Package 'portalr'

August 28, 2020

Title Create Useful Summaries of the Portal Data

Version 0.3.5

Description Download and generate summaries for the rodent, plant, ant, and weather data from the Portal Project. Portal is a long-term (and ongoing) experimental monitoring site in the Chihuahua desert. The raw data files can be found at <https://github.com/weecology/portaldata>.

License MIT + file LICENSE

URL https://weecology.github.io/portalr/, https://github.com/weecology/portalr

BugReports https://github.com/weecology/portalr/issues

Depends R (>= 3.2.3)

Imports clipr, clisymbols, crayon, dplyr, forecast, gh (>= 1.1.0), httr, lubridate, lunar, magrittr, rlang, tidyr, tidyselect (>= 1.0.0), zoo

Suggests covr, cowplot, ggplot2, httptest, knitr, pkgdown, rmarkdown, testthat, usethis

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

NeedsCompilation no

Author Glenda M. Yenni [aut, cre] (<https://orcid.org/0000-0001-6969-1848>), Hao Ye [aut] (<https://orcid.org/0000-0002-8630-1458>), Erica M. Christensen [aut] (<https://orcid.org/0000-0002-5635-2502>), Juniper L. Simonis [aut] (<https://orcid.org/0000-0001-9798-0460>), Ellen K. Bledsoe [aut] (<https://orcid.org/0000-0002-3629-7235>), Renata M. Diaz [aut] (<https://orcid.org/0000-0003-0803-4734>), Shawn D. Taylor [aut] (<https://orcid.org/0000-0002-6178-6903>), Ethan P, White [aut] (<https://orcid.org/0000-0001-6728-7745>), S.K. Morgan Ernest [aut] (<https://orcid.org/0000-0002-6026-8530>), Weecology [cph]

Maintainer Glenda M. Yenni <glenda@weecology.org>

Repository CRAN

Date/Publication 2020-08-28 11:00:07 UTC

R topics documented:

`add_seasons` *Add Seasons*

Description

Higher-order data summaries, by 6-month seasons, 3-month seasons, or year. Also applies specified functions to the specified summary level.

`yearly` generates a table of yearly means

Usage

```
add_seasons(
  data,
  level = "site",
  season_level = 2,
  date_column = "yearmon",
  summary_funs = NA,
  path = get_default_data_path(),
  download_if_missing = TRUE,
  clean = TRUE
)

yearly(...)
```

Arguments

data	data frame containing columns: date, period, newmoonnumber, or year and month
level	summarize by "Plot", "Treatment", or "Site"
season_level	either year, 2: winter = Oct-March summer = April-Sept 4: winter = Dec-Feb spring = March-May summer = Jun-Aug fall = Sep-Nov
date_column	either "date" (must be in format "y-m-d"), "period", "newmoonnumber", or "yearmon" (data must contain "year" and "month")
summary_funs	A function specified by its name (e.g. "mean"). Default is NA (returned with seasons added but not summarized).
path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
clean	logical, load only QA/QC rodent data (TRUE) or all data (FALSE)
...	arguments passed to add_seasons

Value

a data.frame with additional "season" and "year" column, and other columns summarized as specified. If no summary function is specified, "season" and "year" columns are added to original dataframe, as well as a "seasonyear" column which correctly assigns months to seasons for grouping (eg December 2000 in winter 2001, rather than winter 2000).

Examples

```
yearly(abundance(path = "repo", time = "newmoon"),
       date_column = "newmoonnumber", path = "repo")
```

bait_presence_absence *Ant Bait Presence Absence*

Description

Get ant species presence/absence by year/plot/stake from bait census data

Bait census data is more consistent over time than the colony census data. This function assumes that all species present in at least one census were censused in all years.

