# Package: ARPALData (via r-universe)

August 27, 2024

**Type** Package

**Title** Retrieving and Analyzing Air Quality and Weather Data from ARPA Lombardia

**Version** 1.6.0

**Description** Contains functions for retrieving, managing and analysing air quality and weather data from Regione Lombardia open database (<https://www.dati.lombardia.it/>). Data are collected by ARPA Lombardia (Lombardia Environmental Protection Agency), Italy, through its ground monitoring network (<https://www.dati.lombardia.it/stories/s/auv9-c2sj>). See the webpage <https://www.arpalombardia.it/> for further information on ARPA Lombardia's activities and history. Data quality (e.g. missing values, exported values, graphical mapping) has been checked involving members of the ARPA Lombardia's office for air quality control. The package makes available observations since 1989 (for weather) and 1968 (for air quality) and are updated with daily frequency by the regional agency. Full description of the package can be retrieved in the companion paper Maranzano \& Algieri (2024), ``ARPALData: an R package for retrieving and analyzing air quality and weather data from ARPA Lombardia (Italy)'', Environmental and Ecological Statistics, <doi:10.1007/s10651-024-00599-6>.

**License** GPL (>= 2)

**Imports** tidyselect,dplyr,utils,lubridate,rlang,readr,stringr,tm,tidyr,purrr,eurostat,sf,ggplot2,tibble,aweek,curl,archive,future,future.a

**Suggests** tidyverse, RSocrata, knitr, rmarkdown

**Encoding** UTF-8

**Language** en-US

**RoxygenNote** 7.3.1

**NeedsCompilation** no

**Author** Paolo Maranzano [aut, cre, cph] (<https://orcid.org/0000-0002-9228-2759>), Andrea Algieri [aut, cph]

**Maintainer** Paolo Maranzano <pmaranzano.ricercastatistica@gmail.com>

**Date/Publication** 2024-08-26 14:30:05 UTC

**Repository** https://paolomaranzano.r-universe.dev

**RemoteUrl** https://github.com/cran/ARPALData

**RemoteRef** HEAD

**RemoteSha** acabdb546a8a98219798062be1e8d3266f2acf98

# Contents



---

`ARPALData` *ARPALData Package*

---

## Description

Contains functions for downloading and managing air quality and weather data from Regione Lombardia open database. Data are collected by ARPA Lombardia (Lombardia Environmental Protection Agency), Italy.

## Author(s)

Paolo Maranzano <pmaranzano.ricercastatistica@gmail.com>

ARPALdf_Summary *Summary statistics for a data frame of class 'ARPALdf'*

## Description

'ARPALdf_Summary' returns many descriptive statistics summaring the data contained in a data frame of class ARPALdf. Statistics are calculated at overall level (full sample), by station ID and by year. For each variable are reported the basic positioning indices (min, max, mean, median, quantile) and variability indices (range, standard deviation). Other reported statistics are the Pearson's linear correlation by station and some graphical representation of the distribution (kernel density plot, histogram, boxplot). In addition, the function returns useful data-quality information, such as gap length statistics (i.e. number of missing observations for each variable by station and by year) and outlier detection tools (e.g., Hampel filter and boxplot rule)

## Usage

```
ARPALdf_Summary(
  Data,
  by_IDStat = TRUE,
  by_Year = TRUE,
  gap_length = TRUE,
  correlation = TRUE,
  histogram = FALSE,
  density = FALSE,
  outlier = FALSE,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| Data | Dataset of class 'ARPALdf' containing the data to be summarised. |
| by_IDStat | Logic value (TRUE or FALSE). Use TRUE (default) to compute summary statistics by Station ID. |
| by_Year | Logic value (TRUE or FALSE). Use TRUE (default) to compute summary statistics by year. |
| gap_length | Logic value (TRUE or FALSE). Use TRUE (default) to compute summary statistics for the gap length of each variable. |
| correlation | Logic value (TRUE or FALSE). Use TRUE (default) to compute linear correlation of available variables. |
| histogram | Logic value (TRUE or FALSE). Use TRUE to plot the histogram of each variable. Default is FALSE. |
| density | Logic value (TRUE or FALSE). Use TRUE to plot the kernel density plot of each variable. Default is FALSE. |
| outlier | Logic value (TRUE or FALSE). Use TRUE to analyse extreme values of each variable (boxplot and Hampel filter). Default is FALSE. |

verbose — Logic value (TRUE or FALSE). Toggle warnings and messages. If 'verbose = TRUE' (default) the function prints on the screen some messages describing the progress of the tasks. If 'verbose = FALSE' any message about the progression is suppressed.

## Value

A list of data.frames containing summary descriptive statistics for a data frame of class 'ARPALdf'. Summary statistics are computed for the overall sample (Descr), by Station ID (Descr_by_IDStat) and by year (Descr_by_Year). Available statistics are: number of NAs, number of negative values, minimum, mean, maximum and standard deviation.

## Examples

