Package 'prism'

May 9, 2026

Title Access Data from the Oregon State Prism Climate Project

Description Allows users to access the Oregon State Prism climate data (<https://prism.nacse.org/>). Using the web service API data can easily downloaded in bulk and loaded into R for spatial analysis. Some user friendly visualizations are also provided.

URL https://docs.ropensci.org/prism/, https://github.com/ropensci/prism

BugReports https://github.com/ropensci/prism/issues

Version 0.3.0

License MIT + file LICENSE

Imports dplyr, ggplot2, httr, magrittr, raster, readr, stringr, terra, utils

VignetteBuilder knitr

Suggests covr, knitr, rmarkdown, testthat (>= 2.1.0)

RoxygenNote 7.3.3

Encoding UTF-8

NeedsCompilation no

Author Hart Edmund [aut, ccp], Kendon Bell [aut], Alan Butler [ctb, cre] (ORCID: <https://orcid.org/0000-0002-8625-6386>)

Maintainer Alan Butler <rabutler@usbr.gov>

Repository CRAN

Date/Publication 2025-11-14 21:20:02 UTC

Contents

`get_prism_annual` *Download prism data*

Description

Download grid cell data from the prism project. Temperature (min, max, and mean), mean dewpoint temperature, precipitation, and vapor pressure deficit (min and max) can be downloaded for annual (`get_prism_annual()`), monthly (`get_prism_monthlys()`), daily (`get_prism_dailys()`), and 30-year averages (`get_prism_normals()`). Data are available at 4km and 800m resolution for daily, monthly, and annual data. Normals can also be downloaded at 800m resolution.

Download data from the prism project for 30 year normals at 4km or 800m grid cell resolution for precipitation; mean, min and max temperature; clear sky, sloped, and total solar radiation; and cloud transmittance.

Usage

```
get_prism_annual(
  type,
  years,
  keepZip = TRUE,
  keep_pre81_months = NULL,
  service = NULL,
  resolution = "4km"
)

get_prism_dailys(
  type,
  minDate = NULL,
  maxDate = NULL,
  dates = NULL,
  keepZip = TRUE,
  check = "httr",
  service = NULL,
  resolution = "4km"
)
```

```
get_prism_monthlys(
  type,
  years,
  mon = 1:12,
  keepZip = TRUE,
  keep_pre81_months = NULL,
  service = NULL,
  resolution = "4km"
)

get_prism_normals(
  type,
  resolution,
  mon = NULL,
  annual = FALSE,
  keepZip = TRUE,
  day = NULL
)
```

Arguments

type	The type of data to download. Must be "ppt", "tmean", "tmin", "tmax", "tdmean", "vpdmin", or "vpdmax". "solclear", "solslope", "soltotal", and "soltrans" are also available only for `get_prism_normals()`. Note that `tmean == mean(tmin, tmax)`.
years	a valid numeric year, or vector of years, to download data for.
keepZip	if `TRUE`, leave the downloaded zip files in your 'prism.path', if `FALSE`, they will be deleted.
keep_pre81_months	Deprecated
service	Either `NULL` (default) or a URL provided by PRISM staff for subscription-based service. Example: "http://services.nacse.org/prism/data/subscription/800m". To use the subscription option, you must use an IP address registered with PRISM staff. When `NULL`, the new PRISM web service endpoints are used based on the specified resolution.
resolution	The spatial resolution of the data, must be either "4km" or "800m".
minDate	Date to start downloading daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
maxDate	Date to end downloading daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
dates	A vector of dates to download daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
check	One of "httr" or "internal". See details.

mon	a valid numeric month, or vector of months. Required for get_prism_monthlys(). Can be NULL for get_prism_normals().
annual	if TRUE download annual normals.
day	Download daily normals. If TRUE, then download all daily data for the specified months or entire year (if annual is TRUE). Individual days can be specified as a Date object, where the year is ignored or specified as characters in "mm-dd" or "mmdd" form.

Details

A valid download directory must exist before downloading any prism data. This can be set using prism_set_dl_dir() and can be verified using prism_check_dl_dir().

For the check parameter, "httr", the default, checks the file name using the web service, and downloads if that file name is not in the file system. "internal" (much faster) only attempts to download layers that are not already in the file system as stable. "internal" should be used with caution as it is not robust to changes in version or file names.

