Package: fingerPro (via r-universe)

October 26, 2026

Type Package

Title Unmixing Model Framework

Version 2.1

Encoding UTF-8

Date 2026-04-09

Description Quantifies the provenance of sediments by applying a mixing model algorithm to end sediment mixtures based on a comprehensive characterization of the sediment sources. The 'fingerPro' model builds upon the foundational concept of using mass balance linear equations for sediment source quantification by incorporating several distinct technical advancements. It employs an optimization approach to normalize discrepancies in tracer ranges and minimize the objective function. Latin hypercube sampling is used to explore all possible combinations of source contributions (0-100%), mitigating the risk of local minima. Uncertainty in source estimates is quantified through a Monte Carlo routine, and the model includes additional metrics, such as the normalized error of the virtual mixture, to detect mathematical inconsistencies, non-physical solutions, and biases. A new linear variability propagation (LVP) method is also included to address and quantify potential bias in model outcomes, particularly when dealing with dominant or non-contributing sources and high source variability, offering a significant advancement for field studies where direct comparison with theoretical apportionments is not feasible. In addition to the unmixing model, a complete framework for tracer selection is included. Several methods are implemented to evaluate tracer behaviour by considering both source and mixture information. These include the Consistent Tracer Selection (CTS) method to explore all tracer combinations and select the optimal ones improving the robustness and interpretability of the model results. A Conservative Balance (CB) method is also incorporated to enable the use of isotopic tracers. The package also provides several graphical tools to support data exploration and interpretation,

including box plots, correlation plots, Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA).

License GPL-2

URL https://github.com/eead-csic-eesa/fingerPro

Depends R (>= 3.5)

Imports Rcpp (>= 0.11.3), klaR (>= 0.6-12), ggplot2 (>= 2.2.1), GGally (>= 1.3.2), plyr (>= 1.8.4), MASS (>= 7.3-45), reshape (>= 0.8.7), grid (>= 3.1.1), gridExtra (>= 2.3), scales (>= 0.5.0), car (>= 3.0.0), RcppProgress (>= 0.4), Ternary (>= 1.2.2), dplyr (>= 1.0.7), crayon (>= 1.4.2), plotly (>= 4.10.3)

Suggests knitr, rmarkdown

LinkingTo Rcpp, RcppGSL, RcppProgress

RoxygenNote 7.3.3

VignetteBuilder knitr

NeedsCompilation yes

Config/pak/sysreqs cmake make libgsl0-dev libicu-dev libuv1-dev libssl-dev libx11-dev zlib1g-dev

Repository https://eead-csic-eesa.r-universe.dev

Date/Publication 2026-08-27 09:45:29 UTC

RemoteUrl https://github.com/eead-csic-eesa/fingerpro

RemoteRef HEAD

RemoteSha e1809b9d7639f6b2fa061b48a656b6ab074ed716

Contents

averaged_dataset . . . 3
box_plot . . . 3
CB_method . . . 4
CI . . . 5
correlation_plot . . . 6
CR . . . 7
CTS_explore . . . 8
CTS_select . . . 9
LDA_plot . . . 10
PCA_plot . . . 11
plot_results . . . 11
range_test . . . 12
raw_dataset . . . 12
read_database . . . 13
ternary_diagram . . . 15
unmix . . . 16
validate_results . . . 17
virtual_mixture . . . 18

Index **20**

averaged_dataset *Builds an averaged dataset from raw data*

Description

Generates an averaged dataset from individual (non-averaged) observations.

Usage

```
averaged_dataset(data, na.omit = T)
```

Arguments

data	A data frame containing raw source and mixture data.
na.omit	Boolean to omit or not NA values when computing the mean and SD

Value

A data frame representing the averaged dataset.

box_plot *Generate box-and-whisker plots for sediment tracers*

Description

This function creates a series of box and whisker plots arranged in a grid. It uses a paging system to prevent overlapping and ensures equal-sized plots.

Usage

```
box_plot(data, page = 1, n_row = 2, n_col = 3, colors = NULL)
```

Arguments

data	A data frame containing sediment source and mixture data.
page	Integer specifying which set of tracers to display (default = 1).
n_row	Number of rows per page (default = 3).
n_col	Number of columns per page (default = 2).
colors	Optional character vector of colors for the groups.

