Package: OceanTracer (via r-universe)

October 15, 2026

Type Package

Title Marine Source Unmixing Toolkit

Version 1.0.2

Date 2026-10-04

Description Reconstructs the provenance of particulate matter and marine sediments by 'unmixing' isotopic and elemental signatures. Provides a specialized toolkit for marine biogeochemistry, coastal tracing, and oceanographic sediment transport following the step-by-step procedure described in Cepeda 'et' 'al.' (2026) <doi:10.1002/lom3.70083>, which is tested using artificial mixtures.

License GPL (>= 2)

Depends R (>= 4.1.0)

Imports ggplot2, klaR, GGally, plotly, gridExtra, Ternary, MASS, dplyr, magrittr, tidyr, crayon, reshape2, tidyselect, stats, utils, scales, cluster, ggbiplot, MixSIAR

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

NeedsCompilation no

Author Esther Cepeda [aut, cre], Maarten Wynants [aut], Ivan Lizaga [aut, ths, ldr], Samuel Bodé [ctb], Pascal Boeckx [ctb], Jan Vanaverbeke [ctb], Ulrike Braeckman [ctb], Margarita Villalonga [ctb], Aiming Liang [ctb], Borja Latorre [ctb], Leticia Gaspar [ctb], Ana Navas [ctb], Vince Q Vu [ctb]

Maintainer Esther Cepeda <cepeda.esther@hotmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-15 16:20:15 UTC

RemoteUrl https://github.com/cran/OceanTracer

RemoteRef HEAD

RemoteSha 93bcaaded1ecc779e141f8b7e734a4321e32e3f5

Contents

aa_baseline_correction . . . 2
artificial_dataset . . . 3
CI_Method . . . 4
cluster_solutions . . . 5
Conservative_Balance . . . 6
consistency_error . . . 7
Correlation_Chart . . . 8
CR_Method . . . 9
CTS_2_sources . . . 10
CTS_3_sources . . . 11
CTS_4_sources . . . 13
CTS_5_sources . . . 14
CTS_n_sources . . . 15
DFATest . . . 16
frequentist_unconstrained . . . 16
frequentist_unconstrained_lvp . . . 17
ggbiplot . . . 19
Graph_boxplots . . . 20
inputSample . . . 21
inputSource . . . 22
KWTest . . . 23
LDA_Chart . . . 24
plot_apportionment . . . 25
plot_pca . . . 27
prefilter_tracers . . . 28
rangeTest . . . 29
Ternary_Plot . . . 29
tracer_pull_analysis . . . 31
tracer_selection_diagnostic_plot . . . 32
triangles_dataset . . . 33
triangles_virtual . . . 34
unmix_mixsiar . . . 35
unmixing . . . 36

Index **38**

`aa_baseline_correction`

Amino Acid Isotopic Baseline Correction

Description

Corrects isotopic ratios by subtracting a source amino acid baseline (e.g., Phe or Lys) to isolate trophic enrichment. Automatically handles both raw data and averaged (Mean/SD) data, applying exact quadrature error propagation for standard deviations.

Usage

```
aa_baseline_correction(
  data,
  references = "Phe",
  invert = "Thr",
  drop_ref = TRUE
)
```

Arguments

`data`	A data frame containing isotopic ratios.
`references`	Character vector of the amino acid(s) to use as the baseline. If multiple are provided (e.g., `c("Phe", "Lys")`), their row-wise mean is used.
`invert`	Character vector of amino acids that undergo inverse trophic fractionation (typically `"Thr"`). The baseline will be added to these instead of subtracted.
`drop_ref`	Logical. If `TRUE`, the reference AA columns are removed from the final dataset. Default is `TRUE`.

Value

A `data.frame` of the same class as `data` with baseline-corrected isotopic values. For raw datasets, ratios are updated in place; for averaged datasets, means and standard deviations are updated with propagated quadrature errors. If `drop_ref = TRUE`, columns corresponding to `references` are omitted.

Examples

```
# Example using raw isotopic ratio data
raw_df <- data.frame(
  R_Phe = c(5.1, 5.3, 5.0),
  R_Glu = c(15.2, 16.0, 15.5),
  R_Thr = c(-2.1, -2.5, -2.0)
)
aa_baseline_correction(raw_df, references = "Phe", invert = "Thr")
```

`artificial_dataset` *Artificial Dataset for OceanTracer Workflows*

Description

Example dataset included with OceanTracer containing isotopic tracer ratios and concentration data for sediment sources and mixture samples.

Usage

```
artificial_dataset
```

Format

A data frame with 13 rows and 26 columns:

ID Character. Unique sample identifier.

Sources Factor or Character. Source classification group (`"S1"`, `"S2"`, `"S3"`) or mixture sample (`"Mx"`).

R_Ala, R_Asp, R_Glu, R_Ile, R_Leu, R_Pro, R_Val, R_Gly, R_Lys, R_Phe, R_Ser, R_Thr Numeric. Isotopic ratios for individual amino acids.

Conc_Ala, Conc_Asp, Conc_Glu, Conc_Ile, Conc_Leu, Conc_Pro, Conc_Val, Conc_Gly, Conc_Lys, Conc_Phe, Conc Numeric. Amino acid concentrations.

Details

The dataset contains isotopic amino acid tracer ratios and concentration measurements for three potential sediment sources (S1, S2, S3) and one mixture sample (Mx). It is designed for demonstrating source apportionment, conservativeness index (CI) calculations, and tracer selection workflows.

Value

A data frame containing the artificial data values.

Source

Synthetic example dataset created for the OceanTracer package.

Examples

```
data("artificial_dataset")
head(artificial_dataset)
```

CI_Method — *Conservativeness Index (CI) for individual tracer evaluation*

Description

Quantifies how consistently each tracer constrains the mixture apportionment when paired with randomly chosen complementary tracers.

Usage

```
CI_Method(
  data,
  points = 2000,
  Means = NULL,
  triangles = "virtual",
  seed = 123456L
)
```

Arguments

`data`	Data frame in standard OceanTracer format (sources + mixture).
`points`	Integer. Number of Monte Carlo draws per tracer. Default 2000.
`Means`	Logical or NULL (default). Auto-detected if NULL.
`triangles`	Character. Strategy to select accompanying tracers ("virtual" or "dataset").
`seed`	Integer. Random seed for reproducibility. Default 123456L.

Value

A list containing tracer output data frames, the original dataset, and the final summary CI index table.

`cluster_solutions` *Cluster CTS solutions by source contribution similarity*

Description

Takes the output of `CTS_n_sources` and groups the tracer combinations into clusters of similar source contributions (w1, w2, ...). This helps identify the dominant apportionment solutions and how many individual tracers support each one.

Usage

```
cluster_solutions(
  cts_result,
  top_n = 50L,
  k = NULL,
  max_k = 6L,
  interactive = FALSE,
  seed = NULL
)
```

Arguments

`cts_result`	Data frame returned by `CTS_n_sources` or any of the `CTS_*_sources` functions.
`top_n`	Integer. Number of top rows to use for clustering. Default 50.
`k`	Integer or `NULL`. Number of clusters. If `NULL` (default), chosen automatically via silhouette width.
`max_k`	Integer. Maximum k to evaluate when `k = NULL`. Default 6.
`interactive`	Boolean. If `TRUE` and n_sources == 3, produces an interactive 3D plotly scatter. Ignored for 2 or 4+ sources. Default `FALSE`.
`seed`	Integer or `NULL`. Random seed for reproducibility.

