Package 'TSENAT'

September 1, 2026

Type Package

Title Tsallis Entropy Analysis Toolbox

Version 0.99.35

Description Quantifies and models isoform-usage complexity in RNA-seq data using Tsallis entropy, a scale-dependent diversity measure. By tuning the entropic index parameter (q), TSENAT examines transcriptome heterogeneity at different scales: rare variants (low q) or dominant isoforms (high q). It enables computing Tsallis entropy and Tsallis divergence from transcript-level estimates, comparing measures between conditions, testing for differences, and visualizing scale-dependent complexity via q-curves.

License GPL (>= 3) + file LICENSE

Encoding UTF-8

LazyData false

Depends R (>= 4.5.0)

Imports SummarizedExperiment, methods, stats, ggplot2, withr, BiocParallel, matrixStats, mgcv, geepack, pheatmap, cowplot, dplyr, tidyr, S4Vectors, rlang, Rcpp, memoise, digest, readr, ARTool, nlme, splines

LinkingTo Rcpp, RcppArmadillo

Suggests testthat, knitr, rmarkdown, BiocStyle, kableExtra, patchwork, covr, RColorBrewer, gamsel, glmmTMB, glmnet, gridExtra, fdrtool, MASS, SplicingFactory, devtools, gtable

VignetteBuilder knitr

URL https://gallardoalba.github.io/TSENAT

BugReports https://github.com/gallardoalba/TSENAT/issues

biocViews Transcriptomics, RNASeq, DifferentialSplicing, AlternativeSplicing, TranscriptomeVariant, GeneExpression, DifferentialExpression

Config/roxygen2/version 8.0.0

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

git_branch devel

git_last_commit f723821

git_last_commit_date 2026-08-18

Repository Bioconductor 3.24

Date/Publication 2026-08-31

Author Cristóbal Gallardo Alba [aut, cre] (ORCID:
<https://orcid.org/0000-0002-5752-2155>)

Maintainer Cristóbal Gallardo Alba <gallardoalba@pm.me>

Contents

`TSENAT-package` *TSENAT: Tsallis Entropy for Systems Evolutionary Analysis and Network Analysis Tool*

Description

TSENAT is a Bioconductor package that quantifies transcriptomic complexity at the isoform level using Tsallis entropy, a generalization of Shannon entropy that captures rare-event dynamics and scale-dependent information structure. The package integrates diversity quantification, statistical resampling, linear mixed models, and information-theoretic divergence metrics into a unified analytical framework for comparative transcriptomics.

Scientific Rationale

Classical Shannon entropy (a special case of Tsallis with q=1) treats all probability states equally. Tsallis entropy with order parameter q weights rare and common transcripts differently:

- $q < 1$: Emphasizes rare isoforms, capturing tail behavior
- $q = 1$: Shannon entropy (equipartition weighting)
- $q = 2$: Simpson entropy (dominance weighting, less sensitive to rare species)
- $q \to \infty$: Reciprocal of the maximum transcript fraction

This multiscale perspective reveals condition-specific dependencies in splicing architecture and detects genes with asymmetric isoform distributions that traditional metrics miss.

Workflow Overview

TSENAT implements a complete analytical pipeline:

1. **Diversity Quantification**: Compute Tsallis entropy H_q across a range of q-values (0 to ∞) for each gene-condition group. Returns Hill numbers ($D_q = \exp(H_q)$) for interpretable effective isoform counts.
2. **Jackknife Resampling**: Generate confidence intervals via leave-one-out resampling. Identifies genes with statistically significant entropy outliers and condition-biased isoform usage (isoform switching).
3. **Linear Mixed Models**: Tests gene-by-condition interaction effects on diversity landscapes using nlme, MASS, or geepack backends. Supports paired designs and random effect structures.
4. **Divergence Analysis**: Computes Tsallis divergence (generalized Kullback-Leibler) between condition groups. Detects differential transcript complexity via effect size estimation.
5. **Visualization**: Generates publication-ready plots: q-curves, volcano plots, heatmaps, violin distributions, and dimension-reduced embeddings.

Core Functions

`TSENAT` Main orchestration function. Executes complete pipeline with configurable method backends and parallelization.

`TSENAT_config` Configure analysis parameters: q-values, condition grouping, sample pairing, and method selection.

`calculate_diversity` Compute Tsallis entropy and Hill numbers. Vectorized over genes and q-values with numerical stability safeguards.

`calculate_jeo` Resampling-based confidence intervals for diversity. Identifies outlier genes and validates q-value signal.

`calculate_sait` Statistical testing for gene-by-condition interactions. LMM, GAM, and rank-based approaches.

`calculate_divergence` Tsallis divergence between groups with effect size (Cohen's d, rank-biserial) and hypothesis testing.

`filter_analysis` Filter genes by diversity, effect size, or interaction significance (adjustable FDR, Benjamini-Hochberg).

Statistical Methods

Diversity Metrics Tsallis Entropy: $H_q = \frac{1}{q-1}\left(1 - \sum_i p_i^q\right)$ Hill Numbers: $D_q = \left(\sum_i p_i^q\right)^{\frac{1}{1-q}}$

Divergence Tsallis Divergence: $D_q(p||r) = \frac{1}{q-1}\left(1 - \sum_i p_i^q r_i^{1-q}\right)$ Kullback-Leibler distance (q=1 limit): $D_{KL}(p||r) = \sum_i p_i \log(p_i/r_i)$

Resampling Jackknife leave-one-out, block bootstrap (paired data), and percentile/BCA confidence intervals with accuracy enhancement for skewed distributions.

Interaction Testing LMM: $H_{g,c} = \beta_0 + \beta_g(\text{gene}) + \beta_c(\text{condition}) + \beta_{gc}(\text{gene} \times \text{condition}) + \epsilon$ With random effects for paired/nested designs.

Key Features

- **Multiscale q-parameter sweep**: Detects biological signals at different probability scales without multiple testing correction for q-values.
- **Robust resampling**: Leave-one-out jackknife + block bootstrap handles paired designs and maintains correlation structure.
- **Multiple method backends**: nlme, MASS, geepack, ARTool, or rank-based (ART/Conover-Iman), sign test) for model flexibility.
- **FDR-corrected testing**: Benjamini-Hochberg adjustment across genes and q-values.
- **Production-grade visualization**: ggplot2-based plots optimized for large gene sets and publication quality.
- **Reproducibility tracking**: Automatic logging of function calls, parameters, timestamps, and package version for audit trails.
- **S4 object-oriented design**: Unified `TSENATAnalysis` container ensures metadata integrity throughout the pipeline.

Data Input & Output

Input SummarizedExperiment object with gene-by-sample count matrix (raw or normalized) and sample metadata (condition, pairing, batch).

Output TSENATAnalysis S4 object containing: (1) Diversity values across q-parameter range, (2) Jackknife CIs and switching indicators, (3) LMM/interaction statistics with p-values and effect sizes, (4) Divergence metrics with hypothesis test results, (5) Cached publication-ready plots, (6) Reproducibility metadata.

Interpreter Notes

- **q-value sweeps reveal biological complexity**: Genes showing entropy peaks at intermediate q-values (0.5-1.5) suggest balanced isoform distributions; peaks at q \to \infty indicate dominant-isoform architectures.
- **Interaction q-profiles**: q-dependent gene-by-condition interactions signal condition-specific isoform switching; flat profiles indicate constitutive splicing.
- **Divergence as biological distance**: Tsallis divergence quantifies transcript composition distance; high divergence (D_q > 0.5) indicates distinct splicing programs.

- **Statistical interpretation**: Jackknife CIs that exclude the point estimate (valid for skewed distributions) occur with bounded entropies; not a software error.

References

Tsallis entropy and information theory:

- Tsallis, C. (1988). Possible generalization of Boltzmann-Gibbs statistics. *Journal of Statistical Physics*, 52(1-2), 479-487.
- Furuichi, S. (2006). Information theoretical properties of Tsallis entropies. *Journal of Mathematical Physics*, 47(2), 023302.

Hill numbers and diversity indices:

- Hill, M. O. (1973). Diversity and evenness: A unifying notation and its consequences. *Ecology*, 54(2), 427-432.
- Chao, A., Chiu, C. H., & Jost, L. (2014). Unifying species diversity, phylogenetic diversity, functional diversity, and related similarities into a single framework. *Annual Review of Ecology, Evolution, and Systematics*, 45, 297-324.

Isoform switching and transcriptomics:

- Vitting-Seerup, K., et al. (2017). IsoformSwitchAnalyzeR enables robust detection of isoform switches and novel isoforms in the human transcriptome from long-read cDNA sequencing data. *Genome Biology*, 18(1), 122.

S4 Container

`TSENATAnalysis-class` — Central container unifying all analysis components. Access results via slots: `@diversity_results`, `@sait_results`, `@jackknife_results`, `@divergence_results`, or accessor methods `se`, `metadata<-`, and `results`.

Author(s)

Maintainer: Cristóbal Gallardo Alba <gallardoalba@pm.me> (ORCID)

Authors:

- Cristóbal Gallardo Alba <gallardoalba@pm.me> (ORCID)

See Also

`TSENAT` for running the complete analysis pipeline. `se`, `metadata<-`, and `results` for accessor functions. `TSENATAnalysis-class` for the S4 object structure. `build_analysis` for creating TSENATAnalysis objects.

AllGenerics *Generic Functions for TSENATAnalysis S4 Class*

Description

Define generic functions for accessing components of TSENATAnalysis objects. These are the recommended way to extract results from analysis objects, following Bioconductor best practices.

Value

These generic functions return different types depending on the specific method: - Accessor methods return data.frames or lists containing analysis results - See individual method documentation for specific return types

build_analysis *Build a Complete TSENATAnalysis Object*

Description

Convenience wrapper that combines . `build_se()` and `TSENATAnalysis()` into a single function call. This creates a complete analysis object ready for Tsallis entropy computation and downstream analysis.

Usage

```
build_analysis(
  readcounts = NULL,
  salmon_dir = NULL,
  tx2gene,
  assay_name = "counts",
  metadata = NULL,
  tpm = NULL,
  effective_length = NULL,
  config = list(),
  skip = FALSE,
  verbose = FALSE
)
```

Arguments

`readcounts`	A matrix or data.frame of transcript-level read counts with transcript IDs as row names and sample names as column names. Typically output from quantification tools (SALMON, kallisto, etc.). Optional when `salmon_dir` is provided (in which case readcounts are auto-loaded from quant.sf files).
`salmon_dir`	Optional character path to directory containing Salmon quantification output. Expected structure: `salmon_dir/sample_name/quant. sf`. When provided, automatically discovers and reads all quant. sf files. If both `readcounts` and `salmon_dir` are provided, `salmon_dir` takes precedence. Default: `NULL`.

`tx2gene` Either: - A path to a GFF3 or GFF3.gz file containing transcript-to-gene mapping - A path to a TSV file with columns 'Transcript' and 'Gene' - A data.frame with transcript-to-gene mapping

`assay_name` Character. Name for the assay (default: 'counts').

`metadata` Optional data.frame with sample metadata. Should have sample names as row names and metadata columns (e.g., sample_type, condition, etc.). If NULL, will attempt to read from `config$metadata`. Priority: explicit `metadata` argument > `config$metadata` > NULL.

`tpm` REQUIRED matrix of transcript-level TPM values (Transcripts Per Million). Must be provided and will be stored in the SummarizedExperiment for use during diverse filtering. Same dimensions as readcounts required (rows = transcripts, columns = samples). Typically from SALMON quantification output. If not provided to build_analysis(), subsequent filter_analysis() calls will fail with an explicit error message.

`effective_length` REQUIRED numeric vector of transcript effective lengths (e.g., from SALMON EffectiveLength column). Length must match nrow(readcounts). Typically obtained as the median effective length across all samples. If not provided to build_analysis(), the data will not be stored for later use in length-normalized calculations.

`config` Optional list of configuration parameters to store in the TSENATAnalysis object. Can also contain `config$metadata` which will be used if the `metadata` argument is NULL. Following Bioconductor best practices (fail-fast principle), create configuration via `TSENAT_config()` FIRST, then pass to `build_analysis()` at object construction time. This ensures invalid parameters are caught immediately, before analysis proceeds. See examples below for recommended usage pattern.

`skip` Logical. If TRUE, allow unmapped transcripts (transcripts not found in tx2gene mapping) and remove them from analysis. If FALSE (default), stop with an error when unmapped transcripts are detected. Useful for handling data with transcript IDs that don't match the annotation file provided.

`verbose` Logical. If TRUE, print informative messages during execution (e.g., Salmon sample discovery, progress on data loading). Default: TRUE.

Details

When using `salmon_dir`, the function automatically:

1. Discovers all Salmon sample folders and quant.sf files
2. Reads transcript counts (NumReads), TPM, and effective_length
3. Extracts sample names from directory structure
4. Creates count matrix ready for analysis

The `salmon_dir` parameter provides a convenient alternative to manually constructing the `readcounts` matrix, especially useful in Galaxy workflows.

Value

A `TSENATAnalysis` S4 object with:

`@se` The SummarizedExperiment containing transcript counts and metadata

@config	Analysis configuration (empty list or user-provided)
@diversity_results	Empty list (populated by calculate_diversity())
@divergence_results	Empty list (populated by calculate_divergence())
@sait_results	Empty list (populated by calculate_sait())
@jackknife_results	Empty list (populated by jackknife functions)
@plots	Empty list (populated by plotting functions)
@metadata	Metadata with package version and creation timestamp

CRITICAL: TPM and effective_length Requirements

Both tpm and effective_length MUST be provided to ensure correct filtering and normalization in downstream analysis:

- filter_analysis() requires TPM data (stored in metadata). If TPM is missing, the function will fail with an explicit error message that guides you to pass it to build_analysis().
- calculate_diversity() uses effective_length for length-normalized entropy calculations from RAW counts.
- TPM and effective-length correction are mutually exclusive and are never combined: TPM already incorporates effective-length normalization, so diversity on TPM together with an effective_length is a hard error (double normalization).

Following Bioconductor best practices (fail-fast principle), these are explicit parameters, not optional. They must be passed at object construction time:

```
analysis <- build_analysis(
  readcounts = readcounts,
  metadata = metadata_df,
  tx2gene = gff3_file,
  tpm = tpm,                      # REQUIRED from Salmon output
  effective_length = effective_length,  # REQUIRED from Salmon output
  config = config
)
```

This wrapper combines two steps into one:

1. Call .build_se() to create a SummarizedExperiment from transcript counts
2. Wrap the result in TSENATAnalysis() to create the analysis object

The returned object is ready for diversity analysis via calculate_diversity().

