

Courte initiation à R

Data science
Master 2 ISIDIS

SÉBASTIEN VEREL

verel@lisic.univ-littoral.fr

<http://www-lisic.univ-littoral.fr/~verel>

Université du Littoral Côte d'Opale

Laboratoire LISIC

Equipe OSMOSE

Plan

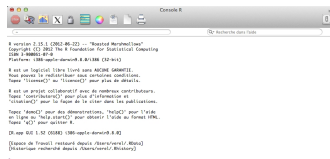
- 1 Bases de R
- 2 Données bi-variées

Logiciel R

Historique

- R est un logiciel libre (projet GNU), open source
- Langage et environnement dédié au traitement statistique et à la représentation graphique des données
- Clone du logiciel S+ (qui n'est pas libre)
- Disponible sous toutes les plateformes :
<http://www.r-project.org>
- Enormement de bibliothèques disponibles (très pointues), projet très dynamique
- Langage basé sur le calcul matriciel de la même famille que Matlab, Scilab

Environnement



```

R version 2.15.1 (2012-06-22) -- "Ropeful Mariposa"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-87-9
Platform: i386-apple-darwin8.0 (i386_64)

R est un logiciel libre livré sans ADOBE GADGETS.
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'informations et
"credits()" pour la liste de la liste dans les publications.

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

DR.app OUI 1.52 (5882) (386-apple-darwin8.0)
[Process de France] restauré depuis /Users/renel/.Rprofile
[Historique restauré depuis /Users/renel/.Rhistory]
>

```

- Les commandes peuvent être exécutées directement en les frappant au niveau du prompt
- Possibilité d'écrire des scripts (extension .R), lecture : `source("blabla.R", echo = T)`
- Pour quitter la session : `q()`
- L'historique peut être enregistré pour ne pas perdre les données (fichier `.Rhistory`)

Calculatrice

- le "top level" peut être utilisé comme une calculatrice :
`sqrt(3 * 3 + 4 * 4)`
`sin(pi / 2)`
- Notion de variable :
`a <- 2`
`a <- a + 1`
- Lister l'ensemble des variables de l'environnement :
`ls()`
- Effacer une variable de l'environnement :
`rm(a)`

Vecteurs

- Objets de base de R
- Suite d'éléments de même type :
 - numérique (réelles ou complexes),
 - booléen (TRUE, FALSE)
 - alphanumérique (chaîne de caractères)

```
c(elt1, elt2, elt3, ...)
```

Vecteurs

Création

- En extension (c pour combine) :
`c(100, 20, 50)`
vecteur composé des 3 entiers 100, 20 et 50.
- Par "répétition" d'un élément avec `rep` :
`rep("a", 20)`
vecteur composé de 20 éléments égaux au caractère "a"
- Par description d'une suite d'entiers consécutifs avec `:` :
`1:100`
vecteur composé des 100 premiers entiers
- Par description d'une suite d'entiers consécutif avec `seq` :
`seq(11, 31, 2)`
vecteur composé des nombres impairs entre 11 et 31.

Vecteurs

Longueur

- La longueur d'un vecteur est donné par `length` :
`length(rep(1, 100))` donne 100

Concaténation

- L'ajout d'éléments s'effectue de la manière suivante :
`u <- c(100, 20, 50)`
`u <- c(u, 500)`
donne : 100, 20, 50, 500

Tirage sans remise

- `sample` effectue un tirage sans remise des éléments :
`sample(1:100, 10)`
donne un vecteur constitué de 10 valeurs aléatoirement sélectionnées sans remise parmi le vecteur 1:100

Vecteurs

Lecture / écriture

- Pour accéder à un élément donnée :
`u[2]`
élément d'indice 2 du vecteur u
- Attention le premier élément est d'indice 1 !
`u <- c(100, 20, 50)`
`u[1]` a pour valeur 100
- Sous-vecteur :
`u[2:50]`
éléments de l'indice 2 à l'indice 50.
`u[c(2, 4, 6)]`
éléments d'indice 2, 4 et 6.
- Sous-vecteur avec sélection :
`u[u > 0]`
vecteur composé des éléments strictement positifs de u

Opération sur les vecteurs

- Les opérations sont rapides et ne nécessite pas de boucles :
 $u + v$
somme du vecteur u et v
- Opération sur chaque élément :
 $u * 2$
multiplie tous les éléments par 2
- Application de fonction :
 $\sin(u)$
applique la fonction sinus à tous les éléments
- Tests :
 $u <- c(100, 20, 50)$
 $u > 30$
donne le vecteur dont les éléments sont le résultat du test
TRUE FALSE TRUE

