

Package: TweedSpot (via r-universe)

October 14, 2026

Title Self-Adaptive Spatial Tweedie Models for Spatially Variable Gene Detection

Version 0.99.3

Description TweedSpot detects spatially variable genes (SVGs) in spatially resolved transcriptomics data using self-adaptive Tweedie generalized additive models with nonparametric spatial smoothing. The method models raw counts directly, avoiding the bias introduced by log-transformation, adapts a gene-specific Tweedie index parameter to the mean-variance relationship of each gene, and offers an optional two-part formulation to accommodate excess zeros. Inputs and outputs use the SpatialExperiment infrastructure for interoperability with the Bioconductor ecosystem.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.5.0)

Imports mgcv, MatrixGenerics, scran, SpatialExperiment, SummarizedExperiment, S4Vectors, BiocParallel, methods, stats, withr

Suggests SpaNorm, STexampleData, testthat (>= 3.0.0), ggplot2, patchwork, svglite, Matrix, DelayedArray, knitr, rmarkdown, BiocStyle

biocViews Software, Transcriptomics, GeneExpression, RiboSeq, Spatial, Regression, StatisticalMethod, DifferentialExpression, QualityControl

URL <https://github.com/mallick-lab/TweedSpot>

BugReports <https://github.com/mallick-lab/TweedSpot/issues>

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

Config/pak/sysreqs
libglpk-dev libmagick++-dev gsfonts libicu-dev libxml2-dev libssl-dev zlib1g-dev
Repository https://biocstaging.r-universe.dev
Date/Publication 2026-10-14 16:02:04 UTC
RemoteUrl https://github.com/BiocStaging/TweedSpot
RemoteRef HEAD
RemoteSha 7cdedd3c7e9fae4b8fb711b4d6a137a4c2796b8d

Contents

filter_features	2
get_tweedspot_results	5
plot_feature	7
plot_top_features	9
top_spatial_genes	11
tweedspot	12
Index	17

filter_features	<i>Filter low-information features and low-quality locations</i>
-----------------	--

Description

filter_features() prepares a spatial dataset for tweedspot(). It removes (or flags) features with too little signal to model, features that cannot support the spatial smooth TweedSpot will fit, named features such as control probes, and low-quality spatial locations. All criteria are modality agnostic and apply to genes, proteins, peaks, or metabolites, and to spots, cells, or pixels; mitochondrial and ribosomal gene removal is available as a transcriptomics-specific option.

Usage

```
filter_features(  
  input,  
  assay_name = "counts",  
  min_total = 5,  
  min_pct_locations = 5,  
  min_locations = 0,  
  min_mean = 0,  
  min_var = 0,  
  detection_threshold = 0,  
  model_aware = TRUE,  
  smooth_k = NULL,  
  group = NULL,  
  min_group_size = 50,  
  two_part = FALSE,
```

```

exclude = NULL,
exclude_pattern = NULL,
feature_name_col = NULL,
exclude_mito = FALSE,
exclude_ribo = FALSE,
feature_type_col = NULL,
min_total_per_location = 0,
min_features_per_location = 0,
outlier_nmds = NULL,
max_excluded_fraction = NULL,
location_mask = NULL,
drop_empty_locations = TRUE,
action = c("drop", "flag"),
report = FALSE
)

```

Arguments

input	A SpatialExperiment::SpatialExperiment with a non-negative feature-by-location assay.
assay_name	Name of the assay to filter. Default "counts".
min_total	Minimum total signal (sum across locations) for a feature. Default 5.
min_pct_locations	Minimum percent of locations in which the feature is detected. Default 5.
min_locations	Minimum number of locations in which the feature is detected. Default 0.
min_mean	Minimum mean signal across locations. Default 0.
min_var	Minimum variance across locations. Default 0.
detection_threshold	A feature counts as detected at a location when its value is strictly greater than this. Default 0, which suits counts; raise it for intensity data with a background floor.
model_aware	Logical; require each feature to be detected in at least as many locations as the spatial smooth has basis functions (see Details). Default TRUE.
smooth_k	Basis dimension of the spatial smooth, as passed to tweedspot() . Default NULL uses TweedSpot's default.
group	Optional name of a discrete colData(input) column, as passed to tweedspot(group = ...). The model-aware minimum is then checked within each group (of at least min_group_size locations), and a feature is removed only if no group can support it. Default NULL.
min_group_size	Groups with fewer locations than this are ignored when group is given. Default 50.
two_part	Logical; whether you will fit tweedspot() with two_part = TRUE. Only affects reporting of features whose presence/absence part will be skipped. Default FALSE.
exclude	Optional character vector of feature names to remove, matched exactly against row names and, if supplied, feature_name_col.

