Package ‘pastclim’

January 6, 2023

Type Package

Title Manipulate Time Series of Palaeoclimate Reconstructions

Version 1.2.3

Maintainer Andrea Manica <am315@cam.ac.uk>

Description Methods to easily extract and manipulate palaeoclimate
reconstructions for ecological and anthropological analyses, as described
in Leonardi et al. (2022) <doi:10.1101/2022.05.18.492456>.

License CC BY 4.0

Language en-GB

URL https://github.com/EvolEcolGroup/pastclim,
https://evolecolgroup.github.io/pastclim/

BugReports https://github.com/EvolEcolGroup/pastclim/issues

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.2.2

Depends R (>= 4.0.0), terra (>= 1.6.41)

Imports curl, ncdf4, utils

Suggests rmarkdown, knitr, sf, ggplot2, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Michela Leonardi [aut],
Emily Y. Hallet [ctb],
Robert Beyer [ctb],
Mario Krapp [ctb],
Andrea Manica [aut, cre]

Repository CRAN

Date/Publication 2023-01-06 11:10:05 UTC

R topics documented:

`Beyer2020` *Documentation for the Beyer2020 dataset*

Description

This dataset covers the last 120k years, at intervals of 1/2 k years, and a resolution of 0.5 degrees in latitude and longitude.

Details

If you use this dataset, make sure to cite the original publication:

Beyer, R.M., Krapp, M. & Manica, A. High-resolution terrestrial climate, bioclimate and vegetation for the last 120,000 years. Sci Data 7, 236 (2020). doi:doi.org/10.1038/s4159702005521

The version included in 'pastclim' has the ice sheets masked, as well as internal seas (Black and Caspian Sea) removed. The latter are based on:

`https://www.marineregions.org/gazetteer.php?p=details&id=4278`

`https://www.marineregions.org/gazetteer.php?p=details&id=4282`

As there is no reconstruction of their depth through time, modern outlines were used for all time steps.

Also, for bio15, the coefficient of variation was computed after adding one to monthly estimates, and it was multiplied by 100 following `https://pubs.usgs.gov/ds/691/ds691.pdf`

Changelog

v1.1.0 Added monthly variables. Files can be downloaded from: `https://zenodo.org/deposit/7062281`

v1.0.0 Remove ice sheets and internal seas, and use correct formula for bio15. Files can be downloaded from: doi:doi.org/10.6084/m9.figshare.19723405.v1

`clean_data_path` *Clean the data path*

Description

This function deletes old reconstructions that have been superseded in the data_path. It assumes that the only files in data_path are part of pastclim (i.e. there are no custom datasets stored in that directory).

Usage

```
clean_data_path(ask = TRUE)
```

Arguments

`ask`	boolean on whether the user should be asked before deleting

Value

TRUE if files are deleted successfully

`climate_for_locations` *Extract local climate for one or more locations for a given time slice.*

Description

Deprecated version of `location_slice`

Usage

```
climate_for_locations(...)
```

Arguments

`...`	arguments to be passed to `location_slice`

Value

a data.frame with the climatic variables of interest

`climate_for_time_slice` *Extract a climate slice for a region*

Description

Deprecated version of `region_slice`

Usage

```
climate_for_time_slice(...)
```

Arguments

`...`	arguments to be passed to `region_slice`

Value

a SpatRaster `terra::SpatRaster` object, with each variable as a layer.

`df_from_region_series` *Extract data frame from a region series*

Description

Extract the climatic information from a region series and organise them as a data frame.

Usage

```
df_from_region_series(x, xy = TRUE)
```

Arguments

`x`	climate time series generated with `region_series`
`xy`	a boolean whether x and y coordinates should be added to the dataframe (default to TRUE)

Details

To extract a data frame from a region slice, see `df_from_region_slice`.

Value

a data.frame where each cell each raster layer (i.e. timestep) is a row, and the available variables are columns.