If you need to inspect or filter the SummarizedExperiment before creating the TSENATAnalysis object, call .build_se() and TSENATAnalysis() separately.

See Also

TSENATAnalysis for the S4 class structure calculate_diversity for computing Tsallis entropy

Examples

```
# Create example transcript count data
set.seed(42)
n_genes <- 10
n_isoforms_per_gene <- 3
n_isoforms <- n_genes * n_isoforms_per_gene
n_samples <- 10

# Generate count matrix
counts <- matrix(rpois(n_isoforms * n_samples, lambda = 20),
                 nrow = n_isoforms, ncol = n_samples)
rownames(counts) <- paste0('TX_', 1:n_isoforms)
colnames(counts) <- paste0('Sample_', 1:n_samples)

# Create tx2gene mapping
tx2gene <- data.frame(
  Transcript = rownames(counts),
  Gene = rep(paste0('GENE_', 1:n_genes), each = n_isoforms_per_gene))

# Create sample metadata
metadata <- data.frame(
  sample = colnames(counts),
  condition = rep(c('control', 'treatment'), each = 5),
  row.names = colnames(counts))

# Build analysis object - use NAMED parameters to avoid confusion
# Method 1: With explicit tx2gene data.frame (most common)
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(
  readcounts = counts,
  tx2gene = tx2gene,
  metadata = metadata,
  config = config)

# Verify the analysis object was created
analysis
print(dim(analysis))

# Method 2: From Salmon quantification folder
# Requires directory structure like:
#   salmon_output/
#     sample1/quant.sf
#     sample2/quant.sf
#     ...
#
# salmon_dir <- '/path/to/salmon/directory'
#
# First create sample metadata matching Salmon sample names
# salmon_metadata <- data.frame(
#   condition = c('control', 'control', 'treatment', 'treatment'),
#   row.names = c('sample1', 'sample2', 'sample3', 'sample4')
# )
#
# analysis_salmon <- build_analysis(
#   salmon_dir = salmon_dir,
#   tx2gene = 'annotation.gff3.gz',  # Auto-parsed from GFF3
```

```
#   metadata = salmon_metadata
# )
#
# Method 3: Hybrid - Salmon counts with manual tx2gene
# analysis_hybrid <- build_analysis(
#   salmon_dir = salmon_dir,
#   tx2gene = tx2gene,  # data.frame instead of file
#   metadata = salmon_metadata
# )
#
# Method 4: Pass metadata via config (parameter resolution pattern)
# cfg <- TSENAT_config()
# cfg$metadata <- metadata
# analysis_with_config <- build_analysis(
#   readcounts = counts,
#   tx2gene = tx2gene,
#   config = cfg
#   # Note: metadata argument omitted - will be read from config$metadata
# )

# Advanced: Assigning metadata to assays after object creation
# When adding metadata to SummarizedExperiment assays, always use the
# S4Vectors namespace to ensure proper method dispatch:
#
#   se <- getSE(analysis)
#   assay_with_ci <- SummarizedExperiment::assay(se, 'log2fc_ci')
#   S4Vectors::metadata(assay_with_ci)$lower <- ci_lower_bounds
#   S4Vectors::metadata(assay_with_ci)$upper <- ci_upper_bounds
#
# Note: Avoid using metadata(assay) without the namespace - this can
# cause silent failures in S4 object metadata assignment.
```

`calculate_assumptions` *Test statistical assumptions on diversity data in TSENATAnalysis*

Description

Test statistical assumptions on diversity data in TSENATAnalysis

Usage

```
calculate_assumptions(
  analysis,
  q = NULL,
  checks = "rank",
  alpha = 0.05,
  format = "text",
  ...
)

## S4 method for signature 'TSENATAnalysis'
calculate_assumptions(
  analysis,
```

```
  q = NULL,
  checks = "rank",
  alpha = 0.05,
  format = "text",
  ...
)
```

Arguments

analysis	TSENATAnalysis object with diversity results stored in @diversity_results.
q	numeric. Q-value(s) to extract from diversity results. If NULL, uses the first available diversity result or q=1.0.
checks	character. Which assumptions to test (default: 'rank'). Presets: - 'rank': core assumption checks (exchangeability, monotonicity, consistency) - 'all': all checks including GAM diagnostics Explicit: character vector like c('exchangeability', 'monotonicity').
alpha	numeric. Significance level for tests (default: 0.05).
format	character. Output format when used with results(). 'text' (default): formatted text output for display 'list': returns structured list for programmatic access.
...	Additional arguments (output_file, verbose for file output).

Details

This wrapper calls .calculate_assumptions() on diversity data extracted from the analysis object. Evaluates data stability and consistency across dimensions. Results include:

exchangeability Permutation test for independence and temporal/spatial structure

monotonicity Spearman correlation consistency across rows

consistency Kendall's W concordance and ICC across samples

Data Extraction Priority: 1. If q specified: uses diversity result for that q-value 2. If q NULL: uses first available diversity result 3. If no diversity results: extracts from cached combined result (@metadata$diversity_combined)

Value

Modified TSENATAnalysis object with assumption test results stored in @metadata$rankbased_assumptions.

Examples

```
# Load example data (matching TSENAT.Rmd workflow)
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# Build analysis from vignette data and create small subset
config <- TSENAT_config(
```

```
  sample_col = 'sample',
  condition_col = 'condition',
  q_values = seq(0, 2, by = 0.05),
  paired = FALSE
)
analysis <- build_analysis(
  readcounts = readcounts,
  metadata = metadata_df,
  tx2gene = gff3_dataset,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes = 200)
analysis <- calculate_diversity(analysis, q = c(0.5, 1.0, 1.5))
analysis <- calculate_assumptions(analysis, q = 1.0)
# Check results using rank_test accessor
results_df <- results(analysis, type = 'rank_test')
```

`calculate_concordance` *Compare method concordance for differential analysis results*

Description

Compares statistical results from two different methods (typically SAIT/GAM for continuous data and Conover-Iman Rank Transform tests) to assess agreement and identify genes detected by one method but not the other.

Usage

```
calculate_concordance(analysis_sait, analysis_rank = NULL, ...)

## S4 method for signature 'TSENATAnalysis'
calculate_concordance(
  analysis_sait,
  analysis_rank = NULL,
  verbose = FALSE,
  output_file = NULL,
  ...
)
```

Arguments

`analysis_sait`	`TSENATAnalysis` object containing SAIT/GAM analysis results (from `calculate_sait()`).
`analysis_rank`	`TSENATAnalysis` object or NULL. If NULL, uses legacy single-object API with analysis_sait containing both results. If provided, compares SAIT results from analysis_sait with rank-test results from analysis_rank.
`...`	Additional arguments for future extensibility.
`verbose`	`logical`. Print progress messages (default: FALSE).
`output_file`	`character` or NULL. Optional file path to save results. Supported formats: .rds (for S4 objects). Default: NULL (no file output).

Details

Compares results from two different statistical methods (typically GAM for continuous and Conover-Iman Rank Transform for rank-based analysis) on the same data. Identifies: - Genes significant in both methods (high confidence) - Genes detected by one method only (potential false positives or method-specific signal) - Spearman correlation of p-values (overall agreement trends)

Value

Modified TSENATAnalysis object with concordance results stored in: `@metadata$method_concordance`:

comparison_df Data frame comparing results from both methods

spearman_rho Spearman correlation between adjusted p-values

high_confidence Genes with strong agreement

agreement_table Contingency table of significant/non-significant calls

sait_method Method name used for SAIT/GAM analysis

rank_method Method name used for rank-based analysis

timestamp When concordance was computed

Examples

```
# Compare results from SAIT and rank-based testing
# (Requires pre-computed analysis objects from calculate_sait and calculate_rank_transform)
# results_df <- results(calculate_concordance(analysis_sait, analysis_rank))
```

`calculate_divergence` *Calculate divergence metrics and store in TSENATAnalysis*

Description

Wrapper around .calculate_divergence() that manages TSENATAnalysis object. Calculates Tsallis divergence between experimental conditions for transcripts across multiple q-values to detect condition-specific transcript remodeling.

Usage

```
calculate_divergence(
  analysis,
  q = NULL,
  verbose = FALSE,
  nthreads = NULL,
  output_file = NULL,
  control_group = NULL,
  paired = NULL,
  method = NULL,
  bootstrap = NULL,
  nboot = NULL,
  progress = FALSE,
  ...
)
```

Arguments

analysis	TSENATAnalysis object.
q	numeric. Q-value(s) for divergence (single value or vector). If NULL, reads from analysis@config$q. If not in config, defaults to seq(0.01, 2, by = 0.05) for full spectrum computation.
verbose	logical. Print progress messages. Default: TRUE.
nthreads	numeric or NULL. Number of CPU threads for parallel processing. If NULL, reads from @config$nthreads (or defaults to 1).
output_file	character or NULL. Optional file path to save results. Supported formats: .rds (for S4 objects), .tsv, .csv, .txt (for tables). Default: NULL (no file output).
control_group	character or NULL. Control group identifier for divergence comparison. If NULL, reads from @config$control_group if available.
paired	logical. Whether to use paired design. Default: FALSE. If not specified, reads from @config$paired if available.
method	character or NULL. Statistical method for divergence calculation. If NULL, reads from @config$method if available.
bootstrap	logical. Whether to compute bootstrap confidence intervals. Default: FALSE. If not specified, reads from @config$bootstrap if available.
nboot	numeric or NULL. Number of bootstrap replicates. Default: NULL. If NULL, reads from @config$nboot if available.
progress	logical. Show progress bar during computation. Default: FALSE.
...	Additional arguments passed to the base divergence function.

Details

Key Features:

- Multi-q analysis: Divergence computed across full q-spectrum simultaneously
- Bootstrap confidence intervals: Quantify uncertainty in divergence estimates
- Multiple testing correction: Hochberg, Benjamini-Yekutieli, or no correction
- Paired designs: Supports paired/repeated measures via subject_col parameter
- Partially paired designs: Incomplete pairs (a sample missing its control or treatment counterpart) are treated as unpaired units and do not receive the paired covariance treatment
- Effect size reporting: Log-fold-change and confidence intervals per gene
- Flexible control group: Compare any condition vs. any other condition

Mathematical Background: Tsallis divergence between the per-condition isoform-usage distributions P (control) and Q (treatment), built by summing transcript counts across samples within each condition:

```
D_q(P||Q) = (sum_i P_i^q * Q_i^(1-q) - 1) / (q - 1)   for q > 0, q != 1
D_1(P||Q) = sum_i P_i * log(P_i / Q_i)                 (KL limit)
```

q = 0 convention: D_0(P||Q) = 1 - sum_{i: P_i > 0} Q_i (the Q-mass on P's zero support, with 0^0 = 0). Under pseudocount regularization all bins are positive and D_0 = 0. Measures how much isoform composition changes from control to condition. Values near 0: Similar isoform composition; Large positive values: Major change. Note: the default is norm = 'none', so reported values are raw divergences on the count scale (not rescaled across genes).

Example Use Case: Control sample: All reads from dominant isoform (low entropy)
Tumor sample: Reads spread across multiple isoforms (high entropy)
Result: Large divergence indicating condition-specific isoform switching.

Parameter Resolution: Parameters are resolved in priority order: 1. Explicit arguments passed to function 2. Values from analysis@config (if present) 3. Function defaults

Requires diversity results from calculate_diversity() as prerequisite.

Value

Modified TSENATAnalysis with divergence metrics in `@divergence_results` (stored as list of data.frames or matrices).

Examples

```
# Load example data (matching TSENAT.Rmd workflow)
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package = 'TSENAT')

# Build analysis from vignette data and create manageable subset
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(
  readcounts = readcounts,
  tx2gene = gff3_dataset,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)
# Use 200+ genes to ensure diversity filtering doesn't remove all genes
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes = 200)

# Compute diversity first (required for divergence)
analysis <- calculate_diversity(analysis, q = c(0.5, 1.0, 1.5))

# Calculate divergence across q-values
analysis <- calculate_divergence(analysis, q = c(0.5, 1.0, 1.5))

# Check divergence results using unified accessor
head(results(analysis, type = 'divergence'))
```

`calculate_diversity` *Calculate diversity and store in TSENATAnalysis*

Description

Wrapper around .calculate_diversity() that manages TSENATAnalysis object. Calculates Tsallis entropy (diversity) across multiple q-values for each gene to quantify isoform complexity and transcript heterogeneity.

Usage

```
calculate_diversity(
  analysis,
  q = NULL,
  norm = NULL,
  norm_method = NULL,
  reference_group = NULL,
  verbose = NULL,
  show_messages = FALSE,
  what = NULL,
  nthreads = NULL,
  pseudocount = NULL,
  min_valid_frac = NULL,
  shrinkage = NULL,
  bootstrap = NULL,
  nboot = NULL,
  bootstrap_method = NULL,
  bootstrap_ci = NULL,
  bootstrap_include_diagnostics = NULL,
  output_file = NULL,
  log_base = NULL,
  ...
)
```

Arguments

analysis	TSENATAnalysis object.
q	numeric. Q-value(s) for Tsallis entropy (single value or vector). If NULL, reads from analysis@config$q. If not in config, defaults to seq(0.01, 2, by = 0.05) for full spectrum computation.
norm	logical or character. Normalization method: TRUE, FALSE, 'none', 'range', 'zscore', 'log_odds_ratio', 'relative_reference'. If NULL, reads from @config$norm or defaults to TRUE.
norm_method	character. Post-hoc normalization method applied after diversity computation. Options: • 'default' - Simple normalization by theoretical maximum (current behavior) • 'zscore' - Z-score normalization per q-value • 'log_odds_ratio' - Log-ratio of entropy to the uniform maximum (q and isoform-aware). Values are <= 0: 0 = uniform, < 0 = more concentrated. Name kept for backwards compatibility; this is a log-relative-entropy ratio, not an odds ratio. • 'relative_reference' - Divide by reference group mean (requires reference_group) • NULL - No post-hoc normalization (default)

If NULL, reads from `@config$norm_method` if available.

`reference_group`
: `character`. For `norm_method = 'relative_reference'`, the reference group column name (e.g., from colData). If NULL, uses first group in colData. If NULL, reads from `@config$reference_group` if available.

`verbose`
: `logical`. Print progress messages. Default: TRUE. If not specified, reads from `@config$verbose` if available.

`show_messages`
: `logical`. Display verbose messages during computation. Default: FALSE.

`what`
: `character`. Output type: 'S' (entropy) or 'D' (diversity). Default: 'S'. If NULL, reads from `@config$what` if available.

