Package ‘InterpolateR’

May 2, 2025

Title A Comprehensive Toolkit for Fast and Efficient Spatial Interpolation

Version 1.3-4

Maintainer Jonnathan Landi <jonnathan.landi@outlook.com>

Description Spatial interpolation toolkit designed for environmental and geospatial applications. It includes a range of methods, from traditional techniques to advanced machine learning approaches, ensuring accurate and efficient estimation of values in unobserved locations.

License GPL (>= 3)

Depends R (>= 4.4.0)

Imports data.table, future.apply, future, stats, terra, pbapply, randomForest, qmap

URL https://github.com/Jonnathan-Landi/InterpolateR

BugReports https://github.com/Jonnathan-Landi/InterpolateR/issues

Encoding UTF-8

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, testthat (>= 3.0.0), covr, withr, ncdf4

Config/testthat/edition 3

VignetteBuilder knitr

LazyData true

NeedsCompilation no

Author Jonnathan Landi [aut, cre, cph] (ORCID: <https://orcid.org/0009-0003-3162-6647>)

Repository CRAN

Date/Publication 2025-05-02 18:50:06 UTC

Contents

BD_Coord *Precipitation Station Coordinates Dataset*

Description

This dataset contains the coordinates (in UTM format) of several precipitation stations. Each station is uniquely identified by the `Cod` column, which corresponds to the station identifiers used in the `BD_Obs` dataset. The coordinates of each station are provided in two columns: - `X` for the Easting (longitude), - `Y` for the Northing (latitude).

Usage

```
data("BD_Coord")
```

Format

A ‘data.table‘ or ‘data.frame‘ object with station coordinates. The dataset includes the following columns:

`Cod` The unique identifier for each station. This should correspond to the station columns in the `BD_Obs` dataset.

`X` The Easting (X-coordinate) of the station in UTM format (numeric).

`Y` The Northing (Y-coordinate) of the station in UTM format (numeric).

Details

The data represents the geographic coordinates of precipitation stations used in the analysis. The first column, `Cod`, contains the unique identifiers of the stations, which should match the column names in the `BD_Obs` dataset. The subsequent columns, `X`, `Y`, contain the UTM coordinates for each station, representing the station's location on the Earth's surface and `Z`, contain the altitude in meters of each station.

Source

The data was generated for use in the bias correction model for satellite products, RFplus.

Examples

```
data(BD_Coord)
## You can use str(BD_Coord) to get a description of the structure
## or view some of the first rows using head(BD_Coord)
```

BD_Obs *Observed Data from Ground Stations*

Description

A dataset containing daily precipitation observations from multiple ground stations.

Usage

```
data(BD_Obs)
```

Format

A `data.table` or `data.frame` with *n* rows (dates) and *m+1* columns (stations + Date):

- **Date**: A `Date` object representing the observation date.
- **STxxx**: Numeric values representing observed precipitation measurements for each station, where `STxxx` is the unique station identifier.

Details

The dataset follows the structure:

```
  > BD_Obs
# A data.table or data.frame with n rows (dates) and m+1 columns (stations + Date)
    Date        ST001  ST002  ST003  ST004  ...
    <date>      <dbl>  <dbl>  <dbl>  <dbl>  ...
  1 2015-01-01    0      0      0      0    ...
  2 2015-01-02    0      0      0     0.2   ...
  3 2015-01-03   0.1     0      0     0.1   ...
```

- Each station column contains numeric values representing observed measurements. - The column names (station identifiers) must be unique and match those in `BD_Coord$Cod` to ensure proper spatial referencing.

See Also

`BD_Coord` for station metadata.

Examples

```
# Load dataset
data(BD_Obs)

# Check structure
str(BD_Obs)

# Display first rows
head(BD_Obs)
```

create_data

Create a Unified Observation Dataset in the BD_Obs Format from Multiple CSV Files This function constructs a unified dataset (`BD_Obs` structure) by merging multiple CSV files, each containing in-situ observations from different stations. The function standardizes the format required by downstream interpolation or bias correction algorithms by aligning all station data into a single `data.table`, with dates as rows and station identifiers as columns.

Description

Each input CSV file must contain exactly two columns: the first with dates (`Date`) and the second with the in-situ measurements of the variable to be interpolated.