Usage

```
bait_presence_absence(
  path = get_default_data_path(),
  level = "Site",
  download_if_missing = TRUE,
  quiet = FALSE
)
```

Arguments

path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
level	level at which to summarize data: 'Site', 'Plot', or 'Stake'
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
quiet	logical, whether to run without version messages

Value

data frame with year, species, (plot if applicable), and presence [1, 0]

check_default_data_path *Manage the default path for downloading Portal Data into*

Description

check_default_data_path checks if a default data path is set, and prompts the user to set it if it is missing.

get_default_data_path gets the value of the data path environmental variable

use_default_data_path has 3 steps. First, it checks for the presence of a pre-existing setting for the environmental variable. Then it checks if the folder exists and creates it, if needed. Then it provides instructions for setting the environmental variable.

Usage

```
check_default_data_path(
  ENV_VAR = "PORTALR_DATA_PATH",
  MESSAGE_FUN = message,
  DATA_NAME = "Portal data"
)

get_default_data_path(fallback = "~", ENV_VAR = "PORTALR_DATA_PATH")

use_default_data_path(path = NULL, ENV_VAR = "PORTALR_DATA_PATH")
```

Arguments

ENV_VAR	the environmental variable to check (by default '"PORTALR_DATA_PATH"')
MESSAGE_FUN	the function to use to output messages
DATA_NAME	the name of the dataset to use in output messages
fallback	the default value to use if the setting is missing
path	Folder into which data will be downloaded

Value

FALSE if there is no path set, TRUE otherwise

None

---

check_for_newer_data *Check for latest version of data files*

Description

Check the latest version against the data that exists on the GitHub repo

Usage

```
check_for_newer_data(path = get_default_data_path(), mustWork = TRUE)
```

Arguments

path	Folder in which data will be checked
mustWork	logical: if TRUE then an error is given if the result cannot be determined; if NA then a warning.

Value

bool TRUE if there is a newer version of the data online

clean_plant_data *Do basic cleaning of Portal plant data*

Description

This function does basic quality control of the Portal plant data. It is mainly called from summarize_plant_data, with several arguments passed along.

The specific steps it does are, in order: (1) correct species names according to recent vouchers, if requested (2) restrict species to annuals or non-woody (3) remove records for unidentified species (5) exclude the plots that aren't long-term treatments

Usage

```
clean_plant_data(
  data_tables,
  type = "All",
  unknowns = FALSE,
  correct_sp = TRUE
)
```

Arguments

data_tables	the list of data_tables, returned from calling load_plant_data
type	specify subset of species; If type=Annuals, removes all non-annual species. If type=Non-woody, removes shrub and subshrub species If type=Perennials, returns all perennial species (includes shrubs and subshrubs) If type=Shrubs, returns only shrubs and subshrubs If type=Winter-annual, returns all annuals found in winter IF type=Summer-annual, returns all annuals found in summer
unknowns	either removes all individuals not identified to species (unknowns = FALSE) or sums them in an additional column (unknowns = TRUE)
correct_sp	T/F whether or not to use likely corrected plant IDs, passed to rename_species_plants

clean_rodent_data *Do basic cleaning of Portal rodent data*

Description

This function does basic quality control of the Portal rodent data. It is mainly called from summarize_rodent_data, with several arguments passed along.

The specific steps it does are, in order: (1) add in missing weight data (2) remove records with "bad" period codes or plot numbers (3) remove records for unidentified species (4) exclude non-granivores (5) exclude incomplete trapping sessions (6) exclude the plots that aren't long-term treatments

Usage

```
clean_rodent_data(
  rodent_data,
  species_table,
  fillweight = FALSE,
  type = "Rodents",
  unknowns = FALSE
)
```

Arguments

rodent_data	the raw rodent data table
species_table	the species table
fillweight	specify whether to fill in unknown weights with other records from that individual or species, where possible
type	specify subset of species; either all "Rodents" or only "Granivores"
unknowns	either removes all individuals not identified to species (unknowns = FALSE) or sums them in an additional column (unknowns = TRUE)

---

colony_presence_absence

Ant Colony Presence Absence

Description

Get ant species presence/absence by year/plot/stake from colony census data

Anomalies in ant colony census protocol over the years means that it can be difficult to discern true absences of all species in all years. This function uses information from Portal_ant_species.csv and Portal_ant_dataflags.csv to predict true presence/absence of species per plot per year. If a more conservative estimate is desired, setting the argument 'rare_sp = T' will only include species we are confident were censused regularly. Setting 'rare_sp = F' may include some false absences, since it is unknown if some rare species were censused in all years. Unknowns may also be excluded from output if desired.