`nthreads`
: `numeric` or `NULL`. Number of CPU threads for parallel processing. If NULL, reads from `@config$nthreads` (or defaults to 1).

`pseudocount`
: `numeric` or `character`. Pseudocount value or 'auto'. Default: 0. If NULL, reads from `@config$pseudocount` if available.

`min_valid_frac`
: `numeric`. Minimum fraction of valid samples for gene filtering. Default: NULL. If NULL, reads from `@config$min_valid_frac` if available.

`shrinkage`
: `character`. Shrinkage method: 'none' or 'empirical_bayes'. Default: 'none'. If NULL, reads from `@config$shrinkage` if available.

`bootstrap`
: `logical`. Compute bootstrap confidence intervals. Default: FALSE. If not specified, reads from `@config$bootstrap` if available.

`nboot`
: `numeric` or `NULL`. Number of bootstrap replicates. Default: NULL. If NULL, reads from `@config$nboot` if available.

`bootstrap_method`
: `character`. Bootstrap method: 'percentile' or others. Default: 'percentile'. If NULL, reads from `@config$bootstrap_method` if available.

`bootstrap_ci`
: `numeric`. Bootstrap confidence interval level (0-1). Default: 0. 95. If NULL, reads from `@config$bootstrap_ci` if available.

`bootstrap_include_diagnostics`
: `logical`. Include bootstrap diagnostic information. Default: FALSE. If NULL, reads from `@config$bootstrap_include_diagnostics` if available.

`output_file`
: `character` or `NULL`. Optional file path to save results. When provided, generates TWO files:

  1. Primary output: Analysis object (.rds) or table (.tsv/.csv/.txt)
  2. Secondary output: Diversity spectrum statistics (TSV format) with suffix `_spectrum.tsv`

  Example: output_file = 'analysis.rds' generates:

  - `analysis.rds` - TSENATAnalysis object
  - `analysis_spectrum.tsv` - Spectrum statistics

  The spectrum file contains columns: q, central (median diversity), spread (IQR), count, and group (if grouping variable available). Default: NULL (no file output).

`log_base`
: `numeric`. Logarithm base for Shannon entropy computation (default: `exp(1)` for natural log). Set to 2 for bits or 10 for dits. Only affects Shannon entropy (q = 1); Tsallis entropy (q != 1) is scale-invariant and does not use a logarithm base.

`...`
: Additional arguments passed to underlying functions for extensibility.

Details

Key Features:

- Multi-q analysis: Entropy computed for q = 0.01 to 2.00 (by default)
- Normalized entropy: Scale to [0, 1] using theoretical maximum log(m)
- Bootstrap confidence intervals: Quantify uncertainty in estimates
- TPM normalization: Optional SALMON TPM-based weighting
- Multiple normalization methods: Range, Z-score, log-odds-ratio, relative
- Spectrum computation: Aggregate diversity statistics across all q-values

Mathematical Background: Tsallis entropy H_q for q-parameter:

```
H_q(X) = (1/(q-1)) * (1 - Sum p_i^q)  [for q != 1]
H_1(X) = -Sum p_i * log(p_i)  [Shannon entropy, limit q→1]
```

where p_i = relative abundance of isoform i for a gene. Larger q emphasizes dominant isoforms; smaller q emphasizes rare ones.

Example Interpretation:

- Gene with 1 isoform: H_q = 0 for all q (no diversity)
- Gene with 2 equal isoforms: H_q ~ 0.5-1.0 depending on q
- Gene with m equally abundant isoforms: H_q = 1.0 (maximum diversity)

This wrapper calls `.calculate_diversity()` once per q-value, storing results as SummarizedExperiment objects. It extracts key parameters from `analysis@config` with priority resolution (explicit > `@config` > default).

Counts, TPM and effective length: The wrapper always computes entropy from the RAW count assay with effective-length correction ('tpm = FALSE' internally). The TPM matrix stored in metadata by `build_analysis()` is used only for filtering/QC (e.g., `filter_analysis()`). Because TPM already incorporates effective-length normalization, the two are never combined in a single computation (combining them at the core level is a hard error).

Diversity Spectrum Computation: By default, this function computes and saves a diversity spectrum (aggregated statistics across all q-values and groups) when `output_file` is provided. The spectrum contains:

- `q`: Diversity parameter value
- `central`: Median (or mean) diversity across all genes
- `spread`: IQR (or SD) around central value
- `count`: Number of valid measurements
- `group`: Condition group (if applicable)

This provides a quick summary of how diversity changes across q-values, useful for q-curve visualization and statistical comparisons. Spectrum is saved as: `*_spectrum.tsv`

Parameter Priority Resolution:

q Priority 1 (explicit) > Priority 2 (`@config$q`) > Priority 3 (default: seq(0.01, 2, by=0.05)). Accepts single or multiple q-values (for spectrum computation).

nthreads Priority: explicit > `@config$nthreads` > 1

verbose Priority: explicit > `@config$verbose` > TRUE

bootstrap Priority: explicit > `@config$bootstrap` > FALSE

pseudocount Priority: explicit > `@config$pseudocount` > 0

norm Priority: explicit > `@config$norm` > TRUE

what Priority: explicit > `@config$what` > 'S' (Tsallis entropy)

Audit Trail: After execution, check:

- `analysis@config$last_diversity_run$parameters_used`: Actual parameters (not original `@config`)
- `attr(diversity(analysis, q=X), 'computed_with')`: Per-q-value metadata (timestamp, bootstrap setting, nthreads, etc.)

For additional details on diversity spectrum calculations and normalization methods, see the package vignettes.

Value

Modified TSENATAnalysis object with diversity results stored in `@diversity_results`, keyed by 'q_X. XX. . . ' format (e. g. , 'q_1. 000'). When `output_file` is provided, also generates:

- Primary file: Analysis object or table export
- Spectrum file: `*_spectrum.tsv` containing aggregated diversity statistics across q-values and groups

Examples

```
# Load vignette data and build analysis
data(readcounts)
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'

config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_dataset, metadata = metadata_df, config = config,
  tpm = tpm, effective_length = effective_length)

# Filter to manageable size (use 200+ genes to survive diversity filtering)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes
= 200)

# Compute diversity and access results using unified accessor
analysis <- calculate_diversity(analysis, q = c(0.5, 1.0), verbose =
FALSE)
head(results(analysis, type = 'diversity', q = 1.0))
```

`calculate_effect_sizes` *Compute Effect Sizes from Divergence Results (S4 Wrapper)*

Description

S4 wrapper for . `calculate_effect_sizes()` that extracts divergence and LM results directly from a TSENATAnalysis object.

Usage

```
calculate_effect_sizes(
  analysis,
  significance_threshold = NULL,
  enrich_per_q_pattern = NULL,
  verbose = NULL,
  output_file = NULL,
  ...
)
```

Arguments

`analysis`	`TSENATAnalysis`. An S4 object containing divergence results (from `calculate_divergence()`) and SAIT interaction results (from `calculate_sait()`).
`significance_threshold`	`numeric`. Adjusted p-value threshold for filtering significant genes (default: 0.05).
`enrich_per_q_pattern`	`logical`. If TRUE, enriches results with per-q divergence patterns (default: TRUE).
`verbose`	`logical`. If TRUE, print diagnostic messages (default: TRUE).
`output_file`	`character` or `NULL`. Optional file path to save results. Supported formats: .rds (for S4 objects). Default: NULL (no file output).
`...`	Additional arguments passed to the base function.

Details

Workflow steps:

Validating Input analysis object and required results

Extracting Divergence SE and SAIT results from analysis slots

Enriching Divergence SE with gene names via tx2gene or direct mapping

Computing Effect sizes using base `.calculate_effect_sizes()`

Storing Results in metadata with function call tracking

Parameter resolution priority (explicit > config > default):

- `significance_threshold`: Uses explicit arg, else `analysis@config$significance_threshold`, else 0.05

- enrich_per_q_pattern: Uses explicit arg, else analysis@config$enrich_per_q_pattern, else TRUE
- verbose: Uses explicit arg, else analysis@config$verbose, else TRUE

Results are accessed via: metadata(analysis)$effect_sizes_divergence

Value

Modified TSENATAnalysis with effect size results stored via metadata(analysis)$effect_sizes_divergence. Returns the analysis object visibly to support piping and method chaining.

See Also

calculate_divergence for divergence wrapper, calculate_sait for SAIT interaction wrapper

Examples

```
# Setup: Create test analysis with divergence and SAIT interaction results
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# Configure analysis parameters (best practice for reproducibility)
## Not run:
config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  subject_col = 'paired_samples',
  paired = TRUE,
  control = 'normal',
  q = seq(0, 2, by = 0.5)  # Multiple q-values for SAIT (5 unique values)
)
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_dataset, metadata = metadata_df, config = config,
  tpm = tpm, effective_length = effective_length)

analysis <- filter_analysis(analysis, stringency = 'severe')
analysis <- calculate_diversity(analysis)
analysis <- calculate_divergence(analysis)
analysis <- suppressWarnings(calculate_sait(analysis, method = 'gam'))

# Compute effect sizes from divergence results
analysis <- calculate_effect_sizes(analysis,
  significance_threshold = 0.05)

# Access results using unified results accessor
effect_size_results <- results(analysis, type = 'effect_sizes_divergence')

# View structure of results
str(effect_size_results, max.level = 1)
```

```
## End(Not run)
```

`calculate_jeo` *Jackknife resampling with confidence intervals*

Description

Jackknife resampling with confidence intervals

Usage

```
calculate_jeo(
  analysis,
  q = NULL,
  norm = NULL,
  log_base = NULL,
  top_n = NULL,
  verbose = NULL,
  nthreads = NULL,
  pseudocount = NULL,
  output_file = NULL,
  ...
)
```

Arguments

`analysis`	`TSENATAnalysis` object.
`q`	`numeric`. Q-value(s) for jackknife estimation. If NULL, reads from `analysis@config$q`. Default: c(0, 0.5, 1, 1.5, 2) (matches `calculate_jis` spectrum).
`norm`	`logical`. Normalization flag. Default: NULL (uses @config$norm or TRUE).
`log_base`	`numeric`. Logarithm base for entropy normalization. Default: NULL (uses e).
`top_n`	`numeric`. Number of top outlier samples to report. Default: 5.
`verbose`	`logical`. Print jackknife results summary. Default: FALSE.
`nthreads`	`numeric` or `NULL`. Number of CPU threads for parallel processing. If NULL, reads from `@config$nthreads` (or defaults to 1). If > 1 and multiple q-values provided, uses parallel PSOCK cluster.
`pseudocount`	`numeric`. Pseudocount value for count regularization. Default: NULL (uses @config$pseudocount or 0).
`output_file`	`character` or `NULL`. Optional file path to save results. Supported formats: .tsv, .csv, .txt (for jackknife results table with estimates, influence, outliers), .rds (for entire S4 object). Default: NULL (no file output).
`...`	Additional arguments passed to the base function.

Details

Requires diversity results to exist first. Will error if `calculate_diversity()` has not been run.

Parameter Priority Resolution:

nthreads Priority: explicit > `@config$nthreads` > 1

Value

Modified TSENATAnalysis with jackknife results in @jackknife_results.

Examples

```
# Load example data (matching TSENAT.Rmd workflow)
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# TPM and effective_length REQUIRED for filter_analysis()
tpm <- matrix(runif(nrow(readcounts) * ncol(readcounts), 0.1, 10),
              nrow = nrow(readcounts), ncol = ncol(readcounts),
              dimnames = dimnames(readcounts))
effective_length <- matrix(100, nrow = nrow(readcounts), ncol = ncol(readcounts))

# Create config (metadata passed as explicit parameter to build_analysis)
config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  q = seq(0, 2, by = 0.05),
  paired = FALSE
)

# Build analysis from vignette data - metadata as explicit parameter
analysis <- build_analysis(
  readcounts = readcounts,
  metadata = metadata_df,
  tx2gene = gff3_dataset,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)

# Filter low-abundance genes (required for reliable jackknife estimates)
analysis <- filter_analysis(analysis, stringency = 'severe')

# Compute diversity first (required for jackknife)
analysis <- calculate_diversity(analysis, q = c(0.5, 1.0, 1.5))

# Run jackknife estimation
analysis <- calculate_jeo(analysis, q = c(0.5, 1.0, 1.5))
# Check jackknife results using unified accessor
jackknife_res <- results(analysis, type = 'jackknife')
if (!is.null(jackknife_res)) names(jackknife_res)
```

`calculate_jis` *Jackknife isoform switching analysis on TSENATAnalysis object*

Description

Wrapper around .calculate_jis() that manages TSENATAnalysis object. Identifies transcripts with significant isoform switching patterns using jackknife resampling across samples to detect influential isoforms.

