Package 'karioCaS'

October 30, 2026

Title Kraken Confidence Scores for Reliable Domain-Specific Microbiota Inference and Discovery

Version 0.99.16

Description Provides a comprehensive framework for analyzing Kraken2 metagenomic classification using multiple confidence scores. karioCaS implements a robust comparative approach to evaluate taxonomic stability across different stringency levels, facilitating the identification of reliable microbiota components and the discovery of potentially novel or unrepresented species. A key innovation is its domain-specific analysis, separately processing Bacteria, Archaea, Eukaryota, and Viruses to ensure balanced visibility of all biological domains. Includes high-quality visualization tools for retention rates, shared taxa (Upset plots), and stability heatmaps.

License GPL (>= 3)

Depends R (>= 4.5.0)

Encoding UTF-8

RoxygenNote 8.1.0

Imports ggplot2, patchwork, UpSetR, dplyr, readr, stringr, tidyr, scales, ggtext, SummarizedExperiment, TreeSummarizedExperiment, S4Vectors, forcats, tibble, methods, rlang, grid, grDevices, stats, utils

Suggests BiocStyle, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

biocViews Metagenomics, Microbiome, DataImport, Software

URL https://github.com/thiagoparentefiocruz/karioCaS

BugReports https://github.com/thiagoparentefiocruz/karioCaS/issues

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

git_branch devel

git_last_commit f2bbabd

git_last_commit_date 2026-09-28

Repository Bioconductor 3.24

Date/Publication 2026-10-29

Author Thiago Parente [aut, cre] (ORCID:
<https://orcid.org/0000-0001-7501-726X>),
Fiocruz [fnd],
IOC-Fiocruz [fnd],
CAPES [fnd]

Maintainer Thiago Parente <thiago.parente@fiocruz.br>

Contents

karioCaS-package 2
group_upset 3
heatmaps_karioCaS 4
import_karioCaS 6
kariocas_result 7
reads_per_taxa 8
retrieve_selected_taxa 9
taxa_resolution 11
taxa_retention 13
upset_kariocas 15

Index **16**

`karioCaS-package` *karioCaS: Kraken Confidence Scores for Reliable Domain-Specific Microbiota Inference and Discovery*

Description

karioCaS provides a framework to explore the effect of the Kraken2 confidence score (CS) on metagenomic classification. Reports produced at several CS values are imported into a single `TreeSummarizedExperiment`, and taxonomic stability is then evaluated across stringency levels, separately for Bacteria, Archaea, Eukaryota and Viruses, to identify an objective CS threshold per domain.

Typical workflow

1. `import_karioCaS()`: import the per-CS reports into a `TreeSummarizedExperiment`.
2. `taxa_retention()`: taxa retention curves across CS and the Stability Index used to select the optimal CS per domain.
3. `reads_per_taxa()`: saturation analysis and optimal minimum reads per domain.
4. `retrieve_selected_taxa()`: build the final, domain-specific taxonomic mosaic.
5. `upset_kariocas()`, `group_upset()`, `heatmaps_karioCaS()`, `taxa_resolution()`: exploratory and comparative visualisations.

Acknowledgement of AI assistance

The scientific concept, the method design and its validation are the author's. The package code was written with the assistance of large language models, which also contributed to the formulation of some of the calculations. All code was reviewed and tested by the author.

Author(s)

Maintainer: Thiago Parente <thiago.parente@fiocruz.br> (ORCID)

Other contributors:

- Fiocruz [funder]
- IOC-Fiocruz [funder]
- CAPES [funder]

See Also

Useful links:

- https://github.com/thiagoparentefiocruz/karioCaS
- Report bugs at https://github.com/thiagoparentefiocruz/karioCaS/issues

---

`group_upset` *Cross-Sample UpSet: Core vs Unique Taxa per Biological Group (Step 008)*

Description

For each biological group (inferred from sample names by stripping trailing digits, e.g. `SAMPLE33`, `SAMPLE34` -> `SAMPLE`), draws an UpSet plot comparing which taxa (at `tax_level`) are present across the samples of the group, per Domain. This separates the **core** taxa (present in every sample) from **unique**/rare taxa (present in one or a few samples) - the expected pattern for pathogens and false positives. A membership TSV (presence matrix plus `N_Samples` and a Core/Shared/Unique `Category`) is written alongside each plot.