CB_method *Apply the Conservative Balance (CB) method for isotopic tracer analysis*

Description

This function transforms isotopic ratio and content data of individual tracers in a dataset into virtual elemental tracers, which can then be combined with classical tracers and analyzed with standard unmixing models.

Usage

```
CB_method(data)
```

Arguments

data	A data frame containing the isotopic tracer characteristics of sediment sources and mixtures. The data should be correctly formatted for isotopic analysis, including both isotopic ratio and isotopic content.

Details

The Conservative Balance (CB) method provides a novel, physically-based framework for analyzing isotopic tracers in sediment fingerprinting.

The core of the method is an exact transformation that combines the isotopic ratio and isotopic content into a virtual elemental tracer. This approach has two key advantages: it allows isotopic tracers to be analyzed using classical unmixing models, and it enables their combined use with elemental tracers to potentially increase the discriminant capacity of the fingerprinting analysis.

This function implements the simplified approximation of the CB transformation, assuming that the isotopic ratio is much smaller than 1. The calculation is performed for both averaged and non-averaged datasets.

A key feature of this transformation is that the tracer values for the mixture are set to zero. This is a direct consequence of the method, as the isotopic ratio of each source is subtracted from the mixture's isotopic ratio, meaning the mixture's own value minus itself results in zero.

Value

A data frame where isotopic tracers have been converted into scalar virtual tracers for further analysis. After the transformation, the mixture's row will have tracer values of zero.

References

Lizaga, I., Latorre, B., Gaspar, L., & Navas, A. (2022). Combined use of fingerprinting and tracing. Science of The Total Environment, 832, 154834.

`CI` *Compute the Conservativeness Index (CI) for individual tracers*

Description

This function calculates the Conservativeness Index (CI) for each tracer based on the results of an individual tracer analysis.

The CI index was adapted from its original definition to better describe the conservativeness of tracers in a high-dimensional space of multiple sources. The predicted source contributions from each tracer were first calculated and characterized by their centroid. Then, the CI index was calculated as the percentage of solutions with conservative apportionments (0 <= wi <= 1) relative to the centroid position. This new definition of the CI does not penalize tracers with dominant apportionments from one source and distributions close to a vertex of the physical space, unlike the previous definition.

Usage

```
CI(data, completion_method = "virtual", iter = 5000, rng_init = NULL)
```

Arguments

`data`	A data frame containing the characteristics of sediment sources and mixtures.
`completion_method`	A character string specifying the method for selecting the required remaining tracers to form a determined system of equations in the individual tracer analysis. Possible values are: "virtual": Fabricate remaining tracers virtually using generated random numbers. This method is valuable for an initial assessment of the tracer's consistency without the influence of other tracers from the dataset. "random": Randomly select remaining tracers from the dataset to complete the system. This method is useful for understanding how the tracer behaves when paired with others from the dataset.
`iter`	The number of iterations for the variability analysis in the individual tracer analysis. Increase 'iter' to improve the reliability and accuracy of the results. A sufficient number of iterations is reached when the output no longer changes significantly with further increases.
`rng_init`	An integer value used to initialize the random number generator (RNG). Providing a starting value ensures that the sequence of random numbers generated is reproducible. This is useful for debugging, testing, and comparing results across different runs. If no value is provided, a random one will be generated.

Value

A data frame containing the CI value for each tracer.

References

Lizaga, I., Latorre, B., Bodé, S., Gaspar, L., Boeckx, P., & Navas, A. (2024). Combining isotopic and elemental tracers for enhanced sediment source partitioning in complex catchments. *Journal of Hydrology*, 631, 130768. https://doi.org/10.1016/j.jhydrol.2024.130768

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

`correlation_plot` *Create a correlation matrix chart for tracer redundancy*

Description

The function displays a correlation matrix of each of the properties divided by the different sources to help the user in the decision.