Value

Nothing is returned - an error will occur if download is not successful.

Annual and Monthly

Annual and monthly prism data are available from 1895 to present. For 1895-1980 data, monthly and annual data are grouped together in one download file; keep_pre81_months determines if the other months/yearly data are kept after the download. Data will be downloaded for all specified months (mon) in all the years in the supplied vectors.

Data are available at two spatial resolutions: 4km (approximately 2.5 arc-minutes) and 800m. The 4km resolution covers the entire CONUS and is suitable for most applications. The 800m resolution provides higher spatial detail but results in larger file sizes and longer download times.

Daily

Daily prism data are available beginning on January 1, 1981. To download the daily data, dates must be in the proper format or downloading will not work properly. Dates can be specified using either a date range via minDate and maxDate, or a vector of dates, but not both.

Data are available at two spatial resolutions: 4km (approximately 2.5 arc-minutes) and 800m. The 4km resolution covers the entire CONUS and is suitable for most applications. The 800m resolution provides higher spatial detail but results in larger file sizes and longer download times.

Normals

30-year normals are currently computed using 1991-2020 and are available at 4km and 800m resolution. See https://prism.nacse.org/normals/. If mon is specified and annual is TRUE, then monthly and annual normal data will be downloaded. Clear sky, sloped, and total solar radiation; and cloud transmittance are not available for daily normals.

Examples

```
## Not run:
# Get all annual average temperature data from 1990 to 2000 at default resolution
get_prism_annual(type = "tmean", years = 1990:2000, keepZip = FALSE)

# Get annual precipitation for multiple years at 800m resolution
get_prism_annual(
  type = "ppt",
  years = 2020:2022,
  resolution = "800m",
  keepZip = FALSE
)

# Get single year of annual temperature data at high resolution
get_prism_annual(
  type = "tmax",
  years = 2023,
  resolution = "800m",
  keepZip = TRUE
)

# will fail - invalid resolution:
get_prism_annual(
  type = "tmean",
  years = 2023,
  resolution = "1km",
  keepZip = FALSE
)

## End(Not run)

## Not run:
# get daily average temperature data for June 1 - 14, 2013 at 4km resolution
get_prism_dailys(
  type = "tmean",
  minDate = "2013-06-01",
  maxDate = "2013-06-14",
  keepZip = FALSE
)

# get precipitation data for June 1, 2013 at 800m resolution
get_prism_dailys(
  type = "ppt",
  dates = "2013/06/01",
  resolution = "800m",
  keepZip = FALSE
)

# get average temperature for three specific days at default resolution
get_prism_dailys(
  type = "tmean",
  dates = c("2013-06-01", "2013-06-14", "2014-06-30"),
```

```
  keepZip = FALSE
)

# get high-resolution daily maximum temperature for a date range
get_prism_dailys(
  type = "tmax",
  minDate = "2013-06-01",
  maxDate = "2013-06-07",
  resolution = "800m",
  keepZip = FALSE
)

# will fail - both minDate and dates specified:
get_prism_dailys(
  type = "ppt",
  minDate = "2013-06-01",
  dates = "2013-06-14",
  keepZip = FALSE
)

# will fail - minDate without maxDate:
get_prism_dailys(
  type = "ppt",
  minDate = "2013-06-01",
  keepZip = FALSE
)

# will fail - invalid resolution:
get_prism_dailys(
  type = "tmean",
  dates = "2013-06-01",
  resolution = "1km",
  keepZip = FALSE
)

## End(Not run)

## Not run:
# Get all the precipitation data for January from 1990 to 2000 at 4km resolution
get_prism_monthlys(type = "ppt", years = 1990:2000, mon = 1, keepZip = FALSE)

# Get January-December 2005 monthly precipitation at default resolution
get_prism_monthlys(type = "ppt", years = 2005, mon = 1:12, keepZip = FALSE)

# Get high-resolution monthly temperature data for summer months
get_prism_monthlys(
  type = "tmean",
  years = 2023,
  mon = 6:8,
  resolution = "800m",
  keepZip = FALSE
)
```

```
# Get multiple years of winter precipitation at 800m resolution
get_prism_monthlys(
  type = "ppt",
  years = 2020:2022,
  mon = c(12, 1, 2),
  resolution = "800m",
  keepZip = TRUE
)

# will fail - invalid resolution:
get_prism_monthlys(
  type = "ppt",
  years = 2023,
  mon = 6,
  resolution = "1km",
  keepZip = FALSE
)

## End(Not run)

## Not run:
# Get 30 year normal values for January rainfall
get_prism_normals(type = "ppt", resolution = "4km", mon = 1, keepZip = FALSE)

# Get monthly (every month) and annual 30-year normals for mean temperature
get_prism_normals(
  type = "tmean",
  resolution = "800m",
  mon = 1:12,
  annual = TRUE,
  keepZip = FALSE
)

# Get daily precip normals for January 1 and March 1
get_prism_normals('ppt', '4km', NULL, FALSE, TRUE, c('0101', '0301'))

# Get daily precip normals for all of February
get_prism_normals('ppt', '4km', 2, FALSE, TRUE, TRUE)

# Get July 2nd average temperature 30-year average
get_prism_normals('tmean', '800m', NULL, FALSE, TRUE, as.Date('2000-07-02'))

## End(Not run)
```