Usage

```
create_data(file.path, Start_date, End_Date, ncores = NULL, max.na = NULL)
```

Arguments

`file.path`	`character`. Path to the folder containing the CSV files. Each file should represent a single station and be named using the station ID (e.g., `M001.csv`). Each file must have exactly two columns: a date column and a column of in-situ observations for the variable to be interpolated.
`Start_date`	`Date`. Start date of the period to be included in the merged dataset. If the CSV files cover different date ranges, this defines the initial bound of the common time window for merging.
`End_Date`	`Date`. End date of the period to be included in the merged dataset. This sets the upper bound of the time window to consider across all files.
`ncores`	`integer`. Number of processing cores to be used when reading and merging CSV files in parallel. If If you want to perform the procedure without parallelization, set `ncores = NULL`. The default is `NULL`.
`max.na`	`numeric`, optional. Maximum acceptable percentage of missing values per station (from 0 to 100). Stations exceeding this threshold will be excluded. If `NULL`, no filtering is performed. Default is `NULL`.

Value

If `max.na` is `NULL`, the function returns a `data.table` structured in the `BD_Obs` format, where the first column contains the dates and the remaining columns correspond to individual stations. This format preserves the full dataset without filtering for missing values.

If `max.na` is not `NULL`, the function returns a named list containing:

`data` A `data.table` in the `BD_Obs` format that includes only stations with a percentage of missing values less than or equal to `max.na`.

`Na_stations` A `data.table` summarizing the percentage of missing values for each station, useful for assessing data quality and supporting decisions about station selection.

Author(s)

Jonnathan Augusto landi Bermeo, jonnathan.landi@outlook.com

Examples

```
# Example usage
file.path <- system.file("extdata/Folds_ejs_create_data", package = "InterpolateR")

# Create a data with all stations
data <- create_data(file.path, Start_date = "2015-01-01", End_Date = "2015-03-01", ncores = NULL)
```

`Cressman` *Cressman Objective Analysis Method*

Description

The Cressman objective analysis computes values at grid points Z_{ij}^a (where i and j are the grid point indices for a 2D grid) as the weighted average of the difference between observed values Z_k^o and background values interpolated to the observation locations Z_k^b (i.e., $Z_k^o - Z_k^b$, called the observation increment) plus the background value at the grid point Z_{ij}^b.

Usage

```
Cressman(
  BD_Obs,
  BD_Coord,
  shapefile,
  grid_resolution,
  search_radius,
  training = 1,
  stat_validation = NULL,
  Rain_threshold = NULL,
  save_model = FALSE
)
```

Arguments

`BD_Obs` A `data.table` or `data.frame` containing observational data with the following structure:

- The first column (`Date`): A `Date` object representing the observation date.
- The remaining columns: Each column corresponds to a unique ground station, where the column name is the station identifier.

The dataset should be structured as follows:

```
> BD_Obs
# A data.table or data.frame with n rows (dates) and m+1 columns (stations + Date)
   Date        ST001  ST002  ST003  ST004  ...
   <date>      <dbl>  <dbl>  <dbl>  <dbl>  ...
1  2015-01-01    0      0      0      0    ...
2  2015-01-02    0      0      0     0.2   ...
3  2015-01-03   0.1     0      0     0.1   ...
```

- Each station column contains numeric values representing observed measurements.
- The column names (station identifiers) must be unique and match those in `BD_Coord$Cod` to ensure proper spatial referencing.

`BD_Coord` A `data.table` or `data.frame` containing the metadata of the ground stations. It must include the following columns:

- `"Cod"`: Unique identifier for each ground station.
- `"X"`: Latitude of the station in UTM format.
- `"Y"`: Longitude of the station in UTM format.

`shapefile` A shapefile defining the study area, used to constrain the interpolation to the region of interest. The shapefile must be of class `SpatVector` (from the `terra` package) and should have a UTM coordinate reference system.

`grid_resolution`
A numeric value indicating the resolution of the interpolation grid in kilometers (`km`).

`search_radius` A numeric vector indicating the search radius in kilometers (`km`) for the Cressman method. **Note:** See the "Notes" section for additional details on how to search radius values.

`training` Numerical value between 0 and 1 indicating the proportion of data used for model training. The remaining data are used for validation. Note that if you enter, for example, 0.8 it means that 80 % of the data will be used for training and 20 % for validation. If you do not want to perform validation, set training = 1. (Default training = 1).

`stat_validation`
A character vector specifying the names of the stations to be used for validation. This option should only be filled in when it is desired to manually enter the stations used for validation. If this parameter is NULL, and the formation is different from 1, a validation will be performed using random stations. The vector must contain the names of the stations selected by the user for validation. For example, stat_validation = c("ST001", "ST002"). (Default stat_validation = NULL).

`Rain_threshold` List of numerical vectors defining precipitation thresholds to classify precipitation into different categories according to its intensity. This parameter should be entered only when the validation is to include categorical metrics such as Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), etc. Each list item should represent a category, with the category name as the list item name and a numeric vector specifying the lower and upper bounds of that category. **Note:** See the "Notes" section for additional details on how to define categories, use this parameter for validation, and example configurations.