`df_from_region_slice` *Extract data frame from a region slice*

Description

Extract the climatic information from a region slice and organise it as a data frame. This is just a wrapper around `terra::as.data.frame`.

Usage

```
df_from_region_slice(x, xy = TRUE)
```

Arguments

`x`	climate time slice (i.e. a `terra::SpatRaster`) generated with `region_slice`
`xy`	a boolean whether x and y coordinates should be added to the dataframe (default to TRUE)

Details

To extract a data frame from a region series, see `df_from_region_series`.

Value

a data.frame where each cell the raster is a row, and the available variables are columns.

`download_dataset` *Download palaeoclimate reconstructions.*

Description

This function downloads palaeoclimate reconstructions. Files will be stored in the data path of ‘pastclim‘, which can be inspected with `get_data_path` and changed with `set_data_path`

Usage

```
download_dataset(dataset, bio_variables = NULL)
```

Arguments

`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). This function will not work on custom datasets.
`bio_variables`	one or more variable names to be downloaded. If left to NULL, all variables available for this dataset will be downloaded

Value

TRUE if the dataset(s) was downloaded correctly.

`Example` *Documentation for the Example dataset*

Description

This dataset is a subset of Beyer2020, used for the vignette of pastclim. Do not use this dataset for any real work, as it might not reflect the most up-to-date version of Beyer2020.

get_available_datasets *Get the available datasets.*

Description

List the datasets available in pastclim. Most functions can also be used on custom datasets by setting ‘dataset="custom"‘

Usage

```
get_available_datasets()
```

Value

a character vector of the available datasets

get_biome_classes *Get the biome classes for a dataset.*

Description

Get a full list of biomes and how their id as coded in the biome variable for a given dataset.

Usage

```
get_biome_classes(dataset)
```

Arguments

`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). This function will not work on custom datasets.

Value

a data.frame with columns id and category.

`get_data_path` *Get the data path where climate reconstructions are stored*

Description

This function returns the path where climate reconstructions are stored.

Usage

```
get_data_path(silent = FALSE)
```

Arguments

`silent`	boolean on whether a message is returned when data_path is not set (i.e. equal to NULL)

Details

The path is stored in an option for ‘pastclim‘ named ‘data_path‘. If a configuration file was saved when using `set_data_path`, the path is retrieved from a file named "pastclim_data.txt", which is found in the directory returned by ‘tools::R_user_dir("pastclim","config")‘ (i.e. the default configuration directory for the package as set in R >= 4.0).

Value

the data path

`get_downloaded_datasets` *Get the variables downloaded for each dataset.*

Description

List the downloaded variable for each dataset.

Usage

```
get_downloaded_datasets(data_path = NULL)
```

Arguments

`data_path`	leave it to NULL to use the default data_path

Value

a list of variable names per dataset.

get_file_for_dataset *Get the file details for a variable and dataset.*

Description

Internal getter function

Usage

```
get_file_for_dataset(variable, dataset)
```

Arguments

`variable`	one or more variable names to be downloaded
`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). This function will not work on custom datasets.

Value

the filename for taht variable and dataset

get_ice_mask *Get the ice mask for a dataset.*

Description

Get the ice mask for a dataset at a given time point.

Usage

```
get_ice_mask(time_bp, dataset)
```

Arguments

`time_bp`	time slice in years before present (negative)
`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). This function will not work on custom datasets.

Value

a binary `terra::SpatRaster` with the ice mask as 1s

`get_land_mask` *Get the land mask for a dataset.*

Description

Get the land mask for a dataset at a given time point.

Usage

```
get_land_mask(time_bp, dataset)
```

Arguments

`time_bp`	time slice in years before present (negative)
`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). This function will not work on custom datasets.

Value

a binary `terra::SpatRaster` with the land mask as 1s

`get_mis_time_steps` *Get time steps for a given MIS*

Description

Get the time steps available in a given dataset for a MIS.

