Séance 1: Premiers pas sous R

J Gaudart, R Giorgi,JC Thalabard, D Thiam, S Whegang

Septembre 2010

Objectifs

- Entrer/ Sortir de **R**
- Calcul et variables
- Lectures/ Ecritures de données dans des fichiers/ Communications avec l'extérieur

Présentation générale

R est un logiciel d'analyse de données d'accès libre, qui a l'avantage d'être disponible aussi bien sous Windows que sous Mac ou Linux. Il est constitué d'un module de base auquel peuvent être rajoutés des modules plus spécifiques à tel ou tel domaine d'analyse ou d'application. Chacun de ces modules, appelÃl's "package", doit,tout d'abord, être chargé à partir du site CRAN (`http://www.r-project.org/`) ou d'un de ses proxies sur votre machine, puis "appelé" dans une session de travail lorsque vous souhaitez l'utiliser.
Pour ce séminaire, le logiciel R ainsi q'un certain nombre de ces modules ont été préalablement chargés sur vos machines.

Démarrer R

- Repérer le sigle ***R*** et double cliquer dessus
- Repérer un éditeur de texte type blocnote et l'ouvrir
  - Ce fichier ouvert va permettre de garder les ordres tapés au fur et à mesure
  - Des commentaires peuvent être insérés précédés du signe **#**
  - Les ordres écrits sont exécutés par un copier (ctl-C)- coller (ctl-V) dans R
- Rien de plus désagréable lorsqu'on aborde un nouveau logiciel que de perdre les ordres tapés ou devoir les retaper fastidieusement à chaque erreur ou lors d'un arrêt intempestif de la machine. Il est possible à chaque instant de sortir d'une session R en conservant tout ce qui a été fait au cours de la session en utilisant la fonction *save*("nom du fichier de sauvegarde"), puis ultérieurement d'ouvrir une nouvelle session R et charger ce qui avait été effectué auparavant en utilisant la fonction *load*("nom du fichier de sauvegarde"). Le fichier *.Rhistory* contient les ordres R utilisés lors d'une session.

R comme calculatrice

Les opérations élémentaires

- La ligne de commande
- Addition, soustraction, multiplication, division

```
>2+2
```

```
[1] 4
```

```
>5.23-4.1

[1] 1.13

>12*2

[1] 24

>34/3

[1] 11.3
```

- Les opérateurs imbriqués

```
>log(0.6/(1-0.6))

[1] 0.405

>(12*(1.5-0.5))/3

[1] 4
```

Exercice

L'indice de Quételet (IMC, BMI) est défini par

$$BMI = \frac{poids}{taille^2} \tag{1}$$

où le poids est exprimé en kg et la taille en m

- Calculer votre BMI

```
>67/(1.80)^2

[1] 20.7
```

- Créer une variable appelée *bmi* conservant cette valeur

```
>bmi = 67/(1.80)^2
```

- Remarque importante: R est sensible aux caractères majuscule/ minuscule. Ainsi si vous taper Bmi ou BMI, R vous indique qu'il ne connait pas cette variable. Par contre, si vous taper bmi, il vous donne la dernière valeur stockée

```
>bmi

[1] 20.7
```

Création d'un ensemble de valeurs: vecteurs et matrices

R est un langage capable de créer et manipuler des objets globalement

- On suppose disposer, chez 5 sujets, de leur couple (poids,taille)

$$\{(65,175),(82,178),(45,152),(63,157),(70,180)\}$$

pour lesquels, on veut calculer les BMI correspondants

```
>poids = c(65,82,45,63,70)
>taille = c(175,178,152,157,180)
>BMI = poids/(taille^2)
>BMI
```

```
[1] 0.00212 0.00259 0.00195 0.00256 0.00216
```

- Visualisation des valeurs des "vecteurs" poids, taille, BMI en regard de chaque sujet

```
>cbind(poids,taille,BMI)
```

```
     poids taille     BMI
[1,]    65    175 0.00212
[2,]    82    178 0.00259
[3,]    45    152 0.00195
[4,]    63    157 0.00256
[5,]    70    180 0.00216
```