`save_model`	Logical value indicating whether the interpolation file should be saved to disk. The default value is `FALSE`. indicating that the interpolated file should not be saved. If set to `TRUE`, be sure to set the working directory beforehand using `setwd(path)` to specify where the files should be saved.

Value

The return value depends on whether validation has been performed.

- **Without validation:** The function returns a `list`, where each element is a `SpatRaster` object containing the interpolated values for a specific search radius defined in `search_radius`. The number of elements in this list matches the length of `search_radius`.
- **With validation:** The function returns a named `list` with two elements:
  - `Ensamble`: A `list` where each element corresponds to a `SpatRaster` object containing the interpolated values for a specific search radius in `search_radius`.
  - `Validation`: A `list` where each element is a `data.table` containing the validation results for the corresponding interpolated `SpatRaster`. Each `data.table` incluye métricas de bondad de ajuste como RMSE, MAE y Kling-Gupta Efficiency (KGE), junto con métricas categóricas si se proporciona `Rain_threshold`.

The number of elements in both the `Ensamble` and `Validation` lists matches the length of `search_radius`, ensuring that each interpolation result has an associated validation dataset.

Details

The Cressman method is defined by the following equation:

$$Z_{ij}^a = Z_{ij}^b + \frac{\sum_{k=1}^{n} w_k (Z_k^o - Z_k^b)}{\sum_{k=1}^{n} w_k}$$

where:

Z_{ij}^a is the analysis value at grid point i, j.

Z_{ij}^b is the background value at grid point i, j.

Z_k^o is the observed value at station k.

Z_k^b is the background value interpolated to station k.

w_k is the weight assigned to station k.

n is the total number of stations used.

The weight w_k is a function of the distance $r = \sqrt{(x_{ij} - x_k)^2 + (y_{ij} - y_k)^2}$ between the individual observation k and grid point (i, j). R is the influence radius. Beyond the influence radius, the weight is set to zero. R is therefore often referred to as the cut-off radius.

Note

The `search_radius` parameter defines the influence range for the Cressman interpolation method. It determines the maximum distance (in kilometers) within which observational data points contribute to the interpolated value at a given location. A larger radius results in smoother interpolated fields but may oversmooth local variations, while a smaller radius preserves finer details but may introduce noise.

The Cressman method typically applies an iterative approach, where the search radius is progressively reduced to refine the interpolation. Each iteration recalculates interpolated values with a smaller radius, allowing a better representation of small-scale features in the dataset.

The `search_radius` should be defined as a numeric vector representing the influence range in kilometers (`km`) for each interpolation iteration. For example, setting `search_radius = c(50, 20, 10)` means the first iteration considers a 50 km influence radius, the second iteration uses 20 km, and the final iteration refines the interpolation with a 10 km radius. The number of elements in `search_radius` determines the total number of iterations.

The `Rain_threshold` parameter is used to calculate categorical metrics such as the Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), success ratio (SR), Hit BIAS (HB),Heidke Skill Score (HSS); Hanssen-Kuipers Discriminant (HK); Equal Threat Score (ETS) or Gilbert Skill Score. The parameter should be entered as a named list, where each item represents a category and the name of the item is the category name. The elements of each category must be a numeric vector with two values: the lower and upper limits of the category. For example: `Rain_threshold = list( no_rain = c(0, 1), light_rain = c(1, 5), moderate_rain = c(5, 20), heavy_rain = c(20, 40), violent_rain = c(40, Inf) )`

Precipitation values will be classified into these categories based on their intensity. Users can define as many categories as necessary, or just two (e.g., "rain" vs. "no rain"). It is important that these categories are entered according to the study region, as each study region may have its own categories.

Author(s)

Jonnathan Landi jonnathan.landi@outlook.com

References

Cressman, G. P., 1959: An operational objective analysis system. Mon. Wea. Rev., 87, 367-374, doi:10.1175/1520-0493(1959)087%3C0367:AOOAS%3E2.0.CO;2.

Examples

```
# Load data from on-site observations
 data("BD_Obs", package = "InterpolateR")
 data("BD_Coord", package = "InterpolateR")

# Load the study area where the interpolation is performed.
shp_path = system.file("extdata", "study_area.shp", package = "InterpolateR")
shapefile = terra::vect(shp_path)
 # Perform the interpolation
Interpolated_Cressman <- Cressman(BD_Obs, BD_Coord, shapefile, grid_resolution = 5,
                                  search_radius = 10, training = 1,
                                  stat_validation = "M001", Rain_threshold = NULL,
                                 save_model = FALSE)
# Results ("Ensamble with 10 km radius")
Radius_10 = Interpolated_Cressman$Ensamble$`10000`
```

```
# Validation statistics
# Validation results with a 10 km radius
Validation_results_10 = Interpolated_Cressman$Validation$`10000`
```

`IDW` *Inverse distance weighted interpolation (IDW)*

Description

This function performs Inverse Distance Weighting (IDW), which is a spatial interpolation method that estimates unknown values using the known values of surrounding points, assigning greater influence to closer points.

