

Package: PEcAn.data.atmosphere (via r-universe)

October 16, 2026

Type Package

Title PEcAn Functions Used for Managing Climate Driver Data

Version 1.9.1.9000

Description The Predictive Ecosystem Carbon Analyzer (PEcAn) is a scientific workflow management tool that is designed to simplify the management of model parameterization, execution, and analysis. The PEcAn.data.atmosphere package converts climate driver data into a standard format for models integrated into PEcAn. As a standalone package, it provides an interface to access diverse climate data sets.

URL <https://pecanproject.github.io>

BugReports <https://github.com/PecanProject/pecan/issues>

Depends R (>= 4.1.0)

Imports abind (>= 1.4.5), amerifluxr, assertthat, arrow, curl, dplyr (>= 0.8.1), foreach, geonames (> 0.998), glue, httr, jsonlite, lubridate (>= 1.6.0), MASS, mgcv, ncdf4 (>= 1.15), PEcAn.DB, PEcAn.logger, PEcAn.remote, PEcAn.utils, purrr (>= 0.2.3), raster, REddyProc, reshape2, rlang (>= 0.2.0), sf, sp, stringr (>= 1.1.0), suntools, terra, tibble, tidyr, tidyselect, truncnorm, units, utils, XML (>= 3.98-1.4), xts, zoo

Suggests doParallel, ecmwfr (>= 2.0.0), doSNOW, furrr, future, getPass, ggplot2, knitr (>= 1.50), mockery, nneo, parallel, PEcAn.settings, progress, reticulate, rmarkdown, testthat (>= 3.1.7), withr

Remotes github::adokter/suntools, github::chuhousen/amerifluxr, github::ropensci/geonames, github::ropensci-archive/nneo

Additional_repositories <https://ropensci.r-universe.dev/>,
<https://adokter.r-universe.dev>

License BSD_3_clause + file LICENSE

Copyright Authors
LazyLoad yes
LazyData FALSE
VignetteBuilder knitr, rmarkdown
Encoding UTF-8
RoxygenNote 7.3.3
X-schema.org-keywords climate-data, data-standardization
Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libmagick++-dev gsfonts libicu-dev libuv1-dev libxml2-dev libnetcdf-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libx11-dev
Repository <https://pecanproject.r-universe.dev>
Date/Publication 2026-10-16 08:17:39 UTC
RemoteUrl <https://github.com/PecanProject/pecan>
RemoteRef HEAD
RemoteSha fdb9b89d6133f78473728f63b7c1b3deb4ee29ec
RemoteSubdir modules/data.atmosphere

Contents

AirDens	5
align.met	5
AmeriFlux_met_ensemble	7
Average_ERA5_2_GeoTIFF	9
build_cf_variables_table_url	9
cfmet.downscale.daily	10
cfmet.downscale.subdaily	11
cfmet.downscale.time	11
check_met_input_file	12
check_unit	13
closest_xy	13
col2ncvar	14
cos_solar_zenith_angle	14
debias.met.regression	15
debias_met	18
download.Ameriflux	19
download.AmerifluxLBL	20
download.CRUNCEP	21
download.ERA5_cds	22
download.FACE	24
download.Fluxnet2015	25
download.FluxnetLaThuile	26
download.Geostreams	27
download.GFDL	28
download.GLDAS	29

download.ICOS	30
download.MACA	31
download.MERRA	32
download.MsTMIP_NARR	33
download.NARR	34
download.NARR_site	35
download.NEONmet	36
download.NLDAS	37
download.NOAA_GEFS	38
download.PaleON	40
download.PaleON_ENS	41
download.raw.met.module	41
download.US_WCr	43
download.US_Wlef	43
download_grid	44
download_NOAA_GEFS_EFI	45
downscale_one_cfmet_day	46
downscale_repeat_6hr_to_half_hrly	46
downscale_repeat_6hr_to_hrly	47
downscale_ShortWave_to_half_hrly	47
downscale_ShortWave_to_hrly	48
downscale_solar_geom	49
downscale_solar_geom_halfhour	49
downscale_spline_to_half_hrly	50
downscale_spline_to_hrly	50
equation_of_time	51
ERA5_met_process	51
exner	52
extract.local.CMIP5	53
extract.local.NLDAS	54
extract.nc	55
extract.nc.ERA5	56
gen.subdaily.models	58
generate_narr_url	59
get.lv	60
get.ncvector	60
get.rh	61
get.vpd	61
get_cf_variables_table	62
get_clowderauth	63
get_NARR_thredds	63
get_narr_url	64
half_hour_downscale	65
latlon2lcc	65
latlon2narr	66
lightME	67
lm_ensemble_sims	67
load.cfmet	69

merge_met_variable	70
met.process	71
met.process.stage	72
met_temporal_downscale.Gaussian_ensemble	73
met2CF.ALMA	74
met2CF.Ameriflux	75
met2CF.AmerifluxLBL	76
met2CF.csv	77
met2CF.ERA5	79
met2CF.FACE	80
met2CF.Geostreams	81
met2CF.ICOS	82
met2CF.NARR	83
met2CF.PaleON	84
met2CF.PaleONregional	85
metgapfill	86
metgapfill.NOAA_GEFS	87
model.train	88
narr_fx_vars	89
nc.merge	89
noaa_grid_download	90
noaa_stage2	91
par2ppfd	92
pecan_standard_met_table	92
permute.nc	93
post_process	94
predict_subdaily_met	94
prepare_narr_year	96
process_gridded_noaa_download	97
qair2rh	98
read.register	98
read_narr_var	99
rh2qair	99
sat_vapor_pressure	100
save.betas	101
save.model	102
site.lst	103
site_from_tag	103
solarMJ2ppfd	104
spin.met	105
split_wind	106
step_means	107
subdaily_pred	108
sw2par	109
sw2ppfd	109
temporal.downscale.functions	110
temporal_downscale	111
upscale_met	112

wide2long

write_noaa_gefs_netcdf

112

113

Index

115

AirDens	<i>Air Density</i>
---------	--------------------

Description

estimate air density from pressure, temperature, and humidity

Usage

```
AirDens(pres, T, rv)
```

Arguments

pres	air pressure (pascals)
T	air temperature (Kelvin)
rv	humidity

Author(s)

Mike Dietze

align.met	<i>align.met</i>
-----------	------------------

Description

This script aligns meteorology datasets in at temporal resolution for debiasing & temporal downscaling. Note: The output here is stored in memory! Note: can probably at borrow from or adapt align_data.R in Benchmarking module, but it's too much of a black box at the moment.

Usage

```
align.met(  
  train.path,  
  source.path,  
  yrs.train = NULL,  
  yrs.source = NULL,  
  n.ens = NULL,  
  pair.mems = FALSE,  
  mems.train = NULL,  
  seed = Sys.Date(),  
  print.progress = FALSE  
)
```

Arguments

<code>train.path</code>	- path to the dataset to be used to downscale the data
<code>source.path</code>	- data to be bias-corrected aligned with training data (from align.met)
<code>yrs.train</code>	- (optional) specify a specific years to be loaded for the training data; prevents needing to load the entire dataset. If NULL, all available years will be loaded. If not null, should be a vector of numbers (so you can skip problematic years)
<code>yrs.source</code>	- (optional) specify a specific years to be loaded for the source data; prevents needing to load the entire dataset. If NULL, all available years will be loaded. If not null, should be a vector of numbers (so you can skip problematic years)
<code>n.ens</code>	- number of ensemble members to generate and save
<code>pair.mems</code>	- logical stating whether ensemble members should be paired in the case where ensembles are being read in in both the training and source data
<code>mems.train</code>	- (optional) string of ensemble identifiers that ensure the training data is read in a specific order to ensure consistent time series & proper error propagation. If null, members of the training data ensemble will be randomly selected and ordered. Specifying the ensemble members IDs (e.g. CCSM_001, CCSM_002) will ensure ensemble members are properly identified and combined.
<code>seed</code>	- specify seed so that random draws can be reproduced
<code>print.progress</code>	- if TRUE, prints progress bar

Details

Align meteorology datasets for debiasing

1. Assumes that both the training and source data are in **at least** daily resolution and each dataset is in a consistent temporal resolution being read from a single file (CF/Pecan format). For example, CMIP5 historical/p1000 runs where radiation drivers are in monthly resolution and temperature is in daily will need to be reconciled using one of the "met2CF" or "download" or "extract" functions 2. Default file structure: Ensembles members for a given site or set of simes are housed in a common folder with the site ID. Right now everything is based off of Christy's PaleON ensemble ID scheme where the site ID is a character string (e.g. HARVARD) followed the SOURCE data family (i.e. GCM) as a string and then the ensemble member ID as a number (e.g. 001). For example, the file path for a single daily ensemble member for PaleON is: "~/Desktop/Research/met_ensembles/data/met_ensembles/HARVARD/day/ensemble/001/csm1-1_004" with each year in a separate netcdf file inside of it. "bcc-csm1-1_004" is an example of an ensemble member ID that might be used if you are specifying mems.train.

Value

2-layered list (stored in memory) containing the training and source data that are now matched in temporal resolution have the specified number of ensemble members - `dat.train` (training dataset) and `dat.source` (source data to be downscaled or bias-corrected) are both lists that contain separate data frames for time indices and all available met variables with ensemble members in columns

Author(s)

Christy Rollinson

See AlsoOther debias - Debias & Align Meteorology Datasets into continuous time series: [debias.met.regression\(\)](#)

AmeriFlux_met_ensemble*Generate AmeriFlux meteorological ensembles*

Description

Downloads AmeriFlux data, applies ERA5 fallback for missing radiation and soil moisture, performs gap-filling, and generates ensembles. This function provides a complete pipeline from raw AmeriFlux data to CF-compliant ensemble meteorological files.

Usage

```
AmeriFlux_met_ensemble(  
  site_id,  
  start_date,  
  end_date,  
  outfolder,  
  ameriflux_username = "pecan",  
  ameriflux_useremail = "@",  
  overwrite = FALSE,  
  verbose = FALSE,  
  format = NULL,  
  n_ens = 10,  
  w_len = 30,  
  era5_user = NULL,  
  era5_key = NULL,  
  threshold = 0.5,  
  dirs = NULL,  
  ...  
)
```

Arguments

site_id	character. AmeriFlux site identifier (e.g, "US-Ha1") The 'SITE_ID' field in list of Ameriflux sites
start_date	character or date. Start date in "YYYY-MM-DD" format
end_date	character or date. End date in "YYYY-MM-DD" format
outfolder	character. Output directory path for ensemble files

ameriflux_username	character. AmeriFlux username for data access.
ameriflux_useremail	character. Email address for AmeriFlux authentication (must contain "@")
overwrite	logical. Whether to overwrite existing files. Default: FALSE
verbose	logical. Whether to print detailed logs. Default: FALSE
format	data frame or List. format specifications for CF conversion. Default: NULL The AmerifluxLBL format is Bety record 5000000002. which could be returned from <code>PEcAn.DB::query.format.vars(format.id=5000000002, bety = con)</code>
n_ens	integer. Number of ensemble members to generate. Default: 10
w_len	integer. Window length in days. Default: 20
era5_user	character. CDS user ID (UID) from your CDS profile. Required for authentication.
era5_key	character. CDS API key from your CDS profile. Required for authentication.
threshold	numeric. Coverage threshold (0-1) for triggering ERA5 fallback. Default: 0.5
dirs	named list. Optional configuration for existing data directory paths. Default: NULL
...	additional arguments passed to <code>download.AmerifluxLBL</code>

Value

A data frame with the paths to the generated ensemble files and their metadata.

Author(s)

Akash

Examples

```
## Not run:
result <- AmeriFlux_met_process(
  site_id = "US-Ha1",
  start_date = "2010-01-01",
  end_date = "2010-12-31",
  outfolder = "/path/to/output",
  ameriflux_username = "your_username",
  ameriflux_useremail = "your.email@domain.com",
  format = format,
  era5_user = "your_cds_user",
  era5_key = "your_cds_api_key",
  n_ens = 5,
  verbose = TRUE
)
```

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

Average_ERA5_2_GeoTIFF

Average_ERA5_2_GeoTIFF

Description

This function helps to average the ERA5 data based on the start and end dates, and convert it to the GeoTIFF file.

Usage

```
Average_ERA5_2_GeoTIFF(start.date, end.date, in.path, outdir)
```

Arguments

<code>start.date</code>	character: start point of when to average the data (e.g., 2012-01-01).
<code>end.date</code>	character: end point of when to average the data (e.g., 2021-12-31).
<code>in.path</code>	character: the directory where your ERA5 data stored (they should named as ERA5_YEAR.nc).
<code>outdir</code>	character: the output directory where the averaged GeoTIFF file will be generated.

Value

character: path to the exported GeoTIFF file.

Author(s)

Dongchen Zhang

`build_cf_variables_table_url`

Construct a URL to a specific version of the CF variables table XML

Description

This uses [sprintf](#) to construct the URL with the version number as the first argument.

Usage

```
build_cf_variables_table_url(
  version,
  url_format_string = paste0("https://cf-convention.github.io/",
    "Data/cf-standard-names/%d/src/", "src-cf-standard-name-table.xml")
)
```

Arguments

version CF variables table version number (integer/numeric)

url_format_string A format string passed to [sprintf](#). This should contain the entire target URL with the version number replaced by "%d", and *no other string substitutions*.

Value

Complete URL, as a string

Author(s)

Alexey Shiklomanov

cfmet.downscale.daily

Simple, Fast Daily to Hourly Climate Downscaling

Description

Based on weach family of functions but 5x faster than weachNEW, and requiring metric units (temperature in Kelvins on input and celsius on output, windspeed in kph, precip in mm, relative humidity as fraction). Derived from the weachDT function in the BioCro package.

Usage

```
cfmet.downscale.daily(dailymet, output.dt = 1, lat)
```

Arguments

dailymet data frame with climate variables

output.dt output timestep

lat latitude (for calculating solar radiation)

Value

weather file with subdaily timesteps

Author(s)

David LeBauer

`cfmet.downscale.subdaily`*Subdaily to hourly (or less) downscaling*

Description

Uses simple spline to interpolate variables with diurnal variability, otherwise uses averaging or repeating for variables with no clear diurnal pattern. For all variables except temperature, negative values are set to zero.

Usage

```
cfmet.downscale.subdaily(subdaily, output.dt = 1)
```

Arguments

<code>subdaily</code>	data frame with climate variables queried from load.cfmet
<code>output.dt</code>	output timestep. default is one hour

Value

weather file with subdaily met variables rescaled to output time step

Author(s)

David LeBauer

`cfmet.downscale.time` *Temporal downscaling of daily or subdaily CF met data*

Description

Temporal downscaling of daily or subdaily CF met data

Usage

```
cfmet.downscale.time(cfmet, output.dt = 1, lat = lat, ...)
```

Arguments

<code>cfmet</code>	data frame with CF variables generated by load.cfmet
<code>output.dt</code>	time step (hours) for output
<code>lat</code>	latitude (for calculating solar radiation)
<code>...</code>	ignored

Value

downscaled result

Author(s)

David LeBauer

check_met_input_file *Check a meteorology data file for compliance with the PEcAn standard*

Description

Check a meteorology data file for compliance with the PEcAn standard

Usage

```
check_met_input_file(
  metfile,
  variable_table = pecan_standard_met_table,
  required_vars = variable_table %>% dplyr::filter(.data$is_required) %>%
    dplyr::pull("cf_standard_name"),
  warn_unknown = TRUE
)
```

Arguments

metfile	Path of met file to check, as a scalar character.
variable_table	‘data.frame’ linking standard names to their units. Must contain columns “cf_standard_name” and “units”. Default is [pecan_standard_met_table].
required_vars	Character vector of required variables. Defaults to variables marked as required in ‘variable_table’.
warn_unknown	Logical. If ‘TRUE’ (default), throw a warning for variables not in ‘variable_table’. Otherwise, ignore unknown variables.

Value

‘data.frame’ summarizing the results of the tests.

Author(s)

Alexey Shiklomanov

<code>check_unit</code>	<i>Check that the unit of a variable in a NetCDF file is equivalent to the expected unit.</i>
-------------------------	---

Description

Check that the unit of a variable in a NetCDF file is equivalent to the expected unit.

Usage

```
check_unit(variable, nc, variable_table, warn_unknown = TRUE)
```

Arguments

<code>variable</code>	Name of target variable, as a length 1 character
<code>nc</code>	NetCDF object containing target variable
<code>variable_table</code>	‘data.frame’ linking standard names to their units. Must contain columns “cf_standard_name” and “units”. Default is [pecan_standard_met_table].
<code>warn_unknown</code>	Logical. If ‘TRUE’ (default), throw a warning for variables not in ‘variable_table’. Otherwise, ignore unknown variables.