Usage

```
correlation_plot(
  data,
  columns = c(1:ncol(data) - 1),
  mixtures = FALSE,
  nmixtures = 1,
  colors = NULL
)
```

Arguments

`data`	Data frame containing sediment source and mixture data.
`columns`	Numeric vector containing the index of the columns in the chart (the first column refers to the grouping variable)
`mixtures`	Boolean to include or exclude the mixture samples in the chart
`nmixtures`	Number of mixtures in the dataset
`colors`	Vector of colors to use for the scatterplot

CR *Rank tracers using the Consensus Ranking (CR) method*

Description

This function computes the Consensus Ranking (CR) method, an ensemble technique to identify non-conservative and dissenting tracers in sediment fingerprinting studies. The method combines predictions from single-tracer models and is based on a scoring function derived from a series of random "debates" between tracers.

Usage

```
CR(data, debates = 1000, rng_init = NULL)
```

Arguments

`data`	A data frame containing sediment source and mixture data.
`debates`	An integer specifying the target number of debates each tracer should participate in. The function will run until each tracer has participated in at least this many debates.
`rng_init`	An integer value used to initialize the random number generator (RNG). Providing a starting value ensures that the sequence of random numbers generated is reproducible. This is useful for debugging, testing, and comparing results across different runs. If no value is provided, a random one will be generated.

Details

The Consensus Ranking method is based on a series of random debates to test the compatibility of tracers. In each debate, a random subset of tracers is selected. The size of this subset is determined by the number of sources, corresponding to the minimum number of equations needed to overdetermine the unmixing model.

For each debate, a least-squares method is used to find a solution to the overdetermined mass balance equations. The consensus of the debate is measured by the mathematical compatibility of the tracers, specifically using the Root Mean Square Error (RMSE) of the mass balance equations. The tracer whose exclusion from the debate results in lowest RMSE is identified as the "dissenting" tracer for that round.

This process is repeated for a specified number of debates. Each tracer accumulates a count of total participations and a count of lost debates (being identified as dissenting). The final CR score is a quantitative measure of consensus, calculated as ‘100 - (lost debates / total debates) * 100‘.

A low CR score indicates that a tracer frequently disrupts the consensus and is considered a non-conservative or dissenting tracer. Conversely, a high CR score suggests the tracer is in frequent agreement with the others, making it a reliable and conservative tracer for the unmixing model. This method is robust and does not require pre-screening or filtering of tracers.

Value

A data frame containing the CR score for each tracer. The score, ranging from 100 to 0, indicates the tracer's rank in terms of consensus and conservativeness. Tracers are ordered by their score in descending order, with the most conservative tracers having high scores and dissenting tracers having low scores.

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_explore *Identify minimal tracer combinations with high discriminant power*

Description

This function generates a list of all possible minimal tracer combinations and serves as a crucial initial step (a "seed") in building a consistent tracer selection within a sediment fingerprinting study. This analysis systematically explores various minimal tracer combinations and solves the resulting determined systems of equations to assess the variability and reliability of each combination. The dispersion of the solution directly reflects the discriminant capacity of each tracer combination, where a lower dispersion indicates a higher capacity to distinguish between sources. Furthermore, by evaluating solutions in an unconstrained manner, the function assesses the conservativeness of the tracers; it identifies whether they remain within a physically plausible range or if they exhibit non-conservative behavior. While traditional methods like Discriminant Function Analysis (DFA) also identify discriminant tracer combinations, this function provides solutions that are not restricted to the physically feasible space ($0 < wi < 1$). This unconstrained approach is valuable for identifying problematic tracer selections that might otherwise be masked when using constrained unmixing models, as discussed by Latorre et al. (2021).

Usage

```
CTS_explore(data, iter = 1000, rng_init = NULL)
```

Arguments

`data`	Data frame containing sediment source and mixtures.
`iter`	The number of iterations for the variability analysis. Increase 'iter' to improve the reliability and accuracy of the results. A sufficient number of iterations is reached when the output no longer changes significantly with further increases.
`rng_init`	An integer value used to initialize the random number generator (RNG). Providing a starting value ensures that the sequence of random numbers generated is reproducible. This is useful for debugging, testing, and comparing results across different runs. If no value is provided, a random one will be generated.

