Package ‘climaemet’

October 12, 2022

Title Climate AEMET Tools

Version 1.0.2

Description Tools to download the climatic data of the Spanish Meteorological Agency (AEMET) directly from R using their API and create scientific graphs (climate charts, trend analysis of climate time series, temperature and precipitation anomalies maps, warming stripes graphics, climatograms, etc.).

License GPL-3


BugReports https://github.com/rOpenSpain/climaemet/issues

Depends R (>= 3.6.0)

Imports dplyr (>= 1.0.0), ggplot2 (>= 3.3.2), httr (>= 1.4.1), jsonlite (>= 1.7.0), rappdirs (>= 0.3.3), readr (>= 1.4.0), rlang (>= 0.4.6), tibble (>= 3.0.3), tidyr (>= 1.1.0)

Suggests climatol (>= 3.1.2), gganimate (>= 1.0.5), jpeg (>= 0.1.8), knitr, rmarkdown, sf (>= 0.9.0)

VignetteBuilder knitr

Copyright © AEMET. See file COPYRIGHTS

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

X-schema.org-applicationCategory Meteorology

X-schema.org-isPartOf https://ropenspain.es/

NeedsCompilation no

Author Manuel Pizarro [aut, cph] (<https://orcid.org/0000-0002-6981-0154>), Diego Hernangómez [aut, cre] (<https://orcid.org/0000-0001-8457-4658>, rOpenSpain), Gema Fernández-Avilés [aut] (<https://orcid.org/0000-0001-5934-1916>)

Maintainer Diego Hernangómez <diego.hernangomezherrero@gmail.com>
Description

This function will store your AEMET API key on your local machine so it can be called securely without being stored in your code. After you have installed your key, it can be called any time by typing `Sys.getenv("AEMET_API_KEY")` and can be used in package functions by simply typing `AEMET_API_KEY`.

Alternatively, you can install the API Key manually:

- Run `Sys.setenv(AEMET_API_KEY = "Your_Key")`. You would need to run this command on each session (Similar to `install = FALSE`).
- Write this line on your .Renviron file: `AEMET_API_KEY = "Your_Key"` (same behavior than `install = TRUE`). This would store your API key permanently.
aemet_api_key

Usage

aemet_api_key(apikey, overwrite = FALSE, install = FALSE)

Arguments

apikey       The API key provided to you from the AEMET formatted in quotes. A key can be acquired at https://opendata.aemet.es/centrodedescargas/inicio.
overwrite    If this is set to TRUE, it will overwrite an existing AEMET_API_KEY that you already have in local machine.
install      if TRUE, will install the key in your local machine for use in future sessions. Defaults to FALSE.

Value

None

Note

To locate your API Key on your local machine, run rappdirs::user_cache_dir("climaemet", "R").

See Also

Other aemet_auth: aemet_detect_api_key()

Examples

# Don't run these examples!

if (FALSE) {
  aemet_api_key("11111abc", install = TRUE)
  # You can check it with:
  Sys.getenv("AEMET_API_KEY")
}

if (FALSE) {
  # If you need to overwrite an existing key:
  aemet_api_key("22222abc", overwrite = TRUE, install = TRUE)
  # You can check it with:
  Sys.getenv("AEMET_API_KEY")
}
Description

Get climatology values for a station or for all the available stations. Note that aemet_daily_period() and aemet_daily_period_all() are shortcuts of aemet_daily_clim().

Usage

```r
aemet_daily_clim(
  station = "all",
  start = Sys.Date() - 7,
  end = Sys.Date(),
  verbose = FALSE,
  return_sf = FALSE
)
```

```r
aemet_daily_period(
  station, 
  start = 2020,
  end = 2020,
  verbose = FALSE,
  return_sf = FALSE
)
```

```r
aemet_daily_period_all(
  start = 2020,
  end = 2020,
  verbose = FALSE,
  return_sf = FALSE
)
```

Arguments

- **station**: Character string with station identifier code(s) (see `aemet_stations()`) or "all" for all the stations.
- **start, end**: Character string with start and end date. See Details.
- **verbose**: Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
- **return_sf**: Logical TRUE or FALSE. Should the function return an sf spatial object? If FALSE (the default value) it returns a tibble. Note that you need to have the sf package installed.
Details

start and end parameters should be:

- For `aemet_daily_clim()`: A Date object or a string with format: YYYY-MM-DD (2020-12-31) coercible with `as.Date()`.
- For `aemet_daily_period()` and `aemet_daily_period_all()`: A string representing the year(s) to be extracted: "2020", "2018".