Details

Hierarchical clustering with Ward.D2 linkage is used because contribution clusters are rarely spherical or equal-sized. The optimal number of clusters is selected automatically via the average silhouette width criterion, but can be overridden with the `k` argument.

Visualisation adapts to the number of sources:

- 2 sources: single 2D panel (w1 vs w2).
- 3 sources: two 2D panels (w1~w2, w1~w3) plus optional interactive 3D plotly scatter.
- 4+ sources: all pairwise 2D panels as a grid. Interactive 3D is not available (dimensionality reduction would obscure interpretation).

Value

A named list:

`summary` Data frame: one row per cluster with mean/SD of each contribution, n_pairs, n_unique_tracers, avg_silhouette.

`tracers` Named list. Each element has `$pairs` (character vector) and `$individual_tracers` (data frame ranked by frequency).

`plot` ggplot2 or plotly visualisation.

`Conservative_Balance` *Conservative Balance (CB) transformation for isotopic tracers*

Description

Transforms isotopic ratio and elemental content data into virtual scalar tracers that can be directly used in classical unmixing models.

Usage

```
Conservative_Balance(
  data,
  error_propagation = "ratio",
  impute_nas = FALSE,
  ...
)

CB_Method(data, ...)

cb_method(...)
```

Arguments

`data`	Data frame. Contains the isotopic tracer characteristics of sediment sources and one mixture. Both averaged and raw formats are supported.
`error_propagation`	Character. Error propagation method for averaged data: "ratio" (default) or "full".
`impute_nas`	Logical. If TRUE, automatically imputes missing values (NAs) in source samples using their respective group means. Default FALSE.
`...`	Additional arguments passed to internal helper functions or aliases.

Value

A data frame with isotopic tracers replaced by their CB-transformed scalar equivalents.

`consistency_error` *Consistency error of a tracer selection for an apportionment solution*

Description

Assesses the mathematical consistency of a tracer selection by computing the normalized error between the predicted and observed tracer concentrations in the mixture. A low error (typically below 0.05) for all tracers indicates a consistent selection. High errors point to non-conservative tracers or multiple partial solutions in the dataset.

Usage

```
consistency_error(data, solution)

CTS_error(data, solution)

cts_error(data, solution)
```

Arguments

`data`	A data frame containing the characteristics of sediment sources and one mixture, in the standard fingerprinting format.
`solution`	Either a numeric vector of apportionment weights (one per source, in the same order as they appear in `data`), or a single-row data frame with columns named `w1`, `w2`, ... as returned by the unmixing functions.

Details

The error for each tracer is the absolute difference between the observed mixture concentration and the virtual mixture reconstructed from the apportionment solution, after normalizing by the range of the source confidence intervals.

Value

A data frame with columns `tracer` and `consistency_error`, one row per tracer.

References

Latorre, B., Lizaga, I., Gaspar, L., & Navas, A. (2021). A novel method for analysing consistency and unravelling multiple solutions in sediment fingerprinting. *Science of The Total Environment*, *789*, 147804.

`Correlation_Chart` *Correlation matrix chart*

Description

Generates a correlation matrix scatterplot of dataset properties, grouped by source, to aid in collinearity diagnosis and optimal tracer selection.

Usage

```
Correlation_Chart(
  data,
  tracers = 1:(ncol(data) - 2),
  mixtures = TRUE,
  nmixtures = 2,
  colors = NULL,
  seed = NULL
)

correlationPlot(
  data,
  tracers = 1:(ncol(data) - 2),
  mixtures = TRUE,
  nmixtures = 2,
  colors = NULL,
  seed = NULL
)

correlation_plot(
  data,
  tracers = 1:(ncol(data) - 2),
  mixtures = TRUE,
  nmixtures = 2,
  colors = NULL,
  seed = NULL
)
```

Arguments

`data`	Data frame. Contains source and mixture data in the standard fingerprinting format.
`tracers`	Numeric vector. Indices of the tracers to include in the matrix. Default `1:(ncol(data) - 2)`.
`mixtures`	Logical. If TRUE, includes mixture samples in the correlation plots. Default TRUE.
`nmixtures`	Integer. Number of mixture samples present in the dataset. Default 2.
`colors`	Character vector. Custom colors for the scatterplot groups. Default NULL.
`seed`	Integer. Random seed for reproducibility. Default NULL.

Value

A GGally::ggpairs object.

CR_Method — *Consensus Ranking (CR) method for tracer selection*

Description

Computes the Consensus Ranking (CR) method, an ensemble technique to identify non-conservative and dissenting tracers. The method combines predictions from single-tracer models and is based on a scoring function derived from a series of random "debates" between tracers, utilizing advanced thresholding logic for extreme datasets.

Usage

```
CR_Method(
  data,
  debates = 1000,
  completion_quantile = 0.9,
  apportionment_threshold = 0.05,
  seed = 123456,
  verbose = TRUE
)

CR(
  data,
  debates = 1000,
  completion_quantile = 0.9,
  apportionment_threshold = 0.05,
  seed = 123456,
  verbose = TRUE
)
```

```
cr_ns(
  data,
  debates = 1000,
  completion_quantile = 0.9,
  apportionment_threshold = 0.05,
  seed = 123456,
  verbose = TRUE
)
```

Arguments

`data` Data frame. Contains source and mixture data in the standard OceanTracer format.

`debates` Integer. Target number of random debates per tracer. Default 1000.

`completion_quantile` Numeric [0,1]. Quantile threshold for debate completion. For example, 0.5 stops the simulation once half the tracers hit the debates target. Default 0.9.

`apportionment_threshold` Numeric. Tolerance boundary for valid debate apportionments (e.g., defining valid as -threshold to 1 + threshold). Default 0.05.

`seed` Integer. Random seed for reproducibility. Default 123456.

`verbose` Logical. Whether to show a progress bar in the console. Default is TRUE.

Value

A data frame containing the CR score, debates won, and debates lost per tracer.

References

Lizaga, I., Latorre, B., Gaspar, L., & Navas, A. (2020). Consensus ranking as a method to identify non-conservative and dissenting tracers in fingerprinting studies. *Science of The Total Environment*, *720*, 137537. https://doi.org/10.1016/j.scitotenv.2020.137537

CTS_2_sources *CTS Engine for 2 Sources*

Description

Solves determined systems for 1 tracer and 2 sources to assess solution dispersion and discriminant capacity.

Usage

```
CTS_2_sources(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

singles(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

CTS_seeds_singles(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)
```

Arguments

source	Data frame of sediment sources containing source means, standard deviations, and sample sizes.
mixture	Data frame of mixture samples.
iter	Number of Monte Carlo iterations. Default is 1000.
seed	Integer seed for reproducibility. Default is 123456.
verbose	Logical. Whether to show a progress bar in the console. Default is TRUE.

