Package ‘rWBclimate’

February 20, 2015

Version 0.1.3
License MIT + file LICENSE
URL http://github.com/ropensci/rWBclimate
BugReports http://github.com/ropensci/rWBclimate/issues
Type Package
Title A package for accessing World Bank climate data
Description This package will download model predictions from 15 different global circulation models in 20 year intervals from the world bank. Users can also access historical data, and create maps at 2 different spatial scales.
LazyData True
VignetteBuilder knitr
Suggests knitr
Imports ggplot2, httr, plyr, rgdal, jsonlite, reshape2, sp
Author Edmund Hart [aut, cre]
Maintainer Edmund Hart <edmund.m.hart@gmail.com>
NeedsCompilation no
Repository CRAN
Date/Publication 2014-04-19 00:53:12

R topics documented:

   Africa_basin                        ........................................ 2
   Africa_country                      ........................................ 2
   Asia_basin                          ........................................ 3
   Asia_country                        ........................................ 3
   check_ISO_code                      ........................................ 3
   check_locator                       ........................................ 4
   climate_map                         ........................................ 4
   codes                               ........................................ 5
   create_map_df                       ........................................ 5
Index

**Africa_basin**

Basin codes for Africa, used in downloading maps

**Description**

Basin codes for Africa, used in downloading maps

**Africa_country**

Country codes for all of Africa

**Description**

Country codes for all of Africa
### Asia_basin

**Basin codes for Asia, used in downloading maps**

### Description

Basin codes for Asia, used in downloading maps

### Asia_country

**Country codes for all of Asia**

### Description

Country codes for all of Asia

### check_ISO_code

**check country codes**

### Description

Checks if the country code entered is a valid country code that data exists for

### Usage

```r
check_ISO_code(iso)
```

### Arguments

<table>
<thead>
<tr>
<th>iso</th>
<th>The 3 letter country code based on ISO3 Country abbreviations (<a href="http://unstats.un.org/unsd/methods/m49/m49alpha.htm">http://unstats.un.org/unsd/methods/m49/m49alpha.htm</a>)</th>
</tr>
</thead>
</table>

### Value

TRUE if a valid code, otherwise an error is returned

### Examples

```r
## Not run:
check_ISO_code("USA")

## End(Not run)
```
check_locator  \hspace{3cm} \textit{Checks for what kind of locator a user input}

\textbf{Description}

Checks for what kind of locator a user input

\textbf{Usage}

\texttt{check\_locator(locator)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{locator} The ISO3 country code that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm) or the basin ID [1-468]
\end{itemize}

\textbf{Value}

\begin{itemize}
  \item \texttt{geo\_ref} a string indicating what kind of geography to use in the api
\end{itemize}

\textbf{climate\_map} \hspace{3cm} \textit{Map climate data}

\textbf{Description}

Create maps of climate data. It requires two data inputs, a map dataframe, and a climate dataframe. The climate data must have one data point per spatial mapping point, e.g. 1 data point per country or basin being mapped.

\textbf{Usage}

\texttt{climate\_map(map\_df, data\_df, return\_map = TRUE)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{map\_df} a map dataframe generated from create\_map\_df()
  \item \texttt{data\_df} a climate dataframe with one piece of data to be mapped to each unique spatial polygon.
  \item \texttt{return\_map} True returns a ggplot2 object. False returns a dataframe where data items are matched to their polygon that you can plot later on.
\end{itemize}

\textbf{Value}

Either a ggplot2 map or a dataframe depending on the parameter return\_map
Examples

```r
## Not run:
# Set the kmlpath option
options(kmlpath = "/Users/edmundhart/kmltemp")  
## Here we use a list basins for Africa
af_basin <- create_map_df(Africa.basin)
af_basin_dat <- get_ensemble_temp(Africa.basin,"annualanom",2080,2100)
## Subset data to just one scenario, and one percentile
af_basin_dat <- subset(af_basin_dat,af_basin_dat$scenario == "a2")
af_basin_dat <- subset(af_basin_dat,af_basin_dat$percentile == 50)
af_map <- climate_map(af_basin,af_basin_dat,return_map = T)
af_map + scale_fill_continuous("Temperature \n anomaly",low="yellow",high = "red") + theme_bw()
```

## End(Not run)

---

codes

### isocodes data

#### Description

**isocodes data**

#### create_map_df

**Create mapable dataframe**

#### Description

A function that will download maps for a vector of basins or country codes and return a dataframe that has the kml output processed such that it can be plotted with ggplot2 and other mapping functions.

