Practice the basic skills of R on world forest data

B. Ferry, August 2019

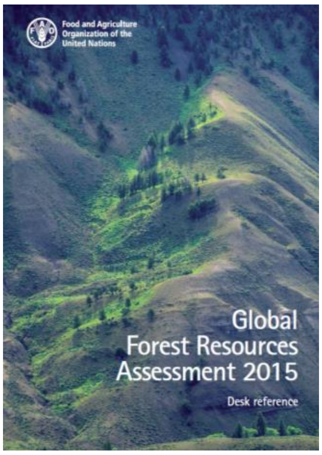
1 - Aim of this tutorial

This tutorial aims at practicing basic skills in R use: getting general information on data frames, selecting subsets of data, getting simple statistics for whole data sets and for data subsets, taking missing values into account, manipulating data frames, creating graphics.

At the same time, this tutorial gives the opportunity to improve your knowledge of some forest data at the world scale.

2 - Data

Almost all data provided for this tutorial are extracted from the Global Forest Resources Assessment (FRA), 2015, published by the Forest Department of the Food and Agriculture Organization (FAO) of the United Nations. The dominant climates of the territories were established separately, by using the Koeppen's Climate Classification map of FAO, SDRN, Agrometeorology Group, 1997.



Three files are provided: 2 with forest data, and 1 giving the signification of the names of the variables. All these files are under the *.csv* format. Create a folder on the D disk of your computer, and copy the files in it.

3 - Organization of the tutorial

3.1 – Work to be done

1rst part : ***run the scripts and see the results***. All the R-scripts are provided, you only have to copy them in a R-sript file in R-studio, run them and **answer the questions in the Excel file *FRA_questions***.

2nd part : ***see the results and write the scripts.*** You should try to replicate the figures provided, with your R scripts. Copy your figures on a file, and save it with your names in the file name.

Send the file with the figures and the Excel file completed, saved as FRA_questions_names.

3.2 – Colors used for the R scripts

The R scripts are written with 3 colors :

Code lines in red should be run for performing the calculations,

Code lines in purple are added for looking at the effects of the previous code lines,

Text lines starting with # give information on what will be done in the next code lines.

Only these lines should be copied on your R-studio file.

4 – Run the scripts and see the results

4.1 - Importing data from files

Data can be directly created with R. However, they are most often imported from files realized with other softwares : text files, Excel files…

We used *csv*. files, separated by semicolons. They are easy to create or modify with Excel, and easy to import from R. We strongly recommend to use the *read.csv2* function to read them, and to avoid *read.table*.

```
# Setting the working directory
setwd("D:/Address_of_the_folder_containing_the_files")
# Reading the first data set and the titles
Data = read.csv2("FRA_data1.csv",row.names=1)
Titles = read.csv2("FRA_titles.csv",row.names=1)
```

4.2 - General information on the data structure

4.2.1 - Main functions

str(x)	structure of an object (size + types and names of its variables)
nrow(df)	number of rows of a data frame
ncol(df)	number of columns of a data frame
dim(df)	numbers of rows and of columns of a data frame
names(df)	names of the columns of a data frame
rownames(df)	names of the rows of a data frame
head(x)	shows the 6 first rows of an object
tail(x)	shows the 6 last rows of an object

```
# Table structure
str(Data)
```

 # Questions 1 and 2

```
# Row names of the table
rownames(Data)
# First rows of the table
head(Data)
```

 # Question 3

```
head(Titles)
```

 # Questions 4 and 5

4.2.2 - Types of variables

For applying a graphic or statistic method to a variable, it is of upmost importance to know if this variable is either a factor or a numeric/integer. This information is given by the *str()* function, but it might be useful to get 2 lists of variables with a script, in case the data frame has a lot of variables mixing both types.

---

class(x)	type of object
sapply(df, fun)	applies a function to all variables of a data frame
is.factor(x)	checks if the variable x is a factor → TRUE or FALSE
is.numeric(x)	checks if the variable x is numeric (or integer) → TRUE or FALSE
which(x)	x : list of elements having the value TRUE or FALSE; the function *which* returns references of the elements having the TRUE value.

```
# Identifying the variables types
sapply(Data, class)
# Tracking the factor variables
sapply(Data,is.factor)
# Positions of the factor variables in the data frame
Vf = which(sapply(Data,is.factor))
Vf
# Positions of the numeric variables in the data frame
Vn = which(sapply(Data,is.numeric))
Vn
```

 # Question 6

4.3 – Selecting subsets of data

Selecting data subsets from an object depends of the type of object. We focus on selecting data from a data frame df (= table with rows and columns), or a vector v (= list of values)

