Package 'climaemet'

April 1, 2021

Title Climate AEMET Tools

Version 1.0.0.9000

Description Tools to download the climatic data of the Spanish Meteorological Agency (AEMET) directly from R using their API <https://opendata.aemet.es/> 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

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Depends R (>= 3.6.0)

Imports tibble (>= 3.0.3), dplyr (>= 1.0.0), tidyr (>= 1.1.0), readr (>= 1.4.0), lubridate (>= 1.7.9), httr (>= 1.4.1), jsonlite (>= 1.7.0), ggplot2 (>= 3.3.2), rlang (>= 0.4.6)

Suggests sf (>= 0.9), climatol (>= 3.1.2), gganimate (>= 1.0.5), jpeg (>= 0.1.8)

URL https://ropenspain.github.io/climaemet/, https://github.com/rOpenSpain/climaemet

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

R topics documented:

aemet_api_key 2
aemet_daily_clim 3
aemet_detect_api_key 4

aemet_extremes_clim 5
aemet_last_obs 6
aemet_monthly_clim 7
aemet_normal_clim 8
aemet_stations 9
climaemet_9434_climatogram 10
climaemet_9434_temp 10
climaemet_9434_wind 11
climaemet_news 11
climatestripes_station 12
climatogram_normal 13
climatogram_period 14
dms2decdegrees 15
first_day_of_year 16
get_data_aemet 16
ggclimat_walter_lieth 17
ggstripes 19
ggwindrose 20
windrose_days 22
windrose_period 23

Index **25**

`aemet_api_key` *Install a AEMET API Key in Your* `.Renviron` *File for Repeated Use*

Description

This function will add your AEMET API key to your `.Renviron` file 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 If you do not have an `.Renviron` file, the function will create on for you. If you already have an `.Renviron` file, the function will append the key to your existing file, while making a backup of your original file for disaster recovery purposes.

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 your `.Renviron` file.
`install`	if TRUE, will install the key in your `.Renviron` file for use in future sessions. Defaults to FALSE.

Note

Code adapted from https://walker-data.com/tidycensus/reference/census_api_key.html

Examples

```
# Don't run these examples!

if (FALSE) {
  aemet_api_key("111111abc", install = TRUE)
  # First time, reload your environment o restart your session.
  readRenviron("~/.Renviron")
  # You can check it with:
  Sys.getenv("AEMET_API_KEY")
}

if (FALSE) {
  # If you need to overwrite an existing key:
  aemet_api_key("111111abc", overwrite = TRUE, install = TRUE)
  readRenviron("~/.Renviron")
  # You can check it with:
  Sys.getenv("AEMET_API_KEY")
}
```

---

aemet_daily_clim *Daily/annual climatology values*

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

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

aemet_daily_period(
  station,
  apikey = NULL,
  start = 2020,
  end = 2020,
  verbose = FALSE,
  return_sf = FALSE
)

aemet_daily_period_all(
  apikey = NULL,
  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.
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
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. 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: %Y%m%d (2020-12-31).
- 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

Examples

```
# Run this example only if AEMET_API_KEY is detected

if (aemet_detect_api_key()) {
  library(tibble)
  obs <- aemet_daily_clim(c("9434", "3195"))
  glimpse(obs)
}
```

---

aemet_detect_api_key *Check if an AEMET API Key is present as environment variable*

Description

The function returns TRUE/FALSE

Usage

```
aemet_detect_api_key(...)
```

Arguments

...	Ignored

`aemet_extremes_clim` *Extreme values for a station*

Description

Get recorded extreme values for a station.

Usage

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

Arguments

`station`	Character string with station identifier code(s) (see `aemet_stations()`)
`apikey`	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see `aemet_api_key()`.
`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. 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

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  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

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

Arguments

station	Character string with station identifier code(s) (see aemet_stations()) or "all" for all the stations.
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
verbose	Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
return_sf	Logical. 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.

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  library(tibble)
  obs <- aemet_last_obs(c("9434", "3195"))
  glimpse(obs)
}
```

`aemet_monthly_clim` *Monthly/annual climatology*

Description

Get monthly/annual climatology values for a station or all the stations. `aemet_monthly_period()` and `aemet_monthly_period_all()` allows requests that span several years.

Usage

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

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

aemet_monthly_period_all(
  apikey = NULL,
  start = 2019,
  end = 2020,
  verbose = FALSE,
  return_sf = FALSE
)
```

Arguments

`station`	Character string with station identifier code(s) (see `aemet_stations()`)
`apikey`	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see `aemet_api_key()`.
`year`	Numeric value as date (format: %Y).
`verbose`	Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
`return_sf`	Logical. 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: %Y).
`end`	Numeric value as end year (format: %Y).