Value

A tibble or a sf object

API Key

You need to set your API Key globally using `aemet_api_key()`.

See Also

`aemet_api_key()`, `as.Date()`

Other aemet_api_data: `aemet_extremes_clim()`, `aemet_last_obs()`, `aemet_monthly`, `aemet_normal`, `aemet_stations()`

Examples

```r
library(tibble)
obsv <- aemet_daily_clim(c("9434", "3195"))
glimpse(obs)
```

aemet_detect_api_key

Check if an AEMET API Key is present for the current session

Description

The function would detect if an API Key is available on this session:

- If an API Key is already set as an environment variable it would be preserved
- If no environment variable has been set and you have stored permanently an API Key using `aemet_api_key()`, the latter would be loaded.

Usage

`aemet_detect_api_key(...)`
Arguments

... Ignored

Value

TRUE or FALSE

See Also

Other aemet_auth: aemet_api_key()

Examples

aemet_detect_api_key()

aemet_extremes_clim

Extreme values for a station

Description

Get recorded extreme values for a station.

Usage

aemet_extremes_clim(
  station = NULL,
  parameter = "T",
  verbose = FALSE,
  return_sf = FALSE
)

Arguments

station Character string with station identifier code(s) (see aemet_stations())
parameter Character string as temperature ("T"), precipitation ("P") or wind ("V") parameter.
verbose Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
return_sf Logical TRUE or FALSE. Should the function return an sf spatial object? If FALSE (the default value) it returns a tibble. Note that you need to have the sf package installed.

Value

A tibble or a sf object
API Key

You need to set your API Key globally using `aemet_api_key()`.

See Also

- `aemet_api_key()`
- Other `aemet_api_data`: `aemet_daily_clim()`, `aemet_last_obs()`, `aemet_monthly`, `aemet_normal`, `aemet_stations()`

Examples

```r
library(tibble)
obs <- aemet_extremes_clim(c("9434", "3195"))
glimpse(obs)
```

---

**aemet_last_obs**

*Last observation values for a station*

Description

Get last observation values for a station.

Usage

```r
aemet_last_obs(station = "all", verbose = FALSE, return_sf = FALSE)
```

Arguments

- `station` Character string with station identifier code(s) (see `aemet_stations()`) or "all" for all the stations.
- `verbose` Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
- `return_sf` Logical TRUE or FALSE. Should the function return an sf spatial object? If FALSE (the default value) it returns a tibble. Note that you need to have the sf package installed.

Value

A tibble or a sf object.

API Key

You need to set your API Key globally using `aemet_api_key()`.
See Also

Other aemet_api_data: aemet_daily_clim(), aemet_extremes_clim(), aemet_monthly, aemet_normal, aemet_stations()

Examples

```r
library(tibble)
obs <- aemet_last_obs(c("9434", "3195"))
glimpse(obs)
```

## aemet_monthly

### Monthly/annual climatology

#### Description

Get monthly/annual climatology values for a station or all the stations.

#### Usage

```r
aemet_monthly_clim(
  station = NULL,  
  year = 2020,  
  verbose = FALSE, 
  return_sf = FALSE
)
```

```r
aemet_monthly_period(
  station = NULL,  
  start = 2018,  
  end = 2020,  
  verbose = FALSE, 
  return_sf = FALSE
)
```

```r
aemet_monthly_period_all(  
  start = 2019,  
  end = 2020,  
  verbose = FALSE, 
  return_sf = FALSE
)
```
aemet_normal

Arguments

station
Character string with station identifier code(s) (see aemet_stations()).

year
Numeric value as date (format: YYYY).

verbose
Logical TRUE/FALSE. Provides information about the flow of information between the client and server.

return_sf
Logical TRUE or FALSE. Should the function return an sf spatial object? If FALSE (the default value) it returns a tibble. Note that you need to have the sf package installed.

start
Numeric value as start year (format: YYYY).

date
Numeric value as end year (format: YYYY).

Value

A tibble or a sf object

API Key

You need to set your API Key globally using aemet_api_key().

