Package: FluvialTracer (via r-universe)

November 12, 2026

Type Package

Title A Comprehensive Package for Sediment Source Unmixing

Version 1.0.0

Date 2026-10-26

Description Source apportionment toolkit for fluvial catchments and river sediment provenance. It estimates how much each source contributes to a sediment mixture, using frequentist, Bayesian, and machine-learning models on the same dataset. The software includes tools to check which tracer is causing an unrealistic result, plus built-in validation against artificial mixtures with known, true source proportions.

License GPL (>= 2)

Imports car, cluster, crayon, dplyr, GGally, ggplot2, ggtext, grDevices, gridExtra, klaR, magrittr, MASS, MixSIAR, plotly, plyr, randomForest, reshape2, rjags, scales, stats, Ternary, tibble, tidyr, tidyselect, utils, xgboost

Suggests patchwork, rgl, stringr

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Config/roxygen2/version 8.1.1

RoxygenNote 7.3.3

NeedsCompilation no

Author Margarita Villalonga [aut, cre], Esther Cepeda [aut], Maarten Wynants [aut], Ivan Lizaga [aut, ths, ldr], Samuel Bodé [ctb], Pascal Boeckx [ctb], William Blake [ctb], Jan Vanaverbeke [ctb], Ulrike Braeckman [ctb], Aiming Lian [ctb], Borja Latorre [ctb], Yuri Jacques Agra Bezerra da Silva [ctb], Leticia Gaspar [ctb], Jorge Crespo [ctb], Ana Navas [ctb], Maziar Mohammadi [ctb], Vince Q Vu [ctb]

Maintainer Margarita Villalonga <mvillalonga@ipe.csic.es>

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

Date/Publication 2026-11-12 12:57:20 UTC

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

RemoteRef HEAD

RemoteSha fa7871ade8e1523a82b8310e23d71fd965672403

Contents

checking_diff_mult_mix . . . 3
checking_diff_single_mix . . . 4
CI_Method . . . 5
cluster_solutions . . . 6
Conservative_Balance . . . 7
consistency_error . . . 8
Correlation_Chart . . . 9
CR_Method . . . 10
CTS_2_sources . . . 11
CTS_3_sources . . . 12
CTS_4_sources . . . 13
CTS_5_sources . . . 13
CTS_n_sources . . . 14
DFATest . . . 15
fluvial_palette . . . 15
fluvialdata_example . . . 16
frequentist_unconstrained . . . 16
frequentist_unconstrained_lvp . . . 17
ggbiplot . . . 18
Graph_boxplots . . . 20
inputSample . . . 21
inputSource . . . 22
KWTest . . . 22
LDA_Chart . . . 23
plot_apportionment . . . 24
plot_pca . . . 26
rangeTest . . . 26
Ternary_Plot . . . 27
tracer_pull_analysis . . . 28
triangles_dataset . . . 29
triangles_virtual . . . 31
unmix_mixsiar . . . 32
unmix_rf . . . 33
unmix_xgboost . . . 34
unmixing . . . 34

Index **36**

checking_diff_mult_mix *Checking the difference between virtual and artificial mixture (for data sets containing multiple mixtures)*

Description

A function that calculates and visualizes the percentage deviation between an experimental artificial mixture and a theoretical virtual mixture calculated from user-defined source proportions. Source proportions should be defined as described in Option A: manual list or Option B: CSV file

Usage

```
checking_diff_mult_mix(
  data,
  proportions = props_table,
  sample_col = 1,
  source_col = 2
)
```

Arguments

`data`	Data frame. Contains source and mixture data in the standard format.
`proportions`	Data.frame. With one row per mixture and one column per source. Can be created in two ways: Option A - manually: `proportions = data.frame(rbind(c(0.5, 0.25, 0.25, 0), c(0.3, 0.3, 0.2, 0.2)))` Option B - from a CSV file (one row per mixture, one column per source): `proportions = read.csv()`
`sample_col`	Character. Defines which column of the data contains the sample names.
`source_col`	Character. Defines which column of the data contains the source names. Indicate the column number in which the sources are defined independently from the sample number.

Value

No return value, called for side effects (draws one plot per mixture showing the deviation between the virtual and artificial mixture).

checking_diff_single_mix *Checking the difference between virtual and artificial mixture (for data sets containing only one mixture)*

Description

A function that calculates and visualizes the percentage deviation between an experimental artificial mixture and a theoretical virtual mixture calculated from user-defined source proportions.

