

Package: epiPortrait (via r-universe)

October 4, 2026

Type Package

Title Replicate-Aware Epigenomic Domain Profiling

Version 0.99.3

Depends R (>= 4.5.0)

Description Provides methods for replicate-aware quantitative phenotyping and comparison of epigenomic domains from normalized continuous signal tracks and per-sample native peak/domain calls (histone-mark ChIP-seq / CUT&Tag as the primary use case; also applicable to ATAC-seq-derived domains). It characterizes each shared genomic domain by integrated signal magnitude (Intensity), sample-specific native peak breadth geometry, and continuous within-domain signal architecture (SignalDispersion). Replicate-level evidence is retained and aggregated with specified support rules, enabling condition-specific classification of Intensity-Super, Breadth-Super, Dual-Super, Typical, or Uncertain domain states without using the consensus-domain width as the breadth caller. The package further supports condition transitions, continuous domain-width remodeling, signal quality control, genome-aware domain annotation with a three-level export (per-domain summary, per-domain-gene pair, and raw relationship detail), optional BEDPE contact evidence, expression-aware candidate-gene prioritization, differential domain analysis on continuous BigWig signal using limma, and result visualization.

License GPL (>= 3) + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/testthat/edition 3

biocViews ChIPSeq, ATACSeq, Epigenetics, Coverage, Normalization, DifferentialExpression, GenomeAnnotation, QualityControl, Visualization, PrincipalComponent

URL <https://github.com/zying106/epiPortrait>

BugReports <https://github.com/zying106/epiPortrait/issues>

Imports digest, GenomeInfoDb, GenomicRanges, rtracklayer, SummarizedExperiment, BiocParallel, S4Vectors, IRanges, GenomicFeatures, ggplot2, scales, ggrepel, limma, methods, stats, AnnotationDbi, withr

Suggests BiocStyle, edgeR, HDF5Array, knitr, org.Hs.eg.db, org.Mm.eg.db, ragg, rmarkdown, svglite, testthat, txdbmaker, TxDb.Hsapiens.UCSC.hg19.knownGene, TxDb.Hsapiens.UCSC.hg38.knownGene, TxDb.Mmusculus.UCSC.mm10.knownGene

VignetteBuilder knitr

Config/pak/sysreqs

make libbz2-dev liblzma-dev libpng-dev libxml2-dev libssl-dev xz-utils zlib1g-dev

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

Date/Publication 2026-09-04 00:02:14 UTC

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

RemoteRef HEAD

RemoteSha 6dd14984eba9dfb38708b30501b2edd85b6dcbf9

Contents

analyze_differential_domains	3
annotate_epi_domains	5
build_portrait_matrix	8
call_super_domains	10
check_signal_compatibility	13
combine_superdomain_calls	14
compare_superdomain_classes	15
compare_superdomains	16
compute_width_transition	17
epiPortrait	19
example_se	20
export_epiportrait_results	21
filter_blacklist	22
filter_promoter_peaks	22
find_hockey_inflection	23
get_call_provenance	24
get_call_results	25
get_combined_class_results	26
get_consensus_peaks	26
get_domain_genes	27
get_domain_results	29
get_mark_preset	29
get_replicate_calls	30
get_sample_results	31

get_transition_results	32
get_uncertain_cause	32
integrate_cross_mark	33
normalize_portrait	35
plot_class_transition	36
plot_differential_volcano	37
plot_domain_class_composition	38
plot_domain_feature_profile	39
plot_domain_landscape	40
plot_hockey_stick	41
plot_peak_track	43
plot_portrait_correlation	44
plot_portrait_pca	45
plot_replicate_support	46
plot_uncertain_cause	47
save_epiportrait_figure	48
stitch_epi_peaks	48
summarize_epiportrait_object	49
transition_matrix_per_sample	50
validate_epiportrait_object	51

Index 52

analyze_differential_domains

Differential Domain Analysis on Continuous Signal (limma)

Description

Performs a group-wise differential analysis of a continuous domain assay (default Intensity, the integrated BigWig signal within each candidate domain) using limma's empirical-Bayes framework. This complements the Super-domain state calls: where `compare_superdomains()` reports relative *state* transitions between discrete phenotypes, `analyze_differential_domains()` reports a statistical *P-value* for a continuous signal change between condition groups.

Usage

```
analyze_differential_domains(
  se,
  feature = "Intensity",
  group_var = "Condition",
  ref_group = NULL,
  target_group = NULL,
  design = NULL,
  contrast = NULL,
  transform = c("log2"),
  min_signal = 0,
  logFC_cutoff = 1,
```

```

    fdr_cutoff = 0.05,
    trend = TRUE,
    robust = FALSE
  )

```

Arguments

se	A SummarizedExperiment from build_portrait_matrix().
feature	Character. Continuous assay to test (default "Intensity").
group_var	Character. colData column holding the groups (default "Condition"). Only used when design is NULL.
ref_group, target_group	Character. Reference and target group labels for the convenience two-group contrast (used when design is NULL).
design	A model formula (e.g. ~ Condition) defining the linear model, or NULL to build from group_var + ref_group/ target_group. If supplied, contrast must also be supplied.
contrast	A contrast specification: NULL (uses ref/target), a length-2 character vector of column names to subtract (coef1 - coef2), or a numeric matrix from limma::makeContrasts.
transform	Character. "log2" (only; v1.0). Transform applied to the (non-negative) signal before limma.
min_signal	Numeric. Filter: domains whose mean transformed signal is below this are dropped from the fit (NA results), to avoid fitting noise. Default 0 (no filtering).
logFC_cutoff	Numeric. llog2-fold-change threshold (on log2 scale) used only for the descriptive DiffStatus label (default 1).
fdr_cutoff	Numeric. FDR threshold for the DiffStatus label (default 0.05).
trend	Logical. Passed to limma::eBayes: fit a mean-variance trend across domains before moderation. Integrated intensity exhibits a strong mean-variance dependence, so the default is TRUE; it is disabled automatically (with a warning) when fewer than 20 domains pass filtering.
robust	Logical. Passed to limma::eBayes: robust empirical-Bayes moderation, resistant to outlier domains (default FALSE). Consider TRUE when a small subset of extreme domains is expected.

Details

Quantitative unit. epiPortrait extracts *continuous* signal magnitude (e.g. Intensity) from normalized BigWig tracks rather than integer read counts. limma is applied to log-transformed continuous intensity; unlike DESeq2 or edgeR, this is not a negative-binomial count model. Valid inference therefore requires quantitatively comparable normalized tracks and an appropriate experimental design.

Explicit caveat (recorded in provenance). The test is on the integrated BigWig signal, *not* an exact read-count model. If the user requires DESeq2/DiffBind-style count statistics, the recommended path is to obtain per-domain counts externally and run those tools; epiPortrait provides the coordinate universe and continuous phenotype layer.

Design and contrasts. design defines the linear model on the samples (e.g. $\sim \text{Condition}$ or $\sim 0 + \text{Condition} + \text{batch}$); contrast is set either via the convenience ref_group/target_group pair (default two-group comparison on a Condition column) or via a contrast matrix / limma::makeContrasts-style specification.

Transform offset caveat. The variance-stabilizing transform is $\log(x + 1)$; the pseudo-count 1 is expressed in the units of the input BigWigs, so its relative weight depends on the upstream normalization scheme (CPM vs RPGC vs spike-in). With CPM-scale tracks the offset is negligible for enriched domains but dominates near-background ones; set min_signal accordingly to exclude domains where the fit would be driven by the offset.

Mean-variance trend. Integrated intensity shows a strong mean-variance dependence across domains; by default a limma mean-variance trend is fitted (trend = TRUE), which prevents low-signal domains from being over-called as significant. The automatic fallback (fewer than 20 tested domains) and robust = TRUE outlier-resistant moderation follow limma's recommendations for non-count genomic data.

Value

se with rowData columns: <feature>_logFC, <feature>_AveExpr, <feature>_t, <feature>_P.Value, <feature>_adj.P.Val, <feature>_DiffStatus (Gain / Loss / NS / NA); and metadata(se)\$differential_domains with the limma fit summary, design, contrast and provenance.

Examples

```
data(example_se)
if (requireNamespace("limma", quietly = TRUE)) {
  se <- analyze_differential_domains(
    example_se, group_var = "Condition",
    ref_group = "Control", target_group = "Treatment")
  table(SummarizedExperiment::rowData(se)$Intensity_DiffStatus)
}
```

annotate_epi_domains *Domain-Aware Annotation of epiPortrait Domains*

Description

Annotates the final shared domains (not individual peaks) with genome-aware linear evidence plus optional BEDPE 3D-contact and RNA-seq expression evidence. Adds 7 compact rowData columns and stores a long-format domain-gene link table in metadata(se)\$domain_gene_links.

Usage

```
annotate_epi_domains(
  se,
  genome = "hg38",
  txdb = NULL,
  anno_db = NULL,
```

```

promoter_upstream = 3000,
promoter_downstream = 3000,
bedpe = NULL,
bedpe_score_col = NULL,
min_anchor_overlap_bp = NULL,
expression = NULL,
expression_type = NULL,
aggregate_fun = "median",
gene_id_keytype = NULL
)

```

Arguments

se	A SummarizedExperiment from build_portrait_matrix().
genome	Character. Built-in shortcut ("hg38", "hg19", "mm10", "mm9") or a TxDb object. For non-built-in species supply txdb.
txdb	A TxDb object or NULL (resolved from genome shortcut).
anno_db	Character. OrgDb package name, or NULL (resolved).
promoter_upstream	Numeric. Promoter window upstream of the TSS (default 3000 bp).
promoter_downstream	Numeric. Promoter window downstream of the TSS (default 3000 bp).
bedpe	Character path to a BEDPE file, a data.frame, or NULL. Only bedpe_promoter_contact evidence is added (3D promoter-contact); records are read-only (no calling / filtering / merging).
bedpe_score_col	Numeric or character or NULL. Optional column holding a per-record contact STRENGTH (e.g. the FitHiChIP TMM-normalized contact frequency in column 8). Integer column index (≥ 7 ; columns 1-6 are the coordinates) or a column name (file inputs expose V1..Vn names; a data.frame input can carry arbitrary names). When given, quantitative outputs are produced IN ADDITION to the count-based evidence: metadata(se)\$domain_gene_links\$contact_score (per evidence row), and four per-record aggregates (sum / max / mean / n) over the UNIQUE supporting records, exposed as bedpe_contact_score(_max/_mean/_n) in both the dedup pair table and the annotation summary (and mirrored on row-Data). Rationale: FitHiChIP-style TMM BEDPEs often share an identical loop SET across conditions and vary only in score, so count-based integration alone cannot capture contact dynamics; sum reflects total contact burden, max a single dominant interaction, mean per-record intensity, and n record count. NULL (default) keeps the previous count-only behaviour unchanged.
min_anchor_overlap_bp	Numeric or NULL. Minimum base pairs of overlap required between a domain anchor and a BEDPE anchor for the 3D contact to be counted (default NULL = any overlap, i.e. ≥ 1 bp). A value of 1 requires at least 1 bp, 2 at least 2 bp, 100 at least 100 bp of anchor overlap; typical meaningful thresholds are tens to hundreds of base pairs depending on anchor resolution. Does NOT reject 1-bp overlaps (use ≥ 2 for that). Applies to BOTH anchor sides (domain-anchor and promoter-anchor) of the domain-gene contact.

expression	A gene-level expression object: a wide matrix (genes x samples), a long data.frame (gene_id, SampleID, Condition, expression), or NULL.
expression_type	Character. "TPM", "CPM", "normalized_counts", "VST", "rlog". Must be declared; never auto-guessed.
aggregate_fun	Character. Per-condition aggregation over RNA replicates (default "median").
gene_id_keytype	Character or NULL. OrgDb keytype for the gene IDs (default "ENTREZID" for built-in UCSC genomes; required for custom TxDb's using other ID systems such as ENSEMBL / TAIR / FlyBase).