Details

The Consistent Tracer Selection (CTS) method, as described by Latorre et al. (2021), begins by considering all possible sets of $n-1$ tracers, where n is the number of sources. Each of these sets forms a determined system of linear equations that can be solved. To account for the variability within the sources, each tracer set is iteratively solved. This process involves sampling the source average values from a t-distribution, reflecting the discrepancy between the true mean and the measured mean due to finite observations. The maximum dispersion observed in the average apportionments for each tracer set is then used as a criterion to rank them, with lower dispersion indicating higher discriminant capacity. This initial step is crucial for identifying multiple discriminant solutions within the dataset, a problem often unexplored by traditional tracer selection methods.

Value

The function returns a data frame summarizing all possible tracer combinations. The data frame includes the following columns for a scenario with three sources: ‘tracers‘, ‘w1‘, ‘w2‘, ‘w3‘, ‘percent_physical‘, ‘sd_w1‘, ‘sd_w2‘, ‘sd_w3‘, and ‘max_sd_wi‘. Each row represents a tracer combination, detailing its corresponding solution (w_i), the percentage of solutions that are physically feasible (0 < w_i < 1), the standard deviation of the results (sd_w_i), and the maximum dispersion among all sources (max_sd_w_i). The solutions are sorted in descending order, with the solution having the lowest dispersion appearing first. This highlights the most discriminant and conservative combinations.

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.

`CTS_select` *Extend minimal tracer sets by evaluating mathematical consistency*

Description

This function extends a minimal tracer combination obtained from the ‘CTS_explore‘ function ensuring its mathematical consistency in order to select optimum tracers to perform the unmix.

Usage

```
CTS_select(data, tracers_seeds, seed_id, error_threshold = 0.05)
```

Arguments

`data`	A data frame containing the characteristics of sediment sources and mixtures.
`tracers_seeds`	A data frame containing the output from the ‘CTS_explore‘ function.
`seed_id`	A numeric ID to select a specific row from ‘tracers_seeds‘.

`error_threshold`
: A numeric value (e.g., 0.05). Only tracers with a normalized error below this value will be retained.

Details

The function calculates a normalized error for each tracer to assess the consistency of a given apportionment solution. The method involves first computing a "virtual mixture" by using the proposed apportionment values to perform a weighted average of the source tracer concentrations. The error for each tracer is then the difference between the tracer concentration in the real mixture and the virtual mixture. This error is normalized by the range of the tracer, which is estimated from the extremes of the sources' confidence intervals.

A low normalized error for all tracers (i.e., less than a predefined threshold like 0.05) indicates a mathematically consistent tracer selection. If most tracers show low errors while a few have high errors, it suggests that those tracers may be non-conservative or less influential on the model's result. Conversely, high normalized errors in most tracers indicate mathematical inconsistency and can point to the existence of multiple partial solutions in the dataset.

Value

A data frame containing the normalized error for each 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.

LDA_plot *Perform and visualize Linear Discriminant Analysis (LDA)*

Description

The function performs a linear discriminant analysis and displays the data in the relevant dimensions.

Usage

```
LDA_plot(data, text = TRUE, colors = NULL)
```

Arguments

`data`	Data frame containing source and mixtures data
`text`	Boolean to show or not the identification number of each sample point in the plot
`colors`	Allows choosing between a different set of colors in the plots

PCA_plot *Perform and visualize Principal Component Analysis (PCA)*

Description

The function performs a principal components analysis on the given data matrix and displays a biplot using vqv.ggbiplot package of the results for each different source to help the user in the decision.

Usage

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

Arguments

data	Data frame containing source and mixtures data
components	Numeric vector containing the index of the two principal components in the chart
colors	Vector of colors to use for the groups in the plot

plot_results *Visualize the results of a sediment unmixing analysis*

Description

This function generates a plot showing the relative contribution of sediment sources to each mixture. The output of the unmix function should be used as input for this function.