L'objet ainsi créé peut être gardé pour être manipulé globalement

```
>Donnees = cbind(poids,taille,BMI)
>#  Visualisation du tableau
>Donnees
```

```
     poids taille     BMI
[1,]    65    175 0.00212
[2,]    82    178 0.00259
[3,]    45    152 0.00195
[4,]    63    157 0.00256
[5,]    70    180 0.00216
```

```
>#  Utilisation de la fonction summary()
>summary(Donnees)
```

```
     poids          taille          BMI
 Min.   :45   Min.   :152   Min.   :0.00195
 1st Qu.:63   1st Qu.:157   1st Qu.:0.00212
 Median :65   Median :175   Median :0.00216
 Mean   :65   Mean   :168   Mean   :0.00227
 3rd Qu.:70   3rd Qu.:178   3rd Qu.:0.00256
 Max.   :82   Max.   :180   Max.   :0.00259
```

- On souhaite rajouter à ce tableau les identifiants des sujets, en précisant s'il s'agit d'un homme ou d'une femme

```
>Iden = c("Victor","Georges","Suzanne","Anne","Jean")
>Genre =c("Masculin","Masculin","Feminin","Feminin","Masculin")
>Donnees=cbind(Iden,Genre,poids,taille,BMI)
># Visualisation du nouveau tableau
>Donnees
```

```
     Iden      Genre      poids taille
[1,] "Victor"  "Masculin" "65"  "175"
[2,] "Georges" "Masculin" "82"  "178"
[3,] "Suzanne" "Feminin"  "45"  "152"
[4,] "Anne"    "Feminin"  "63"  "157"
[5,] "Jean"    "Masculin" "70"  "180"
     BMI
[1,] "0.00212244897959184"
[2,] "0.00258805706350208"
[3,] "0.00194771468144044"
[4,] "0.00255588462006572"
[5,] "0.00216049382716049"
```

```
># Que fait ici la fonction summary()
>summary(Donnees)
```

```
      Iden        Genre   poids  taille
 Anne   :1   Feminin :2   45:1   152:1
 Georges:1   Masculin:3   63:1   157:1
 Jean   :1                65:1   175:1
 Suzanne:1                70:1   178:1
 Victor :1                82:1   180:1
                  BMI
 0.00194771468144044:1
 0.00212244897959184:1
 0.00216049382716049:1
 0.00255588462006572:1
 0.00258805706350208:1
```

- Les colonnes correspondant au poids, à la taille et au BMI ont visiblement perdu leur caractère numérique et sont devenus des caractères. Comment faire pour garder aux variables leur nature lorsqu'elles sont regroupées dans un objet tableau?
- structure *data.frame()*

```
>Donnees= data.frame(Iden,Genre,poids,taille,BMI)
>Donnees
```

```
     Iden    Genre poids taille     BMI
1  Victor Masculin    65    175 0.00212
2 Georges Masculin    82    178 0.00259
3 Suzanne  Feminin    45    152 0.00195
4    Anne  Feminin    63    157 0.00256
5    Jean Masculin    70    180 0.00216
```

```
>summary(Donnees)
```

```
     Iden        Genre       poids        taille
 Anne   :1   Feminin :2   Min.   :45   Min.   :152
 Georges:1   Masculin:3   1st Qu.:63   1st Qu.:157
 Jean   :1                Median :65   Median :175
 Suzanne:1                Mean   :65   Mean   :168
 Victor :1                3rd Qu.:70   3rd Qu.:178
                          Max.   :82   Max.   :180
     BMI
 Min.   :0.00195
 1st Qu.:0.00212
 Median :0.00216
 Mean   :0.00227
 3rd Qu.:0.00256
 Max.   :0.00259
```

- Accès aux éléments d'une structure de data.frame
  - Noms des colonnes individuelles

    ```
    >names(Donnees)