See Also

Other aemet_api_data: aemet_daily_clim(), aemet_extremes_clim(), aemet_last_obs(), aemet_normal, aemet_stations()

Examples

library(tibble)
obs <- aemet_monthly_clim(station = c("9434", "3195"), year = 2000)
glimpse(obs)

aemet_normal

Normal climatology values

Description

Get normal climatology values for a station (or all the stations with aemet_normal_clim_all()). Standard climatology from 1981 to 2010.

Usage

aemet_normal_clim(station = NULL, verbose = FALSE, return_sf = FALSE)
aemet_normal_clim_all(verbose = FALSE, return_sf = FALSE)
Arguments

- **station**: Character string with station identifier code(s) (see `aemet_stations()`) or "all" for all the stations.
- **verbose**: Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
- **return_sf**: Logical TRUE or FALSE. Should the function return an sf spatial object? If FALSE (the default value) it returns a tibble. Note that you need to have the sf package installed.

Value

A tibble or a sf object.

API Key

You need to set your API Key globally using `aemet_api_key()`.

Note

Code modified from project https://github.com/SevillaR/aemet

See Also

Other `aemet_api_data`: `aemet_daily_clim()`, `aemet_extremes_clim()`, `aemet_last_obs()`, `aemet_monthly`, `aemet_stations()`

Examples

```r
library(tibble)
obs <- aemet_normal_clim(c("9434", "3195"))
glimpse(obs)
```

---

### `aemet_stations`  

**AEMET stations**

**Description**

Get AEMET stations.

**Usage**

```r
aemet_stations(verbos = FALSE, return_sf = FALSE)
```
Arguments

verbose Logical TRUE/FALSE. Provides information about the flow of information between the client and server.

return_sf Logical TRUE or FALSE. Should the function return an sf spatial object? If FALSE (the default value) it returns a tibble. Note that you need to have the sf package installed.

Value

A tibble or a sf object

API Key

You need to set your API Key globally using aemet_api_key().

Note

Code modified from project https://github.com/SevillaR/aemet

See Also

Other aemet_api_data: aemet_daily_clim(), aemet_extremes_clim(), aemet_last_obs(), aemet_monthly, aemet_normal

Examples

library(tibble)
stations <- aemet_stations()
stations

climaemnet_9434_climatogram

Climatogram data for Zaragoza Airport ("9434") period 1981-2010

Description

Normal data for Zaragoza Airport (1981-2010). This is an example dataset used to plot climatograms.

Format

A data.frame with columns 1 to 12 (months) and rows:

- p_mes_md: Precipitation (mm).
- tm_max_md: Maximum temperature (Celsius).
- tm_min_md: Minimum temperature (Celsius).
- ta_min_md: Absolute monthly minimum temperature (Celsius).
climaemet_9434_temp

Source

AEMET.

See Also

`ggclimat_walter_lieth(), climatogram_period(), climatogram_normal()`

Other dataset: `climaemet_9434_temp, climaemet_9434_wind`

Other climatogram: `climatogram_normal(), climatogram_period(), ggclimat_walter_lieth()`

Examples

data(climaemet_9434_climatogram)

climaemet_9434_temp  Average annual temperatures for Zaragoza Airport ("9434") period 1950-2020

Description

Yearly observations of average temperature for Zaragoza Airport (1950-2020). This is an example dataset.

Format

A tibble with columns:

- **year**: Year of reference.
- **indicativo**: Identifier of the station.
- **temp**: Avg temperature (Celsius).

Source

AEMET.

See Also

Other dataset: `climaemet_9434_climatogram, climaemet_9434_wind`

Other stripes: `climatestripes_station(), ggstripes()`

Examples

data(climaemet_9434_temp)
climaemet_9434_wind  Wind conditions for Zaragoza Airport ("9434") period 2000-2020

Description

Daily observations of wind speed and directions for Zaragoza Airport (2000-2020). This is an example dataset.

Format

A tibble with columns:

- **fecha**: Date of observation.
- **dir**: Wind directions (0-360).
- **velmedia**: Avg wind speed (km/h).

Source

AEMET.

See Also

Other dataset: climaemet_9434_climatogram, climaemet_9434_temp
Other wind: ggwindrose(), windrose_days(), windrose_period()

Examples

data(climaemet_9434_wind)

climaemet_news  climaemet_news

Description

Show the NEWS file of the climaemet package.

Usage

climaemet_news()

Details

(See description)
Value

Open NEWS from climaemet.

