Package 'epwshiftr'

July 7, 2020

Title Create Future 'EnergyPlus' Weather Files using 'CMIP6' Data

Version 0.1.0

Description Query, download climate change projection data from the 'CMIP6' (Coupled Model Intercomparison Project Phase 6) project <https://pcmdi.llnl.gov/CMIP6> in the 'ESGF' (Earth System Grid Federation) platform <https://esgf.llnl.gov>, and create future 'EnergyPlus' <https://energyplus.net> Weather ('EPW') files adjusted from climate changes using data from Global Climate Models ('GCM').

Imports checkmate, data.table (>= 1.12.4), eplusr, future.apply, fst, jsonlite, progressr, psychrolib, rappdirs, RNetCDF, units

Suggests testthat, pingr, covr

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/ideas-lab-nus/epwshiftr

BugReports https://github.com/ideas-lab-nus/epwshiftr/issues

RoxygenNote 7.0.2

Collate 'coord.R' 'utils.R' 'epwshiftr-package.R' 'esgf.R' 'morph.R' 'netcdf.R'

NeedsCompilation no

Author Hongyuan Jia [aut, cre] (<https://orcid.org/0000-0002-0075-8183>)

Maintainer Hongyuan Jia <hongyuan.jia@bears-berkeley.sg>

Repository CRAN

Date/Publication 2020-07-07 14:00:07 UTC

R topics documented:

epwshiftr-package 2
esgf_query 3
extract_data 7

future_epw 9
get_data_dir 10
get_data_node 11
init_cmip6_index 11
load_cmip6_index 15
match_coord 16
morphing_epw 17
set_cmip6_index 18
summary_database 19

Index **22**

`epwshiftr-package` *epwshiftr: Create future EnergyPlus Weather files using CMIP6 data*

Description

Query, download climate change projection data from the CMIP6 (Coupled Model Intercomparison Project Phase 6) project in the ESGF (Earth System Grid Federation) platform, and create future EnergyPlus Weather (EPW) files adjusted from climate changes using data from Global Climate Models (GCM).

Package options

- `epwshiftr.verbose`: If `TRUE`, more detailed message will be printed. Default: `FALSE`.
- `epwshiftr.dir`: The directory to store package data, including CMIP6 model output file index and etc. If not set, the current user data directory will be used.

Author(s)

Hongyuan Jia

See Also

Useful links:

- https://github.com/ideas-lab-nus/epwshiftr
- Report bugs at https://github.com/ideas-lab-nus/epwshiftr/issues

esgf_query *Query CMIP6 data using ESGF search RESTful API*

Description

Query CMIP6 data using ESGF search RESTful API

Usage

```
esgf_query(
  activity = "ScenarioMIP",
  variable = c("tas", "tasmax", "tasmin", "hurs", "hursmax", "hursmin", "pr", "rsds",
    "rlds", "psl", "sfcWind", "clt"),
  frequency = "day",
  experiment = c("ssp126", "ssp245", "ssp370", "ssp585"),
  source = c("AWI-CM-1-1-MR", "BCC-CSM2-MR", "CESM2", "CESM2-WACCM", "EC-Earth3",
    "EC-Earth3-Veg", "GFDL-ESM4", "INM-CM4-8", "INM-CM5-0", "MPI-ESM1-2-HR",
    "MRI-ESM2-0"),
  variant = "r1i1p1f1",
  replica = FALSE,
  latest = TRUE,
  resolution = c("100 km", "50 km"),
  type = "Dataset",
  limit = 10000L,
  data_node = NULL
)
```

Arguments

`activity` A character vector indicating activity identifiers. Default: `"ScenarioMIP"`. Possible values:

- `"AerChemMIP"`: Aerosols and Chemistry Model Intercomparison Project,
- `"C4MIP"`: Coupled Climate Carbon Cycle Model Intercomparison Project,
- `"CDRMIP"`: Carbon Dioxide Removal Model Intercomparison Project,
- `"CFMIP"`: Cloud Feedback Model Intercomparison Project,
- `"CMIP"`: CMIP DECK: 1pctCO2, abrupt4xCO2, amip, esm-piControl, esm-historical, historical, and piControl experiments,
- `"CORDEX"`: Coordinated Regional Climate Downscaling Experiment,
- `"DAMIP"`: Detection and Attribution Model Intercomparison Project,
- `"DCPP"`: Decadal Climate Prediction Project,
- `"DynVarMIP"`: Dynamics and Variability Model Intercomparison Project,
- `"FAFMIP"`: Flux-Anomaly-Forced Model Intercomparison Project,
- `"GMMIP"`: Global Monsoons Model Intercomparison Project,
- `"GeoMIP"`: Geoengineering Model Intercomparison Project,
- `"HighResMIP"`: High-Resolution Model Intercomparison Project,