Value

A `data.frame` ordered by `max_dw` (lowest dispersion first) containing:

id	Tracer name identifier.
w1, w2	Calculated proportion contributions for Source 1 and Source 2.
dw1, dw2	Dispersion uncertainties for Source 1 and Source 2 (half 84% - 16% quantile range).
cons	Proportion of Monte Carlo iterations yielding physically valid proportions between 0 and 1.
max_dw	Maximum dispersion uncertainty across both sources.

Examples

```
# Prepare small dummy datasets
src <- data.frame(
  ID = c("Src1", "Src2"),
  Tracer1 = c(10.5, 20.2),
  Tracer1_SD = c(1.1, 1.8),
  n = c(10, 10)
)
mix <- data.frame(ID = "Mix1", Tracer1 = 15.0)

# Run engine
result <- CTS_2_sources(source = src, mixture = mix, iter = 100, verbose = FALSE)
head(result)
```

CTS_3_sources *CTS Engine for 3 Sources*

Description

Solves determined systems for 2 tracers and 3 sources to assess solution dispersion and discriminant capacity.

Usage

```
CTS_3_sources(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

pairs(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

CTS_seeds_pairs(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)
```

Arguments

source	Data frame of sediment sources containing 3 source rows, mean values, standard deviations, and sample size ('n').
mixture	Data frame of mixture samples.
iter	Number of Monte Carlo iterations. Default is 1000.
seed	Integer seed for reproducibility. Default is 123456.
verbose	Logical. Whether to show a progress bar in the console. Default is TRUE.

Value

A `data.frame` ordered by `max_dw` (lowest dispersion first) containing:

id	Pair of tracer names used for the system.
w1, w2, w3	Calculated proportion contributions for Sources 1, 2, and 3.
dw1, dw2, dw3	Dispersion uncertainties for Sources 1, 2, and 3 (half 84% - 16% quantile range).
cons	Proportion of Monte Carlo iterations yielding physically valid proportions between 0 and 1.
max_dw	Maximum dispersion uncertainty across all three sources.

Examples

```
# Prepare small dummy datasets for 3 sources and 2 tracers
src <- data.frame(
  ID = c("Src1", "Src2", "Src3"),
  Tracer1 = c(10.5, 20.2, 5.0),
  Tracer2 = c(2.1, 8.5, 12.3),
  Tracer1_SD = c(1.1, 1.8, 0.5),
  Tracer2_SD = c(0.3, 0.9, 1.1),
  n = c(10, 10, 10)
)
mix <- data.frame(ID = "Mix1", Tracer1 = 12.0, Tracer2 = 6.0)

# Run engine
result <- CTS_3_sources(source = src, mixture = mix, iter = 100, verbose = FALSE)
head(result)
```

CTS_4_sources *CTS Engine for 4 Sources*

Description

Solves determined systems for 3 tracers and 4 sources to assess solution dispersion and discriminant capacity.

Usage

```
CTS_4_sources(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

triplets(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

CTS_seeds_triplets(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)
```

Arguments

source	Data frame of sediment sources containing 4 source rows, mean values, standard deviations, and sample size ('n').
mixture	Data frame of mixture samples.
iter	Number of Monte Carlo iterations. Default is 1000.
seed	Integer seed for reproducibility. Default is 123456.
verbose	Logical. Whether to show a progress bar in the console. Default is TRUE.

Value

A `data.frame` ordered by `max_dw` (lowest dispersion first) containing:

id	Triplet of tracer names used for the system.
w1, w2, w3, w4	Calculated proportion contributions for Sources 1 through 4.
dw1, dw2, dw3, dw4	Dispersion uncertainties for Sources 1 through 4 (half 84% - 16% quantile range).
cons	Proportion of Monte Carlo iterations yielding physically valid proportions.
max_dw	Maximum dispersion uncertainty across all four sources.

Examples

```
# Prepare small dummy datasets for 4 sources and 3 tracers
src <- data.frame(
  ID = c("Src1", "Src2", "Src3", "Src4"),
  Tracer1 = c(10.5, 20.2, 5.0, 15.1),
  Tracer2 = c(2.1, 8.5, 12.3, 4.2),
  Tracer3 = c(0.5, 1.2, 3.4, 0.9),
  Tracer1_SD = c(1.1, 1.8, 0.5, 0.8),
```

```
  Tracer2_SD = c(0.3, 0.9, 1.1, 0.4),
  Tracer3_SD = c(0.1, 0.2, 0.3, 0.1),
  n = c(10, 10, 10, 10)
)
mix <- data.frame(ID = "Mix1", Tracer1 = 12.0, Tracer2 = 6.0, Tracer3 = 1.5)

# Run engine
result <- CTS_4_sources(source = src, mixture = mix, iter = 100, verbose = FALSE)
head(result)
```

`CTS_5_sources` *CTS Engine for 5 Sources*

Description

Solves determined linear systems for 4 tracers and 5 sources using Monte Carlo simulation to assess solution dispersion and discriminant capacity.

Usage

```
CTS_5_sources(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

quartets(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)

CTS_seeds_quartets(source, mixture, iter = 1000, seed = 123456, verbose = TRUE)
```

Arguments

`source`	Data frame of sediment sources containing 5 source rows, tracer mean values, standard deviations, and sample size ('n').
`mixture`	Data frame of mixture samples containing tracer measurements.
`iter`	Number of Monte Carlo iterations. Default is 1000.
`seed`	Integer seed for reproducibility. Default is 123456.
`verbose`	Logical. Whether to show a progress bar in the console. Default is TRUE.

Value

A `data.frame` ordered by `max_dw` (lowest dispersion first) containing:

`id`	Quartet of tracer names used for the system.
`w1`, `w2`, `w3`, `w4`, `w5`	Calculated proportion contributions for Sources 1 through 5.
`dw1`, `dw2`, `dw3`, `dw4`, `dw5`	Dispersion uncertainties for Sources 1 through 5 (half 84% - 16% quantile range).
`cons`	Proportion of Monte Carlo iterations yielding physically valid proportions.
`max_dw`	Maximum dispersion uncertainty across all five sources.

Examples

```
# Prepare small dummy datasets for 5 sources and 4 tracers
src <- data.frame(
  ID = c("Src1", "Src2", "Src3", "Src4", "Src5"),
  Tracer1 = c(10.5, 20.2, 5.0, 15.1, 8.4),
  Tracer2 = c(2.1, 8.5, 12.3, 4.2, 6.7),
  Tracer3 = c(0.5, 1.2, 3.4, 0.9, 2.1),
  Tracer4 = c(100, 250, 400, 180, 310),
  Tracer1_SD = c(1.1, 1.8, 0.5, 0.8, 0.6),
  Tracer2_SD = c(0.3, 0.9, 1.1, 0.4, 0.5),
  Tracer3_SD = c(0.1, 0.2, 0.3, 0.1, 0.2),
  Tracer4_SD = c(10, 20, 35, 15, 22),
  n = c(10, 10, 10, 10, 10)
)
mix <- data.frame(ID = "Mix1", Tracer1 = 12.0, Tracer2 = 6.0, Tracer3 = 1.5, Tracer4 = 220)

# Run engine
result <- CTS_5_sources(source = src, mixture = mix, iter = 100, verbose = FALSE)
head(result)
```

CTS_n_sources *Consistent Tracer Selection (CTS) for multiple sources*

Description

Identifies the most discriminant minimal tracer combinations (n-1) by solving determined systems and assessing solution dispersion. Lower dispersion indicates higher discriminant capacity.

