Package 'TaxSEA'

November 12, 2026

Type Package

Title Taxon Set Enrichment Analysis

Version 1.4.0

Description TaxSEA is an R package for Taxon Set Enrichment Analysis, which utilises a Kolmogorov-Smirnov test analyses to investigate differential abundance analysis output for whether there are alternations in a-priori defined sets of taxa from public databases (BugSigDB, MiMeDB, GutMGene, mBodyMap, BacDive and GMRepoV2) and collated from the literature. TaxSEA takes as input a list of taxonomic identifiers (e.g. species names, NCBI IDs etc.) and a rank (E.g. fold change, correlation coefficient). TaxSEA be applied to any microbiota taxonomic profiling technology (array-based, 16S rRNA gene sequencing, shotgun metagenomics & metatranscriptomics etc.) and enables researchers to rapidly contextualize their findings within the broader literature to accelerate interpretation of results.

License GPL-3

Encoding UTF-8

LazyData false

VignetteBuilder knitr

RoxygenNote 7.3.3

biocViews Microbiome, Metagenomics, Sequencing, GeneSetEnrichment, RNASeq

URL https://github.com/feargalr/taxsea, https://feargalr.github.io/TaxSEA/

BugReports https://github.com/feargalr/taxsea/issues

Depends R (>= 4.5.0)

Suggests BiocStyle, bugsigdbr, fgsea, knitr, mia, rmarkdown, SummarizedExperiment, testthat

Imports stats, utils

Config/testthat/edition 3

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

git_branch RELEASE_3_23

git_last_commit 76ffa27

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Author Feargal Ryan [aut, cre, fnd] (ORCID:
<https://orcid.org/0000-0002-1565-4598>, funding: Supported by
NHMRC Investigator Grant)

Maintainer Feargal Ryan <feargalr@gmail.com>

Contents

get_ncbi_taxon_ids 2
get_taxon_sets 3
NCBI_ids 3
ssTaxSEA 4
taxon_rank_sets 6
TaxSEA 8
TaxSEA_db 9
TaxSEA_test_data 10

Index **11**

`get_ncbi_taxon_ids` *Retrieve NCBI Taxonomy IDs for a list of taxon names*

Description

This function takes a vector of taxon names and returns a vector of NCBI taxonomy IDs by querying the NCBI Entrez API.

Usage

```
get_ncbi_taxon_ids(taxon_names)
```

Arguments

`taxon_names`	A character vector of taxon names

Value

A character vector of NCBI taxonomy IDs corresponding to the input taxon names

Examples

```
taxon_names <- c("Escherichia coli", "Staphylococcus aureus")
taxon_ids <- get_ncbi_taxon_ids(taxon_names)
```

---

get_taxon_sets *Retrieve Taxon Sets from TaxSEA Library*

Description

Retrieve from the TaxSEA database which taxon sets (metabolite producers and disease signatures) contain a taxon of interest.

Usage

```
get_taxon_sets(taxon_to_fetch = taxon)
```

Arguments

taxon_to_fetch	The taxon to search for in the TaxSEA database.

Value

A character vector containing the names of taxonomic sets where the specified taxon is present.

Examples

```
# Retrieve sets for Bifidobacterium longum
get_taxon_sets(taxon="Bifidobacterium_longum")
```

NCBI_ids *NCBI IDs Dataset*

Description

A dataset for mapping NCBI IDs to species/genus names. This named vector allows for lookup of NCBI IDs associated with species or genus names.

Usage

```
NCBI_ids
```

Format

A named vector where:

names NCBI IDs

values Species or genus names

A named vector mapping NCBI IDs to species or genus names.

Source

NCBI

Examples

```
data(NCBI_ids)
# Can look up either with or without spaces
NCBI_ids["Bifidobacterium_breve"]
NCBI_ids["Bifidobacterium breve"]
```

`ssTaxSEA` *Single-Sample Taxon Set Enrichment Analysis*

Description

Computes per-sample enrichment scores for taxon sets using an ssGSEA-style approach. Counts are CLR-transformed, then each taxon is z-scored across samples to capture between-sample variation. Per-sample enrichment is then computed by ranking each sample's z-scores and applying a weighted running-sum statistic against taxon sets from the TaxSEA database.

Usage

```
ssTaxSEA(
  counts,
  lookup_missing = FALSE,
  min_set_size = 5,
  max_set_size = 300,
  custom_db = NULL
)
```

Arguments

`counts`	A numeric matrix, data.frame, or `SummarizedExperiment`/`TreeSummarizedExperiment` object. For matrix/data.frame input: rows are taxa, columns are samples, and row names must be taxon names (e.g. species names or NCBI IDs). For SummarizedExperiment input: the first assay is used and `rownames()` provide taxon identifiers.
`lookup_missing`	Logical indicating whether to fetch missing NCBI IDs via the NCBI API. Default is FALSE.

`min_set_size`	Minimum size of taxon sets to include. Default is 5.
`max_set_size`	Maximum size of taxon sets to include. Default is 300.
`custom_db`	A user-provided list of taxon sets. If NULL (default), the built-in TaxSEA database is used (excluding BugSigDB).

