# Courte introduction à R

Florent Benaych-Georges

Université Paris Descartes

Janvier 2015

***Séries temporelles pratiques***

# Commencer

2 étapes :

1. Télécharger et installer **R** depuis
   `http://cran.r-project.org`

2. Utiliser **R studio**, un software open-source pour R :
   `http://www.rstudio.com`

# Principaux types

Essentiellement 3 types d'objets :

char, logical, numeric

```
1  x=5
2  class(x)
3
4  y=(x<6)
5  y
6  class(y)
7
8  (z="Hello world") #parentheses: result display
9  class(z)
```

*(file:* Types.R*)*

# Principaux conteneurs

Essentiellement 5 types de conteneurs :

- vector
- list
- factor
- data.frame
- ts

# Vecteurs

Toutes les entrées ont le **même type**

```r
(v=c(2,5,7,8)) #c=concatenates entries or vectors
                    #in a single vector

(u=c(v,9,15:18)); u[u<10]

(w=letters[1:7]); class(w) #vectors: not only with
                           #numerics as entries
w[3] #third coordinate
w[length(v4)] #last entry
w[c(1,4,5)] #1st, 4th and 5th entries
w[-c(1,4,5)] #all but 1st, 4th and 5th entries

(x=seq(from=1, to=2, by=.2))

(y=rep(x,2))
```

*(file:* Vectors.R*)*

Conteneur similaire en 2 dimensions : **matrix**

# Listes

Les coordonnées peuvent avoir des **types différents** et des **noms**

```
1  L=list(name="Alban",age=1,country="France",
2           size_in_cm=83)
3  str(L)
4  L$name
5  L[[2]]
```

*(file:* Lists.R*)*

# Facteurs

Adaptés pour les données **qualitatives** (i.e. non quantitatives)

```r
C=c("blue","blue","red","red","blue","green",
    "yellow","green","red") #vector of characters
(my_factor1=factor(C))
str(my_factor1)
table(my_factor1)

L2=c("blue","red","green","yellow","orange")
(my_factor2=factor(C,levels=L2))
table(my_factor2)
```

*(file:* Factors.R*)*

# Data.frames

**data.frame** : **liste** de vecteurs de **même taille** mais éventuellement de classes différentes

```r
name=c("Clo","Tony","Alban","Tom","Lou")
city=c("Paris","Paris","Lyon","NYC","Shanghai")
size=c(170,181,83,185,179)

(d = data.frame(name,city,size))
str(d)
dim(d); nrow(d); ncol(d)
d$name
d[2,3] #2nd entry of 3rd column
d$year = c(1983,1980,2013,1988,1986) #add column
d[2,] #2nd row
d[,c(3,1)] #3rd and 1st columns
d[,-c(2,4)] #all but 2nd and 4th columns
d[d$city=="Paris",] #extracts a subset
colnames(d)[4]="birth_year" #changes column name
summary(d)
```

*(file:* Data.frame.R*)*

# ts (Time Series) (1)

Objet de type **ts** : un **vecteur** de valeurs, un vecteur **start** (année, moment), un vecteur **end** (année, moment) et une **fréquence**

```r
# save a numeric vector containing 72 monthly observations
myvector = .1*(1:72)+sin((1:72)*pi/12) +
  rnorm(72,mean=0,sd=.3)

# from Jan 2009 to Dec 2014 as a time series object
myts = ts(myvector, start=c(2009, 1), end=c(2014, 12),
          frequency=12)
class(myts)
plot(myts,col="red")

# subset the time series (June 2014 to December 2014)
myts2 <- window(myts, start=c(2010, 6), end=c(2014, 12))
class(myts2)
plot(myts2,col="blue")
```

*(file:* Example_ts_1.R*)*

# ts (Time Series) (2)

Objet de type **ts** : un **vecteur** de valeurs, un vecteur **start** (année, moment), un vecteur **end** (année, moment) et une **fréquence**

```r
# save a numeric vector containing 72 monthly observations
myvector = .1*(1:72)+sin((1:72)*pi/12) +
  rnorm(72,mean=0,sd=.3)

# from Jan 2009 to Dec 2014 as a time series object
myts = ts(myvector, start=c(2009, 1), end=c(2014, 12),
          frequency=12)

# Seasonal decomposition
fit = stl(myts, s.window="period")
plot(fit,col="purple")

# additional plots
monthplot(myts)
library(forecast)
seasonplot(myts)
```

*(file:* Example_ts_2.R*)*

# ts (Time Series, Corrélation) (2)

Objet de type **ts** : un **vecteur** de valeurs, un vecteur **start** (année, moment), un vecteur **end** (année, moment) et une **fréquence**

```r
data(USAccDeaths)

USAccDeaths

bruit.blanc=ts(rnorm(100),frequency = 1, start = c(1), end=c

plot( bruit.blanc)

acf(bruit.blanc, type="correlation") #ou type="covariance" o
```