Usage

```
colony_presence_absence(
  path = get_default_data_path(),
  level = "Site",
  rare_sp = FALSE,
  unknowns = FALSE,
  download_if_missing = TRUE,
  quiet = FALSE
)
```

Arguments

path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
level	level at which to summarize data: 'Site', 'Plot', or 'Stake'
rare_sp	include rare species (T) or not (F). Rare species may or may not have been censused in all years. Setting 'rare_sp = FALSE' gives a more conservative estimate of presence/absence
unknowns	include unknown species (TRUE) or not (FALSE). Unknowns include those only identified to genus.
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
quiet	logical, whether to run without version messages

Value

data frame with year, species, (plot if applicable), and presence [1, 0, NA]

---

download_observations *Download the PortalData repo*

Description

This downloads the latest portal data regardless if they are actually updated or not. TODO: incorporate data retriever into this when it's pointed at the github repo

Usage

```
download_observations(
  path = get_default_data_path(),
  version = "latest",
  from_zenodo = FALSE,
  quiet = FALSE
)
```

Arguments

path	Folder into which data will be downloaded
version	Version of the data to download (default = "latest")
from_zenodo	logical; if 'TRUE', get info from Zenodo, otherwise GitHub
quiet	logical, whether to download data silently

Value

None

Examples

```
download_observations()
download_observations("~/old-data", version = "1.50.0")
```

`fcast_ndvi` *Forecast ndvi using a seasonal auto ARIMA*

Description

Forecast ndvi using a seasonal auto ARIMA

Usage

```
fcast_ndvi(hist_ndvi, level, lead, moons = NULL)
```

Arguments

`hist_ndvi`	historic ndvi data
`level`	specify "monthly" or "newmoon"
`lead`	number of steps forward to forecast
`moons`	moon data (required if level = "newmoon")

Details

ndvi values are forecast using auto.arima with seasonality (using a Fourier transform)

Value

a data.frame with time and ndvi values

`fill_missing_ndvi` *Fill in historic ndvi data to the complete timeseries being fit*

Description

Fill in historic ndvi data to the complete timeseries being fit

Usage

```
fill_missing_ndvi(ndvi, level, last_time, moons = NULL)
```

Arguments

ndvi	ndvi data
level	specify "monthly" or "newmoon"
last_time	the last time step to have been completed
moons	moon data (required if level = "newmoons" and forecasts are needed)

Details

missing values during the time series are replaced using na.interp, missing values at the end of the time series are forecast using auto.arima with seasonality (using Fourier transform)

Value

a data.frame with time and ndvi values

---

find_incomplete_censuses

Period code for incomplete censuses

Description

Determines incomplete censuses by finding dates when some plots were trapped, but others were not.

Usage

```
find_incomplete_censuses(trapping_table, min_plots, min_traps)
```

Arguments

trapping_table	Data_table of when plots were censused.
min_plots	minimum number of plots within a period for an observation to be included
min_traps	minimum number of traps for a plot to be included

Value

Data.table of period codes when not all plots were trapped.

format_code *Format content as code*

Description

Format content as code

Usage

```
format_code(...)
```

Arguments

...	strings

Value

a formatted string to output

format_todo *Format content as an action to be performed by the user*

Description

Format content as an action to be performed by the user

Usage

```
format_todo(...)
```

Arguments

...	strings

Value

a formatted string to output

format_value *Format content as a variable value*

Description

Format content as a variable value

Usage

```
format_value(...)
```

Arguments

...	strings

Value

a formatted string to output

get_dataset_citation *Return Citation for Portal Data*

Description

Return Citation for Portal Data

Usage

```
get_dataset_citation()
```

Value

An object of class "citation". For more details, see ‘citation()‘

get_data_versions *get version and download info for PortalData*

Description

Check either Zenodo or GitHub for the version and download link for PortalData.

