ΔΙΑΧΕΙΡΙΣΗ ΥΔΑΤΙΚΩΝ ΠΟΡΩΝ - ΕΡΓΑΣΤΗΡΙΟ

Το πακέτο hydroTSM της R

Σήμερα θα χρησιμοποιήσουμε το πακέτο hydroTSM με δικά μας δεδομένα, τα οποία προέρχονται από υπάρχουσες βάσεις δεδομένων.

Για παράδειγμα θα φτιάξουμε δικά μας δεδομένα με τυχαία δειγματοληψία από πληθυσμό με συγκεκριμένα χαρακτηριστικά κατανομής πυκνότητας πιθανότητας.

Επίσης δεδομένα από την βάση κλιματικών δεδομένων του NOAA
https://www.ncdc.noaa.gov/cdo-web/search

και τέλος δεδομένα από την βάση δεδομένων Giovanni της NASA.

Θα χρησιμοποιήσουμε δύο είδη αντικειμένων χρονοσειρών: ts & zoo.
Επίσης θα δούμε και το αντικείμενο zooreg.

Ft: Παράγοντας Τάσης (Trend Strength)

$$F_T = \max\left(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(T_t + R_t)}\right)$$

Fs: Παράγοντας Εποχικότητας (Seasonal Strength)

$$F_S = \max\left(0, 1 - \frac{\text{Var}(R_t)}{\text{Var}(S_t + R_t)}\right)$$

Οι παράγοντες αυτοί διακυμαίνονται μεταξύ 0 και 1, όπου 1 σημαίνει ότι υπάρχει πολύ ισχυρή τάση και πολύ ισχυρή εποχικότητα στην χρονοσειρά.

```r
#Set the working directory
setwd("C:/Users/user/OneDrive/Notes/Xanthi/ΔιαχείρισηΥδατικώνΠόρων")

packages = c("hydroTSM", "plyr", "ggfortify",
            "forecast", "tsutils", "ggplot2")

install.packages(packages)

library(hydroTSM)
library(plyr)
library(ggfortify)
library(forecast)
library(tsutils)
library(ggplot2)


#Read a csv file as a zoo object
z <- read.zoo("data.csv",
             header = TRUE,
             sep = ",")
```

```
#make an artificial dataset
#based on the best fitting probability density function
test = rweibull(365*5,
	shape=1.440047,
	scale = 124.590789)

test.ts = ts(test, start = c(2010,1,1), frequency = 365)

#Check the time-series
index(test.ts)
start(test.ts)
end(test.ts)
frequency(test.ts)

#plot the daily precipitation time-series
plot(test.ts)
```

```
#To shift the data forward one day, we use k = +1
#Lag the data by +1
lag(test.ts, k = +1, na.pad = TRUE)
#To shift the data backward one day, we use k = -1.
lag(test.ts, k = -1, na.pad = TRUE)
#The function is called lag, but a positive k
#actually generates leading data, not lagging data.

#Decomposition of daily time series using stl()
#be careful to use the ts object only
decomp <- stl(test.ts, s.window = "per")

#Plot decomposition
plot(decomp)
Tt <- trendcycle(decomp)
St <- seasonal(decomp)
Rt <- remainder(decomp)
#Trend Strength Calculation
Ft <- round(max(0,1 - (var(Rt)/var(Tt + Rt))),1)
#Seasonal Strength Calculation
Fs <- round(max(0,1 - (var(Rt)/var(St + Rt))),1)
```

```
#Store trend and seasonality into a dataframe
results = data.frame('Trend Strength' = Ft , 'Seasonal Strength' = Fs)


#Transform a ts object into zoo object
dt <- seq(as.Date("2010-01-01"), as.Date("2015-01-01"), by = "days")
test.zoo = as.zoo(test.ts)
test.zoo = zoo(test.zoo,dt)

#Plot data with the hydroplot function
hydroplot(test.zoo,
        var.type="Precipitation",
        main="Randomized Data",
        pfreq = "dm")

#Get data summary
smry(test.zoo)

#compute and plot the moving average series
test.zoo.ma <- rollmean(test.zoo, 31)
test.ma <- merge(test.zoo, test.zoo.ma)
plot(test.ma)
```

```
#Transform daily into monthly values
test.zoo.m <- daily2monthly(test.zoo, FUN=sum)

#apply a linear model on monthly time-series
linear.model <- lm(coredata(test.zoo.m) ~ index(test.zoo.m))

#create a time-series from the residuals of the linear model
detr <- zoo(resid(linear.model), index(test.zoo.m))
autoplot(detr)


# Creating a matrix with monthly values per year in each column
M <- matrix(test.zoo.m, ncol=12, byrow=TRUE)

colnames(M) <- month.abb
rownames(M) <- unique(format(time(test.zoo.m), "%Y"))

# Plotting the monthly precipitation values
print(matrixplot(M,
        ColorRamp="Precipitation",
        main="Randomized Monthly precipitation, mm"))
```

```r
#Transform daily into yearly values
test.zoo.yr = daily2annual(test.zoo, FUN=sum, na.rm=TRUE)