exclude_pattern	Optional character vector of regular expressions; features whose row name or feature_name_col value matches any of them are removed, for example "^NegPrb".
feature_name_col	Optional rowData(input) column holding readable feature names, used by the exclusion arguments.
exclude_mito	Logical; transcriptomics only. Remove mitochondrial genes (names beginning MT-/mt-, or a feature_type_col value containing "mitochond"). Default FALSE.
exclude_ribo	Logical; transcriptomics only. Remove ribosomal protein genes (RPL10A, RPLP0, RPSA, MRPL12, ..., but not the RPS6K kinases), or a feature_type_col value containing "ribosom". Default FALSE.
feature_type_col	Optional rowData(input) column of feature categories (for example gene bio-types) used by exclude_mito and exclude_ribo.
min_total_per_location	Minimum total signal for a location. Default 0.
min_features_per_location	Minimum number of features detected at a location. Default 0.
outlier_nmads	Optional number of median absolute deviations; a location is removed when its log total signal or log number of detected features falls more than this many MADs below the median. Default NULL (off); 3 is a common choice.
max_excluded_fraction	Optional maximum fraction of a location's signal that may come from excluded features (for example 0.2 with exclude_mito = TRUE for a mitochondrial-fraction filter, or with exclude_pattern for control-probe signal). Default NULL (off).
location_mask	Optional name of a logical colData(input) column; locations where it is FALSE are removed, for example "in_tissue".
drop_empty_locations	Logical; remove locations with no signal after feature filtering. Default TRUE.
action	"drop" (default) returns only retained features and locations. "flag" returns every feature and location, with filter_keep (logical) and filter_reason (failed criteria separated by ";", empty when kept) added to rowData() and colData().
report	Logical; print how many features and locations were kept and how many failed each criterion. Default FALSE.

Details

Filtering happens in this order:

1. **Named exclusions** (exclude, exclude_pattern, exclude_mito, exclude_ribo) mark features to remove.
2. **Location QC** is computed on the remaining features: minimum total signal, minimum detected features, lower-tail outliers, the fraction of signal from excluded features, and an optional mask.

3. **Feature thresholds** are computed on the locations that pass QC, including the model-aware minimum below.
4. Locations with no remaining signal are removed (drop_empty_locations).

Model-aware minimum. With model_aware = TRUE, a feature must be detected in at least as many locations as the spatial smooth has basis functions (smooth_k, or TweedSpot's default of up to 30). Features below this cannot support the smooth and would otherwise fail or give unstable fits. Pass the same smooth_k you will give tweedspot(). With group, the minimum is checked within each group using that group's locations, and a feature is kept if at least one group can support it; the per-group result is stored in metadata(output)\$filter_features\$group_support. For the two-part model the presence/absence part additionally needs that many detected *and* undetected locations; features that fail only this are kept, and tweedspot() then tests them with the count part alone.

All feature criteria depend only on abundance and detection, not on spatial arrangement, so filtering does not bias the spatial test's p-values.

Value

A SpatialExperiment. With action = "drop", only retained features and locations; with action = "flag", all of them with filter_keep and filter_reason columns. In both cases S4Vectors::metadata(output)\$filter_reasons records the settings and the number of features and locations failing each criterion.

Examples

```
set.seed(1)
counts <- matrix(stats::rpois(5 * 36, 4), nrow = 5)
counts[2, ] <- 0
counts[2, 1:2] <- 3
rownames(counts) <- c("GENE1", "RARE1", "MT-CO1", "NegPrb1", "GENE5")
colnames(counts) <- paste0("spot", seq_len(ncol(counts)))
spe <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = counts),
  spatialCoords = as.matrix(expand.grid(x = 1:6, y = 1:6))
)
# Drop features, using a smooth with 10 basis functions
filter_features(spe, smooth_k = 10, exclude_pattern = "^NegPrb")
# Keep everything but record why each feature or location would be removed
flagged <- filter_features(spe, smooth_k = 10, exclude_mito = TRUE,
  max_excluded_fraction = 0.3, action = "flag")
SummarizedExperiment::rowData(flagged)[, c("filter_keep", "filter_reason")]
```

get_tweedspot_results *Extract TweedSpot results as a tidy data frame*

Description

get_tweedspot_results() pulls the TweedSpot result columns from rowData(input) into a base data.frame and can optionally sort the output.

