R Lab. - Exercise 1

Michele Guadagnini - Mt. 1230663

April 15, 2020

Exercise 1 - Vectors and dataframes

Importing scottish lakes data:

```
Names <- c("Ness", "Lomond", "Morar", "Tay", "Awe", "Maree", "Ericht", "Lochy", "Rannoch",
           "Shiel", "Katrine", "Arkaig", "Shin")
Vols  <- c(7.45, 2.6, 2.3,  1.6,  1.2, 1.09, 1.08, 1.07, 0.97, 0.79, 0.77, 0.75, 0.35)#km3
Area  <- c(56,   71,  27,   26.4, 39,  28.6, 18.6, 16,   19,   19.5, 12.4, 16,   22.5)#km2
Lengs <- c(39,   36,  18.8, 23,   41,  20,   23,   16,   15.7, 28,   12.9, 19.3, 27.8)#km
Maxdp <- c(230,  190, 310,  150,  94,  114,  156,  162,  134,  128,  151,  109,  49  )#m
Meandp<- c(132,  37,  87,   60.6, 32,  38,   57.6, 70,   51,   40,   43.4, 46.5, 15.5)#m

scottish.lakes <- data.frame(Names, Vols, Area, Lengs, Maxdp, Meandp)
colnames(scottish.lakes) <- c("Name","Volume [km³]","Area [km²]","Length [km]",
                              "Max dp [m]","Mean dp [m]")
knitr::kable(scottish.lakes, caption="Scottish Lakes data frame")
```

Table 1: Scottish Lakes data frame

Name	Volume [km³]	Area [km²]	Length [km]	Max dp [m]	Mean dp [m]
Ness	7.45	56.0	39.0	230	132.0
Lomond	2.60	71.0	36.0	190	37.0
Morar	2.30	27.0	18.8	310	87.0
Tay	1.60	26.4	23.0	150	60.6
Awe	1.20	39.0	41.0	94	32.0
Maree	1.09	28.6	20.0	114	38.0
Ericht	1.08	18.6	23.0	156	57.6
Lochy	1.07	16.0	16.0	162	70.0
Rannoch	0.97	19.0	15.7	134	51.0
Shiel	0.79	19.5	28.0	128	40.0
Katrine	0.77	12.4	12.9	151	43.4
Arkaig	0.75	16.0	19.3	109	46.5
Shin	0.35	22.5	27.8	49	15.5

1) Evaluate the highest and lowest volume and area lake

```
maxvol <- max(scottish.lakes$Volume, na.rm=TRUE)
minvol <- min(scottish.lakes$Volume, na.rm=TRUE)
maxn <- scottish.lakes$Name[which.max(scottish.lakes$Volume)]
minn <- scottish.lakes$Name[which.min(scottish.lakes$Volume)]

message(paste("Maximum volume is", maxvol, "km³ of Loch", maxn,
              "\nMinimum volume is", minvol, "km³ of Loch", minn))
```

```
## Maximum volume is 7.45 km³ of Loch Ness
## Minimum volume is 0.35 km³ of Loch Shin
```

```
maxarea <- max(scottish.lakes$Area, na.rm=TRUE)
minarea <- min(scottish.lakes$Area, na.rm=TRUE)
maxn <- scottish.lakes$Name[which.max(scottish.lakes$Area)]
minn <- scottish.lakes$Name[which.min(scottish.lakes$Area)]

message(paste("Maximum area is", maxarea, "km² of Loch", maxn,
              "\nMinimum area is", minarea, "km² of Loch", minn))
```

```
## Maximum area is 71 km² of Loch Lomond
## Minimum area is 12.4 km² of Loch Katrine
```

2) Order the frame with respect to the area and determine the two largest area lakes

```
scottish.lakes <- scottish.lakes[order(-scottish.lakes$Area),]
lan <- scottish.lakes$Name[1:2]
la  <- c(scottish.lakes$Area[1:2])

message(paste("The lake with largest area is Loch", lan[1], "with", la[1], "km²" ,
              "\nThe lake with second largest area is Loch", lan[2], "with", la[2], "km²"))
```

```
## The lake with largest area is Loch Lomond with 71 km²
## The lake with second largest area is Loch Ness with 56 km²
```