- "ISMIP6": Ice Sheet Model Intercomparison Project for CMIP6,
- "LS3MIP": Land Surface, Snow and Soil Moisture,
- "LUMIP": Land-Use Model Intercomparison Project,
- "OMIP": Ocean Model Intercomparison Project,
- "PAMIP": Polar Amplification Model Intercomparison Project,
- "PMIP": Palaeoclimate Modelling Intercomparison Project,
- "RFMIP": Radiative Forcing Model Intercomparison Project,
- "SIMIP": Sea Ice Model Intercomparison Project,
- "ScenarioMIP": Scenario Model Intercomparison Project,
- "VIACSAB": Vulnerability, Impacts, Adaptation and Climate Services Advisory Board,
- "VolMIP": Volcanic Forcings Model Intercomparison Project

variable

A character vector indicating variable identifiers. The 12 most related variables for EPW are set as defaults. If NULL, all possible variables are returned. Default: c("tas","tasmax","tasmin","hurs","hursmax","hursmin","psl","rss","rls","sfcWind","pr
where:

- tas: Near-surface (usually, 2 meter) air temperature, units: K.
- tasmax: Maximum near-surface (usually, 2 meter) air temperature, units: K.
- tasmin: Minimum near-surface (usually, 2 meter) air temperature, units: K.
- hurs: Near-surface relative humidity, units: %.
- hursmax: Maximum near-surface relative humidity, units: %.
- hursmin: Minimum near-surface relative humidity, units: %.
- psl: Sea level pressure, units: Pa.
- rsds: Surface downwelling shortwave radiation, units: W m-2.
- rlds: Surface downwelling longware radiation, units: W m-2.
- sfcWind: Near-surface (usually, 10 meters) wind speed, units: m s-1.
- pr: Precipitation, units: kg m-2 s-1.
- clt: Total cloud area fraction for the whole atmospheric column, as seen from the surface or the top of the atmosphere. Units: %.

frequency

A character vector of sampling frequency. If NULL, all possible frequencies are returned. Default: "day". Possible values:

- "1hr": sampled hourly,
- "1hrCM": monthly-mean diurnal cycle resolving each day into 1-hour means,
- "1hrPt": sampled hourly, at specified time point within an hour,
- "3hr": sampled every 3 hours,
- "3hrPt": sampled 3 hourly, at specified time point within the time period,
- "6hr": sampled every 6 hours,
- "6hrPt": sampled 6 hourly, at specified time point within the time period,
- "day": daily mean samples,
- "dec": decadal mean samples,
- "fx": fixed (time invariant) field,

- "mon": monthly mean samples,
- "monC": monthly climatology computed from monthly mean samples,
- "monPt": sampled monthly, at specified time point within the time period,
- "subhrPt": sampled sub-hourly, at specified time point within an hour,
- "yr": annual mean samples,
- "yrPt": sampled yearly, at specified time point within the time period

`experiment` A character vector indicating root experiment identifiers. The **Tier-1** experiment of activity ScenarioMIP are set as defaults. If NULL, all possible experiment are returned. Default: c("ssp126","ssp245","ssp370","ssp585").

`source` A character vector indicating model identifiers. Defaults are set to 11 sources which give outputs of all 4 experiment of activity ScenarioMIP with daily frequency, i.e. "AWI-CM-1-1-MR", "BCC-CSM2-MR", "CESM2", "CESM2-WACCM", "EC-Earth3", "EC-Earth3-Veg", "GFDL-ESM4", "INM-CM4-8", "INM-CM5-0", "MPI-ESM1-2-HR" and "MRI-ESM2-0". If NULL, all possible sources are returned.

`variant` A character vector indicating label constructed from 4 indices stored as global attributes in format r<k>i<l>p<m>f<n> described below. Default: "r1i1p1f1". If NULL, all possible variants are returned.

- r: realization_index (<k>) = realization number (integer >0)
- i: initialization_index (<l>) = index for variant of initialization method (integer >0)
- p: physics_index (<m>) = index for model physics variant (integer >0)
- f: forcing_index (<n>) = index for variant of forcing (integer >0)

`replica` Whether the record is the "master" copy, or a replica. Use FALSE to return only originals and TRUE to return only replicas. Default: FALSE.

`latest` Whether the record is the latest available version, or a previous version. Use TRUE to return only the latest version of all records and FALSE to return previous versions. Default: FALSE.