Usage

```
calculate_jis(
  analysis,
  condition_col = NULL,
  subject_col = NULL,
  gene_col = NULL,
  isoform_col = NULL,
  q = c(0, 0.5, 1, 1.5, 2),
  norm = NULL,
  log_base = NULL,
  threshold = 90,
  nboot = 1000,
  pseudocount = NULL,
  sait_results = NULL,
  sait_p_threshold = 0.05,
  use_sait_fdr = TRUE,
  nthreads = NULL,
  output_file = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

`analysis`	`TSENATAnalysis` object containing: • `@se`: SummarizedExperiment with count data • `@config`: Configuration metadata
`condition_col`	`character`. Column name in colData(se) specifying group assignments (default: 'sample_type'). If NULL, attempts auto-detection.
`subject_col`	`character`. Optional column for paired/repeated measures design. If provided, enables paired analysis. Default: NULL (unpaired).
`gene_col`	`character`. Column name in rowData(se) or metadata identifying genes. Default: 'gene'.
`isoform_col`	`character`. Column name in rowData(se) or metadata identifying isoforms/transcripts. Default: 'transcript' or 'isoform'.
`q`	`numeric`. Tsallis entropy parameter(s) to analyze. Can be single value or vector for multi-q analysis (default: c(0, 0.5, 1, 1.5, 2)). If NULL, uses @config$q.
`norm`	`logical`. Whether to use normalized diversity values (default: TRUE).

log_base	numeric. Logarithm base for entropy calculations. Default: NULL (uses e).
threshold	numeric. Percentile threshold for detecting transcript switching (default: 90). Transcripts with delta_influence >= threshold percentile are classified as 'switching'.
nboot	integer. Number of bootstrap resamples for confidence intervals (default: 1000).
pseudocount	numeric. Pseudocount value for count regularization. Default: NULL (uses @config$pseudocount or 0).
sait_results	data. frame. Optional SAIT interaction results to filter genes. If provided, only genes in sait_results are analyzed.
sait_p_threshold	numeric. P-value threshold for filtering genes from sait_results (default: 0.05).
use_sait_fdr	logical. If TRUE, uses adjusted p-values from sait_results (default: TRUE).
nthreads	numeric or NULL. Number of CPU threads for parallel processing. If NULL, reads from @config$nthreads (or defaults to 1). If > 1 and multiple q-values provided, uses parallel PSOCK cluster.
output_file	character or NULL. Optional file path to save results. Supported formats: .tsv, .csv, .txt (for tables), or .rds (for S4 objects). When text format is specified, generates TWO files: • output_file: Gene-level summary (one row per gene with switching statistics) • output_file_transcripts. ext: Transcript-level details (one row per transcript with p-values and FDR) For .rds format, saves only the full analysis object. Default: NULL (no file output).
verbose	logical. Print progress messages (default: FALSE).
...	Additional arguments for future extensibility.

Details

Key Features:

- Jackknife resampling: Robust outlier detection across all samples
- Delta-influence metric: Measures how much each isoform drives phenotype
- Confidence intervals: Bootstrap-based uncertainty quantification
- Multi-q analysis: Tests across full q-spectrum simultaneously
- LM filtering: Optional restriction to genes with significant interactions
- Paired designs: Supports repeated measures/longitudinal data

Parameter Resolution from Config

The following parameters are resolved using a three-level priority system:

1. User-provided argument (if not NULL)
2. Value from analysis@config (if key exists)
3. Function default value

Affected parameters:

- q: Multi-q vector c(0, 0.5, 1, 1.5, 2) if not provided, or `@config$q` if available
- nboot: Uses `@config$nboot` if available, else 1000
- threshold: Uses `@config$threshold` if available, else 90
- sait_p_threshold: Uses `@config$sait_p_threshold` if available, else 0.05

This allows setting defaults once in the config and reusing across multiple analyses.

Mathematical Background: Delta-influence measures how much removing each sample changes entropy:

```
Delta = H_q(leave-one-out) - H_q(original)
```

High |Delta| for specific isoforms indicates those isoforms drive differences. Identifies 'outlier samples' where isoforms contribute unusually much.

Example Use Case: Sample shows high Delta for isoform X → X has outsized importance in that sample
Classifying transcripts as 'switching' if top percentile (e.g., 90th) Delta
Reveals condition-specific isoforms crucial for phenotype determination.

Automatic Setup: This wrapper automatically: 1. Extracts SummarizedExperiment from `@se` slot 2. Detects condition_col, gene_col, isoform_col from colData/rowData or `@config` 3. Calls `.calculate_jis()` with extracted parameters

Parameter Auto-Detection:

1. condition_col: Uses explicit parameter, then @config, then 'sample_type'
2. gene_col: Uses explicit parameter, then looks for 'gene' or 'Gene'
3. isoform_col: Uses explicit parameter, then looks for 'transcript', 'isoform', or 'Isoform'

Value

`TSENATAnalysis` object with jackknife results stored in `@jackknife_results` slot. Results are keyed by q-value (e.g., 'q_1.00'). For multi-q analysis, multiple calls will accumulate results in the slot.

The analysis object is returned visibly to support method chaining:

```
analysis <- calculate_jis(analysis, q = 0.5)
analysis <- calculate_jis(analysis, q = 1.0)
```

See Also

`jackknife_isoform_switching` for the underlying implementation, `TSENATAnalysis` for object structure.

Examples

```
data(readcounts)
metadata_df <- read.table(system.file('extdata', 'metadata.tsv', package
= 'TSENAT'),
                          header = TRUE, sep = '\t')
gff3_file <- system.file('extdata', 'annotation.gff3.gz', package = 'TSENAT')
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
```

```
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_file, metadata = metadata_df, config = config,
                           tpm = tpm,
 effective_length = effective_length)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes
= 20, subset_n_samples = 8)
analysis <- calculate_diversity(analysis, q = 1)
```

`calculate_m_estimator` *M-Estimation for Sample Quality (S4 Wrapper)*

Description

S4 wrapper for `m_estimate` that performs robust M-estimation on diversity results stored in a TSENATAnalysis object and stores results back into the object.

Usage

```
calculate_m_estimator(
  analysis,
  condition_col = NULL,
  loss_type = "huber",
  scale = NULL,
  max_iter = 50,
  tol = 1e-06,
  paired = NULL,
  pcorr = "BH",
  q_combine_method = "mean",
  influence_threshold = 0.75,
  scale_method = "mad",
  output_file = NULL,
  verbose = FALSE
)
```

Arguments

`analysis`	`TSENATAnalysis` object with diversity results (typically via `calculate_diversity`).
`condition_col`	`character`. Column name in sample metadata indicating condition/sample grouping. Auto-detected from `@config$condition_col` if available.
`loss_type`	`character`. Type of loss function: 'huber' (default), 'tukey', or 'lsq'. Determines robustness vs efficiency trade-off.
`scale`	`numeric`. Manual scale parameter. If NULL, estimated from data.
`max_iter`	`integer`. Maximum iterations for M-estimation. Default: 50.
`tol`	`numeric`. Convergence tolerance. Default: 1e-6.
`paired`	`logical`. If TRUE, adjusts degrees of freedom for paired designs. Auto-detected from `@config$paired` if available. Default: FALSE.
`pcorr`	`character`. P-value correction method. Default: 'BH' (Benjamini-Hochberg).
`q_combine_method`	`character`. How to collapse multi-q results: 'mean' (default) or 'median'.

`influence_threshold`
: `numeric`. Threshold for classifying samples as high-influence. Default: 0.75.

`scale_method`
: `character`. Scale estimation method: 'mad' (default), 'proposal2', or 's-estimator'.

`output_file`
: `character` or `NULL`. Optional file path to save results. Supported formats: .rds (for S4 objects), .tsv, .csv, .txt (for tables). Default: NULL (no file output).

`verbose`
: `logical`. Print status messages. Default: TRUE.

Details

This wrapper extracts diversity results from `analysis@diversity_results`, performs robust M-estimation on diversity values (entropy), and stores results in the analysis object metadata.

M-Estimation: Robust regression technique that down-weights outliers based on their residuals. Useful for detecting low-quality samples that show unusual diversity patterns.

M-Estimation Results include:

- `sample_influence`: How much each sample affects the overall fit
- `robustness_weight`: Down-weighting factor
- `entropy_mean`: Average entropy for the sample
- `entropy_sd`: Entropy variability within the sample
- `Status`: QC Classification

Parameter resolution priority (explicit > @config > error):

- `samples`: Uses explicit arg, else `@config$condition_col`, else error. Note: despite parameter name 'samples', maps to condition grouping column

Data Requirements:

- Diversity results must be computed via `calculate_diversity()`
- Sample grouping column required in colData (auto-detected from @config$condition_col or via 'samples' parameter)

Value

Modified TSENATAnalysis object with M-estimation results stored in `analysis@metadata$m_estimate_results`. Contains data frame with influence scores, robustness weights, entropy statistics, and QC classifications. Returns visibly to support method chaining and piping.

See Also

`calculate_diversity` for computing diversity

Examples

```
# Create test analysis and compute M-estimation
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
```

```
'TSENAT')

# Create config (metadata passed as explicit parameter to build_analysis)
config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  q_values = seq(0, 2, by = 0.05),
  paired = FALSE
)

# Build analysis from vignette data
analysis <- build_analysis(
  readcounts = readcounts,
  tx2gene = gff3_dataset,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes = 200)
analysis <- calculate_diversity(analysis, q = c(0.5, 1.0, 1.5))
analysis <- calculate_m_estimator(
  analysis,
  condition_col = 'condition',
  loss_type = 'huber'
)
```

`calculate_rank_transform` *Detect q-dependent gene interactions*

Description

Wrapper around [.calculate_rank_transform()] that manages TSENATAnalysis object. Tests for genes with condition-specific q-dependent entropy patterns by testing whether the effect of q-values DIFFERS between experimental conditions. This detects disease-relevant or condition-specific isoform switching patterns.

Usage

```
calculate_rank_transform(
  analysis,
  condition_col,
  output_file = NULL,
  paired = NULL,
  subject_col = NULL,
  multicorr = c("hochberg", "benjamini-yekutieli", "westfall-young", "none"),
  entropy_col = "diversity",
  q_col = "q",
  gene_col = "gene",
  wy_randomizations = 500,
  nperm_mode = c("standard", "conservative", "interactive"),
```

```
  nthreads = NULL,
  alpha = 0.05,
  p_threshold = 0.05,
  eta2_threshold_moderate = 0.01,
  eta2_threshold_strong = 0.1,
  min_nperm = 100,
  max_nperm = 10000,
  verbose = FALSE,
  method = c("art", "rt"),
  ...
)
```

Arguments

analysis	TSENATAnalysis object. Must have diversity results from calculate_diversity().
condition_col	character. Column name for sample grouping/condition (REQUIRED). Specifies the condition/treatment variable for testing q× condition interactions. Example: 'sample_type', 'treatment', 'disease_status'.
output_file	character or NULL. Optional file path to save results. Supported formats: .rds (for S4 objects). Default: NULL (no file output).
paired	logical or NULL. If TRUE, uses paired/blocked design (requires subject_col). If NULL, reads from @config$paired.
subject_col	character or NULL. Column name for subject/block identifiers (required when paired=TRUE). If NULL, reads from @config$subject_col.
multicorr	character. Multiple testing correction: 'hochberg' (default), 'benjamini-yekutieli', 'westfall-young', or 'none'.
entropy_col	character. Column name containing entropy/diversity data. Default: 'diversity'.
q_col	character. Column name containing q-values. Default: 'q'.
gene_col	character. Column name containing gene identifiers. Default: 'gene'.
wy_randomizations	numeric or character. Number of permutations for Westfall-Young correction. Use 'auto' to estimate from data. Default: 500.
nperm_mode	character. Mode for automatic permutation estimation: 'standard' (default), 'conservative', or 'interactive'.
nthreads	numeric or NULL. Number of parallel threads for computation. If NULL, reads from @config$nthreads.
alpha	numeric. Significance level for p-value correction methods (default: 0.05). Used by all multiple testing correction methods.
p_threshold	numeric. P-value threshold for classification of interaction significance (default: 0.05).
eta2_threshold_moderate	numeric. Effect size boundary for 'moderate' classification (default: 0.01).
eta2_threshold_strong	numeric. Effect size boundary for 'strong' classification (default: 0.10).
min_nperm	integer. Minimum permutations for automatic estimation when wy_randomizations='auto' (default: 100).

max_nperm	integer. Maximum permutations for automatic estimation when wy_randomizations='auto' (default: 10000).
verbose	logical. If TRUE, prints progress messages. Default: FALSE.
method	character. Non-parametric test method: • 'art' (default): Aligned Rank Transform via ARTool package. State-of-the-art for non-parametric interaction testing. Properly handles factorial interactions by stripping main effects before ranking (Higgins & Tashtoush 1994, Wobbrock et al. 2011). • 'rt': Conover-Iman Rank Transform. Legacy non-parametric procedure. Valid for main effects but has known limitations for interaction testing (Conover & Iman 1981). ART is strongly recommended.
...	Additional arguments passed to the base .calculate_rank_transform() function.

Details

Key Features

- **Q× Condition Interaction**: Tests if entropy patterns across q-values differ by condition (main discovery goal) - **Multi-q Analysis**: Combines diversity results for multiple q-values into a single SummarizedExperiment for joint hypothesis testing - **Aligned Rank Transform (ART)**: State-of-the-art non-parametric interaction testing via ARTool package (default). Strips main effects before ranking to preserve interaction structure (Higgins & Tashtoush 1994, Wobbrock et al. 2011). - **Conover-Iman Rank Transform**: Two-way non-parametric ANOVA on ranks (fallback via 'method='rt'') - **Multiple Testing Correction**: Hochberg, Benjamini-Yekutieli, or permutation (Westfall-Young) procedures - **AR(1) Correlation Handling**: Westfall-Young preserves q-value spatial correlations (important for ordered q measurements) - **Effect Sizes**: Eta-squared (η^2) for q× condition interactions

Statistical Hypotheses

Tests the null hypothesis: - **H*0*** = Gene entropy q-effect does NOT differ between conditions (q-independent) - **H*1*** = Gene entropy q-dependence is CONDITION-SPECIFIC (interaction exists)

A significant interaction indicates condition-specific patterns in how entropy varies across the q-value spectrum, revealing biological processes specific to that condition.

Biological Example

Gene shows strong isoform switching (q-dependent entropy) in tumor cells but NOT in healthy cells -> Identified as disease-relevant q-dependent gene.

For condition-specific q-dependent genes: - **Condition A**: Strong entropy variation across q (q-dependent isoform usage) - **Condition B**: Flat entropy profile across q (uniform isoform usage) - **Interaction**: Condition-specific q-dependence pattern reveals disease-associated splicing regulation

Analyzes how gene interactions change across q-value spectrum using rank-based (Conover-Iman Rank Transform) statistical tests.

Parameter resolution priority (explicit > @config > default/auto-detect):

- condition_col: REQUIRED - must be explicitly provided
- q: ALWAYS auto-detected from diversity_results (all q-values tested together)
- paired: explicit arg > @config$paired > FALSE (default)
- subject_col: explicit arg > @config$subject_col
- multicorr: explicit arg > @config$multicorr > 'hochberg'

- nthreads: explicit arg > @config$nthreads > 1 (default)
- nperm_mode: explicit arg > @config$nperm_mode > 'standard'

Value

Modified TSENATAnalysis with interaction results in @rank_test_results.

Examples

```
# Load example data (matching TSENAT.Rmd workflow)
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# Create config first (required when metadata is provided)
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')

# Build analysis from vignette data and create manageable subset
analysis <- build_analysis(
  readcounts = readcounts,
  tx2gene = gff3_dataset,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)
analysis <- filter_analysis(
  analysis,
  min_samples = 1,
  subset_n_genes = 200
)
analysis <- calculate_diversity(analysis, q = c(0.5, 1.0, 1.5))

# Test Q\eqn{\times} Condition interaction (condition_col is REQUIRED)
analysis <- calculate_rank_transform(
  analysis,
  condition_col = 'condition',
  multicorr = 'hochberg'
)
# View results using unified accessor
rank_test_res <- results(analysis, type = 'rank_test')
if (!is.null(rank_test_res)) head(rank_test_res)
```

`calculate_sait` *Calculate SAIT interactions and store in TSENATAnalysis*

Description

Statistical interaction testing for entropy diversity using flexible scale-adaptive models (GAM, LMM, GEE, FPCA) with AR(1) correlation structure for repeated measures. Tests for significant q-by-condition interactions across entropic indices.

Usage

