Sample

June 27, 2023

1 Sample Solution

1.1 Read the data into the program

First, a file (here a CSV file is used) with the level data must be read in. In most cases, the data is stored in a different structure than in the sample files. The interpretation of the data must therefore be adapted. We read the CSV file directly here. Packages can be used to programme faster and more elegant solutions.

Since our CSV file uses the semicolon as a separator instead of the comma, the method `csv2()` must be used instead of `csv()`.

1.1.1 Function

Since the same steps have to be carried out again and again for different files, we first combine them in a function `prepareData`.

[1]:
```
# converts the data from the csv-file
library(lubridate, warn.conflicts = FALSE)  # only used to generate the␣
 ↪attribute time with the method dmy_hm()
prepareData <- function(mydata) {
    # loop over all rows
    for (i in 1:nrow(mydata)) {
        # Does the line start with a date?
        # here: contains a "."
        if (grepl("[.]", mydata[i,1])) {
            date <- mydata[i,1]
        }
        # Does the line start with a time?
        # here: contains a ":"
        if (grepl("[:]", mydata[i,1])) {
            str <- paste(date,mydata[i,1])
            # dmy_hm() is a method by the library lubridate
            time <- dmy_hm(paste(date,mydata[i,1]))
            mydata[i,1] <- time
            # Is there a value entered for the level?
            # here: there is no "X"
            if (!grepl("X",mydata[i,2])) {
                level <- mydata[i,2]
```

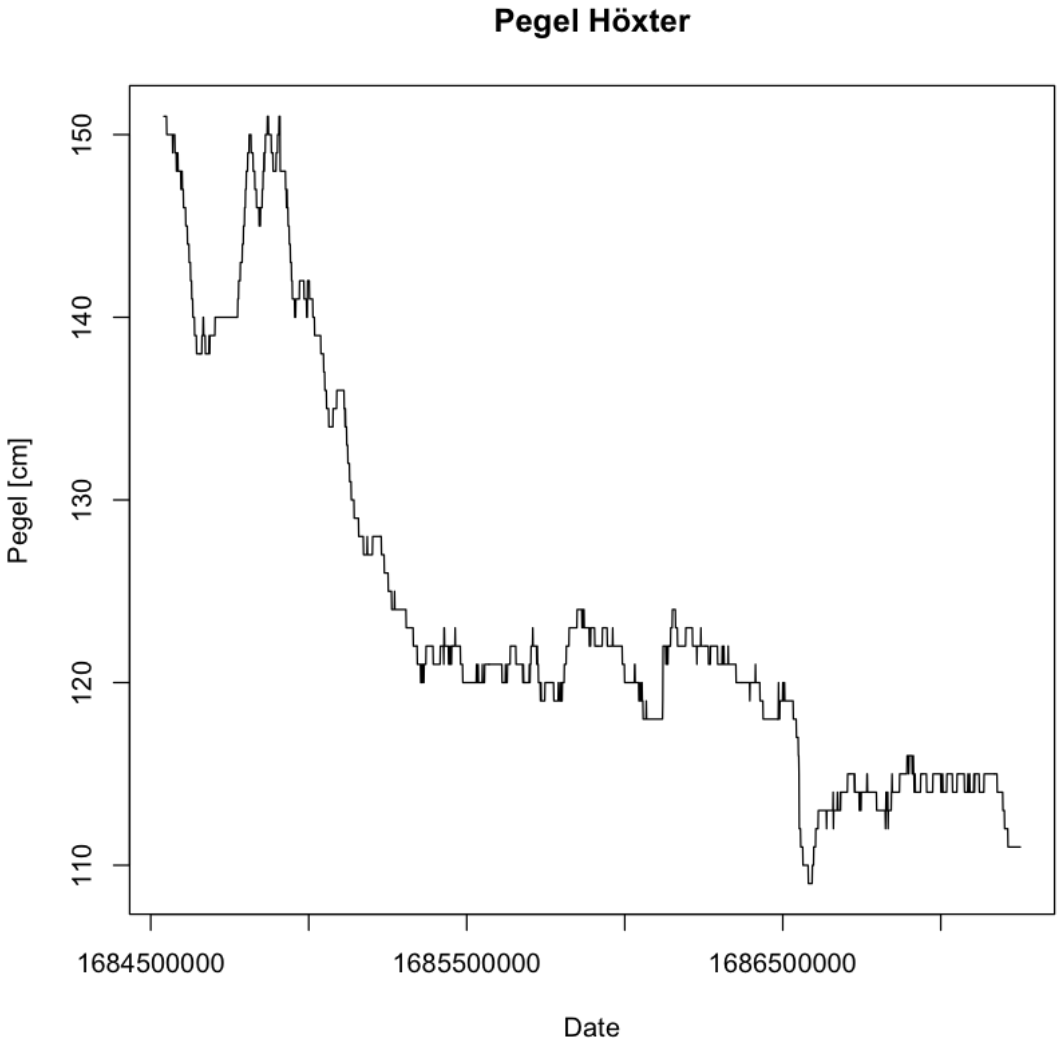
```
                mydata[i,3] <- "yes"
            } else {
                mydata[i,3] <- "no"
            }
        }
    }
    # only three columns
    data_filtered <- mydata[,c(1:3)]
    # only valid measured values
    data_selected <- data_filtered[mydata$V3 == "yes",]
    return (data_selected)
}
```

After we have defined the function, we can read in the CSV file and pass it to the function.