---

pd_get_md *Get prism metadata*

Description

Retrieves prism metadata from the specified prism data. "prism data", i.e., pd are the folder names returned by `prism_archive_ls()` or `prism_archive_subset()`. These functions get the name or date from these data, or convert these data to a file name. A warning is provided if the specified prism data do not exist in the archive.

Usage

```
pd_get_md(pd)
```

Arguments

pd	prism data character vector.

Details

The metadata includes the following variables from the .info.txt file for daily, monthly, and annual data:

- PRISM_DATASET_FILENAME
- PRISM_DATASET_CREATE_DATE
- PRISM_DATASET_TYPE
- PRISM_DATASET_VERSION
- PRISM_CODE_VERSION
- PRISM_DATASET_REMARKS

Additionally, two local variables are added identifying where the file is located on the local system:

- file_path
- folder_path

The annual and monthly normals data includes different keys in the .info.txt, so they are renamed to be the same as those found in the other temporal data. The keys/variables are renamed as follows:

- PRISM_FILENAME –> PRISM_DATASET_FILENAME
- PRISM_CREATE_DATE –> PRISM_DATASET_CREATE_DATE
- PRISM_DATASET –> PRISM_DATASET_TYPE
- PRISM_VERSION –> PRISM_CODE_VERSION
- PRISM_REMARKS –> PRISM_DATASET_REMARKS

Additionally, the normals does not include PRISM_DATASET_VERSION, so that variable is added with `NA` values.

Value

data.frame containing metadata for all specified prism data.

Examples

```
## Not run:
#' # Assumes 2000-2002 annual precipitation data is already downloaded
pd <- prism_archive_subset('ppt', 'annual', years = 2000:2002)
df <- pd_get_md(pd)
head(df)

## End(Not run)
```

`pd_get_name` *Perform action on "prism data"*

Description

`pd_get_name()` extracts a long, human readable name from the prism data.

`pd_get_date()` extracts the date from the prism data. Date is returned in yyyy-mm-dd format. For monthly data, dd is 01 and for annual data mm is also 01. For normals, an empty character is returned.

`pd_get_type()` parses the variable from the prism data.

`prism_md()` is a deprecated function that has been replaced with `pd_get_name()` and `pd_get_date()`

`pd_to_file()` converts prism data to a fully specified .bil file, i.e., the full path to the file in the prism archive. A warning is posted if the file does not exist in the local prism archive.

Usage

```
pd_get_name(pd)

pd_get_date(pd)

pd_get_type(pd)

prism_md(f, returnDate = FALSE)

pd_to_file(pd)
```

Arguments

`pd`	prism data character vector.
`f`	1 or more prism directories name or .bil files.
`returnDate`	TRUE or FALSE. If TRUE, an ISO date is returned. By default years will come back with YYYY-01-01 and months as YYYY-MM-01

Details

"prism data", i.e., pd are the folder names returned by `prism_archive_ls()` or `prism_archive_subset()`. These functions get the name or date from these data, or convert these data to a file name.

Value

`pd_get_name()` and `pd_get_date()` return a character vector of names/dates.

`pd_get_type()` returns a character vector of prism variable types, e.g., 'ppt'.