---

df[r,c]	selects the data of the rows *r* and columns *c* (both conditions) *r* and *c* can be lists of numbers or lists of names, or be defined by conditions,
df$Var	selects the variable called "*Var*" in the data frame *df* (equivalent to df[,"Var"]); it works also if *df* is a list of objects and "*Var*" the name of an object of this list
v[n]	selects the data *n* of the vector *v* *n* can be a list of numbers or be defined by a condition

---

4.3.1 - Selection with numbers

```
# Reminder of the 6 first rows of Data
head(Data)
# Selection of 1 line by its row number
Data[3,]
# Selection of a list of lines, with row numbers (example : 1 to 3, and 6)
Data[c(1:3,6),]
# Selection of one specific column (example for column number 1)
Data[,1]
# Selection of rows and columns
Data[1:5,1:2]
```

```
# Selection by removing 1 variable
Data[1:5,-3]
# Selection by removing several variables
Data[1:5,-c(2,3)]
# Selection by removing rows (numbers 1, 3 and 6 to 230)
Data[-c(1,3,6:230),]
# Selection by removing all rows from the 4th to the last
Data[-(4:nrow(Data)),]
```

 # Question 7

4.3.2 - Selection with names

```
# Selection with a row name
Data["Algeria",]
# Same selection with an abbreviation of the row name
Data["Alg",]
# Warning : the abbreviation should be fitted to only 1 row name
Data["Al",]   # Tentative selection of Albania and Algeria
# Selection of a list of rows
Data[c("Af","Alb","Alg","Ango"),]
# Selection of a variable
Data[,"Continent"]
# Warning : abbreviations cannot be used for variables
Data[,"Contin"]   # Tentative selection of a variable with an abbreviation
# Same selection with the $
Data$Continent
# Selection of a data frame subset with at least 2 columns --> data frame
Selec = Data[1:3, c("LA","FA")]
Selec
class(Selec)
# Selection of a data frame subset with only 1 column --> vector
Selec = Data[1:3, "LA"]
Selec
class(Selec)
```

 # Question 8

```
# How to find easily information on the Data variables in Titles
names(Data)
Titles[names(Data),]
Titles["FA",c(1,3)]
```

Keep this in mind, it is likely to be useful in the second part of the tutorial

4.3.3 - Selection with conditions

```
# Select all data from a continent
Data[Data$Continent=="Asia",]
Data[Data[,1]=="Asia",]
```

Conditions are commonly used for selecting subset of rows. It is rarer for columns, but possible.

==	is equal to	!=	is not equal to	&	and
>	is higher than	<=	is lower or equal to	|	or
<	is lower than	>=	is higher or equal to	%in%	is included in

```
# Select all territories having a territory larger than a threshold
Data[Data$LA >= 500000,]
```

 # Question 9

```
# Select the data of a list of territories
Data[rownames(Data) %in% c("France","French Guiana"),]
```

 # Question 10

4.4 – Descriptive statistics of variables

4.4.1 - General function

summary(x)	gives a statistic summary of the object x (depending of the object's type)

```
# Table content
summary(Data)
```

4.4.2 - Descriptive statistics for factor variables

table(x, y, …)	gives the frequency of all combinations of values of x, y, …

```
# Reminder of the factor variables in Data
Vf
# Frequency of the factor values
table(Data$Continent)
```

 # Question 11

4.4.3 - Descriptive statistics for numeric variables

min(x)	minimum value of the elements of x
max(x)	maximum value of the elements of x
median(x)	median value of x : half of its values are higher, half are lower
quantile(x)	minimum, 1rst quartile (25% of the values are lower), median, 3rd quartile (75% of the values are lower), maximum values of x.
median(x)	mean value of x
sd(x)	standard deviation of x

```
# Reminder of the factor variables in Data
Vn
```

```
Datanum = Data[,Vn]
head(Datanum)
# Examples of statistics
apply(Datanum,2,min)
apply(Datanum,2,quantile)
# Remove the missing values (na.rm = TRUE)
apply(Datanum,2,min, na.rm = TRUE)
apply(Datanum,2,quantile, na.rm = T)
apply(Datanum,2,mean, na.rm = T)
apply(Datanum,2,sd, na.rm = T)
```

 # Question 12

4.5 – Missing values

Missing values make the data analyses more complicated. It is often useful to identify precisely which data are missing.

is.na(x)	replaces x by an object of the same dimension, with TRUE in place of the NAs and FALSE elsewhere
!is.na(x)	replaces x by an object of the same dimension, with FALSE in place of the NAs and TRUE elsewhere
apply(df, n°, fun)	applies a function to all rows (n°=1) or all columns (n°=2) of a data frame
sum(x)	gives the sum of all elements of x
na.omit(df)	removes the rows of df having one or several missing value(s)

```
# Creation of a data frame with TRUE in place of the NAs and FALSE everywhere else
MV = is.na(Data)
head(MV)
# Total number of missing values
sum(MV)
# Number of missing values / variables
MVv = apply(MV,2,sum)
MVv
# Number of missing values / rows
MVr = apply(MV,1,sum)
MVr
# Frequency of the number of missing values per row
table(MVr)
# Rows with missing values
Data[MVr>0,]
```