Details

Representative gene-level TSS. epiPortrait uses one strand-aware 5' boundary per gene derived from `GenomicFeatures::genes(txdb)` (start for +, end for -) as the representative TSS for compact domain annotation. It does not enumerate every alternative-transcript TSS of a gene; the compact `nearest_tss_distance_bp` is therefore a gene-level proximity measure.

Promoter models. `filter_promoter_peaks()` uses `GenomicFeatures::promoters(txdb)` (transcript-level promoter set), while `annotate_epi_domains()` uses a gene-level representative-TSS promoter window. The two are not the same promoter universe; in an enhancer workflow, promoter exclusion and `promoter_overlap_gene_count` should not be assumed to use identical definitions.

Value

se with rowData columns: `primary_genomic_context`, `nearest_tss_gene_id`, `nearest_tss_gene_symbol`, `nearest_tss_distance_bp`, `promoter_overlap_gene_count`, `gene_body_overlap_gene_count`, `fully_contained_gene_count`, `n_bedpe_contact_gene`, `bedpe_contact_score`, `bedpe_contact_score_max`, `bedpe_contact_score_mean`, `bedpe_contact_score_n` (BEDPE score-summary columns are always present: values are NA for supported contacts without a usable score, and 0 for domains with no BEDPE evidence, so downstream schemas are stable), and `top_candidate_gene_symbol`; the annotation result is also exposed as a three-level structure in metadata:

- `metadata(se)$annotation_summary` — ONE ROW PER DOMAIN master table (nearest gene, per-evidence overlap/gene counts, number of BEDPE-contacted genes, and a representative top candidate gene).
- `metadata(se)$domain_gene_links_dedup` — ONE ROW PER domain-gene pair, with summarised `relation_types`, `evidence_sources` and a `best_relation` label.
- `metadata(se)$domain_gene_links` — raw per-relationship detail (for auditing).

`metadata(se)$annotation_provenance`, and (if provided) `metadata(se)$bedpe_provenance` / `metadata(se)$expression_provenance`.

Examples

```
data(example_se)
if (requireNamespace("TxDb.Hsapiens.UCSC.hg38.knownGene", quietly = TRUE)) {
  se <- annotate_epi_domains(example_se, genome = "hg38")
  head(SummarizedExperiment::rowData(se)$primary_genomic_context)
}
```

build_portrait_matrix *Extract and Build Multi-dimensional Portrait Matrix*

Description

Extracts domain-level quantitative features from BigWig files across consensus or stitched peaks: Intensity (integrated signal magnitude) and SignalDispersion (signal-weighted genomic dispersion). When sample-specific native peak files (peak_path) are provided, additional sample-specific breadth geometry is added as NativeMaxPeakWidth, NativeOccupiedWidth and NativePeakCount. This is the core engine of epiPortrait.

Usage

```
build_portrait_matrix(
  sample_sheet,
  consensus_peaks,
  workers = 1,
  on_disk = FALSE,
  h5_dir = "epiPortrait_h5",
  cache_dir = NULL,
  custom_features = NULL,
  negative_policy = c("error", "clip_zero", "allow"),
  fail_action = c("stop", "drop")
)
```

Arguments

sample_sheet	A data.frame containing at least 'SampleID', 'Condition', and 'bw_path'. Optional peak_path: path to the sample-specific native peak call file (BED/narrowPeak/broadPeak readable by rtracklayer::import); when absent, native breadth geometry is not computed and Breadth-Super calling is unavailable for that sample. Recommended additional metadata columns, all passed through to colData(se): 'Replicate', 'Mark', 'Genome', 'Normalization', 'SignalType', 'BinSize', 'SpikeIn'. These enable provenance tracking and condition-aware replicate handling.
consensus_peaks	A GRanges object of unified/stitched peaks (the shared candidate domain universe).
workers	Integer. Number of threads for parallel processing (default: 1).
on_disk	Logical. If TRUE, stores assays as HDF5 files on disk to save RAM, which is highly recommended for mammalian-scale BigWig processing.
h5_dir	Character. Directory to store HDF5 files if on_disk = TRUE.
cache_dir	Character or NULL. If provided, caches per-sample RAW coverage views as RDS files in this directory. Subsequent calls with the same BigWig + consensus peaks combination skip BigWig I/O entirely (default: NULL). The cache stores raw (un-sanitized) coverage so that negative_policy and custom features can change between runs without cache pollution.

custom_features

A named list of functions. Each function must accept a single `NumericList` (the coverage view for one peak) and return a single numeric value. Results are added as additional assays alongside the core dimensions (e.g., `list(Entropy = function(x) ...)`).

negative_policy

Character. How to handle negative signal values in the BigWig tracks:

- "error" (default): fail if any negative signal is found, forcing the user to provide non-negative (e.g. clipped, CPM/RPGC/spike-in) tracks.
- "clip_zero": set negative values to 0 and continue, recording the negative fraction in the returned object.
- "allow": keep negatives in the Intensity assay (use only for advanced within-sample Intensity analyses with an explicitly compatible downstream transform). Note that `SignalDispersion` always uses non-negative weights (negative values are clipped to 0 internally, since signed signal is not a valid probability-like weight), and the default `log10(pmax(x, 0)+1)` transform / tangent inflection of Super calling also clip negatives to 0. Signed tracks are not part of the recommended canonical workflow.

fail_action

Character. Behavior when a BigWig cannot be imported "stop" (default) fails the whole run; "drop" excludes the failed sample(s) with a clear warning about the changed sample composition.

Value

A `SummarizedExperiment` object. Core assays (domain x sample):

- `Intensity`: integrated BigWig signal within the shared domain.
- `SignalDispersion`: signal-weighted genomic SD within the domain (secondary within-domain architecture descriptor).
- `NativeMaxPeakWidth`: max width of native peaks overlapping the domain (NA when `peak_path` missing).
- `NativeOccupiedWidth`: sum of reduced native peak widths inside the domain (NA when `peak_path` missing).
- `NativePeakCount`: number of native peaks overlapping the domain (NA when `peak_path` missing).

The static shared interval length is stored in `rowData(se)$IntervalWidth`. Imported native peaks (if provided) are stored in `metadata(se)$native_peaks` for downstream peak-level Breadth-Super calling.

Examples

```
if (.Platform$OS.type != "windows") {
  extdata <- system.file("extdata", package = "epiPortrait")
  samples <- data.frame(
    SampleID = c("C1", "C2", "T1", "T2"),
    Condition = c("Control", "Control", "Treatment", "Treatment"),
    bw_path = file.path(extdata, c("C1.bw", "C2.bw", "T1.bw", "T2.bw")),
  )
}
```

```

    peak_path = file.path(extdata, c("C1_peaks.bed", "C2_peaks.bed",
                                     "T1_peaks.bed", "T2_peaks.bed"))
    peaks <- rtracklayer::import(file.path(extdata, "peaks.bed"))
    build_portrait_matrix(samples, consensus_peaks = peaks, workers = 1)
}

```

call_super_domains	<i>Call Super-Domains with Multi-Mode Support</i>
--------------------	---

Description

Identifies super-domains by ranking domains on a feature and separating extreme ("super") from typical domains via a hockey-stick inflection, with replicate-aware support aggregation. Supports three modes:

- "global_consensus" (default; alias "consensus"): calls per-sample, computes replicate support per condition group, and requires the support rule to be met in EVERY group. Identifies domains reproducibly super across all conditions (persistent/global consensus).
- "per_group": EACH replicate is called independently using its own ranked feature distribution; replicate-level calls are then aggregated within each condition using the selected support rule. Group-mean ranks are stored for visualization only and do not determine the group call. Stores one call column per group. Recommended for the main analysis of multi-condition studies. The per-domain x per-replicate call matrix is available via `get_replicate_calls()`.
- "per_sample": calls each sample independently, stores all call columns.