Usage

```
CTS_n_sources(data, iter = 1000, seed = 123456)

CTS_seeds(data, iter = 1000, seed = 123456)
```

Arguments

data	Data frame containing sediment source and mixture data.
iter	Number of Monte Carlo iterations for variability analysis.
seed	Integer for reproducibility.

Value

A data frame with all tracer combinations, source apportionments (wi), physical feasibility (0 < wi < 1), and solution dispersion (sd_wi). Results are sorted by the lowest maximum dispersion.

References

Latorre, B., Lizaga, I., Gaspar, L., & Navas, A. (2021). A novel method for analysing consistency and unravelling multiple solutions in sediment fingerprinting.

DFATest — *Discriminant Function Analysis for tracer pre-selection*

Description

Performs stepwise forward variable selection using the Wilks Lambda criterion. Tracers are added one at a time when they improve inter-source discrimination beyond the specified significance threshold.

Usage

```
DFATest(data, niveau = 0.1, verbose = TRUE)
```

Arguments

`data`	Data frame in the standard OceanTracer format (sources + mixture).
`niveau`	Numeric. Significance level for the F-test at each step. Default 0.1.
`verbose`	Logical. Whether to print DFA selection summaries to the console. Default is `TRUE`.

Value

A list containing the retained data frame and the DFA model output.

---

frequentist_unconstrained — *Source apportionment via unconstrained least-squares with classical variability propagation*

Description

Estimates the fractional contribution of each sediment source to a mixture by solving an overdetermined mass-balance system with least-squares minimisation. Source variability is propagated by Monte Carlo resampling: at each iteration, synthetic source signatures are drawn from a normal distribution defined by the measured mean and standard deviation, and the system is solved independently for each draw.

Usage

```
frequentist_unconstrained(
  data,
  variability = "SEM",
  iter = 1000,
  means = FALSE,
  seed = 123456L,
  verbose = TRUE
)
```

Arguments

`data`	Data frame in the standard OceanTracer format (sources + mixture).
`variability`	Character. `"SEM"` for Standard Error of the Mean, `"SD"` for Standard Deviation.
`iter`	Integer or character. Number of Monte Carlo iterations (e.g., `1000`, or character keywords like `"short"`, `"medium"`, `"long"`). Default is 1000.
`means`	Logical. `TRUE` when the input is in averaged format (`Mean_/SD_` columns); `FALSE` for raw observations.
`seed`	Integer. Random seed for reproducibility. Default is 123456L.
`verbose`	Logical. Whether to show progress bars and summary messages in the console. Default is TRUE.

Details

This function is the computational back-end called by `unmixing`. For most users, calling `unmixing` directly is recommended.

Value

A data frame with one row per iteration containing the mixture ID, goodness-of-fit (GOF), and one column per source with the estimated fractional contribution.

`frequentist_unconstrained_lvp` *Source apportionment via unconstrained least-squares with linear variability propagation (LVP)*

Description

Estimates the fractional contribution of each sediment source to a mixture by solving an overdetermined mass-balance system with least-squares minimisation. Unlike classical Monte Carlo propagation, uncertainty is handled analytically through Linear Variability Propagation (LVP): the Jacobian of the least-squares solution propagates source covariances directly into the apportionment uncertainty, avoiding assumptions of independent normally distributed errors.

Usage

```
frequentist_unconstrained_lvp(
  data,
  variability = "SEM",
  iter = 1000,
  means = FALSE,
  seed = 123456L,
  verbose = TRUE
)
```

Arguments

`data`	Data frame in the standard OceanTracer format (sources + mixture).
`variability`	Character. `"SEM"` for Standard Error of the Mean, `"SD"` for Standard Deviation.
`iter`	Integer or character. Number of iterations (e.g., `1000`, or character keywords like `"short"`, `"medium"`, `"long"`). Default is 1000.
`means`	Logical. `TRUE` when the input is in averaged format (`Mean_/SD_` columns); `FALSE` for raw observations.
`seed`	Integer. Random seed for reproducibility. Default is 123456L.
`verbose`	Logical. Whether to show progress bars and summary messages in the console. Default is TRUE.

Details

LVP is recommended when source variability is high or when apportionment are near the physical boundaries (0 or 1), where Monte Carlo resampling tends to underestimate uncertainty.

This function is the computational back-end called by `unmixing` when `lvp = TRUE`. For most users, calling `unmixing` directly is recommended.

Value

A data frame with one row per iteration containing the mixture ID, goodness-of-fit (GOF), and one column per source with the estimated fractional contribution.

References

Latorre, B., Lizaga, I., Gaspar, L., & Navas, A. (2025). Evaluating the impact of high source variability and extreme contributing sources on sediment fingerprinting models. *Water Resources Management*. doi:10.1007/s11269025041698

ggbiplot *Biplot for Principal Components using ggplot2*

Description

Renders a custom publication-ready biplot for PCA models created via `prcomp()`, `princomp()`, `FactoMineR::PCA()`, or `MASS::lda()`.

Usage

```
ggbiplot(
  pcobj,
  choices = 1:2,
  scale = 1,
  pc.biplot = TRUE,
  obs.scale = 1 - scale,
  var.scale = scale,
  groups = NULL,
  ellipse = FALSE,
  ellipse.prob = 0.68,
  labels = NULL,
  labels.size = 3,
  alpha = 1,
  var.axes = TRUE,
  circle = FALSE,
  circle.prob = 0.69,
  varname.size = 3,
  varname.adjust = 1.5,
  varname.abbrev = FALSE,
  ...
)
```

Arguments

`pcobj`	Object. A PCA object returned by `prcomp`, `princomp`, or `PCA()`.
`choices`	Numeric vector of length 2. Indices of principal components to plot. Default `c(1, 2)`.
`scale`	Numeric. Scale factor (0 or 1). Covariance biplot (scale = 1), form biplot (scale = 0). Default 1.
`pc.biplot`	Logical. If `TRUE`, ensures compatibility with `biplot.princomp()` scaling. Default `TRUE`.
`obs.scale`	Numeric. Scale factor applied to observations. Default `1 - scale`.
`var.scale`	Numeric. Scale factor applied to variables. Default `scale`.
`groups`	Factor. Optional grouping variable to color-code observations. Default `NULL`.

ellipse	Logical. If TRUE, draws a normal concentration ellipse for each group. Default FALSE.
ellipse.prob	Numeric. Probability level for concentration ellipse size. Default 0.68.
labels	Character vector. Optional custom labels for observations. Default NULL.
labels.size	Numeric. Text size used for observation labels. Default 3.
alpha	Numeric [0, 1]. Transparency for points. Default 1.
var.axes	Logical. If TRUE, renders variable loading arrows. Default TRUE.
circle	Logical. If TRUE, draws a correlation circle. Default FALSE.
circle.prob	Numeric. Size of the correlation circle in Normal probability. Default 0.69.
varname.size	Numeric. Text size for variable names. Default 3.
varname.adjust	Numeric. Adjustment factor for variable name placement (≥ 1 moves text away from arrow). Default 1.5.
varname.abbrev	Logical. If TRUE, abbreviates variable names. Default FALSE.
...	Additional arguments passed to ggplot2.

