
 (SegmentationComparison-class),
 14

 featureData (SegmentationSet-class), 21
 featureData, SegmentationSet-method
 (SegmentationSet-class), 21
 fromMoleculeExperiment, 8
 fromMoleculeExperiment(), 2
 geometryFromSpatialFeatureExperiment,
 10
 geometryFromSpatialFeatureExperiment(),
 2

 moleculeData (SegmentationSet-class), 21
 moleculeData, SegmentationSet-method
 (SegmentationSet-class), 21

 overlpy_validation_summary, 11

- provenance
 - (SegmentationPartition-class),
17
- provenance, SegmentationComparison-method
 - (SegmentationComparison-class),
14
- provenance, SegmentationPartition-method
 - (SegmentationPartition-class),
17
- provenance, SegmentationSet-method
 - (SegmentationSet-class), 21
- replicateAgreement, 12
- replicateAgreement(), 2
- runMetadata
 - (SegmentationPartition-class),
17
- runMetadata, SegmentationPartition-method
 - (SegmentationPartition-class),
17
- SegmentationComparison, 7
- SegmentationComparison-class, 14
- segmentationExample, 3, 16
- segmentationInfo
 - (SegmentationSet-class), 21
- segmentationInfo, SegmentationSet-method
 - (SegmentationSet-class), 21
- segmentationNames
 - (SegmentationSet-class), 21
- segmentationNames, SegmentationComparison-method
 - (SegmentationComparison-class),
14
- segmentationNames, SegmentationSet-method
 - (SegmentationSet-class), 21
- SegmentationPartition
 - (SegmentationPartition-class),
17
- SegmentationPartition(), 2, 17, 22, 23
- SegmentationPartition-class, 17
- SegmentationSet
 - (SegmentationSet-class), 21
- SegmentationSet(), 2, 13, 16, 17
- SegmentationSet-class, 21
- SegStability (SegStability-package), 2
- SegStability-package, 2
- show, SegmentationComparison-method
 - (SegmentationComparison-class),
14
- show, SegmentationPartition-method
 - (SegmentationPartition-class),
17
- show, SegmentationSet-method
 - (SegmentationSet-class), 21
- stateStability, 24
- stateStability(), 2
- summary, SegmentationComparison-method
 - (SegmentationComparison-class),
14
- summary, SegmentationSet-method
 - (SegmentationSet-class), 21
- supportSensitivity, 26
- supportSensitivity(), 2
- validateCoordinates, 28
- validateGeometry, 29
- validateProvenance, 30
- validateSegmentation, 31
- validateSegmentationSet, 32