 # Question 13

```
# Table without rows containing missing values
Dataw = Data[MVr == 0,]
dim(Dataw) ; dim(Data)
Dataw = na.omit(Data)
dim(Dataw) ; dim(Data)
```

```
# Substitution of all NAs by 0 (only if there are relevant reasons to do it)
Data[MV]=0    # In fact, the values are equal to 0 in the original data base
# Checking the absence of NAs, without removing rows
summary(Data)
dim(Dataw) ; dim(Data)
```

4.6 – Manipulation of data frames

4.6.1 - Creating variables

It is often useful to create new variables, from the existing ones. With our data, it is interesting to calculate a forest cover rate for every territory.

```
# Creation of a variable "FCR" (Forest Cover Rate")
Data$FCR = Data$FA / Data$LA
summary(Data)
```

Statistic summaries applied to the LA and FA variables showed that both variables have means much higher than their medians. It suggests that these variables have very asymmetrical distributions, as the next chapter on graphics will show it clearly. With strongly asymmetrical variables having only positive values, it is often useful to create transformed variables by a logarithm function, for better studying the distribution of the lower values.

---

log10(x)	decimal logarithm of x (values of x should be >0) 0,01 → -2 / 0,1 → -1 / 1 → 0 / 10 → 1 / 100 → 2 / 1000 → 3 /…

```
# Transformation of LA and FA by the decimal logarithm
Data$LAlog = log10(Data$LA)
Data$FAlog = log10(Data$FA)
summary(Data)
```

 # Question 14

If the positive asymmetrical variable has some null values, we have to use a slightly different function log10(x + e), e being a value near to the lowest but not null value of the variable. We will take 1.

```
# Transformed variables avoiding the infinite values (log(0) →-inf)
Data$LAlog = log10(Data$LA+1)
Data$FAlog = log10(Data$FA+1)
summary(Data)
```

We can observe that the log transformed variables have now a mean and a median that are not very different. It suggests that these functions have a relatively symmetrical distribution.

 # Question 15

4.6.2 - Ordering rows or columns

---

df[order(x), order(y)]   returns the data frame *df* with its rows ordered according to the values of the vector x, and its columns ordered according to the values of the vector y

---

Selection of 2 variables of the data frame (continent ant forest cover rate), and ordering the data rows by increasing forest cover rate

```
Data[order(Data$FCR) , c("Continent","FCR")]
# Here we order the rows by decreasing forest cover rate
# and then we select the 10 first rows
Data[order(Data$FCR, decreasing=T),c("Continent","FCR","FA")][1:10,]
```

 # Question 16

4.7 – Statistics calculated for subsets of data

---

aggregate(df, list(x), fun) applies a function (sum, length, mean…) to all variables of a data frame (df), per levels of a list of factor variables (x). Add "na.rm=T" to remove the missing values. The result is a data frame.

```
# Land and forest covers per continents
Continents = aggregate(Data[,2:3], list(Data$Continent), sum, na.rm=T)
Continents
# Transfer the names of the continents in the row names
rownames(Continents) = Continents[,1]
Continents
# Removal of the first column
Continents = Continents[,-1]
Continents
# Calculation of a forest cover rate / continent
Continents$FCR = Continents$FA / Continents$LA
# Ordering the rows by decreasing forest area
Continents = Continents[order(Continents$FA, decreasing=T),]
Continents
```

4.8 - Graphics

4.8.1 - Comparing values of small data sets

Pie charts and bar charts are adapted to small data sets. The pie chart is focusing on proportions relative to a total amount. The bar chart allows better comparisons of neighbor values.

---

pie(x)	draws a pie chart (x : vector)
barplot(x)	draws a bar chart (x : vector or matrix)

And here are some arguments of these graphical functions :

---

main	(all)	main title of the graph
xlab	(all)	title of the x axis
ylab	(all)	title of the y axis
col	(all)	colors used for plotting the data
labels	(pie)	names of the categories (instead of the numbers)

names	(barplot)	names of the categories
las	(barplot)	orientation of the text (values : 1, 2, 3, 4)
width	(barplot)	widths of the bars

```
# Pie chart of the forest area per continent
pie(Continents$FA, main="Forest area / continent", labels=rownames(Continents))
# Bar chart of the forest area per continent
barplot(Continents$FA, main="Forest area / continent", names=rownames(Continents))
```

Other functions add text or drawings on existing graphs :

---

mtext(t)	add text in the margins of an existing graph
abline(eq)	adds a line on an existing graph ; the slope and position are determined by different types of equation (eq)

