Example of script data weather

Juan Camilo Rivera

May 3, 2018

It is recommend to read this example with user guide manual. This code was made under version of R 3.4.4

The data bases are from meterological stations of Chiapas (México). Each station has five variables; precipitation (P), solar radiation (SR), relative humidity (RH), maximum temperature (TX) and minimun temperature (TM). These data are hourly. This data is in folder DATA of repository (https://github.com/j-river1/Weather_Code_v.1.0).

The inputs are text files with Date and Value as columns (see user guide manual, UGM).

Libraries and source

Install and load libraries. The **All_Funcions.R** is a source, this script has all functions used in the code.

```
#Libraries
libraries <- c("tsoutliers","roxygen2","rgeos","dismo","ggdendro",
               "geosphere", "rgdal", "sp", "here",
               "geosphere", "data.table", "RCurl","randomForest",
               "ggplot2","RMAWGEN","stringr", "zoo",
               "abind", "plyr", "imputeTS","reshape2","parallel",
               "maptools","stringr","dplyr")

#Install packages
list.of.packages <- libraries
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages)

#Load libraries
lapply(list.of.packages, require, character.only = TRUE)

#Load functions
source("All_Functions.R")
```

Folders

These folders will save all results and information about process of cleaning data and missing input values. There are 11 folders and 2 subfolders. The original data has to be stored in Original_Data folder according to specific requirements (see UGM).

```
#**The setwd() must be directory where are the source above.
#Create folders
mainDir <- getwd()
create_folders(mainDir)
#**Put into the folder Original_Data all weaher files data.
```

Input variables

There are two data.frames for input variables, Daily_restric or Hourly_restric (dependig on type of recording data, hourly or daily) and variables. In this case, data is daily so Daily_restric is filled and the code saves in *Daily_Restrictions.csv*.

The variables data.frame has to be filled according information of stations. (see UGM)

```
#Daily Restrictions as data frame
Variables <- c("Vmin", "Vmax")
TX <- c(48,0)
TM <- c(48,-10)
#Calories per centimeter. Multiply 12*0.0858
SR <- c(1029,0)
RH <- c(100,0)


#Variables
#Choose Time Data
#If the time is in terms of hours so Hourly_Daily  = 1
#If the time is in terms of days so Hourly_Daily  = 2

Hourly_Daily <- 2
Start_date <- c("2005-1-1")
End_date <- c("2012-12-31")
Percentage <- 0.7
separt <- ""
date_format <- "%Y-%m-%d"
dist_Station <- 20000
```

Spatial Information Stations

In this line of code, the code makes a excel file (**Information_Spatial_Stations.xls**) with names, latitude, longititude and altitude of each station. The file is in the folder **SpatialInformation_InputVariables** and user must fill it. In this example, the information of stations is in the folder "poner el nombe del folder"

```
#Information Spatial information of stations. Longitude and Latitude.
Spatial_Information()

print("Update longitude and latitude in the file Information_Spatial_Stations")
```

Hourly control and clustering stations

The code indentifies the outliers and graphs them. The graphs are in the folder **Outliers**.

Note that for input missing values the user must choose stations. Generally, the stations more closer could be join for input values (not ALWAYS). For this reason, the code makes a clustering according its longitude, latitude. altitude. The result of clustering is folder **Results** and there is graph in the folder **Graphics/Clustering_Stations**. In this example, the stations ChiapasAlpujarras and ChiapasCampoExperimentalRosarioIzapa are in same cluster and the others clusters has one station.

```
#Control hourly or daily.
controlHourlyDaily(type = variables$Time_Type)
```

```
#Stations for Rmwagen
choose_stations()
```

Input missing values

Rmwagen

For choosing the stations, the user has to see Resuls_DailyControl.csv file for knowing date of data for putting in the variable Start_date and End_date(see above Input variables). For example, the ChiapasAgroipsa starts 20013 and ends 2017 so code shows a warning message ** Error in name_station[[i]] : subscripts out of bounds** so these have to change by Start_date <- c("2009-1-1") and End_date <- c("2017-12-31").

```
#File with format for using  Rmwagen
put_rmawgenformat(list.files(here::here("AfterDailyControl_Data")), 'TX')
put_rmawgenformat(list.files(here::here("AfterDailyControl_Data")), 'TM')
put_rmawgenformat(list.files(here::here("AfterDailyControl_Data")), 'P')

#Using Rmwagen

station <- c("ChiapasAlpujarras", "ChiapasCampoExperimentalRosarioIzapa")
graph_all("TEMPERATURE_MAX", 'Temperatura_Maxima', station)
graph_all("TEMPERATURE_MIN", 'Temperatura_Minima', station)
graph_all("PRECIPITATION", 'Precipitacion', station)
```

Random Forest

The code puts missing values using random forest for humility relative and radiation solar.

```
#Moving and merge files
#Moving ALL files to Files_By_Station
move_files_SR_HR()
match_files(type="RandomForest")


#Using Random forest
#setwd("../../Randomforest/")
graph_all_SR_RH(list.files(path = "./Randomforest/", pattern = "\\.txt$"), "Humedad_Relativa")
graph_all_SR_RH(list.files(path = "./Randomforest/", pattern = "\\.txt$"), "Radiacion_Solar")
```

Final Data

The results are in the folder **Results** in text files with name station.

```
#Change final file to folder
match_files(type="Final_Data")


#Moving final data
read_files(list.files(pattern=".txt"))
```