Usage

```
get_mis_time_steps(mis, dataset, path_to_nc = NULL)
```

Arguments

`mis`	string giving the mis; it must use the same spelling as used in /codemis_boundaries
`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). If set to "custom", then a single nc file is used from "path_to_nc"
`path_to_nc`	the path to the custom nc file containing the palaeoclimate reconstructions. All the variables of interest need to be included in this file.

Value

a vector of time steps

`get_time_steps` *Get time steps for a given dataset*

Description

Get the time steps (in time_bp) available in a given dataset.

Usage

```
get_time_steps(dataset, path_to_nc = NULL)
```

Arguments

`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). If set to "custom", then a single nc file is used from "path_to_nc"
`path_to_nc`	the path to the custom nc file containing the palaeoclimate reconstructions. All the variables of interest need to be included in this file.

Value

a vector of time steps (in time_bp)

`get_varname` *Get a the varname for this variable*

Description

Internal function to get the varname for this variable

Usage

```
get_varname(variable, dataset)
```

Arguments

`variable`	string defining the variable name
`dataset`	string defining dataset to be downloaded

Value

the name of the variable

get_vars_for_dataset *Get a list of variables for a given dataset.*

Description

This function lists the variables available for a given dataset. Note that the spelling and use of capitals in names might differ from the original publications, as ‘pastclim‘ harmonises the names of variables across different reconstructions.

Usage

```
get_vars_for_dataset(dataset, path_to_nc = NULL, details = FALSE)
```

Arguments

`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`).
`path_to_nc`	the path to the custom nc file containing the palaeoclimate reconstructions.
`details`	boolean determining whether the output should include information including long names of variables and their units

Value

a vector of variable names

is_region_series *Check the object is a valid region series*

Description

A region series is a `terra::SpatRasterDataset` for which each sub-dataset is a variable, and all variables have the same number of time steps.

Usage

```
is_region_series(x, strict = FALSE)
```

Arguments

`x`	a `terra::SpatRasterDataset` representing a time series of regional reconstructions obtained from `region_series`.
`strict`	a boolean defining whether to preform a thorough test (see description above for details).

Details

The standard test only checks that each SpatRaster has the same number of layers. The more thorough test (obtained with strict=TRUE) actually checks that all time steps are identical by comparing the result of `terra::time` applied to each variable

Value

TRUE if the object is a region series

`Krapp2021` *Documentation for the Krapp2021 dataset*

Description

This dataset covers the last 800k years, at intervals of 1k years, and a resolution of 0.5 degrees in latitude and longitude.

Details

If you use this dataset, make sure to cite the original publication:

Krapp, M., Beyer, R.M., Edmundson, S.L. et al. A statistics-based reconstruction of high-resolution global terrestrial climate for the last 800,000 years. Sci Data 8, 228 (2021). doi:doi.org/10.1038/s41597021010093

The version included in 'pastclim' has the ice sheets masked.

Note that, for bio15, we use the corrected version, which follows `https://pubs.usgs.gov/ds/691/ds691.pdf`

Changelog

v1.1.0 Added monthly variables. Files can be downloaded from: `https://zenodo.org/record/7065055`

v1.0.0 Remove ice sheets and use correct formula for bio15. Files can be downloaded from: doi:doi.org/10.6084/m9.figshare.19733680.v1

`location_series` *Extract a time series of bioclimatic variables for one or more locations.*

Description

This function extract a time series of local climate for a set of locations. Note that this function does not apply any interpolation (as opposed to `location_slice`). If you have a coastal location that just falls into the water for the reconstructions, you will have to amend the coordinates to put it more firmly on land.

Usage

```
location_series(
  x,
  time_bp = NULL,
  bio_variables,
  dataset,
  path_to_nc = NULL,
  nn_interpol = FALSE,
  buffer = FALSE,
  directions = 8
)
```

Arguments

`x` a data.frame with columns ‘longitude‘, ranging -180 to 180, and ‘latitude‘, from -90 to 90 (and an optional ‘name‘), or a vector of cell numbers.