```
calculate_sait(
  analysis,
  fdr_threshold = NULL,
  formula = NULL,
  condition_col = NULL,
  method = "gam",
  paired = NULL,
  subject_col = NULL,
  nthreads = NULL,
  multicorr = NULL,
  corstr = NULL,
  pcorr = NULL,
  verbose = NULL,
  return_model_data = NULL,
  output_file = NULL,
  ...
)
```

Arguments

analysis	TSENATAnalysis object.
fdr_threshold	numeric. FDR cutoff for significance. Default: 0.05.
formula	formula or NULL. Reserved for future use.
condition_col	character or NULL. Column name in colData identifying sample conditions. If NULL, reads from @config$condition_col or auto-detects common column names.
method	character. Statistical method (e. g. , 'lmm', 'gam', 'gee'). If NULL, uses method from @config$method or defaults to 'lmm'. For paired designs, 'gam' dispatches to GAMM: nlme::lme with regression splines (ns(q, df = 3) x condition), subject random intercept and AR(1) within subject x condition (marginal F-test for the interaction); mgcv::gamm() and standard GAM are fallbacks.
paired	logical. Whether to use paired design. Default: FALSE. If not specified, reads from @config$paired if available.
subject_col	character or NULL. Column name identifying subject IDs for paired designs. If NULL, reads from @config$subject_col if available.
nthreads	numeric or NULL. Number of CPU threads for parallel processing. If NULL, reads from @config$nthreads (or defaults to NULL, letting base function decide).
multicorr	character or NULL. Multiple comparison correction method. Options: 'hochberg', 'westfall-young', 'benjamini-yekutieli'. If NULL, uses method from @config or base function defaults.

corstr	character or NULL. Correlation structure for GEE models. Options: 'ar1', 'exchangeable', 'independence', 'auto'. 'auto' selects the best structure via QIC. If NULL, uses method from @config or base function defaults.
pcorr	character or NULL. P-value correction method. Default: 'BH' (Benjamini-Hochberg). If NULL, reads from @config$pcorr if available.
verbose	logical. Print progress messages. Default: FALSE.
return_model_data	logical. Return model data for visualization. Default: TRUE.
output_file	character or NULL. Optional file path to save results. Supported formats: .rds (for S4 objects), .tsv, .csv, .txt (for tables). Default: NULL (no file output).
...	Additional arguments passed to the base LM function, including: pvalue, min_obs, assay_name, bias_correction, regularization, storey, wy_randomizations, adaptive_knots, block_col, strata_col, permutation_scheme (exchangeability-compatible Westfall-Young permutations), etc.

Details

Extracts diversity results from @diversity_results (prerequisite), combines across q-values into single SummarizedExperiment, then runs `.calculate_sait()`.

Parameter Priority Resolution:

- nthreads: Priority: explicit > @config > NULL

Parameters are resolved in priority order: 1. Explicit arguments passed to function 2. Values from analysis@config (if present) 3. Function defaults

Value

Modified TSENATAnalysis with results in @sait_results$sait_interaction.

Examples

```
## Not run:
# Create test analysis with appropriate sample structure for paired design
# Note: requires lme4 package for LMM fitting; uses synthetic data
set.seed(42)

# Create transcript-level counts with biological signal
# Note: Use adequate complexity (transcripts/genes, samples, expression)
# to avoid filtering away all genes during diversity computation
n_genes <- 50
n_transcripts_per_gene <- 30
n_transcripts <- n_genes * n_transcripts_per_gene
n_samples <- 16  # 8 subjects x 2 conditions (paired design)

# Generate counts with clear biological signal
control_idx <- seq(1, n_samples, by = 2)
treatment_idx <- seq(2, n_samples, by = 2)

counts <- matrix(0, nrow = n_transcripts, ncol = n_samples)
for (j in seq_len(n_samples)) {
  lambda <- if (j %in% control_idx) 100 else 180
  counts[, j] <- rpois(n_transcripts, lambda = lambda)
}
```

```
counts <- pmax(counts, 50)  # Ensure minimum expression

rownames(counts) <- paste0('TX_', seq_len(n_transcripts))
colnames(counts) <- paste0('Sample_', seq_len(n_samples))

# Create rowData with gene mapping (tx2gene structure)
rowdata <- data.frame(
  transcript_id = rownames(counts),
  gene_id = rep(paste0('GENE_', 1:n_genes),
                each = n_transcripts_per_gene),
  row.names = rownames(counts)
)

# Create colData with paired design metadata
coldata <- data.frame(
  sample_id = colnames(counts),
  condition = rep(c('control', 'treatment'),
                  length.out = n_samples),
  subject = rep(paste0('Subject_', 1:8),
                length.out = n_samples),
  row.names = colnames(counts)
)

# Build SummarizedExperiment
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(counts = counts),
  rowData = S4Vectors::DataFrame(rowdata),
  colData = S4Vectors::DataFrame(coldata)
)

# Add tx2gene metadata for gene-level aggregation
S4Vectors::metadata(se)$tx2gene <-
  data.frame(Transcript = rowdata$transcript_id,
             Gene = rowdata$gene_id)

# Initialize TSENATAnalysis
analysis <- TSENATAnalysis(se = se, config = list())

# Compute diversity (prerequisite for SAIT interaction analysis)
analysis <- calculate_diversity(
  analysis,
  q = c(0.5, 1.0, 1.5, 2.0, 2.5)
)

# Calculate q x condition interactions using GAM
analysis <- suppressWarnings(calculate_sait(
  analysis,
  condition_col = 'condition',
  method = 'gam'
))

# View top interaction results using unified accessor (first 3 genes)
res <- results(analysis, type = "sait")
if (!is.null(res)) head(res, 3)

## End(Not run)
```

`filter_analysis` *Filter Low-Abundance Transcripts in a TSENATAnalysis Object*

Description

S4 wrapper for `.filter_se()` that filters low-abundance transcripts directly within a `TSENATAnalysis` object. This maintains the consistent S4 workflow pattern where functions accept and return analysis objects.

Usage

```
filter_analysis(
  analysis,
  min_tpm = 1,
  tpm_assay_name = NULL,
  min_samples = 5L,
  stringency = NULL,
  pair_col = NULL,
  min_tx_per_gene = 2L,
  min_isoform_abundance = NULL,
  assay_name = "counts",
  subset_n_genes = NULL,
  subset_genes = NULL,
  subset_n_samples = NULL,
  subset_samples = NULL,
  subset_select_by = c("variance", "mean", "random"),
  subset_seed = 42,
  subset_min_count = NULL,
  verbose = FALSE
)
```

Arguments

`analysis`	A `TSENATAnalysis` S4 object containing the `SummarizedExperiment` to be filtered.
`min_tpm`	Numeric TPM threshold (default 1.0). Keeps transcripts with TPM >= `min_tpm` in >= `min_samples` samples. Ignored if `stringency` is specified.
`tpm_assay_name`	Character; name of assay containing TPM data (default: NULL). If NULL, searches for TPM assay automatically.
`min_samples`	Numeric. Minimum number of samples in which a transcript must be present (default: 5). Ignored if `stringency` is specified.
`stringency`	Character. Filtering stringency level: 'soft' (permissive), 'medium' (balanced), or 'severe' (stringent). When specified, auto-estimates: `min_samples`, `min_tpm`, `min_tx_per_gene`, and `min_isoform_abundance` from data. Requires `pair_col` in colData for paired designs. User-provided values for any parameter override stringency defaults. Default: 'medium' (balanced filtering recommended for most analyses).
`pair_col`	Character; column name in colData containing pair IDs for paired designs. Default: NULL (auto-detect if needed).

`min_tx_per_gene`
: Integer minimum number of transcripts per gene required (default 2L). Single-transcript genes are always kept. Ignored if `stringency` is specified; when specified, automatically adjusted based on stringency level.

`min_isoform_abundance`
: Numeric in [0, 1]; minimum relative abundance threshold for isoforms within each gene. Implements Soneson et al. (2016) filtering. Default behavior: - If `stringency` is specified: uses stringency-based default (soft: 0. 01, medium: 0. 05, severe: 0. 15) - If `stringency` is NULL: uses default 0.05 (5 - If explicitly provided: overrides any stringency default Set to 0 or NULL (post-stringency processing) to skip isoform-level filtering.

`assay_name`
: Character; name or index of the assay to use for filtering (default: 'counts'). Deprecated: use `tpm_assay_name` instead.

`subset_n_genes`
: Integer; optional number of genes to retain after filtering. If provided, genes are selected based on `subset_select_by`. Default: NULL.

`subset_genes`
: Character vector; optional specific genes to retain after filtering. Default: NULL.

`subset_n_samples`
: Integer; optional number of samples to retain after filtering. If provided, samples are selected (balanced by condition if available). Default: NULL.

`subset_samples`
: Character vector; optional specific samples to retain after filtering. Default: NULL.

`subset_select_by`
: Character; gene selection method for `subset_n_genes`: 'variance' (highest variance), 'mean' (highest mean expression), or 'random'. Default: 'variance'.

`subset_seed`
: Integer; random seed for reproducibility when `subset_select_by = 'random'`. Default: 42.

`subset_min_count`
: Numeric; optional minimum count threshold applied during subsetting. Default: NULL.

`verbose`
: Logical. If TRUE, print filtering progress and summary statistics (default: FALSE).

Details

This wrapper applies `.filter_se()` to the SummarizedExperiment within the TSENATAnalysis object, optionally followed by subsetting parameters. The filtering and subsetting operations are applied in sequence:

1. Extracts the SE from `analysis@se` 2. Filters using `.filter_se()` with specified filtering parameters (default: 'medium' stringency) 3. If any subset parameters are provided, applies gene/sample selection to select specific genes and/or samples 4. Stores the filtered/subsetted SE back in `analysis@se` 5. Returns the modified analysis object invisibly

Default Filtering (stringency = 'medium'): By default, filtering applies balanced stringency: requires transcripts in >= 50 isoform abundance of 5 noise reduction with preservation of isoform diversity for reliable entropy calculations.

Important: Filtering should be performed BEFORE computing diversity, divergence, or SAIT interaction results. If called after analysis results have been computed, those results will be based on unfiltered data and may not align with the filtered SE dimensions.

Value

Invisibly returns the modified `analysis` object with filtered `SummarizedExperiment` in the `@se` slot. The filtering operation modifies the analysis object in-place while maintaining all other slots (results, metadata, etc.).

See Also

`build_analysis` for creating a new analysis object

Examples

```
# Create test analysis and filter
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_dataset, metadata = metadata_df, config = config,
  tpm = tpm, effective_length = effective_length)
analysis <- filter_analysis(analysis, stringency = 'medium')
```

---

metadata,TSENATAnalysis-method

Metadata Accessor Methods

Description

Access or set the metadata list stored in a TSENATAnalysis object. The getter function retrieves all metadata or a specific key-value. The setter function replaces the entire metadata list.

Usage

```
## S4 method for signature 'TSENATAnalysis'
metadata(x, key = NULL)

## S4 replacement method for signature 'TSENATAnalysis'
metadata(x) <- value
```

Arguments

`x`	A `TSENATAnalysis` object
`key`	Optional character string specifying a metadata key to retrieve
`value`	A list of metadata to assign

Details

Get or Set Metadata

Value

`metadata` Returns the full metadata list, or a single value if `key` is specified

`metadata<-` Returns the modified `TSENATAnalysis` object

Examples

```
# Create a TSENATAnalysis object
library(SummarizedExperiment)
se <- SummarizedExperiment(assays = list(counts = matrix(1:100, nrow = 10)))
analysis <- new('TSENATAnalysis', se = se, config = list())

# Get metadata (empty by default)
metadata(analysis)


# Set metadata
metadata(analysis) <- list(processing_date = Sys.Date(), method = 'test')

# Retrieve all metadata
metadata(analysis)

# Retrieve specific metadata key
metadata(analysis, key = 'method')
```

---

plot_concordance *Plot method concordance results from TSENATAnalysis*

Description

Plot method concordance results from TSENATAnalysis

Usage

```
plot_concordance(analysis, verbose = FALSE)

## S4 method for signature 'TSENATAnalysis'
plot_concordance(analysis, verbose = FALSE)
```

Arguments

`analysis`	`TSENATAnalysis` object with computed method concordance (from `calculate_concordance()`).
`verbose`	`logical`. Print progress messages. Default: FALSE

Details

Creates visualization of method concordance including: - Comparison of significance across two methods (with color-coded agreement) - P-value distribution histograms for both methods - Significance threshold lines at $p < 0.05$

Requires that `calculate_concordance()` has already been run to populate `@metadata$method_concordance`.

Value

A ggplot/cowplot object showing:

Panel 1 Scatter plot of -log10(p-values) comparing methods

Panel 2 Histogram of p-value distributions by method

Examples

```
# Load example data (matching TSENAT.Rmd workflow)
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# Build analysis from vignette data and create small subset
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_dataset, metadata = metadata_df, config = config,
  tpm = tpm, effective_length = effective_length)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes
= 200)

# Note: calculate_concordance requires additional LM and Conover-Iman Rank Transform
# results computed. For demo purposes, we show that
# plot_concordance needs pre-computed concordance in @metadata
```

---

plot_divergence_distribution

Plot Tsallis Divergence Effect Size Distribution (S4 Wrapper)

Description

S4 wrapper that extracts effect size results from a TSENATAnalysis object and generates a histogram visualization of Tsallis divergence effect sizes across genes.

Usage

```
plot_divergence_distribution(
  analysis,
  threshold = 0.1,
  output_file = NULL,
  width = 12,
  height = 6,
  verbose = FALSE,
  ...
)
```

Arguments

analysis	TSENATAnalysis object with effect sizes computed (typically via calculate_effect_sizes).
threshold	numeric. Effect size threshold for visual marking in the plot. Default is 0.1 (information-theoretic significance level).
output_file	character. Optional file path to save the plot. If NULL, plot is returned but not saved.
width	numeric. Plot width in inches. Default is 10.
height	numeric. Plot height in inches. Default is 6.
verbose	logical. Print status messages. Default is TRUE.
...	Additional arguments passed to the underlying plotting function.

Details

This wrapper extracts the interaction results (with effect size columns) from analysis@metadata$effect_sizes_diver and passes them to the base .plot_divergence_distribution() function.

The function visualizes the distribution of effect sizes using the median q-value's divergence (typically around q=1.0, close to Shannon entropy). A red dashed line marks the information-theoretic significance threshold.

Data Requirements:

- Effect sizes must be computed via calculate_effect_sizes()
- @metadata$effect_sizes_divergence$interaction_results must contain columns matching pattern effect_size_D_q*

Value

Invisibly returns the file path if saved, otherwise the ggplot object. If the plot cannot be created (missing data, ggplot2 not available), returns NULL invisibly with an informative message.

See Also

calculate_effect_sizes for computing effect sizes.

Examples