`pd_to_file()` returns a character vector with the full path to the bil file.

Examples

```
## Not run:
# Assumes 2000-2002 annual precipitation data is already downloaded
pd <- prism_archive_subset('ppt', 'annual', years = 2000:2002)
pd_get_name(pd)
## [1] "2000 - 4km resolution - Precipitation" "2001 - 4km resolution - Precipitation"
## [3] "2002 - 4km resolution - Precipitation"

pd_get_date(pd)
## [1] "2000-01-01" "2001-01-01" "2002-01-01"

pd_get_type(pd)
## [1] "ppt" "ppt" "ppt"

pd_to_file(pd[1])
## [1] "C:/prismdir/PRISM_ppt_stable_4kmM3_2000_bil/PRISM_ppt_stable_4kmM3_2000_bil.bil"

## End(Not run)
```

---

pd_get_station_md *Extract prism station metadata*

Description

`pd_get_station_md()` extracts prism metadata on the stations used to generate the prism data. **The data must already be downloaded and available in the prism download folder.** "prism data", i.e., pd are the folder names returned by `prism_archive_ls()` or `prism_archive_subset()`.

`get_prism_station_md()` is a deprecated version of `pd_get_station_md()` that only works with daily prism data.

Usage

```
pd_get_station_md(pd)

get_prism_station_md(type, minDate = NULL, maxDate = NULL, dates = NULL)
```

Arguments

pd	prism data character vector.
type	The type of data you want to subset. Must be "ppt", "tmean", "tmin", "tmax", "tdmean", "vpdmin", "vpdmax", "solclear", "solslope", "soltotal", or "soltrans".
minDate	Date to start subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
maxDate	Date to end subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
dates	A vector of daily dates to subset. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.

Details

Note that station metadata does not exist for "tmean" type, any "annual" temporal periods, nor for daily normals.

See prism_archive_subset() for further details on specifying ranges of dates for different temporal periods.

Value

A tbl_df containing metadata on the stations used for the specified day and variable. The data frame contains the following columns: "date", "prism_data", "type", "station", "name", "longitude", "latitude", "elevation", "network", "stnid"

The "date" column is a character representation of the data. Monthly and annual data are given first day of month, and first month of year for reporting here. Monthly and annual normals are empty strings.

See Also

prism_archive_subset()

Examples

```
## Not run:
# download and then get meta data for January 1, 2010 precipitation
get_prism_dailys("ppt", dates = "2010-01-01")
pd <- prism_archive_subset("ppt", "daily", dates = "2010-01-01")

# will warn that 2010-01-02 is not found:
pd_get_station_md(pd)

## End(Not run)
```

pd_image *Quick spatial image of prism data*

Description

`pd_image()` makes a spatial image plot of the specified prism data (single variable and time step.). It is meant for rapid visualization, but more detailed plots will require other methods.

`prism_image()` is the deprecated version of `pd_image()`.

Usage

```
pd_image(pd, col = "heat")

prism_image(prismfile, col = "heat")
```

Arguments

pd, prismfile	the name of a single file to be plotted, this is most easily found through `prism_archive_ls()` or `prism_archive_subset()`.
col	the color pattern to use. The default is heat, the other valid option is "redblue".

Value

Invisibly returns gg object of the image.

See Also

`prism_archive_ls()`, `prism_archive_subset()`, `ggplot2::geom_raster()`

Examples

```
## Not run:
get_prism_dailys(
  type = "tmean",
  minDate = "2013-06-01",
  maxDate = "2013-06-14",
  keepZip = FALSE
)

# get June 5th
pd <- prism_archive_subset("tmean", "daily", dates = "2013-06-05")

# and plot it
pd_image(pd)

## End(Not run)
```

pd_plot_slice *Plot a slice of a raster stack*

Description

pd_plot_slice() plots a slice of data at a single point location from the specified prism data.

prism_slice() is the deprecated version of pd_plot_slice().

Usage

```
pd_plot_slice(pd, location)

prism_slice(location, prismfile)
```

Arguments

pd, prismfile	a vector of output from prism_archive_ls() or prism_archive_subset() giving a list of prism files to extract data from and plot. The latter is preferred as it will help ensure the prism data are from the same variable and temporal period.
location	a vector of a single location in the form of long,lat

Details

The user should ensure the prism data comes from a continuous data set and is made up of the same temporal period. Otherwise the plot will look erratic and incorrect.