`time_bp` time slices in years before present (negative values represent time before present, positive values time in the future). This parameter can be a vector of times (the slices need to exist in the dataset), a list with a min and max element setting the range of values, or left to NULL to retrieve all time steps. To check which slices are available, you can use `get_time_steps`.

`bio_variables` vector of names of variables to be extracted.

`dataset` string defining the dataset to use. If set to "custom", then a single nc file is used from "path_to_nc"

`path_to_nc` the path to the custom nc file containing the palaeoclimate reconstructions. All the variables of interest need to be included in this file.

`nn_interpol` boolean determining whether nearest neighbour interpolation is used to estimate climate for cells that lack such information (i.e. they are under water or ice). By default, interpolation is only performed from the first ring of nearest neighbours; if climate is not available, NA will be returned for that location. The number of neighbours can be changed with the argument ‘directions‘. ‘nn_interpol‘ defaults to FALSE (this is DIFFERENT from `location_slice`.

`buffer` boolean determining whether the variable will be returned as the mean of a buffer around the focal cell. If set to TRUE, it overrides ‘nn_interpol‘ (which provides the same estimates as ‘buffer‘ but only for locations that are in cells with an NA). The buffer size is determined by the argument ‘directions‘. ‘buffer‘ defaults to FALSE.

`directions` character or matrix to indicate the directions in which cells are considered connected when using ‘nn_interpol‘ or ‘buffer‘. The following character values are allowed: "rook" or "4" for the horizontal and vertical neighbors; "bishop" to get the diagonal neighbors; "queen" or "8" to get the vertical, horizontal and diagonal neighbors; or "16" for knight and one-cell queen move neighbors. If directions is a matrix it should have odd dimensions and have logical (or 0, 1) values.

Value

a data.frame with the climatic variables of interest

location_series_old — *Extract a time series of bioclimatic variables for one or more locations.*

Description

This function extract a time series of local climate for a set of locations. Note that this function does not apply any interpolation (as opposed to `location_slice`). If you have a coastal location that just falls into the water for the reconstructions, you will have to amend the coordinates to put it more firmly on land.

Usage

```
location_series_old(
  x,
  time_bp = NULL,
  bio_variables,
  dataset,
  path_to_nc = NULL
)
```

Arguments

`x`	a data.frame with columns ‘longitude‘, ranging -180 to 180, and ‘latitude‘, from -90 to 90 (and an optional ‘name‘), or a vector of cell numbers.
`time_bp`	time slices in years before present (negative values represent time before present, positive values time in the future). This parameter can be a vector of times (the slices need to exist in the dataset), a list with a min and max element setting the range of values, or left to NULL to retrieve all time steps. To check which slices are available, you can use `get_time_steps`.
`bio_variables`	vector of names of variables to be extracted.
`dataset`	string defining the dataset to use. If set to "custom", then a single nc file is used from "path_to_nc"
`path_to_nc`	the path to the custom nc file containing the palaeoclimate reconstructions. All the variables of interest need to be included in this file.

Value

a data.frame with the climatic variables of interest

location_slice — *Extract local climate for one or more locations for a given time slice.*

Description

This function extract local climate for a set of locations at the appropriate times (selecting the closest time slice available for the specific date associated with each location).