Usage

```
get_tweedspot_results(
  input,
  sort_by = c("padj", "none", "pval", "stat", "dev_expl"),
  decreasing = NULL,
  group = NULL
)
```

Arguments

input	A SpatialExperiment::SpatialExperiment containing TweedSpot results in <code>rowData(input)</code> .
sort_by	Column to sort by. Options are "none", "padj", "pval", "stat", or "dev_expl". Default "padj".
decreasing	Logical; override the default sort direction for <code>sort_by</code> . If NULL, p-values sort ascending and statistics/effect-size columns sort descending.
group	For results of <code>tweedspot(group = ...)</code> : NULL (default) returns the usual per-feature <code>rowData()</code> results; TRUE returns the full feature-by-group table; one or more group labels return only those groups' rows.

Value

A data frame. Without group, one row per feature with columns feature (row name), feature_name (when `rowData()` has a feature_name or gene_name column that differs from the row names), stat (test statistic of the spatial smooth), pval, padj, edf (effective degrees of freedom of the smooth), dev_expl (deviance explained), and, when available, p (Tweedie variance power) and family. With group, one row per feature and group with columns feature, group, n_locations, n_detected, tested, reason (why a pair was not tested), stat, pval, padj, padj_feature (stage-wise screening p-value), edf, dev_expl, p, and family.

Examples

```
counts <- matrix(c(1, 2, 3, 4, 2, 2), nrow = 2, byrow = TRUE)
rownames(counts) <- c("gene1", "gene2")
colnames(counts) <- paste0("spot", seq_len(ncol(counts)))
spe <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = counts),
  rowData = S4Vectors::DataFrame(
    gene_id = rownames(counts),
    gene_name = rownames(counts),
    tweedspot_stat = c(4, 2),
    tweedspot_pval = c(0.01, 0.05),
    tweedspot_padj = c(0.02, 0.05),
    tweedspot_edf = c(2, 2),
    tweedspot_dev_expl = c(0.3, 0.1)
  ),
  colData = S4Vectors::DataFrame(sample_id = rep("sample1", ncol(counts))),
  spatialCoords = cbind(x = seq_len(ncol(counts)), y = seq_len(ncol(counts)))
)
get_tweedspot_results(spe)
```

plot_feature	<i>Plot one feature: observed signal, TweedSpot fit, and spatial effect</i>
--------------	---

Description

`plot_feature()` refits the `TweedSpot` model for one feature and draws three aligned panels: the observed normalized signal, the fitted mean from the `TweedSpot` model on the same colour scale, and the fitted spatial effect as a $\log_2$ fold change relative to the feature's average. The model is refitted with the settings stored by `tweedspot()` in `S4Vectors::metadata(input)$tweedspot`, so the figure shows exactly the model that was tested; any of those settings can be overridden through

Usage

```
plot_feature(
  input,
  feature,
  group = NULL,
  transform = c("log1p", "none"),
  flip_y = TRUE,
  point_size = NULL,
  base_size = 8,
  base_family = "",
  output_dir = NULL,
  formats = "pdf",
  width = 7.2,
  height = 2.9,
  dpi = 300,
  ...
)
```

Arguments

<code>input</code>	A <code>SpatialExperiment::SpatialExperiment</code> , typically the output of <code>tweedspot()</code> .
<code>feature</code>	A single row index, row name, or value of <code>rowData(input)\$gene_name / rowData(input)\$feature_name</code> .
<code>group</code>	For results of <code>tweedspot(group = ...)</code> , the group (for example the cell type) to plot. The model is then refitted on that group's locations only, as it was tested, and only those locations are drawn. Required for group results and not allowed otherwise. Default <code>NULL</code> .
<code>transform</code>	Display transform for the observed and fitted panels: <code>"log1p"</code> (default) or <code>"none"</code> . The spatial-effect panel is always on the $\log_2$ scale.
<code>flip_y</code>	Logical; reverse the y axis so that images with the origin at the top-left are drawn upright. Default <code>TRUE</code> .
<code>point_size</code>	Size of the location markers. Default <code>NULL</code> chooses a size from the spacing between locations and the figure size so that spots tile without overlapping.