Value

‘TRUE’ if unit is correct, or ‘try-error’ object if there is a mismatch.

Author(s)

Alexey Shiklomanov

<code>closest_xy</code>	<i>Given latitude and longitude coordinates, find NARR x and y indices</i>
-------------------------	--

Description

Given latitude and longitude coordinates, find NARR x and y indices

Usage

```
closest_xy(slat, slon, infolder, infile)
```

Arguments

<code>slat, slon</code>	site location, in decimal degrees
<code>infolder</code>	path to folder containing infile
<code>infile</code>	pattern to match for filename inside infile. Only the first file matching this pattern AND ending with '.nc' will be used

Author(s)

Betsy Cowdery, Ankur Desai

<code>col2ncvar</code>	<i>Create 'ncvar' object from variable name</i>
------------------------	---

Description

Create 'ncvar' object from variable name

Usage

```
col2ncvar(variable, dims)
```

Arguments

<code>variable</code>	CF variable name
<code>dims</code>	List of NetCDF dimension objects (passed to 'ncdf4::ncvar_def(..., dim)')

Value

'ncvar' object (from 'ncvar_def')

<code>cos_solar_zenith_angle</code>	<i>Cosine of Solar Zenith Angle</i>
-------------------------------------	-------------------------------------

Description

Calculates the cosine of the solar zenith angle based on the given parameters. This angle is crucial in determining the amount of solar radiation reaching a point on Earth.

Usage

```
cos_solar_zenith_angle(doy, lat, lon, dt, hr)
```

Arguments

doy	Day of year. Integer representing the day of the year (1-365).
lat	Latitude in degrees. Positive for the Northern Hemisphere and negative for the Southern Hemisphere.
lon	Longitude in degrees. Positive for East and negative for West.
dt	Time interval in seconds. Represents the duration over which the measurement is averaged or integrated.
hr	Hour of the day (0-23). Specifies the specific hour for which the calculation is made.

Details

For explanations of formulae, see <https://web.archive.org/web/20180307133425/http://www.itacanet.org/the-sun-as-a-source-of-energy/part-3-calculating-solar-angles/>

Value

Numeric value representing the cosine of the solar zenith angle.

Author(s)

Alexey Shiklomanov

References

"Understanding Solar Position and Solar Radiation" - RAMMB: [Link](<https://rammb.cira.colostate.edu/wmov1/v1/>)

Examples

```
cos_solar_zenith_angle(doy = 150, lat = 45, lon = -93, dt = 3600, hr = 12)
```

`debias.met.regression`

debias.met.regression

Description

This script debiases one dataset (e.g. GCM, re-analysis product) given another higher resolution product or empirical observations. It assumes input are in annual CF standard files that are generate from the pecan extract or download funcitons.

Usage

```

debias.met.regression(
  train.data,
  source.data,
  n.ens,
  vars.debias = NULL,
  CRUNCEP = FALSE,
  pair.anoms = TRUE,
  pair.ens = FALSE,
  uncert.prop = "mean",
  resids = FALSE,
  seed = Sys.Date(),
  outfolder,
  yrs.save = NULL,
  ens.name,
  ens.mems = NULL,
  force.sanity = TRUE,
  sanity.tries = 25,
  sanity.sd = 8,
  lat.in,
  lon.in,
  save.diagnostics = TRUE,
  path.diagnostics = NULL,
  parallel = FALSE,
  n.cores = NULL,
  overwrite = TRUE,
  verbose = FALSE
)

```

Arguments

<code>train.data</code>	- training data coming out of align.met
<code>source.data</code>	- data to be bias-corrected aligned with training data (from align.met)
<code>n.ens</code>	- number of ensemble members to generate and save for EACH source ensemble member
<code>vars.debias</code>	- which met variables should be debiased? if NULL, all variables in train.data
<code>CRUNCEP</code>	- flag for if the dataset being downscaled is CRUNCEP; if TRUE, special cases triggered for met variables that have been naively gapfilled for certain time periods
<code>pair.anoms</code>	- logical stating whether anomalies from the same year should be matched or not
<code>pair.ens</code>	- logical stating whether ensembles from train and source data need to be paired together (for uncertainty propogation)
<code>uncert.prop</code>	- method for error propogation for child ensemble members 1 ensemble member; options=c(random, mean); randomly strongly encouraged if n.ens>1

<code>resids</code>	- logical stating whether to pass on residual data or not *Not implemented yet
<code>seed</code>	- specify seed so that random draws can be reproduced
<code>outfolder</code>	- directory where the data should go
<code>yrs.save</code>	- what years from the source data should be saved; if NULL all years of the source data will be saved
<code>ens.name</code>	- what is the name that should be attached to the debiased ensemble
<code>ens.mems</code>	- what labels/numbers to attach to the ensemble members so we can gradually build bigger ensembles without having to do do giant runs at once; if NULL will be numbered 1:n.ens
<code>force.sanity</code>	- (logical) do we force the data to meet sanity checks?
<code>sanity.tries</code>	- how many time should we try to predict a reasonable value before giving up? We don't want to end up in an infinite loop
<code>sanity.sd</code>	- how many standard deviations from the mean should be used to determine sane outliers (default 8)
<code>lat.in</code>	- latitude of site
<code>lon.in</code>	- longitude of site
<code>save.diagnostics</code>	- logical; save diagnostic plots of output?
<code>path.diagnostics</code>	- path to where the diagnostic graphs should be saved
<code>parallel</code>	- (experimental) logical stating whether to run <code>temporal_downscale_functions.R</code> in parallel *Not Implemented yet
<code>n.cores</code>	- (experimental) how many cores to use in parallelization *Not implemented yet
<code>overwrite</code>	- overwrite existing files? Currently ignored
<code>verbose</code>	logical: should <code>ncdf4</code> functions print debugging information as they run?

Details

Debias Meteorology using Multiple Linear Regression Statistically debias met datasets and generate ensembles based on the observed uncertainty

Author(s)

Christy Rollinson

See Also

Other debias - Debias & Align Meteorology Datasets into continuous time series: `align.met()`

<code>debias_met</code>	<i>debias_met</i>
-------------------------	-------------------

Description

`debias.met` takes `input_met` and debiases it based on statistics from a `train_met` dataset

Usage

```
debias.met(
  outfolder,
  input_met,
  train_met,
  site_id,
  de_method = "linear",
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	location where output is stored
<code>input_met</code>	- the <code>source_met</code> dataset that will be altered by the training dataset in NC format.
<code>train_met</code>	- the observed dataset that will be used to train the modeled dataset in NC format
<code>site_id</code>	BETY site id
<code>de_method</code>	- select which debias method you would like to use, options are 'normal', 'linear regression'
<code>overwrite</code>	logical: replace output file if it already exists? Currently ignored.
<code>verbose</code>	logical: should ncdf4
<code>...</code>	other inputs functions print debugging information as they run?

Author(s)

James Simkins

download.Ameriflux	<i>download.Ameriflux</i>
--------------------	---------------------------

Description

Download Ameriflux L2 netCDF files

Usage

```
download.Ameriflux(  
    sitename,  
    outfolder,  
    start_date,  
    end_date,  
    overwrite = FALSE,  
    verbose = FALSE,  
    ...  
)
```

Arguments

sitename	the FLUXNET ID of the site to be downloaded, used as file name prefix. The 'SITE_ID' field in list of Ameriflux sites
outfolder	location on disk where outputs will be stored
start_date	the start date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
end_date	the end date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
overwrite	should existing files be overwritten
verbose	should the function be very verbose
...	further arguments, currently ignored

Author(s)

Josh Mantooth, Rob Kooper, Ankur Desai

download.AmerifluxLBL

Download Ameriflux LBL CSV files

Description

download.AmerifluxLBL. Function uses amf_download_base function from amerifluxr package to download a zip-file of data. The zip-file is extracted to a csv-file that is stored to the given outfolder. Details about amf_download_base function can be found here: https://github.com/chuhousen/amerifluxr/blob/master/R/amf_download_base.R

Usage

```
download.AmerifluxLBL(
  sitename,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  username = "pecan",
  method,
  useremail = "@",
  data_product = "BASE-BADM",
  data_policy = "CCBY4.0",
  ...
)
```

Arguments

sitename	the Ameriflux ID of the site to be downloaded, used as file name prefix. The 'SITE_ID' field in list of Ameriflux sites
outfolder	location on disk where outputs will be stored
start_date	the start date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
end_date	the end date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
overwrite	should existing files be overwritten
verbose	should the function be very verbose
username	Ameriflux username
method	Optional. download_file() function option. Use this to set custom programs such as ncftp
useremail	Used email, should include 'address sign' for code to be functional
data_product	AmeriFlux data product
data_policy	Two possible licenses (based on the site): 'CCBY4.0' or 'LEGACY'
...	further arguments, currently ignored

Details

Uses Ameriflux LBL JSON API to download met data from Ameriflux towers in CSV format

Author(s)

Ankur Desai, Henri Kajasilta based on download.Ameriflux.R by Josh Mantooth, Rob Kooper, Shawn Serbin

Examples

```
## Not run:
options(download.ftp.method = "ncftpget")
result <- download.AmerifluxLBL("US-Akn", "~/", "2011-01-01", "2011-12-31", overwrite=TRUE)

## End(Not run)
```

download.CRUNCEP	<i>Download CRUNCEP data</i>
------------------	------------------------------

Description

Download and convert to CF CRUNCEP single grid point from MSTIMIP server using OPENDAP interface

Usage

```
download.CRUNCEP(
  outfolder,
  start_date,
  end_date,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  maxErrors = 10,
  sleep = 2,
  method = "ncss",
  ...
)
```

Arguments

outfolder Directory where results should be written

start_date, end_date Range of years to retrieve. Format is YYYY-MM-DD, but only the year portion is used and the resulting files always contain a full year of data.

<code>lat.in</code>	site latitude in decimal degrees
<code>lon.in</code>	site longitude in decimal degrees
<code>overwrite</code>	logical. Download a fresh version even if a local file with the same name already exists?
<code>verbose</code>	logical. Passed on to <code>ncvar_def</code> and <code>nc_create</code> to control printing of debug info
<code>maxErrors</code>	Maximum times to re-try following an error accessing netCDF data through THREDDS
<code>sleep</code>	Wait time between attempts following a THREDDS or other error
<code>method</code>	(string) Data access method. ‘ <code>opendap</code> ’ (default) attempts to directly access files via OpenDAP. ‘ <code>ncss</code> ’ (NetCDF subset) subsets the file on the server, downloads the subsetted file to ‘ <code>tempfile</code> ’ and then reads it locally. ‘ <code>opendap</code> ’ is faster when it works, but often fails because of server issues. ‘ <code>ncss</code> ’ can be much slower, but is more reliable.
<code>...</code>	Other arguments, currently ignored

Author(s)

James Simkins, Mike Dietze, Alexey Shiklomanov

<code>download.ERA5_cds</code>	<i>Download ERA5 Climate Data from the Copernicus CDS API</i>
--------------------------------	---

Description

Download ERA5 climate data from the Copernicus Climate Data Store (CDS) API as NetCDF files, year by year, according to user-specified parameters. The function saves one NetCDF file per year in the specified output directory.

Usage

```
download.ERA5_cds(
  outfolder,
  start_date,
  end_date,
  extent,
  variables,
  user,
  key,
  time = NULL,
  dataset = "reanalysis-era5-single-levels",
  product_type = "ensemble_members",
  timeout = 36000
)
```

Arguments

outfolder	Character. Directory where downloaded NetCDF files will be saved.
start_date	character: the start date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
end_date	character: the end date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
extent	numeric: a vector of numbers contains the bounding box (formatted as xmin, xmax, ymin, ymax) (longitude and latitude in degrees).
variables	character: a vector contains variables to be downloaded (e.g., c("2m_temperature", "surface_pres")).
user	Character. CDS user ID (UID) from your CDS profile. Required for authentication.
key	Character. CDS API key from your CDS profile. Required for authentication.
time	Character vector or NULL. Hours of the day to download (e.g., c("00:00", "12:00")). Default to NULL to download all hours.
dataset	Character. Name of the CDS dataset to use (default: "reanalysis-era5-single-levels").
product_type	Character. Product type to request from CDS (default: "ensemble_members").
timeout	numeric: the maximum time (in seconds) allowed to download the data. The default is 36000 seconds.

Details

This function requires a valid CDS API key and the `ecmwf` package for accessing the Copernicus Climate Data Store. To get a Copernicus CDS API key, register at <https://cds.climate.copernicus.eu/profile>. You must provide both `user` (UID) and `key` parameters from your CDS profile.

Credentials are stored by `ecmwf::wf_set_key`, which uses the `keyring` package. On a headless machine (cluster node, server, container) there is no OS keyring available and the default backend will prompt for a keyring password. To avoid this, set `Sys.setenv(R_KEYRING_BACKEND = "env")` before calling this function, which keeps the key in an environment variable for the session instead.

You can check the "CC-BY" license under the 'licences' tab of your profile page.

Value

A list where each element is a list containing:

file	File path to the downloaded NetCDF file.
host	Host name where the file was downloaded.
startdate	Start date and time of the data in the file.
enddate	End date and time of the data in the file.
mimetype	MIME type of the file ("application/x-netcdf").
formatname	Format name ("ERA5_year.nc").

Author(s)

Dongchen Zhang, Akash

Examples

```
## Not run:
# Download ERA5 reanalysis data for 2020
output_dir <- withr::local_tempdir()
era5_files <- download.ERA5_cds(
  outfolder = output_dir,
  start_date = "2020-01-01",
  end_date = "2020-06-30",
  extent = c(-72.2215, -72.1215, 42.4878, 42.5878),
  variables = c("2m_temperature", "surface_pressure"),
  user = "your_cds_user_id",
  key = "your_cds_api_key",
  product_type = "reanalysis"
)

# Download ensemble data for specified hours only
era5_files <- download.ERA5_cds(
  outfolder = output_dir,
  start_date = "2020-01-01",
  end_date = "2020-12-31",
  extent = c(-83.05, -82.95, 42.95, 43.05),
  variables = "surface_solar_radiation_downwards",
  user = "your_cds_user_id",
  key = "your_cds_api_key",
  time = c("00:00", "12:00")
)

## End(Not run)
```

download.FACE

Download Raw FACE data from the internet

Description

Download Raw FACE data from the internet

Usage

```
download.FACE(
  sitename,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  method,
```

```
    ...
)
```

Arguments

sitename	sitename
outfolder	location where output is stored
start_date	desired start date YYYY-MM-DD
end_date	desired end date YYYY-MM-DD
overwrite	overwrite existing files? Default is FALSE
method	Optional. Passed to download_file() function. Use this to set custom programs such as ncftp to use when downloading files from FTP sites
...	other inputs

Author(s)

Betsy Cowdery

download.Fluxnet2015 *download.Fluxnet2015*

Description

Download Fluxnet 2015 CSV files

Usage

```
download.Fluxnet2015(
  sitename,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  username = "pecan",
  ...
)
```

Arguments

sitename	the FLUXNET ID of the site to be downloaded, used as file name prefix. The 'SITE_ID' field in list of Ameriflux sites
outfolder	location on disk where outputs will be stored
start_date	the start date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)

<code>end_date</code>	the end date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	should the function be very verbose
<code>username</code>	login name for Ameriflux
<code>...</code>	further arguments, currently ignored

Author(s)

Ankur Desai, based on download.Ameriflux.R by Josh Mantooth, Rob Kooper

```
download.FluxnetLaThuile
      download.FluxnetLaThuile
```

Description

Download Fluxnet LaThuile CSV files

Usage

```
download.FluxnetLaThuile(
  sitename,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  username = "pecan",
  ...
)
```

Arguments

<code>sitename</code>	the FLUXNET ID of the site to be downloaded, used as file name prefix. The 'SITE_ID' field in list of Fluxnet LaThuile sites
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year part of the date)
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	should the function be very verbose
<code>username</code>	should be the registered Fluxnet username, else defaults to pecan
<code>...</code>	further arguments, currently ignored

Author(s)

Ankur Desai

 download.Geostreams *Download Geostreams data from Clowder API*

Description

Download Geostreams data from Clowder API

Usage