Value

A gg object of the plot for the requested location.

Examples

```
## Not run:
### Assumes you have a clean prism directory
get_prism_dailys(
  type="tmean",
  minDate = "2013-06-01",
  maxDate = "2013-06-14",
  keepZip = FALSE
)
p <- pd_plot_slice(
  prism_archive_subset("tmean", "daily", year = 2020),
  c(-73.2119,44.4758)
)
print(p)

## End(Not run)
```

pd_stack *Stack prism data*

Description

pd_stack() creates a raster stack from prism data. It is up to the user to ensure that pd is of the expected variable and temporal period, i.e., the function does no checking and will stack data with different variables or temporal periods.

prism_stack() is the deprecated version of pd_stack().

Usage

```
pd_stack(pd)

prism_stack(prismfile)
```

Arguments

pd, prismfile	A vector of prism data returned by prism_archive_ls() or prism_archive_subset().

Value

A RasterStack object. Raster layers are stacked in the order they are provided in pd.

Examples

```
## Not run:
get_prism_dailys(
  type="tmean",
  minDate = "2013-06-01",
  maxDate = "2013-06-14",
  keepZip = FALSE
)
# get a raster stack of June 1-14 daily tmean
mystack <- prism_stack(prism_archive_subset(
  "tmean",
  minDate = "2013-06-01",
  maxDate = "2013-06-14"
))

## End(Not run)
```

`prism_archive_clean` *Clean the prism data by removing early and provisional data*

Description

`prism_archive_clean()` 'cleans' the prism download data by removing early and/or provisional data if newer (provisional or stable) data also exist for the same variable and temporal period. Stable data are newer than provisional data that are newer than early data; only the newest data are kept when the "clean" is performed.

`del_early_prov()` is a deprecated version of `prism_archive_clean()` that only works for daily data, and does not prompt the user to confirm which folders should be removed.

Usage

```
prism_archive_clean(
  type,
  temp_period,
  years = NULL,
  mon = NULL,
  minDate = NULL,
  maxDate = NULL,
  dates = NULL,
  resolution = NULL
)

del_early_prov(type, minDate = NULL, maxDate = NULL, dates = NULL)
```

Arguments

`type`	The type of data you want to subset. Must be "ppt", "tmean", "tmin", "tmax", "tdmean", "vpdmin", "vpdmax", "solclear", "solslope", "soltotal", or "soltrans".
`temp_period`	The temporal period to subset. Must be "annual", "monthly", "daily", "monthly normals", or "annual normals".
`years`	Valid numeric year, or vector of years.
`mon`	Valid numeric month, or vector of months.
`minDate`	Date to start subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
`maxDate`	Date to end subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
`dates`	A vector of daily dates to subset. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
`resolution`	The spatial resolution of the data, must be either "4km" or "800m". Required for all temporal periods.

Details

`prism_archive_clean()` prompts the user to verify the folders that should be removed when R is running in interactive mode. Otherwise, all data that are identified to be older than the newest available data are removed.

Daily data are considered "early" for the current month. The previous six months are provisional data. After six months data are considered stable. Thus early data only exist for daily data, while there can be monthly (and presumably yearly) provisional data.

Value

Invisibly returns vector of all deleted folders.

Examples

```
## Not run:
# delete any provisional annual precipitation data from 2000-2023
# del_files will containg any deleted files
del_files <- prism_archive_clean('ppt', 'annual', 2000:2023, resolution = "4km")

## End(Not run)
```

`prism_archive_ls` *List available prism data*

Description

`prism_archive_ls()` lists all available prism data (all variables and all temporal periods) that are available in the local archive, i.e., they have already been downloaded and are available in `prism_get_dl_dir()`. `prism_archive_subset()` can be used to subset the archive based on specified variables and temporal periods.

`ls_prism_data()` is a deprecated version of `prism_data_ls()`.

Usage

```
prism_archive_ls()

ls_prism_data(absPath = FALSE, name = FALSE)
```

Arguments

`absPath`	TRUE if you want to return the absolute path.
`name`	TRUE if you want file names and titles of data products.