`resolution` A character vector indicating approximate horizontal resolution. Default: c("50 km","100 km"). If NULL, all possible resolutions are returned.

`type` A single string indicating the intrinsic type of the record. Should be either "Dataset" or "File". Default: "Dataset".

`limit` An integer indicating the maximum of matched records to return. Should be <= 10,000. Default: 10000.

`data_node` A character vector indicating data nodes to be queried. Default to NULL, which means all possible data nodes.

Details

The Earth System Grid Federation (ESGF) is an international collaboration for the software that powers most global climate change research, notably assessments by the Intergovernmental Panel on Climate Change (IPCC).

The ESGF search service exposes a RESTful URL that can be used by clients to query the contents of the underlying search index, and return results matching the given constraints. ributed capabilities

of the ESGF search, the URL at any Index Node can be used to query that Node only, or all Nodes in the ESGF system. `esgf_query()` uses the LLNL (Lawrence Livermore National Laboratory) Index Node.

Value

A data.table::data.table with an attribute named `response` which is a list converted from json response. If no matched data is found, an empty data.table is returned. Otherwise, the columns of returned data varies based on the `type`:

- If `"Dataset"`, returned columns are:

No.	Column	Type	Description
1	`dataset_id`	Character	Dataset universal identifier
2	`mip_era`	Character	Activity's associated CMIP cycle. Will always be `"CMIP6"`
3	`activity_drs`	Character	Activity DRS (Data Reference Syntax)
4	`institution_id`	Character	Institution identifier
5	`source_id`	Character	Model identifier
6	`experiment_id`	Character	Root experiment identifier
7	`member_id`	Character	A compound construction from `sub_experiment_id` and `variant_label`
8	`table_id`	Character	Table identifier, i.e. sampling frequency identifier
9	`frequency`	Character	Sampling frequency
10	`grid_label`	Character	Grid identifier
11	`version`	Character	Approximate date of model output file
12	`nominal_resolution`	Character	Approximate horizontal resolution
13	`variable_id`	Character	Variable identifier
14	`variable_long_name`	Character	Variable long name
15	`variable_units`	Character	Units of variable
16	`data_node`	Character	Data node to download the model output file
17	`dataset_pid`	Character	A unique string that helps identify the dataset

- If `"File"`, returned columns are:

No.	Column	Type	Description
1	`file_id`	Character	Model output file universal identifier
2	`dataset_id`	Character	Dataset universal identifier
3	`mip_era`	Character	Activity's associated CMIP cycle. Will always be `"CMIP6"`
4	`activity_drs`	Character	Activity DRS (Data Reference Syntax)
5	`institution_id`	Character	Institution identifier
6	`source_id`	Character	Model identifier
7	`experiment_id`	Character	Root experiment identifier
8	`member_id`	Character	A compound construction from `sub_experiment_id` and `variant_label`
9	`table_id`	Character	Table identifier, i.e. sampling frequency identifier
10	`frequency`	Character	Sampling frequency
11	`grid_label`	Character	Grid identifier
12	`version`	Character	Approximate date of model output file
13	`nominal_resolution`	Character	Approximate horizontal resolution
14	`variable_id`	Character	Variable identifier

15	variable_long_name	Character	Variable long name
16	variable_units	Character	Units of variable
17	datetime_start	POSIXct	Start date and time of simulation
18	datetime_end	POSIXct	End date and time of simulation
19	file_size	Character	Model output file size in Bytes
20	data_node	Character	Data node to download the model output file
21	file_url	Character	Model output file download url from HTTP server
22	tracking_id	Character	A unique string that helps identify the output file

References

https://github.com/ESGF/esgf.github.io/wiki/ESGF_Search_REST_API

Examples

```
## Not run:
esgf_query(variable = "rss", experiment = "ssp126", resolution = "100 km", limit = 1)

esgf_query(variable = "rss", experiment = "ssp126", type = "File", limit = 1)

## End(Not run)
```

`extract_data` *Extract data*

Description

`extract_data()` takes an `epw_cmip6_coord` object generated using `match_coord()` and extracts CMIP6 data using the coordinates and years of interest specified.