#Plot annual time-series
barplot(test.zoo.yr,
        xlab = "Years",
        ylab = "Precipitation, mm")

#Get Years from time-series
yr = as.numeric(format(index(test.zoo.yr),"%Y"))

#plot again with year-values in x-axis
barplot(test.zoo.yr,
        yr,
        xlab = "Years",
        ylab = "Precipitation, mm")


#Compute annual mean value
test.zoo.yr.mean = sum(test.zoo.yr)/(length(test.zoo.yr)-1)
```

```
#Monthly data analysis
#Median of the monthly values of dataset x
monthlyfunction(test.zoo.m, FUN=median, na.rm=TRUE)
cmonth <- format(time(test.zoo.m), "%b")
months <- factor(cmonth,
                levels=unique(cmonth),
                ordered=TRUE)

#Create boxplot of monthly values
boxplot(coredata(test.zoo.m) ~ months,
        col="lightblue",
        main="Monthly Precipitation",
        ylab="Precipitation, [mm]",
        xlab="Month")

#Seasonal Analysis
#Compute the mean-seasonal values of precipitation
seasonalfunction(test.zoo.m, FUN=sum) / length(test.zoo.m)
```

```
#Extracting the seasonal values for each year
DJF <- dm2seasonal(test.zoo.m, season="DJF", FUN=sum)
MAM <- dm2seasonal(test.zoo.m, season="MAM", FUN=sum)
JJA <- dm2seasonal(test.zoo.m, season="JJA", FUN=sum)
SON <- dm2seasonal(test.zoo.m, season="SON", FUN=sum)

#Plot Seasonal Precipitation plots
par(mfrow = c(2,1))
plot(DJF,type="b",xlab = "Years")
plot(MAM,type="b",xlab = "Years")
```

```
#Read data from a file from NOAA Climate data portal
#https://www.ncdc.noaa.gov/cdo-web/search

new = read.csv("2515380.csv",
		skip=1,
		header=FALSE,
		sep=",",
		quote="")
head(new)
myColNames = c("code","name","state","lat","long","elev","date","prec")
names(new) <- myColNames

new.ts = ts(as.numeric(new$prec), frequency=365, start=c(2020,1,1))

#Check the time-series
index(new.ts)
start(new.ts)
end(new.ts)
frequency(new.ts)
```

```
#To shift the data forward one day, we use k = +1
#Lag the data by +1
lag(test.ts, k = +1, na.pad = TRUE)

#To shift the data backward one day, we use k = -1.
lag(new.ts, k = -1, na.pad = TRUE)

#The function is called lag, but a positive k
#actually generates leading data, not lagging data.


#Transform a ts object into zoo object
dt.new <- seq(as.Date("2020-01-01"), as.Date("2021-01-01"), by = "days")
new.zoo = as.zoo(new.ts)
new.zoo = zoo(new.zoo,dt.new)

#Plot data with the hydroplot function
hydroplot(new.zoo,
        var.type="Precipitation",
        main="Randomized Data",
        pfreq = "dm")
#Get data summary
smry(new.zoo)
```

```
#Download data from Giovanni nasa
#data are in *.csv and *.netcdf formats
packages1 = c("ncdf4", "raster", "rgdal", "ggplot2")
install.packages(packages1)

library(ncdf4) # package for netcdf manipulation
library(raster) # package for raster manipulation
library(rgdal) # package for geospatial analysis
library(ggplot2) # package for plotting

#Set the working directory
setwd("C:/Users/user/OneDrive/Notes/Xanthi/ΔιαχείρισηΥδατικώνΠόρων")
nc_data <- nc_open('g4.areaAvgTimeSeries.GPM_3IMERGDF_06_precipitationCal.20100101-
20201231.20E_30N_30E_40N.nc')

time <- ncvar_get(nc_data,"time")
tunits <- ncatt_get(nc_data,"time","units")
nt <- dim(time)
```

```
# get precipitation
dname= "GPM_3IMERGDF_06_precipitationCal"
prec_array <- ncvar_get(nc_data,dname)
dlname <- ncatt_get(nc_data,dname,"long_name")
dunits <- ncatt_get(nc_data,dname,"units")
fillvalue <- ncatt_get(nc_data,dname,"_FillValue")
dim(prec_array)
time = as.POSIXct(time, origin="1970-01-01")

par(mfrow=c(1,1))
plot(time,prec_array,type="l")

df = data.frame(time,prec_array)

#Transform the zoo object into ts object
#Transform a ts object into zoo object
prec_data.zoo = zoo(df$prec_array,df$time)
plot(prec_data.zoo)

#Plot data with the hydroplot function
hydroplot(prec_data.zoo,
        var.type="Precipitation",
        main="Randomized Data",
        pfreq = "dm")
```