Details

prism_archive_ls() only returns the values found in the files column as returned by ls_prism_data(). To replicate the behavior of ls_prism_data(), use pd_get_name() and pd_to_file() with the output of prism_archive_ls()

Value

prism_archive_ls() returns a character vector.

ls_prism_data() returns a data frame. It can have 1-3 columns, but always has the files column. abs_path and product_name columns are added if absPath and name are TRUE, respectively.

See Also

prism_archive_subset()

Examples

```
## Not run:
# Get prism data names, used in many other prism* functions
get_prism_dailys(
  type="tmean",
  minDate = "2013-06-01",
  maxDate = "2013-06-14",
  keepZip = FALSE
)
prism_archive_ls()

## End(Not run)
```

---

prism_archive_subset *Subsets PRISM folders on the disk*

Description

prism_archive_subset() subsets the PRISM folders stored on disk by type, temporal period, and date. It looks through all of the PRISM data that have been downloaded in the prism archive (prism_get_dl_dir()) and returns the subset based on the specified type, temp_period, and dates.

Usage

```
prism_archive_subset(
  type,
  temp_period,
  years = NULL,
  mon = NULL,
  minDate = NULL,
```

```
  maxDate = NULL,
  dates = NULL,
  resolution = NULL
)
```

Arguments

type	The type of data you want to subset. Must be "ppt", "tmean", "tmin", "tmax", "tdmean", "vpdmin", "vpdmax", "solclear", "solslope", "soltotal", or "soltrans".
temp_period	The temporal period to subset. Must be "annual", "monthly", "daily", "monthly normals", or "annual normals".
years	Valid numeric year, or vector of years.
mon	Valid numeric month, or vector of months.
minDate	Date to start subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
maxDate	Date to end subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
dates	A vector of daily dates to subset. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
resolution	The spatial resolution of the data, must be either "4km" or "800m". Required for all temporal periods.

Details

`temp_period` must be specified so the function can distinguish between wanting annual data or wanting monthly data for a specified year. For example `prism_archive_subset("tmean", "annual", years = 2012)` would provide only one folder: the annual average temperature for 2012. However, `prism_archive_subset("tmean", "monthly", years = 2012)` would provide 12 folders: each monthly tmean folder for 2012.

`temp_period`, `years`, and `mon` can be combined in various different ways to obtain different groupings of data. `years`, `mon`, and the daily specifiers (`minDate`/`maxDate` or `dates`) are optional. Not specifying any of those would result in getting all annual, monthly, or daily data.

`minDate`/`maxDate` or `dates` should only be specified for a `temp_period` of "daily". Additionally, only `dates`, or `minDate` and `maxDate`, should be specified, but all three should not be specified. Nor should the daily arguments be combined with `years` and/or `mon`. For example, if daily folders are desired, then specify `years` and/or `mon` to get all days for those years and months **or** specify the specific dates using `minDate`/`maxDate` or `dates`

Value

A character vector of the folders that meet the type and temporal period specified. `character(0)` is returned if no folders are found that meet the specifications.

See Also

prism_archive_ls()

Examples

```
## Not run:
# get all annual tmin
prism_archive_subset("tmin", "annual")
# get only 2000-2015 annual tmin at 800m resolution
prism_archive_subset("tmin", "annual", years = 2000:2015, resolution = "800m")

# get monthly precipitation for 2000-2010
prism_archive_subset("ppt", "monthly", years = 2000:2010)
# get only June-August monthly precip data for 2000-2010 at 4km resolution
prism_archive_subset("ppt", "monthly", years = 2000:2010, mon = 6:8, resolution = "4km")

# get all daily tmax for July-August in 2010
prism_archive_subset("tmax", "daily", years = 2010, mon = 7:8)
# get 800m daily tmax for July-August in 2010
prism_archive_subset("tmax", "daily", years = 2010, mon = 7:8, resolution = "800m")
# same as:
prism_archive_subset(
  "tmax",
  "daily",
  minDate = "2010-07-01",
  maxDate = "2010-08-31",
  resolution = "800m"
)

# get the 4km 30-year average precip for January and February
prism_archive_subset("ppt", "monthly normals", mon = 1:2, resolution = "4km")

## End(Not run)
```