base_size	Base font size in points. Default 8, suited to a full-width journal figure.
base_family	Font family. Default "" (the device's sans-serif).
output_dir	Optional directory in which to save the figure. It is created if it does not exist. Default NULL (do not save).
formats	File formats to write when output_dir is given; any of "pdf", "png", "svg", "tiff". Default "pdf".
width, height	Figure size in inches, used for saving and for choosing point_size. Defaults 7.2 by 2.9, a two-column journal width.
dpi	Resolution for raster formats. Default 300.
...	Model settings that override those stored by <code>tweedspot()</code> : assay_name, covariates, family, tweedie_p, p_criterion, fit_method, use_bam, bam_discrete, smooth_k, size_factor_method, spanorm_output, spanorm_args, seed. Unless the normalization is overridden, the refit uses the size factors the run used.

Details

For a two-part run (`two_part = TRUE`) the panels show the count (abundance) part of the model.

Value

A patchwork object. When `output_dir` is supplied, the paths of the written files are attached as `attr("files")`.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE) &&
    requireNamespace("patchwork", quietly = TRUE)) {
  set.seed(1)
  grid <- expand.grid(x = 1:12, y = 1:12)
  counts <- rbind(
    band = stats::rpois(144, ifelse(grid$y > 6, 12, 2)),
    flat = stats::rpois(144, 5)
  )
  colnames(counts) <- paste0("spot", seq_len(ncol(counts)))
  spe <- SpatialExperiment::SpatialExperiment(
    assays = list(counts = counts),
    spatialCoords = as.matrix(grid)
  )
  spe <- tweedspot(spe, smooth_k = 10, size_factor_method = "library_size",
    BPPARAM = BiocParallel::SerialParam(), verbose = FALSE)
  plot_feature(spe, "band")
}
```

plot_top_features	<i>Plot the spatial effects of the top spatially variable features</i>
-------------------	--

Description

`plot_top_features()` draws the fitted spatial effect of the most significant features from a `tweedspot()` run as small multiples on one shared diverging scale, so that the strength of different patterns can be compared directly. Each panel is labelled with the feature name and, by default, its adjusted p-value and deviance explained.

Usage