Usage

```
extract_data(
  coord,
  years = NULL,
  unit = FALSE,
  out_dir = NULL,
  by = NULL,
  keep = is.null(out_dir),
  compress = 100
)
```

Arguments

coord	An epw_cmip6_coord object created using match_coord()
years	An integer vector indicating the target years to be included in the data file. All other years will be excluded. If NULL, no subsetting on years will be performed. Default: NULL.
unit	If TRUE, units will be added to values using units::set_units().
out_dir	The directory to save extracted data using fst::write_fst(). If NULL, all data will be kept in memory by default. Default: NULL.
by	A character vector of variable names used to split data during extraction. Should be a subset of: • "experiment": root experiment identifiers • "source": model identifiers • "variable": variable identifiers • "activity": activity identifiers • "frequency": sampling frequency • "variant": variant label • "resolution": approximate horizontal resolution If NULL and out_dir is given, file name data.fst will be used. Default: NULL.
keep	Whether keep extracted data in memory. Default: TRUE if out_dir is NULL, and FALSE otherwise.
compress	A single integer in the range 0 to 100, indicating the amount of compression to use. Lower values mean larger file sizes. Default: 100.

Details

extract_data() uses future.apply underneath. You can use your preferable future backend to speed up data extraction in parallel. By default, extract_data() uses future::sequential backend, which runs things in sequential.

Value

An epw_cmip6_data object, which is basically a list of 3 elements:

- epw: An eplusr::Epw object whose longitude and latitute are used to extract CMIP6 data. It is the same object as created in match_coord()
- meta: A list containing basic meta data of input EPW, including city, state_province, country, latitute and longitude.
- data: An empty data.table::data.table() if keep is FALSE or a data.table::data.table() of 12 columns if keep is TRUE:

No.	Column	Type	Description
1	activity_drs	Character	Activity DRS (Data Reference Syntax)
2	institution_id	Character	Institution identifier
3	source_id	Character	Model identifier
4	experiment_id	Character	Root experiment identifier

5	member_id	Character	A compound construction from sub_experiment_id and variant_label
6	table_id	Character	Table identifier
7	lat	Double	Latitude of extracted location
8	lon	Double	Latitude of extracted location
9	variable	Character	Variable identifier
10	description	Character	Variable long name
11	units	Character	Units of variable
12	value	Double	Start date and time of simulation

Examples

```
## Not run:
coord <- match_coord("path_to_an_EPW")
extract_data(coord, years = 2030:2060)

## End(Not run)
```

---

future_epw *Create future EPW files using morphed data*

Description

Create future EPW files using morphed data

Usage

```
future_epw(
  morphed,
  by = c("experiment", "source", "interval"),
  dir = ".",
  separate = TRUE,
  overwrite = FALSE
)
```

Arguments

morphed	An epw_cmip6_morphed object created using morphing_epw().
by	A character vector of columns to be used as grouping variables when creating EPW files. Should be a subeset of: • "experiment": root experiment identifiers • "source": model identifiers • "variable": variable identifiers • "activity": activity identifiers • "frequency": sampling frequency • "variant": variant label

- `"resolution"`: approximate horizontal resolution
- `"longitude"`: averaged longitude of input data
- `"latitude"`: averaged latitude of input data

`dir` The parent directory to save the generated EPW files. If not exist, it will be created first. Default: `"."`, i.e., current working directory.

`separate` If `TRUE`, each EPW file will be saved into a separate folder using grouping variables specified in `by`.

`overwrite` If `TRUE`, overwrite existing files if they exist. Default: `FALSE`.

Value

A list of generated eplusr::Epw objects, invisibly

---

get_data_dir *Get the path of directory where epwshiftr data is stored*

Description

If option `epwshiftr.dir` is set, use it. Otherwise, get package data storage directory using `rappdirs::user_data_dir()`.

Usage

```
get_data_dir()
```

Value

A single string.

Examples

```
options(epwshiftr.dir = tempdir())
get_data_dir()
```

get_data_node *Get data nodes which store CMIP6 output*

Description

Get data nodes which store CMIP6 output

Usage

```
get_data_node(speed_test = FALSE, timeout = 3)
```

Arguments

speed_test	If TRUE, use pingr::ping() to perform connection speed test on each data node. A ping column is appended in returned data.table which stores each data node response in milliseconds. This feature needs pingr package already installed. Default: FALSE.
timeout	Timeout for a ping response in seconds. Default: 3.

Value

A data.table::data.table() of 2 or 3 (when speed_test is TRUE) columns:

Column	Type	Description
data_node	character	Web address of data node
status	character	Status of data node. "UP" means OK and "DOWN" means currently not available
ping	double	Data node response in milliseconds during speed test

Examples

```
get_data_node()
```

init_cmip6_index *Build CMIP6 experiment output file index*

Description

init_cmip6_index() will search the CMIP6 model output file using esgf_query() , return a data.table::data.table() containing the actual NetCDF file url to download, and store it into user data directory for future use.