See Also

Other helpers: `dms2decdegrees()`, `first_day_of_year()`

Examples

```r
## Not run:
climaemet_news()
## End(Not run)
```

---

**climatestripes_station**

*Station climate stripes graph*

Description

Plot climate stripes graph for a station

Usage

```r
climaemet_station(
  station,
  start = 1950,
  end = 2020,
  with_labels = "yes",
  verbose = FALSE,
  ...
)
```

Arguments

- **station**: Character string with station identifier code(s) (see `aemet_stations()`)
- **start**: Numeric value as start year (format: YYYY).
- **end**: Numeric value as end year (format: YYYY).
- **with_labels**: Character string as yes/no. Indicates whether to use labels for the graph or not.
- **verbose**: Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
- **...**: Arguments passed on to `ggstripes`
- **n_temp**: Numeric value as the number of colors of the palette. (default 11).
- **col_pal**: Character string indicating the name of the `hcl.pals()` color palette to be used for plotting.
climatogram_normal

Value

A ggplot2 object

API Key

You need to set your API Key globally using `aemet_api_key()`.

See Also

- `ggstripes()`
- Other aemet_plots: `climatogram_normal()`, `climatogram_period()`, `ggclimat_walter_lieth()`, `ggstripes()`, `ggwindrose()`, `windrose_days()`, `windrose_period()`
- Other stripes: `climaemet_9434_temp`, `ggstripes()`

Examples

```r
climatogram_normal(  
  station,  
  labels = "en",  
  verbose = FALSE,  
  ggplot2 = TRUE,  
  ...  
)
```

climatogram_normal  Walter & Lieth climatic diagram from normal climatology values

Description

Plot of a Walter & Lieth climatic diagram from normal climatology data for a station. This climatogram are great for showing a summary of climate conditions for a place over a time period (1981-2010).

Usage

```r
climatogram_normal(  
  station,  
  labels = "en",  
  verbose = FALSE,  
  ggplot2 = TRUE,  
  ...  
)
```
climatogram_normal

Arguments

- **station**: Character string with station identifier code(s) (see `aemet_stations()`)
- **labels**: Character string as month labels for the X axis: "en" (english), "es" (spansih), "fr" (french), etc.
- **verbose**: Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
- **ggplot2**: TRUE/FALSE. On TRUE the function uses `ggclimat_walter_lieth()`, if FALSE uses `climatol::diagwl()`.
- **...**: Further arguments to `climatol::diagwl()` or `ggclimat_walter_lieth()`, depending on the value of `ggplot2`

Value

A plot.

API Key

You need to set your API Key globally using `aemet_api_key()`.

Note

The code is based on code from the CRAN package "climatol" by Jose A. Guijarro jguijarrop@aemet.es.

References


See Also

Other aemet_plots: `climatestripes_station()`, `climatogram_period()`, `ggclimat_walter_lieth()`, `ggstripes()`, `ggwindrose()`, `windrose_days()`, `windrose_period()`

Other climatogram: `climaemet_9434_climatogram`, `climatogram_period()`, `ggclimat_walter_lieth()`

Examples

climatogram_normal("9434")
climatogram_period

Walter & Lieth climatic diagram for a time period

Description
Plot of a Walter & Lieth climatic diagram from monthly climatology data for a station. This climatogram are great for showing a summary of climate conditions for a place over a specific time period.

Usage
climatogram_period(
    station = NULL,
    start = 1990,
    end = 2020,
    labels = "en",
    verbose = FALSE,
    ggplot2 = TRUE,
    ...
)

Arguments
station Character string with station identifier code(s) (see aemet_stations())
start Numeric value as start year (format: YYYY).
end Numeric value as end year (format: YYYY).
labels Character string as month labels for the X axis: "en" (english), "es" (spanish), "fr" (french), etc.
verbose Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
ggplot2 TRUE/FALSE. On TRUE the function uses ggclimat_walter_lieth(), if FALSE uses climatol::diagwl().
...
Further arguments to climatol::diagwl() or ggclimat_walter_lieth(), depending on the value of ggplot2

Value
A plot.

API Key
You need to set your API Key globally using aemet_api_key().