Usage

```
location_slice(
  x,
  time_bp = NULL,
  bio_variables,
  dataset,
  path_to_nc = NULL,
  nn_interpol = TRUE,
  buffer = FALSE,
  directions = 8
)
```

Arguments

x	a data.frame with columns ‘longitude‘, ranging -180 to 180, and ‘latitude‘, from -90 to 90, plus optional columns ‘time_bp‘ and ‘name‘. Alternatively, a vector of cell numbers.
time_bp	used if no ‘time_bp‘ column is present in ‘x‘: the dates in years before present (negative values represent time before present, i.e. 1950, positive values time in the future) for each location.
bio_variables	vector of names of variables to be extracted.
dataset	string defining the dataset to use. If set to "custom", then a single nc file is used from "path_to_nc"
path_to_nc	the path to the custom nc file containing the palaeoclimate reconstructions. All the variables of interest need to be included in this file.
nn_interpol	boolean determining whether nearest neighbour interpolation is used to estimate climate for cells that lack such information (i.e. they are under water or ice). By default, interpolation is only performed from the first ring of nearest neighbours; if climate is not available, NA will be returned for that location. The number of neighbours can be changed with the argument ‘directions‘. ‘nn_interpol‘ defaults to TRUE.
buffer	boolean determining whether the variable will be returned as the mean of a buffer around the focal cell. If set to TRUE, it overrides ‘nn_interpol‘ (which provides the same estimates as ‘buffer‘ but only for locations that are in cells with an NA). The buffer size is determined by the argument ‘directions‘. ‘buffer‘ defaults to FALSE.

`directions`	character or matrix to indicate the directions in which cells are considered connected when using ‘nn_interpol‘ or ‘buffer‘. The following character values are allowed: "rook" or "4" for the horizontal and vertical neighbors; "bishop" to get the diagonal neighbors; "queen" or "8" to get the vertical, horizontal and diagonal neighbors; or "16" for knight and one-cell queen move neighbors. If directions is a matrix it should have odd dimensions and have logical (or 0, 1) values.

Value

a data.frame with the climatic variables of interest.

`mis_boundaries` *Time boundaries of marine isotope stages (MIS).*

Description

A dataset containing the beginning and end of MIS.

Usage

```
mis_boundaries
```

Format

A data frame with 24 rows and 2 variables:

mis the stage, a string

start the start of a given MIS, in kya

end the start of a given MIS, in kya

`region_extent` *Region extents.*

Description

A list of extents for major regions.

Usage

```
region_extent
```

Format

A list of vectors giving the extents.

region_outline *Region outlines.*

Description

An sf object containing outlines for major regions. Outlines that span the antimeridian have been split into multiple polygons.

Usage

```
region_outline
```

Format

sf of outlines.

name names of regions

region_outline_union *Region outlines unioned.*

Description

An sf object containing outlines for major regions. Each outline is represented as a single polygon. If you want multiple polygons, use region_outline.

Usage

```
region_outline_union
```

Format

sf of outlines.

name names of regions

region_series *Extract a time series of climate variables for a region*

Description

This function extracts a time series of one or more climate variables for a given dataset covering a region (or the whole world). The function returns a SpatRasterDataset `terra::sds` object, with each variable as a sub-dataset.

Usage

```
region_series(
  time_bp = NULL,
  bio_variables,
  dataset,
  path_to_nc = NULL,
  ext = NULL,
  crop = NULL
)
```

Arguments

`time_bp` time slices in years before present (negative values represent time before present, positive values time in the future). This parameter can be a vector of times (the slices need to exist in the dataset), a list with a min and max element setting the range of values, or left to NULL to retrieve all time steps. To check which slices are available, you can use `get_time_steps`.

`bio_variables` vector of names of variables to be extracted

`dataset` string defining the dataset to use. If set to "custom", then a single nc file is used from "path_to_nc"

`path_to_nc` the path to the custom nc file containing the palaeoclimate reconstructions. All the variables of interest need to be included in this file.

`ext` an extent, coded as numeric vector (length=4; order= xmin, xmax, ymin, ymax) or a `terra::SpatExtent` object. If NULL, the full extent of the reconstruction is given.

`crop` a polygon used to crop the reconstructions (e.g. the outline of a continental mass). A `sf:sfg` or a `terra::SpatVector` object is used to define the polygon.

Value

a SpatRasterDataset `terra::sds` object, with each variable as a sub-dataset.