*(file:* Example_ts_3.R*)*

# Autres types, conversions

Beaucoup d'autres types en R (date, matrix, etc...). Conversions possibles grâce à as.mon_type(mon_object)

```
1  (d1=as.Date("1999-09-17"))
2  class(d1)
3
4  (d2=as.Date("19980901",format="%Y%m%d"))
5  (my_dates=seq(d1,d2,by="-1 month"))
6
7  (x=as.numeric(d1))
8
9  (my_ch=as.character(d2))
10
11 (my_factor1=factor(c("blue","red","green","red")))
12 (v=as.character(my_factor1))
```

*(file:* Dates_and_conversions.R*)*

# Obtenir et modifier le répertoire de travail

```
1  getwd() #get working directory
2  setwd("~/Desktop/") #set working directory
3  #setwd works with linux absolute or relative pathnames:
4
5  # ~/ returns you to your login directory
6
7  # /home/ takes you to the home directory
8  #where user login directories are usually stored
9
10 # ../  moves you up one directory
11
12 # ./my_sub_dir  moves you down to the subdirectory
13   #named "my_sub_dir"
14
15 getwd() #get working directory
16 dir() #list of directory files
17 ls() #list of R objects
```

*(file:* Linux_pathnames.R*)*

# Importer des données

Une **étape clé** dans l'analyse de données Souvent : **compliqué ! !**

```r
(x=scan("my_data_file.dat"))
class(x) #scan gives a vector back

(y=read.table("my_table.dat", header=1))
class(y) #read.table gives a data.frame back

ta=read.csv(file="exo1.csv", header=1, skip=6)
#here, read.table also works
#"skip" allows to skip a certain number of rows

h=read.table("http://www.cmapx.polytechnique.fr/~benaych/201
             header=T, fill=T)#fill: blank fields
                              #added when rows
                              #have missing values
names(h)
str(h)
summary(h)
```

*(file:* Import_data.R*)*

# Export de données

Beaucoup plus facile que l'importation

```
1  write(runif(5),file="my_writing_file.dat")
2
3  y=read.table("my_table.dat", header=1)
4  write.table(y,file="my_writing_file_for_table.dat")
5
6  write.csv(y,file="my_writing_file_for_table.csv")
7  (z=read.table("my_writing_file_for_table.dat",
8                header=1))
9
10 ta=read.csv(file="exo1.csv", header=1, skip=6)
11 write.csv(ta, file = "../my_csv_file.csv")
```

*(file:* Export_data.R*)*

# Import/export d'objects R

```r
myvector = .1*(1:72)+sin((1:72)*pi/12) +
  rnorm(72,mean=0,sd=.3)

# from Jan 2009 to Dec 2014 as a time series object
myts = ts(myvector, start=c(2009, 1), end=c(2014, 12),
          frequency=12)

save(myts,file="my_rda_file.rda")
rm(list=ls())
print(myts)
load("my_rda_file.rda")
print(myts)
plot(myts)
```

*(file:* Import_export_R_objects.R*)*

# Fonctions

```r
f = function(x) {x*x}
f(5)
f(1:7)

PF = function(n=1,proba=.5) {
           as.integer(runif(n)<proba)}
PF(10)
PF(10,.1)

PF2 = function(n=1,proba=.5) {
   ech=as.integer(runif(n)<proba)
   freq=mean(ech)
   return (list(sample=ech,frequency=freq))}
PF2(10,.6)
```

*(file:* Functions.R*)*

# Structures de contrôle

```
1  x=runif(1); y=c(); z=c(); #x: random in [0,1]
2  for(i in 1:10) y[i]=sin(pi*i/10)
3  y==sin(pi*(1:10)/10)
4
5  for(i in 1:10) {
6    if (y[i]<.5) {z[i]=y[i]} else {z[i]=y[i]+1}}
7  z==((y<.5)*y+(y>=.5)*(y+1))
8
9  while(x<.5) {x=runif(1)}
10 x
11
12 #way quicker : matrix approach:
13 v=runif(1E5) #10^5 random numbers in [0,1]
14 t=Sys.time()
15 for (i in 1:length(v)){v[i]=v[i]+5}
16 t1=Sys.time()-t
17 t=Sys.time()
18 v=v+5
19 t2=Sys.time()-t; as.numeric(t1)/as.numeric(t2)
```

*(file:* Control structures.R*)*

# Graphiques : plot, points, lines

```
graphics.off() #erases previous graphics
x=seq(.4,4,.1); y=log(x^2+1/x^2)
plot(x,y,type="p") #p for points
plot(x,y,type="l") #l for line
plot(x,y,type="s") #l for step
plot(x,y,type="p",pch=2) #pch : type of points
##ylim=c(ay,by), xlim=c(ax,bx) : bounds of the axes
##lty : type of line, col : color, main : title
x=rnorm (20); y=rexp (20)
plot(x,y,col="red",ylim=c(-0.5 , 5),xlim=c(-3,2))
points(x+.5,y-1,pch=2,col="blue") #add points
legend("topleft",c("plot_1","plot_2"),col=c("red","blue"),
       pch=c(1,2))
lines(x,y, lty=2, col="green") #add line
abline (h=3,col="yellow",lty=1)#add horizontal line
abline (v=-1.3,col="aquamarine2")#add vertical line
abline (a=1,b=1,col="coral3")#add oblique line
text(-2,2 ,"commentary"); title("Example␣curves")
```

