Package 'worldmet'

November 7, 2025

Type Package

Title Import Surface Meteorological Data from NOAA Integrated Surface Database (ISD)

Version 0.10.2

Maintainer David Carslaw <david.carslaw@york.ac.uk>

Description Functions to import data from more than 30,000 surface meteorological sites around the world managed by the National Oceanic and Atmospheric Administration (NOAA) Integrated Surface Database (ISD, see <https://www.ncei.noaa.gov/products/land-based-station/integrated-surface-database>).

License MIT + file LICENSE

URL https://openair-project.github.io/worldmet/, https://github.com/openair-project/worldmet

BugReports https://github.com/openair-project/worldmet/issues

Depends R (>= 4.1.0)

Imports cli, dplyr, leaflet, purrr (>= 1.1.0), readr, rlang, sf, tidyr

Suggests carrier, knitr, mirai, rmarkdown

ByteCompile true

Config/Needs/website openair-project/openairpkgdown,openair

Encoding UTF-8

Language en-GB

LazyData true

LazyLoad true

RoxygenNote 7.3.3

NeedsCompilation no

Author David Carslaw [aut, cre] (ORCID: <https://orcid.org/0000-0003-0991-950X>), Jack Davison [aut] (ORCID: <https://orcid.org/0000-0003-2653-6615>)

Repository CRAN

Date/Publication 2025-11-07 13:10:02 UTC

Contents

`exportADMS` *Export a meteorological data frame in ADMS format*

Description

Writes a text file in the ADMS format to a location of the user's choosing, with optional interpolation of missing values.

Usage

```
exportADMS(dat, out = "./ADMS_met.MET", interp = FALSE, maxgap = 2)
```

Arguments

`dat`	A data frame imported by `importNOAA()`.
`out`	A file name for the ADMS file. The file is written to the working directory by default.
`interp`	Should interpolation of missing values be undertaken? If `TRUE` linear interpolation is carried out for gaps of up to and including `maxgap`.
`maxgap`	The maximum gap in hours that should be interpolated where there are missing data when `interp = TRUE`. Data with gaps more than `maxgap` are left as missing.

Value

`exportADMS()` returns the input `dat` invisibly.

Examples

```
## Not run:
# import some data then export it
dat <- importNOAA(year = 2012)
exportADMS(dat, out = "~/adms_met.MET")

## End(Not run)
```

getMeta *Find a ISD site code and other meta data*

Description

Get information on meteorological sites

Usage

```
getMeta(
  site = "heathrow",
  lat = NA,
  lon = NA,
  crs = 4326,
  country = NA,
  state = NA,
  n = 10,
  end.year = "current",
  provider = c("OpenStreetMap", "Esri.WorldImagery"),
  plot = TRUE,
  returnMap = FALSE
)
```

Arguments

site	A site name search string e.g. site = "heathrow". The search strings and be partial and can be upper or lower case e.g. site = "HEATHR".
lat, lon	Decimal latitude and longitude (or other Y/X coordinate if using a different crs). If provided, the n closest ISD stations to this coordinate will be returned.
crs	The coordinate reference system (CRS) of the data, passed to sf::st_crs(). By default this is EPSG:4326, the CRS associated with the commonly used latitude and longitude coordinates. Different coordinate systems can be specified using crs (e.g., crs = 27700 for the British National Grid). Note that non-lat/lng coordinate systems will be re-projected to EPSG:4326 for making comparisons with the NOAA metadata plotting on the map.
country	The country code. This is a two letter code. For a full listing see https://www1.ncdc.noaa.gov/pub/data/noaa/isd-history.csv.
state	The state code. This is a two letter code.
n	The number of nearest sites to search based on latitude and longitude.
end.year	To help filter sites based on how recent the available data are. end.year can be "current", "any" or a numeric year such as 2016, or a range of years e.g. 1990:2016 (which would select any site that had an end date in that range. **By default only sites that have some data for the current year are returned**.

provider	By default a map will be created in which readers may toggle between a vector base map and a satellite/aerial image. provider allows users to override this default; see http://leaflet-extras.github.io/leaflet-providers/preview/ for a list of all base maps that can be used. If multiple base maps are provided, they can be toggled between using a "layer control" interface.
plot	If TRUE will plot sites on an interactive leaflet map.
returnMap	Should the leaflet map be returned instead of the meta data? Default is FALSE.

Details

This function is primarily used to find a site code that can be used to access data using importNOAA(). Sites searches of approximately 30,000 sites can be carried out based on the site name and based on the nearest locations based on user-supplied latitude and longitude.

Value

A data frame is returned with all available meta data, mostly importantly including a code that can be supplied to importNOAA(). If latitude and longitude searches are made an approximate distance, dist in km is also returned.