3) By summing up the areas occupied by the lakes, determine the area of Scotland covered by water

```
WaterArea <- sum(scottish.lakes$Area)
message(paste("Total surface covered by water is", WaterArea, "km²"))
```

```
## Total surface covered by water is 372 km²
```

Exercise 2 - DAAG and Tibble

Importing needed packages:

```
#install.packages(c('DAAG','tibble'), type='source')
library(DAAG, tibble)
#library(help=DAAG)
library(tidyverse)
```

Loading the Australian athletes data frame:

```
data(ais)
#?ais
tbais <- tibble(ais)
knitr::kable(tbais[1:5,], caption="Australian athletes data frame")
```

Table 2: Australian athletes data frame

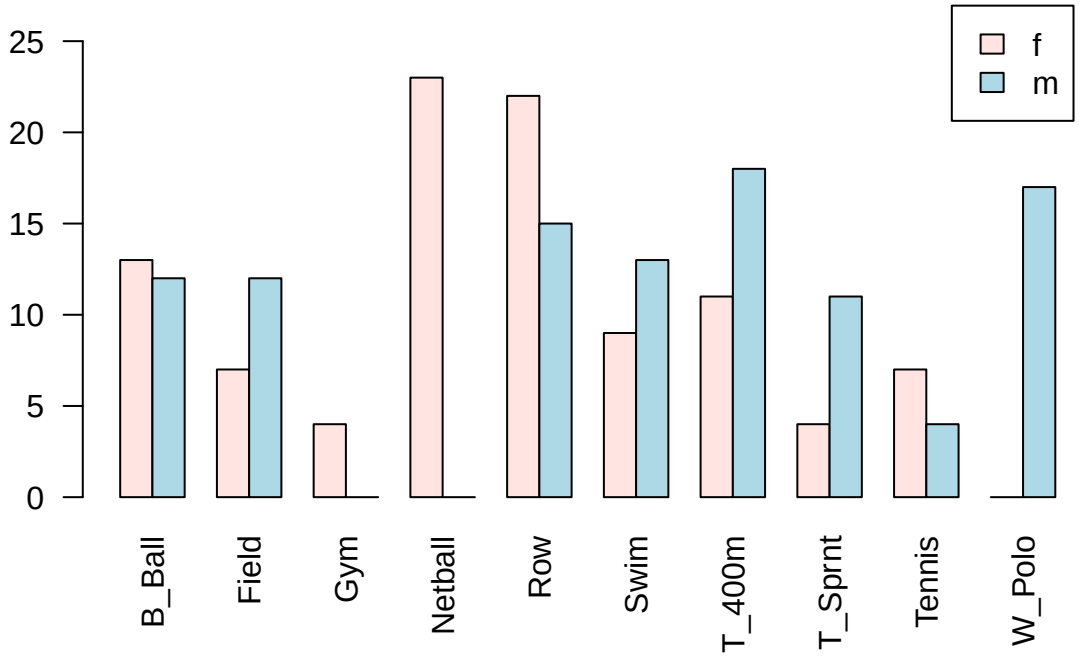
rcc	wcc	hc	hg	ferr	bmi	ssf	pcBfat	lbm	ht	wt	sex	sport
3.96	7.5	37.5	12.3	60	20.56	109.1	19.75	63.32	195.9	78.9	f	B_Ball
4.41	8.3	38.2	12.7	68	20.67	102.8	21.30	58.55	189.7	74.4	f	B_Ball
4.14	5.0	36.4	11.6	21	21.86	104.6	19.88	55.36	177.8	69.1	f	B_Ball
4.11	5.3	37.3	12.6	69	21.88	126.4	23.66	57.18	185.0	74.9	f	B_Ball
4.45	6.8	41.5	14.0	29	18.96	80.3	17.64	53.20	184.6	64.6	f	B_Ball

1) Create a table grouping the data by gender and by sport; produce a barplot with the table adding a legend

```
tbl <- table(tbais$sex, tbais$sport)
tbl
```

```
##
##     B_Ball Field Gym Netball Row Swim T_400m T_Sprnt Tennis W_Polo
##   f     13     7   4      23  22    9     11       4      7      0
##   m     12    12   0       0  15   13     18      11      4     17
```

```
maxY <- max(tbl)
barplot(tbl, beside=TRUE, col=c("mistyrose", "lightblue"),
        legend=rownames(tbl), las=2, ylim=c(0,maxY+5))
title( main="Australian athletes" )
```