Details

The approach works as follows:

1. Raw counts are CLR-transformed (centered log-ratio with pseudocount of 0.5 for zeros).
2. Each taxon is then z-scored across all samples, so that values represent how much higher or lower a taxon is in a given sample relative to the cohort mean.
3. For each sample, the z-scores are ranked and an ssGSEA-style weighted running-sum enrichment score is computed for each taxon set.
4. A KS test p-value is also computed per sample per set.

This cohort-relative approach ensures that taxon sets which are consistently elevated in a subset of samples (e.g. disease samples) will produce high enrichment scores in those samples, even if the taxa are not the most abundant within any single sample.

Value

A list with two elements:

scores A matrix (samples x taxon sets) of enrichment scores. Positive scores indicate the set taxa tend to have higher abundance in that sample relative to the cohort.

pvalues A matrix (samples x taxon sets) of KS test p-values for each sample-set combination.

Examples

```
## Not run:
# From a count matrix (taxa x samples)
counts <- matrix(rpois(500, lambda = 10), nrow = 50, ncol = 10)
rownames(counts) <- paste0("Taxon_", seq_len(50))
colnames(counts) <- paste0("Sample_", seq_len(10))
res <- ssTaxSEA(counts, custom_db = list(
  set1 = paste0("Taxon_", 1:10),
  set2 = paste0("Taxon_", 20:30)
), min_set_size = 2)
head(res$scores)
head(res$pvalues)

## End(Not run)
```

taxon_rank_sets *Taxon Rank Set Enrichment Analysis*

Description

Groups species by taxonomic ranks and performs TaxSEA enrichment analysis at each rank level. Returns a named list of data frames, one per taxonomic rank (excluding species).

Usage

```
taxon_rank_sets(taxon_ranks, lineage_df, min_set_size = 5, max_set_size = 100)
```

Arguments

taxon_ranks	A named numeric vector of log2 fold changes. Names should be feature identifiers matching the species column in lineage_df, or matching rownames() of a SummarizedExperiment/TreeSummarizedExperiment.
lineage_df	Either a data frame or a SummarizedExperiment/TreeSummarizedExperiment object. **Data frame input:** Must include a species column and one or more taxonomic rank columns (e.g., kingdom, phylum, class, order, family, genus). The species column is used for matching and is excluded from the enrichment analysis. **SummarizedExperiment input:** Taxonomy is extracted from rowData() and feature identifiers from rownames(). If the mia package is installed, taxonomyRanks() is used to identify taxonomic rank columns; otherwise all rowData() columns are used. The Species rank column is automatically excluded from the analysis. Requires the SummarizedExperiment package.
min_set_size	Minimum number of species in a set to include in the analysis. Default is 5.
max_set_size	Maximum number of species in a set to include in the analysis. Default is 100.

Value

A named list of data frames, one per taxonomic rank. Each data frame contains columns: taxonSetName, median_rank_of_set_members, PValue, Test_statistic, and FDR. Ranks that produce no valid sets (e.g., due to size filtering) are included as empty data frames with a message.