Author(s)

David Carslaw

See Also

Other NOAA ISD functions: getMetaLive(), importNOAA(), importNOAAlite()

Examples

```
## Not run:
## search for sites with name beijing
getMeta(site = "beijing")

## End(Not run)

## Not run:
## search for near a specified lat/lon - near Beijing airport
## returns 'n' nearest by default
getMeta(lat = 40, lon = 116.9)

## End(Not run)
```

getMetaLive *Obtain site meta data from NOAA server*

Description

Download all NOAA meta data, allowing for re-use and direct querying.

Usage

```
getMetaLive(...)
```

Arguments

...	Currently unused.

Value

a tibble

See Also

Other NOAA ISD functions: `getMeta()`, `importNOAA()`, `importNOAAlite()`

Examples

```
## Not run:
meta <- getMetaLive()
head(meta)

## End(Not run)
```

importNOAA *Import Meteorological data from the NOAA Integrated Surface Database (ISD)*

Description

This is the main function to import data from the NOAA Integrated Surface Database (ISD). The ISD contains detailed surface meteorological data from around the world for over 30,000 locations. For general information of the ISD see https://www.ncei.noaa.gov/products/land-based-station/integrated-surface-database and the map here https://gis.ncdc.noaa.gov/maps/ncei.

Usage

```
importNOAA(
  code = "037720-99999",
  year = 2014,
  hourly = TRUE,
  source = c("delim", "fwf"),
  quiet = FALSE,
  path = NA,
  n.cores = NULL
)
```

Arguments

code	The identifying code as a character string. The code is a combination of the USAF and the WBAN unique identifiers. The codes are separated by a "-" e.g. code = "037720-99999".
year	The year to import. This can be a vector of years e.g. year = 2000:2005.
hourly	Should hourly means be calculated? The default is TRUE. If FALSE then the raw data are returned.
source	The NOAA ISD service stores files in two formats; as delimited CSV files ("delim") and as fixed width files ("fwf"). importNOAA() defaults to "delim" but, if the delimited data store is down, users may wish to try "fwf" instead. Both data sources should be identical to one another.
quiet	If FALSE, print missing sites / years to the screen, and show a progress bar if multiple sites are imported.
path	If a file path is provided, the data are saved as an rds file at the chosen location e.g. path = "C:/Users/David". Files are saved by year and site.
n.cores	No longer recommended; please set mirai::daemons() in your R session. This argument is provided for back compatibility, and is passed to the n argument of mirai::daemons() on behalf of the user. Any set daemons will be reset once the function completes. Default is NULL, which means no parallelism. n.cores = 1L is equivalent to n.cores = NULL.

Details

Note the following units for the main variables:

date Date/time in POSIXct format. **Note the time zone is GMT (UTC) and may need to be adjusted to merge with other local data. See details below.**

latitude Latitude in decimal degrees (-90 to 90).

longitude Longitude in decimal degrees (-180 to 180). Negative numbers are west of the Greenwich Meridian.

elevation Elevation of site in metres.

wd Wind direction in degrees. 90 is from the east.

ws Wind speed in m/s.

ceil_hgt The height above ground level (AGL) of the lowest cloud or obscuring phenomena layer aloft with 5/8 or more summation total sky cover, which may be predominantly opaque, or the vertical visibility into a surface-based obstruction.

visibility The visibility in metres.

air_temp Air temperature in degrees Celcius.

dew_point The dew point temperature in degrees Celcius.

atmos_pres The sea level pressure in millibars.

RH The relative humidity (%).

cl_1, ..., cl_3 Cloud cover for different layers in Oktas (1-8).

cl Maximum of cl_1 to cl_3 cloud cover in Oktas (1-8).

cl_1_height, ..., cl_3_height Height of the cloud base for each later in metres.

precip_12 12-hour precipitation in mm. The sum of this column should give the annual precipitation.

precip_6 6-hour precipitation in mm.

precip This value of precipitation spreads the 12-hour total across the previous 12 hours.

pwc The description of the present weather description (if available).

The data are returned in GMT (UTC). It may be necessary to adjust the time zone when combining with other data. For example, if air quality data were available for Beijing with time zone set to "Etc/GMT-8" (note the negative offset even though Beijing is ahead of GMT. See the `openair` package and manual for more details), then the time zone of the met data can be changed to be the same. One way of doing this would be `attr(met$date, "tzone") <- "Etc/GMT-8"` for a meteorological data frame called `met`. The two data sets could then be merged based on `date`.

Value

Returns a data frame of surface observations. The data frame is consistent for use with the `openair` package. Note that the data are returned in GMT (UTC) time zone format. Users may wish to express the data in other time zones, e.g., to merge with air pollution data.