Usage

```
group_upset(project_dir, tax_level = "Species", CS = NULL, export = TRUE)
```

Arguments

`project_dir`	Path to the project root.
`tax_level`	Taxonomic rank to compare (default: `"Species"`).
`CS`	Confidence Score to analyse. `NULL` (default) uses the final mosaic; a numeric value (Kraken fraction `0-1` or percentage `0-100`) compares the imported data at that single CS.
`export`	Logical (default `TRUE`). When `TRUE`, an UpSet PDF and a membership TSV per group and domain are written to <project_dir>/008_taxa_intersections_across_samples/<gr together with a log. When `FALSE`, nothing is written and the results are only returned.

Details

By default the analysis uses the **final mosaic** from `retrieve_selected_taxa()`; set `CS` to compare at a single Confidence Score from the imported data instead.

Groups with a single sample cannot be compared and are skipped with a log message; if no group has at least two samples, the function warns that nothing was generated. `Category` is `"Core"` when a taxon is present in every sample of the group, `"Unique"` when it is present in exactly one, and `"Shared"` otherwise.

Value

A kariocas_result object. $data is the membership table: one row per group, domain and taxon, with N_Samples, Category, Unique_Sample and one 0/1 column per sample. $plots holds one UpSetR plot per group and domain (printing an element draws it) and $paths lists the files written when export = TRUE.

Examples

```
# The toy project has a single sample; add a second sample of the same
# group (a copy of SAMPLE01 named SAMPLE02) so they can be compared.
toy_project <- file.path(tempdir(), "toy_karioCaS_groups")
in_dir <- file.path(toy_project, "000_mpa_original")
dir.create(in_dir, recursive = TRUE, showWarnings = FALSE)
src <- list.files(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    full.names = TRUE
)
file.copy(src, in_dir)
file.copy(src, file.path(in_dir, sub("SAMPLE01", "SAMPLE02", basename(src))))
import_karioCaS(toy_project)

# Core vs unique genera across the samples of each group, at CS 40
res <- group_upset(toy_project, tax_level = "Genus", CS = 40, export = FALSE)
res
table(res$data$Domain, res$data$Category)

unlink(toy_project, recursive = TRUE)
```

---

heatmaps_karioCaS *Generate Heatmaps of Taxa Abundance with Extinction Patterns (Step 006)*

Description

Creates Relative Abundance (Confidence Score. "Elite" survivors are clustered by similarity, while lost taxa are aggregated into "Loss Groups" at the bottom. Handles domains with zero survivors at the target CS (showing only loss groups).

Usage

```
heatmaps_karioCaS(
  project_dir,
  analysis_rank = "Genus",
  confidence_score = NULL,
  top_n = 20,
  export = TRUE
)
```

Arguments

project_dir	Path to the project root.
analysis_rank	Taxonomic rank to analyze (default: "Genus").
confidence_score	Target CS to define survivors, as a Kraken fraction (0-1) or a percentage (0-100). NULL (default) uses the highest CS available in each sample, and says so in the log. A value above a sample's highest CS is replaced by that maximum with a warning.
top_n	Number of top survivors to display individually (default: 20).
export	Logical (default TRUE). When TRUE, one PDF per sample and a log are written to <project_dir>/006_relative_abundance_across_CS/. When FALSE, nothing is written and the heatmaps are only returned.

Value

A kariocas_result object. $data holds the plotted values (one row per sample, domain, taxon or loss group and CS, with Counts, Rel_Abund and the Target_CS used), $plots one ggplot heatmap per sample and $paths the files written when export = TRUE.

How the quantities are calculated

For each sample and domain, a taxon's reads at analysis_rank are the count of its own MPA row, which already includes the reads classified to lower ranks. The top_n taxa with most reads at the target CS are shown individually. The remaining taxa are aggregated: taxa present at a CS below the target but absent at the next CS form a row *"Recovered only in CSxx"*; taxa still present at the target CS but not among the top_n form a row *"Lowest abundance ... in CSxx"*. Rel_Abund is the reads of a row divided by the total reads of all taxa of that domain at analysis_rank at the same CS, times 100. Individual taxa are ordered by hierarchical clustering (Euclidean distance, complete linkage) of their relative-abundance profiles across CS.