Usage

```

call_super_domains(
  se,
  feature = "Intensity",
  mode = c("global_consensus", "consensus", "per_group", "per_sample"),
  group_var = "Condition",
  method = "elbow",
  log_transform = NULL,
  quantile_cutoff = NULL,
  min_replicate_support = 0.5,
  support_rule = c("majority", "all", "fraction"),
  min_valid_replicates = NULL,
  min_quality = 0.1,
  tie_policy = c("strict", "inclusive"),
  n_bootstrap = NULL,
  seed = NULL,
  min_peak_overlap_fraction = 0.5,
  valid_chroms = NULL,
  verbose = TRUE
)

```

Arguments

se	A SummarizedExperiment from build_portrait_matrix().
feature	Character. Ranking feature. • Per-sample assays: "Intensity" (integrated signal magnitude), or "SignalDispersion" (within-domain signal architecture). • "Breadth": peak-level Breadth-Super calling. Requires the object to carry sample-specific native peak calls (see peak_path in build_portrait_matrix; stored in metadata(se)\$native_peaks). Each replicate's genome-wide eligible native PeakWidth distribution is ranked and cut by an elbow/inflection; broad peaks are then mapped to the shared domains by unique assignment and aggregated across replicates. This keeps Breadth-Super calling decoupled from consensus construction). The static genomic interval length is available as "IntervalWidth" (alias "Width"; read from rowData, constant across samples — use only for static broadness ranking, never for per-sample calls).
mode	Character. "global_consensus" (default), "per_group", or "per_sample". The alias "consensus" is also accepted.
group_var	Character. Column in colData for grouping (used when mode = "per_group" or "global_consensus").
method	Character. Inflection-detection heuristic: • "elbow" (default): point with maximum perpendicular distance to the first-last ranked line (classic knee detection). • "tangent": a ROSE-inspired tangent-optimization inflection (Whyte et al., 2013, Cell 153:307-319). Slides a line whose slope is (max - min)/n and minimizes the number of points below it. Note that this is a geometric variant of ROSE's calculate_cutoff() step — not a bit-exact reproduction — and implements only an inflection step, not the full ROSE pipeline. Ignored when quantile_cutoff is provided.
log_transform	Logical or NULL. Whether to log10(x + 1)-transform the feature before ranking. NULL (default) auto-selects per feature (Intensity and SignalDispersion are log-transformed, matching epiPortrait conventions). Set FALSE to rank on the raw feature scale (needed for a ROSE-style tangent benchmark on untransformed signals).
quantile_cutoff	Numeric or NULL. If provided (e.g. 0.95), bypasses inflection detection and uses an explicit top fraction instead. For "Intensity" the cutoff is the quantile_cutoff quantile of the ranked domain values. For "Breadth" it is the quantile_cutoff quantile of each replicate's genome-wide eligible native PeakWidth distribution — i.e. the top 100*(1-quantile_cutoff)\ "top fraction" convention as Intensity: 0.95 = top 5\ explicit user opt-in (a common broad-domain practice) with no quality gate; it never auto-falls-back to, or from, the data-driven inflection path, so the no-call semantics of the default path are unchanged. Must be in (0, 1).
min_replicate_support	Numeric. Minimum support fraction used when support_rule = "fraction" (default 0.5).

support_rule	Character. Replicate-support rule for per-group / consensus calls: • "majority" (default): strictly more than half of the replicates must call Super (n=2 needs 2, n=3 needs 2, n=4 needs 3). • "all": every replicate must call Super. • "fraction": at least min_replicate_support fraction. no-call replicates never count as Super and never inflate support.
min_valid_replicates	Integer or NULL. Minimum number of valid (non-NA) replicates required before a domain can be called Super in a group. When NULL (default), it is set to the number required by the selected support rule (majority: $\text{floor}(n/2)+1$; all: n; fraction: $\text{ceiling}(\text{fraction} \cdot n)$), so a "Super + no_call" pair in an $n=2$ group is Uncertain rather than Typical. Below the requirement the group call is NA / Uncertain. Note the deliberate asymmetry with support_rule = "fraction": because the user explicitly chose how much support suffices, a low threshold (e.g. 0.5 with $n=2$ resolves the requirement to 1) can call a "Super + no_call" pair Super — the no-call replicate never counts as Super, it simply does not lower the bar the user set. Use the stricter majority rule if that behaviour is unwanted.
min_quality	Numeric. Minimum inflection quality for a reliable call (see find_hockey_inflection; default 0.1).
tie_policy	Character. How to classify domains whose value equals the cutoff. "strict" (default) requires value > cutoff, matching the original ROSE super-enhancer selection (rankBy_vector > cutoff); this avoids inflating the super set when discrete features tie exactly at the cutoff. "inclusive" uses >= cutoff (legacy behaviour).
n_bootstrap	Integer or NULL. Bootstrap resamples for the cutoff stability interval and success rate (default NULL; set e.g. 200 to enable). The interval reflects resampling stability over the fixed ranked set, not a classical iid confidence interval. Its information content shrinks as the ranked set grows (interval width $\sim 1/\sqrt{n}$); at genome-wide scale the primary cutoff-stability check is per-replicate cutoff consistency via get_call_provenance()\$replicates. For "Intensity" the ranked set is the candidate domains; for "Breadth" it is each replicate's eligible native PeakWidth distribution (a per-replicate interval is stored in prov\$replicates[[s]]), and it applies only to the data-driven inflection path (ignored with a warning when quantile_cutoff is set).
seed	Integer or NULL. Random seed for the bootstrap (default NULL). A NULL seed leaves the resampling truly random (recommended for general use); pass a fixed integer to make the bootstrap reproducible.
min_peak_overlap_fraction	Numeric. Only used when feature = "Breadth". Minimum fraction of a native peak's genomic extent that must fall inside a shared domain for the peak's Broad evidence to be assigned to that domain (default 0.5). A native peak is assigned to at most ONE shared domain (unique assignment by maximum overlap); peaks not reaching the threshold are Unmapped, and tied maxima are Ambiguous. Prevents edge-overlap double counting of Broad evidence.

valid_chroms	Character or NULL. Only used when feature = "Breadth". Optional vector of allowed chromosomes for the genome-wide eligible native peak set (e.g. c("chr1", ..., "chr22") or the standard chromosomes of the assembly). NULL (default) keeps all finite-length seqlevels.
verbose	Logical. Print messages (default TRUE).

Details

Optional H3K27ac / ROSE benchmark. If the goal is a head-to-head comparison with the original ROSE method (Whyte et al., 2013, Cell 153:307-319), use method = "tangent" with log_transform = FALSE (raw signal, ROSE-style scale). Because the tangent implementation is a geometric variant rather than a bit-exact port of ROSE's calculate_cutoff(), validate any ROSE-reproduction numbers against the reference implementation before drawing quantitative conclusions. This is an optional benchmark setting, not the package's primary workflow; the core value of epiPortrait is the Intensity x native-Breadth decomposition with replicate support. The log-transformed default is retained as the epiPortrait convention because it stabilises the skewed intensity distribution for the elbow heuristic and downstream analysis.

Value

A SummarizedExperiment with rowData columns for each call. Call provenance (feature, method, requested/used log_transform, cutoff, cutoff_stability_interval, bootstrap_success_rate, quality_score, call_status, n_super) is stored in metadata(se)\$superdomain_calls.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        method = "tangent", log_transform = FALSE)
table(SummarizedExperiment::rowData(se)$Intensity_Domain_Type)
```

check_signal_compatibility

Check Signal Compatibility Across BigWig Samples

Description

Performs pre-analysis QC on a sample sheet of BigWig tracks Checks file existence, signal ranges, negative/zero fractions, chromosome naming, and cross-sample global scale consistency. Returns a per-sample summary with a PASS / WARNING / FAIL status and emits warnings for global incompatibilities.

Usage

```
check_signal_compatibility(
  sample_sheet,
  regions = NULL,
  max_windows = 2000,
```

```

    seed = NULL,
    verbose = TRUE
  )

```

Arguments

sample_sheet	A data.frame with at least SampleID, Condition, and bw_path.
regions	A GRanges or NULL. If provided, signals are evaluated within these regions (e.g. candidate domains). If NULL, a small set of genomic windows is sampled from each track.
max_windows	Integer. Number of windows to sample when regions is NULL (default 2000).
seed	Integer or NULL. Optional random seed for reproducible window / region sub-sampling (default NULL = random draw). Pass a fixed seed to make the QC reproducible; NULL keeps the draw truly random.
verbose	Logical. Print a summary (default TRUE).

Value

A data.frame with one row per sample: SampleID, Condition, signal-sum (QCWindowSignalSum when sampling tiles, DomainSetSignalSum when regions are provided), MedianSignal, MeanSignal, ZeroFraction, NegativeFraction, UpperQuantile, GlobalScaleRatio, Status. The object carries a print method that displays the summary table when verbose = TRUE; the returned value is otherwise an ordinary data.frame.

Examples

```

if (.Platform$OS.type != "windows") {
  extdata <- system.file("extdata", package = "epiPortrait")
  ss <- data.frame(
    SampleID = c("C1", "C2", "T1", "T2"),
    Condition = c("Control", "Control", "Treatment", "Treatment"),
    bw_path = file.path(extdata, c("C1.bw", "C2.bw", "T1.bw", "T2.bw"))
  )
  check_signal_compatibility(ss, max_windows = 500)
}

```

combine_superdomain_calls

Combine Intensity and Breadth Super-Domain Calls

Description

Combines the Intensity-Super and Breadth-Super calls into the final 2D architecture taxonomy: Typical / Intensity-Super / Breadth-Super / Dual-Super / Uncertain. Dual-Super means the domain is simultaneously high in integrated signal and unusually broad.

Usage

```
combine_superdomain_calls(
  se,
  intensity_feature = "Intensity",
  width_feature = "Breadth",
  call_col_suffix = "Domain_Type",
  group_var = NULL
)
```

Arguments

<code>se</code>	SummarizedExperiment after call_super_domains on Intensity and Breadth.
<code>intensity_feature</code>	Character (default "Intensity").
<code>width_feature</code>	Character. Breadth axis feature (default "Breadth"). This is the peak-level native breadth call, NOT a static interval length.
<code>call_col_suffix</code>	"Domain_Type" (consensus) or "Call__Control" (per_group).
<code>group_var</code>	Character or NULL. If provided, the calls are combined per condition group (requires mode = "per_group"), producing Combined_Class__<group> columns used by compare_superdomain_classes(). If NULL (default), the single call_col_suffix columns are combined into Combined_Domain_Class.

Value

se with Combined_Domain_Class column (or Combined_Class__).

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
  method = "tangent", log_transform = FALSE)
se <- call_super_domains(se, feature = "Breadth")
se <- combine_superdomain_calls(se)
table(SummarizedExperiment::rowData(se)$Combined_Domain_Class)
```

compare_superdomain_classes

Compare Combined Super-Domain Classes Between Conditions

Description

Computes transitions of the combined taxonomy (Intensity-Super / Breadth-Super / Dual-Super / Typical / Uncertain) between two conditions, using the Combined_Class__<group> columns produced by combine_superdomain_calls(se, group_var = ...). This closes the core loop: the combined architecture taxonomy is tracked across conditions not just a single feature.

Usage

```
compare_superdomain_classes(se, ref_group, target_group)
```

Arguments

se	SummarizedExperiment after combine_superdomain_calls(group_var).
ref_group	Reference condition.
target_group	Treatment condition.