Usage

```
plot_results(
  data,
  violin = T,
  bounds = c(0, 1),
  scaled = T,
  y_high = 1,
  colors = NULL,
  ncol = 1
)
```

Arguments

data	A data frame, typically the output from the unmix function, containing the relative contributions of sediment sources.
violin	A logical value. If TRUE, violin charts are used instead of density plots.
bounds	A numeric vector of length 2 specifying the lower and upper bounds for the data.
scaled	A logical value. If TRUE, the density plots are scaled.
y_high	The maximum value for the y-axis.
colors	A character vector of colors to use for the plots.
ncol	The number of plots per row.

---

range_test *Verifies if target sediment concentrations fall within the range of potential source values.*

Description

Function that excludes the properties of the sediment mixture/s outside the minimum and maximum values in the sediment sources.

Usage

```
range_test(data)
```

Arguments

data	Data frame containing source and mixtures

Value

Data frame containing sediment sources and mixtures

---

raw_dataset *Build a raw dataset from averaged data*

Description

Generates a raw (non-averaged) dataset by sampling individual observations from the mean and standard deviation values provided in an averaged input data frame. For each source, it generates 'n' observations for each tracer by sampling from a normal distribution using the provided mean and standard deviation. Mixture data is appended directly without sampling.

Usage

```
raw_dataset(data)
```

Arguments

data	A data frame containing averaged source and mixture data. It is expected to have columns for tracer means (prefixed with "mean_"), standard deviations (prefixed with "sd_"), and a column "n" indicating the number of observations for each source.

Value

A data frame representing the raw, non-averaged dataset, with each row corresponding to an individual observation.

---

read_database *Read a sediment unmixing database*

Description

This function automatically infers the type of sediment database ("raw", "averaged", or "isotopic") based on its column names and verifies its integrity. It validates column names and their order to ensure data is correctly structured for subsequent package functions.

To retain conservative tracers for subsequent analyses, it is recommended to perform a minimal dataset cleaning beforehand:

- Replace BDL (below detection limit) entries with a small positive number.
- Exclude tracers whose mixture value is BDL or zero.
- Optionally, remove tracers with predominantly BDL values.

Database 'raw' format: This database contains individual measurements for scalar tracers. It must have the following columns in order:

- **ID**: Unique identifier for each sample.
- **samples**: A categorical column identifying each source and mixture. The unique value representing the mixture must appear last. In cases with multiple mixture samples, they must all share the same mixture name but will be distinguished by unique entries in the **ID** column.
- **tracer1, tracer2, ...**: Columns for each tracer measurement.

Database 'isotopic raw' format: This database contains individual measurements for isotopic tracers, which require both ratio and content data. It must have the following columns in order:

- **ID**: Unique identifier for each sample.
- **samples**: A categorical column identifying each source and mixture. The unique value representing the mixture must appear last. In cases with multiple mixture samples, they must all share the same mixture name but will be distinguished by unique entries in the **ID** column.
- **ratio1, ratio2, ...**: Columns with the isotopic ratio values for each tracer.
- **cont_ratio1, cont_ratio2, ...**: Columns with the corresponding content (concentration) values for each tracer.

Database 'averaged' format: This database contains statistical summaries of the scalar tracer data. It must have the following columns in order:

- **ID**: Unique identifier for each sample.
- **samples**: A categorical column identifying each source and mixture. The unique value representing the mixture must appear last. In cases with multiple mixture samples, they must all share the same mixture name but will be distinguished by unique entries in the **ID** column.
- **mean_tracer1, mean_tracer2, ...**: Columns with the mean value for each tracer.
- **sd_tracer1, sd_tracer2, ...**: Columns with the standard deviation for each tracer.
- **n**: The number of measurements used to calculate the mean and standard deviation.

Database 'isotopic averaged' format: This database contains statistical summaries for isotopic tracers. It must have the following columns in order:

- **ID**: Unique identifier for each sample.
- **samples**: A categorical column identifying each source and mixture. The unique value representing the mixture must appear last. In cases with multiple mixture samples, they must all share the same mixture name but will be distinguished by unique entries in the **ID** column.
- **mean_ratio1, mean_ratio2, ...**: Columns with the mean isotopic ratio values.
- **mean_cont_ratio1, mean_cont_ratio2, ...**: Columns with the mean isotopic content values.
- **sd_ratio1, sd_ratio2, ...**: Columns with the standard deviation of the isotopic ratio values.
- **sd_cont_ratio1, sd_cont_ratio2, ...**: Columns with the standard deviation of the isotopic content values.
- **n**: The number of measurements.