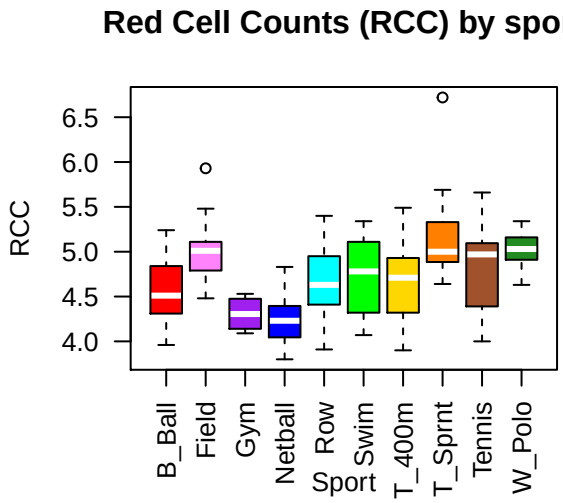
2) Determine if any of the columns holds missing values

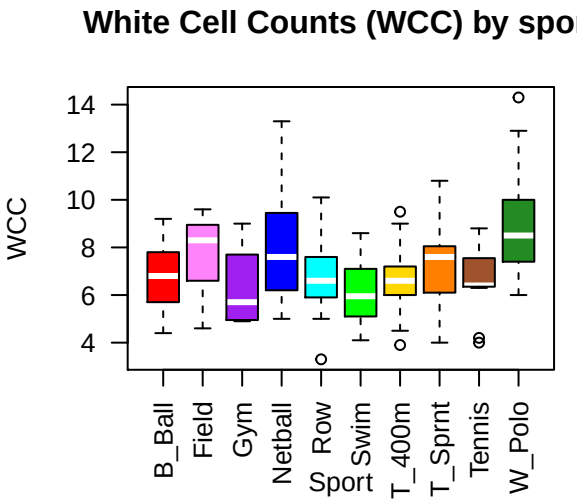
```
any(is.na(tbais))
```

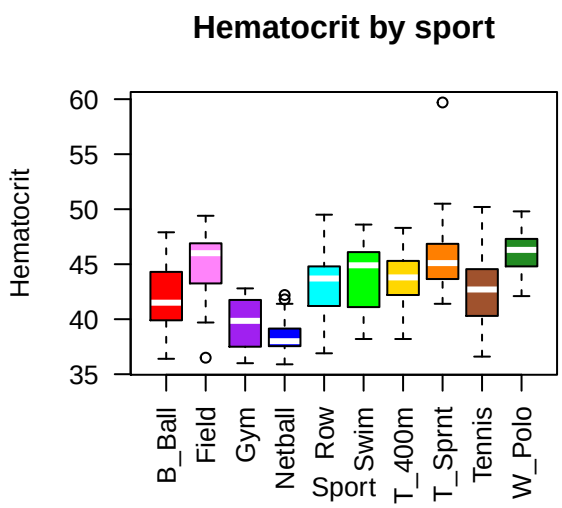
```
## [1] FALSE
```

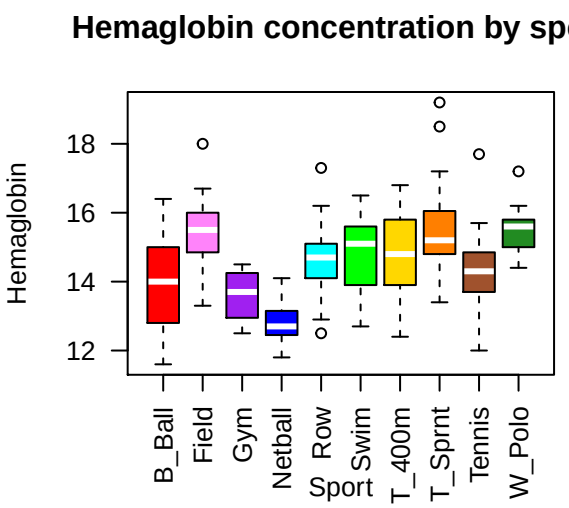
3) Produce boxplots of the main blood variables ('red blood cell counts', 'white blood cell counts', 'hematocrit' and 'hemaglobin concentration'), for different kind of sports

```
# custom colors list
cols <- c("red","orchid1","purple","blue","cyan","green","gold1",
          "darkorange1","sienna","forestgreen")

par(mfrow=c(2,2))
boxplot(rcc~sport, data=tbais, main='Red Cell Counts (RCC) by sport', xlab='Sport',
        ylab='RCC', col=cols, medcol='white', las=2)
boxplot(wcc~sport, data=tbais, main='White Cell Counts (WCC) by sport', xlab='Sport',
        ylab='WCC', col=cols, medcol='white', las=2)
boxplot(hc~sport, data=tbais, main='Hematocrit by sport', xlab='Sport',
        ylab='Hematocrit', col=cols, medcol='white', las=2)
boxplot(hg~sport, data=tbais, main='Hemaglobin concentration by sport', xlab='Sport',
        ylab='Hemaglobin', col=cols, medcol='white', las=2)
```