Value

se with Combined_Relative_Class_Transition column.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        method = "tangent", log_transform = FALSE)
se <- call_super_domains(se, feature = "Breadth",
                        mode = "per_group", group_var = "Condition")
se <- combine_superdomain_calls(se, group_var = "Condition")
se <- compare_superdomain_classes(se, ref_group = "Control",
                                target_group = "Treatment")
table(SummarizedExperiment::rowData(se)$Combined_Relative_Class_Transition)
```

compare_superdomains	<i>Compare Super-Domain Status Between Conditions</i>
----------------------	---

Description

Compare Super-Domain Status Between Conditions

Usage

```
compare_superdomains(
  se,
  group_var = "Condition",
  ref_group,
  target_group,
  feature = "Intensity",
  cutoff_scope = c("relative", "reference", "pooled")
)
```

Arguments

se	after call_super_domains(mode="per_group").
group_var	Character.
ref_group	Reference condition.
target_group	Treatment condition.
feature	Feature name (default "Intensity").
cutoff_scope	Character. How the super/non-super threshold is defined The default "relative" calls each group independently, so a transition such as Typical_to_Super reflects a RELATIVE prominence change and must NOT be interpreted as an absolute signal gain. • "relative" (default): per-group cutoff; output is a relative class switch (rename the classes accordingly, do not call them Gain/Loss). • "reference": use the reference group's cutoff for both groups; closer to absolute gain/loss when BigWigs are quantitatively comparable. • "pooled": a common cutoff estimated on the pooled (ref + target) signal, applied to both groups.

For "reference" / "pooled" the feature must be a per-sample assay, and the two groups must be quantitatively comparable. These modes inherit the FULL replicate-support configuration of the primary call from stored provenance (support_rule, min_replicate_support, tie_policy, min_valid_replicates) so transition semantics match the group calls.

Value

se with *_Transition column.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        method = "tangent", log_transform = FALSE)
se <- compare_superdomains(se, group_var = "Condition",
                        ref_group = "Control", target_group = "Treatment")
table(SummarizedExperiment::rowData(se)$Intensity_Transition)
```

Description

Computes a continuous effect descriptor for domain breadth between two condition groups. For each shared domain it summarizes the mapped native width per group (median NativeMaxPeakWidth across replicates) and reports:

- WidthDelta_bp: median(target) - median(ref)
- log2WidthRatio: log2(median(target)/median(ref))
- WidthDirection: "Expansion" / "Contraction" / "Stable"

This is a CONTINUOUS effect descriptor, not a significance test: with a handful of replicates it reports the observed magnitude of width remodeling. Statistical testing is left to downstream / external frameworks.

Usage

```
compute_width_transition(
  se,
  group_var = "Condition",
  ref_group,
  target_group,
  effect_threshold = log2(1.5),
  min_replicates = 1L
)
```

Arguments

- | | |
|-------------------------|---|
| se | A SummarizedExperiment after build_portrait_matrix() with native peaks (requires the NativeMaxPeakWidth assay). |
| group_var | Character. colData column for grouping (default "Condition"). |
| ref_group, target_group | Character. Reference and target condition groups. |
| effect_threshold | Numeric or NULL. Minimum log2WidthRatio (fold-change on log2 scale) above which a domain is called "Expansion" or "Contraction"; below this it is "Stable" (default 0.585 ~ 1.5-fold). This is a DESCRIPTIVE effect-size label, not a statistical significance call. Set NULL to emit only the continuous columns (no hard classification). |
| min_replicates | Integer. Minimum number of replicates per group with a finite width for the domain to be scored (default 1; NA otherwise). |

Value

A SummarizedExperiment with pair-specific rowData columns WidthDelta_bp__<ref>_vs_<target> and log2WidthRatio__<ref>_vs_<target> (and WidthDirection__<ref>_vs_<target> when a threshold is set), so multiple comparisons can be stored without overwriting each other.

Examples

```
data(example_se)
se <- compute_width_transition(example_se, ref_group = "Control",
                             target_group = "Treatment")
head(SummarizedExperiment::rowData(se)$WidthDelta_bp__Control_vs_Treatment)
```

epiPortrait

epiPortrait: Replicate-Aware Epigenomic Domain Profiling

Description

epiPortrait provides replicate-aware quantitative phenotyping of epigenomic domains across biological samples and conditions. Each shared genomic domain is characterized by three measurement layers: **Intensity** (integrated signal magnitude), **native peak breadth geometry** (per-replicate native peak width), and **SignalDispersion** (continuous within-domain signal architecture). Intensity and Breadth are the canonical classification axes; super-domain states (Intensity-Super, Breadth-Super, Dual-Super) are called with replicate-level evidence and explicit support rules, while SignalDispersion is retained as a secondary architecture descriptor. The package supports condition transitions, continuous domain-width remodeling, signal quality control, genome-aware domain annotation, optional BEDPE contact evidence, expression-aware candidate-gene prioritization, and result visualization. Primary use case: histone-mark ChIP-seq / CUT&Tag; continuous-track workflows also apply to ATAC-seq-derived domains.

Core workflow

1. [get_consensus_peaks](#) — Build consensus domain universe
2. [stitch_epi_peaks](#) — Stitch proximal peaks into macro-domains
3. [build_portrait_matrix](#) — Extract portrait matrix from BigWigs
4. [normalize_portrait](#) — Normalize Intensity while preserving SignalDispersion
5. [call_super_domains](#) — Intensity / peak-level Breadth calling
6. [annotate_epi_domains](#) — Domain-aware annotation & candidate genes

Visualization

- [plot_portrait_pca](#) — PCA of quantitative domain features
- [plot_portrait_correlation](#) — Sample correlation heatmap
- [plot_peak_track](#) — Raw BigWig coverage track
- [plot_hockey_stick](#) — Super-domain ranking plot

Supplementary

- [combine_superdomain_calls](#) — Combine Intensity/Breadth super calls
- [compare_superdomains](#) — Condition-aware super-domain transitions
- [filter_promoter_peaks](#) — Exclude promoter-proximal peaks
- [filter_blacklist](#) — Filter blacklisted regions

Author(s)

Maintainer: Ying ZHANG <12207129@zju.edu.cn> ([ORCID](#))

Other contributors:

- Xingze HUANG <22407026@zju.edu.cn> ([ORCID](#)) [contributor]
- Ye CHEN <chenyephd@zju.edu.cn> [contributor]
- Liang XU <xuliang.phd@zju.edu.cn> (This work is funded by the Fundamental Research Funds for the Central Universities (226-2025-00101) and the National Natural Science Foundation of China (32270746, 82203247, 82203415).) [contributor, funder]

See Also

Useful links:

- <https://github.com/zying106/epiPortrait>
- Report bugs at <https://github.com/zying106/epiPortrait/issues>

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity", verbose = FALSE)
table(SummarizedExperiment::rowData(se)$Intensity_Domain_Type)
```

example_se

Example Epigenetic Portrait Dataset

Description

A simulated SummarizedExperiment containing 500 genomic domains on chr21 (hg38) with quantitative features (Intensity, SignalDispersion, NativeMaxPeakWidth, NativeOccupiedWidth, NativePeakCount) measured across 6 samples (3 Control, 3 Treatment). The synthetic data deliberately mixes replicate-support patterns (strong 3/3, boundary 2/3, weak 1/3, and a few no-call domains) so replicate-aware calling is instructive; see `metadata(example_se)$simulation_provenance`.

Usage

```
data("example_se")
```

Format

A SummarizedExperiment object with:

assays 5 matrices (Intensity, SignalDispersion, NativeMaxPeakWidth, NativeOccupiedWidth, NativePeakCount), each 500 x 6

rowRanges GRanges on chr21 (hg38, all intervals in-bounds)

rowData includes Simulation_Scenario (the generative mechanism: Typical / Intensity-high / Breadth-contraction / Dual-with-breadth-contraction) and IntervalWidth (static genomic interval length)

colData SampleID, Condition (Control/Treatment), BioReplicate, bw_path (NA), peak_path (NA)

metadata native_peaks, native_peak_available, feature_definitions, simulation_provenance

Examples

```
data(example_se)
example_se
SummarizedExperiment::assayNames(example_se)
S4Vectors::metadata(example_se)$simulation_provenance
```

```
export_epiportrait_results
```

Export Complete epiPortrait Results to Disk

Description

Writes a self-contained results directory: flat TSV tables, per-assay matrices, per-call tables, an object manifest, and the complete SummarizedExperiment as RDS.

Usage

```
export_epiportrait_results(
  se,
  outdir = "epiPortrait_results",
  save_object = TRUE,
  group_var = "Condition"
)
```

Arguments

se	A SummarizedExperiment.
outdir	Character. Output directory (created if missing).
save_object	Logical. Also save se as RDS (default TRUE).
group_var	Character. Passed to get_domain_results for group-mean columns.

Value

Invisibly the path to the output directory.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity", verbose = FALSE)
out <- export_epiportrait_results(se, outdir = tempfile("epi_export"))
```

filter_blacklist	<i>Filter ENCODE Blacklist Regions</i>
------------------	--

Description

Filter ENCODE Blacklist Regions

Usage

```
filter_blacklist(gr, genome = "hg38", blacklist_path = NULL)
```

Arguments

gr	A GRanges object.
genome	Character. "hg38", "hg19", "mm10", "mm9".
blacklist_path	Custom BED path or NULL (uses built-in).

Value

Filtered GRanges.

Examples

```
gr <- GenomicRanges::GRanges("chr1", IRanges::IRanges(c(100, 5000000), width = 200))
filter_blacklist(gr, genome = "hg38")
```

filter_promoter_peaks	<i>Filter Peaks by Genomic Annotations (Promoter Exclusion)</i>
-----------------------	---

Description

Filters out peaks that overlap with specified genomic regions, typically used to remove promoter-proximal peaks before Super-Element analysis.

Usage

```
filter_promoter_peaks(gr, genome = "hg38", upstream = 2000, downstream = 2000)
```

Arguments

gr	A GRanges object of peaks.
genome	A character string ("hg38", "hg19", "mm10", etc.) or a TxDb object.
upstream	Number of bp upstream of TSS to define promoter (default: 2000).
downstream	Number of bp downstream of TSS to define promoter (default: 2000).

Value

A filtered GRanges object.