Usage

```
read_database(file, mixture = 1)
```

Arguments

`file`	Character string. The name of the CSV file or the path to it.
`mixture`	Integer. The index of the mixture sample to keep if multiple are present. Defaults to 1.

Value

A data frame representing the sediment unmixing database

`ternary_diagram` *Visualize tracer distributions using ternary diagrams*

Description

This function creates ternary diagrams to visualize the results of the individual tracer analysis. Each ternary diagram represents the predicted apportionments for a specific tracer.

Usage

```
ternary_diagram(
  data,
  page = 1,
  rows = 2,
  cols = 3,
  solution = NA,
  completion_method = "virtual",
  iter = 5000,
  rng_init = NULL
)
```

Arguments

`data`	A data frame containing the characteristics of sediment sources and mixtures.
`page`	Integer specifying which set of tracers to display (default = 1).
`rows`	An integer specifying the number of rows in the grid.
`cols`	An integer specifying the number of columns in the grid.
`solution`	A vector containing an optional reference solution.
`completion_method`	A character string specifying the method for selecting the required remaining tracers to form a determined system of equations in the individual tracer analysis. Possible values are: "virtual": Fabricate remaining tracers virtually using generated random numbers. This method is valuable for an initial assessment of the tracer's consistency without the influence of other tracers from the dataset. "random": Randomly select remaining tracers from the dataset to complete the system. This method is useful for understanding how the tracer behaves when paired with others from the dataset.
`iter`	The number of iterations for the variability analysis in the individual tracer analysis. Increase 'iter' to improve the reliability and accuracy of the results. A sufficient number of iterations is reached when the output no longer changes significantly with further increases.
`rng_init`	An integer value used to initialize the random number generator (RNG). Providing a starting value ensures that the sequence of random numbers generated is reproducible. This is useful for debugging, testing, and comparing results across different runs. If no value is provided, a random one will be generated.

Value

A grid of ternary diagrams, each representing the predicted apportionments for a specific tracer. If there are three sources, the function generates one ternary triangle for each tracer. If there are four sources, the function generates six triangles for each tracer. The six triangles represent the following source combinations at their vertices: 1. (S1, S2, S3+S4) 2. (S2, S3, S1+S4) 3. (S3, S4, S1+S2) 4. (S4, S1, S2+S3) 5. (S1, S3, S2+S4) 6. (S2, S4, S1+S3)

`unmix` *Perform sediment source apportionment (unmixing)*

Description

This function assesses the relative contribution of potential sediment sources to each sediment mixture in a dataset using a mass balance approach. It supports both unconstrained and constrained optimization, allowing for different methods of handling source variability.

Usage

```
unmix(
  data,
  iter = 1000L,
  variability = "SEM",
  lvp = TRUE,
  constrained = FALSE,
  resolution = NA,
  rng_init = 123456L
)
```

Arguments

`data`	Data frame containing sediment source and mixture data.
`iter`	The number of iterations for the variability analysis. Increase ‘iter‘ to improve the reliability and accuracy of the results. A sufficient number of iterations is reached when the output no longer changes significantly with further increases.
`variability`	A character string specifying the type of variability to calculate. Possible values are "SD" for Standard Deviation or "SEM" for Standard Error of the Mean.
`lvp`	A logical value to switch between classical variability analysis (lvp = FALSE) and Linear Variability Propagation (lvp = TRUE). LVP is a more accurate method for calculating uncertainty in unmixing models under high variability and extreme source apportionments.
`constrained`	A logical value indicating whether the optimization should be constrained to physical solutions. If constrained = TRUE, the optimization will be restricted to solutions where all source contributions are within the range of 0 to 1. If constrained = FALSE, the optimization is unconstrained.

`resolution` An integer specifying the number of samples used in each hypercube dimension for constrained optimization. This parameter is only used when constrained = TRUE and is required to perform the analysis.

