



HEC MONTRÉAL

Self-learning

Introduction to R

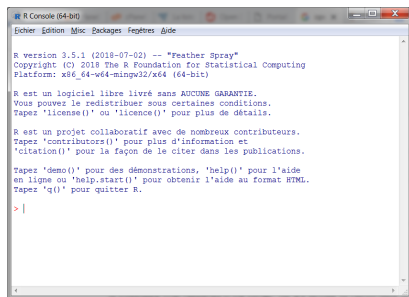
Getting started

You may use the lab computers or your own computer.

To install R on your own computer, visit <http://www.r-project.org>.

R is an open source software. It is free and available for many different OS.

Some prefer to work with RStudio, an interface for R – you still need to install R first.



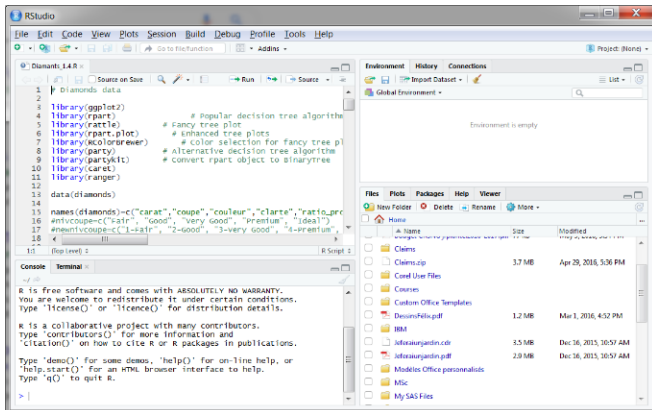
```
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

>
```



Self-paced learning

Learning to code can only be done by doing.

To lead you in this journey, I distributed a handout with commented examples

- Read the handout, and try the examples,
- Try to understand the logic and syntax of the code,
- Try to create some bits of code yourself as you move along,
- During the programming workshop, I will be around to answer your questions and help you understand how to code.
- Try your best to solve the exercises by yourself. I will present possible solutions later.
- Collaborate with one another, ask questions to your peers.

Learning to code is not easy: you need to create something.

Driving a car is easier than fixing it, and fixing it is easier than engineering it from scratch...



Solutions to exercises (Part I)

These suggested solutions are not unique. There are numerous ways to achieve the same task.

```
x=read.table("employees.txt")
```

We have 8 employees and 12 months of data for each.

Data from a given employee appear on consecutive lines.

Question 1: *Total number of transactions and total amounts for all employees all year.*

```
sum(x$nSales)  
sum(x$TotSales)
```

The vectors contain the data for all salesmen and all months.

The `sum` function sums them.

Question 2: Record for most sales (and who got it).

```
which.max(x$Sales)      # Entry number for the record  
x[which.max(x$Sales),]  # Value of that entry
```

Browsing the help allows to find the `which.max()` function. Other solutions could use the `max` function instead.

Question 3: Total sales for each employee (and who sold the most).

```
tots=rep(0,8)           # Creating a vector of zeroes  
emps=unique(x$EmpID)     # Creating a list of employees  
for(i in 1:8){  
  tots[i]=sum(x$TotSales[x$EmpID==emps[i]])  
}  
cbind(emps,tots)         # Place data in two columns and show it
```

A loop was used to treat employees one by one.

Inside the loop, we use the fact that the `sum` function adds all the values together.

Question 4: *Figures to compare average sales across months and across employees.*

```
AveSales=x$TotSales/x$nSales # Computing average sale
boxplot(AveSales~x$Month)    # Variation by month
boxplot(AveSales~x$EmpID)    # Variation by employee
```

Computing average sales uses the fact that operations are done component-wise. Boxplots allow to convey some information about the distribution of the average.

Question 5: *Average sale for the entire year.*

```
stot=rep(0,8)           # Creating a vector of zeroes
ntot=rep(0,8)
emps=unique(x$EmpID)    # Creating a list of the employees
for(i in 1:8){
  stot[i]=sum(x$TotSales[x$EmpID==emps[i]])
  ntot[i]=sum(x$nSales[x$EmpID==emps[i]])
}
stot/ntot
```

Avoid this error: computing the mean of the monthly averages:

```
mean(AveSales[x$EmpID==1000])    # Error!!!
```

Question 6: *Plots showing the evolution of monthly average sale.*

```
emps=unique(x$EmpID)
par(mfrow=c(2,4))      # Multiple plits in the same window
for(i in 1000:1007){
  plot(1:12,AveSales[x$EmpID==i],type='l',xlab="Mois",
       ylab="Average Sale")
  title(paste("Monthly Average sale for employee",i))
}
```

This solution reuses code from the examples of the handout.