`region_slice` *Extract a climate slice for a region*

Description

This function extracts a slice of one or more climate variables for a given dataset covering a region (or the whole world). The function returns a SpatRaster `terra::SpatRaster` object, with each variable as a layer.

Usage

```
region_slice(
  time_bp,
  bio_variables,
  dataset,
  path_to_nc = NULL,
  ext = NULL,
  crop = NULL
)
```

Arguments

`time_bp`	the time slice in years before present (negative values represent time before present, positive values time in the future). The slice needs to exist in the dataset. To check which slices are available, you can use `get_time_steps`.
`bio_variables`	vector of names of variables to be extracted
`dataset`	string defining the dataset to use. If set to "custom", then a single nc file is used from "path_to_nc"
`path_to_nc`	the path to the custom nc file containing the palaeoclimate reconstructions. All the variables of interest need to be included in this file.
`ext`	an extent, coded as numeric vector (length=4; order= xmin, xmax, ymin, ymax) or a `terra::SpatExtent` object. If NULL, the full extent of the reconstruction is given.
`crop`	a polygon used to crop the reconstructions (e.g. the outline of a continental mass). A `sf:sfg` or a `terra::SpatVector` object is used to define the polygon.

Value

a SpatRaster `terra::SpatRaster` object, with each variable as a layer.

sample_region_series *Sample points from a region time series*

Description

This function samples points from a region time series. Sampling can either be performed for the same locations at all time steps (if only one value is given for ‘size‘), or for different locations for each time step (if ‘size‘ is a vector of length equal to the number of time steps). To sample the same number of points, but different locations, for each time step, provide a vector repeating the same value for each time step.

Usage

```
sample_region_series(x, size, method = "random", replace = FALSE, na.rm = TRUE)
```

Arguments

`x`	a `terra::SpatRasterDataset` returned by `region_series`
`size`	number of points sampled. A single value is used to sample the same locations across all time steps, a vector of values to sample different locations at each time step.
`method`	one of the sampling methods from `terra::spatSample`. It defaults to "random"
`replace`	boolean determining whether we sample with replacement
`na.rm`	boolean determining whether NAs are removed

Details

This function wraps `terra::spatSample` to appropriate sample the `terra::SpatRasters` in the `terra::SpatRasterDataset` returned by `region_series`.

Value

a data.frame with the sampled cells and their respective values for the climate variables.

sample_region_slice *Sample points from a region time slice*

Description

This function samples points from a region time slice (i.e. a time point).

Usage

```
sample_region_slice(x, size, method = "random", replace = FALSE, na.rm = TRUE)
```

Arguments

`x`	a `terra::SpatRaster` returned by `region_slice`
`size`	number of points sampled.
`method`	one of the sampling methods from `terra::spatSample`. It defaults to "random"
`replace`	boolean determining whether we sample with replacement
`na.rm`	boolean determining whether NAs are removed

Details

This function wraps `terra::spatSample` to appropriate sample the `terra::SpatRaster` returned by `region_slice`. You can also use `terra::spatSample` directly on a slice (which is a standard `terra::SpatRaster`).

Value

a data.frame with the sampled cells and their respective values for the climate variables.

`set_data_path` *Set the data path where climate reconstructions will be stored*

Description

This function sets the path where climate reconstructions will be stored. This information is stored in a file names "pastclim_data.txt", which is found in the directory returned by ‘tools::R_user_dir("pastclim","config")‘ (i.e. the default configuration directory for the package as set in R >= 4.0).