Usage

```
IDW(
  BD_Obs,
  BD_Coord,
  shapefile,
  grid_resolution,
  p = 2,
  n_round = NULL,
  training = 1,
  stat_validation = NULL,
  Rain_threshold = NULL,
  save_model = FALSE,
  name_save = NULL
)
```

Arguments

`BD_Obs` A `data.table` or `data.frame` containing observational data with the following structure:

- The first column (`Date`): A `Date` object representing the observation date.
- The remaining columns: Each column corresponds to a unique ground station, where the column name is the station identifier.

The dataset should be structured as follows:

```
> BD_Obs
# A data.table or data.frame with n rows (dates) and m+1 columns (stations + Date)
   Date        ST001  ST002  ST003  ST004  ...
   <date>      <dbl>  <dbl>  <dbl>  <dbl>  ...
1  2015-01-01    0      0      0      0    ...
2  2015-01-02    0      0      0     0.2   ...
3  2015-01-03   0.1     0      0     0.1   ...
```

	• Each station column contains numeric values representing observed measurements. • The column names (station identifiers) must be unique and match those in `BD_Coord$Cod` to ensure proper spatial referencing.
`BD_Coord`	A `data.table` or `data.frame` containing the metadata of the ground stations. It must include the following columns: • `"Cod"`: Unique identifier for each ground station. • `"X"`: Latitude of the station in UTM format. • `"Y"`: Longitude of the station in UTM format.
`shapefile`	A shapefile defining the study area, used to constrain the interpolation to the region of interest. The shapefile must be of class `SpatVector` (from the `terra` package) and should have a UTM coordinate reference system.
`grid_resolution`	A numeric value indicating the resolution of the interpolation grid in kilometers (`km`).
`p`	'Numeric' value that controls how the influence decreases with distance. The default value is 2
`n_round`	An integer specifying the number of decimal places to round the interpolated results. If set to `NULL`, all decimal places will be preserved. The default value is `1`.
`training`	Numerical value between 0 and 1 indicating the proportion of data used for model training. The remaining data are used for validation. Note that if you enter, for example, 0.8 it means that 80 % of the data will be used for training and 20 % for validation. If you do not want to perform validation, set training = 1. (Default training = 1).
`stat_validation`	A character vector specifying the names of the stations to be used for validation. This option should only be filled in when it is desired to manually enter the stations used for validation. If this parameter is NULL, and the formation is different from 1, a validation will be performed using random stations. The vector must contain the names of the stations selected by the user for validation. For example, stat_validation = c("ST001", "ST002"). (Default stat_validation = NULL).
`Rain_threshold`	List of numerical vectors defining precipitation thresholds to classify precipitation into different categories according to its intensity. This parameter should be entered only when the validation is to include categorical metrics such as Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), etc. Each list item should represent a category, with the category name as the list item name and a numeric vector specifying the lower and upper bounds of that category. **Note:** See the "Notes" section for additional details on how to define categories, use this parameter for validation, and example configurations.
`save_model`	Logical value indicating whether the interpolation file should be saved to disk. The default value is `FALSE`. indicating that the interpolated file should not be saved. If set to `TRUE`, be sure to set the working directory beforehand using `setwd(path)` to specify where the files should be saved.

`name_save`	Character string indicating the name under which the interpolation raster file will be saved. By default the algorithm sets as output name: 'Model_IDW'.

Value

The return value will depend on whether validation has been performed or not. If validation is not performed, the function will return a `SpatRaster` object with the interpolated values. If validation is performed, the function will return a list with two elements:

- `Ensamble`: A `SpatRaster` object with the interpolated values.
- `Validation`: A `data.table` with the validation results, including goodness-of-fit metrics and categorical metrics (if `Rain_threshold` is provided).

Details

Inverse distance weighting (IDW) works as a deterministic mathematical interpolator that assumes that values closer to an unknown point are more closely related to it than those farther away. When using this method, sample points are weighted during the interpolation process so that the influence of a known point on the unknown point decreases as the distance between them increases. The IDW method calculates the value at an unknown point by a weighted average of the values of the known points, where the weights are inversely proportional to the distances between the prediction point and the known points. The basic formula defined by Shepard states that:

$$\hat{Z}(s_0) = \frac{\sum_{i=1}^{N} w_i Z(s_i)}{\sum_{i=1}^{N} w_i}$$

where:

$\hat{Z}(s_0)$ is the estimated value at the unknown point.