Note
The code is based on code from the CRAN package "climatol" by Jose A. Guijarro jguijarrop@aemet.es.
dms2decdegrees

References

See Also
Other aemet_plots: climatestripes_station(), climatogram_normal(), ggclimat_walter_lieth(),
ggstripes(), ggwindrose(), windrose_days(), windrose_period()
Other climatogram: climaemet_9434_climatogram, climatogram_normal(), ggclimat_walter_lieth()

Examples
climatogram_period("9434", start = 2015, end = 2020, labels = "en")

dms2decdegrees

Converts dms to decimal degrees

Description
Converts degrees, minutes and seconds to decimal degrees.

Usage
dms2decdegrees(input = NULL)

Arguments
input Character string as DMS coordinates.

Value
A numeric value.

Note
Code modified from project https://github.com/SevillaR/aemet

See Also
Other helpers: climaemet_news(), first_day_of_year()

Examples
dms2decdegrees("055245W")
first_day_of_year

Description

Get first and last day of year.

Usage

first_day_of_year(year = NULL)

last_day_of_year(year = NULL)

Arguments

year

Numeric value as year (format: YYYY).

Value

Character string as date (format: YYYY-MM-DD).

See Also

Other helpers: climaemet_news(), dms2decdegrees()

Examples

first_day_of_year(2000)

last_day_of_year(2020)

get_data_aemet

Client tool for AEMET API

Description

Client tool to get data and metadata from AEMET and convert json to tibble.

Usage

get_data_aemet(apidest, verbose = FALSE)

get_metadata_aemet(apidest, verbose = FALSE)
Arguments

apidest  Character string as destination URL. See https://opendata.aemet.es/dist/index.html.
verbose Logical TRUE/FALSE. Provides information about the flow of information between the client and server.

Value

A tibble or an empty tibble if no valid results from the API.

Source

https://opendata.aemet.es/dist/index.html

Examples

# Run this example only if AEMET_API_KEY is detected
url <- "/api/valores/climatologicos/inventarioestaciones/todasestaciones"
get_data_aemet(url)

# Metadata
get_metadata_aemet(url)

ggclimat_walter_lieth  Walter and Lieth climatic diagram on ggplot2

Description

Plot of a Walter and Lieth climatic diagram of a station. This function is an updated version of climatol::diagwl(), by Jose A. Guijarro.

Usage

ggclimat_walter_lieth(
  dat,
  est = "",
  alt = NA,
  per = NA,
  mlab = "es",
  pcol = "#002F70",
  tcol = "#FF0000",
  pfcol = "#9BAEE2",
)
sfcol = ",3C6FC4",
shem = FALSE,
p3line = FALSE,
...)

Arguments

dat Monthly climatic data for which the diagram will be plotted.
est Name of the climatological station
alt Altitude of the climatological station
per Period on which the averages have been computed
mlab Month labels for the X axis. Use 2-digit language code ("en", "es", etc.). See readr::locale() for info.
pcol Color pen for precipitation.
tcol Color pen for temperature.
pfcol Fill color for probable frosts.
sfcol Fill color for sure frosts.
shem Set to TRUE for southern hemisphere stations.
p3line Set to TRUE to draw a supplementary precipitation line referenced to three times the temperature (as suggested by Bogdan Rosca).
... Other graphic parameters

Details

See Details on climatol::diagwl().

Climatic data must be passed as a 4x12 matrix of monthly (January to December) data, in the following order:

- Row 1: Mean precipitation.
- Row 2: Mean maximum daily temperature.
- Row 3: Mean minimum daily temperature.
- Row 4: Absolute monthly minimum temperature.

See climaemet_9434_climatogram for a sample dataset.

Value

A ggplot2 object. See help("ggplot2").

API Key

You need to set your API Key globally using aemet_api_key().
Warming stripes graph

Description

Plot different "climate stripes" or "warming stripes" using `ggplot2`. This graphics are visual representations of the change in temperature as measured in each location over the past 70-100+ years. Each stripe represents the temperature in that station averaged over a year.