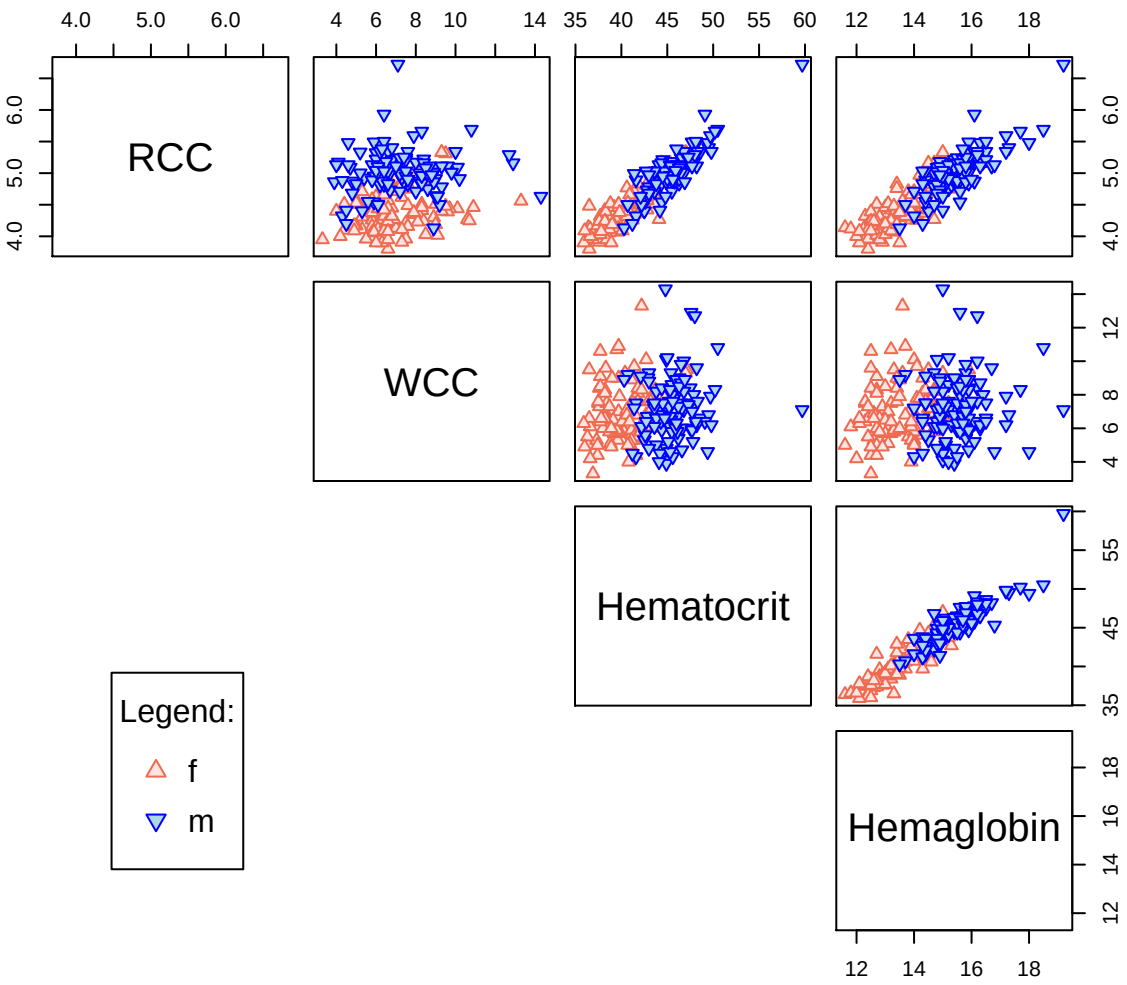






4) Make some scatter plot correlations of the same blood variables using different colors and symbols for the two genders in the sample

```
vars <- list(tbais$rcc, tbais$wcc, tbais$hc, tbais$hg)
keys <- list("RCC", "WCC", "Hematocrit", "Hemaglobin")

pairs(vars, keys, main="Correlations scatter plots matrix",
      col=ifelse(tbais$sex=='f', "coral2", "blue"),
      pch=ifelse(tbais$sex=='f', 24, 25),
      bg =ifelse(tbais$sex=='f', "mistyrose", "lightblue"),
      lower.panel=NULL,
     )
par(xpd=TRUE)
legend(0.1,0.3, legend=unique(tbais$sex),
       col = c("coral2", "blue"),
       pt.bg = c("mistyrose", "lightblue"),
       pch = c(24, 25),
       title="Legend:"
      )
```


Correlations scatter plots matrix
RCC
WCC
Hematocrit
Hemaglobin
Legend:
f
m

Exercise 3 - COVID-19

```
needed_packages <- c('lubridate', 'readxl', 'curl')
already_installed <- needed_packages %in% installed.packages()
for (pack in needed_packages [!already_installed]) {
    message(paste("To be installed : ", pack, sep = " "))
    install.packages(pack)
}
library(lubridate)
library(readxl)
library(curl)
url <- "https://www.ecdc.europa.eu/sites/default/files/documents/"
fname <- "COVID-19-geographic-disbtribution-worldwide-"
date <- lubridate::today() - 1
ext = ".xlsx"
target <- paste(url, fname, date, ext, sep = "")
message("target :", target)
tmp_file <- tempfile("data", "/tmp", fileext = ext)
tmp <- curl::curl_download(target, destfile = tmp_file)
covid <- readxl::read_xlsx(tmp_file)
```