Examples

```
gr <- GenomicRanges::GRanges("chr1", IRanges::IRanges(c(100, 5000000), width = 200))
if (requireNamespace("TxDb.Hsapiens.UCSC.hg38.knownGene", quietly = TRUE)) {
  filter_promoter_peaks(gr, genome = "hg38")
}
```

find_hockey_inflection

Find the Hockey-Stick Inflection Point

Description

Locates the inflection point of a ranked (ascending) value curve used to separate super-domains from typical domains. Two heuristics are supported:

Usage

```
find_hockey_inflection(
  x,
  method = c("elbow", "tangent"),
  min_quality = 0.1,
  verbose = FALSE
)
```

Arguments

x	Numeric vector of feature values (e.g. per-domain intensity).
method	Character. "elbow" (default) or "tangent".
min_quality	Numeric. Minimum quality_score for a reliable call. The score is dimensionless (0-1, computed on [0, 1]-normalized coordinates) and measures curve prominence: ~0 for a linear curve, ~0.3-0.7 for real hockey-stick profiles. Default 0.1 rejects near-linear curves while retaining genuine inflection points.
verbose	Logical. Print warnings on low-quality calls (default FALSE).

Details

- "elbow" (default): the point with the maximum perpendicular distance to the line connecting the first and last ranked points. Classic knee/elbow detection; cheap and robust when signals are log-transformed.

- "tangent": a ROSE-inspired tangent-optimization inflection (Whyte et al., 2013, Cell 153:307-319). Slides a line whose slope is $(\max - \min)/n$ and minimizes the number of points below it. Note that this is a geometric variant of ROSE's `calculate_cutoff()` step — not a bit-exact reproduction — and implements only an inflection step, not the full ROSE pipeline (stitching, input subtraction). `epiPortrait` does not log-transform the input unless `log_transform` is requested.

A method-agnostic `quality_score` (max perpendicular distance relative to the value range) is returned. When it falls below `min_quality`, `call_status` is "no_call" and the inflection is not reliable.

Value

A list with elements:

<code>inflection_idx</code>	Index of the inflection point in the sorted vector.
<code>cutoff_value</code>	Feature value at the inflection point.
<code>quality_score</code>	Right-tail curve-prominence quality (0-1). A left-loaded / inverse hockey-stick curve has quality 0 and is not called.
<code>method</code>	The method used.
<code>call_status</code>	"called" or "no_call".
<code>reason</code>	Reason for "no_call", if any.

Examples

```
x <- c(rep(1, 50), seq(1, 100, length.out = 50))
find_hockey_inflection(x, method = "elbow")
find_hockey_inflection(x, method = "tangent")
```

<code>get_call_provenance</code>	<i>Extract Super-Domain Call Provenance</i>
----------------------------------	---

Description

Returns the provenance for a feature's super-domain calls (cutoff, stability interval, quality, per-group replicate calls), stored in `metadata(se)$superdomain_calls`.

Usage

```
get_call_provenance(se, feature = "Intensity")
```

Arguments

<code>se</code>	A SummarizedExperiment after <code>call_super_domains</code> .
<code>feature</code>	Character. Feature name (default "Intensity").

Value

A list (the stored provenance), or NULL if unavailable.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity", verbose = FALSE)
get_call_provenance(se, "Intensity")$method
```

get_call_results

Extract Call Results as a Flat Table

Description

Returns per-domain call results for a feature in long or wide form: for each condition group, the call, replicate support and rank.

Usage

```
get_call_results(se, feature = "Intensity", long = FALSE)
```

Arguments

se	A SummarizedExperiment after call_super_domains.
feature	Character. Feature name (default "Intensity").
long	Logical. If TRUE, returns long format (one row per domain x group); if FALSE, wide format with one column per group.

Value

A data.frame.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
head(get_call_results(se, "Intensity"))
head(get_call_results(se, "Intensity", long = TRUE))
```

```
get_combined_class_results
```

Extract Combined Class Results

Description

Extract Combined Class Results

Usage

```
get_combined_class_results(se)
```

Arguments

`se` A SummarizedExperiment after combine_superdomain_calls.

Value

A data.frame of combined classes (and Combined_Domain_Class if present).

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity", verbose = FALSE)
se <- call_super_domains(se, feature = "Breadth", verbose = FALSE)
se <- combine_superdomain_calls(se)
head(get_combined_class_results(se))
```

```
get_consensus_peaks
```

Generate Consensus Domain Universe for epiPortrait

Description

Merges multiple sample peak/domain sets into a unified genomic coordinate system while applying a reproducibility filter. This is CONSENSUS / domain-universe construction, not a peak caller (epiPortrait does not call MACS2/SICER-type peaks).

Usage

```
get_consensus_peaks(gr_list, min_reps = 2)
```

Arguments

`gr_list` A GRangesList or list of GRanges objects containing peaks/domains from all samples.

`min_reps` Minimum number of replicate peak sets contributing at least one overlap to a reduced consensus interval for it to be retained (default: 2).

Details

Union-support chaining semantics. All sample peaks are first `reduce()`-merged into connected intervals, then each reduced interval counts how many sample peak sets contribute at least one overlapping peak (`Support_Reps`). Connected overlapping peaks can chain into a wider reduced interval; `Support_Reps` is an interval-level replicate contribution count, NOT base-pair-wise support across replicates. A `Support_Reps = 3` interval does not imply every base pair is covered by all three replicates.

Value

A filtered `GRanges` object representing the consensus domain universe.

Examples

```
gr1 <- GenomicRanges::GRanges("chr1", IRanges::IRanges(c(100, 5000), width = 200))
gr2 <- GenomicRanges::GRanges("chr1", IRanges::IRanges(c(150, 5200), width = 200))
get_consensus_peaks(list(rep1 = gr1, rep2 = gr2), min_reps = 2)
```

get_domain_genes

Extract and Prioritize Candidate Genes for Domains

Description

Extracts candidate (domain-associated) genes from the stored domain-gene evidence (`metadata(se)$domain_gene_links`), optionally applying RNA-seq expression prioritization. Ordinal prioritization is explicit: evidence relationships come first, expression is a secondary tie-breaker; no black-box composite score is computed.

Usage

```
get_domain_genes(
  se,
  domains = NULL,
  relations = c("nearest_tss", "promoter_overlap", "gene_body_overlap",
    "fully_contained", "bedpe_promoter_contact"),
  group = NULL,
  expression_priority = c("none", "expressed_first", "high_expression_first"),
  min_expression = NULL,
  unique_genes = TRUE,
  nearest_tss_cutoff_bp = getOption("epiPortrait.nearest_tss_cutoff_bp", 10000),
  rank_by = c("tier", "bedpe_score")
)
```

Arguments

<code>se</code>	A SummarizedExperiment after <code>annotate_epi_domains()</code> .
<code>domains</code>	Logical vector (length <code>nrow(se)</code>) selecting which domains to consider, or <code>NULL</code> for all.
<code>relations</code>	Character. Evidence relation types to include (default all: <code>nearest_tss</code> , <code>promoter_overlap</code> , <code>gene_body_overlap</code> , <code>fully_contained</code> , <code>bedpe_promoter_contact</code>).
<code>group</code>	Character or <code>NULL</code> . Condition group whose expression is used for prioritization (<code>NULL</code> = no expression).
<code>expression_priority</code>	Character. "none" (default), "expressed_first", or "high_expression_first".
<code>min_expression</code>	Numeric or <code>NULL</code> . Descriptive threshold for <code>expression_status</code> = "Expressed". For TPM/CPM with <code>expression_priority</code> = "expressed_first", a value like 1 is a convenience default. For VST / rlog / normalized_counts, <code>expressed_first</code> requires an explicit user threshold (no universal meaning of 1). For "high_expression_first" no threshold is needed.
<code>unique_genes</code>	Logical. Collapse to unique genes (default <code>TRUE</code>), retaining the highest-priority domain association per gene.
<code>nearest_tss_cutoff_bp</code>	Numeric. Distance (bp) below which a <code>nearest_tss</code> relation is treated as a strong PROXIMAL candidate and ranked above <code>gene_body_overlap</code> /distal links; a currently-nearest but FAR TSS (beyond this cutoff) is downgraded to the lowest evidence tier. This prevents a "nearest but remote" gene (e.g. 500 kb away) from ranking equal to <code>promoter_overlap</code> evidence. Defaults to the option <code>epiPortrait.nearest_tss_cutoff_bp</code> , or 10000 if unset. This keeps <code>annotate_epi_domains()</code> and <code>get_domain_genes()</code> on the same proximity definition.
<code>rank_by</code>	Character. Primary ordering key for <code>candidate_priority</code> . "tier" (default): evidence hierarchy first (see Details), expression as tie-breaker — the original behaviour. "bedpe_score": descending 3D contact strength (<code>bedpe_contact_score</code>) first, then the evidence tier and expression as tie-breakers. Requires that <code>annotate_epi_domains()</code> was run with <code>bedpe_score_col</code>; pairs without a usable score fall back to the tier ordering. This implements contact-strength-first target prioritization (dominant-loop style).

Value

A data.frame: `domain_id`, `gene_id`, `gene_symbol`, `relation_types`, `bedpe_supported`, `bedpe_support_count`, `bedpe_contact_score`, `nearest_tss_distance_bp` (when available), `expression_value`, `expression_status`, `expression_rank`, `candidate_priority`.

Examples

```
data(example_se)
if (requireNamespace("TxDb.Hsapiens.UCSC.hg38.knownGene", quietly = TRUE)) {
  se <- annotate_epi_domains(example_se, genome = "hg38")
}
```

```
    head(get_domain_genes(se))
}
```

get_domain_results	<i>Extract Per-Domain Results Table</i>
--------------------	---

Description

Returns a flat data.frame of domain-level results for export or inspection. Includes genomic coordinates, IntervalWidth, and — when present — per-group calls/support/rank, combined classes, transitions and group mean quantitative values.

Usage

```
get_domain_results(se, group_var = "Condition", include_assay_means = TRUE)
```

Arguments

- se A SummarizedExperiment with super-domain calls.
- group_var Character. Column in colData for group-mean feature columns. If NULL, no group-mean columns are added.
- include_assay_means Logical. Add per-group mean of each dynamic assay (default TRUE when group_var is provided).

Value

A data.frame.

Examples

```
data(example_se)
head(get_domain_results(example_se, group_var = "Condition"))
```

get_mark_preset	<i>Get Mark-Specific Preset</i>
-----------------	---------------------------------

Description

Get Mark-Specific Preset

Usage