Value

A ggplot object.

Graph_boxplots *Box and Whiskers Plot for Tracers*

Description

Visualizes the distribution of continuous tracer variables across sources and mixtures. It displays summary statistics (median, hinges, whiskers) alongside individual outliers.

Usage

```
Graph_boxplots(
  data,
  tracers = 1:(ncol(data) - 2),
  ncol = 3,
  colors = NULL,
  seed = NULL
)

boxPlot(
  data,
  tracers = 1:(ncol(data) - 2),
  ncol = 3,
  colors = NULL,
  seed = NULL
)
```

```
box_plot(
  data,
  tracers = 1:(ncol(data) - 2),
  ncol = 3,
  colors = NULL,
  seed = NULL
)
```

Arguments

`data`	Data frame containing source and mixture data in the standard format.
`tracers`	Numeric vector. Indices of the tracers to plot (1 refers to the first tracer column). Default is 1 to (ncol(data) - 2).
`ncol`	Integer. Number of plot columns per row in the grid layout. Default is 3.
`colors`	Character vector. Optional hex codes or color names for custom boxplot colors. Default is NULL.
`seed`	Integer. Random seed for reproducible jittering or sampling. Default is NULL.

Value

Invisibly returns the combined plot grid object created by `gridExtra::grid.arrange()`.

Examples

```
data <- data.frame(
   ID = paste0("S", 1:12),
   Group = rep(c("SourceA", "SourceB", "Mixture"), each = 4),
   Tracer1 = c(10, 12, 11, 13, 20, 22, 21, 23, 15, 16, 15, 17),
   Tracer2 = c(5, 6, 5, 7, 15, 14, 16, 15, 10, 11, 10, 12)
)
Graph_boxplots(data, tracers = 1:2, ncol = 2)
```

`inputSample` *Extract mixture samples from a fingerprinting dataset*

Description

Identifies and returns the rows corresponding to the sediment mixture(s) in a combined source-mixture data frame. The mixture is assumed to be the last unique group in the second column of `data`. Both raw and averaged (`Mean_/SD_`) formats are supported.

Usage

```
inputSample(data)

inputMixture(data)
```

Arguments

`data`	Data frame in the standard OceanTracer format, where the second column encodes the group identity (sources and mixture).

Value

A data frame containing only the mixture rows, with column names standardised to the OceanTracer convention.

See Also

`inputSource` for extracting source rows.

`inputSource` *Input Sediment Sources*

Description

Selects and extracts the source samples from the dataset, computing mean, standard deviation, and sample size (`n`) when the input is in raw format. When pre-aggregated data is provided, column names are normalized to `Mean_/SD_` so that all downstream functions work correctly.

Usage

```
inputSource(data, na.omit = TRUE)
```

Arguments

`data`	Data frame containing source and mixture data.
`na.omit`	Logical. If `TRUE` (default), `NA` values are omitted when computing mean and SD.

Value

A `data.frame` where each row represents a sediment source containing:

`id`	Factor or character vector identifying the source group.
`Mean_...`	Mean tracer values for each tracer.
`SD_...`	Standard deviations of tracer values for each tracer.
`n`	Sample size for each source group.

Examples

```
# Example 1: Processing raw sample data
raw_df <- data.frame(
  ID = paste0("S", 1:9),
  Group = rep(c("SourceA", "SourceB", "Mixture"), each = 3),
  Tracer1 = c(10, 12, 11, 20, 22, 21, 15, 16, 17),
  Tracer2 = c(5, 6, 5, 15, 14, 16, 10, 11, 12)
)
inputSource(raw_df)

# Example 2: Normalizing pre-averaged data with legacy column names
summary_df <- data.frame(
  ID = c("Src1", "Src2", "Mix"),
  Group = c("SourceA", "SourceB", "Mixture"),
  mean_Tracer1 = c(11.0, 21.0, 16.0),
  mean_Tracer2 = c(5.33, 15.0, 11.0),
  sd_Tracer1 = c(1.0, 1.0, 1.0),
  sd_Tracer2 = c(0.58, 1.0, 1.0),
  n = c(3, 3, 3)
)
inputSource(summary_df)
```

KWTest — *Kruskal-Wallis test for tracer pre-selection*

Description

Retains only tracer properties that show statistically significant differences across source groups according to the non-parametric Kruskal-Wallis rank-sum test. Tracers with no discriminant power add noise to the unmixing model and should be removed before unmixing.

Usage

```
KWTest(data, pvalue = 0.05, verbose = TRUE)
```

Arguments

`data`	Data frame in the standard OceanTracer format (sources + mixture).
`pvalue`	Numeric. Significance threshold. Default 0.05.
`verbose`	Logical. Whether to print test summaries to the console. Default is `TRUE`.

Value

A data frame retaining only tracer columns with p-value below pvalue.

LDA_Chart *Linear Discriminant Analysis Chart*

Description

Performs a linear discriminant analysis (LDA) to project data into maximum-separation dimensions. Includes a comprehensive discrimination summary detailing variance explained, a leave-one-out cross-validation confusion matrix, and misclassification rates to rigorously assess source separability.

Usage

```
LDA_Chart(
  data,
  Plot3D = FALSE,
  text = FALSE,
  colors = NULL,
  interactive = FALSE,
  seed = NULL,
  verbose = TRUE
)

LDA_plot(
  data,
  Plot3D = FALSE,
  text = FALSE,
  colors = NULL,
  interactive = FALSE,
  seed = NULL,
  verbose = TRUE
)

LDAPlot(
  data,
  Plot3D = FALSE,
  text = FALSE,
  colors = NULL,
  interactive = FALSE,
  seed = NULL,
  verbose = TRUE
)
```

Arguments

data	Data frame. Contains source and mixture data in standard format.
Plot3D	Logical. If TRUE, renders a 3D scatter plot instead of a 2D projection. Default is FALSE.

text	Logical. If TRUE, overlays the sample identification numbers on the plot. Default is FALSE.
colors	Character vector. Custom colors for the source groups. Default is NULL.
interactive	Logical. If TRUE, renders an interactive HTML widget for 3D plots. Default is FALSE.
seed	Integer. Random seed for reproducibility. Default is NULL.
verbose	Logical. Whether to print discrimination summaries and variance explained to the console. Default is TRUE.

Value

Invisibly returns a named list containing:

lda	An object of class lda representing the fitted linear discriminant model.
confusion	A cross-tabulation table of leave-one-out cross-validation predictions versus actual sources.
prop_var	A numeric vector indicating the percentage of variance explained by each discriminant axis.