---

line	(mtext)	distance to the border of the graph
lty	(abline)	type of line (2 : dashed line)

```
# Barplot of the forest cover rate with bar's width proportional to land area
# --> bars area proportional to forest area
barplot(Continents$FCR, width = Continents$LA,
    names=rownames(Continents), las=2,
    main = "Forest cover rate / continent",
    ylab="Forest cover rate")
mtext("(forest areas proportional to bar areas)",line=0.3)
# Addition of the mean forest cover rate (world scale)
FCRw = sum(Continents$FA)/sum(Continents$LA)
abline(h= FCRw, col="red", lty=2)
mtext(paste("Global FC rate :",round(FCRw*100,1),"%"),line=-4, col="red")
```

---

paste(t,u,v)	paste elements of text
round(x,n)	rounded the x value, at a precision depending of n

---

 # Question 17

```
# Example of small data set : number of territories per continent
table(Data$Continent)
# Pie chart
pie(table(Data$Continent), main="Number of territories")
# Bar chart
barplot(table(Data$Continent), main="Number of territories")
```

4.8.2 - Personalised colors

The colours associated to the values of a factor variable plotted on a graph can be changed.

```
# Choice of colours for the continents
rownames(Continents)
Colcont = c("red","orange","green","blue","magenta")
```

```
# Application to the pie chart
pie(table(Data$Continent), col=Colcont, main="Number of territories")
# Application to the bar chart
barplot(table(Data$Continent), col=Colcont, main="Number of territories")
```

Examples of color names

ivory	pink	lightgreen
lightyellow	violet	lawngreen
yellow	orchid	green
gold	magenta	darkcyan
orange	purple	darkgreen
coral	navy	darkred
tomato	blue	brown
red	cyan	sienna
salmon	skyblue	peru
hotpink	lightblue	tan

4.8.3 - Distributions of large data sets

A histogram gives a complete view of the distribution of a numeric variable. A boxplot gives a more simplified view, but allows more easily to compare several distributions.

Some graphical functions add text or drawings on existing graphs.

----

hist(x)	draws an histogram (x : vector of numeric variables)
boxplot(x)	draws a box plot (x : vector or data frame of numeric variables)

Here are some arguments that we will use with these graphical functions (

----

breaks	(hist)	define how to share the total range of values into intervals
line	(mtext)	distance to the border of the graph

```
# Histograms
hist(Data$LA, main="Histogram of the land areas", xlab="Land area (1000 ha)")
hist(Data$FA, main="Histogram of the forest areas", xlab="Forest area (1000 ha)")
hist(Data$FCR, main="Histogram of the forest cover rates", xlab="Forest cover rate")
hist(Data$LAlog, main="Histogram of the land areas (log10 scale)",
   xlab="log10(Land area (1000 ha)+1)")
```

R chooses by default how to bin your histogram plot using an algorithm, but sometimes the plot is not really easy to read. If you want coarser or finer groups, you can use the breaks() option to change this in a number of ways

```
# Controlling the number of intervals with "breaks"
hist(Data$LAlog, main="Histogram of the land areas (log10 scale)",
   xlab="log10(Land area (1000 ha)+1)", breaks=6)
hist(Data$FAlog, main="Histogram of the forest areas (log10 scale)",
   xlab="log10(Forest area (1000 ha)+1)", breaks=6)
```

 # Question 18

```
# Boxplots of areas (original scales)
```

```
boxplot(Data[,2:3], ylab="Areas (1000 ha)",
    names=c("Land area" ,"Forest area"),
    main="Distribution of the land and forest areas")
# Boxplots of forest areas (original scales)
boxplot(Data[,4], ylab="Forest cover rate",
    main="Distribution of the forest cover rate")
# Boxplots of forest areas (log scales)
boxplot(Data[,5:6], ylab="log(Areas (1000 ha) + 1)",
    names=c("Land area" ,"Forest area"),
    main="Distribution of the land and forest areas")
    mtext("(decimal log scale)",line=0.3)
```

 # Question 19

4.8.4 – Displaying several graphs on a divided window

For plotting several graphs on the same figure, we can divide the window.