#### Usage

`create_map_df(locator)`

#### Arguments

- **locator**

#### Details

kml files can be quite large (100k-600k per country) making downloading them every time you want to make a map time consuming. To reduce this time it’s easiest to download kml files and store them. To set the directory use a line like this: `options(kmlpath="/Users/emh/kmltemp")` The option must be called "kmlpath". These files will be persistent until you delete them.
Examples

```r
## Not run:
to_map <- create_map_df(c("USA","MEX","CAN"))
ggplot(to_map, aes(x=long, y=lat,group=group)) + geom_polygon()

## End(Not run)
```

date_correct  correct data values

Description

Round start and end dates to conform with data api standards. See api documentation (http://data.worldbank.org/developers/climate-data-api) for full details

Usage

date_correct(start, end)

Arguments

- `start` The start year
- `end` The end year

Value

- a 2xM matrix where M is the number of periods in the data api

Examples

```r
## Not run:
date_correct(1921,1957)

## End(Not run)
```

download_kml  Download kml files

Description

Downloads map data from kml format and writes it to a temporary directory. You must specify a temporary directory to write files to in your options.

Usage

download_kml(locator)
**Eur_basin**

**Arguments**

locator

The a vector of ISO3 country code’s that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm)
or the basin ID’s [1-468] (http://data.worldbank.org/sites/default/files/climate_data_api_basins.pdf)

**Details**

kml files can be quite large making downloading them every time you want to make a map time consuming. To reduce this time it’s easiest to download kml files and store them. To set the directory use a line like this: options(kmlpath="/Users/emh/kmltemp") The option must be called "kmlpath".

---

**Eur_basin**

*Basin codes for Eur, used in downloading maps*

**Description**

Basin codes for Eur, used in downloading maps

---

**Eur_country**

*Country codes for all of Eur*

**Description**

Country codes for all of Eur

---

**get_climate_data**

*get_climate_data*

**Description**

Download monthly average climate data from the world bank climate data api. Ideally you’ll want to use the wrapper functions that call this.

**Usage**

get_climate_data(locator, geo_type, type, cvar, start, end)
Arguments

**locator**  
The ISO3 country code that you want data about. ([http://unstats.un.org/unsd/methods/m49/m49alpha.htm](http://unstats.un.org/unsd/methods/m49/m49alpha.htm)) or the basin ID [1-468]

**geo_type**  
basin or country depending on the locator type

**type**  
the type of data you want "mavg" for monthly averages, "annualavg"

**cvar**  
The variable you're interested in. "pr" for precipitation, "tas" for temperature in celsius.

**start**  
The starting year you want data for, can be in the past or the future. Must conform to the periods outlined in the world bank API. If given values don't conform to dates, the function will automatically round them.

**end**  
The ending year you want data for, can be in the past or the future. Similar to the start date, dates will be rounded to the nearest end date.

---

get_data_recursive  
wrapper for get_climate_data()

Description

Function to recursively call the get_climate_data(). Handles a vector of basins or countries as well as multiple dates.

Usage

get_data_recursive(locator, geo_type, type, cvar, start, end)

---

Arguments

**locator**  
The ISO3 country code that you want data about. ([http://unstats.un.org/unsd/methods/m49/m49alpha.htm](http://unstats.un.org/unsd/methods/m49/m49alpha.htm)) or the basin ID [1-468]

**geo_type**  
basin or country depending on the locator type

**type**  
the type of data you want "mavg" for monthly averages, "annualavg"

**cvar**  
The variable you're interested in. "pr" for precipitation, "tas" for temperature in celsius.

**start**  
The starting year you want data for, can be in the past or the future. Must conform to the periods outlined in the world bank API. If given values don't conform to dates, the function will automatically round them.

**end**  
The ending year you want data for, can be in the past or the future. Similar to the start date, dates will be rounded to the nearest end date.