$Z(s_i)$ are the known values.

w_i are the weights assigned to each known point.

N is the total number of known points used.

The weights are calculated by:

$$w_i = \frac{1}{d(s_0, s_i)^p}$$

where:

$d(s_0, s_i)$ is the distance between the unknown point s_0 and the known point s_i.

p is the power parameter that controls how the influence decreases with distance.

The `Rain_threshold` parameter is used to calculate categorical metrics such as the Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), success ratio (SR), Hit BIAS (HB),Heidke Skill Score (HSS); Hanssen-Kuipers Discriminant (HK); Equal Threat Score (ETS) or Gilbert Skill Score. The parameter should be entered as a named list, where each item represents a category and the name of the item is the category name. The elements of each category must be a numeric vector with two values: the lower and upper limits of the category. For example: `Rain_threshold = list( no_rain = c(0, 1), light_rain = c(1, 5), moderate_rain = c(5, 20), heavy_rain = c(20, 40), violent_rain = c(40, Inf) )`

Precipitation values will be classified into these categories based on their intensity. Users can define as many categories as necessary, or just two (e.g., "rain" vs. "no rain"). It is important that these categories are entered according to the study region, as each study region may have its own categories.

Author(s)

Jonnathan Landi jonnathan.landi@outlook.com

References

Shepard, D. (1968) A two-dimensional interpolation function for irregularly-spaced data. Proceedings of the 1968 ACM National Conference, 1968, pages 517–524. DOI: 10.1145/800186.810616

Examples

```
# Load data from on-site observations
data("BD_Obs", package = "InterpolateR")
data("BD_Coord", package = "InterpolateR")

# Load the study area where the interpolation is performed.
shp_path = system.file("extdata", "study_area.shp", package = "InterpolateR")
shapefile = terra::vect(shp_path)

# Perform the interpolation
Interpolated_data <- IDW(BD_Obs, BD_Coord, shapefile,
  grid_resolution = 5, p = 2,
  n_round = 1, training = 0.8, Rain_threshold = NULL,
  stat_validation = NULL, save_model = FALSE, name_save = NULL)
```

`RFmerge` *Merging of satellite datasets with ground observations using Random Forest*

Description

RFmerge is a methodology developed by Baez-Villanueva et al. (2020) for the fusion of satellite precipitation datasets with ground-based observations, with the objective of improving the accuracy and spatial representativeness of the data. This package implements RFmerge using Random Forest as a machine learning technique to correct biases and adjust the distribution of satellite products to in situ measurements. In addition, it allows the integration of multiple sources of information, including geographic and environmental variables, optimizing the interpolation and spatial extrapolation of precipitation in data-limited regions. Unlike previous implementations, this package has been optimized to improve computational efficiency and reduce processing times by incorporating advanced data manipulation techniques with `data.table`.

Usage

```
RFmerge(
  BD_Obs,
  BD_Coord,
  cov,
  mask = NULL,
  n_round = NULL,
  ntree = 2000,
  seed = 123,
  training = 1,
  stat_validation = NULL,
  Rain_threshold = NULL,
  save_model = FALSE,
  name_save = NULL
)
```

Arguments

`BD_Obs`
: A `data.table` or `data.frame` containing observational data with the following structure:

- The first column (`Date`): A `Date` object representing the observation date.
- The remaining columns: Each column corresponds to a unique ground station, where the column name is the station identifier.

The dataset should be structured as follows:

```
> BD_Obs
# A data.table or data.frame with n rows (dates) and m+1 columns (stations + Date)
   Date        ST001  ST002  ST003  ST004  ...
   <date>      <dbl>  <dbl>  <dbl>  <dbl>  ...
1  2015-01-01    0      0      0      0    ...
2  2015-01-02    0      0      0     0.2   ...
3  2015-01-03   0.1     0      0     0.1   ...
```

- Each station column contains numeric values representing observed measurements.
- The column names (station identifiers) must be unique and match those in `BD_Coord$Cod` to ensure proper spatial referencing.

`BD_Coord`
: A `data.table` or `data.frame` containing the metadata of the ground stations. It must include the following columns:

- `"Cod"`: Unique identifier for each ground station.
- `"X"`: Latitude of the station in UTM format.
- `"Y"`: Longitude of the station in UTM format.
- `"Z"`: Altitude of the station in meters.

`cov`
: A list of cov used as independent variables in the RFmerge. Each covariate should be a `SpatRaster` object (from the `terra` package) and can represent satellite-derived weather variables or a Digital Elevation Model (DEM). All cov should have the same number of layers (bands), except for the DEM, which must have only one layer.