Usage

```
get_data_versions(from_zenodo = FALSE, halt_on_error = FALSE)
```

Arguments

from_zenodo	logical; if 'TRUE', get info from Zenodo, otherwise GitHub
halt_on_error	logical; if 'FALSE', return NULL on errors, otherwise whatever got returned (could be an error or warning)

Value

A data.frame with two columns, 'version' (string with the version #) and 'zipball_url' (download URLs for the corresponding zipped release).

get_future_moons *Get future moon dates*

Description

Get next 12 new moon dates and assign newmoon numbers for forecasting

Usage

```
get_future_moons(moons, num_future_moons = 12)
```

Arguments

moons	current newmoonnumber table
num_future_moons	number of future moons to get

Value

expected moons table for 12 future new moons

load_datafile *read in a raw datafile from the downloaded data or the GitHub repo*

Description

does checking for whether a particular datafile exists and then reads it in, using na_strings to determine what gets converted to NA. It can also download the dataset if it's missing locally.

Usage

```
load_datafile(
  datafile,
  na.strings = "",
  path = get_default_data_path(),
  download_if_missing = TRUE,
  quiet = TRUE
)
```

Arguments

datafile	the path to the datafile within the folder for Portal data
na.strings	a character vector of strings which are to be interpreted as NA values. Blank fields are also considered to be missing values in logical, integer, numeric and complex fields. Note that the test happens *after* white space is stripped from the input, so na.strings values may need their own white space stripped in advance.
path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
quiet	logical, whether to perform operations silently

Examples

```
rodent_species <- load_datafile("Rodents/Portal_rodent_species.csv")
```

load_rodent_data *Read in the Portal data files*

Description

Loads Portal data files from either a user-defined path or the online Github repository. If the user-defined path is un- available, the default option is to download to that location.

load_rodent_data loads the rodent data files

load_plant_data loads the plant data files

load_ant_data loads the ant data files

load_trapping_data loads just the rodent trapping files

Usage

```
load_rodent_data(
  path = get_default_data_path(),
  download_if_missing = TRUE,
  clean = TRUE,
  quiet = FALSE
)

load_plant_data(
  path = get_default_data_path(),
  download_if_missing = TRUE,
  quiet = FALSE
)

load_ant_data(
  path = get_default_data_path(),
  download_if_missing = TRUE,
  quiet = FALSE
)

load_trapping_data(
  path = get_default_data_path(),
  download_if_missing = TRUE,
  clean = TRUE,
  quiet = FALSE
)
```

Arguments

path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it

clean	logical, load only QA/QC rodent data (TRUE) or all data (FALSE)
quiet	logical, whether to run without version messages

Value

load_rodent_data returns a list of 5 dataframes:

rodent_data	raw data on rodent captures
species_table	species code, names, types
trapping_table	when each plot was trapped
newmoons_table	pairs census periods with newmoons
plots_table	rodent treatment assignments for each plot

load_plant_data returns a list of 7 dataframes:

quadrat_data	raw plant quadrat data
species_table	species code, names, types
census_table	indicates whether each quadrat was counted in each census; area of each quadrat
date_table	start and end date of each plant census
plots_table	rodent treatment assignments for each plot
transect_data	raw plant transect data with length and height (2015-present)
oldtransect_data	raw plant transect data as point counts (1989-2009)

load_ant_data returns a list of 4 dataframes:

bait_data	raw ant bait data
colony_data	raw ant colony data
species_table	species code, names, types
plots_table	treatment assignments for each plot

load_trapping_data returns a list of 2 dataframes:

trapping_table	when each plot was trapped
newmoons_table	pairs census periods with newmoons