Usage

```
checking_diff_single_mix(
  data,
  proportions = c(1, 0, 0, 0),
  sample_col = 1,
  source_col = 2,
  subtitle_extra = NULL
)
```

Arguments

data	Data frame. Contains source and mixture data in the standard format.
proportions	Numeric vector. Defines the proportions of each source in the mixture. Required to calculate virtual mixture.
sample_col	Character. Defines which column of the data contains the sample names.
source_col	Character. Defines which column of the data contains the source names. Indicate the column number in which the sources are defined independently from the sample number.
subtitle_extra	Character. Custom subtitle for the plot; overrides the automatic one if provided. Default NULL.

Value

Invisibly returns a gtable object with the combined plot (both panels, title, legend and caption). The plot is also drawn on the current graphics device.

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. For each tracer of interest, the function solves a large number of minimal determined systems and records the spread of the resulting source contributions. A narrow spread indicates that the tracer is conservative and highly informative, regardless of which other tracers accompany it.

Usage

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

Arguments

data	Data frame in the standard FluvialTracer format (sources + mixture). Raw and averaged formats are both accepted.
points	Integer. Number of Monte Carlo draws per tracer.
Means	Logical or NULL (default). When NULL, the format is detected automatically from column names. Set explicitly to TRUE or FALSE to override.
triangles	Logical. If TRUE (default), both virtual and dataset strategies are computed.
seed	Integer. Random seed for reproducibility.

Details

Two strategies are available for selecting the accompanying tracers:

- **Virtual method**: remaining tracers are replaced by values drawn uniformly from $[0, 1]$, isolating the tracer of interest from correlations with other measured properties.
- **Dataset method**: remaining tracers are sampled at random from the measured dataset, reflecting realistic geochemical combinations.

Value

A list with one data frame per tracer containing the predicted source contributions across all iterations, followed by two named elements: DB, the input data, and CI, a data frame with the CI score, normalized score, category and status 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.doi: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. doi:10.1016/j.scitotenv.2020.137537

`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).

The physically feasible region (0-100 shaded white; everything outside it (negative or >100 i.e. not physically possible) is shaded grey.

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.

References

Wynants, M., Verboom, C., Lizaga, I., de Boer, H., van der Perk, M., Bennett, W., Boeckx, P., & Brooks, A. Sediment and phosphorus source hysteresis and tracing during an extreme flood in a subtropical Australian river.

`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 other methods.

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 FluvialTracer 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. If NULL (default), colors are pulled automatically from fluvial_palette: the first N-1 Source colors, plus one Mixture color if mixtures = TRUE.
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 = 1,
  apportionment_threshold = 0,
  seed = 123456
)

CR(
  data,
  debates = 1000,
  completion_quantile = 1,
  apportionment_threshold = 0,
  seed = 123456
)