Examples

```r
## Not run:
get_ensemble_data_recursive(c("1","2"),"basin","mavg","pr",1920,1940)
```

```r
## End(Not run)
```
**get_ensemble_climate_data**

*Description*

Download ensemble data for all models, returns the 10th, 50th and 90th percentile of all models (15 for A1, 13 for B2). Ensemble requests can be for countries or basins.

**Usage**

```r
get_ensemble_climate_data(locator, geo_type, type, cvar, start, end)
```

**Arguments**

- **locator**: The ISO3 country code that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm) or the basin ID [1-468]
- **geo_type**: basin or country depending on the locator type
- **type**: the type of data you want "mavg" for monthly averages, "annualavg"
- **cvar**: The variable you're interested in. "pr" for precipitation, "tas" for temperature in celsius.
- **start**: The starting year you want data for, can be in the past or the future. Must conform to the periods outlined in the world bank API. If given values don't conform to dates, the function will automatically round them.
- **end**: The ending year you want data for, can be in the past or the future. Similar to the start date, dates will be rounded to the nearest end dat.

---

**get_ensemble_data_recursive**

*Wrapper for get_ensemble_climate_data()*

**Description**

Function to recursively call the get_ensemble_climate_data(). Handles a vector of basins or countries as well as multiple dates.

**Usage**

```r
get_ensemble_data_recursive(locator, geo_type, type, cvar, start, end)
```
get_ensemble_precip

Arguments

locator The ISO3 country code that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm) or the basin ID [1-468]

geo_type basin or country depending on the locator type

type the type of data you want "mavg" for monthly averages, "annualavg"

cvar The variable you're interested in. "pr" for precipitation, "tas" for temperature in celcius.

start The starting year you want data for, can be in the past or the future. Must conform to the periods outlined in the world bank API. If given values don't conform to dates, the fuction will automatically round them.

end The ending year you want data for, can be in the past or the future. Similar to the start date, dates will be rounded to the nearest end dat.

Examples

## Not run:

```r
get_ensemble_data_recursive(c("1","2"),"basin","mavg","pr",1920,1940)
```

## End(Not run)

get_ensemble_precip Download ensemble precipitation data

Description

Function wraps get_ensemble_climate_data() and returns precipitation by basin or country in mm. Output is the 10th 50th and 90th percentile for all gcm's for the a1 and b2 scenarios.

Usage

```r
get_ensemble_precip(locator, type, start, end)
```

Arguments

locator A vector of either watershed basin ID’s from http://data.worldbank.org/sites/default/files/climate_data_api_basins.pdf. It can be just a single basin id, or a vector of ids. ids should be strings.

type the type of data to retrieve, must be "mavg" for monthly averages, "annualavg" for annual averages, "manom" for monthly anomaly, and "annualanom" for annual anomaly.

start the start year to gather data for.

end the end year to gather data to.
get_ensemble_stats

Details

start and end year can be any years, but all years will be coerced into periods outlined by the API (http://data.worldbank.org/developers/climate-data-api) anomaly periods are only valid for future scenarios and based on a reference period of 1969 - 1999, see API for full details.

Value

a dataframe with precipitation predictions in mm for all scenarios, gcms, for each time period.

Examples

# Not run:
# # get data for R basins annual average precipitation for all valid time periods
# then subset them and plot
precip_dat <- get_ensemble_precip(c("2","311"),"annualavg",1900,3000)
p precip_dat <- subset(precip_dat,precip_dat$scenario!="b1")
p precip_dat$uniqueGroup <- paste(precip_dat$percentile,precip_dat$locator,sep="-")
ggplot(precip_dat,aes(x=fromYear,y=annualVal,group=uniqueGroup,colour=as.factor(locator),linetype=as.factor(percentile)))+ geom_path()

### Get data for 2 countries with monthly precipitation values
precip_dat <- get_ensemble_precip(c("USA","BRA"),"mavg",2020,2030)
p precip_dat <- subset(precip_dat,precip_dat$scenario!="b1")
p precip_dat$uniqueGroup <- paste(precip_dat$percentile,precip_dat$locator,sep="-")
ggplot(precip_dat,aes(x=as.factor(month),y=monthVals,group=uniqueGroup,
colour=locator))+geom_path()