Examples

```
# Copy the bundled toy project to a temporary folder and import it
toy_project <- file.path(tempdir(), "toy_karioCaS")
dir.create(toy_project, showWarnings = FALSE)
file.copy(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    toy_project,
    recursive = TRUE
)
import_karioCaS(toy_project)

# Species-level heatmap at CS 50, top 15 taxa, kept in memory
res <- heatmaps_karioCaS(
    toy_project,
    analysis_rank = "Species", confidence_score = 50, top_n = 15,
    export = FALSE
)
res
res$plots[[1]]

unlink(toy_project, recursive = TRUE)
```

`import_karioCaS` *Import Kraken MPA Reports to TreeSummarizedExperiment (Step 001)*

Description

Reads Kraken2 MPA-style reports from the `000_mpa_original` folder. Parses filenames like `SAMPLE_CSXX.mpa` (e.g., `PILO_CS09.mpa`). Generates a detailed log file for traceability and saves a `TreeSummarizedExperiment` object for downstream analysis.

Usage

```
import_karioCaS(project_dir)
```

Arguments

`project_dir`	Path to the project root. Must contain a `000_mpa_original` subfolder with `.mpa` files.

Details

Each report becomes one column of the count matrix, named `SAMPLE_CSxx`; the sample name and the CS (as an integer percentage, e.g. `CS09` = 90) are stored in `colData()`. Each distinct taxonomic path becomes one row: taxa absent from a report get a count of 0 in that column, and identical paths within a report are summed. The lineage is split into eight rank columns (Domain to Species) in `rowData()`, together with `Rank`, the lowest rank present in the path. No normalisation or transformation is applied; counts are the read counts of the reports, which in MPA format are cumulative (a genus count includes the reads of its species). The object and a tab-separated copy of the matrix are saved to `<project_dir>/001_imported_matrix/`, where the other steps read it from.

Value

Invisibly returns the `TreeSummarizedExperiment` object.

Examples

```
# Copy the bundled toy reports to a temporary project folder
toy_project <- file.path(tempdir(), "toy_karioCaS_import")
dir.create(toy_project, showWarnings = FALSE)
file.copy(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    toy_project,
    recursive = TRUE
)
tse <- import_karioCaS(toy_project)
tse
SummarizedExperiment::colData(tse)

unlink(toy_project, recursive = TRUE)
```

kariocas_result *Result of a karioCaS analysis step*

Description

Every analysis function of karioCaS returns an object of class `kariocas_result`. It bundles the computed table(s), the generated plots and, when `export = TRUE`, the paths of the files written to the project folder. Elements are accessed with `$`.

Usage

```
kariocas_result(
  step,
  data = NULL,
  plots = list(),
  paths = character(0),
  output_dir = NULL
)

## S3 method for class 'kariocas_result'
print(x, ...)
```

Arguments

`step`	Character. Name of the step, e.g. `"002_taxa_retention"`.
`data`	A data frame with the computed results, or `NULL`.
`plots`	A named list of ggplot/patchwork objects (may be empty).
`paths`	Character vector of files written (empty if nothing exported).
`output_dir`	Output folder, or `NULL` when not exporting.
`x`	A `kariocas_result` object.
`...`	Ignored.

Value

A list of class `kariocas_result` with elements `step`, `data`, `plots`, `paths` and `output_dir`.

Methods (by generic)

- `print(kariocas_result)`: Compact summary of a result object.

Examples

```
res <- kariocas_result(
    step = "demo", data = data.frame(x = 1:3),
    plots = list(), paths = character(0), output_dir = NULL
)
res
res$data
```

`reads_per_taxa` *Read Cutoff Saturation Analysis & Optimal Minimum Reads (Step 003)*

Description

Saturation analysis: progressively raises a per-taxon read-count cutoff and tracks how many taxa survive, on a log read axis. By default it draws one **group overlay** per biological group (every sample a faint line, group mean highlighted) and marks each domain's **median optimal minimum reads** - the elbow of the saturation curve, found with the same engine used for the optimal CS - as a dashed line. The per-sample optimal-reads values are written to `Reads_Audit_<rank>.tsv/.rds`, giving a quantitative threshold for excluding low-abundance background/false-positive taxa.