Value

a tibble or a sf object

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  library(tibble)
  obs <- aemet_monthly_clim(station = c("9434", "3195"), year = 2000)
  glimpse(obs)
}
```

---

aemet_normal_clim *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.

Get normal climatology values for all stations.

Usage

```
aemet_normal_clim(
  station = NULL,
  apikey = NULL,
  verbose = FALSE,
  return_sf = FALSE
)

aemet_normal_clim_all(apikey = NULL, verbose = FALSE, return_sf = FALSE)
```

Arguments

station	Character string with station identifier code(s) (see aemet_stations()) or "all" for all the stations.
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
verbose	Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
return_sf	Logical. 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

Note

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

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  library(tibble)
  obs <- aemet_normal_clim(c("9434", "3195"))
  glimpse(obs)
}
```

`aemet_stations` *AEMET stations*

Description

Get AEMET stations.

Usage

```
aemet_stations(apikey = NULL, verbose = FALSE, return_sf = FALSE)
```

Arguments

`apikey`	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see `aemet_api_key()`.
`verbose`	Logical TRUE/FALSE. Provides information about the flow of information between the client and server.
`return_sf`	Logical. 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

Note

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

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  library(tibble)
  stations <- aemet_stations()
  stations
}
```

climaemet_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).

Source

AEMET.

See Also

`ggclimat_walter_lieth()`

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.

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.

climaemet_news *climaemet_news*

Description

Show the NEWS file of the **climaemet** package.

Usage

```
climaemet_news()
```

Details

(See description)

climatestripes_station *Station climate stripes graph*

Description

Plot climate stripes graph for a station

Usage

```
climatestripes_station(
  station,
  apikey = NULL,
  start = 1950,
  end = 2020,
  with_labels = "yes",
  verbose = FALSE,
  ...
)
```

Arguments

station	Character string as station identifier code (see aemet_stations()).
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
start	Numeric value as start year (format: %Y).
end	Numeric value as end year (format: %Y).
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.
...	Further arguments of ggstripes()

Value

a plot.

See Also

ggstripes()

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  climatestripes_station(
    "9434",
    start = 2010,
    end = 2020,
    with_labels = "yes",
```

```
    col_pal = "Inferno"
  )
}
```

`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

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

Arguments

`station`	Character string as station identifier code (see `aemet_stations()`).
`apikey`	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see `aemet_api_key()`.
`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()`, in FALSE uses `climatol::diagwl()`.
`...`	Further arguments to `climatol::diagwl()`/`ggclimat_walter_lieth()`

Value

a plot.

Note

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

References

Walter, H. & Lieth, H (1960): Klimadiagramm Weltatlas. G. Fischer, Jena.

See Also

ggclimat_walter_lieth()

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  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,
  apikey = NULL,
  start = 1990,
  end = 2020,
  labels = "en",
  verbose = FALSE,
  ggplot2 = TRUE,
  ...
)
```

Arguments

station	Character string as station identifier code (see aemet_stations()).
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
start	Numeric value as start year (format: %Y).
end	Numeric value as end year (format: %Y).
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(), in FALSE uses climatol::diagwl().
...	Further arguments to climatol::diagwl()/ggclimat_walter_lieth()

Value

a plot.

Note

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

References

Walter, H. & Lieth, H (1960): Klimadiagramm Weltatlas. G. Fischer, Jena.

See Also

`ggclimat_walter_lieth()`

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  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

Examples

```
dms2decdegrees("055245W")
```

first_day_of_year *First and last 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: %Y).

Value

Character string as date (format: %Y%m%d).

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, apikey = NULL, verbose = FALSE)

get_metadata_aemet(apidest, apikey = NULL, verbose = FALSE)
```

Arguments

apidest	Character string as destination URL. See https://opendata.aemet.es/dist/index.html.
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
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"

if (aemet_detect_api_key()) {
  get_data_aemet(url)
}

# Metadata
if (aemet_detect_api_key()) {
  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.

References

Walter, H., and Lieth, H. 1960. *Klimadiagramm-Weltatlas.* G. Fischer.

See Also

`climatol::diagwl()`, `readr::locale()`

Examples

```
library(ggplot2)

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

wl
```

```
# 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")
)
```

`ggstripes` *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.

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()` colour palette to be used for plotting, see **Palette selection**.
`...`	further arguments passed to `theme`.