```
download.Geostreams(
    outfolder,
    sitename,
    start_date,
    end_date,
    url = "https://terraref.ncsa.illinois.edu/clowder/api/geostreams",
    key = NULL,
    user = NULL,
    pass = NULL,
    ...
)
```

Arguments

<code>outfolder</code>	directory in which to save json result. Will be created if necessary
<code>sitename</code>	character. Must match a Geostreams sensor__name
<code>start_date, end_date</code>	datetime
<code>url</code>	base url for Clowder host
<code>key, user, pass</code>	authentication info for Clowder host.
<code>...</code>	other arguments passed as query parameters

Details

Depending on the setup of your Clowder host, authentication may be by username/password, by API key, or skipped entirely. `download.Geostreams` looks first in its call arguments for an API key, then a username and password, then if these are NULL it looks in the user's home directory for a file named `~/pecan.clowder.xml`, and finally if no keys or passwords are found there it attempts to connect unauthenticated.

If using `~/pecan.clowder.xml`, it must be a valid PEcAn-formatted XML settings file and must contain a `<clowder>` key that specifies hostname, user, and password for your Clowder server:

```
<?xml version="1.0"?>
<pecan>
  <clowder>
    <hostname>terraref.ncsa.illinois.edu</hostname>
    <user>yourname</user>
    <password>superSecretPassw0rd</password>
  </clowder>
</pecan>
```

Author(s)

Harsh Agrawal, Chris Black

Examples

```
## Not run:
download.Geostreams(outfolder = "~/output/dbfiles/Clowder_EF",
                    sitename = "UIUC Energy Farm - CEN",
                    start_date = "2016-01-01", end_date="2016-12-31",
                    key="verysecret")

## End(Not run)
```

download.GFDL	<i>Download GFDL CMIP5 outputs for a single grid point using OPeNDAP and convert to CF</i>
---------------	--

Description

Download GFDL CMIP5 outputs for a single grid point using OPeNDAP and convert to CF

Usage

```
download.GFDL(
  outfolder,
  start_date,
  end_date,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  model = "CM3",
  scenario = "rcp45",
  ensemble_member = "r1i1p1",
  ...
)
```

Arguments

<code>outfolder</code>	Directory for storing output
<code>start_date</code>	Start date for met (will be converted via <code>[base::as.POSIXlt]</code>)
<code>end_date</code>	End date for met (will be converted via <code>[base::as.POSIXlt]</code>)
<code>lat.in</code>	Latitude coordinate for met
<code>lon.in</code>	Longitude coordinate for met
<code>overwrite</code>	Logical: Download a fresh version even if a local file with the same name already exists?
<code>verbose</code>	Logical, passed on to <code>ncvar_def</code> and <code>nc_create</code> to control printing of debug info
<code>model</code>	Which GFDL model to run (options are CM3, ESM2M, ESM2G)
<code>scenario</code>	Which scenario to run (options are rcp26, rcp45, rcp60, rcp85)
<code>ensemble_member</code>	Which ensemble_member to initialize the run (options are r1i1p1, r3i1p1, r5i1p1)
<code>...</code>	further arguments, currently ignored

Author(s)

James Simkins, Alexey Shiklomanov, Ankur Desai

<code>download.GLDAS</code>	<i>Download GLDAS data</i>
-----------------------------	----------------------------

Description

Download and convert single grid point GLDAS to CF single grid point from hydro1.sci.gsfc.nasa.gov using OPENDAP interface

Usage

```
download.GLDAS(
  outfolder,
  start_date,
  end_date,
  site_id,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	location where output is stored
<code>start_date</code>	desired start date
<code>end_date</code>	desired end date
<code>site_id</code>	desired site id
<code>lat.in</code>	latitude of site
<code>lon.in</code>	longitude of site
<code>overwrite</code>	overwrite existing files? Default is FALSE
<code>verbose</code>	Default is FALSE, used as input for <code>ncdf4::ncvar_def</code>
<code>...</code>	other inputs

Author(s)

Christy Rollinson

<code>download.ICOS</code>	<i>Download ICOS Ecosystem data products</i>
----------------------------	--

Description

Currently available products: Drought-2018 ecosystem eddy covariance flux product <https://www.icos-cp.eu/data-products/YVR0-4898> ICOS Final Fully Quality Controlled Observational Data (Level 2) <https://www.icos-cp.eu/data-products/ecosystem-release>

Usage

```
download.ICOS(
  sitename,
  outfolder,
  start_date,
  end_date,
  product,
  overwrite = FALSE,
  ...
)
```

Arguments

<code>sitename</code>	ICOS id of the site. Example - "BE-Bra"
<code>outfolder</code>	path to the directory where the output file is stored. If specified directory does not exist, it is created.
<code>start_date</code>	start date of the data request in the form YYYY-MM-DD
<code>end_date</code>	end date area of the data request in the form YYYY-MM-DD

product	ICOS product to be downloaded. Currently supported options: "Drought2018", "ETC"
overwrite	should existing files be overwritten. Default False.
...	used when extra arguments are present.

Value

information about the output file

Author(s)

Ayush Prasad

Examples

```
## Not run:
download.ICOS("FI-Sii", "/home/carya/pecan", "2016-01-01", "2018-01-01", product="Drought2018")

## End(Not run)
```

download.MACA	<i>download.MACA</i>
---------------	----------------------

Description

Download MACA CMIP5 outputs for a single grid point using OPeNDAP and convert to CF

Usage

```
download.MACA(
  outfolder,
  start_date,
  end_date,
  site_id,
  lat.in,
  lon.in,
  model = "IPSL-CM5A-LR",
  scenario = "rcp85",
  ensemble_member = "r1i1p1",
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	location where output is stored
<code>start_date</code>	, of the format "YEAR-01-01 00:00:00"
<code>end_date</code>	, of the format "YEAR-12-31 23:59:59"
<code>site_id</code>	BETY site id
<code>lat.in</code>	latitude of site
<code>lon.in</code>	longitude of site
<code>model</code>	, select which MACA model to run (options are BNU-ESM, CNRM-CM5, CSIRO-Mk3-6-0, bcc-csm1-1, bcc-csm1-1-m, CanESM2, GFDL-ESM2M, GFDL-ESM2G, HadGEM2-CC365, HadGEM2-ES365, inmcm4, MIROC5, MIROC-ESM, MIROC-ESM-CHEM, MRI-CGCM3, CCSM4, IPSL-CM5A-LR, IPSL-CM5A-MR, IPSL-CM5B-LR, NorESM1-M)
<code>scenario</code>	, select which scenario to run (options are rcp45, rcp85)
<code>ensemble_member</code>	, r1i1p1 is the only ensemble member available for this dataset, CCSM4 uses r6i1p1 instead
<code>overwrite</code>	overwrite existing files? Default is FALSE
<code>verbose</code>	Default is FALSE, used as input in <code>ncdf4::ncvar_def</code>
<code>...</code>	other inputs

Author(s)

James Simkins

download.MERRA	<i>Download MERRA data</i>
----------------	----------------------------

Description

Download MERRA data

Usage

```
download.MERRA(
  outfolder,
  start_date,
  end_date,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	Directory where results should be written
<code>start_date, end_date</code>	Range of years to retrieve. Format is YYYY-MM-DD, but only the year portion is used and the resulting files always contain a full year of data.
<code>lat.in</code>	site latitude in decimal degrees
<code>lon.in</code>	site longitude in decimal degrees
<code>overwrite</code>	logical. Download a fresh version even if a local file with the same name already exists?
<code>verbose</code>	logical. Passed on to <code>ncvar_def</code> and <code>nc_create</code> to control printing of debug info
<code>...</code>	Not used – silently soak up extra arguments from ‘convert_input’, etc.

Value

‘data.frame’ of meteorology data metadata

Author(s)

Alexey Shiklomanov

`download.MsTMIP_NARR` *download.MsTMIP_NARR*

Description

Download and conver to CF NARR single grid point from MSTIMIP server using OPEN-DAP interface

Usage

```
download.MsTMIP_NARR(
  outfolder,
  start_date,
  end_date,
  site_id,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	location where output is stored
<code>start_date</code>	YYYY-MM-DD
<code>end_date</code>	YYYY-MM-DD
<code>site_id</code>	BETY site id
<code>lat.in</code>	latitude of site
<code>lon.in</code>	longitude of site
<code>overwrite</code>	overwrite existing files? Default is FALSE
<code>verbose</code>	Default is FALSE, used in <code>ncdf4::ncvar_def</code>
<code>...</code>	Other inputs

Author(s)

James Simkins

<code>download.NARR</code>	<i>Download NARR files</i>
----------------------------	----------------------------

Description

Download NARR files

Usage

```
download.NARR(
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  method,
  ...
)
```

Arguments

<code>outfolder</code>	location where output is stored
<code>start_date</code>	desired start date YYYY-MM-DD
<code>end_date</code>	desired end date YYYY-MM-DD
<code>overwrite</code>	Overwrite existing files? Default=FALSE
<code>verbose</code>	Turn on verbose output? Default=FALSE
<code>method</code>	Method of file retrieval. Can set this using the options(<code>download.ftp.method=[method]</code>) in your Rprofile.
<code>...</code>	other inputs

Author(s)

Betsy Cowdery, Shawn Serbin

Examples

```
## Not run:
options(download.ftp.method = "ncftpget")
download.NARR(
  outfolder = "~/",
  start_date = "2000/01/01",
  end_date = "2000/01/02",
  overwrite = TRUE,
  verbose = TRUE)

## End(Not run)
```

download.NARR_site	<i>Download NARR time series for a single site</i>
--------------------	--

Description

Download NARR time series for a single site

Usage

```
download.NARR_site(
  outfolder,
  start_date,
  end_date,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  progress = TRUE,
  parallel = TRUE,
  ncores = if (parallel) parallel::detectCores() else NULL,
  ...
)
```

Arguments

outfolder	Target directory for storing output
start_date	Start date for met data
end_date	End date for met data
lat.in	Site latitude coordinate
lon.in	Site longitude coordinate

<code>overwrite</code>	Overwrite existing files? Default=FALSE
<code>verbose</code>	Turn on verbose output? Default=FALSE
<code>progress</code>	Whether or not to show a progress bar. Requires the ‘progress’ package to be installed.
<code>parallel</code>	Download in parallel? Default = TRUE
<code>ncores</code>	Number of cores for parallel download. Default is ‘parallel::detectCores()’
<code>...</code>	further arguments, currently ignored

Author(s)

Alexey Shiklomanov

Examples

```
## Not run:
download.NARR_site(tempdir(), "2001-01-01", "2001-01-12", 43.372, -89.907)

## End(Not run)
```

download.NEONmet	<i>Download NEON Site Met CSV files</i>
------------------	---

Description

download.NEONmet

Usage

```
download.NEONmet(
  sitename,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>sitename</code>	the NEON ID of the site to be downloaded, used as file name prefix. The 4-letter SITE code in list of NEON sites
<code>outfolder</code>	location on disk where outputs will be stored

<code>start_date</code>	the start date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year and month of the date)
<code>end_date</code>	the end date of the data to be downloaded. Format is YYYY-MM-DD (will only use the year and month part of the date)
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	makes the function output more text
<code>...</code>	further arguments, currently ignored

Details

Uses NEON v0 API to download met data from NEON towers and convert to CF NetCDF

Examples

```
## Not run:
result <- download.NEONmet(
  sitename = 'HARV',
  outfolder = '~/ ',
  start_date = '2017-01-01',
  end_date = '2017-01-31',
  overwrite = TRUE)

## End(Not run)
```

download.NLDAS	<i>Download NLDAS met data</i>
----------------	--------------------------------

Description

Download and convert single grid point NLDAS to CF single grid point from hydro1.sci.gsfc.nasa.gov using OPENDAP interface

Usage

```
download.NLDAS(
  outfolder,
  start_date,
  end_date,
  site_id,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	location of output
<code>start_date</code>	desired start date YYYY-MM-DD
<code>end_date</code>	desired end date YYYY-MM-DD
<code>site_id</code>	site id (BETY)
<code>lat.in</code>	latitude of site
<code>lon.in</code>	longitude of site
<code>overwrite</code>	overwrite existing files? Default is FALSE
<code>verbose</code>	Turn on verbose output? Default=FALSE
<code>...</code>	Other inputs

Author(s)

Christy Rollinson (with help from Ankur Desai)

`download.NOAA_GEFS` *Download NOAA GEFS Weather Data*

Description

Download NOAA GEFS Weather Data

Usage

```
download.NOAA_GEFS(
  site_id,
  lat.in,
  lon.in,
  outfolder,
  start_date = Sys.Date(),
  end_date = start_date + lubridate::days(16),
  downscale = TRUE,
  overwrite = FALSE,
  ...
)
```

Arguments

<code>site_id</code>	The unique ID given to each site. This is used as part of the file name.
<code>lat.in</code>	site latitude in decimal degrees
<code>lon.in</code>	site longitude in decimal degrees
<code>outfolder</code>	Directory where results should be written
<code>start_date</code>	Range of dates/times to be downloaded (default assumed to be time that function is run)

end_date	end date for range of dates to be downloaded (default 16 days from start_date)
downscale	logical, assumed True. Indicates whether data should be downscaled to hourly
overwrite	logical. Download a fresh version even if a local file with the same name already exists?
...	Additional optional parameters, currently ignored

Value

A list of data frames is returned containing information about the data file that can be used to locate it later. Each data frame contains information about one file.

Information on Units

Information on NOAA weather units can be found below. Note that the temperature is measured in degrees C, but is converted at the station and downloaded in Kelvin.

NOAA_GEFS General Information

This function downloads NOAA GEFS weather data. GEFS is an ensemble of 31 different weather forecast models. A 16 day forecast is available every 6 hours and a 35 day forecast is available every 24 hours. Both are at 3-hour frequency for the first 10 days of the forecast and 6-hour frequency beyond that. Each forecast includes information on a total of 8 variables. These are transformed from the NOAA standard to the internal PEcAn standard.

Data Availability

NOAA GEFS weather data is available on a rolling 4 day basis. Dates provided in "start_date" must be within this range. The end date can be any point after that, but if the end date is beyond 16 days (35 days for the midnight UTC forecast), only 16 (35) days worth of forecast are retrieved. Times are rounded down to the previous 6 hour forecast.

NOAA GEFS weather data isn't always posted immediately. Each 16-day forecast takes approximately three hours to run, and the once-a-day forecasts for days 17-35 are posted much later (up to 21 hours) than the forecasts for days 0 to 16. See the [GEFS v12 release announcement](https://www.weather.gov/media/notification/pdf2/scn20-75gefs_v12_changes.pdf) for details.

Data Save Format

Data is saved in the netcdf format to the specified directory. File names reflect the precision of the data to the given range of days. NOAA.GEFS.willow.creek.3.2018-06-08T06:00.2018-06-24T06:00.nc specifies the forecast, using ensemble number 3 at willow creek on June 6th, 2018 at 6:00 a.m. to June 24th, 2018 at 6:00 a.m.

Author(s)

Quinn Thomas, modified by K Zarada

References

<https://www.ncdc.noaa.gov/crn/measurements.html>

Examples

```
## Not run:
download.NOAA_GEFS(
  site_id = 676,
  lat.in= 45.805925,
  lon.in = -90.07961,
  outfolder="~/Working/results")

## End(Not run)
```

<code>download.PaleON</code>	<i>download.PaleON</i>
------------------------------	------------------------

Description

Download PaleON files

Usage

```
download.PaleON(
  sitename,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  ...
)
```

Arguments

<code>sitename</code>	sitename
<code>outfolder</code>	desired output location
<code>start_date</code>	desired start date YYYY-MM-DD
<code>end_date</code>	desired end date YYYY-MM-DD
<code>overwrite</code>	overwrite existing files? Default is FALSE
<code>...</code>	Other inputs

Author(s)

Betsy Cowdery

`download.PalEON_ENS` *Download PalEON met ensemble files*

Description

Download PalEON met ensemble files

Usage

```
download.PalEON_ENS(
    sitename,
    outfolder,
    start_date,
    end_date,
    overwrite = FALSE,
    ...
)
```

Arguments

<code>sitename</code>	sitename
<code>outfolder</code>	desired output folder
<code>start_date</code>	desired start date YYYY-MM-DD
<code>end_date</code>	desired end date YYYY-MM-DD
<code>overwrite</code>	overwrite existing files? Default is FALSE
<code>...</code>	Other inputs

Author(s)

Betsy Cowdery, Mike Dietze

`download.raw.met.module`
download.raw.met.module

Description

download.raw.met.module

Usage