```
## Download daily air quality data from all the stations for year 2020
if (require("RSocrata")) {
  d <- get_ARPA_Lombardia_AQ_data(ID_station = NULL, Date_begin = "2020-01-01",
            Date_end = "2020-12-31", Frequency = "daily")
 }
## Summarising observed data
sum_stats <- ARPALdf_Summary(Data = d)
```

---

# ARPALdf_Summary_map *Generate a map of summary statistics for a given ARPALdf data.frame*

---

## Description

'ARPALdf_Summary_map' represents on a map (polygon of Lombardy) the data contained in a data frame of class 'ARPALdf' containing the values or the descriptive statistics by station. Data can be either a ARPALdf of observed data (from 'get_ARPA_Lombardia_xxx' commands) and an ARPALdf obtained as summary descriptive statistic (from 'ARPALdf_Summary' command).

## Usage

```
ARPALdf_Summary_map(
  Data,
  Title_main,
  Title_legend = "Variable",
  Variable,
  prov_line_type = 1,
  prov_line_size = 1,
  col_scale = c("#00FF00", "#FFFF00", "#FF0000"),
  val_midpoint = NULL,
  xlab = "Longitude",
  ylab = "Latitude"
)
```

## Arguments

| | |
|---|---|
| Data | Dataset of class 'ARPALdf' containing the values or the descriptive statistics to plot on the map. Data can be either a ARPALdf of observed data (from 'get_ARPA_Lombardia_xxx' commands) and an ARPALdf obtained as summary descriptive statistic (from 'ARPALdf_Summary' command). |
| Title_main | Title of the plot. |
| Title_legend | Title fo the legend |
| Variable | Summary variable to represent |
| prov_line_type | Linetype for Lombardy provinces. Default is 1. |
| prov_line_size | Size of the line for Lombardy provinces. Default is 1. |
| col_scale | Vector indicating the minimum, the middle and the average point colors. Default is c("green","yellow","red"). |
| val_midpoint | Numeric. Value associated to the middle-point scale color. Default is NULL (midpoint is set equal to the average of the variable to represent). |
| xlab | x-axis label. Default is 'Longitude'. |
| ylab | y-axis label. Default is 'Latitude'. |

## Value

A map of selected stations across the Lombardy region

## Examples

```
## Download daily air quality data from all the stations for year 2020
if (require("RSocrata")) {
    d <- get_ARPA_Lombardia_AQ_data(ID_station = NULL, Date_begin = "2020-01-01",
              Date_end = "2020-12-31", Frequency = "daily")
}
## Summarising observed data
s <- ARPALdf_Summary(Data = d)
## Mapping of the average NO2 in 2020 at several stations
ARPALdf_Summary_map(Data = s$Descr_by_IDStat$Mean_by_stat,
        Title_main = "Mean NO2 by station in 2020", Variable = "NO2")
```

---

`get_ARPA_Lombardia_AQ_data`

*Download air quality data from ARPA Lombardia website*

---

## Description

'get_ARPA_Lombardia_AQ_data' returns observed air quality measurements collected by ARPA Lombardia ground detection system for Lombardy region in Northern Italy. Available airborne pollutant concentrations are: NO2, NOx, PM10, PM2.5, Ozone, Arsenic, Benzene, Benzo-a-pirene, Ammonia, Sulfur Dioxide, Black Carbon, CO, Nikel, Cadmium and Lead. Data are available from 1968 and are updated up to the current date (2023). For more information about the municipal data visit the section 'Monitoraggio aria' at the webpage: https://www.dati.lombardia.it/stories/s/auv9-c2sj

## Usage