par	modifies the graphical parameters (2 arguments are following)
mfrow=c(n,m)	divides the window into n x m windows (n rows, m columns)
mar=c(b,l,u,r)	widths of the four margins : bottom, left, up, right

```
# Drawing 5 histograms in a same figure
# Sharing the window into 2x3 small windows
par(mfrow=c(2,3))
# Reduction of the margins
par(mar=c(2,2.5,2.5,0.5))
# Histograms
hist(Data$LA, main="Land areas")
mtext("(1000 ha)", cex=0.7, line=-0.5)
hist(Data$FA, main="Forest areas")
mtext("(1000 ha)", cex=0.7, line=-0.5)
hist(Data$FCR, main="Forest cover rates")
hist(Data$LAlog, main="log10(Land area +1)" ,breaks=6,
    xlab="log10(Land area (1000 ha)+1)")
mtext("(1000 ha)", cex=0.7, line=-0.5)
hist(Data$FAlog, main="log10(Forest area +1)" ,breaks=6,
    xlab="log10(Forest area (1000 ha)+1)")
mtext("(1000 ha)", cex=0.7, line=-0.5)
# Back to a unique window
par(mfrow=c(1,1))
# Back to the standard margins
par(mar=c(5,4,4,2)+0.1)

# Drawing 3 graphs with boxplots on the same figure
# Sharing the window into 2x2 small windows
par(mfrow=c(2,2))
```

```
# Reduction of the margins
par(mar=c(2,4,4,0.5))
# Boxplots
boxplot(Data[,2:3], ylab="Areas (1000 ha)",
    names=c("Land area" ,"Forest area"),
    main="Land and forest areas")
mtext("(1000 ha)",line=0.3, cex=0.9)
boxplot(Data[,4], ylab="Forest cover rate",
    main="Forest cover rate")
boxplot(Data[,5:6], ylab="log(Areas (1000 ha) + 1)",
    names=c("Land area" ,"Forest area"),
    main="Land and forest areas")
mtext("(log10(1000 ha +1))",line=0.3, cex=0.9)
# Back to a unique window
par(mfrow=c(1,1))
# Back to the standard margins
par(mar=c(5,4,4,2)+0.1)
```

4.8.5 – Relations between a numeric and a factor variable

```
# Relations with the variable "Continent"
boxplot(FCR~Continent, data=Data, main="Forest cover rates / continent")
boxplot(LAlog~Continent, data=Data, main="Land area (log10 scale) / continent")
```

4.8.6 – Relations between 2 numeric variables

plot(x, y) or plot(y~x)	draws a plot with the values of x on the horizontal axis and those of y on the vertical axis (x, y: vectors of numeric variables having the same length)
legend(x)	add a legend on an existing graph

Here are some arguments that we will use with these graphical functions (

log	(plot)	defines if a log scale should be used for one axis or both
type	(plot)	type of the graph ; type="n", only the frame is plotted, not the data

```
# Drawing 3 graphs in a same figure
# Sharing the window into 2x3 small windows
par(mfrow=c(2,2))
# Reduction of the margins
par(mar=c(2,2.5,2.5,0.5))
# Relationships between land and forest area
plot(Data$LA, Data$FA)
plot(Data$LA, Data$FA, log="xy")
plot(Data$LAlog, Data$FAlog)
# Adding a line crossing the origin and with a slope equal to 1
abline(0,1)
```

```
# Back to a unique window
par(mfrow=c(1,1))
# Back to the standard margins
par(mar=c(5,4,4,2)+0.1)

# Plotting the country names, with colors according to the continents
plot(Data$LAlog, Data$FAlog, main="Forest / land areas of the territories",
   xlab="Land area", ylab="Forest area", type="n")
text(Data$LAlog, Data$FAlog, labels=rownames(Data), cex=0.7,
   col=Colcont[Data$Continent])
legend("topleft", legend=levels(Data$Continent), fill=Colcont)

# Selection of territories having a forest area above a threshold
Threshold = 1000
Datab = Data[Data$FA > Threshold,]
plot(Datab$LAlog, Datab$FAlog, main="Forest / land areas of the territories",
   xlab="Land area", ylab="Forest area", type="n")
text(Datab$LAlog, Datab$FAlog, labels=rownames(Datab), cex=0.9,
   col=Colcont[Datab$Continent])
legend("topleft", legend=levels(Datab$Continent), fill=Colcont)
mtext(paste("Forest area >",Threshold,"000 ha"), line=0.5)

# Question 20
```

5 –See the results and write the scripts

5.1 - Importing data from files

Import the file FRA_data2.csv
It contains more variables than FRA_data1.csv, but only for territories having a total forest area larger than or equal to 1000 000 ha (1 million of hectares) = 10 000 km^2.

5.2 - General information on the data structure

 # Questions 21 to 25

5.3 – Some information on the data content

 # Questions 26 to 29

5.4 – Missing values

5.4.1 – Missing values per variable

Calculate the number of missing values per variable, and plot the figure 1 (with the *barplot* function).