Value

a `ggplot2` object

Palette selection

Any of the sequential `hcl.pals()` colour palettes are recommended for colour plots.

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

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

Examples

```
library(ggplot2)

data <- 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 **ggplot2**.

Usage

```
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

`speed`	Numeric vector of wind speeds.
`direction`	Numeric vector of wind directions.
`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() colour palette to be used for plotting, see **Palette selection**.
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.

Palette selection

Any of the sequential hcl.pals() colour palettes are recommended for colour plots.

See Also

ggplot2::theme() for more possible arguments to pass to ggwindrose.

Examples

```
library(ggplot2)

speed <- climaemet_9434_wind$velmedia
direction <- climaemet_9434_wind$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"
)
```

windrose_days *Windrose (speed/direction) diagram of a station over a days period*

Description

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

Usage

```
windrose_days(
  station,
  apikey = NULL,
  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.
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
start	Character string as start date (format: %Y%m%d).
end	Character string as end date (format: %Y%m%d).
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() colour palette to be used for plotting, see **Palette selection**.
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 plot.

See Also

ggwindrose(), aemet_daily_clim()

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  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

```
windrose_period(
  station,
  apikey = NULL,
  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

station	Character string with station identifier code(s) (see aemet_stations()) or "all" for all the stations.
apikey	Character string as personal API key. It can be set on the environment variable AEMET_API_KEY, see aemet_api_key().
start	Numeric value as start year (format: %Y).
end	Numeric value as end year (format: %Y).
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() colour palette to be used for plotting, see **Palette selection**.
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 plot.

See Also

ggwindrose(), windrose_days(), aemet_daily_period()

Examples

```
# Run this example only if AEMET_API_KEY is set

if (aemet_detect_api_key()) {
  windrose_period("9434",
    start = 2000, end = 2010,
    speed_cuts = 4
  )
}
```

Index

∗ **aemet_api_data**
  aemet_daily_clim, 3
  aemet_extremes_clim, 5
  aemet_last_obs, 6
  aemet_monthly_clim, 7
  aemet_normal_clim, 8
  aemet_stations, 9
∗ **aemet_api**
  get_data_aemet, 16
∗ **aemet_plots**
  climatestripes_station, 12
  climatogram_normal, 13
  climatogram_period, 14
  ggclimat_walter_lieth, 17
  ggstripes, 19
  ggwindrose, 20
  windrose_days, 22
  windrose_period, 23
∗ **dataset**
  climaemet_9434_climatogram, 10
  climaemet_9434_temp, 10
  climaemet_9434_wind, 11
∗ **helpers**
  aemet_api_key, 2
  aemet_detect_api_key, 4
  climaemet_news, 11
  dms2decdegrees, 15
  first_day_of_year, 16

aemet_api_key, 2
aemet_api_key(), *4–9*, *12–14*, *16*, *22, 23*
aemet_daily_clim, 3
aemet_daily_clim(), *23*
aemet_daily_period *(*aemet_daily_clim*)*, 3
aemet_daily_period(), *24*
aemet_daily_period_all *(*aemet_daily_clim*)*, 3
aemet_detect_api_key, 4
aemet_extremes_clim, 5
aemet_last_obs, 6
aemet_monthly_clim, 7
aemet_monthly_period *(*aemet_monthly_clim*)*, 7
aemet_monthly_period_all *(*aemet_monthly_clim*)*, 7
aemet_normal_clim, 8
aemet_normal_clim_all *(*aemet_normal_clim*)*, 8
aemet_stations, 9
aemet_stations(), *4–8*, *12–14*, *22, 23*

climaemet_9434_climatogram, 10, *18*
climaemet_9434_temp, 10
climaemet_9434_wind, 11
climaemet_news, 11
climatestripes_station, 12
climatogram_normal, 13
climatogram_period, 14
climatol::diagwl(), *13, 14*, *17, 18*

dms2decdegrees, 15

first_day_of_year, 16

get_data_aemet, 16
get_metadata_aemet *(*get_data_aemet*)*, 16
ggclimat_walter_lieth, 17
ggclimat_walter_lieth(), *10*, *13–15*
ggplot2::theme(), *20, 21*
ggstripes, 19
ggstripes(), *12*
ggwindrose, 20
ggwindrose(), *23, 24*

hcl.pals(), *19*, *21, 22*, *24*

last_day_of_year *(*first_day_of_year*)*, 16

readr::locale(), *18*

theme, *19*

windrose_days, 22
windrose_days(), *24*
windrose_period, 23