```
plot_top_features(
  input,
  n_top = 6,
  features = NULL,
  group = NULL,
  ncol = NULL,
  label_stats = TRUE,
  title = NULL,
  flip_y = TRUE,
  point_size = NULL,
  base_size = 8,
  base_family = "",
  output_dir = NULL,
  formats = "pdf",
  width = 7.2,
  height = NULL,
  dpi = 300,
  ...
)
```

Arguments

<code>input</code>	A <code>SpatialExperiment::SpatialExperiment</code> , typically the output of <code>tweedspot()</code> .
<code>n_top</code>	Number of top features to show, ranked by adjusted p-value and then by test statistic. Default 6.
<code>features</code>	Optional vector of features (row indices, row names, or feature names) to show instead of the top <code>n_top</code> , in the given order.
<code>group</code>	For results of <code>tweedspot(group = ...)</code> , the group to plot: features are ranked by that group's adjusted p-values and refitted on that group's locations. Required for group results and not allowed otherwise. When given and <code>title</code> is <code>NULL</code> , the group is used as the title.
<code>ncol</code>	Number of panel columns. Default <code>NULL</code> uses 3 for up to nine panels and 4 otherwise.

label_stats	Logical; add the adjusted p-value and deviance explained under each feature name. Default TRUE.
title	Optional figure title. Default NULL (none).
flip_y	Logical; reverse the y axis so that images with the origin at the top-left are drawn upright. Default TRUE.
point_size	Size of the location markers. Default NULL chooses a size from the spacing between locations and the figure size so that spots tile without overlapping.
base_size	Base font size in points. Default 8, suited to a full-width journal figure.
base_family	Font family. Default "" (the device's sans-serif).
output_dir	Optional directory in which to save the figure. It is created if it does not exist. Default NULL (do not save).
formats	File formats to write when output_dir is given; any of "pdf", "png", "svg", "tiff". Default "pdf".
width	Figure width in inches, used for saving and for choosing point_size. Default 7.2, a two-column journal width.
height	Figure height in inches. Default NULL derives it from the number of rows and the tissue's aspect ratio.
dpi	Resolution for raster formats. Default 300.
...	Model settings that override those stored by <code>tweedspot()</code> : assay_name, covariates, family, tweedie_p, p_criterion, fit_method, use_bam, bam_discrete, smooth_k, size_factor_method, spanorm_output, spanorm_args, seed. Unless the normalization is overridden, the refit uses the size factors the run used.

Value

A ggplot object. When output_dir is supplied, the paths of the written files are attached as `attr("files")`.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  set.seed(1)
  grid <- expand.grid(x = 1:12, y = 1:12)
  counts <- rbind(
    band = stats::rpois(144, ifelse(grid$y > 6, 12, 2)),
    spot = stats::rpois(144, 2 + 10 * exp(-((grid$x - 4)^2 + (grid$y - 4)^2) / 6)),
    flat = stats::rpois(144, 5)
  )
  colnames(counts) <- paste0("spot", seq_len(ncol(counts)))
  spe <- SpatialExperiment::SpatialExperiment(
    assays = list(counts = counts),
    spatialCoords = as.matrix(grid)
  )
  spe <- tweedspot(spe, smooth_k = 10, size_factor_method = "library_size",
    BPPARAM = BiocParallel::SerialParam(), verbose = FALSE)
  plot_top_features(spe, n_top = 2)
}
```

top_spatial_genes	<i>Return the top TweedSpot hits</i>
-------------------	--------------------------------------

Description

top_spatial_genes() returns the top n rows from [get_tweedspot_results\(\)](#) using a user-selected ranking metric.

Usage

```
top_spatial_genes(
  input,
  n = 10L,
  rank_by = c("padj", "pval", "stat", "dev_expl"),
  group = NULL
)
```

Arguments

input	A SpatialExperiment::SpatialExperiment containing TweedSpot results in rowData(input).
n	Number of rows to return. Default 10.
rank_by	Ranking column. Options are "padj", "pval", "stat", or "dev_expl". Default "padj".
group	For results of tweedspot(group = ...): a group label (or several) to return the top hits within those groups, or TRUE for the top feature-group pairs overall. Default NULL uses the per-feature results.

Value

A ranked data.frame containing the top TweedSpot hits.

Examples

```
counts <- matrix(c(1, 2, 3, 4, 2, 2), nrow = 2, byrow = TRUE)
rownames(counts) <- c("gene1", "gene2")
colnames(counts) <- paste0("spot", seq_len(ncol(counts)))
spe <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = counts),
  rowData = S4Vectors::DataFrame(
    gene_id = rownames(counts),
    gene_name = rownames(counts),
    tweedspot_stat = c(4, 2),
    tweedspot_pval = c(0.01, 0.05),
    tweedspot_padj = c(0.02, 0.05),
    tweedspot_edf = c(2, 2),
    tweedspot_dev_expl = c(0.3, 0.1)
  ),
  colData = S4Vectors::DataFrame(sample_id = rep("sample1", ncol(counts))),
```

```

    spatialCoords = cbind(x = seq_len(ncol(counts)), y = seq_len(ncol(counts)))
  )
  top_spatial_genes(spe, n = 1)

```

tweedspot	<i>Detect spatially variable genes with self-adaptive spatial Tweedie models</i>
-----------	--

Description

tweedspot() fits a Tweedie GAM with a spatial smooth $s(x, y)$ per gene and tests each gene for overall spatial variation.