```
.download.raw.met.module(
    dir,
    met,
    register,
    machine,
    start_date,
    end_date,
    str_ns,
    con,
    input_met,
    site.id,
    lat.in,
    lon.in,
    host,
    site,
    username,
    overwrite = FALSE,
    dbparms,
    Ens.Flag = FALSE
)
```

Arguments

<code>dir</code>	directory to write outputs to
<code>met</code>	source included in <code>input_met</code>
<code>register</code>	register.xml, provided by <code>met.process</code>
<code>machine</code>	machine associated with hostname, provided by <code>met.process</code>
<code>start_date</code>	the start date of the data to be downloaded (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded (will only use the year part of the date)
<code>str_ns</code>	substitute for <code>site_id</code> if not provided, provided by <code>met.process</code>
<code>con</code>	database connection based on <code>dbparms</code> in <code>met.process</code>
<code>input_met</code>	Which data source to process
<code>site.id</code>	site id
<code>lat.in</code>	site latitude, provided by <code>met.process</code>
<code>lon.in</code>	site longitude, provided by <code>met.process</code>
<code>host</code>	host info from settings file
<code>site</code>	site info from settings file
<code>username</code>	database username
<code>overwrite</code>	whether to force <code>download.raw.met.module</code> to proceed
<code>dbparms</code>	database settings from settings file
<code>Ens.Flag</code>	default set to FALSE

Value

A list of data frames is returned containing information about the data file that can be used to locate it later. Each data frame contains information about one file.

download.US_WCr	<i>download.US-WCr</i>
-----------------	------------------------

Description

download.US-WCr

Usage

download.US_WCr(start_date, end_date, timestep = 1)

Arguments

- start_date Start date/time data should be downloaded for
- end_date End date/time data should be downloaded for
- timestep How often to take data points from the file. Must be a multiple of 0.5

General Description

Obtains data from Ankur Desai’s Willow Creek flux tower, and selects certain variables (NEE and LE) to return Data is returned at the given timestep in the given range.
This data includes information on a number of flux variables.
The timestep parameter is measured in hours, but is then converted to half hours because the data’s timestep is every half hour.

Author(s)

Luke Dramko

download.US_Wlef	<i>download.US_Wlef</i>
------------------	-------------------------

Description

download.US_Wlef

Usage

download.US_Wlef(start_date, end_date, timestep = 1)

Arguments

<code>start_date</code>	Start date/time data should be downloaded for
<code>end_date</code>	End date/time data should be downloaded for
<code>timestep</code>	How often to take data points from the file. Must be integer

General Description

Obtains data from Ankur Desai's WLEF/ Parks Fall flux tower, and selects certain variables (NEE and LE) to return Data is returned at the given timestep in the given range.

This data includes information on a number of flux variables.

Author(s)

Luke Dramko and K Zarada

<code>download_grid</code>	<i>Download all requested timepoints of one GEFS ensemble member</i>
----------------------------	--

Description

Download all requested timepoints of one GEFS ensemble member

Usage

```
download_grid(
  ens_index,
  location,
  directory,
  hours_char,
  cycle,
  base_filename1,
  vars,
  working_directory
)
```

Arguments

<code>ens_index</code>	ensemble member as integer (1-31)
<code>location</code>	bounding box portion of query, as a single URL-escaped string
<code>directory</code>	server path portion of query, as a single URL-escaped string (not to be confused with local output dir – that's 'working_directory')
<code>hours_char</code>	timepoints to retrieve, as zero-padded strings e.g. 'c("000", "384", "840")'
<code>cycle</code>	forecast hour to use ("00", "06", "12", or "18")

base_filename1 URL onto which to append query string components

vars variable listing component of query, as a single URL-escaped string

working_directory path on local disk to write output

download_NOAA_GIFS_EFI

download_NOAA_GIFS_EFI

Description

download_NOAA_GIFS_EFI

Usage

download_NOAA_GIFS_EFI(siteName, outFolder, start_date, site.lat, site.lon)

Arguments

siteName NEON site name

outFolder filepath to save ensemble member .nc files

start_date start date for met forecast

site.lat site lat

site.lon site lon

Value

message confirming download complete and location of .nc files

Author(s)

Alexis Helgeson

```
downscale_one_cfmet_day
```

Internal helper to downscale a single row from a daily file

Description

Internal helper to downscale a single row from a daily file

Usage

```
downscale_one_cfmet_day(df, tseq, lat)
```

Arguments

<code>df</code>	one row from dailymet
<code>tseq</code>	vector of hours at which to estimate
<code>lat</code>	latitude

Value

df with one row for each hour in ‘tseq’

```
downscale_repeat_6hr_to_half_hrly
```

Downscale repeat to half hourly

Description

Downscale repeat to half hourly

Usage

```
downscale_repeat_6hr_to_half_hrly(df, varName, hr = 0.5)
```

Arguments

<code>df</code>	dataframe of data to be downsampled (Longwave)
<code>varName</code>	variable names to be downsampled
<code>hr</code>	hour to downscale to- default is 0.5

Value

A dataframe of downsampled data

Author(s)

Laura Puckett

downscale_repeat_6hr_to_hrly
Downscale repeat to hourly

Description

Downscale repeat to hourly

Usage

```
downscale_repeat_6hr_to_hrly(df, varName, hr = 1)
```

Arguments

df	dataframe of data to be downscaled (Longwave)
varName	variable names to be downscaled
hr	hour to downscale to- default is 1

Value

A dataframe of downscaled data

Author(s)

Laura Puckett

downscale_ShortWave_to_half_hrly
Downscale shortwave to half hourly

Description

Downscale shortwave to half hourly

Usage

```
downscale_ShortWave_to_half_hrly(df, lat, lon, hr = 0.5)
```

Arguments

df	data frame of variables
lat	lat of site
lon	long of site
hr	hour to downscale to- default is 1

Value

A dataframe of downscaled state variables
ShortWave.ds

Author(s)

Laura Puckett

<code>downscale_ShortWave_to_hrly</code>
<i>Downscale shortwave to hourly</i>

Description

Downscale shortwave to hourly

Usage

```
downscale_ShortWave_to_hrly(df, lat, lon, hr = 1)
```

Arguments

- | | |
|------------------|------------------------------------|
| <code>df</code> | data frame of variables |
| <code>lat</code> | lat of site |
| <code>lon</code> | long of site |
| <code>hr</code> | hour to downscale to- default is 1 |

Value

A dataframe of downscaled state variables
ShortWave.ds

Author(s)

Laura Puckett

`downscale_solar_geom` *Calculate potential shortwave radiation*

Description

Calculate potential shortwave radiation

Usage

```
downscale_solar_geom(doy, lon, lat)
```

Arguments

<code>doy</code>	day of year in decimal
<code>lon</code>	longitude
<code>lat</code>	latitude

Value

vector of potential shortwave radiation for each doy

Author(s)

Quinn Thomas

`downscale_solar_geom_halfhour`
Calculate potential shortwave radiation

Description

Calculate potential shortwave radiation

Usage

```
downscale_solar_geom_halfhour(doy, lon, lat)
```

Arguments

<code>doy</code>	day of year in decimal
<code>lon</code>	longitude
<code>lat</code>	latitude

Value

vector of potential shortwave radiation for each doy

Author(s)

Quinn Thomas

`downscale_spline_to_half_hrly`

Downscale spline to half hourly

Description

Downscale spline to half hourly

Usage

```
downscale_spline_to_half_hrly(df, VarNames, hr = 0.5)
```

Arguments

<code>df</code>	dataframe of data to be downscales
<code>VarNames</code>	variable names to be downscaled
<code>hr</code>	hour to downscale to- default is 0.5

Value

A dataframe of half hourly downscaled state variables

Author(s)

Laura Puckett

`downscale_spline_to_hrly`

Downscale spline to hourly

Description

Downscale spline to hourly

Usage

```
downscale_spline_to_hrly(df, VarNames, hr = 1)
```

Arguments

<code>df</code>	dataframe of data to be downscales
<code>VarNames</code>	variable names to be downscaled
<code>hr</code>	hour to downscale to- default is 1

Value

A dataframe of downscaled state variables

Author(s)

Laura Puckett

equation_of_time	<i>Equation of time: Eccentricity and obliquity</i>
------------------	---

Description

For description of calculations, see https://en.wikipedia.org/wiki/Equation_of_time#Calculating_the_equation_of_time

Usage

```
equation_of_time(doy)
```

Arguments

doy	Day of year
-----	-------------

Value

‘numeric(1)’ length of the solar day, in hours.

Author(s)

Alexey Shiklomanov

ERA5_met_process	<i>Met Processes for ERA5 data</i>
------------------	------------------------------------

Description

Met Processes for ERA5 data

Usage

```
ERA5_met_process(
  settings,
  in.path,
  out.path,
  write.db = FALSE,
  write = TRUE,
  ncores = 1
)
```

Arguments

<code>settings</code>	a multi-settings object
<code>in.path</code>	met input path
<code>out.path</code>	output path
<code>write.db</code>	if write into Bety database
<code>write</code>	if write the settings into pecan.xml file in the outdir of settings.
<code>ncores</code>	numeric: the number of CPUs for the parallel compute. Default is 1.

Value

if `write.db` is `True` then return input IDs with physical paths; if `write.db` is `False` then return just physical paths of extracted ERA5 clim files.

Author(s)

Dongchen Zhang

`exner`

Exner function

Description

estimated exner function

Usage

`exner(pres)`

Arguments

<code>pres</code>	air pressure (Bar)
-------------------	--------------------

Author(s)

Mike Dietze

<code>extract.local.CMIP5</code>	<i>Extract NLDAS from local download Extract NLDAS meteorology for a point from a local download of the full grid</i>
----------------------------------	---

Description

This function extracts CMIP5 data from grids that have been downloaded and stored locally. Files are saved as a netCDF file in CF conventions at *DAILY* resolution. Note: At this point in time, variables that are only available at a native monthly resolution will be repeated to give a pseudo-daily record (and can get dealt with in the downscaling workflow). These files are ready to be used in the general PEcAn workflow or fed into the downscaling workflow.

Usage

```
extract.local.CMIP5(
  outfolder,
  in.path,
  start_date,
  end_date,
  lat.in,
  lon.in,
  model,
  scenario,
  ensemble_member = "r1i1p1",
  date.origin = NULL,
  adjust.pr = 1,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	- directory where output files will be stored
<code>in.path</code>	- path to the raw full grids
<code>start_date</code>	- first day for which you want to extract met (yyyy-mm-dd)
<code>end_date</code>	- last day for which you want to extract met (yyyy-mm-dd)
<code>lat.in</code>	site latitude in decimal degrees
<code>lon.in</code>	site longitude in decimal degrees
<code>model</code>	which GCM to extract data from
<code>scenario</code>	which experiment to pull (p1000, historical, ...)
<code>ensemble_member</code>	which CMIP5 experiment ensemble member

<code>date.origin</code>	(optional) specify the date of origin for timestamps in the files being read. If NULL defaults to 1850 for historical simulations (except MPI-ESM-P) and 850 for p1000 simulations (plus MPI-ESM-P historical). Format: YYYY-MM-DD
<code>adjust.pr</code>	- adjustment factor fore precipitation when the extracted values seem off
<code>overwrite</code>	logical. Download a fresh version even if a local file with the same name already exists?
<code>verbose</code>	logical. to control printing of debug info
<code>...</code>	Other arguments, currently ignored

Author(s)

Christy Rollinson

`extract.local.NLDAS` *Extract NLDAS from local download Extract NLDAS meteorology for a point from a local download of the full grid*

Description

This function extracts NLDAS data from grids that have been downloaded and stored locally. Once upon a time, you could query these files directly from the internet, but now they're behind a tricky authentication wall. Files are saved as a netCDF file in CF conventions. These files are ready to be used in the general PEcAn workflow or fed into the downscaling workflow.

Usage

```
extract.local.NLDAS(
  outfolder,
  in.path,
  start_date,
  end_date,
  lat.in,
  lon.in,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	- directory where output files will be stored
<code>in.path</code>	- path to the raw full grids
<code>start_date</code>	- first day for which you want to extract met (yyyy-mm-dd)

<code>end_date</code>	- last day for which you want to extract met (yyyy-mm-dd)
<code>lat.in</code>	site latitude in decimal degrees
<code>lon.in</code>	site longitude in decimal degrees
<code>overwrite</code>	logical. Download a fresh version even if a local file with the same name already exists?
<code>verbose</code>	logical. Passed on to <code>ncvar_def</code> and <code>nc_create</code> to control printing of debug info
<code>...</code>	Other arguments, currently ignored

Author(s)

Christy Rollinson

<code>extract.nc</code>	<i>extract.nc</i>
-------------------------	-------------------

Description

Given latitude and longitude coordinates, extract site data from NARR file

Usage

```
extract.nc(
  in.path,
  in.prefix,
  outfolder,
  start_date,
  end_date,
  slat,
  slon,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be permuted (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be permuted (will only use the year part of the date)
<code>slat</code>	the latitude of the site

<code>slon</code>	the longitude of the site
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	should output of function be extra verbose
<code>...</code>	further arguments, currently ignored

Author(s)

Betsy Cowdery

<code>extract.nc.ERA5</code>	<i>ERA5_extract</i>
------------------------------	---------------------

Description

ERA5_extract

Usage

```
extract.nc.ERA5(
  slat,
  slon,
  in.path,
  start_date,
  end_date,
  outfolder,
  in.prefix,
  newsite,
  ncores = 1,
  vars = NULL,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>slat</code>	numeric: vector of latitudes.
<code>slon</code>	numeric: vector of longitudes.
<code>in.path</code>	character: path to the directory containing the file to be inserted
<code>start_date</code>	character: start date (in YYYY-MM-DD format).
<code>end_date</code>	character: end date (in YYYY-MM-DD format).
<code>outfolder</code>	character: Path to directory where nc files need to be saved.
<code>in.prefix</code>	character: initial portion of the filename that does not vary by date. Does not include directory; specify that as part of in.path.

newsite	character: vector of site names. The length should match with that of slat and slon.
ncores	numeric: the number of CPUs for the parallel compute. Default is 1.
vars	character: names of variables to be extracted. If NULL all the variables will be returned. Default is NULL.
overwrite	Logical if files needs to be overwritten.
verbose	Decide if we want to stop printing info.
...	other inputs.

Details

For the list of variables check out the documentation at <https://confluence.ecmwf.int/display/CKB/ERA5%3A+data+documentation>

Value

a list of xts objects with all the variables for the requested years

Author(s)

Dongchen Zhang, Akash

Examples

```
## Not run:
point.data <- extract.nc.ERA5(
  slat = 43.25,
  slon = -83.25,
  in.path = "path/to/era5/files",
  start_date = "1990-01-01",
  end_date = "1995-12-31",
  outfolder = "path/to/output",
  in.prefix = "ERA5_",
  newsite = "my_site",
  vars = NULL,
  overwrite = FALSE,
  verbose = TRUE
)

## End(Not run)
```

gen.subdaily.models *Generate Subdaily Models*

Description

Create statistical models to predict subdaily meteorology This is the 2nd function in the tdm workflow that takes the dat.train_file that is created from the nc2dat.train function and generates "lag.days" and "next.days". These variables pass along information of the previous time step and provides a preview of the next time step. After these variables are created, the models are generated by calling the tdm_temporal_downscale_functions.R scripts and these models and betas are saved separately. Please note that these models and betas require a significant amount of space. The storage required varies by the size of the training dataset, but prepare for >100 GB. These will be called later in tdm_predict_subdaily_met to perform the linear regression analysis.

Usage

```
gen.subdaily.models(
  outfolder,
  path.train,
  yrs.train,
  direction.filter = "forward",
  in.prefix,
  n.beta,
  day.window,
  seed = Sys.time(),
  resids = FALSE,
  parallel = FALSE,
  n.cores = NULL,
  overwrite = TRUE,
  verbose = FALSE,
  print.progress = FALSE
)
```

Arguments

outfolder	- directory where models will be stored *** storage required varies by size of training dataset, but prepare for >10 GB
path.train	- path to CF/PEcAn style training data where each year is in a separate file.
yrs.train	- which years of the training data should be used for to generate the model for the subdaily cycle. If NULL, will default to all years
direction.filter	- Whether the model will be filtered backward or forward in time. options = c("backward", "forward") (PaleON will go backward, anybody interested in the future will go forward)

<code>in.prefix</code>	not used
<code>n.betas</code>	- number of betas to save from linear regression model
<code>day.window</code>	- integer specifying number of days around the day being modeled you want to use data from for that specific hours coefficients. Must be integer because we want statistics from the same time of day for each day surrounding the model day
<code>seed</code>	- seed for randomization to allow for reproducible results
<code>resids</code>	- logical stating whether to pass on residual data or not (this increases both memory & storage requirements)
<code>parallel</code>	- logical stating whether to run <code>temporal_downscale_functions.R</code> in parallel
<code>n.cores</code>	- deals with parallelization
<code>overwrite</code>	logical: replace output file if it already exists?
<code>verbose</code>	logical, currently ignored
<code>print.progress</code>	- print progress bar? (gets passed through)

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: `lm_ensemble_sims()`, `model.train()`, `nc.merge()`, `predict_subdaily_met()`, `save.betas()`, `save.model()`, `subdaily_pred()`, `temporal_downscale.functions()`

<code>generate_narr_url</code>	<i>Generate NARR url from a vector of dates</i>
--------------------------------	---

Description

Figures out file names for the given dates, based on NARR's convoluted and inconsistent naming scheme.

Usage

