

Introduction au logiciel R

Biomining team

Département d'économétrie, Université de Genève

Séminaire de recherche StatS-Demo

Outline

- 1 Introduction
- 2 Installation et démarrage
- 3 Objets et opérateurs
- 4 Fonction et programmation

R

R est :

- un environnement logiciel d'analyse statistique et graphique.
- un langage dérivé de S (langage de S-PLUS)
- distribué librement sous licence GPL
- disponible sur les plates-formes : Windows/Mac/Linux/Unix
- facilement extensible à l'aide de modules

Installation

- R et les modules sont disponibles sur le site :
<http://cran.r-project.org>
- Par défaut, aucune interface graphique n'est proposée sous Linux.
- Sous Windows et MacOSX, elle est sommaire.

Première utilisation

Quatre possibilités pour envoyer des commandes à R :

- ① Taper les commandes directement dans la console
- ② Utiliser l'éditeur de script -> Fichier/Nouveau script (uniquement Windows/Mac)
- ③ Utiliser le module Rcmdr
- ④ Utiliser un éditeur externe (Tinn-R, WinEDT etc.)

Objets

R manipule des objets :

- Assignment d'une valeur à un objet "a" :

```
> a <- 50
```
- Case-sensitive : $a \neq A$
- Opération sur un objet :

```
> a / 50  
[1] 25
```

Types d'objets

Plusieurs types d'objets :

- vecteur : 4 5 1 ou dans R `c(4,5,1)`
"D" "E" "A" ou dans R `c("D","E","A")`
- facteur : variable catégorielle
- matrice : tableau de données **numériques**
- tableau de données (data frame) : tableau de données
- ...

Objets

- On peut aussi placer des résultats dans un objet
- Suivant son type, des méthodes peuvent lui être appliquées

- Exemple :

```
> iris.reg <- lm(Petal.Length ~ Petal.Width, data=iris)
> summary(iris.reg)
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.08356     0.07297   14.85  <2e-16 ***
Petal.Width  2.22994     0.05140   43.39  <2e-16 ***
> anova(iris.reg)
Response: Petal.Length
              Df Sum Sq Mean Sq F value    Pr(>F)
Petal.Width   1  430.48   430.48  1882.5 < 2.2e-16 ***
Residuals    148   33.84    0.23
```

Indexes

Indexing vectors

```
x[n] nth element
x[-n] all but the nth element
x[1:n] first n elements
x[-(1:n)] elements from n+1 to the end
x[c(1,4,2)] specific elements
x["name"] element named "name"
x[x > 3] all elements greater than 3
x[x > 3 & x < 5] all elements between 3 and 5
x[x %in% c("a","and","the")] elements in the given set
```

Indexing matrices

```
x[i,j] element at row i, column j
x[i,] row i
x[,j] column j
x[,c(1,3)] columns 1 and 3
x["name",] row named "name"
Indexing data frames (matrix indexing plus the following)
x[["name"]] column named "name"
x$name id.
```

Ouverture et fermeture

- R sauvegarde l'environnement de travail dans le fichier .RData du répertoire courant.
 - `getwd()`
affiche le répertoire courant
 - `setwd("C:/introR/")`
définit le répertoire courant
 - `save.image()`
enregistre l'environnement de travail dans .RData
 - `load("example.RData")`
charge l'environnement de travail example.RData
- Commande d'aide en ligne : `help(sujet)`, ou `?sujet`

Importation de fichiers textes

R peut importer des fichiers textes (tab-delimited, CSV etc.) grâce à la fonction "read.table" :

```
read.table(file, header = FALSE, sep = "", quote = "\"'", dec = ".",
row.names, col.names, as.is = FALSE, na.strings = "NA",
colClasses = NA, nrows = -1,
skip = 0, check.names = TRUE, fill = !blank.lines.skip,
strip.white = FALSE, blank.lines.skip = TRUE,
comment.char = "#")
```

Ex : importation d'un fichier tab-delimited avec le nom des variables en première ligne :

```
example <- read.table(file="example.dat", header=TRUE, sep="\t")
```

Importation de fichiers de données externes

R peut également importer des fichiers SPSS/Stata/SAS/minitab etc. grâce à la librairie "foreign"

- Chargement de la librairie :
`library(foreign)`
- Chargement d'un fichier SPSS :
`donnees <- read.spss("example.sav", to.data.frame=TRUE)`
- Même principe pour les autres formats ->
`help(library="foreign")`

Exportation

- Exportation possible en fichier texte :
 - `write.table(donnees, file="export.txt", sep="")`
 - On perd les étiquettes, les variables facteurs deviennent des strings
- Ou avec "foreign"
 - `write.foreign(donnees, datafile="export.txt", codefile="export.sps", package="SPSS")`

Fonctions

```
discretize <- function(a) {  
  if (a < 5.1) { return(1) }  
  else {  
    if(a < 6) { return(2) }  
    else { return(3) }  
  }  
}
```