1) Exploring the structure of the loaded data frame:

```
summary(covid)
```

```
##     dateRep                         day            month
##  Min.   :2019-12-31 00:00:00   Min.   : 1.00   Min.   : 1.000
##  1st Qu.:2020-02-09 00:00:00   1st Qu.: 7.00   1st Qu.: 2.000
##  Median :2020-03-17 00:00:00   Median :14.00   Median : 3.000
##  Mean   :2020-03-05 20:49:06   Mean   :15.09   Mean   : 2.753
##  3rd Qu.:2020-04-01 00:00:00   3rd Qu.:23.00   3rd Qu.: 4.000
##  Max.   :2020-04-14 00:00:00   Max.   :31.00   Max.   :12.000
##
##       year          cases             deaths
##  Min.   :2019   Min.   :   -9.0   Min.   :   0.00
##  1st Qu.:2020   1st Qu.:    0.0   1st Qu.:   0.00
##  Median :2020   Median :    1.0   Median :   0.00
##  Mean   :2020   Mean   :  174.4   Mean   :  11.06
##  3rd Qu.:2020   3rd Qu.:   17.0   3rd Qu.:   0.00
##  Max.   :2020   Max.   :35527.0   Max.   :2087.00
##
##  countriesAndTerritories    geoId           countryterritoryCode
##  Length:10742            Length:10742       Length:10742
##  Class :character        Class :character   Class :character
##  Mode  :character        Mode  :character   Mode  :character
##
##
##
##
##   popData2018
##  Min.   :1.000e+03
##  1st Qu.:3.170e+06
##  Median :1.028e+07
##  Mean   :6.078e+07
```

```
##  3rd Qu.:4.180e+07
##  Max.   :1.393e+09
##  NA's   :67
```

```
#excluding last column (popData2018) due to page size limit
knitr::kable(covid[order(-covid$year, -covid$month, -covid$day),][1:10,1:9],
             caption="Head of covid data frame")
```