Usage

```

tweedspot(
  input,
  assay_name = "counts",
  covariates = NULL,
  two_part = FALSE,
  family = "tw",
  tweedie_p = NULL,
  fit_method = "REML",
  use_bam = FALSE,
  bam_discrete = TRUE,
  bam_nthreads = 1L,
  smooth_k = NULL,
  size_factor_method = c("scrn", "library_size", "spanorm"),
  spanorm_output = c("size_factor", "counts"),
  spanorm_args = list(),
  p_criterion = c("BIC", "AIC"),
  padj_method = "BY",
  group = NULL,
  min_group_size = 50,
  group_adjust = c("stagewise", "global", "per_group"),
  alpha = 0.05,
  seed = NULL,
  BPPARAM = BiocParallel::bpparam(),
  verbose = TRUE
)

```

Arguments

input	A SpatialExperiment::SpatialExperiment with a count assay and spatial coordinates.
assay_name	Name of the count assay to model. Default "counts".

covariates	Optional one-sided formula specifying columns from <code>colData(input)</code> to adjust for, for example <code>~ batch + age + sex</code> . Character vectors of <code>colData(input)</code> column names are also supported.
two_part	Logical; if TRUE, fit the two-part (hurdle) model (Tweedie on counts plus a binomial model on presence/absence) and combine the two p-values with the Cauchy combination test (CCT), which tests whether either part varies in space and stays valid when the two p-values are dependent. When the presence/absence part is skipped, the count part's p-value is used. Default FALSE.
family	mgcv family for the count model. Default "tw" (Tweedie). Ignored when <code>tweedie_p</code> is supplied.
tweedie_p	Tweedie variance power p , where $Var(Y) = \phi\mu^p$. If NULL (default), TweedSpot estimates $p \in (1, 2)$ per gene with <code>mgcv::tw()</code> . Supported fixed values are 0 for Gaussian, 1 for Poisson, values in $(1, 2]$ for fixed Tweedie (2 is Gamma and requires strictly positive responses), and 3 for inverse Gaussian with strictly positive responses. All families use a log link so the size-factor offset acts multiplicatively. Use "auto" to choose the variance power per gene by an information criterion (see <code>p_criterion</code> and Details).
fit_method	Smoothing parameter estimation method passed to mgcv, e.g. "REML" (default) or "GCV.Cp".
use_bam	Logical; use <code>mgcv::bam()</code> instead of <code>mgcv::gam()</code> for scalability on large datasets. Default FALSE.
bam_discrete	Logical; when <code>use_bam = TRUE</code> , use discretized fitting via <code>mgcv::bam(discrete = TRUE)</code> for speed. Default TRUE.
bam_nthreads	Integer; number of threads for <code>mgcv::bam()</code> . Default 1.
smooth_k	Optional basis dimension for the spatial smooth. Smaller values can speed fitting at the cost of flexibility. Default NULL, which uses mgcv's default.
size_factor_method	Method used to compute the size-factor offset. "scrn" (default) uses pooling-based deconvolution size factors from <code>scrn::computeSumFactors()</code> ; "library_size" uses total counts scaled to a median of one; "spanorm" uses a SpaNorm fit, either as gene- and spot-specific size factors or as normalized counts (see <code>spanorm_output</code> and Details). For "scrn" and "library_size", an existing <code>colData(input)\$sizeFactor</code> is used as-is. For "spanorm", an existing <code>colData(input)\$sizeFactor</code> is used by SpaNorm as its base library size. The SpaNorm fit is stored in the returned object's metadata with a fingerprint of the data, settings, and seed; later calls (for example <code>plot_feature()</code>) reuse it only when that fingerprint matches, and refit otherwise. SpaNorm subsamples locations at random, so set seed for reproducible results.
spanorm_output	How the SpaNorm fit enters TweedSpot when <code>size_factor_method = "spanorm"</code> : "size_factor" (default) models the raw counts with the SpaNorm library-size (and batch) component as a gene- and spot-specific offset; "counts" models SpaNorm percentile-adjusted counts (PAC) with no further offset. Ignored for other methods.
spanorm_args	Optional named list of additional arguments passed to <code>SpaNorm::SpaNorm()</code> when <code>size_factor_method = "spanorm"</code> , for example <code>list(sample.p = 0.25, df.tps = 6)</code> . <code>adj.method</code> is fixed to "logpac".