Usage

```
init_cmip6_index(
  activity = "ScenarioMIP",
 variable = c("tas", "tasmax", "tasmin", "hurs", "hursmax", "hursmin", "pr", "rsds",
    "rlds", "psl", "sfcWind", "clt"),
  frequency = "day",
  experiment = c("ssp126", "ssp245", "ssp370", "ssp585"),
 source = c("AWI-CM-1-1-MR", "BCC-CSM2-MR", "CESM2", "CESM2-WACCM", "EC-Earth3",
    "EC-Earth3-Veg", "GFDL-ESM4", "INM-CM4-8", "INM-CM5-0", "MPI-ESM1-2-HR",
    "MRI-ESM2-0"),
  variant = "r1i1p1f1",
  replica = FALSE,
  latest = TRUE,
  resolution = c("100 km", "50 km"),
  limit = 10000L,
  data_node = NULL,
  years = NULL,
  save = FALSE
)
```

Arguments

`activity` A character vector indicating activity identifiers. Default: `"ScenarioMIP"`. Possible values:

- `"AerChemMIP"`: Aerosols and Chemistry Model Intercomparison Project,
- `"C4MIP"`: Coupled Climate Carbon Cycle Model Intercomparison Project,
- `"CDRMIP"`: Carbon Dioxide Removal Model Intercomparison Project,
- `"CFMIP"`: Cloud Feedback Model Intercomparison Project,
- `"CMIP"`: CMIP DECK: 1pctCO2, abrupt4xCO2, amip, esm-piControl, esm-historical, historical, and piControl experiments,
- `"CORDEX"`: Coordinated Regional Climate Downscaling Experiment,
- `"DAMIP"`: Detection and Attribution Model Intercomparison Project,
- `"DCPP"`: Decadal Climate Prediction Project,
- `"DynVarMIP"`: Dynamics and Variability Model Intercomparison Project,
- `"FAFMIP"`: Flux-Anomaly-Forced Model Intercomparison Project,
- `"GMMIP"`: Global Monsoons Model Intercomparison Project,
- `"GeoMIP"`: Geoengineering Model Intercomparison Project,
- `"HighResMIP"`: High-Resolution Model Intercomparison Project,
- `"ISMIP6"`: Ice Sheet Model Intercomparison Project for CMIP6,
- `"LS3MIP"`: Land Surface, Snow and Soil Moisture,
- `"LUMIP"`: Land-Use Model Intercomparison Project,
- `"OMIP"`: Ocean Model Intercomparison Project,
- `"PAMIP"`: Polar Amplification Model Intercomparison Project,
- `"PMIP"`: Palaeoclimate Modelling Intercomparison Project,
- `"RFMIP"`: Radiative Forcing Model Intercomparison Project,

- "SIMIP": Sea Ice Model Intercomparison Project,
- "ScenarioMIP": Scenario Model Intercomparison Project,
- "VIACSAB": Vulnerability, Impacts, Adaptation and Climate Services Advisory Board,
- "VolMIP": Volcanic Forcings Model Intercomparison Project

variable A character vector indicating variable identifiers. The 12 most related variables for EPW are set as defaults. If NULL, all possible variables are returned. Default: c("tas","tasmax","tasmin","hurs","hursmax","hursmin","psl","rss","rls","sfcWind","pr where:

- tas: Near-surface (usually, 2 meter) air temperature, units: K.
- tasmax: Maximum near-surface (usually, 2 meter) air temperature, units: K.
- tasmin: Minimum near-surface (usually, 2 meter) air temperature, units: K.
- hurs: Near-surface relative humidity, units: %.
- hursmax: Maximum near-surface relative humidity, units: %.
- hursmin: Minimum near-surface relative humidity, units: %.
- psl: Sea level pressure, units: Pa.
- rsds: Surface downwelling shortwave radiation, units: W m-2.
- rlds: Surface downwelling longware radiation, units: W m-2.
- sfcWind: Near-surface (usually, 10 meters) wind speed, units: m s-1.
- pr: Precipitation, units: kg m-2 s-1.
- clt: Total cloud area fraction for the whole atmospheric column, as seen from the surface or the top of the atmosphere. Units: %.

frequency A character vector of sampling frequency. If NULL, all possible frequencies are returned. Default: "day". Possible values:

- "1hr": sampled hourly,
- "1hrCM": monthly-mean diurnal cycle resolving each day into 1-hour means,
- "1hrPt": sampled hourly, at specified time point within an hour,
- "3hr": sampled every 3 hours,
- "3hrPt": sampled 3 hourly, at specified time point within the time period,
- "6hr": sampled every 6 hours,
- "6hrPt": sampled 6 hourly, at specified time point within the time period,
- "day": daily mean samples,
- "dec": decadal mean samples,
- "fx": fixed (time invariant) field,
- "mon": monthly mean samples,
- "monC": monthly climatology computed from monthly mean samples,
- "monPt": sampled monthly, at specified time point within the time period,
- "subhrPt": sampled sub-hourly, at specified time point within an hour,
- "yr": annual mean samples,
- "yrPt": sampled yearly, at specified time point within the time period

experiment A character vector indicating root experiment identifiers. The **Tier-1** experiment of activity ScenarioMIP are set as defaults. If NULL, all possible experiment are returned. Default: c("ssp126","ssp245","ssp370","ssp585").

source	A character vector indicating model identifiers. Defaults are set to 11 sources which give outputs of all 4 experiment of activity ScenarioMIP with daily frequency, i.e. "AWI-CM-1-1-MR", "BCC-CSM2-MR", "CESM2", "CESM2-WACCM", "EC-Earth3", "EC-Earth3-Veg", "GFDL-ESM4", "INM-CM4-8", "INM-CM5-0", "MPI-ESM1-2-HR" and "MRI-ESM2-0". If NULL, all possible sources are returned.
variant	A character vector indicating label constructed from 4 indices stored as global attributes in format r<k>i<l>p<m>f<n> described below. Default: "r1i1p1f1". If NULL, all possible variants are returned. • r: realization_index (<k>) = realization number (integer >0) • i: initialization_index (<l>) = index for variant of initialization method (integer >0) • p: physics_index (<m>) = index for model physics variant (integer >0) • f: forcing_index (<n>) = index for variant of forcing (integer >0)
replica	Whether the record is the "master" copy, or a replica. Use FALSE to return only originals and TRUE to return only replicas. Default: FALSE.
latest	Whether the record is the latest available version, or a previous version. Use TRUE to return only the latest version of all records and FALSE to return previous versions. Default: FALSE.
resolution	A character vector indicating approximate horizontal resolution. Default: c("50 km","100 km"). If NULL, all possible resolutions are returned.
limit	An integer indicating the maximum of matched records to return. Should be <= 10,000. Default: 10000.
data_node	A character vector indicating data nodes to be queried. Default to NULL, which means all possible data nodes.
years	An integer vector indicating the target years to be include in the data file. All other years will be excluded. If NULL, no subsetting on years will be performed. Default: NULL.
save	If TRUE, the results will be saved into user data directory. Default: FALSE.

Details

For details on where the file index is stored, see rappdirs::user_data_dir().

Value

A data.table::data.table with 22 columns:

No.	Column	Type	Description
1	file_id	Character	Model output file universal identifier
2	dataset_id	Character	Dataset universal identifier
3	mip_era	Character	Activity's associated CMIP cycle. Will always be "CMIP6"
4	activity_drs	Character	Activity DRS (Data Reference Syntax)
5	institution_id	Character	Institution identifier
6	source_id	Character	Model identifier
7	experiment_id	Character	Root experiment identifier

8	member_id	Character	A compound construction from sub_experiment_id and variant_label
9	table_id	Character	Table identifier
10	frequency	Character	Sampling frequency
11	grid_label	Character	Grid identifier
12	version	Character	Approximate date of model output file
13	nominal_resolution	Character	Approximate horizontal resolution
14	variable_id	Character	Variable identifier
15	variable_long_name	Character	Variable long name
16	variable_units	Character	Units of variable
17	datetime_start	POSIXct	Start date and time of simulation
18	datetime_end	POSIXct	End date and time of simulation
19	file_size	Character	Model output file size in Bytes
20	data_node	Character	Data node to download the model output file
21	dataset_pid	Character	A unique string that helps identify the dataset
22	tracking_id	Character	A unique string that helps identify the output file

Note

Argument `limit` will only apply to `Dataset` query. `init_cmip6_index()` will try to get all model output files which match the dataset id.

Examples

```
## Not run:
init_cmip6_index()

## End(Not run)
```

---

load_cmip6_index *Load previously stored CMIP6 experiment output file index*

Description

Load previously stored CMIP6 experiment output file index

Usage

```
load_cmip6_index(force = FALSE)
```

Arguments

force	If `TRUE`, read the index file. Otherwise, return the cached index if exists. Default: `FALSE`.

Value

A data.table::data.table with 20 columns. For detail description on column, see `init_cmip6_index()`.

Examples

```
## Not run:
load_cmip6_index()

## End(Not run)
```

`match_coord` *Match coordinates of input EPW in the CMIP6 output file database*

Description

`match_coord()` takes an EPW and uses its longitude and latitude to match corresponding values that meet specified threshold in NetCDF files.

