TP1 : Introduction à R

1. INITIALISER SON ENVIRONNEMENT

Initialisation
- Créer votre dossier de travail et placer dedans une copie du Raccourci de R dont le paramètre « démarrer dans » est vide.
- Lancer R à partir de ce raccourci puis vérifiez via la commande *getwd()* que vous êtes bien situé au sein de votre dossier.
- Quitter R en sauvegardant l'espace de travail.
- Vérifier la création de deux fichiers ***.Rdata*** (contenant tous les objets crées lors de la session) et **.Rhistory** (conserve l'historique de la session).
- au démarrage 27 librairies de base sont activées.

Télécharger une librairie
Au démarrage 27 librairies de base sont activées, la liste est obtenue via la commande *library().*
Pour télécharger une nouvelle librairie, sélectionnez *Packages>Load Package…* puis sélectionnez dans la liste proposée la librairie de votre choix. Pour utiliser une librairie particulière tapez
> library(nom de la librairie)

2. IMPORTER LES DONNEES

Un fichier de données peut être créé à partir de plusieurs logiciels : tableur, traitement de texte, bloc-notes à condition qu'il soit sauvegardé dans le format `.txt'.
Télécharger dans votre répertoire le fichier *« measure.txt »* situé dans le répertoire « à préciser ».

Sous R étudiez l'effet des commandes suivantes :

> read.table("measure.txt")
> A<-read.table("measure.txt")
>A
>edit(A)

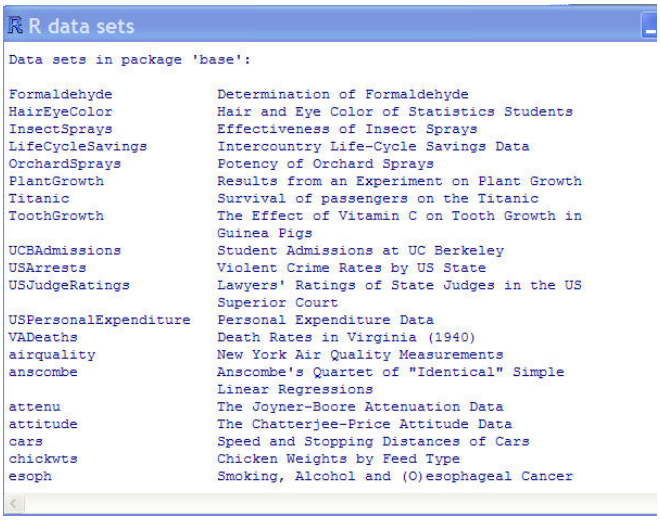
Modifiez le contenu de A, fermer l'éditeur puis exécuter de nouveau edit(A).
Que constatez vous.

>B<-edit(A)
>edit(B)

Modifiez le contenu de B, fermer l'éditeur puis exécuter de nouveau edit(B). Que constatez vous.

Pour conserver les modifications la commande *> fixe(A)* peut également être utilisée.

R fourni un ensemble de données consultables via la commande data().
> data()



Pour télécharger un jeu de données et le manipuler
>data(cars)
> cars
> ?cars
>attributes(cars)
> label<-attributes(cars)$row.names
>plot(cars)
>text(cars,label)

Créer un fichier texte *"cars.txt"* à partir du data.frame *cars :*

> write.table(cars,"cars.txt")

3. CONSULTER LA DOCUMENTATION

> ? read.table ou >help(read.table)

Quittez en sauvegardant l'espace de travail *Save workspace…* ainsi que l'historique *Save history….*

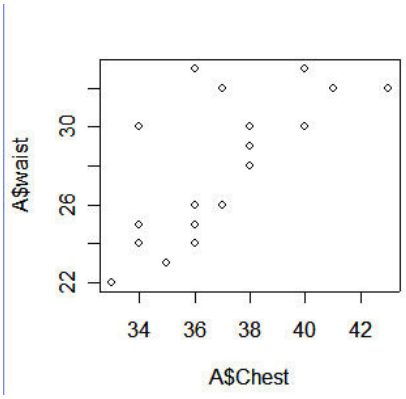
4. UTILISER L'HISTORIQUE

Chargez l'historique par *Load history…* puis faire défiler via les flèches haut et bas les commandes, que retrouve-t-on ?

5. CONSERVER LES RÉSULTATS

considérant les commandes et résultats suivants :

> summary(A)
Chest waist hips
Min. :33.00 Min. :22.0 Min. :32.00
1st Qu.:35.75 1st Qu.:25.0 1st Qu.:35.75
Median :36.50 Median :28.5 Median :37.00
Mean :37.00 Mean :28.0 Mean :37.05
3rd Qu.:38.00 3rd Qu.:30.5 3rd Qu.:38.25
Max. :43.00 Max. :33.0 Max. :42.00
> plot(A$Chest,A$waist)



Copier les listings (sélection à la souris) et les figures (sauvegarder avec click droit souris) et coller dans un document.