Usage

```
reads_per_taxa(
  project_dir,
  analysis_level = "Species",
  method = c("kneedle", "postcliff", "segmented"),
  detail_samples = NULL,
  export = TRUE
)
```

Arguments

`project_dir`	Path to the project root.
`analysis_level`	Taxonomic rank to analyze (default: `"Species"`).
`method`	Elbow strategy for the optimal reads. One of `"kneedle"` (default), `"postcliff"` or `"segmented"`.
`detail_samples`	Which samples to also render as detailed per-sample saturation panels. `NULL` (default) writes only the group overlays; `"all"` renders every sample; a comma-separated string such as `"SAMPLE33, SAMPLE45"` (or a character vector) renders just those. Detailed PDFs are saved to a `per_sample/` subfolder.
`export`	Logical (default `TRUE`). When `TRUE`, PDF plots, the `Reads_Audit_<rank>.tsv/.rds` files and a log are written to `<project_dir>/003_reads_saturation/`. When `FALSE`, nothing is written and all results are only returned. Note that `retrieve_selected_taxa` with `reads_min_* = "auto"` reads the exported audit file, so run this step with `export = TRUE` before it.

Details

The optimal reads is computed *per Confidence Score*, since the saturation curve changes with CS; read it off at your chosen optimal CS (Step 002).

Value

A `kariocas_result` object with `$data`, the optimal-reads audit table (one row per sample, CS, domain and read cutoff), `$plots`, a named list of `patchwork` figures (one saturation overlay per group and CS, plus detailed per-sample panels when requested) and `$paths`, the files written when `export = TRUE`.

How the quantities are calculated

For each sample, CS and domain, the taxa at `analysis_level` are counted at increasing minimum-read cutoffs (1, 2, 3, 4, 5, 7, 10, then 1, 2, 3, 5 and 7 $\times 10^k$ up to just past the largest count). `Taxa_Count` is the number of taxa with at least `Cutoff` reads, where a taxon's reads are the count of its own MPA row (which already includes reads classified to lower ranks). `Pct_Retained` is `Taxa_Count` as a percentage of the count at the lowest cutoff, and `Step_Loss_Pct` the loss since the previous cutoff. The optimal minimum reads (Primary SI) is the elbow of this curve, found with the same engine as in `taxa_retention` on a $\log_{10}$ read axis; with `"kneedle"`, the more conservative post-cliff cutoff is reported as Secondary SI when it is larger.

Examples

```
# Copy the bundled toy project to a temporary folder and import it
toy_project <- file.path(tempdir(), "toy_karioCaS")
dir.create(toy_project, showWarnings = FALSE)
file.copy(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    toy_project,
    recursive = TRUE
)
import_karioCaS(toy_project)

# Saturation overlays + optimal minimum reads (Kneedle), in memory only
res <- reads_per_taxa(toy_project, export = FALSE)
res
head(res$data)
res$plots[[1]]

unlink(toy_project, recursive = TRUE)
```

---

retrieve_selected_taxa

Retrieve Selected Taxa with Domain-Specific Thresholds (Step 004)

Description

Creates a "biological mosaic" for each sample using the `karioCaS_TSE.rds` object from Step 001 and, optionally, the optimal thresholds computed earlier: the optimal Confidence Score (Stability Index audit from `taxa_retention()`, Step 002) and the optimal minimum reads (`Reads_Audit` from `reads_per_taxa()`, Step 003). Both the `CS_*` and `reads_min_*` arguments accept `"auto"`, `"secondary"`, or a manual numeric value per domain. The optimal reads is looked up at each domain's resolved CS, so the mosaic combines both data-driven thresholds automatically. The mosaic always retains **all** taxonomic ranks present in the input (as an MPA profile does); enforces a strict `> 0` read filter.