Nom des éléments

- Par défaut, les éléments sont indexés par des entiers
- Il est possible de donner des noms aux éléments :

```
u <- c(100, 20, 50)
```

```
names(u) <- c("un", "deux", "trois")
```

```
u["un"] a pour valeur 100
```

- Connaître les noms des éléments :

```
names(u)
```

```
"un", "deux", "trois"
```

Matrices

- Objets de base de R
- Tableau à 2 dimensions d'éléments de même type :
 - numérique (réelles ou complexes),
 - booléen (TRUE, FALSE)
 - alphanumérique (chaîne de caractères)

Matrices

Création

- En "colonne" :

```
A <- matrix(c(1:6), ncol = 2)
```

```
  1  4  
  2  5  
  3  6
```

- En "ligne" :

```
A <- matrix(c(1:6), ncol = 2, byrow=TRUE)
```

```
  1  2  
  3  4  
  5  6
```

Matrices

Lecture / écriture

```
A <- matrix(c(1:6), ncols = 2)
```

- Notation matricielle usuelle :

A[1,2] a pour valeur 4

- une ligne entière :

A[1,]

la première ligne

- une colonne entière :

A[,2]

la deuxième colonne

Matrices

Opérations

- Opérations terme à terme : $A + B$
 $A * B$
- Opérations matricielles usuelles :
 $A \% * \% B$

Listes

Liste

- Suite d'objets (vecteurs, matrices, etc.) qui ne sont pas nécessairement du même type
- Utilisées par de nombreuses fonctions en résultat

Listes

Création

- Utilisation du mot clé `list` :

```
maSuperListe <- list(je = c("pense", "suis"),  
  data = matrix(c(1:6) ncols = 2))
```

Listes

Création

- Utilisation du mot clé `list` :

```
maSuperListe <- list(je = c("pense", "suis"),  
data = matrix(c(1:6) ncols = 2))
```

Lecture / écriture

- Utilisation du signe `$` :

```
maSuperListe$je
```

 donne l'élément `je` de la liste
- Utilisation du double crochet :

```
maSuperListe[[1]]
```

 donne le premier élément de la liste (= `maSuperListe$je`)

Les data.frame

data.frame

- Liste de plusieurs vecteurs de même longueur
- Proche de la notion de tableau

Les data.frame : Création

Fichier csv

- Les fichiers d'extension `.csv` (comma-separated values) ont pour convention d'être des fichiers textes où les données sont enregistrées sous forme de tableau (ligne / colonne)
- Sur une même ligne, les données sont séparées par un caractère (espace, virgule, etc.)

```
initial final
155.209786 118.799252
367.658596 118.795867
405.252902 118.795936
...
```

Les data.frame : Création

Création à partir d'un fichier

- Les data frame peuvent se construire à partir d'un fichier csv :

```
frame <- read.table("data.csv", sep = " ",  
header = TRUE)
```

 - le premier argument est le nom du fichier
 - sep indique le caractère de séparation des données
 - header indique si la première ligne contient le nom des colonnes

Les data.frame : Création

Création à partir d'un fichier

- Les data frame peuvent se construire à partir d'un fichier csv :

```
frame <- read.table("data.csv", sep = " ",  
header = TRUE)
```

- le premier argument est le nom du fichier
- sep indique le caractère de séparation des données
- header indique si la première ligne contient le nom des colonnes

Lecture / écriture

Pour obtenir les vecteurs constituant le data frame :

- Utilisation du signe \$:
`frame$initial`
- Utilisation du double crochet :
`frame[[1]]`

Les fonctions

Fonction définie par l'utilisateur

```
maFonction <- function(arg1, arg2) {  
  bloc d'instructions  
}
```

- Le dernier résultat du bloc d'instruction est la valeur finale de la fonction

Exemple

```
plusUn <- function(x) {  
  x + 1  
}
```


Branchements et Itérations

si ... alors ... sinon ...

```
if (x < 0)
  - x
else
  x
```

Itération "pour"

```
for(name in c("un", "deux", "trois"))
  print(name)
```

Itération "tant que"

```
i <- 0
while(i < 100) {
  print(i)
  i <- i + 1
}
```

Statistique descriptive

- Moyenne :
`mean(u)`
- Variance :
`var(u)`
- Ecart-type :
`sd(u)`
- Minimum et maximum :
`min(u)`
`max(u)`
- Quartiles :
`quantile(u)`
- Résumé des principales statistiques :
`summary(u)`