### End(Not run)

---

get_ensemble_stats  Download ensemble statistics

Description

Statistics can be from either two time periods: 2046 - 2065 and 2081 - 2100 and are all given in units relative to a control period: 1961 - 2000. Derived statistics can be any of the following:

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tmin_means</td>
<td>Average daily minimum temperature</td>
</tr>
<tr>
<td>tmax_means</td>
<td>Average daily maximum temperature</td>
</tr>
<tr>
<td>tmax_days90th</td>
<td>Number of days with maximum temperature above the control period 90th percentile (hot days)</td>
</tr>
<tr>
<td>tmin_days90th</td>
<td>Number of days with minimum temperature above the control period 90th percentile (warm nights)</td>
</tr>
<tr>
<td>tmax_days10th</td>
<td>Number of days with maximum temperature below the control period 10th percentile (cool days)</td>
</tr>
<tr>
<td>tmin_days10th</td>
<td>Number of days with minimum temperature below the control period 10th percentile (cold nights)</td>
</tr>
<tr>
<td>tmin_days0</td>
<td>Number of days with minimum temperature below 0 degrees Celsius</td>
</tr>
<tr>
<td>ppt_days</td>
<td>Number of days with precipitation greater than 0.2 mm</td>
</tr>
<tr>
<td>ppt_days2</td>
<td>Number of days with precipitation greater than 2 mm</td>
</tr>
<tr>
<td>ppt_days10</td>
<td>Number of days with precipitation greater than 10 mm</td>
</tr>
</tbody>
</table>

Units

degrees

degrees

days

days

days

days

days

days

---
### ppt_days90th
Number of days with precipitation greater than the control periods 90th percentile

### ppt_dryspell
Average number of days between precipitation events

### ppt_means
Average daily precipitation

**Usage**

```r
get_ensemble_stats(locator, type, stat)
```

**Arguments**

- **locator**: The ISO3 country code that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm) or the basin ID [1-468]
- **type**: the type of data you want "mavg" for monthly averages, "annualavg"
- **stat**: The statistics of interest, must be one of the ones listed above.

**Examples**

```r
## Not run:
## Request data on the US for days of rain over 2 mm
ens_dat <- get_ensemble_stats("USA","mavg","ppt_days2")
# subset to the 50th percentile and just until the year 2100
ens_dat <- subset(ens_dat, ens_dat$percentile == 50)
ens_dat <- subset(ens_dat,ens_dat$toYear == 2100)
ggplot(ens_dat,aes(x = as.factor(month), y= monthVals, group=scenario, colour=scenario)) + geom_point() + geom_line()

## End(Not run)
```

---

### get_ensemble_temp
**Download ensemble temperature data**

**Description**

Function wraps get_ensemble_climate_data() and returns precipitation by basin or country in mm. Output is the 10th 50th and 90th percentile for all gcm’s for the a1 and b2 scenarios.

**Usage**

```r
get_ensemble_temp(locator, type, start, end)
```

**Arguments**

- **locator**: A vector of either watershed basin ID’s from http://data.worldbank.org/sites/default/files/climate_data_api_basins.pdf. It can be just a single basin id, or a vector of ids. ids should be strings.
- **type**: the type of data to retrieve, must be "mavg" for monthly averages, "annualavg" for annual averages, "manom" for monthly anomaly, and "annualanom" for annual anomaly.
- **start**: the start year to gather data for.
- **end**: the end year to gather data to.
get_historical_data

Description
The Climate Data API provides access to historical temperature and precipitation data. These data are separate from the outputs of the GCMs, and they are based on gridded climatologies from the Climate Research Unit.

Usage
get_historical_data(locator, cvar, time_scale)

Arguments
locator The ISO3 country code that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm) or the basin ID [1-468]. The historical period for country is 1901 - 2009, and 1960 - 2009 for basin.

details
start and end year can be any years, but all years will be coerced into periods outlined by the API (http://data.worldbank.org/developers/climate-data-api) anomaly periods are only valid for future scenarios and based on a reference period of 1969 - 1999, see API for full details.

Value
a dataframe with precipitation predictions in mm for all scenarios, gcms, for each time period.