cr_ns(
  data,
  debates = 1000,
  completion_quantile = 1,
  apportionment_threshold = 0,
  seed = 123456
)
```

Arguments

data	Data frame. Contains source and mixture data in the standard FluvialTracer format.
debates	Integer. Target number of random debates per tracer. Default 1000.
completion_quantile	Numeric. Value between 0-1. Quantile threshold for debate completion. For example, 0.5 stops the simulation once half the tracers hit the debates target. Default 1.
apportionment_threshold	Numeric. Tolerance boundary for valid debate apportionments (e.g., defining valid as -threshold to 1 + threshold). Default 0.
seed	Integer. Random seed for reproducibility. Default 123456.

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)

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

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

Arguments

source	Data frame of sediment sources.
mixture	Data frame of mixture samples.
iter	Number of Monte Carlo iterations.
seed	Integer for reproducibility.

Value

A data frame with one row per tracer, sorted by `max_dw` (most discriminant first), with columns: `id` (tracer name), `w1`, `w2` (source contributions from mean values), `dw1`, `dw2` (dispersion of each contribution, half the 16-84% quantile range), `cons` (proportion of physically feasible solutions) and `max_dw` (maximum dispersion).

`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)

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

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

Arguments

`source`	Data frame of sediment sources.
`mixture`	Data frame of mixture samples.
`iter`	Number of Monte Carlo iterations.
`seed`	Integer for reproducibility.

Value

A data frame with one row per tracer pair, sorted by `max_dw` (most discriminant first), with columns: `id` (tracer pair), `w1`, `w2`, `w3` (source contributions from mean values), `dw1`, `dw2`, `dw3` (dispersion of each contribution, half the 16-84% quantile range), `cons` (proportion of physically feasible solutions) and `max_dw` (maximum dispersion).

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)

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

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

Arguments

source	Data frame of sediment sources.
mixture	Data frame of mixture samples.
iter	Number of Monte Carlo iterations.
seed	Integer for reproducibility.

Value

A data frame with one row per tracer triplet, sorted by max_dw (most discriminant first), with columns: id (tracer triplet), w1 to w4 (source contributions from mean values), dw1 to dw4 (dispersion of each contribution, half the 16-84% quantile range), cons (proportion of physically feasible solutions) and max_dw (maximum dispersion).

CTS_5_sources *CTS engine for 5 sources*

Description

Solves determined systems for 4 tracers and 5 sources to assess solution dispersion and discriminant capacity.

Usage

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

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

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

Arguments

source	Data frame of sediment sources.
mixture	Data frame of mixture samples.
iter	Number of Monte Carlo iterations.
seed	Integer for reproducibility.

Value

A data frame with one row per tracer quartet, sorted by max_dw (most discriminant first), with columns: id (tracer quartet), w1 to w5 (source contributions from mean values), dw1 to dw5 (dispersion of each contribution, half the 16-84% quantile range), cons (proportion of physically feasible solutions) and max_dw (maximum dispersion).

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 one row per tracer combination, sorted by max_dw (lowest dispersion first), with columns: id (tracer combination), w1, w2, ... (source apportionments), dw1, dw2, ... (solution dispersion), % in sources range (percentage of physically feasible solutions) and max_dw (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)
```

Arguments

`data`	Data frame in the standard FluvialTracer format (sources + mixture).
`niveau`	Numeric. Significance level for the F-test at each step. Default 0.1..

Value

A list with two elements: (1) a data frame retaining only the tracer columns selected by the stepwise procedure, and (2) the object returned by `klaR::greedy.wilks` with the details of the selection.

fluvial_palette *Default color palette for FluvialTracer plots*

Description

A named vector of hex color codes used as the default palette across FluvialTracer's plotting functions (e.g. `plot_apportionment`, `LDA_Chart`). Designed to remain visually distinct for up to five sources plus the mixture.

Usage

```
fluvial_palette
```

Format

A named character vector of length 20.

Source

Custom palette defined by the FluvialTracer authors.

fluvialdata_example *Example dataset with 3 sources and one mixture*

Description

Virtual dataset in the standard FluvialTracer format, used in the package examples.

Usage

```
fluvialdata_example
```

Format

A data frame with one row per sample: sample ID, source/mixture group, and one column per tracer.

Examples

```
head(fluvialdata_example)
```

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
)
```

Arguments

data	Data frame in the standard FluvialTracer format (sources + mixture).
variability	Character. "SEM" for Standard Error of the Mean, "SD" for Standard Deviation.
iter	Integer. Number of Monte Carlo iterations.
means	Logical. TRUE when the input is in averaged format (Mean_/SD_ columns); FALSE for raw observations.
seed	Integer. Random seed for reproducibility.

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
)
```

Arguments

`data`	Data frame in the standard FluvialTracer format (sources + mixture).
`variability`	Character. `"SEM"` for Standard Error of the Mean, `"SD"` for Standard Deviation.
`iter`	Integer. Number of iterations.
`means`	Logical. `TRUE` when the input is in averaged format (`Mean_`/`SD_` columns); `FALSE` for raw observations.
`seed`	Integer. Random seed for reproducibility.

Details

LVP is recommended when source variability is high or when apportionments 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

Biplot for Principal Components using ggplot2

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(), PCA(), or lda().
choices	Numeric vector. Indices of the 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 data ellipse for each group. Default FALSE.
ellipse.prob	Numeric. Size of the ellipse in Normal probability. 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. Alpha transparency for points (0 = transparent, 1 = opaque). Default 1.
var.axes	Logical. If TRUE, renders variable loading arrows. Default TRUE.
circle	Logical. If TRUE, draws a correlation circle (only valid if scale = TRUE and var.scale = 1). Default FALSE.
circle.prob	Numeric. Size of the 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 further from arrow). Default 1.5.
varname.abbrev	Logical. If TRUE, abbreviates variable names. Default FALSE.
...	Currently not used. Kept for compatibility with the original ggbiplot function.