Table 3: Head of covid data frame

dateRep	day	month	year	cases	deaths	countriesAndTerritories	geoId	countryterritoryCode
2020-04-14	14	4	2020	58	3	Afghanistan	AF	AFG
2020-04-14	14	4	2020	21	0	Albania	AL	ALB
2020-04-14	14	4	2020	69	20	Algeria	DZ	DZA
2020-04-14	14	4	2020	8	0	Andorra	AD	AND
2020-04-14	14	4	2020	0	0	Angola	AO	AGO
2020-04-14	14	4	2020	0	0	Anguilla	AI	NA
2020-04-14	14	4	2020	2	0	Antigua_and_Barbuda	AG	ATG
2020-04-14	14	4	2020	69	3	Argentina	AR	ARG
2020-04-14	14	4	2020	26	1	Armenia	AM	ARM
2020-04-14	14	4	2020	0	0	Aruba	AW	ABW

2) Selecting yesterday data with more cases or more deaths

```
yest <- Sys.Date()-1
covidYest <- covid[covid$dateRep==yest,]

covidYestNewCases <- covidYest[covidYest$cases > 200,]
x <- matrix( c(covidYestNewCases$cases, covidYestNewCases$deaths), ncol=2)
colnames(x) <- c("cases", "deaths")
rownames(x) <- covidYestNewCases$countriesAndTerritories
tblCases <- as.table(x)
tblCases
```

```
##                        cases deaths
## Bahrain                  225      1
## Belarus                  341      3
## Belgium                  942    303
## Brazil                  1261    105
## Canada                  1298     63
## Chile                    312      2
## France                  2673    574
## Germany                 2082    170
## India                   1211     31
## Indonesia                316     26
## Iran                    1617    111
## Ireland                  992     31
## Israel                   441     13
## Italy                   3153    564
## Japan                    390      7
## Kazakhstan               218      4
## Mexico                   353     36
## Netherlands              964     86
## Oman                     214      0
```

```
## Pakistan                   342      3
## Philippines                284     18
## Poland                     260     13
## Portugal                   349     31
## Qatar                      252      0
## Romania                    333     12
## Russia                    2558     18
## Saudi_Arabia               472      6
## Singapore                  386      1
## Spain                     3477    517
## Sweden                     465     20
## Switzerland                279      0
## Turkey                    4093     98
## Ukraine                    325     10
## United_Arab_Emirates       398      3
## United_Kingdom            4342    717
## United_States_of_America 25023   1541
```

```
covidYestNewDeaths <- covidYest[covidYest$deaths > 200,]
y <- matrix( c(covidYestNewDeaths$deaths, covidYestNewDeaths$cases), ncol=2)
colnames(y) <- c("deaths", "cases")
rownames(y) <- covidYestNewDeaths$countriesAndTerritories
tblDeaths <- as.table(y)
tblDeaths
```