```
get_mark_preset(mark)
```

Arguments

mark "H3K27ac", "H3K4me3", "H3K27me3", "H3K9me3", etc.

Value

List of preset parameters, including:

- label: human-readable domain label
- stitch_distance: default stitching gap in bp. 0 / NULL is a SENTINEL meaning "do not automatically re-stitch at the preset-workflow level" — treat it as "skip stitch_epi_peaks()" (e.g. broad repressive marks such as H3K27me3 / H3K9me3, whose domains are already spatially clustered by the upstream broad-domain caller, and unknown/generic marks). Note that stitch_epi_peaks(x, 0) itself still merges overlapping/adjacent ranges; the sentinel must be handled by the caller (see the vignette preset workflow).
- primary_feature: ranking feature
- mark_class: "active" (H3K27ac, H3K4me3, H3K4me1) or "broad_repressive" (H3K27me3, H3K9me3)
- taxonomy_style: "super_domain" (active marks) or "repressive_remodeling" (broad repressive marks). This controls DISPLAY terminology only; the internal canonical classes (Intensity-Super / Breadth-Super / Dual-Super) are unchanged.
- class_labels: display aliases for the canonical taxonomy.

Examples

```
get_mark_preset("H3K27ac")
get_mark_preset("H3K27me3")
```

get_replicate_calls *Extract Per-Domain x Per-Replicate Call Matrix*

Description

Exposes the per-domain x per-replicate super-domain evidence that underlies the group-level replicate-support call. For mode = "per_group" the matrix is read from stored provenance; for mode = "per_sample" it is reconstructed from the <feature>_Call__<sample> rowData columns. Group calls are replicate-support aggregates; this matrix is the auditable evidence trail.

Usage

```
get_replicate_calls(se, feature = "Intensity", group = NULL, long = FALSE)
```

Arguments

se	A SummarizedExperiment after call_super_domains().
feature	Character. Feature (default "Intensity").
group	Character or NULL. For per_group calls, restrict to the samples of this condition group. NULL returns all samples.
long	Logical. If TRUE, returns long format (domain_id, SampleID, Group, call); if FALSE (default), a domain x sample matrix.

Value

A matrix (long = FALSE) or data.frame (long = TRUE).

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
m <- get_replicate_calls(se, feature = "Intensity", group = "Control")
head(m)
head(get_replicate_calls(se, feature = "Intensity", group = "Control",
                        long = TRUE))
```

get_sample_results	<i>Extract Sample-Level Results / QC Table</i>
--------------------	--

Description

Extract Sample-Level Results / QC Table

Usage

```
get_sample_results(se)
```

Arguments

se	A SummarizedExperiment.
----	-------------------------

Value

A data.frame with one row per sample (colData + QC).

Examples

```
data(example_se)
get_sample_results(example_se)
```

```
get_transition_results
```

Extract Transition Results

Description

Extract Transition Results

Usage

```
get_transition_results(se)
```

Arguments

`se` A SummarizedExperiment after `compare_superdomains / compare_superdomain_classes`.

Value

A data.frame of transition columns.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
se <- compare_superdomains(se, group_var = "Condition",
                        ref_group = "Control", target_group = "Treatment")
head(get_transition_results(se))
```

```
get_uncertain_cause
```

Explain Why a Domain Call Is Uncertain

Description

For a per-group feature call, classifies the cause of each NA / Uncertain group call so the ambiguity is auditable:

- `no_native_peak_in_condition` (Breadth): the domain has no native peak evidence in any replicate of the group — e.g. it is condition-specific (present only in the other condition's universe).
- `no_valid_signal_in_any_replicate` (signal features): no replicate produced a usable ranked distribution.
- `insufficient_valid_replicates`: some, but fewer than `min_valid_replicates`, replicates carried evidence (e.g. 1/3 with a majority rule).
- `inflection_no_call_all_replicates`: every replicate's inflection was unreliable (`no_call`).

Domains whose group call is *not* NA get `cause = NA`. This reads stored provenance only and never re-computes calls.

Usage

```
get_uncertain_cause(se, feature = "Breadth", group, group_var = NULL)
```

Arguments

se	A SummarizedExperiment after call_super_domains(mode = "per_group").
feature	Character. Feature (default "Breadth").
group	Character. Condition group.
group_var	Character or NULL. Column used for grouping; resolved from stored provenance when NULL (default "Condition").

Value

A data.frame with columns Domain_ID, Group_Call, N_Valid_Replicates, Min_Valid_Replicates, Cause.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Breadth",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
get_uncertain_cause(se, feature = "Breadth", group = "Control")
```

integrate_cross_mark *Integrate a Second Mark's Super-Domain Call onto Another Object*

Description

Overlays the per-sample super-domain calls of object B (se_b) onto the shared domains of object A (se_a). The two objects may have DIFFERENT domain universes (e.g. a promoter-filtered + 12.5 kb-stitched H3K27ac universe vs a 5 kb-stitched H3K4me3 universe); each se_a domain is classified from the se_b calls of the se_b domains that overlap it. This enables the two-axis design: e.g. whether promoter-proximal ARBS that an enhancer-mark pipeline deliberately excludes are captured by the promoter-mark WIDTH axis ("boundary rescue"), or the orthogonal Intensity x Width combinations.

Usage

```
integrate_cross_mark(
  se_a,
  se_b,
  call_fmt_a = "Breadth_Call__%s",
  call_fmt_b = "Intensity_Call__%s",
  mark_a = NULL,
  mark_b = NULL,
```

```

    timepoints = NULL,
    aggregate_ov = c("any_super", "majority"),
    min_overlap_bp = 0
  )

```

Arguments

se_a	A SummarizedExperiment with per-sample calls (the LEFT object whose domains are annotated).
se_b	A SummarizedExperiment with per-sample calls (the RIGHT object whose evidence is overlaid).
call_fmt_a, call_fmt_b	Character. sprintf formats producing the per-sample call column names, e.g. "Breadth_Call__%s" / "Intensity_Call__%s". At least the union of used time points must exist in each object.
mark_a, mark_b	Character or NULL. Short labels (e.g. "Width", "Intensity") prepended to the new rowData column names. When NULL they are derived from the call column prefix (text before "_Call__").
timepoints	Character or NULL. Time points to integrate. When NULL, the intersection of the column-derived time points of both objects is used (falling back to colData\$TimePoint).
aggregate_ov	Character. How to collapse MULTIPLE overlapping se_b domains onto one se_a domain: "any_super" (default): Super if ANY overlapping se_b domain is super; otherwise Typical if any overlapping domain is called (non-NA); Uncertain if all overlapping calls are NA; NA (outside-universe) when there is no overlap at all. "majority": strict majority (> half) of the overlapping non-NA se_b calls are super.
min_overlap_bp	Numeric. Minimum base pairs of overlap required for a se_b domain to count (default 0 = any overlap within 1 bp).

Value

se_a with per-time-point rowData columns:

- <mark_b>_Call__<tp>: aggregated se_b call
- <mark_b>_NOverlaps__<tp>: number of overlapping se_b domains (0 = not covered by the se_b universe)

plus a long-format integration table stored in metadata(se_a)\$cross_mark_integration and a provenance list in metadata(se_a)\$cross_mark_provenance.

Examples

```

data(example_se)
## both objects must carry per_sample calls; see examples of
## transition_matrix_per_sample() for building a two-object setup.

```

normalize_portrait	<i>Normalize Portrait Assays</i>
--------------------	----------------------------------

Description

Normalizes the Intensity assay while preserving SignalDispersion in its native bp-scale units. Corrects the Intensity matrix to account for differences in sequencing depth or distribution across samples.

Usage

```
normalize_portrait(se, method = "None", force_TMM = FALSE)
```

Arguments

se	A SummarizedExperiment object from build_portrait_matrix().
method	Normalization method. Options are: • "None" (default): Skips normalization. Recommended — epiPortrait expects BigWigs that are already quantitatively comparable (same pipeline, genome, normalization; e.g. CPM/RPGC/ spike-in). Post-hoc rescaling risks removing genuine global biological shifts. • "TotalSignal": Scales libraries to the mean total signal of the analyzed domain set. WARNING: this assumes approximately conserved total signal across samples and may remove real global gains/losses (e.g. drug-induced chromatin loss). • "TMM": Trimmed Mean of M-values. Designed for count data; use force_TMM = TRUE for continuous BigWig signals. • "Quantile": Forces identical distributions across samples (uses limma). Not recommended for between-condition comparison. Row-wise Z-score scaling is not offered here: it is a display/clustering transform that would destroy the cross-domain magnitude ranking used for Super calling. Use plot_portrait_pca() or plotting layers for display-only scaling.
force_TMM	Logical. If TRUE, allows TMM normalization on continuous BigWig signals despite TMM being designed for count data. Default is FALSE.

Value

A normalized SummarizedExperiment object.

Examples

```
data(example_se)
normalize_portrait(example_se, method = "None")
```

plot_class_transition *Plot Combined Architecture Transition Matrix*

Description

Heatmap of the combined architecture class transition between two conditions (Control rows -> Treatment columns). Displays row percentage and domain counts. This is a **RELATIVE** architecture-state transition (per condition-group cutoffs) and must not be read as absolute gain/loss.

Usage

```
plot_class_transition(
  se,
  ref_group,
  target_group,
  normalize = c("row_percent", "count", "none"),
  include_uncertain = TRUE,
  mark = NULL,
  ref_role = NULL,
  target_role = NULL,
  show_axis_labels = TRUE
)
```

Arguments

se	A SummarizedExperiment with Combined_Class__<group> columns.
ref_group	Character. Reference condition (rows).
target_group	Character. Treatment / target condition (columns).
normalize	Character. "row_percent" (default), "count", or "none".
include_uncertain	Logical. Include Uncertain class (default TRUE).
mark	Character or NULL. Histone mark preset for display terminology.
ref_role	Character or NULL. Optional semantic role label appended to the reference group name in the subtitle (e.g. "control"), so the plot states unambiguously which group is the reference. Default NULL.
target_role	Character or NULL. Optional semantic role label appended to the target group name in the subtitle (e.g. "tumor"). Default NULL.
show_axis_labels	Logical. Show the class names on the heatmap margins (default TRUE). The axis lines and ticks are always removed, so the plot reads as a clean heatmap that keeps the row/column class identity; set FALSE for a label-free heatmap.