Usage

```
retrieve_selected_taxa(
  project_dir,
  tax_level = NULL,
```

```
  CS_A = "auto",
  reads_min_A = 0,
  CS_B = "auto",
  reads_min_B = 0,
  CS_E = "auto",
  reads_min_E = 0,
  CS_V = "auto",
  reads_min_V = 0,
  export = TRUE
)
```

Arguments

project_dir	Path to the project root.
tax_level	Which rank's optimization audit the "auto" / "secondary" thresholds are read from, i.e. SI_Audit_<tax_level> and Reads_Audit_<tax_level> (NULL, the default, uses "Species"). This does *not* filter the output, which always contains all ranks.
CS_A	Character or numeric. CS for Archaea: "auto", "secondary", or a numeric value. Default: "auto".
reads_min_A	Minimum reads for Archaea: "auto"/"secondary" (pulled from the Reads_Audit at the resolved CS) or a numeric value. Default: 0.
CS_B	Character or numeric. CS for Bacteria. Default: "auto".
reads_min_B	Minimum reads for Bacteria. Default: 0.
CS_E	Character or numeric. CS for Eukaryota. Default: "auto".
reads_min_E	Minimum reads for Eukaryota. Default: 0.
CS_V	Character or numeric. CS for Viruses. Default: "auto".
reads_min_V	Minimum reads for Viruses. Default: 0.
export	Logical (default TRUE). When TRUE, one mosaic per sample is written to <project_dir>/004_final_ (.mpa files under mpa/, .tsv files under tsv/) together with a log. When FALSE, nothing is written and the mosaic is only returned; note that taxa_resolution and group_upset read the exported mosaic by default.

Details

"auto" and "secondary" read the audit files exported by taxa_retention and reads_per_taxa; if the required file is missing the function stops with an explanation rather than silently applying no threshold. When the audit exists but has no optimal value for a given sample and domain (typically a domain with too few taxa to fit a curve), CS 0 or 0 minimum reads is used for that domain and a single warning lists every such case.

Value

A kariocas_result object. $data is the mosaic in long format: one row per sample and taxon, with the read Counts at the selected CS, the Domain, the CS and Min_Reads applied, and where each threshold came from (CS_Source, Min_Reads_Source). $plots is empty and $paths lists the files written when export = TRUE.

Examples

```
# Copy the bundled toy project to a temporary folder and import it
toy_project <- file.path(tempdir(), "toy_karioCaS")
dir.create(toy_project, showWarnings = FALSE)
file.copy(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    toy_project,
    recursive = TRUE
)
import_karioCaS(toy_project)

# "auto" thresholds come from the exported audits of the previous steps
taxa_retention(toy_project)
reads_per_taxa(toy_project)

# Data-driven CS and minimum reads for Bacteria and Archaea,
# manual thresholds for Eukaryota and Viruses
mosaic <- retrieve_selected_taxa(
    toy_project,
    CS_B = "auto", reads_min_B = "auto",
    CS_A = "auto", reads_min_A = "auto",
    CS_E = 40, reads_min_E = 10,
    CS_V = 0, reads_min_V = 0
)
mosaic
unique(mosaic$data[, c("Domain", "CS", "CS_Source", "Min_Reads")])

unlink(toy_project, recursive = TRUE)
```

`taxa_resolution` *Generate Taxa Resolution Analysis (Step 007)*

Description

Creates stacked bar plots showing resolution efficiency between a Parent Rank and a Child Rank (how much of each parent clade's reads are resolved down to the child rank versus remaining parent-exclusive). Uses `max()` for parent aggregation to prevent double-counting of children in cumulative data.

Usage

```
taxa_resolution(
  project_dir,
  parent_level = "Genus",
  child_level = "Species",
  CS = NULL,
  top_n = 10,
  export = TRUE
)
```

Arguments

project_dir	Path to the project root.
parent_level	Name of the parent rank (default: "Genus").
child_level	Name of the child rank (default: "Species").
CS	Confidence Score to analyse. NULL (default) uses the final mosaic from retrieve_selected_taxa(). A numeric value (Kraken fraction 0-1 or percentage 0-100) analyses the imported data at that single CS instead of every CS.
top_n	Number of top taxa to display per domain (default: 10).
export	Logical (default TRUE). When TRUE, one PDF per sample and a log are written to <project_dir>/007_taxa_resolution/. When FALSE, nothing is written and the results are only returned.

Details

By default the analysis runs on the **final mosaic** produced by retrieve_selected_taxa (004_final_mosaic/), i.e. the data-driven high-confidence selection - one figure per sample. Alternatively, set CS to analyse the raw imported data at a single Confidence Score.

Value

A kariocas_result object. $data holds, for each sample, domain and top parent taxon, the parent's cumulative reads (Parent_Cumulative_Total), the reads resolved to the child rank (Child_Sum_Resolved) and the parent-exclusive reads (Parent_Exclusive); $plots one patchwork figure per sample; $paths the files written when export = TRUE.