Examples

```
portal_data <- load_rodent_data("repo")
```

```
portal_plant_data <- load_plant_data("repo")
```

```
portal_ant_data <- load_ant_data("repo")


trapping_data <- load_trapping_data("repo")
```

`ndvi` *NDVI by calendar month or lunar month*

Description

Summarize NDVI data to monthly or lunar monthly level

Usage

```
ndvi(
  level = "monthly",
  sensor = "landsat",
  fill = FALSE,
  path = get_default_data_path(),
  download_if_missing = TRUE
)
```

Arguments

`level`	specify "monthly" or "newmoon"
`sensor`	specify "landsat", "modis", "gimms", or "all"
`fill`	specify if missing data should be filled, passed to `fill_missing_ndvi`
`path`	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
`download_if_missing`	if the specified file path doesn't have the PortalData folder, then download it

---

`phenocam` *Phenocam data products by day, calendar month, or lunar month*

Description

Summarize phenocam data products to either daily, monthly, or lunar monthly level.

Usage

```
phenocam(level = "daily", path = get_default_data_path())
```

Arguments

level	specify 'monthly', 'daily', or 'newmoon'
path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository

---

portalr *Creates summaries of the Portal data*

Description

This package is designed to be an interface to the Portal data, which resides online at https://github.com/weecology/portalData. Its contains a set of functions to download, clean, and summarize the data.

---

shrub_cover *Generate percent cover from Portal plant transect data*

Description

This function calculates percent cover from transect data. It handles the pre-2015 data differently from the current transects, becase they are collected differently. But it returns a single time-series with all years of transect data available. It also returns mean height beginning in 2015.

Usage

```
shrub_cover(
  path = get_default_data_path(),
  type = "Shrubs",
  plots = "all",
  unknowns = FALSE,
  correct_sp = TRUE,
  download_if_missing = TRUE,
  quiet = FALSE
)
```

Arguments

path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
type	specify subset of species; If type=Annuals, removes all non-annual species. If type=Summer Annuals, returns all annual species that can be found in the summer If type=Winter Annuals, returns all annual species that can be found in the winter If type=Non-woody, removes shrub and subshrub species If type=Perennials, returns all perennial species (includes shrubs and subshrubs) If type=Shrubs, returns only shrubs and subshrubs

plots	specify subset of plots; can be a vector of plots, or specific sets: "all" plots or "Longterm" plots (plots that have had the same treatment for the entire time series)
unknowns	either removes all individuals not identified to species (unknowns = FALSE) or sums them in an additional column (unknowns = TRUE)
correct_sp	correct species names suspected to be incorrect in early data (T/F)
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
quiet	logical, whether to run without version messages

Value

a data.frame of percent cover and mean height

`summarize_individual_rodents`

Return cleaned Portal rodent individual data

Description

This function cleans and subsets the data based on a number of arguments. It returns stake number and individual level data.

Usage

```
summarize_individual_rodents(
  path = get_default_data_path(),
  clean = TRUE,
  type = "Rodents",
  length = "all",
  unknowns = FALSE,
  time = "period",
  fillweight = FALSE,
  min_plots = 1,
  min_traps = 1,
  download_if_missing = TRUE,
  quiet = FALSE
)

summarise_individual_rodents(
  path = get_default_data_path(),
  clean = TRUE,
  type = "Rodents",
  length = "all",
  unknowns = FALSE,
  time = "period",
```

```
  fillweight = FALSE,
  min_plots = 1,
  min_traps = 1,
  download_if_missing = TRUE,
  quiet = FALSE
)
```