Usage

```
match_coord(epw, threshold = list(lon = 1, lat = 1), max_num = NULL)
```

Arguments

`epw`	Possible values: • A file path of EPW file • An eplusr::Epw object • A regular expression used to search locations in EnergyPlus Weather Database, e.g. "los angeles.*tmy3". You will be asked to select a matched EPW to download and read. It will be saved into `tempdir()`. Note that the search is case-insensitive
`threshold`	A list of 2 elements `lon` and `lat` specifying the absolute distance threshold used when matching longitude and latitude. Default: `list(lon = 1.0, lat = 1.0)`
`max_num`	The maximum number to be matched for both longitude and latitude when `threshold` is matched. Default is `NULL`, which means no limit

Details

`match_coord()` uses future.apply underneath. You can use your preferable future backend to speed up data extraction in parallel. By default, `match_coord()` uses future::sequential backend, which runs things in sequential.

Value

An `epw_cmip6_coord` object, which is basically a list of 3 elements:

- `epw`: An eplusr::Epw object parsed from input `epw` argument
- `meta`: A list containing basic meta data of input EPW, including `city`, `state_province`, `country`, `latitute` and `longitude`.

- coord: A data.table::data.table() which is basically CMIP6 index database with an appending new list column coord that contains matched latitudes and longitudes in each NetCDF file. Each element in coord contains 2 elements lat and lon, in which contains the 4 components describing the matched coordinates.
  - index: the indices of matched coordinates
  - value: the actual longitude or latitude in the NetCDF coordinate grids
  - dis: the distance between the coordinate values in NetCDF and input EPW
  - which: The value indices of longitude or latitude in the NetCDF coordinate grids. These values are used to extract the corresponding variable values

Examples

```
## Not run:
# download an EPW from EnergyPlus website
epw <- eplusr::download_weather("los angeles.*TMY3", dir = tempdir(),
    type = "EPW", ask = FALSE)

match_coord(epw, threshold = list(lon = 1.0, lat = 1.0))

## End(Not run)
```

---

morphing_epw *Morphing EPW weather variables*

Description

morphing_epw() takes an epw_cmip6_data object generated using extract_data() and calculates future core EPW weather variables using Morphing Method.

Usage

```
morphing_epw(data, years = NULL, labels = NULL)
```

Arguments

data	An epw_cmip6_dataobject generated using extract_data()
years	An integer vector indicating the target years to be considered. If NULL, all years in input data will be considered. Default: NULL.
labels	A character or factor vector used for grouping input years. Usually are the outputs of base::cut(). labels should have the same length as years. If given, climate data of years grouped by labels will be averaged. Default: NULL.

Details

The EPW weather variables that get morphed are listed in details.

Value

An epw_cmip6_morphed object, which is basically a list of 12 elements:

No.	Element	Type	Morphing Method	Description
1	epw	eplusr::Epw	N/A	The original EPW file used for morphing
2	tdb	data.table::data.table()	Stretch	Data of dry-bulb temperature after morphing
3	tdew	data.table::data.table()	Derived	Data of dew-point temperature after morphing
4	rh	data.table::data.table()	Stretch	Data of relative humidity after morphing
5	p	data.table::data.table()	Stretch	Data of atmospheric pressure after morphing
6	hor_ir	data.table::data.table()	Stretch	Data of horizontal infrared radiation from the sk
7	glob_rad	data.table::data.table()	Stretch	Data of global horizontal radiation after morphi
8	norm_rad	data.table::data.table()	Derived	Data of direct normal radiation after morphing
9	diff_rad	data.table::data.table()	Stretch	Data of diffuse horizontal radiation after morphi
10	wind	data.table::data.table()	Stretch	Data of wind speed after morphing
11	total_cover	data.table::data.table()	Derived	Data of total sky cover after morphing
12	opaque_cover	data.table::data.table()	Derived	Data of opaque sky cover after morphing

Each data.table::data.table() listed above contains x columns

No.	Column	Type	Description
1	activity_drs	Character	Activity DRS (Data Reference Syntax)
2	institution_id	Character	Institution identifier
3	source_id	Character	Model identifier
4	experiment_id	Character	Root experiment identifier
5	member_id	Character	A compound construction from sub_experiment_id and variant_label
6	table_id	Character	Table identifier
7	lat	Double	The **averaged** values of input latitude
8	lon	Double	The **averaged** values of input longitude
9	interval	Factor	The year value of data morphed
10	**Variable Name**	Double	The morphed data, where Variable Name is the corresponding EPW weather variable n
11	delta	Double	The shift factor. Will be NA for derivied values
12	alpha	Double	The stretch factor. Will be NA for derivied values

References

Belcher, S., Hacker, J., Powell, D., 2005. Constructing design weather data for future climates. Building Services Engineering Research and Technology 26, 49–61. https://doi.org/10.1191/0143624405bt112oa

---

set_cmip6_index *Set CMIP6 index*

Description

set_cmip6_index() takes a data.table::data.table() as input and set it as current index.