Examples

```
data <- data.frame(
   ID = paste0("S", 1:12),
   Group = rep(c("SourceA", "SourceB", "SourceC", "Mixture"), each = 3),
   Tracer1 = c(10, 12, 11, 20, 22, 21, 30, 31, 32, 20, 21, 22),
   Tracer2 = c(5, 6, 5, 15, 14, 16, 25, 26, 24, 15, 15, 16),
   Tracer3 = c(1, 2, 1.5, 5, 6, 5.5, 9, 10, 9.5, 5, 5.5, 6)
)
LDA_Chart(data, seed = 123)
```

---

plot_apportionment *Plot Apportionment Results*

Description

Displays the relative contribution of sediment sources to each mixture as density or violin plots. Designed to take the output of unmixing functions directly.

Usage

```
plot_apportionment(
  data,
  violin = FALSE,
  bounds = c(0, 1),
  scaled = TRUE,
  y_high = 1,
```

```
  colors = NULL,
  vlines = NULL,
  title = NULL,
  bw = "nrd"
)

plotResults(
  data,
  violin = FALSE,
  bounds = c(0, 1),
  scaled = TRUE,
  y_high = 1,
  colors = NULL,
  vlines = NULL,
  title = NULL,
  bw = "nrd"
)

plot_results(
  data,
  violin = FALSE,
  bounds = c(0, 1),
  scaled = TRUE,
  y_high = 1,
  colors = NULL,
  vlines = NULL,
  title = NULL,
  bw = "nrd"
)
```

Arguments

`data`	A data frame, typically the output from `unmix()`, containing the relative source contributions. The first two columns must be the mixture identifier (e.g. `id`) and the goodness-of-fit (`gof`), followed by one column per source. Only the source columns are plotted.
`violin`	Logical. If `TRUE`, violin + boxplot charts are used instead of density plots. Default is `FALSE`.
`bounds`	A numeric vector of length 2 specifying the lower and upper display bounds (the camera window). Default is `c(0.0, 1.0)`.
`scaled`	Logical. If `TRUE`, density curves are area-scaled to a common maximum. Only used when `violin = FALSE`. Default is `TRUE`.
`y_high`	Maximum value for the y-axis. Only used when `violin = FALSE` and `scaled = FALSE`. Default is `1`.
`colors`	An optional named character vector of colors, one per source. Default is `NULL`.
`vlines`	An optional named numeric vector specifying true/expected values for specific sources. Draws a dashed vertical line to allow visual validation of the model's accuracy (e.g., `c("SourceA" = 0.45)`). Default is `NULL`.

title	Optional character string to add a main title to the plot. Default is NULL.
bw	The smoothing bandwidth to be used. Can be a numeric value (e.g., 0.06) or a character string like "nrd" (the default automatic method).

Value

A ggplot object visualizing the source contribution distributions.

Examples

```
# Generate mock unmixing output
set.seed(123)
mock_unmix <- data.frame(
  id = 1:100,
  gof = runif(100, 85, 95),
  SourceA = rnorm(100, mean = 0.45, sd = 0.05),
  SourceB = rnorm(100, mean = 0.35, sd = 0.04),
  SourceC = rnorm(100, mean = 0.20, sd = 0.03)
)

# Density Plot
plot_apportionment(mock_unmix, title = "Source Contributions (Density)")

# Violin Plot with true values
true_vals <- c("SourceA" = 0.45, "SourceB" = 0.35, "SourceC" = 0.20)
plot_apportionment(
  mock_unmix, violin = TRUE,
  vlines = true_vals,
  title = "Source Contributions (Violin)"
)
```

---

plot_pca *Principal Component Analysis biplot*

Description

Performs a PCA on the tracer data and displays a biplot with source group ellipses. Averaged datasets (with Mean_/SD_ or mean_/sd_ columns) are automatically expanded to individual observations by sampling from a normal distribution before the PCA is computed, so all formats are accepted.

Usage

```
plot_pca(data, components = c(1L, 2L), colors = NULL, seed = NULL)

PCAPlot(data, components = c(1L, 2L), colors = NULL, seed = NULL)

PCA_plot(data, components = c(1L, 2L), colors = NULL, seed = NULL)
```

Arguments

`data`	A data frame containing source and mixture data in the standard fingerprinting format (raw or averaged).
`components`	A numeric vector of length 2 indicating which principal components to display. Default `c(1, 2)`.
`colors`	An optional character vector of colors, one per group. If `NULL`, ggplot2 defaults are used.
`seed`	An integer value used to initialize the random number generator.

Value

A ggplot object (also printed to the active graphics device).

`prefilter_tracers` *Prefilter tracers and clean raw or summary source datasets*

Description

Evaluates source samples and removes rows that exceed user-defined thresholds for missing isotopic ratios or missing/zero elemental contents. Absolute zeros in content columns are treated as analytically missing data and are actively converted to NAs to prevent downstream statistical bias. The mixture row (assumed to be the last row) is always preserved.

Usage

```
prefilter_tracers(data, max_na_ratio = 0.3, max_zero_content = 0.3)
```

Arguments

`data`	Data frame containing source and mixture samples. Supports raw and averaged formats.
`max_na_ratio`	Numeric. Maximum allowable proportion of missing ratio values per row (0 to 1). Default 0.30.
`max_zero_content`	Numeric. Maximum allowable proportion of missing or zero content values per row (0 to 1). Default 0.30.

Value

A cleaned data frame with failing source rows removed and zeros in content columns converted to NAs.

rangeTest — *Range test for tracer pre-selection*

Description

Removes tracer properties whose mixture value falls outside the envelope defined by the minimum and maximum values observed across all source groups. Tracers failing this test cannot be explained by any linear combination of the measured sources and should be excluded before unmixing.

Usage

```
rangeTest(data, verbose = TRUE)
```

Arguments

`data`	Data frame in the standard OceanTracer format (sources + mixture).
`verbose`	Logical. Whether to print removal summaries to the console. Default is `TRUE`.

Value

A data frame retaining only tracer columns that pass the range test.

Ternary_Plot — *Ternary diagrams with the CI_Method function results*

Description

Visualizes Conservativeness Index (CI) predictions using ternary diagrams. Incorporates advanced hierarchical clustering to automatically group and color-code structurally similar apportionment solutions.

Usage

```
Ternary_Plot(
  data,
  tracers = 1:4,
  n_row = 1,
  n_col = 2,
  Means = FALSE,
  cluster = FALSE,
  dendro_lvls = 5,
  vrtl_sol = FALSE,
  vrtl = c(0.05, 0.9, 0.05),
  seed = NULL
)
```

```
Ternary_diagram(
  data,
  tracers = 1:4,
  n_row = 1,
  n_col = 2,
  Means = FALSE,
  cluster = FALSE,
  dendro_lvls = 5,
  vrtl_sol = FALSE,
  vrtl = c(0.05, 0.9, 0.05),
  seed = NULL
)

ternary_diagram(
  data,
  tracers = 1:4,
  n_row = 1,
  n_col = 2,
  Means = FALSE,
  cluster = FALSE,
  dendro_lvls = 5,
  vrtl_sol = FALSE,
  vrtl = c(0.05, 0.9, 0.05),
  seed = NULL
)
```

Arguments

`data`	List. The output generated by the CI_Method function.
`tracers`	Numeric vector. Indices of the tracers to display. Default 1:4.
`n_row`	Integer. Number of rows in the output grid layout. Default 1.
`n_col`	Integer. Number of columns in the output grid layout. Default 2.
`Means`	Logical. If TRUE, indicates the input utilizes mean and SD formats rather than raw observations. Default FALSE.
`cluster`	Logical. If TRUE, applies hierarchical clustering to group structurally similar ternary solutions. Default FALSE.
`dendro_lvls`	Integer. Number of branch levels to cut the clustering dendrogram. Default 5.
`vrtl_sol`	Logical. If TRUE, overlays a theoretical or target virtual solution on the plot. Default FALSE.
`vrtl`	Numeric vector. Apportionment values for the theoretical solution. Default c(0.05, 0.9, 0.05).
`seed`	Integer. Random seed for reproducible cluster ordering when cluster = TRUE. Default NULL.