---

prism_archive_verify *Check the integrity of downloaded PRISM data*

Description

prism_archive_verify() checks the data in the prism archive to ensure it is valid, or at least can be read into R, i.e., it is not corrupt. The prism variable type, time period, etc. is specified the same as for prism_archive_subset(). Any files that are not readable can automatically be re-downloaded.

check_corrupt() is the deprecated version of prism_archive_verify()

Usage

```
prism_archive_verify(
  type,
  temp_period,
  years = NULL,
  mon = NULL,
  minDate = NULL,
  maxDate = NULL,
  dates = NULL,
  resolution = NULL,
  download_corrupt = TRUE,
  keepZip = TRUE
)

check_corrupt(type, minDate = NULL, maxDate = NULL, dates = NULL)
```

Arguments

type	The type of data you want to subset. Must be "ppt", "tmean", "tmin", "tmax", "tdmean", "vpdmin", "vpdmax", "solclear", "solslope", "soltotal", or "soltrans".
temp_period	The temporal period to subset. Must be "annual", "monthly", "daily", "monthly normals", or "annual normals".
years	Valid numeric year, or vector of years.
mon	Valid numeric month, or vector of months.
minDate	Date to start subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
maxDate	Date to end subsetting daily data. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
dates	A vector of daily dates to subset. Must be specified in a valid iso-8601 (e.g. YYYY-MM-DD) format. May be provided as either a character or base::Date class.
resolution	The spatial resolution of the data, must be either "4km" or "800m". Required for all temporal periods.
download_corrupt	If TRUE, then any unreadable prism data are automatically re-downloaded.
keepZip	If TRUE, leave the downloaded zip files in your 'prism.path', if FALSE, they will be deleted.

Details

Under the hood, it uses `raster::stack()` and then `raster::rasterToPoints()` to determine if the bil files are readable. If both those files are able to successfully read the files, they are assumed to be valid/readable.

Value

`prism_archive_verify()` returns `TRUE` if all data are readable. Any prism data that are not readable are returned (folder names), whether they are re-downloaded or not.

`check_corrupt()` returns `logical` indicating whether the process succeeded.

Examples

```
## Not run:
# check all annual precipitation data from 2000-2023 are readable
# x will contain any corrupt files, or be TRUE if they are all readable
x <- prism_archive_verify('ppt', 'annual', 2000:2023, resolution = "4km")

## End(Not run)
```

---

prism_check *Check if prism files exist*

Description

Helper function to check if files already exist in the prism download directory. Determines if files have **not** been downloaded yet, i.e., returns `TRUE` if they do not exist.

Usage

```
prism_check(prismfiles, lgl = FALSE, pre81_months = NULL)
```

Arguments

`prismfiles`	a list of full prism file names ending in ".zip".
`lgl`	`TRUE` returns a logical vector indicating those not yet downloaded; `FALSE` returns the file names that are not yet downloaded.
`pre81_months`	Numeric vector of months that will be downloaded, if downloading data before 1981. This is so that the existence of the data can be correctly checked, as the file includes all monthly data for a given year.

Value

a character vector of file names that are not yet downloaded or a logical vector indication those not yet downloaded.

prism_set_dl_dir *Set, check, and get prism download directory*

Description

prism_set_dl_dir() sets the directory that downloaded prism data will be saved to. The prism download directory is saved in the "prism.path" option.

prism_get_dl_dir() gets the folder that prism data will be saved to. It is a wrapper around getOption("prism.path") so the user does not have to remember the option name.

prism_check_dl_dir() checks that prism download folder has been set. If it has not been set, and in interactive mode, then prompt user to specify the download location. If not in interactive mode and it has not been set then it will stop and post an error.

path_check() is a deprecated version of prism_check_dl_dir().

Usage

```
prism_set_dl_dir(path, create = TRUE)

prism_get_dl_dir()

prism_check_dl_dir()

path_check()
```

Arguments

path	The path that prism data will be unzipped into.
create	Boolean that determines if the path will be created if it does not already exist.

Value

Invisibly returns path

Examples

```
## Not run:
prism_set_dl_dir('.')
prism_set_dl_dir('~/prism_data')

## End(Not run)

prism_get_dl_dir()


## Not run:
prism_check_dl_dir()
```

```
## End(Not run)
```

Index