Observation d'une distribution empirique

Histogramme

```
hist(u)
```

Boite à moustache

- Unique :
`boxplot(u)`
- Plusieurs vecteurs :
`boxplot(list(u, v))`

Loi normale

Loi normale ou loi gaussienne

- Loi adaptée à modéliser de nombreux phénomènes naturels aléatoires
- Complètement caractérisée par la moyenne μ et la variance σ^2
- Densité :

$$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

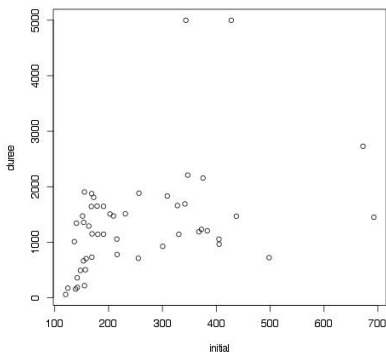
- Notation :

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

en R

- Obtenir des réalisations :
`rnorm(1000, 0, 1)`
1000 réalisations d'une loi normale de moyenne nulle et de variance 0.

Données bivariées



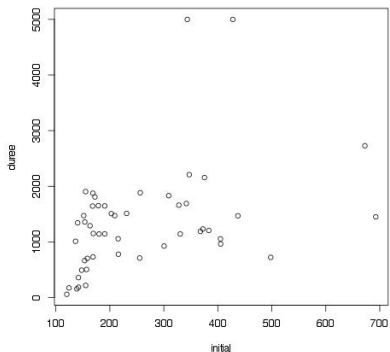
Définition

Ensemble de couples de données
 $\{(x_1, y_1), (x_2, y_2), \dots\}$

en R

- Si x et y sont des vecteurs de même longueur (abscisses et ordonnées) :
`plot(x,y)`
- Si `frame` est un `data.frame` (`initial` et `duree` sont des colonnes) :
`plot(duree ~ initial, frame)`

Statistique inférentielle

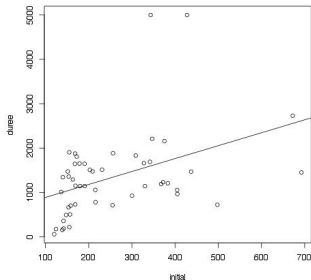


Questions

Existe-t-il une relation entre les variables x et y ?

Laquelle ?

Modèle linéaire



Modèle linéaire

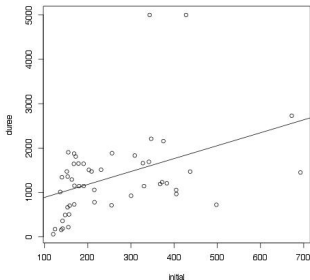
Le nuage de point est remplacé par une droite :

$$y = a.x + b$$

On peut alors :

- Expliquer la relation entre les variables
- Prédire les valeurs

Modélisation et erreur



Erreur quadratique moyenne (MSE)

Erreur du modèle est la distance entre observations et prédictions :

$$MSE = \frac{1}{n} \sum_{i=1}^n (f(x_i) - y_i)^2$$

D'autres mesures de distance sont possibles...

Modélisation et erreur

Ajustement du modèle

$$f(x) = a.x + b$$

- Choisir les paramètres du modèle pour minimiser l'erreur
- Choisir a et b pour minimiser l'erreur MSE
- L'erreur est alors égale au coefficient de corrélation (au signe près) :

$$\rho = \frac{\text{cov}(x, y)}{\sigma_x \sigma_y}$$

Modèle linéaire en R

Fonction `lm`

- Utiliser la fonction `lm` (linear model) :
 - Avec `data.frame` :
`lin <- lm(y ~ x, frame)`
 - Avec vecteurs de même longueur :
`lin <- lm(x, y)`
- Pour avoir le descriptif du modèle :
`summary(lin)`
- En particulier, on peut lire la p-value du modèle :
Si p-value < 0.05, le modèle linéaire est significatif
- Pour avoir les coefficients *a* et *b* :
`lin$coefficients`

Corrélation R

- Utiliser la fonction `cor` :
 - Avec `data.frame` :
`cor(frame)`
On obtient une matrice de corrélation entre toutes les variables du `data.frame`
 - Avec vecteurs de même longueur :
`cor(x, y)`
- Si le coefficient est négatif, les variables sont anti-corrélées (variation inverse)
- Un fort coefficient est supérieur à 0.8 ou 0.9 (en valeur absolue)
- Un faible coefficient est compris entre -0.2 ou 0.2