Value

Ternary diagrams from the individual tracer predictions.

---

tracer_pull_analysis *Analyze the Marginal Effect (Pull) of Individual Tracers*

Description

Evaluates the directional bias introduced by individual tracers within Consistent Tracer Selection (CTS) subset models. This function unpacks unconstrained source apportionments to reveal if specific tracers are systematically pulling source contributions into physically impossible spaces (e.g., >1 or <0).

Usage

```
tracer_pull_analysis(cts_result)
```

Arguments

`cts_result`	A data frame containing the output from a CTS run. It must contain an `id` column with space-separated tracer names, and weight columns named `w1`, `w2`, etc.

Details

Global unmixing models (using all tracers simultaneously) can suffer from "tracer compensation," where extreme mathematical contradictions between non-conservative tracers cancel each other out, resulting in a misleadingly plausible central solution.

This function exposes that illusion. By taking the output of a CTS run (e.g., `CTS_n_sources`), it isolates every debate a specific tracer participated in. It then calculates the marginal effect (median, mean, and SD) that the tracer has on the fractional weight of each source.

Visualizations are automatically generated and stored in the returned list:

- **boxplot:** Displays the distribution of calculated weights for each source, grouped by tracer. Red dashed lines denote the physical boundaries of 0 and 1.
- **heatmap:** Displays the median weight pull. Deep red indicates a strong positive pull (over-estimation), while deep blue indicates a negative pull (pushing the source into negative mass space).

Value

A named list containing four elements:

`summary` A `tibble` containing the Mean, Median, and SD of the weights forced by each tracer on each source.

`boxplot` A `ggplot` object showing the distribution of source weights grouped by tracer.

`heatmap` A `ggplot` object visualizing the median tracer bias across all sources.

`raw_expanded` A `tibble` of the raw, pivoted data used for calculations and plotting.

Examples

```
# Create a toy dataset mimicking a CTS model output
cts_out <- data.frame(
  id = c("TracerA TracerB", "TracerA TracerC", "TracerB TracerC"),
  w1 = c(0.3, 0.4, 0.2),
  w2 = c(0.7, 0.6, 0.8)
)

# Analyze tracer pull
pull_results <- tracer_pull_analysis(cts_out)

# View summary statistics and display boxplot
print(pull_results$summary)
print(pull_results$boxplot)
```

---

tracer_selection_diagnostic_plot

Tracer Selection Diagnostic plot

Description

Generates a generalized, faceted point plot to visualize tracer selection metrics across different datasets. Evaluates Classification Rate (CR), Concentration Indicator (CI), and Consistency Error (CONS) against user-defined thresholds.

Usage

```
tracer_selection_diagnostic_plot(
  data,
  cr_threshold = 70,
  ci_threshold = 2.5,
  cons_threshold = 0.025,
  custom_theme = ggplot2::theme_minimal()
)
```

Arguments

`data`	Data frame. Must contain the columns: `dataset`, `tracer`, `CR_score`, `CI_score`, and `consistency_error`. Optionally include a `tracer_group` column to facet the plot.
`cr_threshold`	Numeric. Minimum passing score for CR. Default 70.
`ci_threshold`	Numeric. Minimum passing score for CI. Default 2.50.
`cons_threshold`	Numeric. Maximum allowable consistency error. Default 0.025.
`custom_theme`	A ggplot2 theme object. Defaults to `ggplot2::theme_minimal()`.

Value

A `ggplot` object visualizing tracer metric statuses.

---

triangles_dataset — *Conservative Index evaluation using randomly sampled companion tracers*

Description

For each tracer of interest, solves a series of minimal determined linear systems to assess how consistently that tracer constrains the source apportionment. In each iteration the tracer under evaluation is kept fixed while the remaining tracers needed to complete the system are drawn at random from the other measured tracers in the dataset.

Usage

```
triangles_dataset(
  sources,
  mixtures,
  tracer = 0L,
  iter = 100L,
  seed = 69512L,
  pb = NULL,
  pb_offset = 0L,
  tracer_current = NULL,
  tracer_total = NULL,
  verbose = TRUE
)
```

Arguments

`sources`	Data frame of source statistics: ID, one mean column per tracer, one SD column per tracer, and sample size `n`.
`mixtures`	Data frame of mixture samples with the same tracer columns.
`tracer`	Integer (0-based). Index of the tracer under evaluation.
`iter`	Integer. Number of Monte Carlo iterations per mixture sample. Default 100L.
`seed`	Integer. Random seed for reproducibility. Default `69512L`.
`pb`	An optional `txtProgressBar` object created externally. Default `NULL`.
`pb_offset`	Integer. Starting offset for the external progress bar. Default `0L`.
`tracer_current`	Integer. Position of this tracer in the outer loop. Default `NULL`.
`tracer_total`	Integer. Total number of tracers in the outer loop. Default `NULL`.
`verbose`	Logical. Whether to print tracer progress messages to the console. Default is `TRUE`.

Details

This function is called internally by `CI_Method()` and is not usually invoked directly by the user.

Value

A data frame with `nrow(mixtures) * iter` rows containing mixture `ID`, `GOF`, and estimated source proportions (`w.*`).

See Also

`CI_Method`

---

triangles_virtual — *Conservative Index evaluation using virtually generated companion tracers*

Description

For each tracer of interest, solves a series of minimal determined linear systems to assess how consistently that tracer constrains the source apportionment. In each iteration the tracer under evaluation is kept fixed while the remaining tracers needed to complete the system are replaced by uniformly random values in $[0, 1]$ (virtual approach).

Usage

```
triangles_virtual(
  sources,
  mixtures,
  tracer = 0L,
  iter = 100L,
  seed = 69512L,
  pb = NULL,
  pb_offset = 0L,
  tracer_current = NULL,
  tracer_total = NULL,
  verbose = TRUE
)
```

Arguments

`sources`	Data frame of source statistics: ID, one mean column per tracer, one SD column per tracer, and sample size `n`.
`mixtures`	Data frame of mixture samples with the same tracer columns.
`tracer`	Integer (0-based). Index of the tracer under evaluation.
`iter`	Integer. Number of Monte Carlo iterations per mixture sample. Default `100L`.
`seed`	Integer. Random seed for reproducibility. Default `69512L`.

pb	External `txtProgressBar` object. Default `NULL`.
pb_offset	Integer. Starting offset for external progress bar. Default `0L`.
tracer_current	Integer. Position of this tracer in outer loop. Default `NULL`.
tracer_total	Integer. Total number of tracers in outer loop. Default `NULL`.
verbose	Logical. Whether to print tracer progress messages to the console. Default is `TRUE`.