Value

A ggplot2 plot.

`Graph_boxplots` *Box and whiskers plot*

Description

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

Usage

```
Graph_boxplots(
  data,
  tracers = 1:(ncol(data) - 2),
  ncol = 3,
  colors = NULL,
  seed = NULL,
  max_per_page = 16,
  label_size = 3
)

boxPlot(
  data,
  tracers = 1:(ncol(data) - 2),
  ncol = 3,
  colors = NULL,
  seed = NULL,
  max_per_page = 16,
  label_size = 3
)

box_plot(
  data,
  tracers = 1:(ncol(data) - 2),
  ncol = 3,
  colors = NULL,
  seed = NULL,
  max_per_page = 16,
  label_size = 3
)
```

Arguments

`data`	Data frame. Contains source and mixture data in the standard FluvialTracer format.
`tracers`	Numeric vector. Indices of the tracers to plot (column 1 refers to the first tracer variable). Default 1:(ncol(data) - 2).

ncol	Integer. Number of plots per row in the grid layout. Default 3.
colors	Character vector. Optional hex codes or names for custom boxplot colors, one per group (sources + mixture). If NULL (default), colors are pulled automatically from fluvial_palette: the first N-1 Source colors, plus one Mixture color for the last group.
seed	Integer. Random seed for reproducible jittering or sampling. Default NULL.
max_per_page	Integer. Maximum number of tracer panels to draw per page/device call. If the number of requested tracers exceeds this, the plots are split across multiple pages, drawn sequentially (each replacing the previous on-screen, or as successive pages if called inside a pdf()/dev device). Default 16.
label_size	Numeric. Font size for the n= and mean annotation labels inside each panel. Lower this if labels overlap when many panels are shown. Default 3.

Value

Invisibly returns a list of gtable objects, one per page. The plots are also drawn on the current graphics device.

---

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 FluvialTracer 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 FluvialTracer convention.

See Also

inputSource for extracting source rows.

inputSource *Input sediment sources*

Description

Selects and extracts the source samples from the dataset, computing mean, SD and n when the input is in raw format. When a different format dataset is detected, column names are normalised to `Mean_`/`SD_` in the output so that all downstream functions work correctly.

Usage

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

Arguments

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

Value

A data frame with one row per source, containing the source identifier (`id`), the mean of each tracer (`Mean_` columns), the standard deviation of each tracer (`SD_` columns) and the number of samples (`n`).

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)
```

Arguments

`data`	Data frame in the standard FluvialTracer format (sources + mixture).
`pvalue`	Numeric. Significance threshold. Default 0.05.

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
)

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

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

Arguments

data	Data frame. Contains source and mixture data in the standard FluvialTracer format.
Plot3D	Logical. If TRUE, renders a 3D scatter plot instead of a 2D projection. Default FALSE.
text	Logical. If TRUE, overlays the sample identification numbers on the plot. Default FALSE.

colors	Character vector. Custom colors for the source groups. If NULL (default), the first N Source colors from fluvial_palette are used automatically, where N is the number of source groups detected in data.
interactive	Logical. If TRUE, renders an interactive HTML widget for 3D plots. Default FALSE.
seed	Integer. Random seed for reproducibility. Default NULL.

Value

Invisibly returns a list with: lda (the fitted MASS::lda object), confusion (leave-one-out confusion matrix), prop_var (percentage of variance explained by each discriminant axis) and plot (the plot object).

---

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 the 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 `FALSE`.
bounds	A numeric vector of length 2 specifying the lower and upper display bounds (the camera window). Default `c(0.0, 1.0)`.
scaled	Logical. If `TRUE`, density curves are area-scaled to a common maximum. Only used when `violin = FALSE`. Default `TRUE`.
y_high	Maximum value for the y-axis. Only used when `violin = FALSE` and `scaled = FALSE`. Default `1`.
colors	An optional named character vector of colors, one per source. If `NULL` (default), the first N colors of `fluvial_palette`'s Source series are used automatically, where N is the number of sources detected in `data`.
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)`).
title	Optional character string to add a main title to the plot.
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 `ggplot2` object.

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` (default), colors are pulled automatically from fluvial_palette: the first N-1 Source colors, plus one Mixture color for the last group.
`seed`	An integer value used to initialize the random number generator.

Value

A ggplot object.

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)
```

Arguments

data	Data frame in the standard FluvialTracer format (sources + mixture).