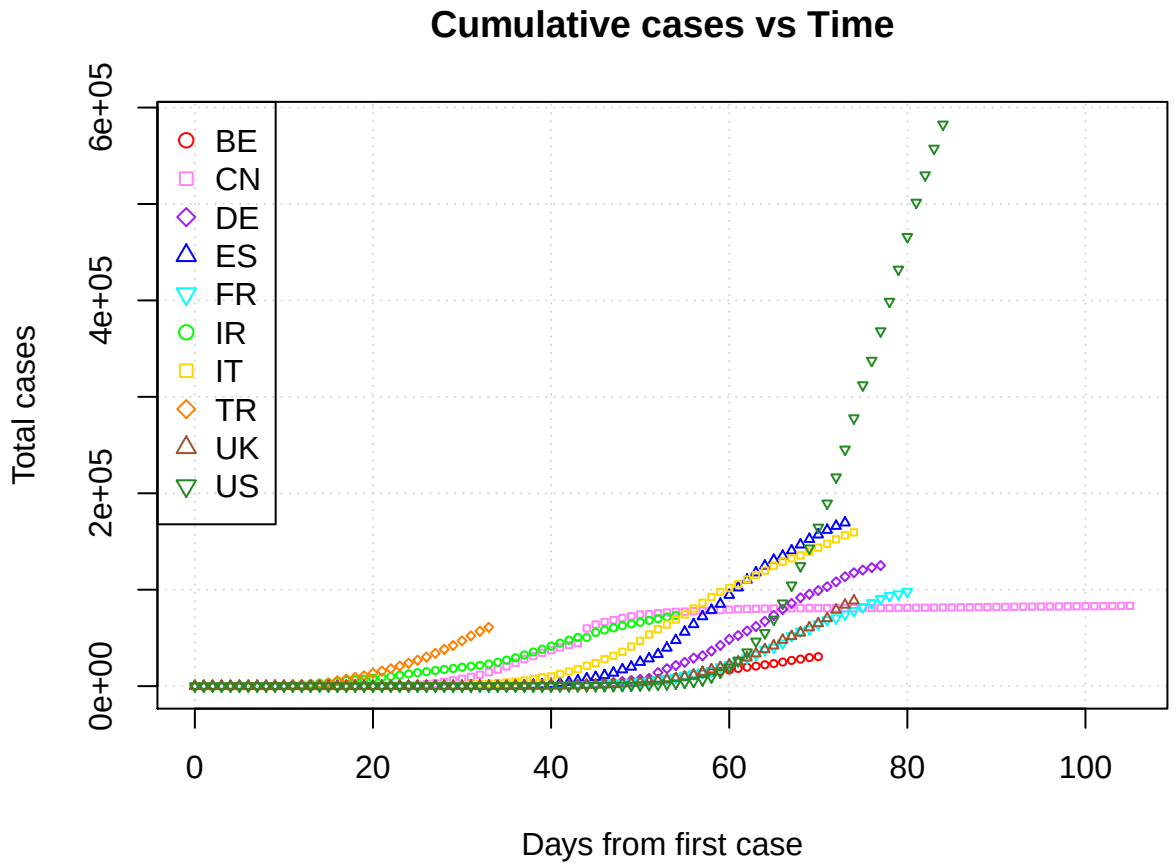
```
##                          deaths cases
## Belgium                     303   942
## France                      574  2673
## Italy                       564  3153
## Spain                       517  3477
## United_Kingdom              717  4342
## United_States_of_America   1541 25023
```

3) Selecting top ten countries in term of cases

```
totCases  <- aggregate(covid$cases, by=list(covid$geoId), FUN=sum)
totCases  <- totCases[order(-totCases$x),][1:10,]
TopgeoIds <- list(totCases[[1]])

covidTop10 <- covid[covid$geoId %in% TopgeoIds[[1]],]
Toplist    <- group_split(covidTop10 %>% group_by(geoId))

for (i in c(1:10)) {
    Toplist[[i]] <- Toplist[[i]][order(Toplist[[i]]$year, Toplist[[i]]$month,
                                       Toplist[[i]]$day),]
    Toplist[[i]][, "cumcases"]  <- cumsum(Toplist[[i]]$cases)
    Toplist[[i]][, "cumdeaths"] <- cumsum(Toplist[[i]]$deaths)
    Toplist[[i]] <- Toplist[[i]][Toplist[[i]]$cumcases>0,]
    Toplist[[i]][, "datenorm"]  <- as.numeric(difftime(Toplist[[i]]$dateRep,
                                                       first(Toplist[[i]]$dateRep),
                                                       units="days"))
}
```