Examples
## Not run:
# Get data for 2 basins, annual average precipitation for all valid time periods
# then subset them, and plot
temp_dat <- get_ensemble_temp(c("2","231"),"annualavg",1900,3000)
temp_dat <- subset(temp_dat,temp_dat$scenario!="b1")
temp_dat$uniqueGroup <- paste(temp_dat$percentile,temp_dat$locator,sep='-')
ggplot(temp_dat,aes(x=fromYear,y=annualVal,group=uniqueGroup,colour=as.factor(locator),
linetype=as.factor(percentile)))+geom_path()

## Get data for 2 countries with monthly precipitation values
temp_dat <- get_ensemble_temp(c("USA","BRA"),"mavg",2020,2030)
temp_dat <- subset(temp_dat,temp_dat$scenario!="b1")
temp_dat$uniqueGroup <- paste(temp_dat$percentile,temp_dat$locator,sep='-')
ggplot(temp_dat,aes(x=as.factor(month),y=monthVals,group=uniqueGroup,
colour=locator))+geom_path()

## End(Not run)
cvar The climate variable you're interested in. "pr" for precipitation, "tas" for temperature in celcius.

time_scale The time scale you want to return values on. Must be "month", "year" or "decade"

Details
The time_scale parameter returns a different number of variables depending on the input timescale. 
Month will return 12 values, a historical average for that month across all years. Year will return yearly averages for each year, and decade will return decade averages.

Value
a dataframe with historical climate data

get_historical_data_recursive

Download historical climate data recursively

Description
Recursively get historical data

Usage
get_historical_data_recursive(locator, cvar, time_scale)

Arguments
locator The ISO3 country code that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm) or the basin ID [1-468]. The historical period for country is 1901 - 2009, and 1960 - 2009 for basin
cvar The climate variable you're interested in. "pr" for precipitation, "tas" for temperature in celcius.
time_scale The time scale you want to return values on. Must be "month", "year" or "decade"

Details
The time_scale parameter returns a different number of variables depending on the input timescale. 
Month will return 12 values, a historical average for that month across all years. Year will return yearly averages for each year, and decade will return decade averages.

Value
a dataframe with historical climate data
**get_historical_precip**

**Description**

The Climate Data API provides access to historical precipitation data. These data are separate from the outputs of the GCMs, and they are based on gridded climatologies from the Climate Research Unit.

**Usage**

`get_historical_precip(locator, time_scale)`

**Arguments**

- **locator**
  - The ISO3 country code that you want data about. ([http://unstats.un.org/unsd/methods/m49/m49alpha.htm](http://unstats.un.org/unsd/methods/m49/m49alpha.htm)) or the basin ID [1-468]. This can be a vector of all basins or all countries.

- **time_scale**
  - The time scale you want to return values on. Must be "month", "year" or "decade"

**Details**

The historical period for country is 1901 - 2009, and 1960 - 2009 for basin. The `time_scale` parameter returns a different number of variables depending on the input timescale. *Month* will return 12 values, a historical average for that month across all years. *Year* will return yearly averages for each year, and *decade* will return decade averages.

**Value**

- a dataframe with historical precipitation data

**Examples**

```r
## Not run:
## Plot annual historical data for USA, Brazil and Australia
hist_dat <- get_historical_precip(c("USA", "BRA", "AUS"), "year")
ggplot(hist_dat, aes(x = year, y = data, group = locator, colour = locator)) + geom_point() + geom_path() + ylab("Mean annual precipitation")

## Plot monthly historical data
hist_mo_dat <- get_historical_precip(c("USA", "AUS", "BRA", "IDN"), time_scale = "month")
ggplot(hist_mo_dat, aes(x = month, y = data, group = locator, colour = locator)) + geom_point() + geom_path() + ylab("Mean monthly precipitation")

## End(Not run)
```
get_historical_temp  Download historical temperature data

Description

The Climate Data API provides access to historical precipitation data. These data are separate from
the outputs of the GCMs, and they are based on gridded climatologies from the Climate Research
Unit.

Usage

going_historical_temp(locator, time_scale)

Arguments

locator The ISO3 country code that you want data about. (http://unstats.un.org/unsd/methods/m49/m49alpha.htm)
or the basin ID [1-468]. This can be a vector of all basins or all countries.
time_scale The time scale you want to return values on. Must be "month", "year" or "decade"

Details

The historical period for country is 1901 - 2009, and 1960 - 2009 for basin. The time_scale param-
eter returns a different number of variables depending on the input timescale. Month will return 12
values, a historical average for that month across all years. Year will return yearly averages for each
year, and decade will return decade averages.