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:(length(data) - 2),
  n_row = 4,
  n_col = 5,
  Means = FALSE,
  cluster = FALSE,
  dendro_lvls = 5,
  vrtl_sol = FALSE,
  vrtl = c(1, 0, 0),
  seed = NULL
)

Ternary_diagram(
  data,
  tracers = 1:(length(data) - 2),
  n_row = 4,
  n_col = 5,
  Means = FALSE,
  cluster = FALSE,
  dendro_lvls = 5,
  vrtl_sol = FALSE,
  vrtl = c(1, 0, 0),
  seed = NULL
)

ternary_diagram(
  data,
  tracers = 1:(length(data) - 2),
```

```
  n_row = 4,
  n_col = 5,
  Means = FALSE,
  cluster = FALSE,
  dendro_lvls = 5,
  vrtl_sol = FALSE,
  vrtl = c(1, 0, 0),
  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

No return value, called for side effects (draws ternary diagrams of the individual tracer predictions on the current graphics device).

---

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 the 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 to aid interpretation:

- **Boxplots:** Display 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 (overestimation), 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.

`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 (dataset approach). Because the companion tracers are real geochemical measurements, this strategy reflects realistic co-variation patterns among tracers and produces a less conservative — but more ecologically meaningful — estimate of tracer informativeness compared to the virtual approach.

Usage

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

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.
seed	Integer. Random seed for reproducibility. Default 69512.
pb	An optional txtProgressBar object created externally. When provided, the function updates this shared bar instead of creating its own, allowing CI_Method() to display a single continuous progress bar across all tracers. 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, used to print a progress message. Default NULL.
tracer_total	Integer. Total number of tracers in the outer loop. Default NULL.

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. Columns are ID (mixture index), GOF (set to 1 for all rows), and one weight column per source (w.S1, w.S2, ...).

See Also

CI_Method, triangles_virtual

`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). The mixture value for each virtual tracer is drawn inside the range spanned by the virtual source values, ensuring the system is geometrically solvable. Because the companion tracers carry no real geochemical information, any consistency in the resulting source weights is attributable solely to the tracer under evaluation, making this a conservative (worst-case) estimator of tracer informativeness.

Usage

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

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.
`seed`	Integer. Random seed for reproducibility. Default `69512`.
`pb`	An optional `txtProgressBar` object created externally. When provided, the function updates this shared bar instead of creating its own, allowing `CI_Method()` to display a single continuous progress bar across all tracers. 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, used to print a progress message. Default `NULL`.
`tracer_total`	Integer. Total number of tracers in the outer loop. Default `NULL`.

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. Columns are `ID` (mixture index), `GOF` (set to 1 for all rows), and one weight column per source (`w.S1`, `w.S2`, ...).

See Also

`CI_Method`, `triangles_dataset`

`unmix_mixsiar` *Bayesian source apportionment using MixSIAR*

Description

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

Usage

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

Arguments

`data`	Data frame. Contains source and mixture data in the standard FluvialTracer format.
`tracers`	Character vector. Names of the tracers to include in the model.
`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 list with three elements:

`apportionment` Data frame with the posterior draws of the global source proportions, in the FluvialTracer format (`ID`, `GOF` and one column per source), compatible with `plot_apportionment`.

`bayesian_summary` Data frame with the summary statistics of all model parameters returned by JAGS.

`jags_model` The full JAGS model object returned by `MixSIAR::run_model`.

`unmix_rf` *Machine Learning source apportionment using Random Forest (with expanded bounds)*

Description

Trains a Random Forest model on simulated virtual mixtures. Features a 'stretch_bounds' parameter to expand training data beyond the 0-1 limits, effectively eliminating the boundary bias typically seen in pure Machine Learning and Bayesian unmixing of extreme sources.

Usage

```
unmix_rf(
  data,
  n_sim = 10000L,
  n_trees = 500L,
  stretch_bounds = 1.2,
  seed = 123456L
)
```

Arguments

`data`	Data frame. Contains source and mixture data in the standard FluvialTracer format.
`n_sim`	Integer. Number of virtual mixtures to simulate for training. Default 10000.
`n_trees`	Integer. Number of trees for the Random Forest model. Default 500.
`stretch_bounds`	Numeric. Factor to expand training boundaries to avoid 0/1 bias. Default 1.2.
`seed`	Integer. Random seed for reproducibility. Default 123456.