```
get_ARPA_Lombardia_AQ_data(
  ID_station = NULL,
  Date_begin = "2022-01-01",
  Date_end = "2022-12-31",
  Frequency = "hourly",
  Var_vec = NULL,
  Fns_vec = NULL,
  by_sensor = FALSE,
  verbose = TRUE,
  parallel = FALSE,
  parworkers = NULL,
  parfuturetype = "multisession"
)
```

## Arguments

| | |
|---|---|
| ID_station | Numeric value. ID of the station to consider. Using ID_station = NULL, all the available stations are selected. Default is ID_station = NULL. |
| Date_begin | Character vector of the first date-time to download. Format can be either "YYYY-MM-DD" or "YYYY-MM-DD hh:mm:ss". Default is Date_begin = "2022-01-01". |
| Date_end | Character vector of the last date-time to download. Format can be either "YYYY-MM-DD" or "YYYY-MM-DD hh:mm:ss". Default is Date_end = "2022-12-31". |
| Frequency | Temporal aggregation frequency. It can be "hourly", "daily", "weekly", "monthly" or "yearly". Default is Frequency = "hourly". |
| Var_vec | Character vector of variables to aggregate. If NULL (default) all the variables are averaged. |
| Fns_vec | Character vector of aggregation function to apply to the selected variables. Available functions are mean, median, min, max, sum, qPP (PP-th percentile), sd, var, vc (variability coefficient), skew (skewness) and kurt (kurtosis). |
| by_sensor | Logic value (TRUE or FALSE). If 'by_sensor = TRUE', the function returns the observed concentrations by sensor code, while if 'by_sensor = FALSE' (default) it returns the observed concentrations by station. |

verbose — Logic value (TRUE or FALSE). Toggle warnings and messages. If 'verbose = TRUE' (default) the function prints on the screen some messages describing the progress of the tasks. If 'verbose = FALSE' any message about the progression is suppressed.

parallel — Logic value (TRUE or FALSE). If 'parallel = FALSE' (default), data downloading is performed using a sequential/serial approach and additional parameters 'parworkers' and 'parfuturetype' are ignored. When 'parallel = TRUE', data downloading is performed using parallel computing through the Futureverse setting. More detailed information about parallel computing in the Futureverse can be found at the following webpages: https://future.futureverse.org/ and https://cran.r-project.org/web/packages/future.apply/vignettes/future.apply-1-overview.html

parworkers — Numeric integer value. If 'parallel = TRUE' (parallel mode active), the user can declare the number of parallel workers to be activated using 'parworkers = integer number'. By default ('parworkers = NULL'), the number of active workers is half of the available local cores.

parfuturetype — Character vector. If 'parallel = TRUE' (parallel mode active), the user can declare the parallel strategy to be used according to the Futureverse syntax through 'parfuturetype'. By default, the 'multisession' (background R sessions on local machine) is used. In alternative, the 'multicore' (forked R processes on local machine. Not supported by Windows and RStudio) setting can be used.

## Value

A data frame of class 'data.frame' and 'ARPALdf'. The object is fully compatible with Tidyverse.

## Examples

```
## Download hourly air quality data for 2022 at station 501.
if (require("RSocrata")) {
    get_ARPA_Lombardia_AQ_data(ID_station=501, Date_begin = "2022-01-01",
        Date_end = "2022-12-31", Frequency="hourly", parallel = TRUE)
}
## Download (parallel) monthly data for NOx and NO2 observed between May and
## August 2021 for all the stations active on the network. For NOx is computed
## the 25th percentile, while for NO2 is computed the maximum concentration observed.
if (require("RSocrata")) {
    get_ARPA_Lombardia_AQ_data(ID_station=NULL,Date_begin = "2024-05-01",
        Date_end = "2024-08-01", Frequency="monthly",Var_vec=c("NOx","NO2"),
        Fns_vec=c("q25","max"), parallel = TRUE)
}
## Download hourly air quality data by sensor for January 2023 at station 501.
if (require("RSocrata")) {
    get_ARPA_Lombardia_AQ_data(ID_station=501,Date_begin = "2023-01-01 00:00:00",
        Date_end = "2023-01-31 23:00:00", by_sensor = TRUE)
}
```

`get_ARPA_Lombardia_AQ_municipal_data` *Download air quality data at manucipal from ARPA Lombardia website*

## Description

'get_ARPA_Lombardia_AQ_municipal_data' returns the air quality levels at municipal level estimated by ARPA Lombardia using a physico-chemical model which simulates air quality based on weather and geo-physical variables. For each municipality of Lombardy, ARPA estimates the average (NO2_mean) and maximum daily (NO2_max_day) level of NO2, the daily maximum (Ozone_max_day) and the 8-hours moving window maximum (Ozone_max_8h) of Ozone and the average levels of PM10 (PM10_mean) and PM2.5 (PM2.5_mean). Data are available from 2011 and are updated up to the current date. For more information about the municipal data visit the section 'Stime comunali dell'aria' at the webpage: https://www.dati.lombardia.it/stories/s/auv9-c2sj

## Usage

