Package 'coorClimR'

June 21, 2016

Title Get Gridded Climate Data at Individual Space-Time Locations

Version 0.0.0.9000

Description This project is an R-wrapper around an API that stores gridded climate data and its metadata in a postgres database. This r-package allows users to get climate data for a single point and return the results as a simple data frame, making visualization and analysis easy. This package integrates with the Neotoma R Package and the VertNet R Package to support existing workflows.

Depends R (>= 3.2.0),

License MIT

Encoding UTF-8

LazyData true

RoxygenNote 5.0.1.9000

Imports jsonlite, rvertnet

R topics documented:

`checkNumeric` *Check if a number is numeric*

Description

Check if a number is numeric

Usage

```
checkNumeric(x)
```

convertNeotomaSDToDF *Get Neotoma Occurrence Data Function uses the Neotoma API SampleData endpoint*

Description

Get Neotoma Occurrence Data Function uses the Neotoma API SampleData endpoint

Usage

```
convertNeotomaSDToDF(taxonname, ageold = "", ageyoung = "", loc = "",
  gpid = "", altmin = "", altmax = "")
```

Arguments

taxonname	String The name of the taxonomic grouping that you wish to query Neotoma for. Matches a taxon in the neotoma database
ageold	Integer Oldest age, as calendar years before present, to include in results from neotoma.
ageyoung	Integer Youngest age, as calendar years before present, to include in results.
loc	A list of the form longitudeWest, latitudeSouth, longitudeEast, latitudeNorth that represents a bounding box in which to search for occurrences in Neotoma
gpid	Integer Limit occurrences to a geopolitical entity id. Valid values provided by GeoPoliticalUnits database table in Neotoma.
altmin	Integer Minimum site altitude in meters.
altmax	Integer Maximum site altitude in meters.

Value

data.frame. Columns: sampleID (actually a Neotoma dataset ID), Latitude, Longitude, Age

Examples

```
convertNeotomaSDToDF("bison bison")
convertNeotomaSDToDF("picea")
```

convertVertnettoDF *Use Vertnet API to get data, and return only specific columns needed.*

Description

Use Vertnet API to get data, and return only specific columns needed.

Usage

```
convertVertnettoDF(taxonname, genus = "", species = "", state = "",
  limit = "")
```

Arguments

taxonname	string: Name(s) of the taxonomic grouping that you wish to query Vertnet for.
genus	string: Target genus name(s).
species	string: Target species name(s).
state	string: Target state name(s).
limit	numeric: Number of results that you would like to accept. Defaults to 10000 if left empty.

Value

output data.frame: Lat, Lon, and Age data.

Examples

```
convertVertnettoDF("bison")
convertVertnettoDF("(kansas state OR KSU)", limit = 200)
convertVertnettoDF(genus = "mustela", species = "(nivalis OR erminea)")
```

---

getData *Get Climate Data*

Description

Get Climate Data

Usage

```
getData(x, y = "", t = "", producer = "", model = "",
  modelVersion = "", variableType = "", variableUnits = "",
  variablePeriod = "", variablePeriodType = "", averagingPeriod = "",
  averagingPeriodType = "", resolution = "", sampleID = "",
  siteName = "", siteID = "", verbose = T)
```

Arguments

x	A dataframe with minimum columns names: Latitude, Longitude, Age OR a double precision longitude value. If x is a dataframe, it can also accept columns siteName, sampleID, siteID to preserve object identification through the api call process
y	A double precision latitude coordinate
t	A double precision or integer number representing years before 1950, can be negative to represent time since 1950 API filter parameters
model	A string specifying the model from which the climate data was created. Default = "" (all)
modelVersion	A string representing the model version that produced the output. Default = "" (all)
variableType	A string representing the type of variable to return. Default = "" (all)

variableUnits A string representing the units in which the variables are measures. Default = "" (all)

variablePeriod An integer representing the measuring period for the variable. Default = "" (all)

variablePeriodType A string representing the type of measuring period for the variable. Default = "" (all)

averagingPeriodType A string representing the type of period over which the data has been averaged. Default = "" (all)

resolution A double precision value representing the native resolution at which the climate data was produced and stored. Default = "" (all)

sampleID A string or integer identifier for the sample at the space-time location. Default = "" (none)

siteName A string or integer identifier for the site at the x-y location of the sample. Default = "" (none)

siteID A string or integer identifier for the site at the x-y location of the sample. Default = "" (none)

averaingPeriod An integer representing the period of which the data has been averaged. Default= "" (all)