```
generate_narr_url(dates, flx)
```

Arguments

<code>dates</code>	Vector of dates for which to generate URL
<code>flx</code>	(Logical) If 'TRUE', format for 'flx' variables. Otherwise, format for 'sfc' variables. See <code>[narr_flx_vars]</code> .

Author(s)

Alexey Shiklomanov

<code>get.lv</code>	<i>Latent heat of vaporization</i>
---------------------	------------------------------------

Description

calculate latent heat of vaporization for water

Usage

```
get.lv(airtemp = 268.6465)
```

Arguments

<code>airtemp</code>	air temperature (Kelvin)
----------------------	--------------------------

Value

LV latent heat of vaporization (J kg⁻¹)

Author(s)

Istem Fer

<code>get.ncvector</code>	<i>Get time series vector from netCDF file</i>
---------------------------	--

Description

internal convenience function for streamlining extraction of data from netCDF files with CF-compliant variable names

Usage

```
get.ncvector(var, lati = lati, loni = loni, run.dates = run.dates, met.nc)
```

Arguments

<code>var</code>	name of variable to extract
<code>lati, loni</code>	latitude and longitude to extract
<code>run.dates</code>	data frame of dates to read
<code>met.nc</code>	netcdf file with CF variable names

Value

numeric vector

Author(s)

David Shaner LeBauer

get.rh	get RH
--------	--------

Description

Calculate RH from temperature and dewpoint

Usage

get.rh(T, Td)

Arguments

T	air temperature, Kelvin
Td	dewpoint, Kelvin

Details

Based on equation 12 in Lawrence 2005, The Relationship between Relative Humidity and the Dewpoint Temperature in Moist Air A Simple Conversion and Applications. BAMS <https://doi.org/10.1175/BAMS-86-2-225> $R = 461.5 \text{ K}^{-1} \text{ kg}^{-1} \text{ gas constant H}_2\text{O}$ L enthalpy of vaporization linear dependence on T (p 226, following eq 9)

Value

Relative Humidity numeric vector

Author(s)

David LeBauer

get.vpd	VPD
---------	-----

Description

Calculate VPD

Usage

get.vpd(rh, temp)

Arguments

rh	relative humidity, in percent
temp	temperature, degrees celsius

Details

Calculate vapor pressure deficit from relative humidity and temperature.

Value

vpd: vapor pressure deficit, in mb

Author(s)

David LeBauer

Examples

```
temp <- -30:30
plot(temp, get.vpd(0, temp))
```

get_cf_variables_table

Retrieve the current CF variables table from `cfconventions.org` and convert it into a `data.frame`

Description

Retrieve the current CF variables table from `cfconventions.org` and convert it into a `data.frame`

Usage

```
get_cf_variables_table(cf_url = build_cf_variables_table_url(57))
```

Arguments

`cf_url` URL of CF variables table XML. See also [build_cf_variables_table_url](#).

Value

CF variables table, as a `tibble`

Author(s)

Alexey Shiklomanov

get_clowderauth	<i>Authentication lookup helper</i>
-----------------	-------------------------------------

Description

Authentication lookup helper

Usage

```
get_clowderauth(key, user, pass, url, authfile = "~/pecan.clowder.xml")
```

Arguments

key, user, pass	passed unchanged from download.Geostreams call, possibly null
url	matched against <hostname> in authfile, ignored if authfile contains no hostname.
authfile	path to a PEcAn-formatted XML settings file; must contain a <clowder> key

get_NARR_thredds	<i>Retrieve NARR data using thredds</i>
------------------	---

Description

Retrieve NARR data using thredds

Usage

```
get_NARR_thredds(  
  start_date,  
  end_date,  
  lat.in,  
  lon.in,  
  progress = TRUE,  
  drop_outside = TRUE,  
  parallel = TRUE,  
  ncores = 1  
)
```

Arguments

<code>start_date</code>	Start date for meteorology
<code>end_date</code>	End date for meteorology
<code>lat.in</code>	Latitude coordinate
<code>lon.in</code>	Longitude coordinate
<code>progress</code>	Whether or not to show a progress bar (default = 'TRUE'). Requires the 'progress' package to be installed.
<code>drop_outside</code>	Whether or not to drop dates outside of 'start_date' to 'end_date' range (default = 'TRUE').
<code>parallel</code>	Download in parallel? Default = TRUE
<code>ncores</code>	Number of cores for parallel download. Default is 'parallel::detectCores()'

Value

'tibble' containing time series of NARR data for the given site

Author(s)

Alexey Shiklomanov

Examples

```
## Not run:
dat <- get_NARR_thredds("2008-01-01", "2008-01-15", 43.3724, -89.9071)

## End(Not run)
```

<code>get_narr_url</code>	<i>Retrieve NARR data from a given URL</i>
---------------------------	--

Description

Retrieve NARR data from a given URL

Usage

```
get_narr_url(url, xy, flx, pb = NULL)
```

Arguments

<code>url</code>	Full URL to NARR thredds file
<code>xy</code>	Vector length 2 containing NARR coordinates
<code>flx</code>	(Logical) If 'TRUE', format for 'flx' variables. Otherwise, format for 'sfc' variables. See [narr_flx_vars].
<code>pb</code>	Progress bar R6 object (default = 'NULL')

Author(s)

Alexey Shiklomanov

half_hour_downscale	<i>half_hour_downscale</i>
---------------------	----------------------------

Description

half_hour_downscale

Usage

```
temporal_downscale_half_hour(  
  input_file,  
  output_file,  
  overwrite = TRUE,  
  hr = 0.5  
)
```

Arguments

- input_file location of NOAAGEFS_1hr files
- output_file location where to store half_hour files
- overwrite whether to force hamf_hour_downscale to proceed
- hr set half hour

Value

A list of data frames is returned containing information about the data file that can be used to locate it later. Each data frame contains information about one file.

latlon2lcc	<i>Convert latitude and longitude to x-y coordinates (in km) in Lambert conformal conic projection (used by NARR)</i>
------------	---

Description

Convert latitude and longitude to x-y coordinates (in km) in Lambert conformal conic projection (used by NARR)

Usage

```
latlon2lcc(lat.in, lon.in)
```

Arguments

<code>lat.in</code>	Latitude coordinate
<code>lon.in</code>	Longitude coordinate

Value

‘sp::SpatialPoints’ object containing transformed x and y coordinates, in km, which should match NARR coordinates

Author(s)

Alexey Shiklomanov

<code>latlon2narr</code>	<i>Convert latitude and longitude coordinates to NARR indices</i>
--------------------------	---

Description

Convert latitude and longitude coordinates to NARR indices

Usage

```
latlon2narr(nc, lat.in, lon.in)
```

Arguments

<code>nc</code>	‘ncdf4’ connection object
<code>lat.in</code>	Latitude coordinate
<code>lon.in</code>	Longitude coordinate

Value

Vector length 2 containing NARR ‘x’ and ‘y’ indices, which can be used in ‘ncdf4::ncvar_get’ ‘start’ argument.

Author(s)

Alexey Shiklomanov

lightME	<i>Simulates the light macro environment</i>
---------	--

Description

Simulates light macro environment based on latitude, day of the year. Other coefficients can be adjusted.

Usage

```
lightME(lat = 40, DOY = 190, t.d = 12, t.sn = 12, atm.P = 1e+05, alpha = 0.85)
```

Arguments

lat	the latitude, default is 40 (Urbana, IL, U.S.).
DOY	the day of the year (1–365), default 190.
t.d	time of the day in hours (0–23), default 12.
t.sn	time of solar noon, default 12.
atm.P	atmospheric pressure, default 1e5 (kPa).
alpha	atmospheric transmittance, default 0.85.

Value

a **list** structure with components:

- '**I.dir**' Direct radiation ($\mu \text{ mol } m^{-2} s^{-1}$)
- '**I.diff**' Indirect (diffuse) radiation ($\mu \text{ mol } m^{-2} s^{-1}$)
- '**cos.th**' cosine of θ , solar zenith angle.
- '**propIdir**' proportion of direct radiation.
- '**propIdiff**' proportion of indirect (diffuse) radiation.

lm_ensemble_sims	<i>lm_ensemble_sims</i>
------------------	-------------------------

Description

This function does the heavy lifting in the final function of the tdm workflow titled predict_subdaily_met(). It uses a linear regression approach by generating the hourly values from the coarse data of the file the user selects to downscale based on the hourly models and betas generated by gen.subdaily.models().

Usage

```
lm_ensemble_sims(
  dat.mod,
  n.ens,
  path.model,
  direction.filter,
  lags.list = NULL,
  lags.init = NULL,
  dat.train,
  precip.distribution,
  force.sanity = TRUE,
  sanity.tries = 25,
  sanity.sd = 6,
  seed = Sys.time(),
  print.progress = FALSE
)
```

Arguments

dat.mod	- dataframe to be predicted at the time step of the training data
n.ens	- number of hourly ensemble members to generate
path.model	- path to where the training model & betas is stored
direction.filter	- Whether the model will be filtered backward or forward in time. options = c("backward", "forward") (PalEON will go backward, anybody interested in the future will go forward)
lags.list	- optional list form of lags.init, with one entry for each unique 'ens.day' in dat.mod
lags.init	- a data frame of initialization parameters to match the data in dat.mod
dat.train	- the training data used to fit the model; needed for night/day in surface_downwelling_shortwave_flux_in_air
precip.distribution	- a list with 2 sub-lists containing the number of observations with precip in the training data per day & the hour of max rain in the training data. This will be used to help solve the "constant drizzle" problem
force.sanity	- (logical) do we force the data to meet sanity checks?
sanity.tries	- how many time should we try to predict a reasonable value before giving up? We don't want to end up in an infinite loop
sanity.sd	- how many standard deviations from the mean should be used to determine sane outliers (default 6)
seed	- (optional) set the seed manually to allow reproducible results
print.progress	- if TRUE will print progress bar

Details

Linear Regression Ensemble Simulation Met downscaling function that predicts ensembles of downscaled meteorology

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: [gen.subdaily.models\(\)](#), [model.train\(\)](#), [nc.merge\(\)](#), [predict_subdaily_met\(\)](#), [save.betas\(\)](#), [save.model\(\)](#), [subdaily_pred\(\)](#), [temporal.downscale.functions\(\)](#)

load.cfmet	<i>Load met data from PEcAn formatted met driver</i>
------------	--

Description

subsets a PEcAn formatted met driver file and converts to a data.frame object

Usage

```
load.cfmet(met.nc, lat, lon, start.date, end.date)
```

Arguments

<code>met.nc</code>	object of class ncdf4 representing an open CF compliant, PEcAn standard netcdf file with met data
<code>lat</code>	numeric value of latitude
<code>lon</code>	numeric value of longitude
<code>start.date</code>	format is 'YYYY-MM-DD'
<code>end.date</code>	format is 'YYYY-MM-DD'

Value

data frame of met data

Author(s)

David LeBauer

merge_met_variable	<i>Merge a new met variable from an external file (e.g. CO2) into existing met files</i>
--------------------	--

Description

Currently modifies the files IN PLACE rather than creating a new copy of the files an a new DB record. Currently unit and name checking only implemented for CO2. Currently does not yet support merge data that has lat/lon New variable only has time dimension and thus MIGHT break downstream code....

Usage

```
merge_met_variable(
  in.path,
  in.prefix,
  start_date,
  end_date,
  merge.file,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

in.path	path to original data
in.prefix	prefix of original data
start_date, end_date	date (or character in a standard date format). Only year component is used.
merge.file	path of file to be merged in
overwrite	logical: replace output file if it already exists?
verbose	logical: should ncdf4 functions print debugging information as they run?
...	other arguments, currently ignored

Value

Currently nothing. TODO: Return a data frame summarizing the merged files.

Examples

```
## Not run:
in.path    <- "~/paleon/PaleONregional_CF_site_1-24047/"
in.prefix  <- ""
outfolder  <- "~/paleon/metTest/"
```

```

merge.file <- "~/paleon/paleon_monthly_co2.nc"
start_date <- "0850-01-01"
end_date   <- "2010-12-31"
overwrite  <- FALSE
verbose    <- TRUE

merge_met_variable(in.path,in.prefix,start_date,end_date,merge.file,overwrite,verbose)
PEcAn.DALEC::met2model.DALEC(in.path,in.prefix,outfolder,start_date,end_date)

## End(Not run)

```

met.process

met.process

Description

met.process

Usage

```

met.process(
  site,
  input_met,
  start_date,
  end_date,
  model,
  host = "localhost",
  dbparms,
  dir,
  spin = NULL,
  overwrite = FALSE
)

```

Arguments

site	Site info from settings file
input_met	Which data source to process.
start_date	the start date of the data to be downloaded (will only use the year part of the date)
end_date	the end date of the data to be downloaded (will only use the year part of the date)
model	model_type name
host	Host info from settings file
dbparms	database settings from settings file
dir	directory to write outputs to

<code>spin</code>	spin-up settings passed to model-specific <code>met2model</code> . List containing <code>nyear</code> (number of years of spin-up), <code>nsample</code> (first n years to cycle), and <code>resample</code> (TRUE/FALSE)
<code>overwrite</code>	Whether to force <code>met.process</code> to proceed. ‘ <code>overwrite</code> ’ may be a list with individual components corresponding to ‘ <code>download</code> ’, ‘ <code>met2cf</code> ’, ‘ <code>standardize</code> ’, and ‘ <code>met2model</code> ’. If it is instead a simple boolean, the default behavior for ‘ <code>overwrite=FALSE</code> ’ is to overwrite nothing, as you might expect. Note however that the default behavior for ‘ <code>overwrite=TRUE</code> ’ is to overwrite everything <i>*except*</i> raw met downloads. I.e., it corresponds to: <code>list(download = FALSE, met2cf = TRUE, standardize = TRUE, met2model = TRUE)</code> List of ‘ <code>url</code> ’, ‘ <code>username</code> ’, ‘ <code>password</code> ’

Author(s)

Elizabeth Cowdery, Michael Dietze, Ankur Desai, James Simkins, Ryan Kelly

<code>met.process.stage</code>	<i>met.process.stage</i>
--------------------------------	--------------------------

Description

`met.process.stage`

Usage

`met.process.stage(input.id, raw.id, con)`

Arguments

<code>input.id</code>	bety db for input format
<code>raw.id</code>	format id for the raw met data
<code>con</code>	database connection

Author(s)

Elizabeth Cowdery

```
met_temporal_downscale.Gaussian_ensemble
    met_temporal_downscale.Gaussian_ensemble
```

Description

takes source data and a training dataset from the same site and temporally downscales the source dataset to the resolution of the training dataset based on statistics of the training dataset.

Usage

```
met_temporal_downscale.Gaussian_ensemble(
    in.path,
    in.prefix,
    outfolder,
    input_met,
    train_met,
    overwrite = FALSE,
    verbose = FALSE,
    swdn_method = "sine",
    n_ens = 10,
    w_len = 20,
    utc_diff = -6,
    ...
)
```

Arguments

<code>in.path</code>	ignored
<code>in.prefix</code>	ignored
<code>outfolder</code>	path to directory in which to store output. Will be created if it does not exist
<code>input_met</code>	- the source dataset that will temporally downscaled by the train_met dataset
<code>train_met</code>	- the observed dataset that will be used to train the modeled dataset in NC format. i.e. Flux Tower dataset (see <code>download.Fluxnet2015</code> or <code>download.Ameriflux</code>)
<code>overwrite</code>	logical: replace output file if it already exists?
<code>verbose</code>	logical: should <code>ncdf4</code> functions print debugging information as they run?
<code>swdn_method</code>	- Downwelling shortwave flux in air downscaling method (options are "sine", "spline", and "Waichler")
<code>n_ens</code>	- numeric value with the number of ensembles to run
<code>w_len</code>	- numeric value that is the window length in days

`utc_diff` - numeric value in HOURS that is local standard time difference from UTC time. CST is -6

`...` further arguments, currently ignored

Author(s)

James Simkins, Akash

`met2CF.ALMA`

met2CF.ALMA

Description

Get meteorology variables from ALMA netCDF files and convert to netCDF CF format

Usage

```
met2CF.ALMA(
  in.path,
  in.prefix,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE
)
```

Arguments

`in.path` location on disk where inputs are stored

`in.prefix` prefix of input and output files

`outfolder` location on disk where outputs will be stored

`start_date` the start date of the data to be downloaded (will only use the year part of the date)

`end_date` the end date of the data to be downloaded (will only use the year part of the date)

`overwrite` should existing files be overwritten

`verbose` logical: enable verbose mode for netcdf writer functions?