How the quantities are calculated

MPA counts are cumulative, so a parent row (e.g. a genus) already includes the reads of its children (its species). For the top_n parent taxa with most reads in each domain: Parent_Cumulative_Total is the count of the parent's own row; Child_Sum_Resolved is the sum of its children's rows at child_level, i.e. the reads resolved down to the child rank; Parent_Exclusive is Parent_Cumulative_Total - Child_Sum_Resolved (floored at 0), the reads classified to the parent but not resolved to the child rank.

Examples

```
# Copy the bundled toy project to a temporary folder and import it
toy_project <- file.path(tempdir(), "toy_karioCaS")
dir.create(toy_project, showWarnings = FALSE)
file.copy(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    toy_project,
    recursive = TRUE
)
import_karioCaS(toy_project)

# Genus vs Species resolution of the imported data at CS 40, in memory
res <- taxa_resolution(toy_project, CS = 40, export = FALSE)
res
head(res$data)
res$plots[[1]]
```

```
unlink(toy_project, recursive = TRUE)
```

`taxa_retention` *Run Confidence Score Retention & Optimization (Step 002)*

Description

Executes taxa retention analysis based on Confidence Score (Kraken/Bracken) and, in the same step, computes the objective optimal CS (Stability Index, SI) for each domain. By default it produces a single, low-clutter **group overlay** per biological group: every sample of the group is drawn as a faint line with the group mean (±SD) highlighted and each domain's median optimal CS marked with a dashed line, faceted by Domain. It also writes the machine-readable SI audit (`SI_Audit_<rank>.tsv/.rds`) used by `retrieve_selected_taxa`. Detailed per-sample panels (all ranks, taxa vs reads) are written only on request.

Usage

```
taxa_retention(
  project_dir,
  tax_level = "Species",
  method = c("kneedle", "postcliff", "segmented", "dynamic", "manual"),
  manual_toll = 1,
  detail_samples = NULL,
  export = TRUE
)
```

Arguments

`project_dir`	Root path of the project.
`tax_level`	Taxonomic rank used for the group overlay and SI (default: `"Species"`).
`method`	Optimal-CS strategy. One of `"kneedle"` (default), `"postcliff"`, `"segmented"`, `"dynamic"`, or `"manual"`.
`manual_toll`	Numeric or named list. Acceptable step-wise loss percentage, used only when `method = "manual"` (default: 1.0).
`detail_samples`	Which samples to also render as detailed per-sample panels. `NULL` (default) writes only the group overlay; `"all"` renders every sample; a comma-separated string such as `"SAMPLE33, SAMPLE45"` (or a character vector) renders just those. Detailed PDFs are saved to a `per_sample/` subfolder.
`export`	Logical (default `TRUE`). When `TRUE`, PDF plots, the `SI_Audit_<rank>.tsv/.rds` files and a log are written to `<project_dir>/002_taxa_retention/`. When `FALSE`, nothing is written to disk and all results are only returned. Note that `retrieve_selected_taxa` with `CS_* = "auto"` reads the exported audit file, so run this step with `export = TRUE` before it.

Details

Groups are inferred from sample names by stripping trailing digits (e.g. `SAMPLE33`, `SAMPLE34` both belong to group `SAMPLE`).

The optimal CS is found with a multi-strategy engine: `"kneedle"` (default, parameter-free elbow detection), `"postcliff"` (a more conservative threshold past the steepest drop), `"segmented"` (broken-stick regression), `"dynamic"`, or `"manual"`.

Value

A kariocas_result object with $data, the SI audit table (one row per sample, domain and CS, with the retention percentages and the Stability Index tags), $plots, a named list of patchwork figures (one group overlay per biological group, plus the detailed per-sample panels when requested) and $paths, the files written when export = TRUE.

How the quantities are calculated

For each sample and domain, at each CS:

- Taxa_Count: number of distinct taxa at tax_level detected with at least one read.
- Pct_Retained: Taxa_Count as a percentage of its maximum across CS values (normally the value at CS 0).
- Step_Loss_Pct: percentage points of Pct_Retained lost since the previous CS.