References


See Also

- `climatol::diagwl()`, `readr::locale()`
- Other aemet_plots: `climatestripes_station()`, `climatogram_normal()`, `climatogram_period()`, `ggstripes()`, `ggwindrose()`, `windrose_days()`, `windrose_period()`
- Other climatogram: `climaemet_9434_climatogram`, `climatogram_normal()`, `climatogram_period()`

Examples

```r
library(ggplot2)

wl <- ggclimat_walter_lieth(
  climaemet::climaemet_9434_climatogram,
  alt = "249",
  per = "1981-2010",
  est = "Zaragoza Airport"
)

# As it is a ggplot object we can modify it

wl + theme(
  plot.background = element_rect(fill = "grey80"),
  panel.background = element_rect(fill = "grey70"),
  axis.text.y.left = element_text(
    colour = "black",
    face = "italic"
  ),
  axis.text.y.right = element_text(
    colour = "black",
    face = "bold"
  )
)
```
Usage

ggstripes(
  data,
  plot_type = "stripes",
  plot_title = "",
  n_temp = 11,
  col_pal = "RdBu",
  ...
)

Arguments

data a data.frame with date(year) and temperature(temp) variables.
plot_type plot type (with labels, background, stripes with line trend and animation). Accepted values are "background", "stripes", "trend" or "animation".
plot_title character string to be used for the graph title.
n_temp Numeric value as the number of colors of the palette. (default 11).
col_pal Character string indicating the name of the hcl.pals() color palette to be used for plotting.
... further arguments passed to ggplot2::theme().

Value

A ggplot2 object.

API Key

You need to set your API Key globally using aemet_api_key().

Note

"Warming stripes" charts are a conceptual idea of Professor Ed Hawkins (University of Reading) and are specifically designed to be as simple as possible and alert about risks of climate change. For more details see ShowYourStripes.

See Also

climatestripes_station(), ggplot2::theme() for more possible arguments to pass to ggstripes.

Other aemet_plots: climatestripes_station(), climatogram_normal(), climatogram_period(), ggclimat_walter_lieth(), ggwindrose(), windrose_days(), windrose_period()

Other stripes: climaemet_9434_temp, climatestripes_station()
Examples

library(ggplot2)

data <- climaemet::climaemet_9434_temp

ggstripes(data, plot_title = "Zaragoza Airport") +
  labs(subtitle = "(1950-2020)")

ggstripes(data, plot_title = "Zaragoza Airport", plot_type = "trend") +
  labs(subtitle = "(1950-2020)")


ggwindrose

Windrose (speed/direction) diagram

Description

Plot a windrose showing the wind speed and direction using \texttt{ggplot2}.

Usage

\texttt{ggwindrose(speed, direction, n_directions = 8, n_speeds = 5, speed_cuts = NA, col_pal = "GnBu", legend_title = "Wind speed (m/s)", calm_wind = 0, n_col = 1, facet = NULL, plot_title = "", ... )}

Arguments

- \texttt{speed} \hspace{1cm} Numeric vector of wind speeds.
- \texttt{direction} \hspace{1cm} Numeric vector of wind directions.
- \texttt{n_directions} \hspace{1cm} Numeric value as the number of direction bins to plot (petals on the rose). The number of directions defaults to 8.
- \texttt{n_speeds} \hspace{1cm} Numeric value as the number of equally spaced wind speed bins to plot. This is used if \texttt{speed_cuts} is NA (default 5).
- \texttt{speed_cuts} \hspace{1cm} Numeric vector containing the cut points for the wind speed intervals, or NA (default).
`ggwindrose`  

- **col_pal**: Character string indicating the name of the `hcl.pals()` color palette to be used for plotting.
- **legend_title**: Character string to be used for the legend title.
- **calm_wind**: Numeric value as the upper limit for wind speed that is considered calm (default 0).
- **n_col**: The number of columns of plots (default 1).
- **facet**: Character or factor vector of the facets used to plot the various windroses.
- **plot_title**: Character string to be used for the plot title.
- **...**: further arguments (ignored).

**Value**

A ggplot object.

**API Key**

You need to set your API Key globally using `aemet_api_key()`.

**See Also**

- `ggplot2::theme()` for more possible arguments to pass to `ggwindrose`.
- Other aemet_plots: `climatestripes_station()`, `climatogram_normal()`, `climatogram_period()`, `ggclimat_walter_lieth()`, `ggstripes()`, `windrose_days()`, `windrose_period()`
- Other wind: `climaemet_9434_wind`, `windrose_days()`, `windrose_period()`

**Examples**

```r
library(ggplot2)

speed <- climaemet::climaemet_9434_wind$velmedia
direction <- climaemet::climaemet_9434_wind$dir

rose <- ggwindrose(
  speed = speed,
  direction = direction,
  speed_cuts = seq(0, 16, 4),
  legend_title = "Wind speed (m/s)",
  calm_wind = 0,
  n_col = 1,
  plot_title = "Zaragoza Airport"
)
rose + labs(
  subtitle = "2000-2020",
  caption = "Source: AEMET"
)
```
Description

Plot a windrose showing the wind speed and direction for a station over a days period.