`mask`	A shapefile defining the study area. If provided, the shapefile must be of class `SpatVector` (from the `terra` package) with a UTM coordinate reference system. When specified, a spatial mask is applied to ensure that the final precipitation estimates are restricted to the defined study area. Defaults to `NULL`, meaning no spatial mask is applied.
`n_round`	An integer specifying the number of decimal places to round the interpolated results. If set to `NULL`, all decimal places will be preserved. The default value is `1`.
`ntree`	Numeric indicating the maximum number trees to grow in the Random Forest algorithm. The default value is set to 2000. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. If this value is too low, the prediction may be biased.
`seed`	Integer for setting the random seed to ensure reproducibility of results (default: 123).
`training`	Numerical value between 0 and 1 indicating the proportion of data used for model training. The remaining data are used for validation. Note that if you enter, for example, 0.8 it means that 80 % of the data will be used for training and 20 % for validation. If you do not want to perform validation, set training = 1. (Default training = 1).
`stat_validation`	A character vector specifying the names of the stations to be used for validation. This option should only be filled in when it is desired to manually enter the stations used for validation. If this parameter is NULL, and the formation is different from 1, a validation will be performed using random stations. The vector must contain the names of the stations selected by the user for validation. For example, stat_validation = c("ST001", "ST002"). (Default stat_validation = NULL).
`Rain_threshold`	List of numerical vectors defining precipitation thresholds to classify precipitation into different categories according to its intensity. This parameter should be entered only when the validation is to include categorical metrics such as Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), etc. Each list item should represent a category, with the category name as the list item name and a numeric vector specifying the lower and upper bounds of that category. **Note:** See the "Notes" section for additional details on how to define categories, use this parameter for validation, and example configurations.
`save_model`	Logical value indicating whether the interpolation file should be saved to disk. The default value is `FALSE`. indicating that the interpolated file should not be saved. If set to `TRUE`, be sure to set the working directory beforehand using `setwd(path)` to specify where the files should be saved.
`name_save`	Character string indicating the name under which the interpolation raster file will be saved. By default the algorithm sets as output name: 'Model_RFmerge'. as the code will internally assign it.

Value

If a value other than 1 is set (point to pixel validation is performed), a list containing two elemeentis returned:

Ensamble: A `SpatRaster` object containing the bias-corrected layers for each time step. The number of layers corresponds to the number of dates for which the correction is applied. This represents the corrected satellite data adjusted for bias.

Validation: A list containing the statistical results obtained from the validation process. This list includes:

- `gof`: A data table with goodness-of-fit metrics such as Kling-Gupta Efficiency (KGE), Nash-Sutcliffe Efficiency (NSE), Percent Bias (PBIAS), Root Mean Square Error (RMSE), and Pearson Correlation Coefficient (CC). These metrics assess the overall performance of the bias correction process.
- `categorical_metrics`: A data frame containing categorical evaluation metrics such as Probability of Detection (POD), Success Ratio (SR), False Alarm Rate (FAR), Critical Success Index (CSI), and Hit Bias (HB). These metrics evaluate the classification performance of rainfall event predictions based on user-defined precipitation thresholds.

If training is set to 1 (No validation is performed) only the Assembly mentioned above is returned.

Details

The `Rain_threshold` parameter is used to calculate categorical metrics such as the Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), success ratio (SR), Hit BIAS (HB),Heidke Skill Score (HSS); Hanssen-Kuipers Discriminant (HK); Equal Threat Score (ETS) or Gilbert Skill Score. The parameter should be entered as a named list, where each item represents a category and the name of the item is the category name. The elements of each category must be a numeric vector with two values: the lower and upper limits of the category. For example: `Rain_threshold = list( no_rain = c(0, 1), light_rain = c(1, 5), moderate_rain = c(5, 20), heavy_rain = c(20, 40), violent_rain = c(40, Inf) )`

Precipitation values will be classified into these categories based on their intensity. Users can define as many categories as necessary, or just two (e.g., "rain" vs. "no rain"). It is important that these categories are entered according to the study region, as each study region may have its own categories.

References

Baez-Villanueva, O. M.; Zambrano-Bigiarini, M.; Beck, H.; McNamara, I.; Ribbe, L.; Nauditt, A.; Birkel, C.; Verbist, K.; Giraldo-Osorio, J.D.; Thinh, N.X. (2020). RF-MEP: a novel Random Forest method for merging gridded precipitation products and ground-based measurements, Remote Sensing of Environment, 239, 111610. doi:10.1016/j.rse.2019.111606.