Value

a dataframe with historical temperature data

Examples

## Not run:
## Plot annual historical data for USA, Brazil and Australia
hist_dat <- get_historical_precip(c("USA", "BRA", "AUS"), "year")
ggplot(hist_dat, aes(x = year, y = data, group = locator,
 colour = locator)) + geom_point() + geom_path() + ylab("Mean annual temperature")

## Plot monthly historical data
hist_mo_dat <- get_historical_precip(c("USA", "AUS", "BRA", "IDN"), time_scale = "month")
ggplot(hist_mo_dat, aes(x = month, y = data, group = locator,
 colour = locator)) + geom_point() + geom_path() + ylab("Mean monthly temperature")

## End(Not run)
get_model_precip

Description
Function wraps get_climate_data() and returns precipitation by basin or country in mm as output from all 15 models, for the a1 and b2 scenarios.

Usage
get_model_precip(locator, type, start, end)

Arguments
- locator: A vector of either watershed basin ID’s from http://data.worldbank.org/sites/default/files/climate_data_api_basins.pdf. It can be just a single basin id, or a vector of ids. ids should be strings.
- type: the type of data to retrieve, must be "mavg" for monthly averages, "annualavg" for annual averages, "manom" for monthly anomaly, and "annualanom" for annual anomaly.
- start: the start year to gather data for.
- end: the end year to gather data to.

Details
Start and end year can be any years, but all years will be coerced into periods outlined by the API (http://data.worldbank.org/developers/climate-data-api) anomaly periods are only valid for future scenarios and based on a reference period of 1969 - 1999, see API for full details.

Value
A dataframe with precipitation predictions in mm for all scenarios, gcms, for each time period.

Examples
```r
# Not run:
# Get data for 2 basins, annual average precipitation for all valid time periods
# then subset them, and plot
precip_dat <- get_model_precip(c("2", "231"), "annualavg", 1900, 3000)
precip_dat <- subset(precip_dat, precip_dat$gcm == "ukmo_hadcm3")
precip_dat <- subset(precip_dat, precip_dat$scenario == "b1")
ggplot(precip_dat, aes(x=fromYear, y=annualData, group=locator, colour=locator)) + geom_path()

### Get data for 4 countries with monthly precipitation values
precip_dat <- get_model_precip(c("USA", "BRA", "CAN", "YEM"), "mavg", 2020, 2030)
precip_dat <- subset(precip_dat, precip_dat$gcm == "ukmo_hadcm3")
precip_dat <- subset(precip_dat, precip_dat$scenario == "b1")
ggplot(precip_dat, aes(x=as.factor(month), y=monthVals, group=locator, colour=locator)) + geom_path()

# End(Not run)
```
get_model_temp

Description

Function wraps get_climate_data() and returns temperature by basin or country in degrees C as output from all 15 models, for the a1 and b2 scenarios.

Usage

get_model_temp(locator, type, start, end)

Arguments

- **locator**: A vector of either watershed basin ID’s from http://data.worldbank.org/sites/default/files/climate_data_api_basins.pdf. It can be just a single basin id, or a vector of ids. ids should be strings.
- **type**: the type of data to retrieve, must be "mavg" for monthly averages, "annualavg" for annual averages, "manom" for monthly anomaly, and "annualanom" for annual anomaly.
- **start**: the start year to gather data for.
- **end**: the end year to gather data to.

Details

start and end year can be any years, but all years will be coerced into periods outlined by the API (http://data.worldbank.org/developers/climate-data-api) anomaly periods are only valid for future scenarios and based on a reference period of 1969 - 1999, see API for full details.

Value

a dataframe with temperature predictions in degrees C for all scenarios, gcms, for each time period.