    [1] "Iden"   "Genre"  "poids"  "taille" "BMI"
    ```

  - Contenu de la colonne correspondant au poids
  - Repérage d'un élément particulier par les indices de ligne et de colonne

    ```
    >#  Sujet 2, colonne 3
    >Donnees[2,3]

    [1] 82

    >#  Sujet 2, toutes les valeurs associees
    >Donnees[2,]

         Iden    Genre poids taille     BMI
    2 Georges Masculin    82    178 0.00259

    >#  Toutes les valeurs de la variable poids (colonne 3)
    >Donnees[,3]

    [1] 65 82 45 63 70

    >Donnees$poids

    [1] 65 82 45 63 70
    ```

- Transformation de la variable *Genre* en indicateur logique: utilisation d'un comparateur logique == appliquée à un vecteur.
  Il peut être souhaité de disposer d'un indicateur VRAI/ FAUX (TRUE/ FALSE) pour remplacer les deux valeurs "Masculin"/"Feminin"

  ```
  >Indicateur <- Donnees$Genre == "Masculin"
  ```

  Bien noter la différence entre un signe = unique lorsqu'il s'agit d'égaler des nombres ou valeurs numériques entre eux du double signe == lorsqu'il s'agit de faire une comparaison logique avec un résultat en TRUE/FALSE Cette nouvelle variable logique est incorporée au *data.frame* Donnees précédent de la manière suivante

  ```
  >Donnees = data.frame(Donnees,Indicateur)
  >Donnees
  ```

```
     Iden    Genre poids taille     BMI Indicateur
1  Victor Masculin    65    175 0.00212       TRUE
2 Georges Masculin    82    178 0.00259       TRUE
3 Suzanne  Feminin    45    152 0.00195      FALSE
4    Anne  Feminin    63    157 0.00256      FALSE
5    Jean Masculin    70    180 0.00216       TRUE
```

Remarque: une autre manière plus directe aurait été de créer directement la variables dans le tableau en écrivant

```
>Donnees$Indicateur = Donnees$Genre == "Masculin"
```

Dans ce dernier cas, la variable Indicateur est directement créée dans le *data.frame* Donnees et n'est pas connue en dehors du *data.frame* Donnees

Quelques repères "topographiques"

Liste des variables présentes en mémoire: fonction *ls()*

```
>ls()
```

```
[1] "bmi"        "BMI"        "Donnees"    "Genre"
[5] "Iden"       "Indicateur" "poids"      "taille"
```

Nature des variables créées: fonctions *class()* et *str()*

```
>class(Donnees)
```

```
[1] "data.frame"
```

```
>class(poids)
```

```
[1] "numeric"
```

```
>class(Indicateur)
```

```
[1] "logical"
```

```
>class(Genre)
```

```
[1] "character"
```

```
>str(Donnees)
```

```
'data.frame':        5 obs. of  6 variables:
 $ Iden      : Factor w/ 5 levels "Anne","Georges",..: 5 2 4 1 3
 $ Genre     : Factor w/ 2 levels "Feminin","Masculin": 2 2 1 1 2
 $ poids     : num  65 82 45 63 70
 $ taille    : num  175 178 152 157 180
 $ BMI       : num  0.00212 0.00259 0.00195 0.00256 0.00216
 $ Indicateur: logi  TRUE TRUE FALSE FALSE TRUE
```

Nous avons vu les types élémentaires *numeric*, *character*, *logic*. Il existe également le type *factor*. Ces types élémentaires peuvent être assemblés sous forme de *vector*, de *matrix*, ensembles homogènes de ces briques, qui peuvent être manipulés globalement. Enfin, des éléments de natures différentes peuvent être assemblés en *data.frame()* (tailles identiques) ou en *list()* (tailles éventuellement différentes).