```
get_ARPA_Lombardia_AQ_municipal_data(
  ID_station = NULL,
  Date_begin = "2021-01-01",
  Date_end = "2022-12-31",
  Frequency = "daily",
  Var_vec = NULL,
  Fns_vec = NULL,
  by_sensor = FALSE,
  verbose = TRUE,
  parallel = FALSE,
  parworkers = NULL,
  parfuturetype = "multisession"
)
```

## Arguments

| | |
|---|---|
| `ID_station` | Numeric value. ID of the station to consider. Using ID_station = NULL, all the available stations are selected. Default is ID_station = NULL. |
| `Date_begin` | Character vector of the first date-time to download. Format can be either "YYYY-MM-DD" or "YYYY-MM-DD hh:mm:ss". Default is Date_begin = "2022-01-01". |
| `Date_end` | Character vector of the last date-time to download. Format can be either "YYYY-MM-DD" or "YYYY-MM-DD hh:mm:ss". Default is Date_end = "2022-12-31". |
| `Frequency` | Temporal aggregation frequency. It can be "daily", "weekly", "monthly" or "yearly". Default is Frequency = "daily". |
| `Var_vec` | Character vector of variables to aggregate. If NULL (default) all the variables are averaged. |

| | |
|---|---|
| Fns_vec | Character vector of aggregation function to apply to the selected variables. Available functions are mean, median, min, max, sum, qPP (PP-th percentile), sd, var, vc (variability coefficient), skew (skewness) and kurt (kurtosis). |
| by_sensor | Logic value (TRUE or FALSE). If 'by_sensor = TRUE', the function returns the observed concentrations by sensor code, while if 'by_sensor = FALSE' (default) it returns the observed concentrations by station. |
| verbose | Logic value (TRUE or FALSE). Toggle warnings and messages. If 'verbose = TRUE' (default) the function prints on the screen some messages describing the progress of the tasks. If 'verbose = FALSE' any message about the progression is suppressed. |
| parallel | Logic value (TRUE or FALSE). If 'parallel = FALSE' (default), data downloading is performed using a sequential/serial approach and additional parameters 'parworkers' and 'parfuturetype' are ignored. When 'parallel = TRUE', data downloading is performed using parallel computing through the Futureverse setting. More detailed information about parallel computing in the Futureverse can be found at the following webpages: https://future.futureverse.org/ and https://cran.r-project.org/web/packages/future.apply/vignettes/future.apply-1-overview.html |
| parworkers | Numeric integer value. If 'parallel = TRUE' (parallel mode active), the user can declare the number of parallel workers to be activated using 'parworkers = integer number'. By default ('parworkers = NULL'), the number of active workers is half of the available local cores. |
| parfuturetype | Character vector. If 'parallel = TRUE' (parallel mode active), the user can declare the parallel strategy to be used according to the Futureverse syntax through 'parfuturetype'. By default, the 'multisession' (background R sessions on local machine) is used. In alternative, the 'multicore' (forked R processes on local machine. Not supported by Windows and RStudio) setting can be used. |

## Details

More detailed description.

## Value

A data frame of class 'data.frame' and 'ARPALdf'. The object is fully compatible with Tidyverse. The column 'NameStation' identifies the name of each municipality. The column 'IDStation' is an ID code (assigned from ARPA) uniquely identifying each municipality.

## Examples

```
## Download daily concentrations at municipal levels observed in 2020
## for all the municipalities in Lombardy
if (require("RSocrata")) {
    get_ARPA_Lombardia_AQ_municipal_data(ID_station=NULL,Date_begin = "2022-01-01",
          Date_end = "2023-12-31", Frequency="daily")
}
## Download monthly concentrations of NO2 (average and maximum) observed in 2021
## at city number 100451.
```

```
if (require("RSocrata")) {
    get_ARPA_Lombardia_AQ_municipal_data(ID_station=100451,Date_begin = "2023-01-01",
          Date_end = "2023-12-31", Frequency="monthly",Var_vec=c("NO2_mean","NO2_mean"),
          Fns_vec=c("mean","max"))
}
## Download daily concentrations observed in March and April 2022 at city number 100451.
## Data are reported by sensor.
if (require("RSocrata")) {
    get_ARPA_Lombardia_AQ_municipal_data(ID_station=100451, Date_begin = "2024-03-01",
          Date_end = "2024-04-30", by_sensor = TRUE)
}
```

---

`get_ARPA_Lombardia_AQ_municipal_registry` *Download metadata (registry) on air quality monitoring stations at municipal level from ARPA Lombardia website*

---

## Description

'get_ARPA_Lombardia_AQ_municipal_registry' returns the registry (list) of all the air quality sensors owned by ARPA Lombardia for each municipality of Lombardy. The information reported are: ID of each sensor and station, starting date and ending date. The column 'NameStation' identifies the name of each municipality. The column 'IDStation' is an ID code (assigned from ARPA) uniquely identifying each municipality. For more information about the municipal data visit the section 'Stime comunali sull'aria' at the webpage: https://www.dati.lombardia.it/stories/s/auv9-c2sj

## Usage

```
get_ARPA_Lombardia_AQ_municipal_registry()
```

## Value

A data frame of class 'data.frame' and 'ARPALdf'. The object is fully compatible with Tidyverse.

## Examples