`rng_init` An integer value used to initialize the random number generator (RNG). Providing a starting value ensures that the sequence of random numbers generated is reproducible. This is useful for debugging, testing, and comparing results across different runs. If no value is provided, a random one will be generated.

Value

A data frame containing the relative contributions of the sediment sources to each sediment mixture, across all iterations. The second and third rows of the result correspond to the solution for the central or mean value of the sources. The output includes an ID column to identify each mixture, a GOF (Goodness of Fit) column, and columns for each source showing their calculated contributions.

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*, *1-15*. https://doi.org/10.1007/s11269-025-04169-8

`validate_results` *Evaluate the mathematical consistency of a tracer selection*

Description

This function assesses the mathematical consistency of a tracer selection for an apportionment result by computing the normalized error between the predicted and observed tracer concentrations in the virtual mixture. A low normalized error for all tracers indicates a consistent tracer selection. This function can be used to diagnose problems in the results of fingerprinting models.

Usage

```
validate_results(selected_data, apportionments, error_threshold = 0.05)
```

Arguments

`selected_data` A data frame containing the characteristics of sediment sources and mixtures for the specific tracer selection to be evaluated.

`apportionments` A numeric vector containing the apportionment values (contributions) to be evaluated for each source, in the same order as they appear in the data.

`error_threshold`
A numeric value (e.g., 0.05) representing the maximum acceptable normalized error. This value is used as a benchmark to categorize tracers as consistent or inconsistent in the diagnostic messages.

Details

The function calculates a normalized error for each tracer to assess the consistency of a given apportionment solution. The method involves first computing a "virtual mixture" by using the proposed apportionment values to perform a weighted average of the source tracer concentrations. The error for each tracer is then the difference between the tracer concentration in the real mixture and the virtual mixture. This error is normalized by the range of the tracer, which is estimated from the extremes of the sources' confidence intervals.

A low normalized error for all tracers (i.e., less than a predefined threshold like 0.05) indicates a mathematically consistent tracer selection. If most tracers show low errors while a few have high errors, it suggests that those tracers may be non-conservative or less influential on the model's result. Conversely, high normalized errors in most tracers indicate mathematical inconsistency and can point to the existence of multiple partial solutions in the dataset.

Value

A data frame containing the normalized error for each 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.

`virtual_mixture` *Create a synthetic sediment mixture for validation*

Description

This function generates a virtual sediment mixture based on the characteristics of existing sediment sources and a set of user-defined apportionment weights. It effectively simulates a mixture with known source contributions.

Usage

```
virtual_mixture(data, weights)
```

Arguments

`data`	A data frame containing the characteristics of the sediment sources.
`weights`	A numeric vector representing the proportional contributions (apportionment values) of each source to the virtual mixture. The order of weights in the vector must correspond to the order of sources in the ‘data‘ frame. The sum of ‘weights‘ should ideally equal 1.

Details

A virtual mixture is a hypothetical sediment sample created by mathematically combining the tracer characteristics of known sources according to specified proportions (‘weights‘). This is a powerful tool in sediment fingerprinting for:

- **Consistency Checks**: Comparing observed mixture data against a virtual mixture can help assess the consistency of a dataset or the validity of an unmixing solution.
- **Scenario Testing**: Simulating mixtures under different hypothetical source contributions to understand how changes might affect sediment composition.
- **Model Validation**: Generating known virtual mixtures to test the accuracy and performance of unmixing models.

The function calculates the tracer values for the virtual mixture by taking the weighted average of the corresponding tracer values from each source.

Value

A data frame representing the virtual mixture. This data frame will have the same structure as a single row for a mixture in your input ‘data‘, but with tracer values calculated based on the provided ‘weights‘.

Index

averaged_dataset, 3

box_plot, 3

CB_method, 4
CI, 5
correlation_plot, 6
CR, 7
CTS_explore, 8
CTS_select, 9

LDA_plot, 10

PCA_plot, 11
plot_results, 11

range_test, 12
raw_dataset, 12
read_database, 13

ternary_diagram, 15

unmix, 16

validate_results, 17
virtual_mixture, 18