Package: getRad (via r-universe)

June 11, 2026

Title Download Radar Data for Biological Research

Version 0.3.0

Description Load polar volume and vertical profile data for aeroecological research directly into R. With 'getRad' you can access data from several sources in Europe and the US and standardize it to facilitate further exploration in tools such as 'bioRad'.

License MIT + file LICENSE

URL https://github.com/aloftdata/getRad, https://aloftdata.github.io/getRad/

BugReports https://github.com/aloftdata/getRad/issues

Depends R (>= 4.1.0)

Imports bioRad, cachem, cli, dplyr (>= 1.1.0), glue, httr2 (>= 1.1.1), lubridate, purrr (>= 1.0.0), rlang, tibble, tools, utils, vroom, withr, xml2

Suggests askpass, fs, htmltools, keyring, knitr, leaflet, rhdf5, rnaturalearth, rnaturalearthdata, sf, testthat (>= 3.0.0), tidyr, vol2birdR

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Config/Needs/website rmarkdown, leafpop, htmltools

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Bart Kranstauber [aut, cre] (ORCID: <https://orcid.org/0000-0001-8303-780X>, affiliation: University of Amsterdam), Pieter Huybrechts [aut] (ORCID: <https://orcid.org/0000-0002-6658-6062>, affiliation: Research Institute for Nature and Forest (INBO)), Peter Desmet [aut] (ORCID: <https://orcid.org/0000-0002-8442-8025>, affiliation:

Research Institute for Nature and Forest (INBO)), Cecilia Nilsson [ctb] (ORCID: <https://orcid.org/0000-0001-8957-4411>, affiliation: Lund University), Alexander Tedeschi [ctb] (ORCID: <https://orcid.org/0000-0003-0772-6931>, affiliation: Cornell Lab of Ornithology), Hidde Leijnse [ctb] (ORCID: <https://orcid.org/0000-0001-7835-4480>, affiliation: Royal Netherlands Meteorological Institute), Bart Hoekstra [ctb] (ORCID: <https://orcid.org/0000-0002-7085-3805>, affiliation: University of Amsterdam), University of Amsterdam [cph] (ROR: <https://ror.org/04dkp9463>), Biodiversa+ [fnd] (https://hirad.science/)

Maintainer Bart Kranstauber <b.kranstauber@uva.nl>

Config/pak/sysreqs libssl-dev

Repository https://cran.r-universe.dev

Date/Publication 2026-06-11 16:32:48 UTC

RemoteUrl https://github.com/cran/getRad

RemoteRef HEAD

RemoteSha 299a335a2f2947d6f73439a3a389c8bd8bd38632

Contents

get_pvol 2
get_vpts 3
get_vpts_coverage 6
get_weather_radars 6
set_secret 7

Index **9**

`get_pvol` *Get polar volume (PVOL) data from supported sources*

Description

Gets polar volume data from supported sources and returns it as a (list of) polar volume objects. The source is automatically detected based on the provided `radar`.

Usage

```
get_pvol(radar = NULL, datetime = NULL, ...)
```

Arguments

radar	Name of the radar (odim code) as a character string (e.g. "nlhrw" or "fikor").
datetime	Either: • A single POSIXct, for which the most representative data file is downloaded. In most cases this will be the time before. • A lubridate::interval() or two POSIXct, between which all data files are downloaded.
...	Additional arguments passed on to reading functions, for example param = "all" to the bioRad::read_pvolfile().

Details

For more details on supported sources, see vignette("supported_sources"). Within supported countries there might also be temporal restrictions on the radars that are operational. For example, radars with the status 0 in get_weather_radars("opera") are currently not operational.

Not all radars in the nexrad archive can be read successfully. Radars associated with the Terminal Doppler Weather Radar (TDWR) program can not be read. These can be identified using the stntype column in get_weather_radars("nexrad").

Value

Either a polar volume or a list of polar volumes. See bioRad::summary.pvol() for details.

Examples

```
# Get PVOL data for a single radar and datetime
get_pvol("deess", as.POSIXct(Sys.Date()))

# Get PVOL data for multiple radars and a single datetime
get_pvol(
  c("deess", "dehnr", "fianj", "czska", "KABR"),
  as.POSIXct(Sys.Date())
)
```

---

get_vpts *Get vertical profile time series (VPTS) data from supported sources*

Description

Gets vertical profile time series data from supported sources and returns it as a (list of) of vpts objects or a dplyr::tibble().

Usage

```
get_vpts(
  radar,
  datetime,
  source = c("baltrad", "uva", "ecog-04003", "rmi", "birdcast", "dark_ecology"),
  return_type = c("vpts", "tibble"),
  ...,
  path = NULL
)
```

Arguments

radar	Name of the radar (odim code) as a character string (e.g. "nlhrw" or "fikor").
datetime	Either: • A POSIXct datetime (or character representation), for which the data file is downloaded. • A Date date (or character representation), for which all data files are downloaded. • A vector of datetimes or dates, between which all data files are downloaded. • A lubridate::interval(), between which all data files are downloaded.
source	Source of the data. One of "baltrad", "uva", "ecog-04003", "rmi", "dark_ecology" or "birdcast". Only one source can be queried at a time. If not provided, "baltrad" is used.
return_type	Type of object that should be returned. Either: • "vpts": vpts object(s) (default). • "tibble": a dplyr::tibble().
...	Optional arguments, to bioRad::read_cajun() when reading "dark_ecology" data.
path	A local directory where data are read from. If specified the file structure is taken from the source argument. See details for an explanation of the file format.

Details

For more details on supported sources, see vignette("supported_sources").