```
get_ARPA_Lombardia_AQ_municipal_registry()
```

## get_ARPA_Lombardia_AQ_registry

*Download metadata (registry) on air quality monitoring stations from ARPA Lombardia website*

### Description

'get_ARPA_Lombardia_AQ_registry' returns the registry (list) of all the air quality sensors and stations belonging to the ARPA Lombardia network. The information reported are: ID of each sensor and station, geo-location (coordinates in degrees), altitude (mt), starting date and ending date. The column 'NameStation' identifies the name of each station, while 'IDStation' is an ID code (assigned from ARPA) uniquely identifying each station. For more information about the municipal data visit the section 'Monitoraggio aria' at the webpage: https://www.dati.lombardia.it/stories/s/auv9-c2sj

### Usage

```
get_ARPA_Lombardia_AQ_registry()
```

### Value

A data frame of class 'data.frame' and 'ARPALdf'. The object is fully compatible with Tidyverse.

### Examples

```
get_ARPA_Lombardia_AQ_registry()
```

## get_ARPA_Lombardia_W_data

*Download weather/meteorological data from ARPA Lombardia website*

### Description

'get_ARPA_Lombardia_W_data' returns observed air weather measurements collected by ARPA Lombardia ground detection system for Lombardy region in Northern Italy. Available meteorological variables are: temperature (Celsius degrees), rainfall (mm), wind speed (m/s), wind direction (degrees), relative humidity ( Data are available from 1989 and are updated up to the current date. For more information about the municipal data visit the section 'Monitoraggio aria' at the webpage: https://www.dati.lombardia.it/stories/s/auv9-c2sj

## Usage

```
get_ARPA_Lombardia_W_data(
  ID_station = NULL,
  Date_begin = "2021-01-01",
  Date_end = "2022-12-31",
  Frequency = "10mins",
  Var_vec = NULL,
  Fns_vec = NULL,
  by_sensor = FALSE,
  verbose = TRUE,
  parallel = FALSE,
  parworkers = NULL,
  parfuturetype = "multisession"
)
```

## Arguments

| | |
|---|---|
| ID_station | Numeric value. ID of the station to consider. Using ID_station = NULL, all the available stations are selected. Default is ID_station = NULL. |
| Date_begin | Character vector of the first date-time to download. Format can be either "YYYY-MM-DD" or "YYYY-MM-DD hh:mm:ss". Default is Date_begin = "2022-01-01". |
| Date_end | Character vector of the last date-time to download. Format can be either "YYYY-MM-DD" or "YYYY-MM-DD hh:mm:ss". Default is Date_end = "2022-12-31". |
| Frequency | Temporal aggregation frequency. It can be "10mins", "hourly", "daily", "weekly", "monthly". Default is Frequency = "10mins" |
| Var_vec | Character vector of variables to aggregate. If NULL (default) all the variables are averaged, except for 'Temperature' and 'Snow_height', which are cumulated. |
| Fns_vec | Character vector of aggregation function to apply to the selected variables. Available functions are mean, median, min, max, sum, qPP (PP-th percentile), sd, var, vc (variability coefficient), skew (skewness) and kurt (kurtosis). Attention: for Wind Speed and Wind Speed Gust only mean, min and max are available; for Wind Direction and Wind Direction Gust only mean is available. |
| by_sensor | Logic value (TRUE or FALSE). If 'by_sensor = TRUE', the function returns the observed concentrations by sensor code, while if 'by_sensor = FALSE' (default) it returns the observed concentrations by station. |
| verbose | Logic value (TRUE or FALSE). Toggle warnings and messages. If 'verbose = TRUE' (default) the function prints on the screen some messages describing the progress of the tasks. If 'verbose = FALSE' any message about the progression is suppressed. |
| parallel | Logic value (TRUE or FALSE). If 'parallel = FALSE' (default), data downloading is performed using a sequential/serial approach and additional parameters 'parworkers' and 'parfuturetype' are ignored. When 'parallel = TRUE', |

data downloading is performed using parallel computing through the Futureverse setting. More detailed information about parallel computing in the Futureverse can be found at the following webpages: https://future.futureverse.org/ and https://cran.r-project.org/web/packages/future.apply/vignettes/future.apply-1-overview.html

`parworkers` Numeric integer value. If 'parallel = TRUE' (parallel mode active), the user can declare the number of parallel workers to be activated using 'parworkers = integer number'. By default ('parworkers = NULL'), the number of active workers is half of the available local cores.

`parfuturetype` Character vector. If 'parallel = TRUE' (parallel mode active), the user can declare the parallel strategy to be used according to the Futureverse syntax through 'parfuturetype'. By default, the 'multisession' (background R sessions on local machine) is used. In alternative, the 'multicore' (forked R processes on local machine. Not supported by Windows and RStudio) setting can be used.

## Details

More detailed description.

## Value

A data frame of class 'data.frame' and 'ARPALdf'. The object is fully compatible with Tidyverse.

## Examples