The optimal CS (Primary SI) is then located on the Pct_Retained curve. "kneedle": the CS at which the curve lies furthest below the straight line joining its first and last points (falling back to the steepest single drop if the curve is not convex). "postcliff": the first CS, at or after the steepest drop, whose Step_Loss_Pct is at most a tolerance $\max(\bar{t} + 1.5s_t, 0.5)$, where $\bar{t}$ and s_t are the mean and SD of the step losses at CS ≥ 50. "segmented": the breakpoint minimising the total residual sum of squares of two straight-line fits. "dynamic": the first CS above the lowest one whose step loss is within the same tolerance. "manual": the first CS above the lowest one whose step loss is within manual_toll (a single value, or a named list per domain). When no CS qualifies, the highest CS is used. A Secondary SI, a stricter alternative, is also reported by "kneedle" (the post-cliff value, when stricter than the elbow), "dynamic" (the next CS whose step loss is at most $\max(\bar{t}, 0.1)$) and "manual" (the next CS whose step loss is at most 0.2). A domain needs at least three CS values and some loss of taxa to be assessed.

In the detailed per-sample panels, curves are expressed relative to CS 0: taxa at each rank (distinct taxa detected), reads classified to that rank (the sum of the rank's own MPA rows, which already include the reads classified to lower ranks) and total reads of the domain (the domain row).

Examples

```
# Copy the bundled toy project to a temporary folder and import it
toy_project <- file.path(tempdir(), "toy_karioCaS")
dir.create(toy_project, showWarnings = FALSE)
file.copy(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    toy_project,
    recursive = TRUE
)
import_karioCaS(toy_project)

# Retention curves + optimal CS (Kneedle), results kept in memory only
res <- taxa_retention(toy_project, export = FALSE)
res
head(res$data)
names(res$plots)
res$plots[[1]]

unlink(toy_project, recursive = TRUE)
```

upset_kariocas *Generate UpSet Plots per Sample and Domain (Step 005)*

Description

"karioCaS never are upset!" Generates UpSet plots showing taxon persistence across Confidence Score levels, for a single taxonomic rank, with detailed logging. One plot per sample and domain is written to a per-sample subfolder.

Usage

```
upset_kariocas(project_dir, tax_level = "Species", export = TRUE)
```

Arguments

project_dir	Path to the project root.
tax_level	Taxonomic rank to analyze (default: "Species").
export	Logical (default TRUE). When TRUE, one PDF per sample and domain is written to a per-sample subfolder of <project_dir>/005_taxa_intersections_across_CS/, together with a log. When FALSE, nothing is written and the UpSet plots are only returned.

Value

A kariocas_result object. $data is the presence/absence table behind the plots: one row per sample, domain and taxon, one CSxx column per Confidence Score (1 = detected with at least one read at that CS) and N_CS, the number of CS levels at which the taxon is detected. $plots holds one UpSetR plot per sample and domain; printing an element draws it on the current device. $paths lists the PDFs written when export = TRUE. If a plot cannot be drawn, a warning names it and it is not counted as a success.

Examples

```
# Copy the bundled toy project to a temporary folder and import it
toy_project <- file.path(tempdir(), "toy_karioCaS")
dir.create(toy_project, showWarnings = FALSE)
file.copy(
    system.file("extdata", "your_project_name", "000_mpa_original",
        package = "karioCaS"
    ),
    toy_project,
    recursive = TRUE
)
import_karioCaS(toy_project)

# Genus persistence across Confidence Scores, kept in memory
res <- upset_kariocas(toy_project, tax_level = "Genus", export = FALSE)
res
head(res$data)
res$plots[["SAMPLE01_Bacteria_Genus"]]

unlink(toy_project, recursive = TRUE)
```

Index

∗ **internal**
karioCaS-package, 2

group_upset, *2*, 3, *10*

heatmaps_karioCaS, *2*, 4

import_karioCaS, *2*, 6

karioCaS *(*karioCaS-package*)*, 2
karioCaS-package, 2
kariocas_result, *4, 5*, 7, *8*, *10*, *12*, *14, 15*

print.kariocas_result
*(*kariocas_result*)*, 7

reads_per_taxa, *2*, 8, *10*
retrieve_selected_taxa, *2*, *8*, 9, *12, 13*

taxa_resolution, *2*, *10*, 11
taxa_retention, *2*, *9, 10*, 13

upset_kariocas, *2*, 15