p_criterion	Information criterion used when tweedie_p = "auto": "BIC" (default) or "AIC". Both use the log-likelihood and effective degrees of freedom reported by <code>mgcv::logLik.gam()</code> .
padj_method	Multiple-testing correction passed to <code>stats::p.adjust()</code> . Default "BY".
group	Optional name of a discrete <code>colData(input)</code> column, such as a cell-type label. When given, each feature is tested for spatial variation <i>within</i> each group (see Details), and results are stored per feature and group instead of in the usual <code>rowData()</code> columns. Default NULL.
min_group_size	Groups with fewer locations than this are not tested. Default 50.
group_adjust	Multiple-testing adjustment across features and groups: "stagewise" (default), "global", or "per_group". See Details.
alpha	Target significance level for group results: used by the stage-wise adjustment and to summarise <code>rowData(input)\$tweedspot_group_sig</code> . Default 0.05.
seed	Optional whole number. When given, the random-number generator is seeded with it for the duration of the call (and restored afterwards), so steps that use random numbers, such as <code>SpaNorm</code> 's subsampling of locations, give identical results on every run. The seed is stored and reused by <code>plot_feature()</code> and <code>plot_top_features()</code> . Default NULL.
BPPARAM	A <code>BiocParallel::BiocParallelParam</code> object controlling parallelization across genes. Default <code>BiocParallel::bpparam()</code> , which uses the currently registered Bioc-Parallel backend.
verbose	Logical; if TRUE, print lightweight progress messages before and after gene-wise model fitting. Default TRUE.

Details

Normalization. TweedSpot models raw counts with a log-link GAM and a size-factor offset, so only the relative size factors across spots matter. `scuttle::logNormCounts()` is not offered as a separate option because, without user-supplied size factors, it uses library-size factors (`colSums / mean(colSums)`) that differ from "library_size" only by a global constant, which the model intercept absorbs; with existing size factors it reuses them. Its log-transformed output is also incompatible with modelling raw counts. "spanorm" (Salim et al., 2025, Genome Biology) estimates a spatially smooth, gene-specific library-size effect jointly with spatial biology, without domain or cluster labels. TweedSpot treats that estimated effect as known, so any error in separating library size from biology remains in the data as spatial structure. When library size is itself spatially structured, this can make the spatial test strongly anti-conservative (in simulations, most null genes were called significant). Use "spanorm" as an exploratory option and check its calibration on your data; "scran" remains the default. `SpaNorm` is specific to spatial transcriptomics count data; "scran" and "library_size" are the modality-agnostic options.

`SpaNorm` fits, per gene, a negative binomial model whose log mean is $\gamma_g + f_g(x, y) + \{a_g + h_g(x, y)\} \log LS + \text{batch}$, where f_g is spatial biology and the second term is a spatially varying library-size effect. With `spanorm_output = "size_factor"`, TweedSpot models raw counts and uses the library-size and batch part of this predictor as the offset, $\log s_{gi}$, so the spatial biology is tested by the TweedSpot smooth rather than removed or reused. With `spanorm_output = "counts"`, TweedSpot models `SpaNorm`'s percentile-adjusted counts, $2^{\log_{\text{pac}}} - 1$, which are non-negative integers with the library-size effect removed, using a unit offset. The size-factor form is the default because it keeps the original count likelihood.

Automatic variance power. With `tweedie_p = "auto"`, each gene is fitted separately under several candidate variance functions $V(\mu) = \phi\mu^p$: Gaussian ($p = 0$), Tweedie with p estimated in (1, 2), and, when the gene has no zeros, Gamma ($p = 2$) and inverse Gaussian ($p = 3$). Every candidate shares the log link, spatial smooth, covariates, and offset, and estimates its own dispersion, so they differ only in the variance function. Poisson ($p = 1$) is not a candidate: its dispersion is fixed at 1, and choosing it on the same fit that is tested inflates false positives for genes whose noise happens to look spatial. It remains available as `tweedie_p = 1`. Each candidate is scored by

$$\text{BIC} = -2\ell + \log(n) \text{ df}$$

(or $\text{AIC} = -2\ell + 2 \text{ df}$ with `p_criterion = "AIC"`), where ℓ is the maximized log-likelihood and df the effective degrees of freedom from `mgcv::logLik.gam()`: smooth EDF, plus one for an estimated scale, plus one for an estimated Tweedie power. The candidate with the smallest value is used for the spatial test, and its power is reported in `rowData(input)$tweedspot_p` and `rowData(input)$tweedspot_family`. For integer data the estimated Tweedie power is restricted to $p \geq 1.1$, because as $p \rightarrow 1$ the Tweedie density concentrates on spikes at the integers, which inflates its likelihood and the spatial test.