Value

Outputs a data.frame representation of the api response. Columns: From Database: variableunits, variablePeriodType, VariableType, variableID, Producer, sourceID, ModelVersion, tableName, VariableDescription, averagingPeriod, averagingPeriodType, Model, tableName Optional Identifiers: siteName, sampleID, siteID Response Values: value, latitude, longitude, yearsBP

Examples

```
## Create some data
t <- rbind(c(1, -122, 37, 1000), c(2, -100, 38, 1000))
t <- data.frame(t)
names(t) <- c("sampleID", "Longitude", "Latitude", "Age")
getData(t)
```

---

getDataRow *Get climate data for a single space-time coordnate*

Description

Get climate data for a single space-time coordnate

Usage

```
getDataRow(x, y, t, producer = "", model = "", modelVersion = "",
  variableType = "", variableUnits = "", variablePeriod = "",
  variablePeriodType = "", averagingPeriod = "", averagingPeriodType = "",
  resolution = "", siteID = "", siteName = "", sampleID = "",
  verbose = F)
```

Arguments

x	A double precision longitude value.
t	A double precision or integer number representing years before 1950, can be negative to represent time since 1950 API filter parameters
model	A string specifying the model from which the climate data was created. Default = "" (all)
modelVersion	A string representing the model version that produced the output. Default = "" (all)
variableType	A string representing the type of variable to return. Default = "" (all)
variableUnits	A string representing the units in which the variables are measures. Default = "" (all)
variablePeriod	An integer representing the measuring period for the variable. Default = "" (all)
variablePeriodType	A string representing the type of measuring period for the variable. Default = "" (all)
averagingPeriodType	A string representing the type of period over which the data has been averaged. Default = "" (all)
resolution	A double precision value representing the native resolution at which the climate data was produced and stored. Default = "" (all)
siteID	A string or integer identifier for the site at the x-y location of the sample. Default = "" (none)
siteName	A string or integer identifier for the site at the x-y location of the sample. Default = "" (none)
sampleID	A string or integer identifier for the sample at the space-time location. Default = "" (none)
averaingPeriod	An integer representing the period of which the data has been averaged. Default= "" (all)

Value

Outputs a data.frame representation of the api response. Columns: From Database: variableunits, variablePeriodType, VariableType, variableID, Producer, sourceID, ModelVersion, tableName, VariableDescription, averagingPeriod, averagingPeriodType, Model, tableName Optional Identifiers: siteName, sampleID, siteID Response Values: value, latitude, longitude, yearsBP

Examples

```
getData(-122, 37, 16000, siteName="mySite", variableType="Maximum Temperature")
getData(-90, 40, 900, siteName="myOtherSite", sampleID=1291, variableType="Precipitation")
```

makeScatterPlot *Scatter two environmental variables against each other*

Description

Scatter two environmental variables against each other

Usage

```
makeScatterPlot(climateDF, xVariable = "Precipitation",
  yVariable = "Maximum Temperature", xPeriod = 1, yPeriod = 1,
  pointColor = "gray", plotModern = TRUE, modernColor = "black",
  title = paste(xVariable, xPeriod, "vs.\n", yVariable, yPeriod))
```

Arguments

climateDF	A data.frame object produced by getData that has the climate data you wish to plot
xVariable	Name of variable type to put on the x axis
yVariable	Name of variable type to put on the y axis
xPeriod	The measurement period of the variable for the X axis of the plot
yPeriod	The measurement period of the variable for the T axis of the plot
pointColor	Color string in which to plot the xy points
plotModern	boolean flag indicating whether to plot the modern points in a different color than the background points
modernColor	if plotModern is TRUE, then the modern points will be plotted in this color
title	The title to give the plots