In case data is read from a directory, file in the directory should be structures like they are in the monthly folders of the aloft repository. To specify an alternative structure the "getRad.vpts_local_path_format_aloft" option can be used. This can, for example, be used to read daily data. Some example options for the glue formatters are:

- "{radar}/{year}/{radar}_vpts_{year}{month}.csv.gz": The default format, the same structure as the monthly directories in the aloft repository. Or as contained in the tgz files in the aloft zenodo repository.
- "{substr(radar, 1,2)}/{radar}/{year}/{radar}_vpts_{year}{month}.csv.gz": The format as in the files in the zenodo aloft repository

- "{radar}/{year}/{radar}_vpts_{year}{month}{day}.csv": The format as daily data is stored in aloft data

A similar option ("getRad.vpts_local_path_format_aloft") exist for reading dark ecology data. The default value here is "getRad.vpts_local_path_format_aloft". Here the option does refer to the directories where the dark ecology files should be searched.

Besides the examples above there is a date object available for formatting. Note that day and month are zero padded character strings in the glue formating.

Value

Either a vpts object, a list of vpts objects or a tibble. See bioRad::summary.vpts for details.

Examples

```
# Get VPTS data for a single radar and date
get_vpts(radar = "bejab", datetime = "2023-01-01", source = "baltrad")
get_vpts(radar = "bejab", datetime = "2020-01-19", source = "rmi")

# Get VPTS data for multiple radars and a single date
get_vpts(
  radar = c("dehnr", "deflg"),
  datetime = lubridate::ymd("20171015"),
  source = "baltrad"
)

# Get VPTS data for a single radar and a date range
get_vpts(
  radar = "bejab",
  datetime = lubridate::interval(
    lubridate::ymd_hms("2023-01-01 00:00:00"),
    lubridate::ymd_hms("2023-01-02 00:14:00")
  ),
  source = "baltrad"
)
get_vpts("bejab", lubridate::interval("20210101", "20210301"))

# Get VPTS data for a single radar, date range and non-default source
get_vpts(radar = "bejab", datetime = "2016-09-29", source = "ecog-04003")

# Return a tibble instead of a vpts object
get_vpts(
  radar = "chlem",
  datetime = "2023-03-10",
  source = "baltrad",
  return_type = "tibble"
)
#' Get VPTS data from the public BirdCast NEXRAD archive
get_vpts(radar = "KABR", datetime = "2023-01-01", source = "birdcast")
```

get_vpts_coverage *Get VPTS file coverage from supported sources*

Description

Gets the VPTS file coverage from supported sources per radar and date.

Usage

```
get_vpts_coverage(
  source = c("baltrad", "uva", "ecog-04003", "rmi", "birdcast"),
  ...
)
```

Arguments

source	Source of the data. One or more of "baltrad", "uva", "ecog-04003" or "rmi" or "birdcast". If not provided, "baltrad" is used. Alternatively "all" can be used if data from all sources should be returned.
...	Arguments passed on to internal functions.

Value

A data.frame or tibble with at least three columns, source, radar and date to indicate the combination for which data exists.

Examples

```
get_vpts_coverage()
```

get_weather_radars *Get weather radar metadata*

Description

Gets weather radar metadata from OPERA and/or NEXRAD.

Usage

```
get_weather_radars(source = c("opera", "nexrad"), use_cache = TRUE, ...)
```

Arguments

source	Source of the metadata. "opera", "nexrad" or "all". If not provided, "opera" is used.
use_cache	Logical indicating whether to use the cache. Default is TRUE. If FALSE the cache is ignored and the file is fetched anew. This can also be useful if you want to force a refresh of the cache.
...	Additional arguments passed on to reading functions per source, currently not used.

Details

The source files for this function are:

- For opera: OPERA_RADARS_DB.json (main/current) and OPERA_RADARS_ARH_DB.json (archive). A column origin is added to indicate which file the metadata were derived from.
- For nexrad: nexrad-stations.txt.

Value

A sf or tibble with weather radar metadata. In all cases the column source is added to indicate the source of the data and radar to show the radar identifiers used in other functions like get_pvol() and get_vpts().

Examples

```
# Get radar metadata from OPERA
get_weather_radars(source = "opera")

# Get radar metadata from NEXRAD
get_weather_radars(source = "nexrad")
```

---

set_secret *Set or get secrets from the keyring*

Description

Some services require credentials to access data. This function uses keyring to safely store those credentials on your computer.

Usage

```
set_secret(name, secret = NULL)

get_secret(name)
```

Arguments

name	Name of the secret to set or get as a character (e.g. "nl_api_key").
secret	Optionally a character string with the secret, alternatively the system will prompt the user.

Details

When working with a cluster it might be advantageous to use a specific keyring, this can be done by setting the keyring_backend option in R.

The package uses the option getRad.key_prefix as a prefix to all keys stored. If you want to use multiple keys for the same api you can manipulate this option.

Value

set_secret() returns TRUE when a secret has successfully been set. get_secret() returns the secret as a character string.

Index

bioRad::read_cajun(), *4*
bioRad::read_pvolfile(), *3*
bioRad::summary.pvol(), *3*
bioRad::summary.vpts, *5*

Date, *4*
dplyr::tibble(), *3*, *4*

get_pvol, 2
get_pvol(), *7*
get_secret *(*set_secret*)*, 7
get_vpts, 3
get_vpts(), *7*
get_vpts_coverage, 6
get_weather_radars, 6

lubridate::interval(), *3*, *4*

polar volume objects, *2*
POSIXct, *3*, *4*

set_secret, 7

vpts objects, *3*