Group-specific testing. With `group = "cell_type"`, the locations of each group are fitted separately with the same model ($s(x, y)$, covariates, offset, and family settings) to ask whether a feature varies in space among locations of that group. A feature that differs between groups but is flat within each group, a pure composition effect, is therefore not called spatially variable. Size factors and the coordinate scaling are computed once on all locations so that every group shares one scale; the spatial basis size is capped by each group's number of locations; covariates that are constant within a group are dropped for that group; and a feature is tested in a group only if it is detected in at least as many of its locations as the smooth has basis functions. Groups with fewer than `min_group_size` locations are skipped.

Group p-values are adjusted with `group_adjust`. "stagewise" first screens each feature with the Cauchy combination of its group p-values and adjusts those across features with `padj_method`, then Holm-adjusts the group p-values within the feature at the reduced level $\alpha * R / m$ (R of m features pass the screen); a feature-group pair's adjusted p-value is the larger of the screening and confirmation values, so it is significant at α only if the feature passes the screen and the group passes confirmation. Stage-wise adjusted p-values are calibrated for the α used in the run. "global" adjusts all pairs together and "per_group" adjusts within each group. The full table, one row per feature and group, is stored in `S4Vectors::metadata(input)$tweedspot_group$results` and returned by `get_tweedspot_results()` with its group argument; `rowData(input)` gains `tweedspot_group_padj` (feature-level adjusted p-value) and `tweedspot_group_sig` (groups significant at α).

Value

The input `SpatialExperiment` with per-gene statistic, p-value, adjusted p-value, smooth effective degrees of freedom, deviance explained, fitted variance power (`tweedspot_p`), and the corresponding family (`tweedspot_family`) written to `SummarizedExperiment::rowData()`. The model settings are stored in `S4Vectors::metadata(input)$tweedspot` and reused by `plot_feature()` and `plot_top_features()`.

References

Lun ATL, Bach K, Marioni JC (2016). Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biology* 17:75.

Salim A, Bhuva DD, Chen C, et al. (2025). SpaNorm: spatially-aware normalization for spatial transcriptomics data. *Genome Biology*.

Examples

```
counts <- matrix(
  c(0, 1, 2, 3, 4, 5, 5, 4, 5, 4, 3, 2),
  nrow = 2,
  byrow = TRUE
)
rownames(counts) <- c("gene1", "gene2")
colnames(counts) <- paste0("spot", seq_len(ncol(counts)))
spe <- SpatialExperiment::SpatialExperiment(
  assays = list(counts = counts),
  rowData = S4Vectors::DataFrame(
    gene_id = rownames(counts),
    gene_name = rownames(counts)
  ),
  colData = S4Vectors::DataFrame(batch = rep(c("A", "B"), length.out = ncol(counts))),
  spatialCoords = cbind(x = seq_len(ncol(counts)), y = seq_len(ncol(counts)))
)
spe <- tweedspot(
  spe,
  family = "poisson",
  smooth_k = 4,
  size_factor_method = "library_size",
  BPPARAM = BiocParallel::SerialParam()
)
SummarizedExperiment::rowData(spe)[, c("tweedspot_stat", "tweedspot_pval")]
```

Index

BiocParallel::BiocParallelParam, [14](#)
BiocParallel::bpparam(), [14](#)

filter_features, [2](#)

get_tweedspot_results, [5](#)
get_tweedspot_results(), [11](#), [15](#)

mgcv::bam(), [13](#)
mgcv::gam(), [13](#)
mgcv::logLik.gam(), [14](#), [15](#)
mgcv::tw(), [13](#)

plot_feature, [7](#)
plot_feature(), [13–15](#)
plot_top_features, [9](#)
plot_top_features(), [14](#), [15](#)

scrnan::computeSumFactors(), [13](#)
SpatialExperiment::SpatialExperiment,
 [3](#), [6](#), [7](#), [9](#), [11](#), [12](#)
stats::p.adjust(), [14](#)
SummarizedExperiment::rowData(), [15](#)

top_spatial_genes, [11](#)
tweedspot, [12](#)
tweedspot(), [2](#), [3](#), [5](#), [7–10](#)