Usage

```r
windrose_days(
  station,
  start = "2000-12-01",
  end = "2000-12-31",
  n_directions = 8,
  n_speeds = 5,
  speed_cuts = NA,
  col_pal = "GnBu",
  calm_wind = 0,
  legend_title = "Wind Speed (m/s)",
  verbose = FALSE
)
```

Arguments

- **station**: Character string with station identifier code(s) (see `aemet_stations()`) or "all" for all the stations.
- **start**: Character string as start date (format: YYYY-MM-DD).
- **end**: Character string as end date (format: YYYY-MM-DD).
- **n_directions**: Numeric value as the number of direction bins to plot (petals on the rose). The number of directions defaults to 8.
- **n_speeds**: Numeric value as the number of equally spaced wind speed bins to plot. This is used if `speed_cuts` is `NA` (default 5).
- **speed_cuts**: Numeric vector containing the cut points for the wind speed intervals, or `NA` (default).
- **col_pal**: Character string indicating the name of the `hcl.pals()` color palette to be used for plotting.
- **calm_wind**: Numeric value as the upper limit for wind speed that is considered calm (default 0).
- **legend_title**: Character string to be used for the legend title.
- **verbose**: Logical `TRUE/FALSE`. Provides information about the flow of information between the client and server.

Value

A `ggplot2` object
windrose_period

API Key

You need to set your API Key globally using `aemet_api_key()`.

See Also

- `aemet_daily_clim()`
- Other aemet_plots: `climatestripes_station()`, `climatogram_normal()`, `climatogram_period()`, `ggclimat_walter_lieth()`, `ggstripes()`, `ggwindrose()`, `windrose_period()`
- Other wind: `climaemet_9434_wind`, `ggwindrose()`, `windrose_period()`

Examples

```r
windrose_days("9434",
    start = "2000-12-01",
    end = "2000-12-31",
    speed_cuts = 4
)
```

---

`windrose_period`  Windrose (speed/direction) diagram of a station over a time period

Description

Plot a windrose showing the wind speed and direction for a station over a time period.

Usage

```r
windrose_period(
    station,
    start = 2000,
    end = 2010,
    n_directions = 8,
    n_speeds = 5,
    speed_cuts = NA,
    col_pal = "GnBu",
    calm_wind = 0,
    legend_title = "Wind Speed (m/s)",
    verbose = FALSE
)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>station</td>
<td>Character string with station identifier code(s) (see <code>aemet_stations()</code>) or &quot;all&quot; for all the stations.</td>
</tr>
<tr>
<td>start</td>
<td>Numeric value as start year (format: YYYY).</td>
</tr>
<tr>
<td>end</td>
<td>Numeric value as end year (format: YYYY).</td>
</tr>
<tr>
<td>n_directions</td>
<td>Numeric value as the number of direction bins to plot (petals on the rose). The number of directions defaults to 8.</td>
</tr>
<tr>
<td>n_speeds</td>
<td>Numeric value as the number of equally spaced wind speed bins to plot. This is used if <code>speed_cuts</code> is NA (default 5).</td>
</tr>
<tr>
<td>speed_cuts</td>
<td>Numeric vector containing the cut points for the wind speed intervals, or NA (default).</td>
</tr>
<tr>
<td>col_pal</td>
<td>Character string indicating the name of the <code>hcl.pals()</code> color palette to be used for plotting.</td>
</tr>
<tr>
<td>calm_wind</td>
<td>Numeric value as the upper limit for wind speed that is considered calm (default 0).</td>
</tr>
<tr>
<td>legend_title</td>
<td>Character string to be used for the legend title.</td>
</tr>
<tr>
<td>verbose</td>
<td>Logical TRUE/FALSE. Provides information about the flow of information between the client and server.</td>
</tr>
</tbody>
</table>

Value

A ggplot2 object

API Key

You need to set your API Key globally using `aemet_api_key()`.

See Also

- `aemet_daily_period()`
- Other `aemet_plots`: `climatestripes_station()`, `climatogram_normal()`, `climatogram_period()`, `ggclimat_walter_lieth()`, `ggstripes()`, `ggwindrose()`, `windrose_days()`
- Other wind: `climaemet_9434_wind`, `ggwindrose()`, `windrose_days()`

Examples