```
# Plot 2: Distribution of effect sizes across genes
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
```

```
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')
# Configure analysis parameters first
config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  control = 'normal'
)

# Build analysis with configured parameters
analysis <- build_analysis(
  readcounts = readcounts,
  tx2gene = gff3_dataset,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)

analysis <- filter_analysis(analysis, stringency = 'severe')
analysis <- calculate_diversity(
  analysis,
  q = seq(0.2, 2, by = 0.4),
  verbose = FALSE
)
analysis <- calculate_divergence(
  analysis,
  q = seq(0.2, 2, by = 0.4),
  verbose = FALSE
)
analysis <- suppressWarnings(calculate_sait(analysis, method = 'lmm'))
analysis <- calculate_effect_sizes(analysis)
p_dist <- plot_divergence_distribution(analysis)
# print(p_dist)
```

`plot_divergence_spectrum` *Plot Global Divergence q-Curve Across All Genes (S4 Wrapper)*

Description

S4 wrapper that extracts divergence results from a TSENATAnalysis object and visualizes the average Tsallis divergence across all genes (or specified genes) as a function of q-value.

Usage

```
plot_divergence_spectrum(
  analysis,
  gene = NULL,
```

```
  n_genes = 4,
  ncol = 2,
  metric = c("median", "mean"),
  variability_metric = c("iqr", "sd"),
  use_pvalue_ranking = FALSE,
  output_file = NULL,
  width = 12,
  height = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

analysis	TSENATAnalysis object with divergence results (typically via calculate_divergence).
gene	character. Optional specific gene name to plot. If NULL, plots global divergence curve (aggregated across all genes).
n_genes	integer. Number of top genes to plot when showing multi-gene spectra. Default is 4. Genes are sorted by p-value significance.
ncol	integer. Number of columns in grid layout for multi-gene plots. Default is 2. Number of rows is automatically calculated.
metric	character. Summary statistic for global curve: 'median' (default) or 'mean'. Only used when gene = NULL.
variability_metric	character. Error bar type for global curve: 'iqr' (default) or 'sd'. Only used when gene = NULL.
use_pvalue_ranking	logical. If TRUE, uses SAIT results to rank and display top n_genes by p-value significance. If FALSE (default), plots global divergence curve when gene = NULL. Default is FALSE.
output_file	character. Optional file path to save the plot. If NULL, plot is returned but not saved.
width	numeric. Plot width in inches. Default is 10.
height	numeric. Plot height in inches. Default is 6.
verbose	logical. Print status messages. Default is TRUE.
...	Additional arguments passed to the underlying plotting function.

Details

This wrapper extracts the divergence SummarizedExperiment from analysis@divergence_results and optionally the SAIT results from analysis@sait_results$sait_interaction to pass to the base function.

Data Requirements:

- Divergence must be computed via calculate_divergence()
- @divergence_results$divergence_se or direct divergence SE

Modes:

- **Global mode** (gene = NULL): Shows median/mean divergence across all genes with variability bands

- **Gene-specific mode** (gene specified): Shows divergence spectrum for a single named gene
- **Top genes mode** (gene = NULL, sait_res provided): Shows top n_genes by significance

Value

Invisibly returns the file path if saved, otherwise the ggplot object. If the plot cannot be created (missing data, ggplot2 not available), returns NULL invisibly with an informative message.

See Also

calculate_divergence for computing divergence.

Examples

```
# Load example data (matching TSENAT.Rmd workflow)
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# Build analysis from vignette data and create small subset
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_dataset, metadata = metadata_df, config = config,
  tpm = tpm, effective_length = effective_length)
analysis <- filter_analysis(
  analysis,
  min_samples = 1,
  subset_n_genes = 200
)
analysis <- calculate_diversity(
  analysis,
  q = c(0.5, 1, 1.5),
  verbose = FALSE
)
analysis <- calculate_divergence(
  analysis,
  q = c(0.5, 1, 1.5),
  verbose = FALSE
)
p_global <- plot_divergence_spectrum(analysis)
# print(p_global)
```

`plot_diversity_spectrum`

Plot Tsallis Entropy q-Curve

Description

Visualize Tsallis entropy (S_q) as a function of the diversity parameter q across sample groups. Supports three modes: aggregate q-curves (default), gene-specific q-curves (when ‘gene‘ provided), or bootstrap confidence interval bands.

Usage

```
plot_diversity_spectrum(
  se,
  assay_name = "diversity",
  condition_col = NULL,
  gene = NULL,
  sait_res = NULL,
  n_top = NULL,
  metric = "iqr",
  output_file = NULL,
  dev_width = NULL,
  dev_height = NULL
)
```

Arguments

se	A ‘SummarizedExperiment‘ returned by ‘.calculate_diversity()‘ with diversity assay. If pre-computed bootstrap confidence intervals are available (ci_lower/ci_upper assays), they will be displayed automatically. Otherwise, falls back to IQR visualization.
assay_name	Character; name of the assay to plot (default: ’diversity’).
condition_col	Character or NULL; column name in colData indicating group/sample type. If NULL (default), reads from ‘@config$condition_col‘ when input is TSENAT-Analysis, otherwise defaults to ’sample_type’. Only used in aggregate and CI modes.
gene	Character vector (optional); if provided, plot q-curves for specified gene(s). Overrides default aggregate behavior. When provided, uses median +/- SD for each gene.
sait_res	Data frame (optional); gene interaction test results with ‘gene‘ column and p-value column. Accepts either: - Results from ‘.calculate_sait()‘ (has ‘adj_p_interaction‘ or ‘p_interaction‘ columns) - Results from ‘.calculate_rank_transform()‘ (has ‘adj_p_value‘ or ‘p_value‘ columns from Conover-Iman Rank Transform tests) If provided (and ‘gene‘ is NULL), plots top ‘n_top‘ genes ranked by p-value. Useful for plotting significant genes from any interaction analysis.
n_top	Integer or NULL; number of top genes to select from ‘sait_res‘ when ‘gene‘ is NULL (default: NULL). When NULL, defaults to showing the single most significant gene (n_top=1), providing a conservative view of the strongest effect. Set to a numeric value to show that many top genes.
metric	Character; when bootstrap CIs are NOT available, specifies the spread metric to display. Options: ’iqr’ (default, Interquartile Range - more robust) or ’sd’ (Standard Deviation). This parameter is ignored when bootstrap confidence intervals are available. Default: ’iqr’.
output_file	character or NULL. Optional file path to save the plot. Default: NULL (no file output).

dev_width	Numeric or NULL; width in inches for the graphics device when displaying the plot interactively. If specified (along with dev_height), creates a new device with this width. Default: NULL (uses current device).
dev_height	Numeric or NULL; height in inches for the graphics device when displaying the plot interactively. If specified (along with dev_width), creates a new device with this height. Default: NULL (uses current device).

Details

Aggregate mode (default, gene=NULL, sait_res=NULL): - Plots median Tsallis entropy +/- IQR across all genes for each group - Works with any SummarizedExperiment from .calculate_diversity() - Supports single or multiple q values and any number of groups - No CI data required for basic plots; bootstrap CIs optional

Gene-specific mode (gene or sait_res provided): - Plots q-curve separately for each selected gene - Shows median entropy +/- SD (variance) for each gene across q-values and groups - When 'sait_res' provided: automatically ranks genes and selects top 'n_top' by p-value - Single gene: returns a ggplot object; multiple genes: returns a grid plot (2 rows x 2 columns) with shared legend - For multiple genes: legend appears once at the bottom of the grid to avoid repetition and save space - Useful for highlighting specific genes of interest or significant discoveries - Bootstrap mode not supported in gene-specific mode

Automatic Bootstrap CI Detection: - When computing divergence or diversity with bootstrap enabled, ci_lower and ci_upper assays are added to the SummarizedExperiment. - This function automatically detects these assays and displays bootstrap confidence interval bands instead of IQR. No additional parameter needed. - For confidence bands to appear, use 'calculate_diversity(..., bootstrap=TRUE, nboot=1000)' or appropriate divergence function with bootstrap enabled.

Value

Aggregate mode (gene=NULL, sait_res=NULL): - If bootstrap CI assays available: A ggplot object showing median entropy with bootstrap confidence interval bands. - If no CI assays: A ggplot object showing median entropy with IQR ribbons (automatic fallback).

Gene-specific mode (gene or sait_res provided): - Single gene: A ggplot object showing median entropy +/- SD for that gene. - Multiple genes: A grid plot object arranged in 2 rows x 2 columns with a shared legend at the bottom. The legend appears once beneath the grid, avoiding repetition across subplots.

Examples

```
# Plot 7: Tsallis entropy q-curve (combined across all sample diversity)
data(readcounts)
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'

# Create configuration (required when metadata is provided)
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_dataset, metadata = metadata_df, config = config,
```

```
  tpm = tpm, effective_length = effective_length)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes
= 200)
analysis <- calculate_diversity(analysis, q = seq(0, 2, by = 0.5),
)
p <- plot_diversity_spectrum(analysis)
# print(p)
```

`plot_diversity_violin_density` *Combined Violin and Density Plot Grid for Single q Value*

Description

Creates a side-by-side grid layout with a violin plot on the left and a density plot on the right, both showing Tsallis entropy distribution for the q value in the provided SummarizedExperiment (which should contain a single q value).

Usage

```
plot_diversity_violin_density(
  se,
  assay_name = "diversity",
  title = NULL,
  output_file = NULL,
  q = NULL
)
```

Arguments

`se`	A 'SummarizedExperiment' returned by 'calculate_diversity' containing entropy values at a single q value.
`assay_name`	Name of the assay to use (default: 'diversity').
`title`	Optional base title. If NULL, auto-generated based on q value.
`output_file`	Character or NULL. Optional file path to save the plot as an image. If provided, the plot will be saved with appropriate dimensions. Default: NULL (no file output, only return object).
`q`	`numeric` or `NULL`. Single q value to plot. If NULL, the q value must be unambiguous from the stored diversity results (exactly one q); otherwise an error is raised listing the available q values.

Value

A 'ggplot2' object showing a 1x2 grid with violin plot on the left and density plot on the right.

Examples

```
# Plot 8: Violin and density plots of Tsallis entropy distribution
data(readcounts)
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'

# Create configuration (required when metadata is provided)
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_dataset, metadata = metadata_df, config = config,
  tpm = tpm, effective_length = effective_length)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes
= 200)
analysis <- calculate_diversity(analysis, q = 1.0)
p <- plot_diversity_violin_density(analysis)
# print(p)
```

`plot_expression` *Plot Top Transcripts from TSENATAnalysis Object*

Description

S4 wrapper for . `plot_expression()` that extracts data directly from a TSENATAnalysis object. Automatically retrieves the SummarizedExperiment and SAIT results for visualizing transcript abundance across conditions.

Usage

```
plot_expression(
  analysis,
  gene = NULL,
  condition_col = NULL,
  top_n = 4,
  output_file = NULL,
  metric = c("median", "mean", "variance", "iqr"),
  use_tpm = TRUE,
  quantity = c("abundance", "usage"),
  width = NULL,
  height = NULL,
  fontsize = 16,
  cellwidth = 0,
  cellheight = 0,
  layout_ncol = 2,
  verbose = FALSE,
  ...
)
```

Arguments

analysis	TSENATAnalysis. An S4 object containing a processed SummarizedExperiment and optional SAIT interaction results.
gene	character or NULL. Gene identifier(s) to plot. If a vector of multiple genes is provided, plots all of them. If NULL, automatically selects the top genes from SAIT results based on top_n parameter (genes with lowest p-values). Default: NULL (auto-extract from sait_results).
condition_col	character. Column name in colData(se) specifying group assignments (default: 'sample_type').
top_n	numeric. Number of top transcripts to display for each condition (default: 3).
output_file	character or NULL. Optional file path to save the plot. Supported formats: .pdf, .png, .jpg. Default: NULL (no file output).
metric	character. Method for ranking transcripts within genes. One of 'median', 'mean', 'variance', or 'iqr' (default: 'median').
use_tpm	logical. If TRUE, uses TPM (Transcripts Per Million) from metadata instead of raw counts (default: FALSE). TPM is normalized for sequencing depth and is recommended for comparing expression across samples. Requires TPM data in metadata from 'build_analysis()' or '.build_se()' with 'tpm' parameter. Raises error if TPM not available and 'use_tpm = TRUE'.
quantity	character. Quantity to visualize: '"abundance"' plots raw counts (or TPM when 'use_tpm = TRUE'), '"usage"' plots each transcript's proportion of its gene total per sample (default: '"abundance"').
width	numeric or NULL. Output image width in inches. If NULL, automatically calculated based on number of genes (default: ~13 inches per column).
height	numeric or NULL. Output image height in inches. If NULL, automatically calculated based on number of genes (default: ~10 inches per row + headers).
fontsize	numeric. Base font size for heatmap titles and labels (default: 16pt). Automatically scaled for readability.
cellwidth	numeric. Width of individual heatmap cells in pixels. If 0 (default), uses adaptive sizing based on data dimensions and layout. Set > 0 to override dynamic sizing.
cellheight	numeric. Height of individual heatmap cells in pixels. If 0 (default), uses adaptive sizing based on data dimensions and layout. Set > 0 to override dynamic sizing.
layout_ncol	numeric. Number of heatmaps per row in fixed layout (default: 2). If NULL, uses adaptive layout based on transcript counts.
verbose	logical. If TRUE, print diagnostic messages during plotting (default: FALSE).
...	Additional arguments passed to the base plotting function.

Details

This wrapper extracts the following from analysis:

SummarizedExperiment From analysis@se containing transcript counts

SAIT results From analysis@sait_results$sait_interaction for gene selection

If no gene is specified, the function automatically selects the top gene from the SAIT results (lowest p-value). This simplifies visualization of genes with significant q x condition interaction effects.

Value

Invisibly returns the output file path (if ‘output_file‘ provided), or invisible(NULL) if rendering to active graphics device. Graphics are rendered to the active grid device for capture during vignette compilation.