Author(s)

Mike Dietze

met2CF.Ameriflux	<i>met2CF.Ameriflux</i>
------------------	-------------------------

Description

Get meteorology variables from Ameriflux L2 netCDF files and convert to netCDF CF format

Usage

```
met2CF.Ameriflux(  
    in.path,  
    in.prefix,  
    outfolder,  
    start_date,  
    end_date,  
    overwrite = FALSE,  
    verbose = FALSE,  
    ...  
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input and output files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be downloaded (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded (will only use the year part of the date)
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	should output of function be extra verbose
<code>...</code>	further arguments, currently ignored

Author(s)

Josh Mantooth, Mike Dietze, Elizabeth Cowdery, Ankur Desai

met2CF.AmerifluxLBL *met2CF.AmerifluxLBL*

Description

Get meteorology variables from Ameriflux LBL and convert to netCDF CF format

Usage

```
met2CF.AmerifluxLBL(
  in.path,
  in.prefix,
  outfolder,
  start_date,
  end_date,
  format,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input and output files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be downloaded (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded (will only use the year part of the date)
<code>format</code>	is data frame or list with elements as described below The AmerifluxLBL format is Bety record 5000000002 which could be returned from <code>PEcAn.DB::query.format.vars(for</code> format is output from <code>db/R/query.format.vars</code> , and should have: REQUIRED: <code>format\$lat</code> = latitude of site <code>format\$lon</code> = longitude of site <code>format\$header</code> = number of lines of header <code>format\$vars</code> is a <code>data.frame</code> with lists of information for each variable to read, at least <code>airT</code> is required <code>format\$vars\$input_name</code> = Name in CSV file <code>format\$vars\$input_units</code> = Units in CSV file <code>format\$vars\$bety_name</code> = Name in BETY OPTIONAL: <code>format\$na.strings</code> = list of missing values to convert to NA, such as -9999 <code>format\$skip</code> = lines to skip excluding header <code>format\$vars\$column_number</code> = Column number in CSV file (optional, will use header name first) Columns with NA for bety variable name are dropped. Units for datetime field are the lubridate function that will be used to parse the date (e.g. <code>ymd_hms</code> or <code>mdy_hm</code>).
<code>overwrite</code>	should existing files be overwritten

verbose should output of function be extra verbose
... further arguments, currently ignored

Author(s)

Ankur Desai

met2CF.csv

Convert met data from CSV to CF

Description

Convert met data from CSV to CF

Usage

```
met2CF.csv(  
  in.path,  
  in.prefix,  
  outfolder,  
  start_date,  
  end_date,  
  format,  
  lat = NULL,  
  lon = NULL,  
  nc_verbose = FALSE,  
  overwrite = FALSE,  
  ...  
)
```

Arguments

in.path directory in which to find met csv files
in.prefix pattern to match to find met files inside ‘in.path’
outfolder directory name to write CF outputs
start_date, end_date when to start and stop conversion. Specify as ‘Date’ objects, but only the year component is used
format data frame or list produced by ‘PEcAn.DB::query.format.vars’. See details
lat, lon latitude and longitude of site, in decimal degrees. If not provided, these are taken from ‘format’.
nc_verbose logical: run nvar_add in verbose mode?
overwrite Logical: Redo conversion if output file already exists?
... other arguments, currently ignored

Details

The ‘format’ argument takes an output from ‘PEcAn.DB::query.format.vars’, and should have the following components:

REQUIRED:

- ‘format\$lat’: latitude of site (unless passed by ‘lat’)
- ‘format\$lon’: longitude of site (unless passed by ‘lon’)
- ‘format\$header’: number of lines of header
- ‘format\$vars’: a data.frame with lists of information for each variable to read. At least ‘airT’ is required
- ‘format\$vars\$input_name’: name in CSV file
- ‘format\$vars\$input_units’: units in CSV file
- ‘format\$vars\$bety_name’: name in BETY. See <https://pecanproject.github.io/pecan-documentation/develop/NewInput.html#InputConversions> for allowable names.

OPTIONAL:

- ‘format\$na.strings’: list of missing values to convert to NA, such as -9999
- ‘format\$skip’: lines to skip excluding header
- ‘format\$vars\$column_number’: column number in CSV file (optional, will use header name first)

Columns with NA for bety variable name are dropped.

Units for datetime field are the lubridate function that will be used to parse the date (e.g. ymd_hms or mdy_hm).

Author(s)

Mike Dietze, David LeBauer, Ankur Desai

Examples

```
## Not run:
con <- PEcAn.DB::db.open(
  list(user='bety', password='bety', host='localhost',
    dbname='bety', driver='PostgreSQL',write=TRUE))
start_date <- lubridate::ymd_hm('200401010000')
end_date <- lubridate::ymd_hm('200412312330')
file<-PEcAn.data.atmosphere::download.Fluxnet2015('US-WCr','~/',start_date,end_date)
in.path <- '~/ '
in.prefix <- file$dbfile.name
outfolder <- '~/ '
format.id <- 5000000001
format <- PEcAn.DB::query.format.vars(format.id=format.id,bety = bety)
format$lon <- -92.0
format$lat <- 45.0
format$time_zone <- "America/Chicago"
results <- PEcAn.data.atmosphere::met2CF.csv(
```

```

    in.path, in.prefix, outfolder,
    start_date, end_date, format,
    overwrite=TRUE)

## End(Not run)

```

met2CF.ERA5

met2cf.ERA5

Description

met2cf.ERA5

Usage

```

met2CF.ERA5(
  lat,
  long,
  start_date,
  end_date,
  sitename,
  outfolder,
  out.xts,
  overwrite = FALSE,
  verbose = TRUE,
  ens_size = 1
)

```

Arguments

lat	latitude
long	longitude
start_date	start date
end_date	end date
sitename	The name of the site used for making the identifier.
outfolder	Path to directory where nc files need to be saved.
out.xts	Output of the extract.nc.ERA5 function which is a list of time series of met variables for each ensemble member or single reanalysis dataset.
overwrite	Logical if files needs to be overwritten.
verbose	Logical flag defining if ouput of function be extra verbose.
ens_size	Number of ensemble members to process. Default is 1.

Value

list of dataframes

Author(s)

Hamze Dokohaki, Akash

met2CF.FACE	<i>convert FACE files to CF files</i>
-------------	---------------------------------------

Description

Note: ‘in.path’ and ‘in.prefix’ together must identify exactly one file, or this function returns NULL. Further note that despite its name, ‘in.prefix’ will match anywhere in the filename: met2CF.FACE(”dir”, ”a”, ...)‘ will find both ‘dir/a__b.nc’ and ‘dir/b__a.nc’!

Usage

```
met2CF.FACE(  
  in.path,  
  in.prefix,  
  outfolder,  
  start_date,  
  end_date,  
  input.id,  
  site,  
  format,  
  ...  
)
```

Arguments

- in.path directory in which to find inputs (as ‘*.nc’)
- in.prefix pattern to match to select a file within ‘in.path’
- outfolder path to write output. Should contain the substring ”FACE”, which will be rewritten to ”FACE_a” and ”FACE_e” for the corresponding treatments.
- start_date, end_date ignored. Time is taken from the input files.
- input.id ignored
- site list[like]. Only components ‘lat’ and ‘lon’ (both in decimal degrees) are currently used
- format specification of variable names and units in the format returned by ‘PEcAn.DB::query.format.vars
- ... other arguments, currently ignored

Author(s)

Elizabeth Cowdery

met2CF.Geostreams	<i>Convert geostreams JSON to CF met file</i>
-------------------	---

Description

Convert geostreams JSON to CF met file

Usage

```
met2CF.Geostreams(  
    in.path,  
    in.prefix,  
    outfolder,  
    start_date,  
    end_date,  
    overwrite = FALSE,  
    verbose = FALSE,  
    ...  
)
```

Arguments

<code>in.path</code>	directory containing Geostreams JSON file(s) to be converted
<code>in.prefix</code>	initial portion of input filenames (everything before the dates)
<code>outfolder</code>	directory where nc output files should be written. Will be created if necessary
<code>start_date, end_date</code>	beginning and end of run, YYYY-MM-DD.
<code>overwrite</code>	logical: Regenerate existing files of the same name?
<code>verbose</code>	logical, passed on to <code>nc_create</code> to control how chatty it should be during netCDF creation
<code>...</code>	other arguments, currently ignored

Author(s)

Harsh Agrawal, Chris Black

met2CF.ICOS

Convert variables ICOS variables to CF format.

Description

Variables present in the output netCDF file: air_temperature, air_temperature, relative_humidity, specific_humidity, water_vapor_saturation_deficit, surface_downwelling_longwave_flux_in_air, surface_downwelling_shortwave_flux_in_air, surface_downwelling_photosynthetic_photon_flux_in_air, precipitation_flux, eastward_wind, northward_wind

Usage

```
met2CF.ICOS(
    in.path,
    in.prefix,
    outfolder,
    start_date,
    end_date,
    format,
    overwrite = FALSE,
    ...
)
```

Arguments

<code>in.path</code>	path to the input ICOS product CSV file
<code>in.prefix</code>	name of the input file
<code>outfolder</code>	path to the directory where the output file is stored. If specified directory does not exists, it is created.
<code>start_date</code>	start date of the input file
<code>end_date</code>	end date of the input file
<code>format</code>	format is data frame or list with elements as described below REQUIRED: <code>format\$header</code> = number of lines of header <code>format\$vars</code> is a data.frame with lists of information for each variable to read, at least <code>airT</code> is required <code>format\$vars\$input_name</code> = Name in CSV file <code>format\$vars\$input_units</code> = Units in CSV file <code>format\$vars\$bety_name</code> = Name in BETY OPTIONAL: <code>format\$lat</code> = latitude of site <code>format\$lon</code> = longitude of site <code>format\$na.strings</code> = list of missing values to convert to NA, such as -9999 <code>format\$skip</code> = lines to skip excluding header <code>format\$vars\$column_number</code> = Column number in CSV file (optional, will use header name first) Columns with NA for bety variable name are dropped.
<code>overwrite</code>	overwrite should existing files be overwritten. Default False.
<code>...</code>	used when extra arguments are present.

Value

information about the output file

met2CF.NARR	<i>Convert NARR files to CF files</i>
-------------	---------------------------------------

Description

Convert NARR files to CF files

Usage

```
met2CF.NARR(
  in.path,
  in.prefix,
  outfolder,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

in.path	directory in which to find NARR files
in.prefix	pattern to match to find NARR files inside ‘in.path’
outfolder	directory name to write CF outputs
start_date	the start date of the data to be downloaded (will only use the year part of the date)
end_date	the end date of the data to be downloaded (will only use the year part of the date)
overwrite	should existing files be overwritten
verbose	should output of function be extra verbose
...	other arguments, currently ignored

Author(s)

Elizabeth Cowdery, Rob Kooper

met2CF.PalEON	<i>met2CF.PalEON</i>
---------------	----------------------

Description

Get meteorology variables from PalEON netCDF files and convert to netCDF CF format

Usage

```
met2CF.PalEON(
    in.path,
    in.prefix,
    outfolder,
    start_date,
    end_date,
    lat,
    lon,
    overwrite = FALSE,
    verbose = FALSE,
    ...
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input and output files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be downloaded (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded (will only use the year part of the date)
<code>lat, lon</code>	site location in decimal degrees. Caution: both must have length one.
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	logical: enable verbose mode for netcdf writer functions?
<code>...</code>	further arguments, currently ignored

Author(s)

Mike Dietze

```
met2CF.PalEONregional
```

met2CF.PalEONregional

Description

Get meteorology variables from PalEON netCDF files and convert to netCDF CF format

Usage

```
met2CF.PalEONregional(  
    in.path,  
    in.prefix,  
    outfolder,  
    start_date,  
    end_date,  
    overwrite = FALSE,  
    verbose = FALSE,  
    ...  
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input and output files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be downloaded (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded (will only use the year part of the date)
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	logical: enable verbose mode for netcdf writer functions?
<code>...</code>	further arguments, currently ignored

Author(s)

Mike Dietze

<code>metgapfill</code>	<i>Take an Ameriflux NetCDF file Fill missing met values using MDS approach using MPI-BGC REddyProc library Currently Future version: Choose which variables to gap fill Future version will first downscale and fill with NARR, then REddyProc</i>
-------------------------	---

Description

Take an Ameriflux NetCDF file Fill missing met values using MDS approach using MPI-BGC REddyProc library Currently Future version: Choose which variables to gap fill Future version will first downscale and fill with NARR, then REddyProc

Usage

```
metgapfill(
    in.path,
    in.prefix,
    outfolder,
    start_date,
    end_date,
    lst = 0,
    overwrite = FALSE,
    verbose = FALSE,
    ...
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input and output files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be downloaded (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be downloaded (will only use the year part of the date)
<code>lst</code>	is timezone offset from UTC, if timezone is available in time:units attribute in file, it will use that, default is to assume UTC
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	should the function be very verbose
<code>...</code>	further arguments, currently ignored

Author(s)

Ankur Desai

`metgapfill.NOAA_GEFS` *Gapfill NOAA_GEFS weather data*

Description

This function uses simple methods to gapfill NOAA GEFS met data. Temperature and Precipitation are gapfilled with splines; other data sources are gapfilled using linear models fitted to other fitted data.

Usage

```
metgapfill.NOAA_GEFS(  
  in.prefix,  
  in.path,  
  outfolder,  
  start_date,  
  end_date,  
  overwrite = FALSE,  
  verbose = FALSE,  
  ...  
)
```

Arguments

<code>in.prefix</code>	the met file name
<code>in.path</code>	The location of the file
<code>outfolder</code>	The place to write the output file to
<code>start_date</code>	The start date of the contents of the file
<code>end_date</code>	The end date of the contents of the file
<code>overwrite</code>	Whether or not to overwrite the output file if it exists or not
<code>verbose</code>	Passed to nc writing functions for additional output
<code>...</code>	further arguments, currently ignored

Author(s)

Luke Dramko

model.train	model.train
-------------	-------------

Description

Function to create linear regression models for specific met variables. This is used in conjunction with `temporal.downscale.functions()` to generate linear regression statistics and save their output to be called later in `lm_ensemble_sims()`.