Examples

```r
# Not run:
## Get data for 2 basins, annual average temperature for all valid time periods
## then subset them, and plot
temp_dat <- get_model_temp(c("2","231"),"annualavg",1900,3000)
temp_dat <- subset(temp_dat,temp_dat$gcm=="ukmo_hadcm3")
temp_dat <- subset(temp_dat,temp_dat$scenario!="b1")
ggplot(temp_dat,aes(x=fromYear,y=data,group=locator, colour=locator))+geom_path()

### Get data for 4 countries with monthly tempitation values
temp_dat <- get_model_temp(c("USA","BRA","CAN","YEM"),"mavg",2020,2030)
temp_dat <- subset(temp_dat,temp_dat$gcm=="ukmo_hadcm3")
temp_dat <- subset(temp_dat,temp_dat$scenario!="b1")
ggplot(temp_dat,aes(x=as.factor(month),y=data,group=locator,
```
**kml_to_sp**

```
colour=locator()+geom_path()
## End(Not run)
```

---

**kml_to_sp**  
* Convert kml to polygon

---

**Description**

Create an sp SpatialPolygon or SpatialPolygonDataFrame object from a downloaded KML file and data file.

**Usage**

```r
kml_to_sp(map_df, df = NULL, crs_string = "+proj=longlat +datum=WGS84")
```

**Arguments**

- `map_df`: a map dataframe generated from `create_map_df()`.
- `df`: a climate dataframe with one piece of data to be mapped to each unique spatial polygon.
- `crs_string`: Coordinate reference string to use in spatial projection. Default is WGS84.

**Details**

If a dataframe is included, a spatial polygon dataframe object is created. The dataframe must have one unique piece of information per polygon, otherwise an error will be thrown. However just a basic spatial polygon will be created if no dataframe is included.

**Value**

a SpatialPolygon object

**Examples**

```r
## Not run:
sa_map <- create_map_df(locator=SoAm_country)
sa_dat <- get_ensemble_temp(SoAm_country,"annualanom",2080,2100)
sa_dat <- subset(sa_dat,sa_dat$scenario == "a2")
sa_dat <- subset(sa_dat,sa_dat$percentile == 50)
sa_poly <- kml_to_sp(sa_map,df = sa_dat)
### colors are a bit off, but just to verify that data is
spplot(sa_poly,"data")

## End(Not run)
```
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NoAm_basin</td>
<td>Basin codes for NoAm, used in downloading maps</td>
</tr>
<tr>
<td>NoAm_country</td>
<td>Country codes for all of NoAm</td>
</tr>
<tr>
<td>Oceana_basin</td>
<td>Basin codes for Oceana, used in downloading maps</td>
</tr>
<tr>
<td>Oceana_country</td>
<td>Country codes for all of Oceana</td>
</tr>
<tr>
<td>rWBclimate</td>
<td>rWBclimate</td>
</tr>
<tr>
<td>SoAm_basin</td>
<td>Basin codes for SoAm, used in downloading maps</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------</td>
</tr>
</tbody>
</table>

**Description**

Basin codes for SoAm, used in downloading maps

<table>
<thead>
<tr>
<th>SoAm_country</th>
<th>Country codes for all of SoAm</th>
</tr>
</thead>
</table>

**Description**

Country codes for all of SoAm
Index

*Topic datasets
- Africa_basin, 2
- Africa_country, 2
- Asia_basin, 3
- Asia_country, 3
- codes, 5
- Eur_basin, 7
- Eur_country, 7
- NoAm_basin, 20
- NoAm_country, 20
- Oceana_basin, 20
- Oceana_country, 20
- SoAm_basin, 21
- SoAm_country, 21
- get_historical_precip, 15
- get_historical_temp, 16
- get_model_precip, 17
- get_model_temp, 18
- kml_to_sp, 19
- NoAm_basin, 20
- NoAm_country, 20
- Oceana_basin, 20
- Oceana_country, 20
- rWBclimate, 20
- rWBclimate-package (rWBclimate), 20
- SoAm_basin, 21
- SoAm_country, 21

Africabasin, 2
Africa_country, 2
Asia_basin, 3
Asia_country, 3
check_iso_code, 3
check_locator, 4
climate_map, 4
codes, 5
create_map_df, 5
date_correct, 6
download_kml, 6

Eur_basin, 7
Eur_country, 7
get_climate_data, 7
get_data_recursive, 8
get_ensemble_climate_data, 9
get_ensemble_data_recursive, 9
get_ensemble_precip, 10
get_ensemble_stats, 11
get_ensemble_temp, 12
get_historical_data, 13
get_historical_data_recursive, 14