Examples

```
# Load data from on-site observations
 data("BD_Obs", package = "InterpolateR")
 data("BD_Coord", package = "InterpolateR")

# Load the cov
cov <- list(
 MSWEP = terra::rast(system.file("extdata/MSWEP.nc", package = "InterpolateR")),
 CHIRPS = terra::rast(system.file("extdata/CHIRPS.nc", package = "InterpolateR")),
 DEM = terra::rast(system.file("extdata/DEM.nc", package = "InterpolateR"))
 )

 # Apply the RFmerge
 model_RFmerge = RFmerge(BD_Obs, BD_Coord, cov, mask = NULL, n_round = 1, ntree = 2000,
                         seed = 123,  training = 0.8, stat_validation = NULL,
                         Rain_threshold = NULL, save_model = FALSE, name_save = NULL)

# Visualize the results
# Precipitation results within the study area
modelo_rainfall = model_RFmerge$Ensamble

# Validation statistic results
# goodness-of-fit metrics
metrics_gof = model_RFmerge$Validation$gof

# categorical metrics
metrics_cat = model_RFmerge$Validation$categorical_metrics
```

`RFplus` *Machine learning algorithm for fusing ground and satellite precipitation data.*

Description

MS-GOP (RFplus) is a machine learning algorithm for merging satellite-based and ground precipitation data. It combines Random Forest for spatial prediction, residual modeling for bias correction, and quantile mapping for final adjustment, ensuring accurate precipitation estimates across different temporal scales

Usage

```
RFplus(
  BD_Obs,
  BD_Coord,
  Covariates,
  n_round = NULL,
  wet.day = FALSE,
  ntree = 2000,
  seed = 123,
  training = 1,
  stat_validation = NULL,
  Rain_threshold = NULL,
  method = c("RQUANT", "QUANT", "none"),
  ratio = 15,
  save_model = FALSE,
  name_save = NULL
)
```

Arguments

`BD_Obs`	A `data.table` or `data.frame` containing observational data with the following structure:

- The first column (`Date`): A `Date` object representing the observation date.
- The remaining columns: Each column corresponds to a unique ground station, where the column name is the station identifier.

The dataset should be structured as follows:

```
> BD_Obs
# A data.table or data.frame with n rows (dates) and m+1 columns (stations + Date)
   Date        ST001  ST002  ST003  ST004  ...
   <date>      <dbl>  <dbl>  <dbl>  <dbl>  ...
1  2015-01-01    0      0      0      0    ...
2  2015-01-02    0      0      0     0.2   ...
3  2015-01-03   0.1     0      0     0.1   ...
```

- Each station column contains numeric values representing observed measurements.
- The column names (station identifiers) must be unique and match those in `BD_Coord$Cod` to ensure proper spatial referencing.

`BD_Coord` `data.table` containing metadata for the ground stations. Must include the following columns:

- `"Cod"`: Unique identifier for each ground station.
- `"X"`: Latitude of the station in UTM format.
- `"Y"`: Longitude of the station in UTM format.
- `"Z"`: Altitude of the station in meters.

`Covariates` A list of covariates used as independent variables in the RFplus model. Each covariate should be a `SpatRaster` object (from the `terra` package) and can represent satellite-derived weather variables or a Digital Elevation Model (DEM). All covariates should have the same number of layers (bands), except for the DEM, which must have only one layer.

`n_round` Numeric indicating the number of decimal places to round the corrected values. If `n_round` is set to `NULL`, no rounding is applied.

`wet.day` Numeric value indicating the threshold for wet day correction. Values below this threshold will be set to zero.

- `wet.day = FALSE`: No correction is applied (default).
- For wet day correction, provide a numeric threshold (e.g., `wet.day = 0.1`).

`ntree` Numeric indicating the maximum number trees to grow in the Random Forest algorithm. The default value is set to 2000. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times. If this value is too low, the prediction may be biased.

`seed` Integer for setting the random seed to ensure reproducibility of results (default: 123).