6. ÉCRIRE UNE FONCTION

Créer dans votre dossier un fichier txt « testfonction.txt » contenant le code suivant :

```
testfonction<-function()
{
    par(mfrow=c(2,2))
    x<-A$Chest
    y<-A$hips
    plot(x,y)
    abline(lm(y~x))
    hist(x)
    hist(y)
    plot(0,0,type="n", xlim=c(0,100), ylim=c(0,10))
    lines(1:100,sqrt(1:100))
    title("racine carrée")
}
```

Sauvegarder le fichier puis le renommer en « testfonction.R »
Sous R ouvrir le fichier via *File>Source R code…* puis sélectionner le fichier « testfonction.R »
Vérifier l'exécution de la commande
> source("D:/../testfonction.R")
Etudiez et analyser l'effet des commandes

>ls()
>testfonction()

7. EXPORTER DES DONNEES

Etudiez les commandes d'import et d'export des données suivantes :

> dput(A,"toto.txt") *examinez le contenu du fichier toto.txt*
> fich<-dget("toto.txt")
>fich
>ls()
> write.table(A,"toto.txt") *examinez le contenu du fichier toto.txt*
> write.table(A,"toto.txt",row=F,sep="\t") *examinez le contenu du fichier toto.txt*

8. MANIPULATION D'OBJETS ELEMENTAIRES

8.1 VECTEURS

a) Initialisation et mode

```
> v<-c(1,4,3)
> v
[1] 1 4 3
> mode(v)
[1] "numeric"

> v=1:4
> s
[1] 1 2 3 4
> mode(v)
[1] "numeric"

> v=c(1,-2,"a")
> v
[1] "1"  "-2" "a"
> mode(v)
[1] "character"

> v=c(1,-2,3.27)
> v
[1]  1.00 -2.00  3.27
> mode(v)
[1] "numeric"

> v=c(T,F,F)
> v
[1]  TRUE FALSE FALSE
> mode(v)
[1] "logical"
```

b) Accès aux éléments d'un vecteur

```
> v=c(1,-2,4,12,-2.30)
> v
[1]  1.0 -2.0  4.0 12.0 -2.3
> v[3]
[1] 4
> v[2:4]
[1] -2  4 12
> v[-4]
[1]  1.0 -2.0  4.0 -2.3

> v
[1]  1.0 -2.0  4.0 12.0 -2.3
> v[v>0]
[1]  1  4 12
```

```
> v>0
[1]  TRUE FALSE  TRUE  TRUE FALSE

> 1:5
[1] 1 2 3 4 5
> (1:5)[v>0]
[1] 1 3 4          # récupère les indices des valeurs TRUE
```

c) Fonctions de base

Observer l'effet des fonctions ci-dessous puis déduire leur fonctionnement.

append et replace

```
> length(v)
[1] 5
> append(v, c(-3,0), after=4)
[1]  1.0 -2.0  4.0 12.0 -3.0  0.0 -2.3
> replace(v,v>0,0)
[1]  0.0 -2.0  0.0  0.0 -2.3
```

unique, ***sort***, ***rank*** *et* ***order***

```
> x<-Chest[1:10]
> x
 [1] 34 37 38 36 38 43 40 38 40 41
> unique(x)
[1] 34 37 38 36 43 40 41
> sort(x)
 [1] 34 36 37 38 38 38 40 40 41 43
> rank(x)
 [1]  1.0  3.0  5.0  2.0  5.0 10.0  7.5  5.0  7.5  9.0
> order(x)
 [1]  1  4  2  3  5  8  7  9 10  6
> x[order(x)]
 [1] 34 36 37 38 38 38 40 40 41 43
```

sum, ***prod***, ***cumsum***, ***cumprod*** *et* ***diff***

```
> sum(x)
[1] 385
> cumsum(x)
 [1]  34  71 109 145 183 226 266 304 344 385
> diff(x)
[1]  3  1 -2  2  5 -3 -2  2  1
> diff(x,lag=2)
[1]  4 -1  0  7  2 -5  0  3
> prod(x)
[1] 7.00981e+15
> cumprod(x)
 [1] 3.400000e+01 1.258000e+03 4.780400e+04 1.720944e+06 6.539587e+07
 [6] 2.812022e+09 1.124809e+11 4.274274e+12 1.709710e+14 7.009810e+15
```

rep, seq et rev

```
> rep("bc",4)
[1] "bc" "bc" "bc" "bc"
> rep(c("bc","a"),4)
[1] "bc" "a"  "bc" "a"  "bc" "a"  "bc" "a"
> rep(c("bc","a"),c(2,3))
[1] "bc" "bc" "a"  "a"  "a"
> rep(1:3,3:1)
[1] 1 1 1 2 2 3
> seq(from=1,to=8,by=2)
[1] 1 3 5 7
> seq(from=1,to=8,le=3)
[1] 1.0 4.5 8.0
> seq(from=1,to=8,le=6)
[1] 1.0 2.4 3.8 5.2 6.6 8.0
> rev(1:5)
[1] 5 4 3 2 1
```