Value

A ggplot.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
se <- call_super_domains(se, feature = "Breadth",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
se <- combine_superdomain_calls(se, group_var = "Condition")
plot_class_transition(se, ref_group = "Control", target_group = "Treatment")
```

```
plot_differential_volcano
```

Differential Domain Volcano Plot

Description

Volcano plot of a differential-domain analysis performed by [analyze_differential_domains](#), using the stored DiffStatus labels with the package's publication palette. Domains are coloured by Gain / Loss / NS (grey).

Usage

```
plot_differential_volcano(
  se,
  feature = "Intensity",
  logFC_col = NULL,
  padj_col = NULL,
  label_n = 0
)
```

Arguments

se	A SummarizedExperiment after <code>analyze_differential_domains()</code> .
feature	Character. Differential feature column (default "Intensity").
logFC_col, padj_col	Character. Column names for logFC and adjusted P-value (default constructed from feature).
label_n	Integer. Number of top domains to label with <code>top_candidate_gene_symbol</code> / gene symbol if available (default 0).

Value

A ggplot object.

Examples

```
data(example_se)
if (requireNamespace("limma", quietly = TRUE)) {
  se <- analyze_differential_domains(
    example_se, group_var = "Condition",
    ref_group = "Control", target_group = "Treatment")
  plot_differential_volcano(se, label_n = 5)
}
```

plot_domain_class_composition

Plot Domain Class Composition

Description

100% stacked bar of the combined architecture class composition per condition group.

Usage

```
plot_domain_class_composition(
  se,
  group_var = "Condition",
  mark = NULL,
  focus = c("super", "all"),
  show_counts = TRUE
)
```

Arguments

se	A SummarizedExperiment with Combined_Class__<group> columns.
group_var	Character. colData column for groups.
mark	Character or NULL. Histone mark preset for display terminology.
focus	Character. "super" (default): restrict the 100% stacked bar to the super classes (Intensity/Breadth/Dual) so the composition is readable when non-super domains dominate the total, and annotate the super share of all domains above each bar. "all": full composition including Typical / Uncertain.
show_counts	Logical. Draw absolute domain counts inside each segment (default TRUE).

Value

A ggplot.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
se <- call_super_domains(se, feature = "Breadth",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
se <- combine_superdomain_calls(se, group_var = "Condition")
plot_domain_class_composition(se)
```

```
plot_domain_feature_profile
```

Plot Raw Domain Feature Profile

Description

Quantitative per-domain profile across conditions: biological replicates as points and group means as larger open points, faceted by feature. A dashed line connects the group means so the across-condition direction of change is read directly. When combined architecture classes are present (`combine_superdomain_calls`), the domain's class in each condition is annotated under the axis, linking the quantitative values to its architectural identity. Replaces the qualitative radar as the quantitative view.

Usage

```
plot_domain_feature_profile(
  se,
  peak_id,
  group_var = "Condition",
  features = NULL,
  transform = c("log10p1", "none"),
  annotate_class = TRUE,
  mark = NULL
)
```

Arguments

<code>se</code>	A SummarizedExperiment.
<code>peak_id</code>	Character. Domain ID.
<code>group_var</code>	Character. colData column for groups.
<code>features</code>	Character vector of assays to show (default the three dynamic assays).
<code>transform</code>	Character. "log10p1" or "none".
<code>annotate_class</code>	Logical. Annotate the domain's combined architecture class under each condition group (default TRUE; silently skipped when the combined-class columns are absent).
<code>mark</code>	Character or NULL. Histone mark preset for class display terminology.

Value

A ggplot.

Examples

```
data(example_se)
plot_domain_feature_profile(example_se, peak_id = rownames(example_se)[1])
```

plot_domain_landscape *Plot Intensity x Breadth Architecture Landscape*

Description

Signature figure: each domain as a point in the Intensity x Breadth plane, coloured by combined architecture class. This is the visual that distinguishes epiPortrait from single-intensity callers (Intensity-extreme vs breadth-extreme vs dual).

Usage

```
plot_domain_landscape(
  se,
  group = NULL,
  transform = c("log10p1", "none"),
  label_n = 0,
  label_col = NULL,
  label_mode = c("per_class", "top_intensity"),
  x_feature = "Intensity",
  y_feature = "NativeMaxPeakWidth",
  mark = NULL,
  group_var = "Condition",
  show_counts = TRUE
)
```

Arguments

se	A SummarizedExperiment with combined class columns (combine_superdomain_calls).
group	Character or NULL. Condition group whose combined class to colour by (NULL uses the consensus Combined_Domain_Class).
transform	Character. "log10p1" (default) or "none" (display only; classification is never recomputed).
label_n	Integer. Number of representative domains to label per super class (for label_mode = "per_class") or number of top-intensity domains (for label_mode = "top_intensity"). Default 0 = no domain labels (the architecture scatter is the message; labels are opt-in via label_n > 0).
label_col	Character. Column in rowData used for labels (default "SYMBOL" if present, else Domain_ID).

label_mode	Character. "per_class" (default): label the most representative (prototype) domains of each super class — top intensity within Intensity-Super, top breadth within Breadth-Super, top combined rank within Dual-Super — so each architecture class is named in the figure. "top_intensity": label the globally top-intensity domains (legacy behaviour).
x_feature	Character. Assay for the x axis (default "Intensity").
y_feature	Character. Assay for the y axis (default "NativeMaxPeakWidth", the domain-level native breadth summary; falls back to "NativeOccupiedWidth" then "SignalDispersion" when unavailable).
mark	Character or NULL. Histone mark preset for display terminology (e.g. "H3K27me3" renders Extended-Domain / Dual-Extreme labels). NULL uses the canonical Intensity-Super / Breadth-Super / Dual-Super labels.
group_var	Character. colData column used for grouping when group is provided (default "Condition"). Must match the group_var used by call_super_domains(mode = "per_group") / combine_superdomain_calls() for the requested group.
show_counts	Logical. Append the per-class domain count to the legend labels, e.g. "Intensity-Super (n=607)" (default TRUE).

Value

A ggplot.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
se <- call_super_domains(se, feature = "Breadth",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
se <- combine_superdomain_calls(se, group_var = "Condition")
plot_domain_landscape(se, group = "Control")
```

plot_hockey_stick	<i>Plot Hockey Stick Curve for Super-Domain Identification</i>
-------------------	--

Description

Generates a ranking plot for super-domain identification. Domains are ranked by their feature value; super-domains are highlighted and the cutoff (and, when available, the replicate-cutoff stability band) is drawn from the object's stored provenance. In per-group mode the cutoff is shown as a descriptive median replicate cutoff, never as the unique biological decision boundary.

Usage

```
plot_hockey_stick(
  se,
  feature = "Intensity",
  label_genes = FALSE,
  top_n_label = 10,
  group = NULL,
  group_var = "Condition",
  show_cutoff = TRUE,
  show_cutoff_band = TRUE
)
```

Arguments

<code>se</code>	A SummarizedExperiment object after running <code>call_super_domains</code> .
<code>feature</code>	Character. Which feature to plot ("Intensity", "SignalDispersion", or "Breadth"; legacy aliases accepted). For "Breadth" the plot shows the replicate-specific native PeakWidth rank curve from <code>metadata(se)\$breadth_peak_calls</code> , and <code>group</code> must be a sample (replicate) name.
<code>label_genes</code>	Logical. Label top super-domains (default FALSE; the ranking curve and cutoff are the message, per-domain labels are opt-in).
<code>top_n_label</code>	Integer. Number of top domains to label (default: 10).
<code>group</code>	Character or NULL. For signal features: a condition group (per-group call/rank columns <code><feature>_Call__<group></code> , descriptive median replicate cutoff), OR a single sample name when the object was called with <code>mode = "per_sample"</code> — the sample's own ranked distribution is plotted with its per-sample cutoff. For <code>feature = "Breadth"</code> : the replicate whose native PeakWidth distribution to plot (required). NULL (default) uses the consensus <code><feature>_Domain_Type</code> for signal features.
<code>group_var</code>	Character. colData column used for groups when <code>group</code> is provided (default "Condition").
<code>show_cutoff</code>	Logical. Draw the cutoff as a dashed horizontal guide (default TRUE).
<code>show_cutoff_band</code>	Logical. Draw a light horizontal band for the replicate cutoff stability interval (default TRUE).

Value

A ggplot object.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
  method = "tangent", log_transform = FALSE)
plot_hockey_stick(se, feature = "Intensity", label_genes = FALSE)
```

plot_peak_track	<i>Plot Physical Coverage Profile for a Specific Peak (Track View)</i>
-----------------	--

Description

Generates a track-like visualization of the raw BigWig coverage for a specific domain across all samples, for direct visual validation.

Usage

```
plot_peak_track(
  se,
  peak_id,
  group_var = "Condition",
  extend_bp = 1000,
  smooth_window = 1,
  free_y = FALSE,
  show_native_occupancy = FALSE
)
```

Arguments

se	A SummarizedExperiment object from build_portrait_matrix().
peak_id	Character. The ID of the peak to plot.
group_var	Character. The grouping variable in colData(se).
extend_bp	Numeric. Number of base pairs to extend the view around the peak (default: 1000).
smooth_window	Numeric. Window size for smoothing the coverage (default: 1, no smoothing).
free_y	Logical. Use independent y-scales per facet panel (default FALSE = fixed y, so cross-sample intensity differences are visually preserved rather than auto-scaled away).
show_native_occupancy	Logical. Overlay the per-sample NativeOccupiedWidth (sum of reduced native peak widths inside the domain) as horizontal bands (default FALSE). The band is centred on the candidate domain and is a per-sample geometry view, not a new biological call.

Value

A ggplot object.

Examples

```
if (.Platform$OS.type != "windows") {
  extdata <- system.file("extdata", package = "epiPortrait")
  samples <- data.frame(
    SampleID = c("C1", "C2", "T1", "T2"),
    Condition = c("Control", "Control", "Treatment", "Treatment"),
    bw_path = file.path(extdata, c("C1.bw", "C2.bw", "T1.bw", "T2.bw"))
  )
  se_tiny <- build_portrait_matrix(
    samples, consensus_peaks = rtracklayer::import(file.path(extdata, "peaks.bed")),
    workers = 1)
  plot_peak_track(se_tiny, peak_id = rownames(se_tiny)[1],
    group_var = "Condition")
}
```

plot_portrait_correlation

Sample Correlation Heatmap for epiPortrait Features

Description

Computes and visualizes the Spearman (default, robust to heavy-tailed epigenomic signal) or Pearson correlation between samples.