Usage

```
model.train(dat.subset, v, n.beta, resids = resids, threshold = NULL, ...)
```

Arguments

<code>dat.subset</code>	data.frame containing lags, next, and downscale period data
<code>v</code>	variable name, as character
<code>n.beta</code>	number of betas to pull from
<code>resids</code>	TRUE or FALSE, whether to use residuals or not
<code>threshold</code>	NULL except for <code>surface_downwelling_shortwave_radiation</code> , helps with our distinction between day and night (no shortwave without sunlight)
<code>...</code>	further arguments, currently ignored

Details

TDM Model Train Linear regression calculations for specific met variables

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: [gen.subdaily.models\(\)](#), [lm_ensemble_sims\(\)](#), [nc.merge\(\)](#), [predict_subdaily_met\(\)](#), [save.betas\(\)](#), [save.model\(\)](#), [subdaily_pred\(\)](#), [temporal.downscale.functions\(\)](#)

narr_flux_vars	<i>NARR flux and sfc variables</i>
----------------	------------------------------------

Description

NARR flux and sfc variables

Usage

```
narr_flux_vars
```

```
narr_sfc_vars
```

```
narr_all_vars
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 5 rows and 3 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 3 rows and 3 columns.

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 8 rows and 3 columns.

<code>nc.merge</code>	<i>nc.merge</i>
-----------------------	-----------------

Description

This is the 1st function for the tdm (Temporally Downscale Meteorology) workflow. The `nc2dat.train` function parses multiple netCDF files into one central training data file called 'dat.train_file'. This netCDF file will be used to generate the subdaily models in the next step of the workflow, `generate.subdaily.models()`. It is also called in `tdm_predict_subdaily_met` which is the final step of the tdm workflow.

Usage

```
nc.merge(
  outfolder,
  in.path,
  in.prefix,
  start_date,
  end_date,
  upscale = FALSE,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

<code>outfolder</code>	- directory where output will be stored
<code>in.path</code>	- path of coarse model (e.g. GCM output)
<code>in.prefix</code>	- prefix of model string as character (e.g. IPSL.r1i1p1.rcp85)
<code>start_date</code>	- yyyy-mm-dd
<code>end_date</code>	- yyyy-mm-dd
<code>upscale</code>	- Upscale can either be set for FALSE (leave alone) or to the temporal resolution you want to aggregate to
<code>overwrite</code>	logical: replace output file if it already exists?
<code>verbose</code>	logical: should <code>ncdf4</code> functions print debugging information as they run?
<code>...</code>	further arguments, currently ignored

Details

`nc.merge` Parses multiple netCDF files into one central document for temporal downscaling procedure

Author(s)

James Simkins, Christy Rollinson

See Also

Other tdm - Temporally Downscale Meteorology: `gen.subdaily.models()`, `lm_ensemble_sims()`, `model.train()`, `predict_subdaily_met()`, `save.betas()`, `save.model()`, `subdaily_pred()`, `temporal.downscale.functions()`

<code>noaa_grid_download</code>	<i>Download gridded forecast in the box bounded by the latitude and longitude list</i>
---------------------------------	--

Description

Download gridded forecast in the box bounded by the latitude and longitude list

Usage

```
noaa_grid_download(
  lat_list,
  lon_list,
  forecast_time,
  forecast_date,
  model_name_raw,
  output_directory,
  end_hr
)
```

Arguments

<code>lat_list</code>	lat for site
<code>lon_list</code>	long for site
<code>forecast_time</code>	start hour of forecast
<code>forecast_date</code>	date for forecast
<code>model_name_raw</code>	model name for directory creation
<code>output_directory</code>	output directory
<code>end_hr</code>	end hr to determine how many hours to download

Value

NA

<code>noaa_stage2</code>	<i>noaa_stage2</i>
--------------------------	--------------------

Description

noaa_stage2

Usage

```
noaa_stage2(
  cycle = 0,
  version = "v12",
  endpoint = "data.ecoforecast.org",
  verbose = TRUE,
  start_date = ""
)
```

Arguments

<code>cycle</code>	Hour at which forecast was made, as character string ("00", "06", "12" or "18"). Only "00" (default) has 30 days horizon.
<code>version</code>	GEFS forecast version. Prior versions correspond to forecasts issued before 2020-09-25 which have different ensemble number and horizon, among other changes, and are not made available here. Leave as default.
<code>endpoint</code>	the EFI host address (leave as default)
<code>verbose</code>	logical, displays or hides messages
<code>start_date</code>	forecast start date yyyy-mm-dd format

Author(s)

Alexis Helgeson (taken from neon4cast package)

<code>par2ppfd</code>	<i>par2ppfd</i>
-----------------------	-----------------

Description

convert PAR to PPFD

Usage

`par2ppfd(watts)`

Arguments

<code>watts</code>	PAR (W / m2)
--------------------	--------------

Details

Converts photosynthetically active radiation (PAR, units of Watts / m2) to photosynthetic photon flux density (PPFD) in units of umol / m2 / s From Campbell and Norman p151
 $PPFD = PAR * (J/m2/s) * (1 \text{ mol} / 2.35e5 \text{ J})$ 2.35e5 J / mol is the energy content of solar radiation in the PAR waveband

Value

PPFD (umol / m2 / s)

Author(s)

David LeBauer

<code>pecan_standard_met_table</code>	<i>Conversion table for PEcAn standard meteorology</i>
---------------------------------------	--

Description

Conversion table for PEcAn standard meteorology

Usage

`pecan_standard_met_table`

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 20 rows and 9 columns.

`permute.nc`*permute.nc*

Description

Permute netCDF files

Usage

```
permute.nc(  
  in.path,  
  in.prefix,  
  outfolder,  
  start_date,  
  end_date,  
  overwrite = FALSE,  
  verbose = FALSE,  
  ...  
)
```

Arguments

<code>in.path</code>	location on disk where inputs are stored
<code>in.prefix</code>	prefix of input and output files
<code>outfolder</code>	location on disk where outputs will be stored
<code>start_date</code>	the start date of the data to be permuted (will only use the year part of the date)
<code>end_date</code>	the end date of the data to be permuted (will only use the year part of the date)
<code>overwrite</code>	should existing files be overwritten
<code>verbose</code>	should output of function be extra verbose
<code>...</code>	further arguments, currently ignored

Author(s)

Elizabeth Cowdery, Rob Kooper

post_process	<i>Post process raw NARR downloaded data frame</i>
--------------	--

Description

Post process raw NARR downloaded data frame

Usage

```
post_process(dat)
```

Arguments

dat	Nested ‘tibble’ from mapped call to [get_narr_url]
-----	--

predict_subdaily_met	<i>predict_subdaily_met</i>
----------------------	-----------------------------

Description

This is the main function of the tdm family workflow. This function predicts subdaily meteorology from daily means using a linear regression modeling approach. It takes a dataset with daily resolution and temporally downscales it to hourly resolution using the statistics generated by `gen.subdaily.models()`. It references the `predict.subdaily.function` located in `lm_ensemble_sims()` which uses a linear regression based approach to downscale. We generate multiple ensembles of possible hourly values dictated from the models and betas generated in `gen.subdaily.models`. Each ensemble member is saved as a netCDF file in CF conventions and these files are ready to be used in the general PEcAn workflow.

Usage

```
predict_subdaily_met(
  outfolder,
  in.path,
  in.prefix,
  path.train,
  direction.filter = "forward",
  lm.models.base,
  yrs.predict = NULL,
  ens.labs = 1:3,
  resids = FALSE,
  adjust.pr = 1,
  force.sanity = TRUE,
  sanity.tries = 25,
  overwrite = FALSE,
  verbose = FALSE,
```

```

    seed = format(Sys.time(), "%m%d"),
    print.progress = FALSE,
    ...
)

```

Arguments

outfolder	- directory where output file will be stored
in.path	- base path to dataset you wish to temporally downscale; Note: in order for parallelization to work, the in.prefix will need to be appended as the final level of the file structure. For example, if prefix is GFDL.CM3.rcp45.r1i1p1, there should be a directory with that title in in.path.
in.prefix	- prefix of model dataset, i.e. if file is GFDL.CM3.rcp45.r1i1p1.2006 the prefix is 'GFDL.CM3.rcp45.r1i1p1'
path.train	- path to CF/PEcAn style training data where each year is in a separate file.
direction.filter	- Whether the model will be filtered backward or forwards in time. options = c("backward", "forwards") (default is forward; PalEON will go backward, anybody interested in the future will go forwards)
lm.models.base	- path to linear regression model folders generated using gen.subdaily.models
yrs.predict	- years for which you want to generate met. if NULL, all years in in.path will be done
ens.labs	- vector containing the labels (suffixes) for each ensemble member; this allows you to add to your ensemble rather than overwriting with a default naming scheme
resids	- logical stating whether to pass on residual data or not
adjust.pr	- adjustment factor for precipitation when the extracted values seem off
force.sanity	- (logical) do we force the data to meet sanity checks?
sanity.tries	- how many time should we try to predict a reasonable value before giving up? We don't want to end up in an infinite loop
overwrite	logical: replace output file if it already exists?
verbose	logical: should ncdf4 functions print debugging information as they run?
seed	- manually set seed for results to be reproducible
print.progress	- print the progress bar?
...	further arguments, currently ignored

Details

Predict Subdaily Meteorology Predict Subdaily Meteorology based off of statistics created in gen.subdaily.models()

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: `gen.subdaily.models()`, `lm_ensemble_sims()`, `model.train()`, `nc.merge()`, `save.betas()`, `save.model()`, `subdaily_pred()`, `temporal.downscale.functions`

Examples

```
## Not run:
predict_subdaily_met(
  outfolder = "~/Downscaled_GCM",
  in.path = "~/raw_GCM",
  in.prefix = "GFDL",
  path.train = "~/Training_data",
  lm.models.base = "sf_scratch/US-WCr",
  ens.labs = 1:3
)

## End(Not run)
```

prepare_narr_year	<i>Write NetCDF file for a single year of data</i>
-------------------	--

Description

Write NetCDF file for a single year of data

Usage

```
prepare_narr_year(dat, file, lat_nc, lon_nc, verbose = FALSE)
```

Arguments

dat	NARR tabular data for a single year ([get_NARR_thredds])
file	Full path to target file
lat_nc	‘ncdim’ object for latitude
lon_nc	‘ncdim’ object for longitude
verbose	logical: ask ‘ncdf4’ functions to be very chatty while they work?

Value

List of NetCDF variables in data. Creates NetCDF file containing data as a side effect

```
process_gridded_noaa_download
```

Extract and temporally downscale points from downloaded grid files

Description

Extract and temporally downscale points from downloaded grid files

Usage

```
process_gridded_noaa_download(
  lat_list,
  lon_list,
  site_id,
  downscale,
  overwrite,
  forecast_date,
  forecast_time,
  model_name,
  model_name_ds,
  model_name_raw,
  output_directory
)
```

Arguments

<code>lat_list</code>	lat for site
<code>lon_list</code>	lon for site
<code>site_id</code>	Unique <code>site_id</code> for file creation
<code>downscale</code>	Logical. Default is TRUE. Downscales from 6hr to hourly
<code>overwrite</code>	Logical. Default is FALSE. Should exisiting files be overwritten
<code>forecast_date</code>	Date for download
<code>forecast_time</code>	Time (0,6,12,18) for start of download
<code>model_name</code>	Name of model for file name
<code>model_name_ds</code>	Name of downscale file name
<code>model_name_raw</code>	Name of raw file name
<code>output_directory</code>	Output directory

Value

List

qair2rh	<i>qair2rh</i>
---------	----------------

Description

Convert specific humidity to relative humidity

Usage

```
qair2rh(qair, temp, press = 1013.25)
```

Arguments

qair	specific humidity, dimensionless (e.g. kg/kg) ratio of water mass / total air mass
temp	degrees C
press	pressure in mb

Details

converting specific humidity into relative humidity NCEP surface flux data does not have RH from Bolton 1980 Teh computation of Equivalent Potential Temperature https://archive.eol.ucar.edu/projects/ceop/dm/documents/refdata_report/eqns.html

Value

rh relative humidity, ratio of actual water mixing ratio to saturation mixing ratio

Author(s)

David LeBauer

read.register	<i>read.register</i>
---------------	----------------------

Description

read.register

Usage

```
read.register(register.xml, con)
```

Arguments

register.xml	path of xml file
con	betydb connection

Author(s)

Betsy Cowdery

read_narr_var	<i>Read a specific variable from a NARR NetCDF file</i>
---------------	---

Description

Read a specific variable from a NARR NetCDF file

Usage

```
read_narr_var(nc, xy, variable, unit, flx, pb = NULL)
```

Arguments

nc	'ncdf4' connection object
xy	Vector length 2 containing NARR coordinates
variable	NARR name of variable to retrieve
unit	Output unit of variable to retrieve
flx	(Logical) If 'TRUE', format for 'flx' variables. Otherwise, format for 'sfc' variables. See [narr_flx_vars].
pb	Progress bar R6 object (default = 'NULL')

Author(s)

Alexey Shiklomanov

rh2qair	<i>RH to SH</i>
---------	-----------------

Description

converts relative humidity to specific humidity

Usage

```
rh2qair(rh, T, press = 101325)
```

Arguments

rh	relative humidity (proportion, not %)
T	absolute temperature (Kelvin)
press	air pressure (Pascals)

Author(s)

Mike Dietze, Ankur Desai

sat_vapor_pressure	<i>Saturation vapor pressure (t2es)</i>
--------------------	---

Description

Compute saturation vapor pressure from temperature using one of the following methods:

- (Default) Clausius–Clapeyron (FAO-56 style) — Recommended for most applications. Commonly used approximation for terrestrial ecosystem models, consistent with Penman-Monteith and FAO-56 (Allen et al, 1998).
- Magnus — More accurate in the range -40 to $+50$ C. Coefficients as in Alduchov & Eskridge (1996).
- Goff–Gratch - Highest accuracy; use when following WMO-style recommendations. Goff–Gratch 1946; WMO, 2014.

Usage

```
sat_vapor_pressure(
    temp,
    temp_units = "degC",
    out_units = "kPa",
    method = c("ClausiusClapeyron", "Magnus", "GoffGratch")
)

get.es(temp)

t2es(
    temp,
    temp_units = "degC",
    out_units = "kPa",
    method = "ClausiusClapeyron"
)
```

Arguments

temp	numeric vector of temperatures
temp_units	input temperature units ("degC", "K", "degF"), default "degC"
out_units	output pressure units ("kPa", "hPa", "Pa", "mb"), default "kPa"
method	one of "Magnus", "ClausiusClapeyron" (default), or "GoffGratch". See details for references.

Details

Each method uses different units internally, users can specify units for both inputs and outputs, with defaults "degC" and "kPa", respectively.

Value

numeric vector in `out_units`

Author(s)

David LeBauer

References

- Alduchov, O. A., & Eskridge, R. E. (1996). Improved Magnus Form Approximation of Saturation Vapor Pressure. *J. Appl. Meteor.**, 35(4), 601–609. <doi:10.1175/1520-0450(1996)035<0601:IMFAOS>2.0.CO;2>
- Allen, R. G., Pereira, L. S., Raes, D., & Smith, M. (1998). **Crop evapotranspiration – Guidelines for computing crop water requirements**. FAO Irrigation and Drainage Paper 56.
- Goff, J. A., & Gratch, S. (1946). Low-pressure properties of water from –160 to 212F. *Trans. ASHVE*, 52, 95–122.
- WMO (2014) Guide to Instruments and Methods of Observation (WMO-No. 8), ch. 4.

Examples

```
# Calculate saturation vapor pressure at 20°C
sat_vapor_pressure(20)
t2es(20)

# Using different methods
t2es(c(10, 20, 30), method = "Magnus")
t2es(283.15, temp_units = "K", method = "GoffGratch")

# Different output units
t2es(20, out_units = "hPa")
```

`save.betas`

save.betas

Description

Function to save betas as a .nc file. This is utilized in `gen.subdaily.models()` when linear regression models are created

Usage

```
save.betas(model.out, betas, outfile)
```

Arguments

<code>model.out</code>	list linear regression model output
<code>betas</code>	name of the layer of betas to save (e.g. 'betas' or 'betas.resid')
<code>outfile</code>	location where output will be stored

Details

TDM Save Betas Saves betas that are calculated during `gen.subdaily.models()`

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: `gen.subdaily.models()`, `lm_ensemble_sims()`, `model.train()`, `nc.merge()`, `predict_subdaily_met()`, `save.model()`, `subdaily_pred()`, `temporal.downscale.functions()`

`save.model`
save.model

Description

Function to save models as a .nc file. This is utilized in `gen.subdaily.models()` when linear regression models are created

Usage

```
save.model(model.out, model, outfile)
```

Arguments

<code>model.out</code>	list linear regression model output
<code>model</code>	name of the layer of model to save (e.g. 'model' or 'model.resid')
<code>outfile</code>	location where output will be stored

Details

TDM Save Models Saves models that are created during `gen.subdaily.models()`

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: `gen.subdaily.models()`, `lm_ensemble_sims()`, `model.train()`, `nc.merge()`, `predict_subdaily_met()`, `save.betas()`, `subdaily_pred()`, `temporal.downscale.functions()`

<code>site.lst</code>	<i>Find time zone for a site</i>
-----------------------	----------------------------------

Description

Find time zone for a site

Usage

```
site.lst(site.id, con)
```

Arguments

<code>site.id</code>	bety id of site to look up
<code>con</code>	betydb connection object

Author(s)

Betsy Cowdery

<code>site_from_tag</code>	<i>Function to find the site code for a specific tag</i>
----------------------------	--

Description

Example: `sitename = 'Rhinelanders Aspen FACE Experiment (FACE-RHIN)'` `tag = 'FACE'`
`site_from_tag(sitename,tag) = 'RHIN'` Requires that site names be set up specifically with (tag-sitecode) - this may change

Usage

```
site_from_tag(sitename, tag)
```

Arguments

<code>sitename</code>	full name of site
<code>tag</code>	abbreviated name of site

Author(s)