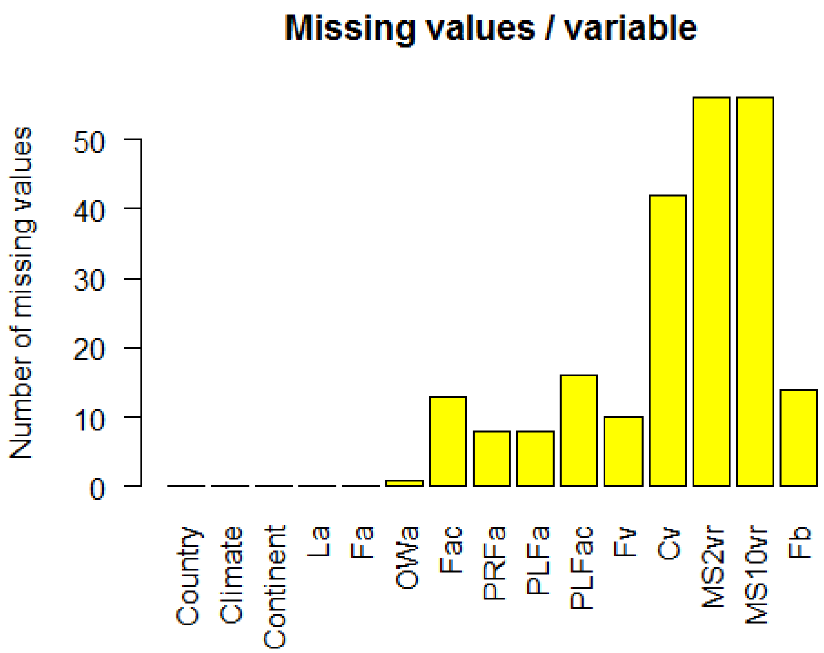


Figure 1 – Number of missing values for every variable of the data set

 # Question 30

5.4.2 – Missing values per row

Calculate the number of missing values per row, then the frequencies of this number (with the *table* function) and plot the figure 2, with the *barplot* function.

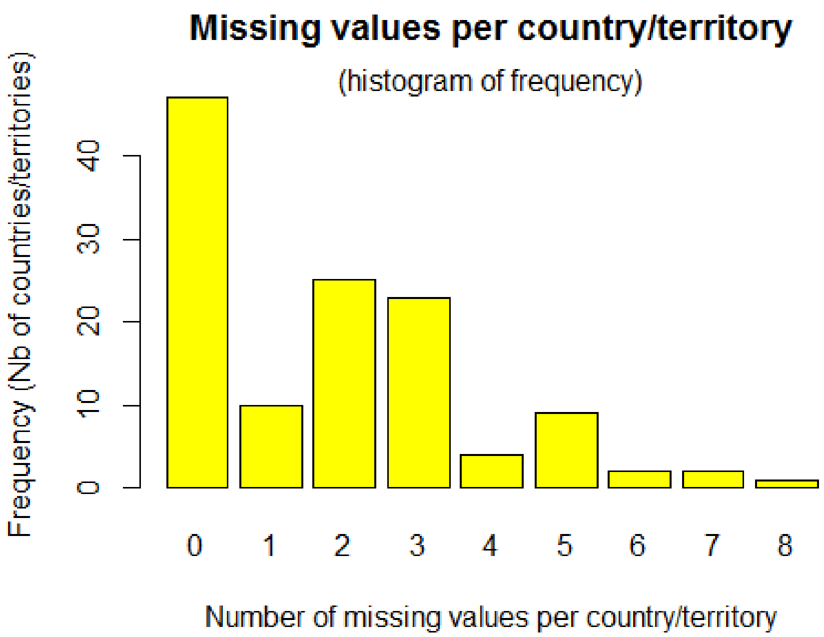


Figure 2 – Histogram of frequency of the number of missing values per country/territory

 # Questions 31 and 32

5.5 – Aggregation of data by climatic zones

Create a data frame called *Climates*, containing the sums of all data for which a sum has a meaning : all numeric variables, except MS2 and MS10 (% of the forest volume).
Put the climate names in the row names, then change the order of the rows with :
Climates = Climates[c("Tr","Te","C","D"),], and define a list of 4 colours for representing the 4 climates (in the same order). Draw the figure 3.

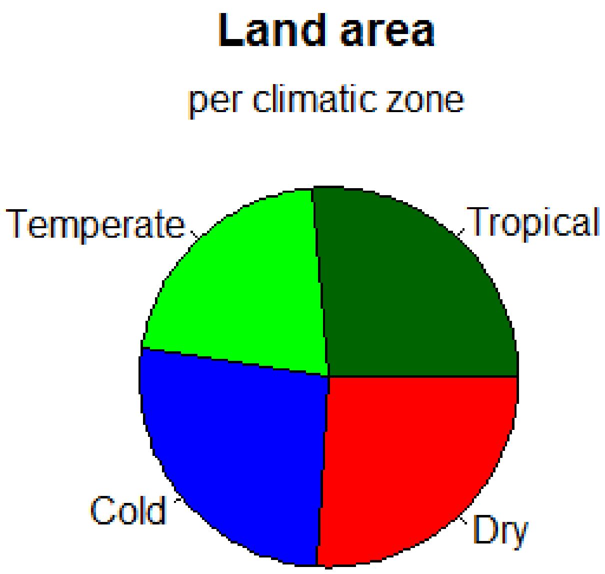


Figure 3 – Distribution of the land area in the five continents, across the main biomes

The pie chart can display only positive values. Therefore, it cannot be used for the forest area change (FAC). The 9 other variables can be plotted, as shown on the figure 4. Do it again.