Usage

```
plot_portrait_correlation(
  se,
  feature = "Intensity",
  method = c("spearman", "pearson"),
  limits = NULL,
  dynamic_mid = TRUE
)
```

Arguments

<code>se</code>	A SummarizedExperiment object.
<code>feature</code>	Character. The assay to use (default: "Intensity").
<code>method</code>	Character. "spearman" (default) or "pearson".
<code>limits</code>	Numeric of length 2 or NULL. Colour-scale limits for the correlation. Default NULL follows the observed data range so that within/between-group differences remain visible even when all samples are highly correlated (typical for biological replicates); when the observed range does not span 0, the colour midpoint is the range centre (see <code>dynamic_mid</code>). Pass an explicit fixed scale (e.g. <code>c(-1, 1)</code>) to keep a statistically anchored, cross-dataset-comparable scale with 0 = unrelated as the semantic midpoint.

dynamic_mid Logical. When `limits = NULL`, whether to centre the colour scale on the observed data midpoint (TRUE, default) instead of the fixed 0 anchor. Centring on the data maximises visual contrast between samples; centring on 0 keeps the "unrelated" semantics but compresses the palette when all correlations are high.

Value

A ggplot object.

Examples

```
data(example_se)
plot_portrait_correlation(example_se, feature = "Intensity")
plot_portrait_correlation(example_se, feature = "Intensity", limits = c(-1, 1))
```

plot_portrait_pca	<i>Exploratory PCA Plot for epiPortrait Features</i>
-------------------	--

Description

Performs Principal Component Analysis (PCA) on a specific quantitative domain feature (e.g., Intensity or SignalDispersion) to visualize sample clustering and batch effects.

Usage

```
plot_portrait_pca(
  se,
  feature = "Intensity",
  group_var = "Condition",
  n_top = 500,
  features = NULL
)
```

Arguments

se	A SummarizedExperiment object.
feature	Character. The assay to use for PCA (default: "Intensity").
group_var	Character. The grouping variable for coloring points.
n_top	Numeric. Use only the top N most variable peaks (default: 500).
features	Character vector. Additional assays to include in a combined-feature PCA (e.g. <code>c("Intensity", "SignalDispersion")</code>). The top-variable domains of each feature are concatenated into the PCA variable set. Default NULL uses feature only.

Value

A ggplot object.

Examples

```
data(example_se)
plot_portrait_pca(example_se, feature = "Intensity", group_var = "Condition")
```

```
plot_replicate_support
```

Plot Replicate-Support Distribution

Description

Bar plot of the fraction of domains with each replicate-support level (0/k, 1/k, ..., k/k), stacked by the final group call. Demonstrates replicate-aware calling and the effect of the support rule.

Usage

```
plot_replicate_support(
  se,
  feature = "Intensity",
  group = NULL,
  group_var = "Condition",
  min_support = 1,
  show_counts = TRUE
)
```

Arguments

<code>se</code>	A SummarizedExperiment after <code>call_super_domains</code> per_group.
<code>feature</code>	Character. Feature (default "Intensity").
<code>group</code>	Character. Condition group (default first group).
<code>group_var</code>	Character. colData column for groups.
<code>min_support</code>	Integer. Drop support levels with fewer than this many supporting replicates. The 0/k bin (domains never super in any replicate) usually dominates the bar and is grey; set <code>min_support = 1</code> (default) to zoom into the replicate-aware distribution.
<code>show_counts</code>	Logical. Draw the domain count above each bar (default TRUE).

Value

A ggplot.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_group", group_var = "Condition",
                        verbose = FALSE)
plot_replicate_support(se, feature = "Intensity", group = "Control")
```

plot_uncertain_cause *Plot the Cause Composition of Uncertain Calls*

Description

Bar chart of why domains are Uncertain for a feature's per-group call (see `get_uncertain_cause`): no native peak in the condition / insufficient valid replicates / inflection no-call. Helps audit whether the Uncertain fraction reflects condition-specific loci rather than analysis artifacts.

Usage

```
plot_uncertain_cause(  
  se,  
  feature = "Breadth",  
  group,  
  group_var = NULL,  
  show_counts = TRUE  
)
```

Arguments

se	A SummarizedExperiment after <code>call_super_domains(mode = "per_group")</code> .
feature	Character. Feature (default "Breadth").
group	Character. Condition group.
group_var	Character or NULL. Grouping column (resolved from stored provenance when NULL).
show_counts	Logical. Label bars with counts (default TRUE).

Value

A ggplot.

Examples

```
data(example_se)  
se <- call_super_domains(example_se, feature = "Breadth",  
                        mode = "per_group", group_var = "Condition",  
                        verbose = FALSE)  
plot_uncertain_cause(se, feature = "Breadth", group = "Control")
```

```
save_epiportrait_figure
```

Save an epiPortrait Figure in Vector or Raster Format

Description

Save an epiPortrait Figure in Vector or Raster Format

Usage

```
save_epiportrait_figure(p, file, width = 3.5, height = 3.3, dpi = 600, ...)
```

Arguments

p	A ggplot.
file	Output path (extension determines format: .pdf, .svg, .png, .tiff).
width, height	Numeric in inches.
dpi	Numeric. For raster output (default 600).
...	Passed to the underlying ggsave / ragg device.

Value

Invisibly the file path.

Examples

```
data(example_se)
p <- plot_portrait_pca(example_se, feature = "Intensity")
save_epiportrait_figure(p, file = tempfile(fileext = ".png"))
```

```
stitch_epi_peaks
```

Stitch Proximal Epigenetic Peaks

Description

Merges closely spaced peaks into larger continuous macro-domains based on a user-defined stitching distance. This is crucial for analyzing Super-Elements or broad heterochromatin spreading in epiPortrait.

Usage

```
stitch_epi_peaks(gr, stitch_distance = 12500)
```

Arguments

gr A GRanges object of peaks (typically the output from `get_consensus_peaks`).

stitch_distance Numeric. Maximum distance (in base pairs) between peaks to be stitched together. For Super-Elements, 12500 (12.5 kb) is standard. For standard broad marks, you might use 3000 to 5000.

Value

A GRanges object of stitched domains, with a metadata column 'Constituent_Peaks' recording how many original peaks were merged into each domain.

Examples

```
gr <- GenomicRanges::GRanges("chr1",
  IRanges::IRanges(start = c(100, 10000), end = c(200, 10100)))
stitch_epi_peaks(gr, stitch_distance = 12500)
```

summarize_epiportrait_object

Summarize an epiPortrait Object

Description

Summarize an epiPortrait Object

Usage

```
summarize_epiportrait_object(se)
```

Arguments

se A SummarizedExperiment.

Value

A list with a short text summary of the object contents.

Examples

```
data(example_se)
summarize_epiportrait_object(example_se)
```

transition_matrix_per_sample

Pairwise Per-Sample Transition Table Across Time Points

Description

Computes super-domain / class transitions between per-sample call columns (mode "per_sample", one sample per time point), which `compare_superdomains()` / `compare_superdomain_classes()` do not support (they require per-GROUP columns). Adds one transition column per (ref, target) time-point pair and a labelled counts table to `metadata(se)$transitions`.

Usage

```
transition_matrix_per_sample(
  se,
  feature = NULL,
  call_fmt = NULL,
  timepoints = NULL,
  ref = NULL,
  verbose = TRUE
)
```

Arguments

se	A SummarizedExperiment carrying per-sample call columns.
feature	Character or NULL. Feature prefix for the call columns (e.g. "Breadth", "Intensity", or "Combined_Class"). Ignored when <code>call_fmt</code> is given.
call_fmt	Character or NULL. <code>sprintf</code> format of the per-sample call columns, e.g. "Breadth_Call_%%s". Defaults to <code>sprintf("%s_Call_%%s", feature, "%s")</code> .
timepoints	Character or NULL. Ordered time points. When NULL, derived from the call column suffixes, preserving their discovery order (use <code>colData\$TimePoint</code> order when present).
ref	Character or NULL. Reference time point for transitions; when given, all other time points are targets. When NULL, all ordered pairs (a, b) with a before b are computed.
verbose	Logical. Print the transition counts (default TRUE).

Value

se with rowData columns `<prefix>_SampleTransition__<ref>_vs_<target>` (labels: `Persistent_<value>`, `<from>_to_<to>`, `Uncertain`, matching `compare_superdomain_classes()` semantics) and `metadata(se)$transitions` entries with the counts tables.

Examples

```
data(example_se)
se <- call_super_domains(example_se, feature = "Intensity",
                        mode = "per_sample", verbose = FALSE)
se <- transition_matrix_per_sample(se, feature = "Intensity",
                                verbose = FALSE)
SummarizedExperiment::rowData(se)$Intensity_SampleTransition__C1_vs_T1
```

`validate_epiportrait_object`*Validate an epiPortrait SummarizedExperiment Object*

Description

Verifies the internal consistency invariants of an epiPortrait object: unique row names, alignment between assays and rowRanges/colData, IntervalWidth matching rowRanges widths, presence of required assays, and consistent per-group class/support columns.

Usage

```
validate_epiportrait_object(se)
```

Arguments

`se` A SummarizedExperiment from `build_portrait_matrix()`.

Value

Invisibly TRUE if all checks pass; otherwise stops with the first failing invariant.

Examples

```
data(example_se)
validate_epiportrait_object(example_se)
```

Index

* datasets

- example_se, 20
- analyze_differential_domains, 3, 37
- annotate_epi_domains, 5, 19
- build_portrait_matrix, 8, 19
- call_super_domains, 10, 19
- check_signal_compatibility, 13
- combine_superdomain_calls, 14, 19
- compare_superdomain_classes, 15
- compare_superdomains, 16, 19
- compute_width_transition, 17
- epiPortrait, 19
- epiPortrait-package (epiPortrait), 19
- example_se, 20
- export_epiportrait_results, 21
- filter_blacklist, 19, 22
- filter_promoter_peaks, 19, 22
- find_hockey_inflection, 23
- get_call_provenance, 24
- get_call_results, 25
- get_combined_class_results, 26
- get_consensus_peaks, 19, 26
- get_domain_genes, 27
- get_domain_results, 29
- get_mark_preset, 29
- get_replicate_calls, 30
- get_sample_results, 31
- get_transition_results, 32
- get_uncertain_cause, 32
- integrate_cross_mark, 33
- normalize_portrait, 19, 35
- plot_class_transition, 36

- plot_differential_volcano, 37
- plot_domain_class_composition, 38
- plot_domain_feature_profile, 39
- plot_domain_landscape, 40
- plot_hockey_stick, 19, 41
- plot_peak_track, 19, 43
- plot_portrait_correlation, 19, 44
- plot_portrait_pca, 19, 45
- plot_replicate_support, 46
- plot_uncertain_cause, 47
- save_epiportrait_figure, 48
- stitch_epi_peaks, 19, 48
- summarize_epiportrait_object, 49
- transition_matrix_per_sample, 50
- validate_epiportrait_object, 51