```r
windrose_period("9434",
    start = 2000, end = 2010,
    speed_cuts = 4
)
```
Index

* aemet_api_data
  aemet_daily_clim, 4
  aemet_extremes_clim, 6
  aemet_last_obs, 7
  aemet_monthly, 8
  aemet_normal, 9
  aemet_stations, 10

* aemet_api
  get_data_aemet, 19

* aemet_author
  aemet_api_key, 2
  aemet_detect_api_key, 5

* aemet_plots
  climatestripes_station, 14
  climatogram_normal, 15
  climatogram_period, 17
  ggclimat_walter_lieth, 20
  ggstripes, 22
  ggwindrose, 24
  windrose_days, 26
  windrose_period, 27

* climatogram
  climaemet_9434_climatogram, 11
  climatogram_normal, 15
  climatogram_period, 17
  ggclimat_walter_lieth, 20

* dataset
  climaemet_9434_climatogram, 11
  climaelemet_9434_temp, 12
  climaelemet_9434_wind, 13

* helpers
  climaemet_news, 13
  dms2decdegrees, 18
  first_day_of_year, 19

* stripes
  climaelemet_9434_temp, 12
  climatestripes_station, 14
  ggstripes, 22

* wind
  climaelemet_9434_wind, 13
  ggwindrose, 24
  windrose_days, 26
  windrose_period, 27

  aemet_api_key, 2, 6
  aemet_api_key(), 5, 7, 9–11, 15–17, 21, 23, 25, 27, 28
  aemet_daily (aemet_daily_clim), 4
  aemet_daily_clim, 4, 7–11
  aemet_daily_clim(), 27
  aemet_daily_period (aemet_daily_clim), 4
  aemet_daily_period(), 28
  aemet_daily_period_all
    (aemet_daily_clim), 4
  aemet_detect_api_key, 3, 5
  aemet_extremes_clim, 5, 6, 8–11
  aemet_last_obs, 5, 7, 7, 9–11
  aemet_monthly, 5, 7, 8, 8, 10, 11
  aemet_monthly_clim (aemet_monthly), 8
  aemet_monthly_period (aemet_monthly), 8
  aemet_monthly_period_all
    (aemet_monthly), 8
  aemet_normal, 5, 7–9, 9, 11
  aemet_normal_clim (aemet_normal), 9
  aemet_normal_clim_all (aemet_normal), 9
  aemet_stations, 5, 7–10, 10
  aemet_stations(), 4, 6, 7, 9, 10, 14, 16, 17, 26, 28
  as.Date(), 5

  climaemet_9434_climatogram, 11, 12, 13, 16, 18, 21, 22
  climaemet_9434_temp, 12, 12, 13, 15, 23
  climaemet_9434_wind, 12, 13, 25, 27, 28
  climaemet_news, 13, 18, 19
  climatestripes_station, 12, 14, 16, 18, 22, 23, 25, 27, 28
  climatestripes_station(), 23
climatogram_normal, 12, 15, 15, 18, 22, 23, 25, 27, 28
climatogram_normal(), 12
climatogram_period, 12, 15, 15, 16, 17, 22, 23, 25, 27, 28
climatogram_period(), 12
climatol::diagwl(), 16, 17, 20–22
dms2decdegrees, 14, 18, 19
first_day_of_year, 14, 18, 19
get_data_aemet. 19
get_metadata_aemet (get_data_aemet), 19
ggclimat_walter_lieth, 12, 15, 16, 18, 20, 23, 25, 27, 28
ggclimat_walter_lieth(), 12, 16, 17
ggplot2::theme(), 23, 25
ggstripes, 12, 14–16, 18, 22, 22, 25, 27, 28
ggstripes(), 15
ggwindrose, 13, 15, 16, 18, 22, 23, 24, 27, 28
hcl.pals(), 14, 23, 25, 26, 28
last_day_of_year (first_day_of_year), 19
readr::locale(), 21, 22
windrose_days, 13, 15, 16, 18, 22, 23, 25, 26, 28
windrose_period, 13, 15, 16, 18, 22, 23, 25, 27, 27