Question 7: *Regression model to look at time trend.*

```
x$EmpID=as.factor(x$EmpID)
model=lm(TotSales~Month+EmpID,x)
summary(model)
```

Defining the ID as a factor will make it a fixed effect for each employee.

This allows to account for difference in the salesmen's abilities.

Month has a positive significant trend.

Solutions to exercises (Part II)

These suggested solutions are not unique. There are numerous ways to achieve the same task.

Question 1: How many missing values

```
x=read.table("salesmen.txt")  
apply(is.na(x),2,sum)
```

Question 2: What about branch 25 (employees 25xx).

```
x[x$EmpID%/%100==25,]
```

All employees have missing values between January and April.
This branch may have just recently open.

Alternative solution: `x[x$EmpID>=2500 & x$EmpID<2600,]`

Question 3: Determine FTEs for all branches.

If a salesman is missing, his number of sales will be missing.

```
branch=x$EmpID%/%100  
works=!is.na(x$nSales)  
etp=tapply(works,branch,sum)/12
```

Question 4: Create a list of sales matrices (employee \times month).

```
extract=function(i){t(matrix(x$TotSales[branch==i],nrow=12))}  
listbybranch=lapply(1:25,extract)
```

The `lapply()` statement may also be replaced with a loop:

```
tmp=list()  
for(i in 1:25){ tmp[[i]]=extract(i) }
```

Question 5: Using an `_apply` statement, compute total sales for each branch (with a plot).

```
totbybranch=sapply(listbybranch,sum,na.rm=TRUE)
pie(totbybranch)
title("Annual total sales by branch")
```

Question 6: Sales per FTE for all branches (and a plot).

```
avebybranch=totbybranch/etp
barplot(avebybranch)
title("Annual sales per FTE for all branches")
```

Question 7: *Function that returns the evolution of sales (number and total).*

```
salesevolution=function(i,data){
  keep=data$EmpID%%100==i
  par(mfrow=c(1,2))

  tabSales=matrix(data$nSales[keep],nrow=12)
  nSmonthly=apply(tabSales,1,sum,na.rm=TRUE)
  plot(1:12,nSmonthly,type='l',xlab="Month",
       ylab="Number of Sales")
  title("Progression of the monthly number of sales ")
  title(sub=paste("Branch",i))

  tabTot=matrix(data$TotSales[keep],nrow=12)
  totmonthly=apply(tabTot,1,sum,na.rm=TRUE)
  plot(1:12,totmonthly,type='l',xlab="Month",
       ylab="Total sales")
  title("Progression of the monthly total sales")
  title(sub=paste("Branch",i))
}
salesevolution(1,x)
```

Part III

Questions from Part I

```
x=read.table("employees.txt")
```

Question 3, Part I

```
x %>%  
  group_by(EmpID) %>%  
  summarise(TotSale=sum(TotSales))
```

Question 5, Part I

```
x %>%  
  group_by(EmpID) %>%  
  summarise(TotSales=sum(TotSales), nSales=sum(nSales)) %>%  
  mutate(AvgSale=TotSales/nSales)
```

Questions from Part II

```
x=read.table("salesmen.txt")
```

Question 2, Part II

```
x %>% mutate(branch=(EmpID%/%100)) %>%  
  filter(branch==25)
```

Question 3, Part II

```
x %>% mutate(branch=(EmpID%/%100)) %>%  
  group_by(as.factor(branch)) %>%  
  summarise(fte=sum(!is.na(nSales))/12)
```

Question 4 to 6, Part II

```
totbybranch = x %>% mutate(branch=(EmpID%/%100)) %>%  
  group_by(as.factor(branch)) %>%  
  summarise(fte=sum(!is.na(nSales))/12,  
    TotSales=sum(TotSales,na.rm=TRUE)) %>%  
  mutate(Avg=TotSales/fte)  
totbybranch  
pie(totbybranch$Avg)  
title("Annual total sales by branch")
```

Vectorial Plots Demo (with CorelDRAW)

Vectorial plots: better quality, more flexibility, editable.