8.2 TABLEAUX (DATA .FRAME)

a) Initialisation et mode

```
> is.vector(A)
[1] FALSE
> is.data.frame(A)
[1] TRUE
```

Un tableau est caractérisé par l'hétérogénéité de ces colonnes (variable quantitative, qualitative, ordinale,…). Une matrice par contre ne contient qu'un seul type de données. La classe des data.frame inclut la classe des matrices. Une matrice peut être redéfinie comme de type data.frame.

```
> m=matrix("a",nrow=3,ncol=2)
> m
     [,1] [,2]
[1,] "a"  "a"
[2,] "a"  "a"
[3,] "a"  "a"
> is.data.frame(m)
[1] FALSE
> is.matrix(m)
[1] TRUE
> as.data.frame(m)
  V1 V2
1  a  a
2  a  a
3  a  a
```

b) Accès aux éléments d'une matrice

```
> A
   Chest waist hips
1     34    30   32
2     37    32   37
3     38    30   36
4     36    33   39
```

```
…
20   35   23   35
> A[3:4,2:3]
  waist hips
3    30   36
4    33   39
> dim(A)
[1] 20  3
> waist
Error: Object "waist" not found
> A$waist
 [1] 30 32 30 33 29 32 …25 26 28 23
> attach(A)
> waist
 [1] 30 32 30 33 29 …25 26 28 23
```

9. STATISTIQUES ELEMENTAIRES

a) min, max, range, mean, var, quantile, median

```
> Chest
 [1] 34 37 38 36 38 43 40 38 40 41 36 36 34 33 36 37 34 36 38 35
> min(Chest)
[1] 33
> max(Chest)
[1] 43
> range(Chest)
[1] 33 43
> var(Ches
+ st)
Error: syntax error
> mean(Chest)
[1] 37
> var(Chest)
[1] 6.631579
> median(Chest)
[1] 36.5
> quantile(Chest)
   0%   25%   50%   75%  100%
33.00 35.75 36.50 38.00 43.00
```

b) Codage d'une variable quantitative en variable qualitative ordinale (cut)

```
> Chest
 [1] 34 37 38 36 38 43 40 38 40 41 36 36 34 33 36 37 34 36 38 35

> classe<-c(33,35,36,38,43)   # définition des bornes des classes
> labelClasse<-c("faible","peu faible","moyen","élevé")

> labelClasse
[1] "faible"     "peu faible" "moyen"      "élevé"
```

```
> ChestQual<-cut(Chest,classe,inc=T,labels=labelClasse)
> ChestQual
 [1] faible     moyen      moyen      peu faible moyen      élevé
 [7] élevé      moyen      élevé      élevé      peu faible peu faible
[13] faible     faible     peu faible moyen      faible     peu faible
[19] moyen      faible
Levels: faible peu faible moyen élevé
```

c) Construire la liste de valeurs par catégorie (split)

```
> w<-split(Chest,ChestQual)
> w
$faible
[1] 34 34 33 34 35
$"peu faible"
[1] 36 36 36 36 36
$moyen
[1] 37 38 38 38 37 38
$"élevé"
[1] 43 40 40 41

> is.vector(w)
[1] TRUE
> is.list(w)
[1] TRUE
```

d) Application d'une fonction à tous les éléments de la liste (lapply)

```
> lapply(w,length)
$faible
[1] 5
$"peu faible"
[1] 5
$moyen
[1] 6
$"élevé"
[1] 4

> ChestWaist<-cbind.data.frame(Chest,waist)
> ChestWaist
   Chest waist
1     34    30
2     37    32
...
18    36    26
19    38    28
20    35    23
> w<-split(ChestWaist,ChestQual)
> w
```

$faible	$"peu faible"	$moyen	$"élevé"
Chest waist	Chest waist	Chest waist	Chest waist
1 34 30	4 36 33	2 37 32	6 43 32
13 34 24	11 36 24	3 38 30	7 40 33
14 33 22	12 36 25	5 38 29	9 40 30
17 34 25	15 36 26	8 38 30	10 41 32
20 35 23	18 36 26	16 37 26	
		19 38 28	

```
> par(mfrow=c(2,2))
> lapply(w,plot,xlim=c(20,50), ylim=c(10,40))
```