Usage

```
set_data_path(
  path_to_nc = NULL,
  ask = TRUE,
  write_config = TRUE,
  copy_example = TRUE
)
```

Arguments

`path_to_nc`	the path to the file that contains the downloaded reconstructions. If left unset, the default location returned by ‘tools::R_user_dir("pastclim","data")‘ will be used
`ask`	boolean on whether the user should be asked to confirm their choices
`write_config`	boolean on whether the path should be saved in a config file
`copy_example`	boolean on whether the example dataset should be saved in the data_path

Value

TRUE if the path was set correctly

`set_data_path_for_CRAN` *Set the data path for examples on CRAN*

Description

Users should NOT need this function. It is used to set up a data path in the temporary directory for examples and tests to run on CRAN.

Usage

```
set_data_path_for_CRAN()
```

Value

None

`slice_region_series` *Extract a slice for a time series of climate variables for a region*

Description

This function extracts a time slice from time series of one or more climate variables for a given dataset covering a region (or the whole world).

Usage

```
slice_region_series(x, time_bp)
```

Arguments

`x`	climate time series generated with `region_series`
`time_bp`	time slices in years before present (i.e. 1950, negative integers for values in the past). The slices need to exist in the dataset. To check which slices are available, you can use `time_bp(x[[1]])` (note that you have to use the ‘time‘ function on the first element of the ’sds’ object, i.e. on one of the ’SpatRaster’ objects)

Value

a `SpatRaster` of the relevant slice.

time_bp *Extract time in years before present from SpatRaster*

Description

A wrapper around `terra::time`, which converts time into years before present

Usage

```
time_bp(x)
```

Arguments

`x`	a `terra::SpatRaster`

Value

a date in years BP (where negative numbers indicate a date in the past)

time_series_for_locations *Extract a time series of bioclimatic variables for one or more locations.*

Description

Deprecated version of `location_series`

Deprecated version of `location_series`

Usage

```
time_series_for_locations(...)

time_series_for_locations(...)
```

Arguments

`...`	arguments to be passed to `series`

Value

a data.frame with the climatic variables of interest

a data.frame with the climatic variables of interest

update_dataset_list *Update the dataset list*

Description

If a newer dataset list (which includes all the information about the files storing the data for pastclim), download it and start using it as 'dataset_list_included.csv' in 'tools::R_user_dir("pastclim","config")'. If the latter is present, the last column, named 'dataset_list_v', provides the version of this table, and the most advanced table is used.

Usage

```
update_dataset_list(on_cran = FALSE)
```

Arguments

on_cran	boolean to make this function run on ci tests using tempdir

Value

TRUE if the dataset was updated

validate_nc *Validate an netcdf file for pastclim*

Description

This function validates a netcdf file as a potential dataset for 'pastlim'. The key checks are: a) that the dimensions (longitude, latitude and time) have been set correctly. b) that all variables have the appropriate metadata (longname and units)

Usage

```
validate_nc(path_to_nc)
```

Arguments

path_to_nc	path to the nc file of interest

Value

TRUE if the file is valid.

var_labels *Generate pretty variable labels for plotting*

Description

Generate pretty labels (in the form of an `expression`) that can be used for plotting

Usage

```
var_labels(x, dataset, with_units = TRUE, abbreviated = FALSE)
```

Arguments

`x`	either a character vector with the names of the variables, or a `SpatRaster` generated with `region_slice`
`dataset`	string defining dataset to be downloaded (a list of possible values can be obtained with `get_available_datasets`). This function will not work on custom datasets.
`with_units`	boolean defining whether the label should include units
`abbreviated`	boolean defining whether the label should use abbreviations for the variable

Value

a `expression` that can be used as a label in plots

Examples

```
var_labels("bio01", dataset = "Example")

# set the data_path for this example to run on CRAN
# users don't need to run this line
set_data_path_for_CRAN()

# for a SpatRaster
climate_20k <- region_slice(
time_bp = -20000,
bio_variables = c("bio01", "bio10", "bio12"),
dataset = "Example"
)
terra::plot(climate_20k, main = var_labels(climate_20k, dataset = "Example"))
terra::plot(climate_20k, main = var_labels(climate_20k, dataset = "Example",
                   abbreviated = TRUE))
```

Index