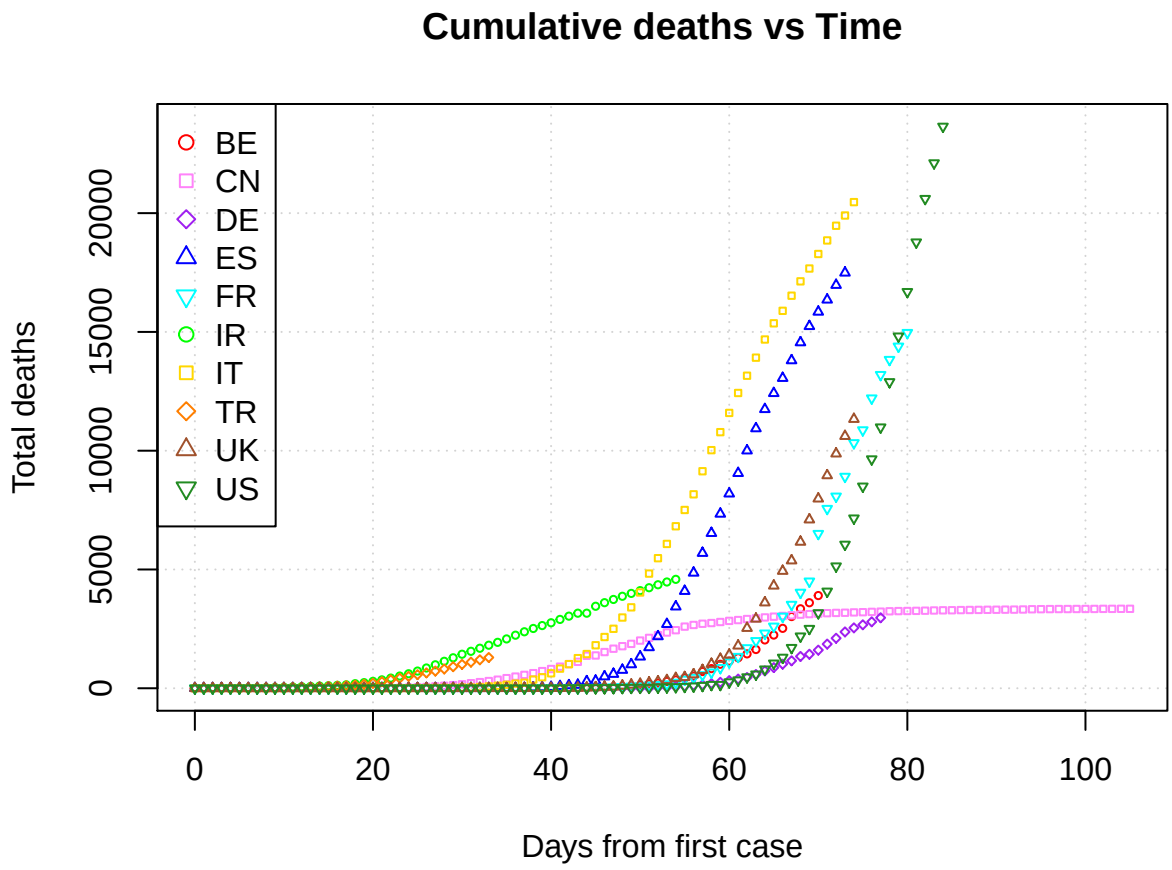
Plotting total number of cases vs time:

```
pchs <- rep(c(21,22,23,24,25), times=2)
maxcases <- max(totCases$x)
maxdays  <- as.numeric(difftime(yest, min(covid$dateRep)), units="days")

plot(Toplist[[1]]$datenorm, Toplist[[1]]$cumcases,
     main="Cumulative cases vs Time", xlab="Days from first case", ylab="Total cases",
     col=cols[1], pch=pchs[1], cex=0.5,
     ylim=c(0,maxcases), xlim=c(0,maxdays),
     panel.first=grid(),
    )
for (i in c(2:10)) {
    points(Toplist[[i]]$datenorm, Toplist[[i]]$cumcases,
           col=cols[i], cex=0.5, pch=pchs[i],
          )
}
par(xpd=TRUE)
legend("topleft", sort(TopgeoIds[[1]]),
       col = cols,
       pch = pchs,
      )
```



Plotting total number of deaths vs time:

```
totDeaths <- aggregate(covid$deaths, by=list(covid$geoId), FUN=sum)
maxdeaths <- max(totDeaths$x)

plot(Toplist[[1]]$datenorm, Toplist[[1]]$cumdeaths,
     main="Cumulative deaths vs Time", xlab="Days from first case", ylab="Total deaths",
     col=cols[1], pch=pchs[1], cex=0.5,
     ylim=c(0,maxdeaths), xlim=c(0,maxdays),
     panel.first=grid(),
    )
for (i in c(2:10)) {
    points(Toplist[[i]]$datenorm, Toplist[[i]]$cumdeaths,
           col=cols[i], pch=pchs[i], cex=0.5,
          )
}
par(xpd=TRUE)
legend("topleft", sort(TopgeoIds[[1]]),
       col = cols,
       pch = pchs,
      )
```



Date of first recorded case in the top-ten countries:

```
for (j in c(1:10)) {
    message(paste0(Toplist[[j]]$countryterritoryCode[1],": ",Toplist[[j]]$dateRep[1]))
}
```

```
## BEL: 2020-02-04
## CHN: 2019-12-31
## DEU: 2020-01-28
## ESP: 2020-02-01
## FRA: 2020-01-25
## IRN: 2020-02-20
## ITA: 2020-01-31
## TUR: 2020-03-12
## GBR: 2020-01-31
## USA: 2020-01-21
```