Package 'rWBclimate'

July 2, 2014

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` *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

```
check_ISO_code(iso)
```

Arguments

iso	The 3 letter country code based on ISO3 Country abbreviations (http://unstats.un.org/unsd/methods/m49/m

Value

TRUE if a valid code, otherwise an error is returned

Examples

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

## End(Not run)
```

check_locator *Checks for what kind of locator a user input*

Description

Checks for what kind of locator a user input

Usage

```
check_locator(locator)
```

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]

Value

geo_ref a string indicating what kind of geography to use in the api

climate_map *Map climate data*

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.

Usage

```
climate_map(map_df, data_df, return_map = TRUE)
```

Arguments

map_df	a map dataframe generated from create_map_df()
data_df	a climate dataframe with one piece of data to be mapped to each unique spatial polygon.
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.

Value

Either a ggplot2 map or a dataframe depending on the parameter return_map

Examples

```
## 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 data frame 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`	The a vector of ISO3 country code's that you want data about. (http://unstats.un.org/unsd/methods/m49/m 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 (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

```
## 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/cl
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 in the number of periods in the data api

Examples

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

## End(Not run)
```

`download_kml` *Download kml files*

Description

Downloads map data from in 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)
```

Arguments

locator	The a vector of ISO3 country code's that you want data about. (http://unstats.un.org/unsd/methods/m49/m 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 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.

---

get_data_recursive *wratpper 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 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:
 get_ensemble_data_recursive(c("1","2"),"basin","mavg","pr",1920,1940)

## End(Not run)
```

get_ensemble_climate_data *Download 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 requets can be for countries or basins.

Usage

```
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 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.

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

```
get_ensemble_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 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:
 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

```
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 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

```
## Not run:
# Get data for 2 basins, annual average precipitation for all valid time periods
# then subset them, and plot
precip_dat <- get_ensemble_precip(c("2","231"),"annualavg",1900,3000)
precip_dat <- subset(precip_dat,precip_dat$scenario!="b1")
precip_dat$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)
precip_dat <- subset(precip_dat,precip_dat$scenario!="b1")
precip_dat$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:

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

ppt_days90th	Number of days with precipitation greater than the control periods 90th percentile	days
ppt_dryspell	Average number of days between precipitation events	days
ppt_means	Average daily precipitation	mm

Usage

```
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

```
## 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

```
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 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

```
## 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)
```

---

get_historical_data *Download historical climate 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

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 *Download historical precipitation 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

```
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) 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

```
## 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 precipitaion")

## 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 precipitaion")


## 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

```
get_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 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 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 *Download GCM precipitation data*

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 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

```
## 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 *Download GCM temperature data*

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 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

```
## 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,
```

```
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

```
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 WSGS84.

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

```
## 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)
```

`NoAm_basin` *Basin codes for NoAm, used in downloading maps*

Description

Basin codes for NoAm, used in downloading maps

`NoAm_country` *Country codes for all of NoAm*

Description

Country codes for all of NoAm

`Oceana_basin` *Basin codes for Oceana, used in downloading maps*

Description

Basin codes for Oceana, used in downloading maps

`Oceana_country` *Country codes for all of Oceana*

Description

Country codes for all of Oceana

`rWBclimate` *rWBclimate*

Description

rWBclimate

SoAm_basin *Basin codes for SoAm, used in downloading maps*

Description

Basin codes for SoAm, used in downloading maps

SoAm_country *Country codes for all of SoAm*

Description

Country codes for all of SoAm

Index