`training`	Numerical value between 0 and 1 indicating the proportion of data used for model training. The remaining data are used for validation. Note that if you enter, for example, 0.8 it means that 80 % of the data will be used for training and 20 % for validation. If you do not want to perform validation, set training = 1. (Default training = 1).
`stat_validation`	A character vector specifying the names of the stations to be used for validation. This option should only be filled in when it is desired to manually enter the stations used for validation. If this parameter is NULL, and the formation is different from 1, a validation will be performed using random stations. The vector must contain the names of the stations selected by the user for validation. For example, stat_validation = c("ST001", "ST002"). (Default stat_validation = NULL).
`Rain_threshold`	List of numerical vectors defining precipitation thresholds to classify precipitation into different categories according to its intensity. This parameter should be entered only when the validation is to include categorical metrics such as Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), etc. Each list item should represent a category, with the category name as the list item name and a numeric vector specifying the lower and upper bounds of that category. **Note:** See the "Notes" section for additional details on how to define categories, use this parameter for validation, and example configurations.
`method`	A character string specifying the quantile mapping method used for distribution adjustment. Options are: • `"RQUANT"`: Robust quantile mapping to adjust satellite data distribution to observed data. • `"QUANT"`: Standard quantile mapping. • `"none"`: No distribution adjustment is applied. Only Random Forest-based bias correction and residual correction are performed.
`ratio`	integer Maximum search radius (in kilometers) for the quantile mapping setting using the nearest station. (default = 15 km)
`save_model`	Logical value indicating whether the interpolation file should be saved to disk. The default value is `FALSE`. indicating that the interpolated file should not be saved. If set to `TRUE`, be sure to set the working directory beforehand using `setwd(path)` to specify where the files should be saved.
`name_save`	Character string indicating the name under which the interpolation raster file will be saved. By default the algorithm sets as output name: 'Model_RFplus'.

Details

The `RFplus` method implements a three-step approach:

- **Base Prediction**: Random Forest model is trained using satellite data and covariates.
- **Residual Correction**: A second Random Forest model is used to correct the residuals from the base prediction.
- **Distribution Adjustment**: Quantile mapping (QUANT or RQUANT) is applied to adjust the distribution of satellite data to match the observed data distribution.

The final result combines all three steps, correcting the biases while preserving the outliers, and improving the accuracy of satellite-derived data such as precipitation and temperature.

Value

A list containing two elements:

Ensamble: A `SpatRaster` object containing the bias-corrected layers for each time step. The number of layers corresponds to the number of dates for which the correction is applied. This represents the corrected satellite data adjusted for bias.

Validation: A list containing the statistical results obtained from the validation process. This list includes:

- `gof`: A data table with goodness-of-fit metrics such as Kling-Gupta Efficiency (KGE), Nash-Sutcliffe Efficiency (NSE), Percent Bias (PBIAS), Root Mean Square Error (RMSE), and Pearson Correlation Coefficient (CC). These metrics assess the overall performance of the bias correction process.
- `categorical_metrics`: A data frame containing categorical evaluation metrics such as Probability of Detection (POD), Success Ratio (SR), False Alarm Rate (FAR), Critical Success Index (CSI), and Hit Bias (HB). These metrics evaluate the classification performance of rainfall event predictions based on user-defined precipitation thresholds.

Notes

The `Rain_threshold` parameter is used to calculate categorical metrics such as the Critical Success Index (CSI), Probability of Detection (POD), False Alarm Rate (FAR), success ratio (SR), Hit BIAS (HB),Heidke Skill Score (HSS); Hanssen-Kuipers Discriminant (HK); Equal Threat Score (ETS) or Gilbert Skill Score. The parameter should be entered as a named list, where each item represents a category and the name of the item is the category name. The elements of each category must be a numeric vector with two values: the lower and upper limits of the category. For example: `Rain_threshold = list( no_rain = c(0, 1), light_rain = c(1, 5), moderate_rain = c(5, 20), heavy_rain = c(20, 40), violent_rain = c(40, Inf) )`

Precipitation values will be classified into these categories based on their intensity. Users can define as many categories as necessary, or just two (e.g., "rain" vs. "no rain"). It is important that these categories are entered according to the study region, as each study region may have its own categories.

Author(s)

Jonnathan Augusto landi Bermeo, jonnathan.landi@outlook.com

Examples

```
# Load the data
 data("BD_Obs", package = "InterpolateR")
 data("BD_Coord", package = "InterpolateR")

# Load the covariates
Covariates <- list(
 MSWEP = terra::rast(system.file("extdata/MSWEP.nc", package = "InterpolateR")),
 CHIRPS = terra::rast(system.file("extdata/CHIRPS.nc", package = "InterpolateR")),
 DEM = terra::rast(system.file("extdata/DEM.nc", package = "InterpolateR"))
 )

 # Apply the RFplus bias correction model
model = RFplus(BD_Obs, BD_Coord, Covariates, n_round = 1, wet.day = 0.1,
        ntree = 2000, seed = 123, training = 0.8,
        Rain_threshold = list(no_rain = c(0, 1), light_rain = c(1, 5)),
        method = "RQUANT", ratio = 10, save_model = FALSE, name_save = NULL)

# Visualize the results
# Precipitation results within the study area
modelo_rainfall = model$Ensamble

# Validation statistic results
# goodness-of-fit metrics
metrics_gof = model$Validation$gof

# categorical metrics
metrics_cat = model$Validation$categorical_metrics
```

Index