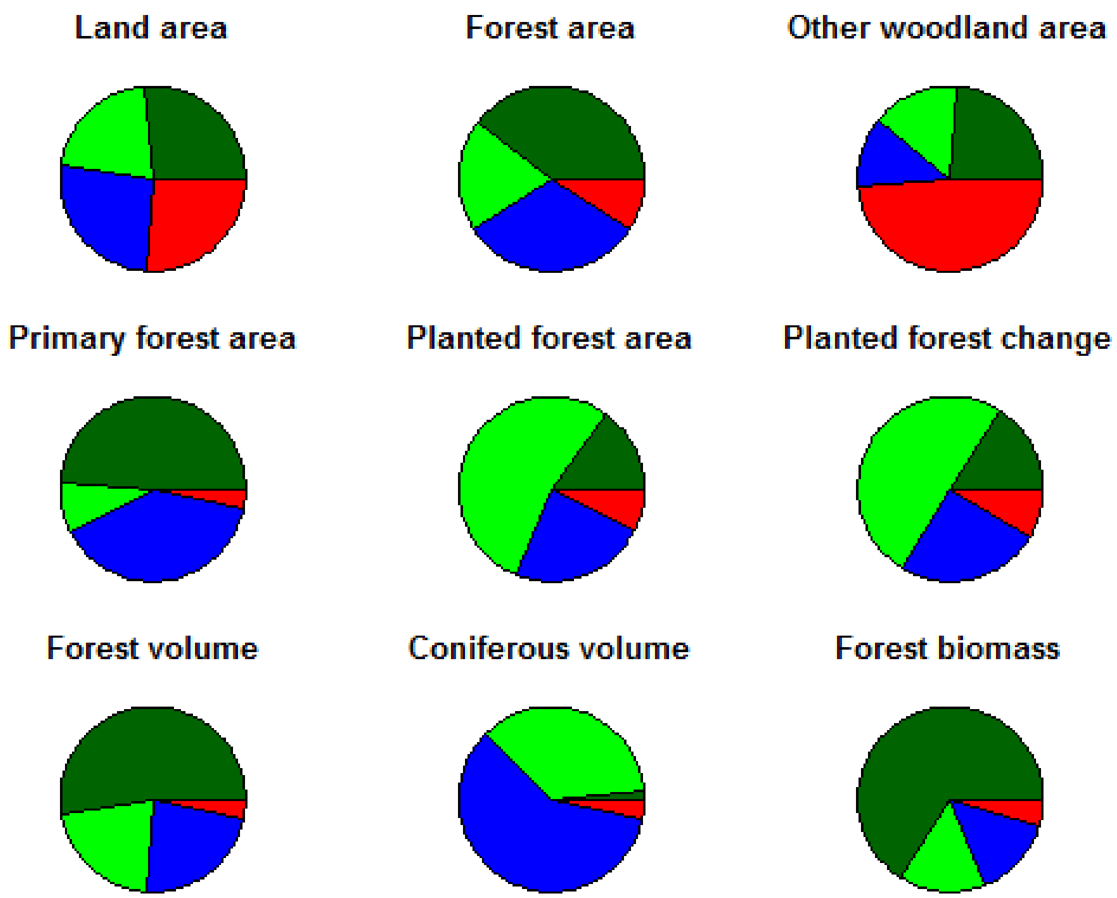


Figure 4 – Distribution of the land area and eight forest indicators, across the main biomes

 # Questions 33 to 37

The *barplot* function can display negative value. Use it for drawing the figure 5.