Value

A list containing the apportionment data frame (one row per tree) and a summary.

unmix_xgboost — *Machine Learning source apportionment using XGBoost (Gradient Boosting)*

Description

Trains an ensemble of XGBoost models on simulated virtual mixtures. Uses bootstrapping to generate uncertainty distributions (density curves).

Usage

```
unmix_xgboost(
  data,
  n_sim = 5000L,
  n_boot = 100L,
  stretch_bounds = 1.2,
  seed = 123456L
)
```

Arguments

data	Data frame. Contains source and mixture data.
n_sim	Integer. Number of virtual mixtures for the base training set. Default 5000.
n_boot	Integer. Number of XGBoost models to train (iterations for uncertainty). Default 100.
stretch_bounds	Numeric. Expansion factor for training boundaries. Default 1.2.
seed	Integer. Random seed. Default 123456..

Value

Invisibly returns a list with two elements: apportionment, a data frame with one row per bootstrapped model (ID, GOF and one column per source), compatible with plot_apportionment; and xgb_summary, a data frame with the mean, SD, median and 95% interval of each source contribution.

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), a standard Bayesian MCMC approach, or a Machine Learning approach.

Usage

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

Arguments

`data`	Data frame in the standard FluvialTracer format.
`model`	Character. `"frequentist"` (default), `"bayesian"` or `"machine_learning"`.
`iter`	Integer. Number of iterations for frequentist Monte Carlo/LVP. Default `1000`.
`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 `123456`.
`ml_type`	Character. Only used if `model = "machine_learning"`. Options are `"RandomForest"` (default) or `"XGBoost"`.
`...`	Additional arguments passed to the Bayesian or ML engines (e.g., `run = "test"`, `n_trees = 500`, `stretch_bounds = 1.2`).

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. For the `"machine_learning"` model, a named list containing the apportionment predictions and a statistical summary (`rf_summary` or `xgb_summary`).

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**
    fluvial_palette, 15
    fluvialdata_example, 16

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

CB_Method *(*Conservative_Balance*)*, 7
cb_method *(*Conservative_Balance*)*, 7
checking_diff_mult_mix, 3
checking_diff_single_mix, 4
CI_Method, 5, *30*, *32*
cluster_solutions, 6
Conservative_Balance, 7
consistency_error, 8
Correlation_Chart, 9
correlation_plot *(*Correlation_Chart*)*, 9
correlationPlot *(*Correlation_Chart*)*, 9
CR *(*CR_Method*)*, 10
CR_Method, 10
cr_ns *(*CR_Method*)*, 10
CTS_2_sources, 11
CTS_3_sources, 12
CTS_4_sources, 13
CTS_5_sources, 13
CTS_error *(*consistency_error*)*, 8
cts_error *(*consistency_error*)*, 8
CTS_n_sources, *6*, 14, *29*
CTS_seeds *(*CTS_n_sources*)*, 14
CTS_seeds_pairs *(*CTS_3_sources*)*, 12
CTS_seeds_quartets *(*CTS_5_sources*)*, 13
CTS_seeds_singles *(*CTS_2_sources*)*, 11
CTS_seeds_triplets *(*CTS_4_sources*)*, 13

DFATest, 15

fluvial_palette, 15
fluvialdata_example, 16
frequentist_unconstrained, 16
frequentist_unconstrained_lvp, 17

ggbiplot, 18
Graph_boxplots, 20

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

KWTest, 22

LDA_Chart, *15*, 23
LDA_plot *(*LDA_Chart*)*, 23
LDAPlot *(*LDA_Chart*)*, 23

pairs *(*CTS_3_sources*)*, 12
PCA_plot *(*plot_pca*)*, 26
PCAPlot *(*plot_pca*)*, 26
plot_apportionment, *15*, 24, *33*, *34*
plot_pca, 26
plot_results *(*plot_apportionment*)*, 24
plotResults *(*plot_apportionment*)*, 24

quartets *(*CTS_5_sources*)*, 13

rangeTest, 26

singles *(*CTS_2_sources*)*, 11

Ternary_diagram *(*Ternary_Plot*)*, 27
ternary_diagram *(*Ternary_Plot*)*, 27
Ternary_Plot, 27
tracer_pull_analysis, 28
triangles_dataset, 29, *32*
triangles_virtual, *30*, 31
triplets *(*CTS_4_sources*)*, 13

unmix_mixsiar, 32
unmix_rf, 33
unmix_xgboost, 34
unmixing, *17*, *18*, 34