Arguments

path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
clean	logical, load only QA/QC rodent data (TRUE) or all data (FALSE)
type	specify subset of species; either all "Rodents" or only "Granivores"
length	specify subset of plots; use "All" plots or only "Longterm" plots (to be deprecated)
unknowns	either removes all individuals not identified to species (unknowns = FALSE) or sums them in an additional column (unknowns = TRUE)
time	specify the format of the time index in the output, either "period" (sequential Portal surveys), "newmoon" (lunar cycle numbering), "date" (calendar date), or "all" (for all time indices)
fillweight	specify whether to fill in unknown weights with other records from that individual or species, where possible
min_plots	minimum number of plots within a period for an observation to be included
min_traps	minimum number of traps for a plot to be included
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
quiet	logical, whether to run without producing messages

Value

a data.frame

`summarize_plant_data` *Generate summaries of Portal plant data*

Description

This function is a generic interface into creating summaries of the Portal plant species data. It contains a number of arguments to specify both the kind of data to summarize, at what level of aggregation, various choices for dealing with data quality, and output format.

`plant_abundance` generates a table of plant abundance

Usage

```
summarize_plant_data(
  path = get_default_data_path(),
  level = "Site",
  type = "All",
  length = "all",
  plots = length,
  unknowns = FALSE,
  correct_sp = TRUE,
  shape = "flat",
  output = "abundance",
 na_drop = switch(tolower(level), quadrat = FALSE, plot = FALSE, treatment = TRUE,
    site = TRUE, TRUE),
 zero_drop = switch(tolower(level), quadrat = TRUE, plot = FALSE, treatment = TRUE,
    site = TRUE, TRUE),
  min_quads = 1,
  effort = TRUE,
  download_if_missing = TRUE,
  quiet = FALSE
)

plant_abundance(..., shape = "flat")

summarise_plant_data(
  path = get_default_data_path(),
  level = "Site",
  type = "All",
  length = "all",
  plots = length,
  unknowns = FALSE,
  correct_sp = TRUE,
  shape = "flat",
  output = "abundance",
 na_drop = switch(tolower(level), quadrat = FALSE, plot = FALSE, treatment = TRUE,
    site = TRUE, TRUE),
 zero_drop = switch(tolower(level), quadrat = TRUE, plot = FALSE, treatment = TRUE,
    site = TRUE, TRUE),
  min_quads = 1,
  effort = TRUE,
  download_if_missing = TRUE,
  quiet = FALSE
)
```

Arguments

path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
level	summarize by "Plot", "Treatment", "Site", or "Quadrat"

type	specify subset of species; If type=Annuals, removes all non-annual species. If type=Summer Annuals, returns all annual species that can be found in the summer If type=Winter Annuals, returns all annual species that can be found in the winter If type=Non-woody, removes shrub and subshrub species If type=Perennials, returns all perennial species (includes shrubs and subshrubs) If type=Shrubs, returns only shrubs and subshrubs
length	specify subset of plots; use "All" plots or only "Longterm" plots (to be deprecated)
plots	specify subset of plots; can be a vector of plots, or specific sets: "all" plots or "Longterm" plots (plots that have had the same treatment for the entire time series)
unknowns	either removes all individuals not identified to species (unknowns = FALSE) or sums them in an additional column (unknowns = TRUE)
correct_sp	correct species names suspected to be incorrect in early data (T/F)
shape	return data as a "crosstab" or "flat" list
output	specify whether to return "abundance", or "cover" [cover data starts in summer 2015]
na_drop	logical, drop NA values (representing insufficient sampling)
zero_drop	logical, drop 0s (representing sufficient sampling, but no detections)
min_quads	numeric [1:16], minimum number of quadrats (out of 16) for a plot to be included
effort	logical as to whether or not the effort columns should be included in the output
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
quiet	logical, whether to run without version messages
...	arguments passed to summarize_plant_data

Value

a data.frame in either "long" or "wide" format, depending on the value of ‘shape‘

Examples

```
plant_abundance("repo")
```

summarize_rodent_data *Generate summaries of Portal rodent data*

Description

This function is a generic interface into creating summaries of the Portal rodent species data. It contains a number of arguments to specify the kind of data to summarize (at what level of aggregation) and various choices for dealing with data quality, and output format.