Usage

```
set_cmip6_index(index, save = FALSE)
```

Arguments

index	A data.table::data.table() containing the same column names and types as the output of init_cmip6_index().
save	If TRUE, besides loaded index, the index file saved to data directory will be also updated. Default: FALSE.

Details

set_cmip6_index() is useful when init_cmip6_index() may give you too much cases of which only some are of interest.

Value

A data.table::data.table().

---

summary_database *Summary CMIP6 model output file status*

Description

summary_database() scan the directory specified and returns a data.table() containing summary information about all the CMIP6 files available against the output file index loaded using load_cmip6_index().

Usage

```
summary_database(
  dir,
  by = c("activity", "experiment", "variant", "frequency", "variable", "source",
    "resolution"),
  mult = c("skip", "latest"),
  append = FALSE,
  recursive = FALSE,
  update = FALSE,
  warning = TRUE
)
```

Arguments

`dir`	A single string indcating the directory where CMIP6 model output NetCDF files are stored.
`by`	The grouping column to summary the database status. Should be a subeset of: • `"experiment"`: root experiment identifiers • `"source"`: model identifiers • `"variable"`: variable identifiers • `"activity"`: activity identifiers • `"frequency"`: sampling frequency • `"variant"`: variant label • `"resolution"`: approximate horizontal resolution
`mult`	Actions when multiple files match a same case in the CMIP6 index. If `"latest"`, the file with latest modification time will be used. If `"skip"`, all matched files will be skip and this case will be kept as unmatched. Default: `"skip"`.
`append`	If `TRUE`, status of CMIP6 files will only be updated if they are not found in previous summary. This is useful if CMIP6 files are stored in different directories. Default: `FALSE`.
`recursive`	If `TRUE`, scan recursively into directories. Default: `FALSE`.
`update`	If `TRUE`, the output file index will be updated based on the matched NetCDF files in specified directory. If `FALSE`, only current loaded index will be updated, but the actual index database file saved in `get_data_dir()` will remain unchanged. Default: `FALSE`.
`warning`	If `TRUE`, warning messages will show when multiple files match a same case. Default: `TRUE`.

Details

`summary_database()` uses future.apply underneath. You can use your preferable future backend to speed up data extraction in parallel. By default, `summary_database()` uses future::sequential backend, which runs things in sequential.

Value

A `data.table::data.table()` containing corresponding grouping columns plus:

Column	Type	Description
`datetime_start`	POSIXct	Start date and time of simulation
`datetime_end`	POSIXct	End date and time of simulation
`file_num`	Integer	Total number of file per group
`file_size`	Units (Mbytes)	Approximate total size of file
`dl_num`	Integer	Total number of file downloaded
`dl_percent`	Units (%)	Total percentage of file downloaded
`dl_size`	Units (Mbytes)	Total size of file downloaded

Also an attribute not_matched is added to the returned data.table::data.table() which contains meta data for those CMIP6 output files that are not covered by current CMIP6 output file index.

For the meaning of grouping columns, see init_cmip6_index().

Examples

```
## Not run:
summary_database()

summary_database(by = "experiment")

## End(Not run)
```

Index

base::cut(), *17*

data.table(), *19*
data.table::data.table, *6*, *14*, *15*
data.table::data.table(), *8*, *11*, *17–21*

eplusr::Epw, *8*, *10*, *16*, *18*
epwshiftr *(*epwshiftr-package*)*, 2
epwshiftr-package, 2
esgf_query, 3
esgf_query(), *11*
extract_data, 7
extract_data(), *17*

fst::write_fst(), *8*
future, *8*, *16*, *20*
future.apply, *8*, *16*, *20*
future::sequential, *8*, *16*, *20*
future_epw, 9

get_data_dir, 10
get_data_dir(), *20*
get_data_node, 11

init_cmip6_index, 11
init_cmip6_index(), *15*, *19*, *21*

load_cmip6_index, 15
load_cmip6_index(), *19*

match_coord, 16
match_coord(), *7*, *8*
morphing_epw, 17
morphing_epw(), *9*

pingr::ping(), *11*

rappdirs::user_data_dir(), *10*, *14*

set_cmip6_index, 18
summary_database, 19

tempdir(), *16*

units::set_units(), *8*