Aide sur les fonctions: *help()*

```
>help(summary)
>help(data.frame)
>help.search("glm")
```

Lorsque le souvenir du nom est inexact, il est possible d'utiliser la fonction *apropos()*

```
>apropos("file")
```

```
 [1] "bzfile"             "close.srcfile"
 [3] "download.file"      "env.profile"
 [5] "file"               "file.access"
 [7] "file.append"        "file.choose"
 [9] "file.copy"          "file.create"
[11] "file.edit"          "file.exists"
[13] "file.info"          "file.path"
[15] "file.remove"        "file.rename"
[17] "file.show"          "file.symlink"
[19] "file_test"          "gzfile"
[21] "list.files"         "memory.profile"
[23] "open.srcfile"       "open.srcfilecopy"
[25] "parseNamespaceFile" "print.srcfile"
[27] "profile"            "readCitationFile"
[29] "srcfile"            "srcfilecopy"
[31] "summary.srcfile"    "system.file"
[33] "tempfile"           "xzfile"
[35] "zip.file.extract"
```

```
>apropos("read")
```

```
 [1] "readBin"          "readChar"
 [3] "readCitationFile" "read.csv"
 [5] "read.csv2"        "read.dcf"
 [7] "read.delim"       "read.delim2"
 [9] "read.DIF"         "read.fortran"
[11] "read.ftable"      "read.fwf"
[13] "readline"         "readLines"
[15] ".readRDS"         "read.socket"
[17] "read.table"       "read.table.url"
[19] "Sys.readlink"
```

```
>apropos("write")
```

```
 [1] "RtangleWritedoc"     "RweaveLatexWritedoc"
 [3] "write"               "writeBin"
 [5] "writeChar"           "write.csv"
 [7] "write.csv2"          "write.dcf"
 [9] "write.ftable"        "writeLines"
[11] "write.socket"        "write.table"
[13] "write.table0"
```

Où sommes- nous?: la fonction *getwd()*

```
>getwd()
```

```
[1] "/media/JCT31012009/Enseignement/Stafav/Seminaire_R_Marseille/Septembre_2010"
```

Travailler sur un directory donné: la fonction *setwd()*

```
>setwd("/media/JCT31012009/Enseignement/Stafav/Seminaire_R_Marseille/Septembre_2010/Fichiers_Travail/")
```

Les packages présents

```
>library()
```

Supprimer des variables de la mémoire: la fonction *rm()*

```
>#  Creation de la variable Glycemie
>Glycemie = c(0.80,1.10,0.95,1.05,0.97)
>ls()
```

```
[1] "bmi"        "BMI"        "Donnees"    "Genre"
[5] "Glycemie"   "Iden"       "Indicateur" "poids"
[9] "taille"
```

```
>#  La liste integre maintenant la variable Glycemie
>rm(Glycemie)
>#  La variable Glycemie est supprimee et n'apparait plus maintenant dans la liste de variables presente
>ls()
```

```
[1] "bmi"        "BMI"        "Donnees"    "Genre"
[5] "Iden"       "Indicateur" "poids"      "taille"
```

Entrées/ Sorties de données

Lecture d'un fichier de données: de l'extérieur vers R

Données rentrées dans un tableau type Excel *donnees.xls*

Il est important de se rappeler que R, comme la plupart des logiciels scientifiques, est un logiciel anglo- saxon, où la marque décimale est le point "." et non pas la virgule ",". Il est sans doute important de prendre l'habitude de travailler en format international, pour éviter des déconvenues lors d'échanges de données avec collègues étrangers. Dès lors, la marque de séparation entre des données ne peut être que l'espace ou le point virgule ";" et non la virgule ",", spontanément proposée dans la version francisée d'Excel.
Les données se présentent, généralement, sous forme d'un fichier .xls, comportant autant de lignes que d'individus.