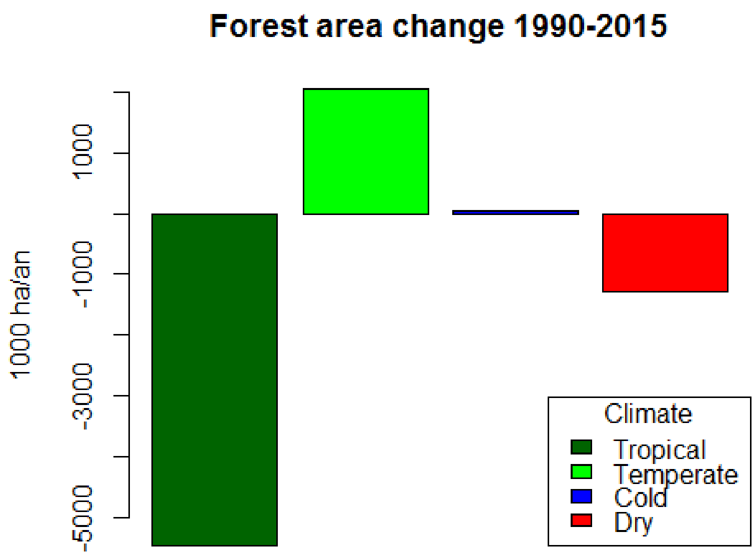


Figure 5 – Global forest area change across the main biomes

 # Questions 38

5.5 – Relations between 2 variables

If we want to compare statistically a variable across different levels of a factor variable, the boxplot is a convenient graphic function. Look at and do yourself the figure 6.

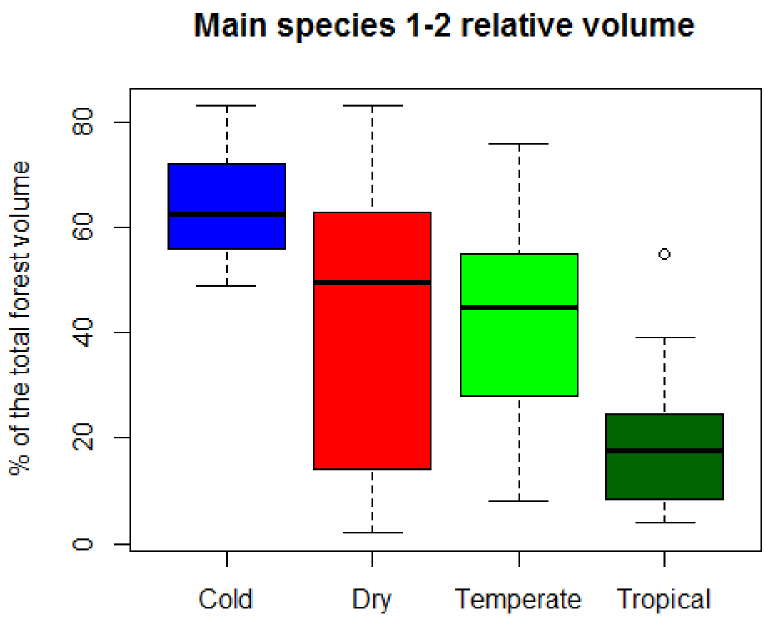


Figure 6 – Distribution of the relative volume of the 2 main forest tree species

 # Question 39

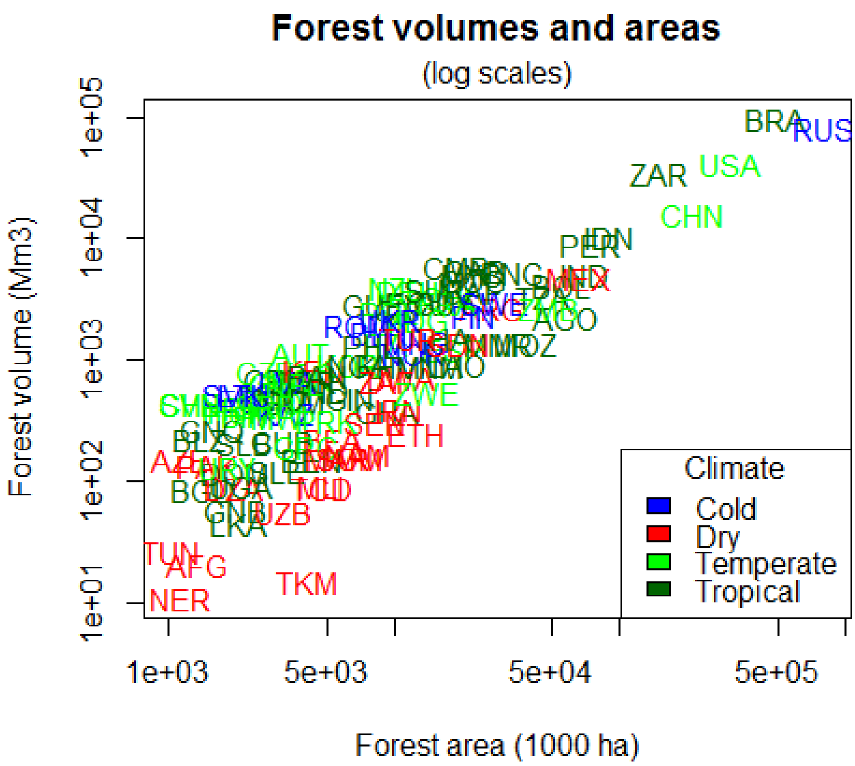


Figure 7 – Relationships between the forest area and the forest volume of countries

 # Question 40