Value

VOID

Examples

```
d <- queryNeotoma("ilex")
makeScatterPlot(d, xVariable='Maximum Temperature', yVariable='Minimum Temperature', xPeriod=7, yPeriod=1, m
```

makeTSPlot *Plot a climate variable through time*

Description

Plot a climate variable through time

Usage

```
makeTSPlot(climateDF, responseVariable = "Precipitation",
  responsePeriod = 1, title = paste(responseVariable, responsePeriod,
  "vs. Time"), plotAVG = T, plotSD = TRUE, pointColor = "gray",
  lineColor = "red")
```

Arguments

climateDF	A data.frame object produced by getData that has the climate data you wish to plot
responseVariable	The name of the variable for the Y axis of the plot
responsePeriod	The measurement period of the variable for the Y axis of the plot
title	The title you want to put on the plot
plotAVG	Boolean flag for plotting the mean of each yearsBP bin (binned average)
plotSD	Boolean flag for plotting the standard deviation of year yearsBP bin(binned SD)
pointColor	Color string representing the color in which to plot the individual points
lineColor	If plotAVG and/or plotSD are true, then those lines will be plotted in this color

Value

data.frame with the binned averages, std, and median for the dataset

Examples

```
sequoia <- queryNeotoma("sequoia")
makeTSPlot(sequoia, lineColor="green")
```

---

queryNeotoma *Get Climate Data for Neotoma Occurrences*

Description

Get Climate Data for Neotoma Occurrences

Usage

```
queryNeotoma(taxonname, ageold = "", ageyoung = "", loc = "", gpid = "",
  altmin = "", altmax = "", producer = "", model = "",
  modelVersion = "", variableType = "", variableUnits = "",
  variablePeriod = "", variablePeriodType = "", averagingPeriod = "",
  averagingPeriodType = "", resolution = "")
```

Arguments

taxonname	String The name of the taxonomic grouping that you wish tok query neotoma for. Matches a taxon in the neotoma database
ageold	Integer Oldest age, as calendar years before present, to include in results from neotoma.
ageyoung	Integer Youngest age, as calendar years before present, to include in results.
loc	A list of the form longitudeWest, latitudeSouth, longitudeEast, latitudeNorth that represents a bounding box in which to search for occurrences in Neotoma
gpid	Integer Limit occurrences to a geopolitical entity id. Valid values provided by GeoPoliticalUnits database table in Neotoma.
altmin	Integer Minimum site altitude in meters.
altmax	Integer Maximum site altitude in meters.
model	A string specifying the model from which the climate data was created. Default = "" (all)
modelVersion	A string representing the model version that produced the output. Default = "" (all)
variableType	A string representing the type of variable to return. Default = "" (all)
variableUnits	A string representing the units in which the variables are measures. Default = "" (all)
variablePeriod	An integer representing the measuring period for the variable. Default = "" (all)
variablePeriodType	A string representing the type of measuring period for the variable. Default = "" (all)
averagingPeriodType	A string representing the type of period over which the data has been averaged. Default = "" (all)
resolution	A double precision value representing the native resolution at which the climate data was produced and stored. Default = "" (all)
averaingPeriod	An integer representing the period of which the data has been averaged. Default= "" (all)

Value

Outputs a data.frame representation of the api response. Columns: From Database: variableunits, variablePeriodType, VariableType, variableID, Producer, sourceID, ModelVersion, tableName, VariableDescription, averagingPeriod, averagingPeriodType, Model, tableName Optional Identifiers: siteName, sampleID, siteID Response Values: value, latitude, longitude, yearsBP

Examples

```
queryNeotoma("bison bison")
queryNeotoma("sedum", altmax=1500, variablePeriod=1, resolution=0.5)
```

queryVertnet *Get climate data for Vertnet occurrences.*

Description

Get climate data for Vertnet occurrences.

Usage

```
queryVertnet(taxonname, genus = "", species = "", state = "",
  limit = "")
```

Arguments

taxonname	string: Name(s) of the taxonomic grouping that you wish to query Vertnet for.
genus	string: Target genus name(s).
species	string: Target species name(s).
state	string: Target state name(s).
limit	numeric: Number of results that you would like to accept. Defaults to 10000 if left empty.

Value

output data.frame: The data which you seek.

Examples

```
queryVertnet("bison")
queryVertnet("(kansas state OR KSU)", limit = 200)
queryVertnet(genus = "mustela", species = "(nivalis OR erminea)")
```

Index