See Also

TSENATAnalysis for object structure

Examples

```
# Plot 6: Top transcripts across groups
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')
# Configure analysis parameters first
config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  subject_col = 'paired_samples',
  paired = TRUE,
  control = 'normal',
  q_values = seq(0, 2, by = 0.1)
)

# Build analysis with configured parameters
analysis <- build_analysis(
  readcounts = readcounts,
  tx2gene = gff3_dataset,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)

analysis <- filter_analysis(analysis, stringency = 'severe')
analysis <- calculate_diversity(
  analysis,
  q = seq(0.2, 2, by = 0.4),
  verbose = FALSE
)
analysis <- suppressWarnings(calculate_sait(
  analysis,
  method = 'lmm',
  verbose = FALSE
))
plot_file <- plot_expression(analysis, top_n = 2)
# print(plot_file)
```

plot_jis_delta *Plot Multi-Q Delta Influence Heatmaps from TSENATAnalysis Object*

Description

S4 wrapper for . plot_jis_delta() that extracts results directly from a TSENATAnalysis object. Automatically retrieves jackknife switching results from the analysis object slots.

Usage

```
plot_jis_delta(
  analysis,
  n_genes = 4,
  sait_results = NULL,
  verbose = FALSE,
  output_file = NULL,
  ...
)
```

Arguments

analysis	TSENATAnalysis. An S4 object containing completed jackknife isoform switching analysis across multiple q-values.
n_genes	numeric. Number of top genes to display in heatmaps (default: 4). Genes are ranked by LM p-values if available, otherwise by order of appearance in results.
sait_results	data.frame or NULL. Optional SAIT interaction results for ranking genes (default: NULL). If NULL, attempts to extract from analysis@sait_results$sait_interaction.
verbose	logical. If TRUE, print diagnostic messages during plot generation (default: FALSE).
output_file	character or NULL. Optional file path to save the plot. Default: NULL (no file output).
...	Additional arguments passed to the base function.

Details

This function extracts the following from analysis:

Jackknife results From analysis@jackknife_results, which should contain multi-q switching results keyed by q-value (e.g., 'q_1.00')

SAIT results From analysis@sait_results$sait_interaction if not explicitly provided, for ranking genes by significance

The wrapper automatically handles parameter extraction and provides a simplified interface compared to the base function.

Value

A file path (character) to the saved heatmap PNG file, invisibly.

See Also

calculate_jis for computing switching results

Examples

```
# Plot 5: Multi-q delta influence (isoform switching) heatmaps
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# Configure analysis parameters first
config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  subject_col = 'paired_samples',
  paired = TRUE,
  control = 'normal',
  q = seq(0, 2, by = 0.1)
)

# Build analysis with configured parameters
analysis <- build_analysis(
  readcounts = readcounts,
  tx2gene = gff3_dataset,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)

analysis <- filter_analysis(analysis, stringency = 'severe')
analysis <- calculate_diversity(
  analysis,
  q = seq(0.2, 2, by = 0.4),
  verbose = FALSE
)
analysis <- calculate_divergence(
  analysis,
  q = seq(0.2, 2, by = 0.4)
)
analysis <- suppressWarnings(calculate_sait(analysis, method = 'lmm'))
analysis <- calculate_jis(
  analysis,
  q = seq(0.2, 2, by = 0.4),
  n_bootstrap = 20
)
heatmap_file <- plot_jis_delta(analysis, n_genes = 2)
```

plot_sait *Plot GAM q-curves from TSENATAnalysis object*

Description

S4 wrapper that accepts a TSENATAnalysis object and generates GAM q-curve plots

Usage

```
plot_sait(
  analysis,
  n_top = 6,
  genes = NULL,
  condition_col = NULL,
  sig_alpha = 0.05,
  assay_name = "diversity",
  output_file = NULL,
  width = 12,
  height = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

analysis	TSENATAnalysis object with diversity and SAIT interaction results.
n_top	integer. Number of top genes (by adjusted p-value) to plot (default: 6). Only used if genes = NULL.
genes	character vector. Optional specific gene names to plot. If provided, these genes are plotted directly regardless of significance. If NULL (default), top n_top significant genes are selected.
condition_col	character. Column name in colData(se) specifying group assignments for samples. If NULL, attempts to auto-detect from @config (looks for 'condition_col' or 'condition'). If still NULL, defaults to 'sample_type'.
sig_alpha	numeric. Significance threshold for adjusted p-values (default: 0.05). Only used if genes = NULL; filters sait_res to significant genes before selecting top n.
assay_name	character. Name of the assay in se to extract (default: 'diversity').
output_file	character or NULL. Optional file path to save the plot. Default: NULL (no file output).
width	numeric. Width of the plot in inches (default: 12). Only used if output_file is not NULL.
height	numeric or NULL. Height of the plot in inches. Default: NULL (automatically calculated based on width and aspect ratio). Only used if output_file is not NULL.
verbose	logical. If TRUE, print diagnostic messages during processing. (default: FALSE).
...	Additional arguments passed to the base function.

Details

This wrapper automatically: 1. Extracts SummarizedExperiment from `@se` slot 2. Extracts SAIT results from `@sait_results$sait_interaction` slot 3. Detects condition_col from `@config` or uses default 4. Calls `.plot_sait()` with extracted parameters

Parameter Resolution (condition_col):

1. Explicit `condition_col` parameter (highest priority)
2. `@config$condition_col` if available
3. `@config$condition` if available
4. Default: 'sample_type'

Value

A single `ggplot` object with all selected genes arranged in a grid layout. Can be saved with `ggplot2::ggsave()`.

See Also

`calculate_sait` for running LM analysis on TSENATAnalysis.

Examples

```
## Not run:
# Plot 3: GAM q-curves for genes with q-by-condition interactions
data(readcounts)
readcounts <- as.matrix(readcounts)
mode(readcounts) <- 'numeric'
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_dataset <- system.file('extdata', 'annotation.gff3.gz', package =
'TSENAT')

# Configure analysis parameters first
config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  subject_col = 'paired_samples',
  paired = TRUE,
  control = 'normal',
  q = seq(0.2, 2, by = 0.4)  # 5 unique q-values: 0.2, 0.6, 1.0, 1.4, 1.8
)

# Build analysis with configured parameters
analysis <- build_analysis(
  readcounts = readcounts,
  tx2gene = gff3_dataset,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)
```

```
analysis <- filter_analysis(analysis, stringency = 'severe')
analysis <- calculate_diversity(analysis, q = seq(0.2, 2, by = 0.4))
analysis <- suppressWarnings(calculate_sait(analysis, method = 'gam'))

p_gam <- plot_sait(analysis, n_top = 2, sig_alpha = 0.15)
# print(p_gam)

## End(Not run)
```

`readcounts` *Example transcript-level read counts dataset*

Description

A dataset containing transcript-level transcript-level read count abundances from salmon quantification. The dataset includes samples as columns (SRR14800475-SRR14800490) and transcript IDs as row names.

Usage

```
data(readcounts)
```

Format

A data frame with 3514 transcripts (rows) and 16 samples (columns). Row names are transcript IDs (ENST format) and column names are sample IDs. Values represent TPM-normalized read counts.

Value

A matrix (data frame) containing transcript-level TPM-normalized read counts from salmon quantification. Rows represent transcripts (ENST format IDs) and columns represent samples. Aliases include `readcounts` (main read count matrix), `tpm` (TPM values), and `effective_length` (transcript lengths from salmon).

Examples

```
data(readcounts, package = 'TSENAT')
dim(readcounts)
head(readcounts[1:4, 1:3])
```

`results` *Extract analysis results from TSENATAnalysis object*

Description

Provides flexible access to diversity, divergence, and statistical test results with options for ranking, filtering, and format conversion.

Usage

```
results(
  analysis,
  type,
  q = NULL,
  rankBy = "none",
  n = NA,
  filterFDR = NULL,
  format = "text",
  n_genes = 4,
  q_values_table = c(0, 0.5, 1, 1.5, 2),
  top_n = NULL,
  sort_by = "adj_p_interaction",
  sample = NULL,
  plot = FALSE
)
```

Arguments

`analysis`	`TSENATAnalysis` object containing computed results.
`type`	`character`. **Required.** Type of results to extract: 'diversity', 'divergence', 'lm', 'jackknife', 'rank_test', 'effect_sizes_divergence', 'assumptions', 'concordance', or 'switching_tables'.
`q`	`numeric`. For diversity results, optionally return results for a specific q-value only. When specified, returns a single SummarizedExperiment for that q-value instead of the full list. Default: NULL (return all q-values as list). Example: q = 1.0 returns only the q=1.0 results.
`rankBy`	`character`. For LM/Jackknife results, ranking method: 'none' (default), 'pvalue', 'effectSize', or 'padj' (adjusted p-values). Applies to statistical test results. Default: 'none'.
`n`	`integer`. Return top N features/genes ranked by rankBy. Use NA (default) to return all results. Requires rankBy != 'none'.
`filterFDR`	`numeric`. FDR threshold for significance filtering (0.0-1.0). Only results with adjusted p-value <= filterFDR retained. Default: NULL (no filtering).
`format`	`character`. Output format. Behavior depends on result type: - For diversity: 'text' (default, table format), 'se' (SummarizedExperiment), or 'table' (data.frame) - For concordance: 'text' (default, pre-formatted), 'list' (structured for programmatic access) - For switching_tables: 'list' (default, structured list with $gene_headers, $comparison_tables, $q_metadata for vignette rendering)

	or 'raw' (original named list per gene) - For assumptions: 'text' (default, preformatted), 'list' (structured for programmatic access) Default: 'text' for most types, 'list' for switching_tables.
n_genes	integer. Number of genes to display in diversity results. Default: 4.
q_values_table	numeric. Vector of q-values to include in diversity results. Default: c(0, 0.5, 1.0, 1.5, 2.0).
top_n	integer. For effect_sizes_divergence, return top N genes ranked by sort_by. When specified, results are sorted by sort_by column and limited to top N rows. Default: NULL (return all results). Use NA to return all.
sort_by	character. For effect_sizes_divergence, column name to sort by. Common choices: 'adj_p_interaction' (p-value, ascending), 'Mean_Divergence' (descending). Default: 'adj_p_interaction' (most significant first).
sample	character or NULL. For diversity results, optionally filter to specific sample(s). If NULL (default), returns results for all samples. If character, filters to matching sample identifiers from colData. Default: NULL (all samples).
plot	logical. Extract cached plot for the specified analysis type. - If FALSE (default): Return results as usual - If TRUE: Return the plot object for the given type Example: `results(analysis, type = 'diversity', plot = TRUE)` returns the diversity plot. Default: FALSE (no plot extraction).

Details

This function provides flexible access to all computed results with ranking, filtering, and format conversion. Compatible with DESeq2/edgeR design patterns for familiar result extraction workflows.

Lazy Computation for switching_tables: When requesting `type = 'switching_tables'`, the function automatically computes and caches the tables if they don't exist yet but the prerequisites do (LM and jackknife results). This eliminates the need for a separate `prepare_gene_switching_tables_s4()` call - simply request the results and they will be computed on-demand.

Value

- For diversity with q=NULL: A named list of SummarizedExperiment objects, one per q-value - For diversity with q specified and format='text' or 'table': A data.frame table for display - For diversity with q specified and format='se': The SummarizedExperiment object directly (useful for SplicingFactory) - For divergence: A SummarizedExperiment (rows=genes, columns=q-values), data.frame, or other format depending on divergence computation method - For lm/jackknife: A data.frame or list based on type and format - For effect_sizes_divergence: A list containing effect size divergence results with components like interaction_results - For assumptions: A list containing rank-based assumption checks (exchangeability, monotonicity, consistency) and optional method-specific diagnostics (gam_metrics, gee_metrics, lmm_metrics, fpca_metrics) - For switching_tables with format='text' (default): A list with components:

- `$gene_headers` Character vector of gene headers ('GeneName (ENSG00...)')
- `$comparison_tables` List of data frames, one per gene, with columns: Transcript, q=0.00, q=0.50, ..., Direction Consistency
- `$q_metadata` List with per-gene metadata: q_values_available and q_key_to_value mapping

- For switching_tables with format='raw': Original named list where names are gene headers and values are data frames with same column structure as format='text' Returns NULL if requested result type not computed or no results pass filtering.

Examples

```
# Load example data
data(readcounts, package = 'TSENAT')

# Create TSENATAnalysis from count matrix
config <- TSENAT_config(
  q = 1.0,
  condition_col = 'group'
)
se <- SummarizedExperiment::SummarizedExperiment(
  assays = list(counts = readcounts),
  colData = data.frame(
    group = rep(c('A', 'B'), length.out = ncol(readcounts))
  )
)
analysis <- TSENATAnalysis(se = se, config = config)
analysis <- calculate_diversity(analysis)

# Get all diversity results (list of SummarizedExperiment objects, one per q)
div_all <- results(analysis, type = 'diversity')

# Get diversity for specific q-value (single SummarizedExperiment table for display)
div_q1 <- results(analysis, type = 'diversity', q = 1.0)

# Get diversity as SummarizedExperiment for downstream processing
div_q1_se <- results(analysis, type = 'diversity', q = 1.0, format = 'se')

# Get results ranked by p-value, top 20 genes
# Using accessor function instead of @ slot access
top_sait <- results(analysis, type = "sait", rankBy = 'pvalue', n = 20)

# Get effect size results, top 6 genes by p-value (most significant first)
top_effect_sizes <- results(analysis, type = 'effect_sizes_divergence',
                            top_n = 6, sort_by = 'adj_p_interaction')

# Get effect sizes sorted by mean divergence (largest effect sizes first)
large_effects <- results(analysis, type = 'effect_sizes_divergence',
                         top_n = 10, sort_by = 'Mean_Divergence')

# Get switching tables - automatically computed if prerequisites exist
# (no need to call prepare_gene_switching_tables_s4 separately)
# Default format='text' returns structured list for vignette rendering
switching <- results(analysis, type = 'switching_tables')

# Get switching tables in raw format (named list of data frames) for direct manipulation
switching_raw <- results(analysis, type = 'switching_tables', format = 'raw')

# ========================================================================
# RETRIEVE CACHED PLOTS using the plot parameter
# ========================================================================

# Get the diversity spectrum plot
diversity_plot <- results(analysis, type = 'diversity', plot = TRUE)

# Get the LM/regularized regression (GAM, LMM, GEE, FPCA) interaction plot
sait_plot <- results(analysis, type = "sait", plot = TRUE)
```

```
# Get the divergence distribution plot
div_dist_plot <- results(analysis, type = 'divergence', plot = TRUE)