```
## Download all the (10 minutes frequency) weather measurements at station 100
## between August 2021 and December 2022.
if (require("RSocrata")) {
    get_ARPA_Lombardia_W_data(ID_station = 100, Date_begin = "2021-08-01",
          Date_end = "2022-12-31", Frequency = "10mins")
}
## Download all the (daily frequency) weather measurements at station 1974 during 2022
if (require("RSocrata")) {
    get_ARPA_Lombardia_W_data(ID_station = 1974, Date_begin = "2023-01-01",
          Date_end = "2023-12-31", Frequency = "daily")
}
```

---

`get_ARPA_Lombardia_W_registry`

*Download metadata (registry) on weather monitoring stations from ARPA Lombardia website*

---

## Description

'get_ARPA_Lombardia_W_registry' returns the registry (list) of all the weather sensors and stations belonging to the ARPA Lombardia network. The information reported are: ID of each sensor and station, geo-location (coordinates in degrees), altitude (mt), starting date and ending date. The column 'NameStation' identifies the name of each station, while 'IDStation' is an ID code (assigned from ARPA) uniquely identifying each station. For more information about the municipal data visit the section 'Meteo' at the webpages: https://www.dati.lombardia.it/stories/s/auv9-c2sj and https://www.dati.lombardia.it/Ambiente/Stazioni-Meteorologiche/nf78-nj6b

## Usage

```
get_ARPA_Lombardia_W_registry()
```

## Value

A data frame of class 'data.frame' and 'ARPALdf'. The object is fully compatible with Tidyverse.

## Examples

```
get_ARPA_Lombardia_W_registry()
```

---

# get_ARPA_Lombardia_zoning

*Download ARPA Lombardia zoning geometries*

---

## Description

'get_ARPA_Lombardia_zoning' returns the geometries (polygonal shape file) and a map of the ARPA zoning of Lombardy. The zoning reflects the main orographic characteristics of the territory. Lombardy region is classified into seven type of areas: large urbanized areas, urbanized areas in rural contexts, rural areas, mountainous areas and valley bottom. For more information about the municipal data visit the section 'Zonizzazione ARPA Lombardia' at the webpages https://www.arpalombardia.it/temi-ambientali/aria/rete-di-rilevamento/classificazione-zone/ and https://www.arpalombardia.it/temi-ambientali/aria/mappa-della-zonizzazione/

## Usage

```
get_ARPA_Lombardia_zoning(
  plot_map = TRUE,
  title = "ARPA Lombardia zoning",
  line_type = 1,
  line_size = 1,
  xlab = "Longitude",
  ylab = "Latitude"
)
```

### Arguments

| | |
|---|---|
| plot_map | Logic value (FALSE or TRUE). If plot_map = TRUE, the ARPA Lombardia zoning is represented on a map, if plot_mat = FALSE only the geometry (polygon shapefile) is stored in the output. Default is plot_map = TRUE. |
| title | Title of the plot. Deafult is 'ARPA Lombardia zoning' |
| line_type | Linetype for the zones' borders. Default is 1. |
| line_size | Size of the line for the zones. Default is 1. |
| xlab | x-axis label. Default is 'Longitude'. |
| ylab | y-axis label. Default is 'Latitude'. |

### Value

The function returns an object of class 'sf' containing the polygon borders of the seven zones used by ARPA Lombardia to classify the regional territory. If plot_map = 1, it also returns a map of the zoning.

### Examples

```
zones <- get_ARPA_Lombardia_zoning(plot_map = TRUE)
```

---

## get_Lombardia_geospatial

*Download geospatial data of Lombardy from Eurostat*

---

### Description

'get_Lombardia_geospatial' returns the polygonal (shape file) object containing the geometries of Lombardy. Shapefile are available at different NUTS levels (https://ec.europa.eu/eurostat/web/nuts/background): 'LAU' for the shapefile of municipalities of Lombardy, 'NUTS3' for the shapefile of provinces of Lombardy and 'NUTS2' for the shapefile of Lombardy.

### Usage

```
get_Lombardia_geospatial(NUTS_level = "LAU")
```

### Arguments

| | |
|---|---|
| NUTS_level | NUTS level required: use "NUTS2" for regional geometries, "NUTS3" for provincial geometries, or "LAU" for municipal geometries. Default NUTS_level = "LAU". |

### Value

A data frame of class 'data.frame', "sf" and 'ARPALdf'.

## Examples

```
shape <- get_Lombardia_geospatial(NUTS_level = "LAU")
```

---

`is_ARPALdf` *Check if a given object belongs to class 'ARPALdf'*

---

## Description

'is_ARPALdf' checks if the input object belongs to the class 'ARPALdf'

## Usage