- Ouvrir le fichier *Excel* **donnees1.xls**
- Faire une sauvegarde du fichier sous le format .csv: *donnees1.csv*
  - Noter que cette sauvegarde vous propose, éventuellement, le type de séparation entre les données: choisissez comme séparateur **";"**

Lecture du fichier *donnees1.csv* dans R: la fonction *read.table*

La fonction permettant de lire un tel fichier dans R à partir de sa création sur un tableur quelconque (Excel, ...) est la fonction *read.table* où il faut préciser le nom du fichier à lire (précédé éventuellement du nom du directory), le mode de séparation des informations par ligne et le fait qu'on veut ou non garder comme noms de variables les noms figurants sur la première ligne.

```
>donnees1 = read.table("donnees1.csv",sep=";",header=TRUE)
>#
># header=TRUE signifie que la premiere ligne du fichier correspond aux noms de variables
># sep= ";" indique le type de separateur utilise dans le fichier
># skip = n indique qu'on souhaite "sauter" les n premieres lignes du fichier
>#
>str(donnees1)
```

```
'data.frame':        72 obs. of  2 variables:
 $ count: int  10 7 20 14 14 12 10 23 17 20 ...
 $ spray: Factor w/ 6 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
>head(donnees1)
```

```
  count spray
1    10     A
2     7     A
3    20     A
4    14     A
5    14     A
6    12     A
```

Exercice: lire le fichier donnees2.xls

Lecture des grands fichiers de données: la fonction *scan()*

Lecture de données saisies dans d'autres formats: le package *foreign*

Dans de nombreuses situations, les données sont saisies sous des formats propres à des logiciels auxquels les chercheurs et leurs associés sont habitués comme EPI-INFO, SAS, S+, Stata, etc....Il existe un package spécifique, le package *foreign* qui possède des fonctions spécifiques pour ce type de lecture

Exemple: lecture d'un fichier stata

```
>library(foreign)
>Data = read.dta("EssaiPatients.dta")
>str(Data)
```

```
'data.frame':        20 obs. of  9 variables:
 $ Iden      : num  1 2 3 4 5 6 7 8 9 10 ...
 $ Traitement: chr  "A" "B" "A" "B" ...
 $ d_clin_j0 : num  16973 16985 16985 16986 17001 ...
 $ deces     : int  0 0 0 1 1 0 0 0 0 1 ...
 $ d_deces   : num  NA NA NA 17021 17019 ...
 $ d_der_clin: num  17734 17811 17734 17002 17016 ...
 $ d_fin     : num  17734 17811 17734 17021 17019 ...
 $ cd4M0     : int  NA 131 NA 220 NA 560 NA NA 142 218 ...
 $ d_cd4M0   : num  NA 16971 NA 16971 NA ...
 - attr(*, "datalabel")= chr "Written by R.              "
 - attr(*, "time.stamp")= chr ""
 - attr(*, "formats")= chr  "%9.0g" "%1s" "%9.0g" "%9.0g" ...
 - attr(*, "types")= int  100 128 100 108 100 100 100 108 100
 - attr(*, "val.labels")= chr  "" "" "" "" ...
 - attr(*, "var.labels")= chr  "Iden" "Traitement" "d_clin_j0" "deces" ...
 - attr(*, "version")= int 7
```

```
>head(Data)
```

```
  Iden Traitement d_clin_j0 deces d_deces d_der_clin d_fin
1    1          A     16973     0      NA      17734 17734
2    2          B     16985     0      NA      17811 17811
3    3          A     16985     0      NA      17734 17734
4    4          B     16986     1   17021      17002 17021
5    5          A     17001     1   17019      17016 17019
6    6          B     17010     0      NA      17741 17741
  cd4M0 d_cd4M0
1    NA      NA
2   131   16971
3    NA      NA
4   220   16971
5    NA      NA
6   560   17006
```

Sortie de données: de R vers l'extérieur

- Ecriture des valeurs prises par une variable unique dans un fichier texte: fonction *dput()*

```
>dput(poids,'poids.txt')
```

Dans ce cas, la relecture de la variable peut se faire en utilisant la fonction *dget()*

```
>poids_bis = dget("poids.txt")
>poids_bis
```

```
[1] 65 82 45 63 70
```

- Ecriture des valeurs de variables contenues dans un data.frame: fonction *write.table()*

```
>write.table(BMI,file="BMI.csv",sep=";",eol="\n")
```