# Get the influence/m-estimator plot
influence_plot <- results(analysis, type = 'influence', plot = TRUE)

# Available plot types correspond to analysis types:
# - type = 'diversity': Returns Tsallis entropy q-spectrum visualization
# - type = "sait": Returns regularized/penalized regression (GAM, LMM, GEE, FPCA) interaction plot
# - type = 'divergence': Returns distribution of divergence metrics across genes
# - type = 'influence': Returns m-estimator sample influence analysis
# - type = 'rank_test': Returns Conover-Iman Rank Transform interaction results
# - type = 'concordance': Returns method concordance comparison (LM vs rank test)
```

---

se *Extract SummarizedExperiment from TSENATAnalysis*

Description

Extract SummarizedExperiment from TSENATAnalysis

Usage

```
se(object)

## S4 method for signature 'TSENATAnalysis'
se(object)
```

Arguments

`object`	`TSENATAnalysis` object.

Details

Provides read-only access to the underlying SummarizedExperiment. To modify the SummarizedExperiment, use the configuration methods or directly access `object@se`.

Value

SummarizedExperiment object

The `SummarizedExperiment` object stored in `@se` slot, containing raw count data and sample metadata.

Examples

```
# Create a simple TSENATAnalysis object
library(SummarizedExperiment)
counts <- matrix(rpois(100, lambda = 10), nrow = 10, ncol = 10)
colnames(counts) <- paste0('sample_', 1:10)
rownames(counts) <- paste0('gene_', 1:10)
```

```
se <- SummarizedExperiment(assays = list(counts = counts))
analysis <- new('TSENATAnalysis', se = se, config = list())

# Extract the SummarizedExperiment
extracted_se <- se(analysis)
dim(extracted_se)
```

TSENAT *Run complete TSENAT analysis pipeline*

Description

Coordinates the full TSENAT workflow: diversity -> jackknife -> LM interactions -> divergence -> gene interactions -> rank-based tests -> concordance -> visualizations.

Usage

```
TSENAT(
  analysis,
  output_dir = "tsenat_outputs",
  save_output = TRUE,
  output_format = c("tsv", "csv", "txt", "rds"),
  verbose = TRUE,
  tips = FALSE
)
```

Arguments

analysis	TSENATAnalysis object created by build_analysis.
output_dir	character. Directory to save results and plots. Default: 'tsenat_outputs'. Set to NULL to disable automatic output saving.
save_output	logical. Whether to save output files (results tables). Default: TRUE. If FALSE, no TSV/CSV output files are written to disk.
output_format	character. Format for output files: 'tsv' (tab-separated), 'csv' (comma-separated), 'txt' (text), or 'rds' (R serialized). Default: 'tsv'.
verbose	logical. Print progress messages. Default: TRUE.
tips	logical. Show tips and usage examples at the end of the pipeline run. Default: FALSE. Set to TRUE to see the '[TIPS]' section displaying common result-extraction commands.

Details

Pipeline execution order (enforced, follows TSENAT.Rmd vignette):

1. filter_analysis() - Filter low-abundance transcripts
2. calculate_diversity() - Tsallis entropy per q-value
3. plot_diversity_spectrum() - Visualize q-spectrum
4. calculate_m_estimator() - Sample influence QC analysis
5. calculate_sait() - LM/SAIT interaction testing

6. plot_sait() - SAIT results visualization
7. calculate_jis() - Transcript switching detection
8. plot_jis_delta() - Multi-q influence heatmap (gene switching tables computed lazily via results())
9. plot_expression() - Top transcript visualization
10. calculate_divergence() - Pairwise divergence metrics
11. calculate_effect_sizes() - Effect size computation
12. plot_divergence_distribution() - Divergence distribution plot
13. plot_divergence_spectrum() - Divergence spectrum plot
14. calculate_assumptions() - Validate rank-based test assumptions
15. calculate_rank_transform() - ART (Aligned Rank Transform) interaction test
16. calculate_concordance() - Compare LM and rank test results

Value

TSENATAnalysis object containing complete analysis results, plots, and metadata.

Examples

```
data(readcounts, package = 'TSENAT')
metadata_df <- read.table(
  system.file('extdata', 'metadata.tsv', package = 'TSENAT'),
  header = TRUE, sep = '\t'
)
gff3_file <- system.file('extdata', 'annotation.gff3.gz', package = 'TSENAT')

config <- TSENAT_config(
  sample_col = 'sample',
  condition_col = 'condition',
  q = seq(0, 2, length.out = 10),
  generate_plots = FALSE
)
analysis <- build_analysis(
  readcounts = as.matrix(readcounts),
  tx2gene = gff3_file,
  metadata = metadata_df,
  config = config,
  tpm = tpm,
  effective_length = effective_length
)

result <- TSENAT(analysis)
```

`TSENATAnalysis` *Constructor for TSENATAnalysis objects*

Description

Creates a new TSENATAnalysis object with a SummarizedExperiment base and optional initial configuration.

Usage

```
TSENATAnalysis(se, config = list())
```

Arguments

`se`	`SummarizedExperiment`. The base expression data object.
`config`	`list`. Optional initial configuration (usually set via `TSENAT_config()` instead).

Details

The constructor initializes all slots with empty lists except @se, which must be provided. The @metadata slot automatically records: - creation timestamp - TSENAT package version - initial function call

Value

A new `TSENATAnalysis` object.

Examples

```
# Load real TSENAT data
data(readcounts)
metadata_df <- read.table(system.file('extdata', 'metadata.tsv', package
= 'TSENAT'),
  header = TRUE, sep = '\t')
gff3_file <- system.file('extdata', 'annotation.gff3.gz', package = 'TSENAT')
config <- TSENAT_config(sample_col = 'sample', condition_col = 'condition')
analysis <- build_analysis(readcounts = readcounts, tx2gene =
gff3_file, metadata = metadata_df, config = config,
  tpm = tpm, effective_length = effective_length)
analysis <- filter_analysis(analysis, min_samples = 1, subset_n_genes
= 200)
```

TSENATAnalysis-class *TSENATAnalysis S4 Class Definition*

Description

Central container for unified analysis workflows in TSENAT.

Usage

```
## S4 method for signature 'TSENATAnalysis,ANY,ANY,ANY'
x[i, j, drop = TRUE]
```

Arguments

x	A TSENATAnalysis object to subset.
i	Gene indices (numeric, logical, or character vector). Defaults to all genes.
j	Sample indices (numeric, logical, or character vector). Defaults to all samples.
drop	Ignored for TSENATAnalysis objects; included for S4 method signature compatibility.

Details

The TSENATAnalysis class encapsulates all components of a complete TSENAT analysis: raw data, configuration metadata, and results from each analytical step. This unified object ensures metadata is never lost through the analysis pipeline and provides consistent accessor methods for result retrieval.

Access results via the unified results(obj, type = ...) accessor method: - type='diversity' for Tsallis entropy across q-values - type = "sait" for regularized/penalized regression (GAM, LMM, GEE, FPCA) interaction results - type='rank_test' for Conover-Iman Rank Transform results - type='divergence' for divergence metrics - type='jackknife' for jackknife resampling results Use metadata(obj) to access reproducibility metadata.

Value

A new TSENATAnalysis object containing only the specified genes and samples, with all associated results and metadata preserved.

Slots

se SummarizedExperiment. The base expression data object (genes x samples) with assays and colData.

config list. Configuration metadata specifying analysis parameters that persist through the workflow (q-values, sample grouping columns, etc.). Set once via TSENAT_config() and used by all downstream wrapper functions.

diversity_results list. Named list of diversity calculation results. Each name corresponds to a q-value (e.g., 'q_0.5', 'q_1.0'). Values are SummarizedExperiment objects or data.frames containing entropy values for each gene at that q-value.

sait_results list. Complex results from scale-adaptive interaction models (GAM with smoothing, LMM with random effects, GEE with working correlations, FPCA for functional data) (GAM, LMM, GEE, FPCA) and statistical testing. Top-level names identify analysis type:

`sait_interaction` Regularized regression (GAM/LMM/GEE/FPCA) model results (list with `$results` data.frame, `$models` list, etc.)

`rank_test` ART (Aligned Rank Transform) or Conover-Iman Rank Transform results

`divergence_difference` Differential divergence comparison

`rank_test_results` list. ART (Aligned Rank Transform) or Conover-Iman Rank Transform test results. Names correspond to q-values (e.g., 'q_0.5', 'q_1.0'). Computed by `calculate_rank_transform()` as a non-parametric alternative to linear mixed model testing.

`jackknife_results` list. Resampling-based confidence intervals. Names correspond to q-values (e.g., 'q_0.5', 'q_1.0'). Values are jackknife result objects containing resamples, CI bounds, and diagnostics.

`divergence_results` list. Divergence metric calculations. Typically contains:

`tsallis_divergence` SummarizedExperiment with divergence values

`effect_sizes` data.frame with Cohen's d, etc.

`plots` list. Cached visualization objects (ggplot). Names identify plot type (e.g., 'q_curve', 'sait_interaction', 'influence'). Populated by `TSENAT()` if `generate_plots=TRUE`.

`metadata` list. Reproducibility and tracking metadata. Automatically maintained by wrapper functions. Includes:

`created_at` Timestamp of object creation

`function_calls` Vector of wrapper functions called

`function_timestamps` Timestamps for each function call

`package_version` TSENAT version at creation

`TSENAT_config` *Create and return TSENAT configuration*

Description

Builds a configuration list for use with `TSENAT()`. Allows specifying analysis parameters once and reusing across multiple analyses.

Usage

```
TSENAT_config(
  q = 1,
  condition_col = "condition",
  subject_col = NULL,
  sample_col = "sample",
  paired = FALSE,
  control = NULL,
  p_threshold = 0.05,
  fdr_threshold = 0.05,
  significance_threshold = 0.05,
  bootstrap = FALSE,
  nboot = 1000,
  bootstrap_method = c("percentile", "bca"),
  stringency = "medium",
  nthreads = 1,
  norm = TRUE,
```

```
  bootstrap_ci = 0.95,
  bootstrap_include_diagnostics = TRUE,
  min_valid_frac = 0.75,
  norm_method = NULL,
  pseudocount = 0,
  shrinkage = "none",
  sait_method = c("gam", "lmm", "fpca", "gee"),
  sait_pcorr = c("BH", "bonferroni", "hochberg", "holm"),
  jis_use_sait_fdr = TRUE,
  divergence_ci = 0.95,
  assumptions_checks = c("all", "rank", "gam"),
  ...
)
```

Arguments

q	numeric. Q-value(s) for Tsallis entropy (single value or vector). Default: 1.0 (Shannon entropy). Usage: calculate_diversity/divergence use this for spectrum computation (if vector) or as default fallback (if single).
condition_col	character. Column name in colData containing conditions. Default: 'condition'.
subject_col	character. Column name in colData containing subject IDs (for paired designs). Default: NULL.
sample_col	character. Column name in colData containing sample IDs. Default: 'sample'.
paired	logical. Whether samples are paired/repeated measures. Default: FALSE.
control	character. Reference/control group label. Default: NULL.
p_threshold	numeric. Raw p-value threshold. Default: 0.05.
fdr_threshold	numeric. FDR-adjusted p-value threshold. Default: 0.05.
significance_threshold	numeric. Significance cutoff. Default: 0.05.
bootstrap	logical. Enable bootstrap CIs. Default: FALSE.
nboot	integer. Bootstrap resamples for CIs. Default: 1000.
bootstrap_method	character. Bootstrap method: 'percentile' or 'bca'. Default: 'percentile'.
stringency	character. Filtering stringency: 'lenient', 'medium', 'severe'. Default: 'medium'.
nthreads	integer. Parallel threads. Default: 1.
norm	logical. Enable normalization. Default: TRUE.
bootstrap_ci	numeric. Confidence level (0-1). Default: 0.95.
bootstrap_include_diagnostics	logical. Include diagnostics. Default: TRUE.
min_valid_frac	numeric. Min valid replicate fraction. Default: 0.75.
norm_method	character. Normalization: NULL, 'zscore', 'log_odds_ratio', 'relative_reference'. Default: NULL.
pseudocount	numeric. Pseudocount for sparse data. Default: 0.
shrinkage	character. Variance reduction: 'none' or 'empirical_bayes'. Default: 'none'.
sait_method	character. SAIT method: 'gam', 'lmm', 'fpca', 'gee'. Default: 'gam'.

`sait_pcorr` `character`. P-value correction: 'BH', 'bonferroni', 'hochberg', 'holm'. Default: 'BH'.

`jis_use_sait_fdr` `logical`. Filter jackknife genes using SAIT p-values. Default: TRUE.

`divergence_ci` `numeric`. Confidence level for divergence CIs. Default: 0.95.

`assumptions_checks` `character`. Which assumptions to test (default: 'all'). Presets: - 'rank': core assumption checks (exchangeability, monotonicity, consistency) - 'all': all checks including method-specific diagnostics (GAM, GEE, LMM, FPCA) Explicit: character vector like `c('exchangeability', 'monotonicity')`.

`...` Additional configuration parameters (stored as-is).

Details

Configuration is stored in the TSENATAnalysis@config slot and used by wrapper functions to configure analysis behavior. Note: Statistical tests (Wilcoxon, shuffle) work on a single q-value, so only one q-value is specified in config.

Value

`list` with class `TSENATConfig` containing all specified parameters.

Examples

```
# Default config with standard parameters (point estimates only)
cfg <- TSENAT_config()

# For Wilcoxon/shuffle tests (single q-value required in config)
cfg <- TSENAT_config(
  q = 1.0,                          # Shannon entropy - for rank tests
  condition_col = 'treatment',
  control = 'untreated'
)

# For Conover-Iman Rank Transform tests (multiple q-values)
cfg <- TSENAT_config(
  q = seq(0, 2, by = 0.5),          # Multiple q-values for spectrum or advanced testing
  condition_col = 'treatment',
  control = 'untreated'
)

# With bootstrap CIs for uncertainty quantification (recommended)
cfg <- TSENAT_config(
  bootstrap = TRUE,                # Enable bootstrap confidence intervals
  bootstrap_method = 'bca',         # Bias-corrected (better for skewed entropy)
  nboot = 1000,                     # 1000 resamples
  bootstrap_ci = 0.95               # 95% CI
)

# Custom with paired analysis, strict filtering, and normalization
cfg <- TSENAT_config(
  q = 1.0,                          # Shannon entropy
  condition_col = 'treatment',
  subject_col = 'subject_id',
  paired = TRUE,
```

```
  control = 'untreated',
  stringency = 'severe',           # High-confidence transcripts only
  norm_method = 'zscore',          # Cross-study standardization
  shrinkage = 'none',              # Empirical estimates
  bootstrap = TRUE,
  bootstrap_method = 'bca',
  nboot = 5000,                    # Higher precision
  pseudocount = 0,                 # Disabled by default; set > 0 to add pseudocount
  significance_threshold = 0.01    # Stricter significance level
)
```

Index