```
is_ARPALdf(Data)
```

## Arguments

| | |
|---|---|
| `Data` | Object to check if the class of a dataframe is 'ARPALdf', i.e. ARPAL dataframe. |

## Value

The function returns 'True' if the object is of class 'ARPALdf' and it returns 'False' if the object isn't of class 'ARPALdf'

## Examples

```
d <- get_ARPA_Lombardia_AQ_registry()
is_ARPALdf(d)
```

---

`is_ARPALdf_AQ` *Check if a given object belongs to class 'ARPALdf_AQ'*

---

## Description

'is_ARPALdf_AQ' checks if the input object belongs to the class 'ARPALdf_AQ'

## Usage

```
is_ARPALdf_AQ(Data)
```

## Arguments

| | |
|---|---|
| `Data` | Object to check if the class of a dataframe is 'ARPALdf_AQ', i.e. ARPAL dataframe for air quality data. |

## Value

The function returns 'True' if the object is of class 'ARPALdf_AQ' and it returns 'False' if the object isn't of class 'ARPALdf_AQ'

## Examples

```
d <- get_ARPA_Lombardia_AQ_registry()
is_ARPALdf_AQ(d)
```

---

`is_ARPALdf_AQ_mun` *Check if a given object belongs to class 'ARPALdf_AQ_mun'*

---

## Description

'is_ARPALdf_AQ_mun' checks if the input object belongs to the class 'ARPALdf_AQ_mun'

## Usage

```
is_ARPALdf_AQ_mun(Data)
```

## Arguments

| | |
|---|---|
| `Data` | Object to check if the class of a dataframe is 'ARPALdf_AQ_mun', i.e. ARPAL dataframe for air quality data at municipal level (See 'get_ARPA_Lombardia_AQ_municipal_data'. command). |

## Value

The function returns 'True' if the object is of class 'ARPALdf_AQ_mun' and it returns 'False' if the object isn't of class 'ARPALdf_AQ_mun'

## Examples

```
d <- get_ARPA_Lombardia_AQ_registry()
is_ARPALdf_AQ_mun(d)
```

## is_ARPALdf_W *Check if a given object belongs to class 'ARPALdf_W'*

### Description

'is_ARPALdf_W' checks if the input object belongs to the class 'ARPALdf_W'

### Usage

```
is_ARPALdf_W(Data)
```

### Arguments

| | |
|---|---|
| Data | Object to check if the class of a dataframe is 'ARPALdf_W', i.e. ARPAL dataframe for weather data. |

### Value

The function returns 'True' if the object is of class 'ARPALdf_W' and it returns 'False' if the object isn't of class 'ARPALdf_W'

### Examples

```
d <- get_ARPA_Lombardia_W_registry()
is_ARPALdf_W(d)
```

## map_Lombardia_stations *Generate a map of the selected stations*

### Description

'get_ARPA_Lombardia_AQ_data' represents on a map (geometries/polygon of Lombardy) the location of the stations contained in a data frame of class 'ARPALdf'. Data can be either a ARPALdf of observed data (from 'get_ARPA_Lombardia_xxx' commands) and an ARPALdf obtained as registry (from 'get_ARPA_Lombardia_xxx_registry' command).

### Usage

```
map_Lombardia_stations(
  data,
  title = "Map of ARPA stations in Lombardy",
  prov_line_type = 1,
  prov_line_size = 1,
  col_points = "blue",
```

```
  xlab = "Longitude",
  ylab = "Latitude"
)
```

## Arguments

| | |
|---|---|
| data | Dataset of class 'ARPALdf' containing the stations to plot on the map. It can be either a ARPALdf of observed data (from 'get_ARPA_Lombardia_xxx' commands) and an ARPALdf obtained as registry (from 'get_ARPA_Lombardia_xxx_registry' command). |
| title | Title of the plot. Deafult is 'Map of ARPA stations in Lombardy' |
| prov_line_type | Linetype for Lombardy provinces. Default is 1. |
| prov_line_size | Size of the line for Lombardy provinces. Default is 1. |
| col_points | Color of the points. Default is 'blue'. |
| xlab | x-axis label. Default is 'Longitude'. |
| ylab | y-axis label. Default is 'Latitude'. |

## Value

A map of selected stations across the Lombardy region

## Examples