Examples

```
# --- Example 1: Data frame input ---
# Create a lineage data frame (e.g., parsed from curatedMetagenomicData)
# The 'species' column must match the names in taxon_ranks.

lineage_df <- data.frame(
  species = c("Cutibacterium_acnes", "Klebsiella_pneumoniae",
              "Propionibacterium_humerusii", "Moraxella_osloensis",
              "Enhydrobacter_aerosaccus", "Staphylococcus_capitis",
              "Staphylococcus_epidermidis", "Staphylococcus_aureus",
```

```
              "Escherichia_coli", "Enterobacter_cloacae",
              "Pseudomonas_aeruginosa", "Acinetobacter_baumannii",
              "Lactobacillus_rhamnosus", "Lactobacillus_acidophilus",
              "Bifidobacterium_longum", "Bifidobacterium_breve"),
  kingdom = rep("Bacteria", 16),
  phylum = c("Actinobacteria", "Proteobacteria",
             "Actinobacteria", "Proteobacteria",
             "Proteobacteria", "Firmicutes",
             "Firmicutes", "Firmicutes",
             "Proteobacteria", "Proteobacteria",
             "Proteobacteria", "Proteobacteria",
             "Firmicutes", "Firmicutes",
             "Actinobacteria", "Actinobacteria"),
  class = c("Actinobacteria", "Gammaproteobacteria",
            "Actinobacteria", "Gammaproteobacteria",
            "Alphaproteobacteria", "Bacilli",
            "Bacilli", "Bacilli",
            "Gammaproteobacteria", "Gammaproteobacteria",
            "Gammaproteobacteria", "Gammaproteobacteria",
            "Bacilli", "Bacilli",
            "Actinobacteria", "Actinobacteria"),
  order = c("Propionibacteriales", "Enterobacterales",
            "Propionibacteriales", "Pseudomonadales",
            "Rhodospirillales", "Bacillales",
            "Bacillales", "Bacillales",
            "Enterobacterales", "Enterobacterales",
            "Pseudomonadales", "Pseudomonadales",
            "Lactobacillales", "Lactobacillales",
            "Bifidobacteriales", "Bifidobacteriales"),
  family = c("Propionibacteriaceae", "Enterobacteriaceae",
             "Propionibacteriaceae", "Moraxellaceae",
             "Rhodospirillaceae", "Staphylococcaceae",
             "Staphylococcaceae", "Staphylococcaceae",
             "Enterobacteriaceae", "Enterobacteriaceae",
             "Pseudomonadaceae", "Moraxellaceae",
             "Lactobacillaceae", "Lactobacillaceae",
             "Bifidobacteriaceae", "Bifidobacteriaceae"),
  genus = c("Cutibacterium", "Klebsiella",
            "Cutibacterium", "Moraxella",
            "Enhydrobacter", "Staphylococcus",
            "Staphylococcus", "Staphylococcus",
            "Escherichia", "Enterobacter",
            "Pseudomonas", "Acinetobacter",
            "Lactobacillus", "Lactobacillus",
            "Bifidobacterium", "Bifidobacterium"),
  stringsAsFactors = FALSE
)

set.seed(42)
fc <- setNames(rnorm(16), lineage_df$species)
results <- taxon_rank_sets(fc, lineage_df, min_set_size = 2)
names(results)
results$family
```

```
# --- Example 2: SummarizedExperiment / TreeSummarizedExperiment input ---
## Not run:
library(mia)
data(GlobalPatterns, package = "mia")
tse <- GlobalPatterns

# Run differential abundance (e.g., ALDEx2) to get fold changes
# aldex_out <- ... (your DA analysis)
# fc <- aldex_out$effect
# names(fc) <- rownames(aldex_out)

# Run taxon rank set enrichment directly from the TSE
results <- taxon_rank_sets(fc, tse, min_set_size = 5)
names(results)  # Kingdom, Phylum, Class, Order, Family, Genus
results$Family

## End(Not run)
```

`TaxSEA` *TaxSEA: Taxon Set Enrichment Analysis*

Description

Modular TaxSEA implementation supporting enrichment (KS) and ORA (Fisher). Provide either `taxon_ranks` for enrichment or `input_taxa` for ORA.

Usage

```
TaxSEA(
  taxon_ranks = NULL,
  input_taxa = NULL,
  mode = NULL,
  lookup_missing = FALSE,
  min_set_size = 5,
  max_set_size = 300,
  custom_db = NULL
)
```

Arguments

`taxon_ranks`	Named numeric vector of statistics (e.g., log2 fold changes). Required for enrichment.
`input_taxa`	Character vector of taxa to treat as "hits"/selected taxa. Required for ORA.
`mode`	Character. One of `"enrichment"` or `"ora"`. If NULL, inferred from which input is provided.
`lookup_missing`	Logical indicating whether to fetch missing NCBI IDs. Default is FALSE.

min_set_size	Minimum size of taxon sets to include in the analysis. Default is 5.
max_set_size	Maximum size of taxon sets to include in the analysis. Default is 100.
custom_db	A user-provided list of taxon sets. If NULL (default), the built-in database is used.

Value

A list of data frames with taxon set results.

Examples

```
data("TaxSEA_test_data")
res <- TaxSEA(taxon_ranks = TaxSEA_test_data)
head(res$All_databases)

# ORA example (toy): treat taxa with positive values as "hits"
hits <- names(TaxSEA_test_data)
res_ora <- TaxSEA(input_taxa = hits, mode = "ora")
head(res_ora$All_databases)
```

TaxSEA_db *TaxSEA Database A dataset containing taxon sets. Each item in the list is a taxon set, and each member within a taxon set is a taxon.*

Description

TaxSEA Database A dataset containing taxon sets. Each item in the list is a taxon set, and each member within a taxon set is a taxon.

Usage

```
TaxSEA_db
```

Format

A list of vectors. Each vector contains character strings representing taxa.

Source

See READ ME.

Examples

```
data(TaxSEA_db)
all_sets <- names(TaxSEA_db)
GABA_producers<-TaxSEA_db[["MiMeDB_producers_of_GABA"]]
```

TaxSEA_test_data *TaxSEA Test Data*

Description

A dataset containing taxon ranks and taxon IDs.

Usage

```
TaxSEA_test_data
```

Format

A data frame with two columns:

rank Character vector representing taxon ranks

id Character vector representing taxon IDs

A data frame with columns 'rank' and 'id' representing taxon ranks and taxon IDs, respectively.

Source

See READ ME.

Examples

```
data(TaxSEA_test_data)
test_results <- TaxSEA(TaxSEA_test_data)
```

Index

∗ **datasets**
NCBI_ids, 3
TaxSEA_db, 9
TaxSEA_test_data, 10

get_ncbi_taxon_ids, 2
get_taxon_sets, 3

NCBI_ids, 3

ssTaxSEA, 4

taxon_rank_sets, 6
TaxSEA, 8
TaxSEA_db, 9
TaxSEA_test_data, 10