*(file:* Plot_Intro.R*)*

# Les nombreuses "méthodes" de plot

```
1  #Images producted by plot(x) depend on the class of x:
2  methods(plot)
3
4  #Examples:
5  class(sin); plot(sin,xlim=c(-pi,pi),xlab="x",ylab="sin(x)")
6
7  x=rpois (1000 ,1); y=table(x);class(y);plot(y)
8
9  # save a numeric vector containing 72 monthly observations
10 myvector = .1*(1:72)+sin((1:72)*pi/12) +
11   rnorm(72,mean=0,sd=.3)
12
13 # from Jan 2009 to Dec 2014 as a time series object
14 myts = ts(myvector, start=c(2009, 1), end=c(2014, 12),
15           frequency=12)
16 class(myts); plot(myts,col="red")
17
18 r=rnorm(1E3,1); w=density(r); class(w)
19 plot(w)
```

*(file:* Plot_Intro_2-methods.R*)*

# Probabilités, variables éléatoires, distributions

```r
#Discrete laws: binom, hyper, pois, geom, sample,...
#Continuous: norm, unif, t, chisq,...

# General syntax, for "my_law" a probability distribution:
# rmy_law(n,...) : sample
# dmy_law(x,...) : density of my_law at x
# pmy_law(x,...) : P( X =< x) for X distributed as my_law
# qmy_law(a,...) : quantile with level a

x=seq(-5,5,.01)
u=seq(0,1,.01)
par(mfrow=c(2,2))#defines an array of plots
plot(rnorm(500,0,1),pch=20,col="red") #sample
plot(x,dnorm(x,0,1),type="l",col="blue") #density
plot(x,pnorm(x,0,1),type="l",col="purple") #P( X =< x)
plot(u,qnorm(u,0,1),type="l",col="green") #quantiles
graphics.off() #erases graphics
```

*(file:* Probability_Random_variables_Distributions.R*)*

# Description graphique des données : hist, boxplot, pie, barplot...

```r
data(iris); str(iris); summary(iris); head(iris)
x=iris$Sepal.Length
mean(x); sd(x); var(x)
quantile(x,c(.25 ,.5 ,.75))
boxplot(x)
hist(x)
hist(x,breaks = 12)

records=sample(c("clear", "cloudy", "rainy"),365,
                    replace=T,prob=c(5,4,1))
pie(table(records))
barplot(table(records))
```

*(file:* Graphical_description_of_data-hist-boxplot-pie-barplot.R*)*

# Description graphique des données : distributions conditionnelles

```
1  xyplot(Ozone~Wind | Month,data=airquality,layout=c(5,1))
2  histogram(~Sepal.Length|Species,data=iris)
3  bwplot(Sepal.Length~Species,data=iris)
4
5  require(UsingR) #opens the functions and data of
6  #the package UsingR
7  attach(kid.weights) #allows to write height
8  #instead of kid.weights$height...
9  histogram(~weight|gender,data=kid.weights,layout=c(1,2))
10 histogram(~height|gender,data=kid.weights,layout=c(1,2))
11 mean(weight[gender=="M"])
12 mean(weight[gender=="F"])
13 mean(height[gender=="M"])
14 mean(height[gender=="F"])
15
16 H=read.csv(file="male_female_over_20_heights.csv")
17 str(H)
18 histogram(~H$height|H$gender,layout=c(1,2))
```

# Graphical description of data : correlations

```r
require(UsingR) #opens the functions and data of
                #the package UsingR
data(kid.weights)
X=kid.weights; str(X)
pairs(X[ ,1:3] , col=X[ ,4], pch=20)

## BMI : body mass index
attach(kid.weights) #allows to write height
             #instead of kid.weights$height...
ht_in_m = height*2.54/100          # 2.54 cm per inch
weight_in_kg = weight / 2.2046          # 2.2046 lbs. per kg
bmi = weight_in_kg/ht_in_m^2
histogram(bmi)
histogram(~bmi|gender)

library(car)
states=as.data.frame(state.x77[,c("Murder", "Population",
                              "Illiteracy", "Income", "Frost")])
scatterplotMatrix(states ,spread=T,pch=20,lty.smooth=2)
```

*(file:* Graphical description of data−correlations.R*)*

# Help, demos, examples

```r
help("plot")

?plot

help.search("plot")

??plot

demo() # demontrations list

demo( graphics )

example(plot) #the example function usually runs
              #the exemples that one can find at
              #the end of the help files
```

*(file:* help_demo_examples.R*)*