```
## Map network from a dataset of measurements
if (require("RSocrata")) {
  # Download daily concentrations observed at all the stations in 2020.
  d <- get_ARPA_Lombardia_AQ_data(ID_station = NULL, Date_begin = "2020-01-01",
                                  Date_end = "2020-12-31", Frequency = "daily")
  # Map the stations included in 'd'
  map_Lombardia_stations(data = d, title = "Air quality stations in Lombardy")
}
## Map network from a registry dataset
if (require("RSocrata")) {
  # Download registry for all the AQ stations in 2020.
  r <- get_ARPA_Lombardia_AQ_registry()
  # Map the stations included in 'r'
  map_Lombardia_stations(data = r, title = "Air quality stations in Lombardy")
}
```

---

`registry_KNN_dist` *Identifies the K-nearest-neighbors (stations) to all the monitoring sites included in a given ARPALdf registry data.frame. The neighbors are identified computing the Euclidean distance among the sites' coordinates.*

---

## Description

For each element included in reg_X, it identifies the k-nearest neighbors locations (among those included in reg_Y) according to an Euclidean distance metric. reg_X and reg_Y must be two 'ARPALdf' objects obtained using get_ARPA_Lombardia_xxx_registry'.

## Usage

```
registry_KNN_dist(reg_X, reg_Y, k = 1)
```

## Arguments

| | |
|---|---|
| reg_X | Dataset of class 'ARPALdf' containing the stations list obtained as registry (from 'get_ARPA_Lombardia_xxx_registry' command). The object must contain the following colums: 'IDStation','NameStation','Longitude' and 'Latitude'. |
| reg_Y | Dataset of class 'ARPALdf' containing the stations list obtained as registry (from 'get_ARPA_Lombardia_xxx_registry' command). The object must contain the following colums: 'IDStation','NameStation','Longitude' and 'Latitude'. |
| k | Integer value. Represents the number of neighbors the user wants to identify. |

## Value

A data.frame object having the same length of reg_X. For each row (stations in reg_X) it contains the name and the IDStation code for the k-nearest neighbors.

## Examples

```
if (require("tidyverse")) {
    regAQ <- get_ARPA_Lombardia_AQ_registry()
    regAQ <- regAQ %>% filter(Pollutant %in% c("PM10","Ammonia"))
    regW <- get_ARPA_Lombardia_W_registry()
    registry_KNN_dist(regAQ,regW,k=2)
}
```

---

# Time_aggregate *Aggregate any ARPALdf object (with higher temporal frequency) to hourly, daily, weekly, monthly and yearly temporal frequencies.*

---

## Description

Starting from an ARPALdf object with high frequency (e.g., 10mins or hourly), 'Time_aggregate' aggregates the dataset to lower temporal frequencies (e.g., hourly, daily, weekly, monthly and yearly) by station. The output is an ARPALdf object with observations having hourly, daily, weekly, monthly or yearly frequency. The function can be applied only to ARPALdf objects. User can indicate specific variables to aggregate and an aggregation function among mean, median, sum

(cumulated), min, max, quantiles, and variability metrics for each variable. It is possible to specify different aggregation functions on the same variable by repeating the name of the variable in 'Var_vec' and specifying the functions in 'Fns_vec'.

## Usage

```
Time_aggregate(Dataset, Frequency, Var_vec = NULL, Fns_vec = NULL, verbose = T)
```

## Arguments

| | |
|---|---|
| Dataset | ARPALdf dataframe to aggregate. |
| Frequency | Temporal aggregation frequency. It can be "hourly", "daily", "weekly", "monthly" or "yearly. |
| Var_vec | Vector of variables to aggregate. If NULL (default) all the variables are averaged, expect for 'Temperature' and 'Snow_height' which are summed. |
| Fns_vec | Vector of aggregation functions to apply to the selected variables. Available functions are 'mean', 'median', 'min', 'max', 'sum', 'qPP' (PP-th percentile), 'sd', 'var', 'vc' (variability coefficient), 'skew' (skewness) and 'kurt' (kurtosis). Attention: for Wind Speed and Wind Speed Gust only mean, min and max are available; for Wind Direction and Wind Direction Gust only mean is available. |
| verbose | Logic value (TRUE or FALSE). Toggle warnings and messages. If 'verbose=T' (default) the function prints on the screen some messages describing the progress of the tasks. If 'verbose=F' any message about the progression is suppressed. |

## Value

A data frame

## Examples

```
## Download hourly observed concentrations during 2020 for station 501 (Milano - Via Marche).
if (require("RSocrata")) {
    data <- get_ARPA_Lombardia_AQ_data(ID_station=501, Date_begin = "2020-01-01",
                                       Date_end = "2020-12-31", Frequency="hourly")
}
## Aggregate all the data to daily frequency
Time_aggregate(Dataset=data,Frequency="daily",Var_vec=NULL,Fns_vec=NULL)
## Aggregate NO2 to weekly maximum concentrations and NOx to weekly minimum concentrations.
Time_aggregate(Dataset=data,Frequency="weekly",Var_vec=c("NO2","NOx"),Fns_vec=c("max","min"))
```

# Index