abundance generates a table of rodent abundance

* biomass() generates a table of rodent biomass

* energy() generates a table of rodent energy (computed as 5.69 * (biomass ^ 0.75) after White et al 2004)

Usage

```
summarize_rodent_data(
  path = get_default_data_path(),
  clean = TRUE,
  level = "Site",
  type = "Rodents",
  length = "all",
  plots = length,
  unknowns = FALSE,
  shape = "crosstab",
  time = "period",
  output = "abundance",
  fillweight = (output != "abundance"),
  na_drop = TRUE,
 zero_drop = switch(tolower(level), plot = FALSE, treatment = TRUE, site = TRUE),
  min_traps = 1,
  min_plots = 24,
  effort = FALSE,
  download_if_missing = TRUE,
  quiet = FALSE,
  include_unsampled = FALSE
)

abundance(...)

biomass(...)

energy(...)

summarise_rodent_data(
  path = get_default_data_path(),
  clean = TRUE,
```

```
  level = "Site",
  type = "Rodents",
  length = "all",
  plots = length,
  unknowns = FALSE,
  shape = "crosstab",
  time = "period",
  output = "abundance",
  fillweight = (output != "abundance"),
  na_drop = TRUE,
 zero_drop = switch(tolower(level), plot = FALSE, treatment = TRUE, site = TRUE),
  min_traps = 1,
  min_plots = 24,
  effort = FALSE,
  download_if_missing = TRUE,
  quiet = FALSE,
  include_unsampled = FALSE
)
```

Arguments

path	either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository
clean	logical, load only QA/QC rodent data (TRUE) or all data (FALSE)
level	summarize by "Plot", "Treatment", or "Site"
type	specify subset of species; either all "Rodents" or only "Granivores"
length	specify subset of plots; use "All" plots or only "Longterm" plots (to be deprecated)
plots	specify subset of plots; can be a vector of plots, or specific sets: "all" plots or "Longterm" plots (plots that have had the same treatment for the entire time series)
unknowns	either removes all individuals not identified to species (unknowns = FALSE) or sums them in an additional column (unknowns = TRUE)
shape	return data as a "crosstab" or "flat" list
time	specify the format of the time index in the output, either "period" (sequential Portal surveys), "newmoon" (lunar cycle numbering), "date" (calendar date), or "all" (for all time indices)
output	specify whether to return "abundance", or "biomass", or "energy"
fillweight	specify whether to fill in unknown weights with other records from that individual or species, where possible
na_drop	logical, drop NA values (representing insufficient sampling)
zero_drop	logical, drop 0s (representing sufficient sampling, but no detections)
min_traps	minimum number of traps for a plot to be included
min_plots	minimum number of plots within a period for an observation to be included

effort	logical as to whether or not the effort columns should be included in the output
download_if_missing	if the specified file path doesn't have the PortalData folder, then download it
quiet	logical, whether to run without producing messages
include_unsampled	logical, overrides settings for 'na_drop' and 'zero_drop', setting both to FALSE
...	arguments passed to summarize_rodent_data

Value

a data.frame in either "long" or "wide" format, depending on the value of 'shape'

Examples

```
abundance("repo")


biomass("repo")


energy("repo")
```

---

weather *Weather by day, calendar month, or lunar month*

Description

Summarize hourly weather data to either daily, monthly, or lunar monthly level.

Usage

```
weather(
  level = "daily",
  fill = FALSE,
  horizon = 360,
  temperature_limit = 4,
  path = get_default_data_path()
)
```

Arguments

level	specify 'monthly', 'daily', or 'newmoon'
fill	specify if missing data should be filled, passed to fill_missing_weather
horizon	Horizon (number of days) to use when calculating cumulative values (eg warm weather precip)

`temperature_limit`
: Temperature limit (in C) to use when calculating cumulative values (eg warm weather precip)

`path`
: either the file path that contains the PortalData folder or "repo", which then pulls data from the PortalData GitHub repository

Index