Value

A data frame with `nrow(mixtures) * iter` rows containing mixture ID, GOF (fixed at 1), and apportioned weights per source.

`unmix_mixsiar` *Bayesian Source Apportionment using MixSIAR*

Description

A wrapper function that seamlessly formats an `OceanTracer` dataset (raw or averaged) and passes it to the `MixSIAR` Bayesian unmixing engine.

Usage

```
unmix_mixsiar(
  data,
  tracers = NULL,
  prior_type = "uninformative",
  prior_values = NULL,
  run = "normal",
  process_err = FALSE,
  resid_err = TRUE
)
```

Arguments

data	Data frame. Contains source and mixture data in standard OceanTracer format.
tracers	Character vector. Names of the tracers to include in the model. Default is `NULL` (auto-detected).
prior_type	Character. `"uninformative"` (default, alpha=1) or `"informative"`.
prior_values	Numeric vector. Required if `prior_type = "informative"`. Must match the number of sources.
run	Character. MCMC run length. Options: `"test"`, `"very short"`, `"short"`, `"normal"` (default), `"long"`, `"very long"`, `"extreme"`.
process_err	Logical. Include process error in the JAGS model? Default `FALSE`.
resid_err	Logical. Include residual error in the JAGS model? Default `TRUE`.

Value

A named list containing:

- `apportionment` A data frame with posterior draw proportions for each source, formatted for `plot_apportionment`.
- `bayesian_summary` A data frame summarizing MCMC parameter posteriors (mean, SD, quantiles, Rhat).
- `jags_model` The raw `rjags` output object returned by MixSIAR.

Examples

```
# Requires the external JAGS C++ library and MixSIAR R package installed
if (requireNamespace("MixSIAR", quietly = TRUE)) {

  # Sample dataset in OceanTracer raw format
  raw_data <- data.frame(
    ID = paste0("Sample_", 1:9),
    Group = rep(c("Source_A", "Source_B", "Mix_1"), each = 3),
    d13C = c(-24, -25, -24.5, -12, -11.5, -12.2, -18, -18.2, -17.9),
    d15N = c(5, 5.5, 5.2, 12, 11.8, 12.1, 8.5, 8.7, 8.4)
  )

  # Run MixSIAR test chain (1,000 iterations)
  try({
    res <- unmix_mixsiar(
      data = raw_data,
      tracers = c("d13C", "d15N"),
      run = "test"
    )

    # Plot posterior proportions
    plot_apportionment(res$apportionment, title = "MixSIAR Apportionment")
  }, silent = TRUE)
}
```

`unmixing` *Unmix sediment or marine particle mixtures*

Description

Estimates the fractional contribution of each potential source to one or more mixtures. Users can seamlessly switch between a lightning-fast Frequentist least-squares approach (with optional Linear Variability Propagation) or a standard Bayesian MCMC approach.

Usage

```
unmixing(
  data,
  model = "frequentist",
  iter = 1000L,
  variability = "SEM",
  lvp = TRUE,
  seed = 123456L,
  ...
)
```

Arguments

`data`	Data frame in the standard OceanTracer format.
`model`	Character. `"frequentist"` (default) or `"bayesian"`.
`iter`	Integer. Number of iterations for frequentist Monte Carlo/LVP. Default `1000L`.
`variability`	Character. Uncertainty metric: `"SEM"` (Standard Error of the Mean, default) or `"SD"` (Standard Deviation).
`lvp`	Logical. `TRUE` (default) uses Linear Variability Propagation; `FALSE` uses classical Monte Carlo resampling.
`seed`	Integer. Random seed for reproducibility. Default `123456L`.
`...`	Additional arguments passed to the Bayesian engine (e.g., `run = "test"`).

Details

Both raw (individual observations) and averaged (`Mean_`/`SD_` columns) input formats are accepted. When multiple mixture samples are present, the model processes each independently.

Value

For the `"frequentist"` model, a data frame with one row per iteration and mixture. For the `"bayesian"` model, a named list containing the formatted apportionment data frame, the MixSIAR summary statistics, and the raw JAGS model object.

References

Latorre, B., Lizaga, I., Gaspar, L., & Navas, A. (2025). Evaluating the impact of high source variability and extreme contributing sources on sediment fingerprinting models. *Water Resources Management*. doi:10.1007/s11269025041698

Index

∗ **datasets**
artificial_dataset, 3

aa_baseline_correction, 2
artificial_dataset, 3

box_plot *(Graph_boxplots)*, 20
boxPlot *(Graph_boxplots)*, 20

CB_Method *(Conservative_Balance)*, 6
cb_method *(Conservative_Balance)*, 6
CI_Method, 4, *34*
cluster_solutions, 5
Conservative_Balance, 6
consistency_error, 7
Correlation_Chart, 8
correlation_plot *(Correlation_Chart)*, 8
correlationPlot *(Correlation_Chart)*, 8
CR *(CR_Method)*, 9
CR_Method, 9
cr_ns *(CR_Method)*, 9
CTS_2_sources, 10
CTS_3_sources, 11
CTS_4_sources, 13
CTS_5_sources, 14
CTS_error *(consistency_error)*, 7
cts_error *(consistency_error)*, 7
CTS_n_sources, *5*, 15, *31*
CTS_seeds *(CTS_n_sources)*, 15
CTS_seeds_pairs *(CTS_3_sources)*, 11
CTS_seeds_quartets *(CTS_5_sources)*, 14
CTS_seeds_singles *(CTS_2_sources)*, 10
CTS_seeds_triplets *(CTS_4_sources)*, 13

DFATest, 16

frequentist_unconstrained, 16
frequentist_unconstrained_lvp, 17

ggbiplot, 19
ggplot, *20*, *27*, *31*

Graph_boxplots, 20

inputMixture *(inputSample)*, 21
inputSample, 21
inputSource, 22, *22*

KWTest, 23

lda, *25*
LDA_Chart, 24
LDA_plot *(LDA_Chart)*, 24
LDAPlot *(LDA_Chart)*, 24

pairs *(CTS_3_sources)*, 11
PCA_plot *(plot_pca)*, 27
PCAPlot *(plot_pca)*, 27
plot_apportionment, 25, *36*
plot_pca, 27
plot_results *(plot_apportionment)*, 25
plotResults *(plot_apportionment)*, 25
prcomp, *19*
prefilter_tracers, 28
princomp, *19*

quartets *(CTS_5_sources)*, 14

rangeTest, 29

singles *(CTS_2_sources)*, 10

Ternary_diagram *(Ternary_Plot)*, 29
ternary_diagram *(Ternary_Plot)*, 29
Ternary_Plot, 29
tibble, *31*
tracer_pull_analysis, 31
tracer_selection_diagnostic_plot, 32
triangles_dataset, 33
triangles_virtual, 34
triplets *(CTS_4_sources)*, 13

unmix_mixsiar, 35
unmixing, *17*, *18*, 36