Betsy Cowdery

solarMJ2ppfd	<i>MJ to PPFD</i>
--------------	-------------------

Description

Solar Radiation to PPFD

Usage

```
solarMJ2ppfd(solarMJ)
```

Arguments

solarMJ	MJ per day
---------	------------

Details

There is no easy straight way to convert MJ/m² to $\mu\text{mol photons} / \text{m}^2 / \text{s}$ (PAR). Note: 1 Watt = 1J/s The above conversion is based on the following reasoning 0.12 is about how much of the total radiation is expected to occur during the hour of maximum insolation (it is a guesstimate) 2.07 is a coefficient which converts from MJ to mol photons (it is approximate and it is taken from ... Campbell and Norman (1998). Introduction to Environmental Biophysics. pg 151 'the energy content of solar radiation in the PAR waveband is $2.35 \times 10^5 \text{ J/mol}$ ' See also the chapter radiation basics (10) Here the input is the total solar radiation so to obtain in the PAR spectrum need to multiply by 0.486 This is based on the approximation that PAR is 0.45-0.50 of the total radiation This means that $1\text{e}6 / (2.35\text{e}6) * 0.486 = 2.07 \text{e}6$ converts from mol to μmol 1/3600 divides the values in hours to seconds

Value

PPFD $\mu\text{mol} / \text{m}^2 / \text{s}$

Author(s)

Fernando Miguez

David LeBauer

spin.met

*Spin-up meteorology***Description**

Spin-up meteorology

Usage

```
spin.met(
  in.path,
  in.prefix,
  start_date,
  end_date,
  nyear = 1000,
  nsample = 50,
  resample = TRUE,
  run_start_date = start_date,
  overwrite = TRUE
)
```

Arguments

<code>in.path</code>	met input folder path
<code>in.prefix</code>	met input file prefix (shared by all annual files, can be "")
<code>start_date</code>	start of met
<code>end_date</code>	end of met
<code>nyear</code>	number of years of spin-up, default 1000
<code>nsample</code>	sample the first nsample years of met, default 50
<code>resample</code>	resample (TRUE, default) or cycle (FALSE) meteorology
<code>run_start_date</code>	date the run itself starts, which can be different than the start of met
<code>overwrite</code>	whether to replace previous resampling

Details

spin.met works by creating symbolic links to the sampled met file, rather than copying the whole file. Be aware that the internal dates in those files are not modified. Right now this is designed to be called within met2model.[MODEL] before the met is processed (it's designed to work with annual CF files not model-specific files) for example with models that process met into one large file

Value

updated start date

Examples

```

start_date <- "0850-01-01 00:00:00"
end_date   <- "2010-12-31 23:59:59"
nyear      <- 10
nsample    <- 50
resample   <- TRUE

## Not run:
if(!is.null(spin)){
  ## if spinning up, extend processed met by resampling or cycling met
  start_date <- PEcAn.data.atmosphere::spin.met(
    in.path, in.prefix,
    start_date, end_date,
    nyear, nsample, resample)
}

## End(Not run)

```

split_wind	<i>Split wind_speed into eastward_wind and northward_wind</i>
------------	---

Description

Currently modifies the files IN PLACE rather than creating a new copy of the files an a new DB record.

Usage

```

split_wind(
  in.path,
  in.prefix,
  start_date,
  end_date,
  overwrite = FALSE,
  verbose = FALSE,
  ...
)

```

Arguments

in.path	path to original data
in.prefix	prefix of original data
start_date, end_date	date (or character in a standard date format). Only year component is used.
overwrite	logical: replace output file if it already exists?
verbose	logical: should ncdf4 functions print debugging information as they run?
...	other arguments, currently ignored

Value

nothing. TODO: Return data frame summarizing results

Examples

```
## Not run:
in.path    <- "~/paleon/PaleONregional_CF_site_1-24047/"
in.prefix  <- ""
outfolder  <- "~/paleon/metTest/"
start_date <- "0850-01-01"
end_date   <- "2010-12-31"
overwrite  <- FALSE
verbose    <- TRUE

split_wind(in.path, in.prefix, start_date, end_date, merge.file, overwrite, verbose)

## End(Not run)
```

<code>step_means</code>	<i>take mean at fixed intervals along a vector</i>
-------------------------	--

Description

take mean at fixed intervals along a vector

Usage

```
step_means(x, step)
```

Arguments

<code>x</code>	numeric vector
<code>step</code>	integer step size

Details

User should check that `length(x)` is an even multiple of `step`

Value

numeric of length `length(x)/step`

subdaily_pred	<i>subdaily_pred</i>
---------------	----------------------

Description

Function to pull objects created in linear regression models and are used to predict subdaily meteorology. This function is called in `lm_ensemble_sims()` to downscale a meteorology product. Linear regression models are created in `gen.subdaily.models()`

Usage

```
subdaily_pred(
  newdata,
  model.predict,
  Rbeta,
  resid.err = FALSE,
  model.resid = NULL,
  Rbeta.resid = NULL,
  n.ens
)
```

Arguments

<code>newdata</code>	dataframe with data to be downscaled
<code>model.predict</code>	saved linear regression model
<code>Rbeta</code>	matrix with Rbetas from saved linear regression model
<code>resid.err</code>	logical, whether to include residual error or not
<code>model.resid</code>	data.frame of model residuals
<code>Rbeta.resid</code>	data.frame of Rbeta residuals
<code>n.ens</code>	number of ensembles to create

Details

Subdaily Prediction Pulls information from linear regression models to predict subdaily meteorology

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: `gen.subdaily.models()`, `lm_ensemble_sims()`, `model.train()`, `nc.merge()`, `predict_subdaily_met()`, `save.betas()`, `save.model()`, `temporal.downscale.functions()`

sw2par	<i>SW to PAR</i>
--------	------------------

Description

Solar Radiation to PPFD

Usage

sw2par(sw)

Arguments

sw shortwave radiation (W/m2 == J/m2/s)

Details

Here the input is the total solar radiation so to obtain in the PAR spectrum need to multiply by 0.486 From Campbell and Norman p151 This is based on the approximation that PAR is 0.45-0.50 of the total radiation

Value

PAR W/m2

Author(s)

David LeBauer

sw2ppfd	<i>SW to PPFD</i>
---------	-------------------

Description

CF Shortwave to PPFD

Usage

sw2ppfd(sw)

Arguments

sw CF surface_downwelling_shortwave_flux_in_air (W/m2)

Details

Cambell and Norman 1998 p 151, ch 10

ValuePPFD $\mu\text{mol} / \text{m}^2 / \text{s}$ **Author(s)**

David LeBauer

```
temporal.downscale.functions
      temporal_downscale_functions
```

Description

This function contains the functions that do the heavy lifting in `gen.subdaily.models()` and `predict.subdaily.workflow()`. Individual variable functions actually generate the models and betas from the `dat.train_file` and save them in the output file. `save.model()` and `save.betas()` are helper functions that save the linear regression model output to a specific location. In the future, we should only save the data that we actually use from the linear regression model because this is a large file. `predict.met()` is called from `predict.subdaily.workflow()` and references the linear regression model output to predict the ensemble data.

Usage

```
temporal.downscale.functions(
  dat.train,
  n.beta,
  day.window,
  resids = FALSE,
  parallel = FALSE,
  n.cores = NULL,
  seed = format(Sys.time(), "%m%d"),
  outfolder,
  print.progress = FALSE,
  ...
)
```

Arguments

<code>dat.train</code>	- training data generated by <code>tdm_nc2dat.train.R</code>
<code>n.beta</code>	- number of betas to generate
<code>day.window</code>	- number of days surrounding current day we want to pull statistics from
<code>resids</code>	- whether or not to propagate residuals, set to <code>FALSE</code>
<code>parallel</code>	- whether or not to run in parallel. this is a feature still being worked on, set to <code>FALSE</code>
<code>n.cores</code>	- number of cores to use parallel processing on, set to <code>NULL</code>

seed - allows this to be reproducible
outfolder = where the output should be stored
print.progress - print progress of model generation?
... further arguments, currently ignored

Details

Temporal Downscale Functions Met variable functions that are called in `gen.subdaily.models` and `predict.subdaily.workflow`

Author(s)

Christy Rollinson, James Simkins

See Also

Other tdm - Temporally Downscale Meteorology: `gen.subdaily.models()`, `lm_ensemble_sims()`, `model.train()`, `nc.merge()`, `predict_subdaily_met()`, `save.betas()`, `save.model()`, `subdaily_pred()`

<code>temporal_downscale</code>	<i>Downscale NOAA GEFS from 6hr to 1hr</i>
---------------------------------	--

Description

Downscale NOAA GEFS from 6hr to 1hr

Usage

```
temporal_downscale(input_file, output_file, overwrite = TRUE, hr = 1)
```

Arguments

input_file full path to 6hr file
output_file full path to 1hr file that will be generated
overwrite logical stating to overwrite any existing output_file
hr time step in hours of temporal downscaling (default = 1)

Value

None

Author(s)

Quinn Thomas

upscale_met	<i>upscale_met</i>
-------------	--------------------

Description

upscale_met upscales the temporal resolution of a dataset

Usage

```
upscale_met(  
  outfolder,  
  input_met,  
  resolution = 1/24,  
  overwrite = FALSE,  
  verbose = FALSE,  
  ...  
)
```

Arguments

- outfolder path to directory where output should be saved Output is netcdf named as <input_met_filename>.upscaled.nc
- input_met path to netcdf file containing met dataset
- resolution desired output resolution, in days
- overwrite logical: replace output file if it already exists?
- verbose logical: should [ncdf4](#) functions print debugging information as they run?
- ... other arguments, currently ignored

Author(s)

James Simkins, Chris Black

wide2long	<i>Wide to Long</i>
-----------	---------------------

Description

Convert raster to lat, lon, var

Usage

```
wide2long(data.wide, lat, lon, var)
```

Arguments

<code>data.wide</code>	data
<code>lat</code>	latitude for rows
<code>lon</code>	longitude for columns
<code>var</code>	variable being measured

Value

data.frame with colnames (lat, lon, var)

Author(s)

David LeBauer

```
write_noaa_gefs_netcdf
```

Write NOAA GEFS netCDF

Description

Write NOAA GEFS netCDF

Usage

```
write_noaa_gefs_netcdf(
  df,
  ens = NA,
  lat,
  lon,
  cf_units,
  output_file,
  overwrite
)
```

Arguments

<code>df</code>	data frame of meteorological variables to be written to netcdf. Columns must start with time with the following columns in the order of ‘cf_units’
<code>ens</code>	ensemble index used for subsetting df
<code>lat</code>	latitude in degree north
<code>lon</code>	longitude in degree east
<code>cf_units</code>	vector of variable names in order they appear in df
<code>output_file</code>	name, with full path, of the netcdf file that is generated
<code>overwrite</code>	logical to overwrite existing netcdf file

Value

NA

Author(s)

Quinn Thomas

Index

- * **datasets**
 - narr_flx_vars, 89
 - pecan_standard_met_table, 92
- * **debias - Debias & Align**
 - Meteorology Datasets into continuous time series
 - align.met, 5
 - debias.met.regression, 15
- * **models**
 - lightME, 67
- * **tdm - Temporally Downscale**
 - Meteorology
 - gen.subdaily.models, 58
 - lm_ensemble_sims, 67
 - model.train, 88
 - nc.merge, 89
 - predict_subdaily_met, 94
 - save.betas, 101
 - save.model, 102
 - subdaily_pred, 108
 - temporal.downscale.functions, 110
- .download.raw.met.module
 - (*download.raw.met.module*), 41
- AirDens, 5
- align.met, 5, 17
- AmeriFlux_met_ensemble, 7
- Average_ERA5_2_GeoTIFF, 9
- build_cf_variables_table_url, 9, 62
- cfmet.downscale.daily, 10
- cfmet.downscale.subdaily, 11
- cfmet.downscale.time, 11
- check_met_input_file, 12
- check_unit, 13
- closest_xy, 13
- col2ncvar, 14
- cos_solar_zenith_angle, 14
- debias.met (*debias_met*), 18
- debias.met.regression, 7, 15
- debias_met, 18
- download.Ameriflux, 19
- download.AmerifluxLBL, 20
- download.CRUNCEP, 21
- download.ERA5_cds, 22
- download.FACE, 24
- download.Fluxnet2015, 25
- download.FluxnetLaThuile, 26
- download.Geostreams, 27, 63
- download.GFDL, 28
- download.GLDAS, 29
- download.ICOS, 30
- download.MACA, 31
- download.MERRA, 32
- download.MsTMIP_NARR, 33
- download.NARR, 34
- download.NARR_site, 35
- download.NEONmet, 36
- download.NLDAS, 37
- download.NOAA_GEFS, 38
- download.PaleON, 40
- download.PaleON_ENS, 41
- download.raw.met.module, 41
- download.US_WCr, 43
- download.US_Wlcf, 43
- download_grid, 44
- download_NOAA_GEFS_EFI, 45
- downscale_one_cfmet_day, 46
- downscale_repeat_6hr_to_half_hrly, 46
- downscale_repeat_6hr_to_hrly, 47
- downscale_ShortWave_to_half_hrly, 47
- downscale_ShortWave_to_hrly, 48
- downscale_solar_geom, 49
- downscale_solar_geom_halfhour, 49
- downscale_spline_to_half_hrly, 50
- downscale_spline_to_hrly, 50
- equation_of_time, 51
- ERA5_met_process, 51

- exner, 52
- extract.local.CMIP5, 53
- extract.local.NLDAS, 54
- extract.nc, 55
- extract.nc.ERA5, 56
- gen.subdaily.models, 58, 69, 88, 90, 96, 102, 103, 108, 111
- generate_narr_url, 59
- get.es (*sat_vapor_pressure*), 100
- get.lv, 60
- get.ncvector, 60
- get.rh, 61
- get.vpd, 61
- get_cf_variables_table, 62
- get_clowderauth, 63
- get_NARR_thredds, 63
- get_narr_url, 64
- half_hour_downscale, 65
- latlon2lcc, 65
- latlon2narr, 66
- lightME, 67
- list, 67
- lm_ensemble_sims, 59, 67, 88, 90, 96, 102, 103, 108, 111
- load.cfmet, 11, 69
- merge_met_variable, 70
- met.process, 71
- met.process.stage, 72
- met2CF.ALMA, 74
- met2CF.Ameriflux, 75
- met2CF.AmerifluxLBL, 76
- met2CF.csv, 77
- met2CF.ERA5, 79
- met2CF.FACE, 80
- met2CF.Geostreams, 81
- met2CF.ICOS, 82
- met2CF.NARR, 83
- met2CF.PalEON, 84
- met2CF.PalEONregional, 85
- met_temporal_downscale.Gaussian_ensemble, 73
- metgapfill, 86
- metgapfill.NOAA_GEFS, 87
- model.train, 59, 69, 88, 90, 96, 102, 103, 108, 111
- narr_all_vars (*narr_flx_vars*), 89
- narr_flx_vars, 89
- narr_sfc_vars (*narr_flx_vars*), 89
- nc.merge, 59, 69, 88, 89, 96, 102, 103, 108, 111
- nc_create, 22, 29, 33, 55, 81
- ncdf4, 17, 18, 70, 73, 90, 95, 106, 112
- ncvar_def, 22, 29, 33, 55
- noaa_grid_download, 90
- noaa_stage2, 91
- par2ppfd, 92
- pecan_standard_met_table, 92
- permute.nc, 93
- post_process, 94
- predict_subdaily_met, 59, 69, 88, 90, 94, 102, 103, 108, 111
- prepare_narr_year, 96
- process_gridded_noaa_download, 97
- qair2rh, 98
- read.register, 98
- read_narr_var, 99
- rh2qair, 99
- rh2rv (*rh2qair*), 99
- sat_vapor_pressure, 100
- save.betas, 59, 69, 88, 90, 96, 101, 103, 108, 111
- save.model, 59, 69, 88, 90, 96, 102, 102, 108, 111
- site.lst, 103
- site_from_tag, 103
- solarMJ2ppfd, 104
- spin.met, 105
- split_wind, 106
- sprintf, 9, 10
- step_means, 107
- subdaily_pred, 59, 69, 88, 90, 96, 102, 103, 108, 111
- sw2par, 109
- sw2ppfd, 109
- t2es (*sat_vapor_pressure*), 100
- temporal_downscale.functions, 59, 69, 88, 90, 96, 102, 103, 108, 110
- temporal_downscale, 111
- temporal_downscale_half_hour (*half_hour_downscale*), 65

`upscale_met`, [112](#)

`wide2long`, [112](#)

`write_noaa_gefs_netcdf`, [113](#)