[2]:
```
data069 <- read.csv2("069_Hoexter.csv", header = FALSE, stringsAsFactors =␣
 ↪FALSE, na=NA)
data_selected <- prepareData(data069)
```

Now we can plot the first data set.

[3]:
```
# with this plot you do not have a normal date
plot(data_selected$V1,data_selected$V2,type="l", main="Pegel Höxter", xlab =␣
 ↪"Date",
     ylab = "Pegel [cm]")
```

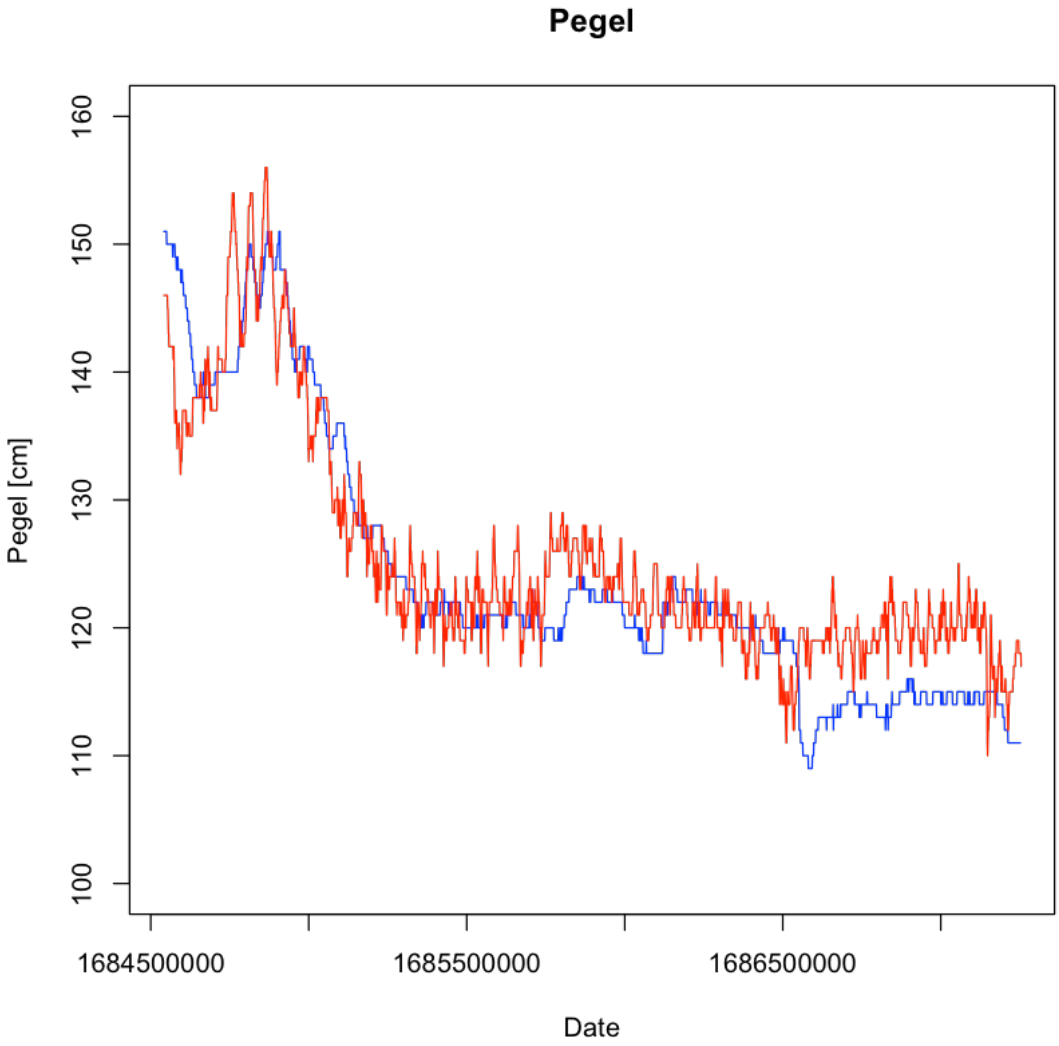


Similarly, we read in a second data set and pass it to the function.

[4]:
```
data000 <- read.csv2("000_Hann_Muenden.csv", header = FALSE, stringsAsFactors =␣
 ↪FALSE, na=NA)
data_sel_000 <- prepareData(data000)
```

Now two data sets have been read in. These are displayed in a graph. You can already see that the red data set is in front of the blue one. The red data was taken 69 km above the blue data.

[18]:
```
plot(data_selected$V1,data_selected$V2,type="l", main="Pegel" , xlab = "Date",
     ylab = "Pegel [cm]", col="blue", ylim=c(100,160))
lines(data_sel_000$V1, data_sel_000$V2, col="red")
```


Pegel
Pegel [cm]
100
110
120
130
140
150
160
1684500000
1685500000
1686500000
Date