Parallel Processing

If you are importing a lot of meteorological data, this can take a long while. This is because each combination of year and station requires downloading a separate data file from NOAA's online data directory, and the time each download takes can quickly add up. `importNOAA()` and `importNOAAlite()` can use parallel processing to speed downloading up, powered by the capable `{mirai}` package. If users have any `{mirai}` "daemons" set, these functions will download files in parallel. The greatest benefits will be seen if you spawn as many daemons as you have cores on your machine, although one fewer than the available cores is often a good rule of thumb. Your mileage may vary, however, and naturally spawning more daemons than station-year combinations will lead to diminishing returns.

```
# set workers - once per session
mirai::daemons(4)

# import lots of data - NB: no change in importNOAA()!
big_met <- importNOAA(code = "037720-99999", year = 2010:2020)
```

Author(s)

David Carslaw

See Also

Other NOAA ISD functions: `getMeta()`, `getMetaLive()`, `importNOAAlite()`

Examples

```
## Not run:
# import some data
beijing_met <- importNOAA(code = "545110-99999", year = 2010:2011)

# importing lots of data? use mirai for parallel processing
mirai::daemons(4)
beijing_met2 <- importNOAA(code = "545110-99999", year = 2010:2025)

## End(Not run)
```

`importNOAAlite` *Import "Lite" Meteorological data from the NOAA Integrated Surface Database (ISD)*

Description

This function is an alternative to `importNOAA()`, and provides access to the "Lite" format of the data. This a subset of the larger `importNOAA()` dataset featuring eight common climatological variables. As it assigns the nearest measurement to the "top of the hour" to the data, specific values are likely similar but different to those returned by `importNOAA()`. Read the technical document for more information.

Usage

```
importNOAAlite(code = "037720-99999", year = 2025, quiet = FALSE, path = NA)
```

Arguments

`code`	The identifying code as a character string. The code is a combination of the USAF and the WBAN unique identifiers. The codes are separated by a "-" e.g. `code = "037720-99999"`.
`year`	The year to import. This can be a vector of years e.g. `year = 2000:2005`.
`quiet`	If `FALSE`, print missing sites / years to the screen, and show a progress bar if multiple sites are imported.
`path`	If a file path is provided, the data are saved as an rds file at the chosen location e.g. `path = "C:/Users/David"`. Files are saved by year and site.

Details

Note the following units for the main variables:

date Date/time in POSIXct format. **Note the time zone is UTC and may need to be adjusted to merge with other local data.

latitude Latitude in decimal degrees (-90 to 90).

longitude Longitude in decimal degrees (-180 to 180). Negative numbers are west of the Greenwich Meridian.

elev Elevation of site in metres.

ws Wind speed in m/s.

wd Wind direction in degrees. 90 is from the east.

air_temp Air temperature in degrees Celcius.

atmos_pres The sea level pressure in millibars.

dew_point The dew point temperature in degrees Celcius.

precip_6 6-hour precipitation in mm.

precip_1 1-hour precipitation in mm.

sky Sky Condition Total Coverage Code.

The data are returned in GMT (UTC). It may be necessary to adjust the time zone when combining with other data. For example, if air quality data were available for Beijing with time zone set to "Etc/GMT-8" (note the negative offset even though Beijing is ahead of GMT. See the `openair` package and manual for more details), then the time zone of the met data can be changed to be the same. One way of doing this would be `attr(met$date, "tzone") <- "Etc/GMT-8"` for a meteorological data frame called `met`. The two data sets could then be merged based on `date`.

Value

Returns a data frame of surface observations. The data frame is consistent for use with the `openair` package. Note that the data are returned in GMT (UTC) time zone format. Users may wish to express the data in other time zones, e.g., to merge with air pollution data.

Parallel Processing

If you are importing a lot of meteorological data, this can take a long while. This is because each combination of year and station requires downloading a separate data file from NOAA's online data directory, and the time each download takes can quickly add up. `importNOAA()` and `importNOAAlite()` can use parallel processing to speed downloading up, powered by the capable `{mirai}` package. If users have any `{mirai}` "daemons" set, these functions will download files in parallel. The greatest benefits will be seen if you spawn as many daemons as you have cores on your machine, although one fewer than the available cores is often a good rule of thumb. Your mileage may vary, however, and naturally spawning more daemons than station-year combinations will lead to diminishing returns.

```
# set workers - once per session
mirai::daemons(4)

# import lots of data - NB: no change in importNOAA()!
big_met <- importNOAA(code = "037720-99999", year = 2010:2020)
```

Author(s)

Jack Davison

See Also

`getMeta()` to obtain the codes based on various site search approaches.

Other NOAA ISD functions: `getMeta()`, `getMetaLive()`, `importNOAA()`

Examples

```
## Not run:
heathrow_lite <- importNOAAlite(code = "037720-99999", year = 2025)

## End(Not run)
```

`weatherCodes` *Codes for weather types*

Description

This data frame consists of the weather description codes used in the ISD. It is not of general use to most users.

Usage

```
weatherCodes
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 100 rows and 2 columns.

Details

pwc Weather code, which can be merged with the `pwc` column in `importNOAA()` datasets.

description Description associated with the weather codes.

Examples

```
weatherCodes
```

Index