Logiciel de Statistiques R

Didier Chauveau

UFR CoST – Université d'Orléans







Données statistiques = table individus-caractères

On "pose" p "questions" (mesures) à n "individus" = unités statistiques (personne, animal, item, jour, lieu...)

Données sous la forme d'une **Table individus-caractères** table ou matrice $(n \times p)$ souvent $n >> p$

$$\begin{bmatrix} X_1^1 & \cdots & X_1^s & \cdots & X_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ X_\ell^1 & \cdots & X_\ell^s & \cdots & X_\ell^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ X_n^1 & \cdots & X_n^s & \cdots & X_n^p \end{bmatrix}$$

- La ℓ-ième ligne $\mathbf{X}_\ell$ est la "réponse" de l'individu ℓ :
- La s-ième colonne $\mathbf{X}^s$ est le s-ième **caractère** ou **variable statistique**

Format des données pour les méthodes de Data Mining

Nature des variables statistiques

Une variable (colonne) de la table individus-caractères peut être de 2 natures :

- **Qualitative** : (facteur)
  - à valeur dans un ensemble fini de *modalités*
  - pas de relation d'ordre entre les modalités
  - pas d'opérations numériques entre modalités

  exemples : CSP, groupe sanguin, région, code postal...
- **Quantitative** : à valeur dans $\mathbb{R}$ (ou $\mathbb{N}, \mathbb{Z}$)
  exemples : mesure physique, revenu, taux de CO^2...
- **Qualitative ordonnée**, variables pouvant avoir les deux statuts (nb d'enfants d'un ménage, classement subjectif d'un parfum...)

Statistiques descriptives ou inférentielles : panorama

Descriptif, exploratoire

Synthèse, résumé des données
Pas d'hypothèse de modèle

résumés graphiques et numériques

Techniques élémentaires permettant de croiser $p \leq 4$ variables

Méthodes de "Data Mining" pour p grand : ACP, AFC, ACM, Clustering...

Inférentiel

Inférence statistique
Modèle statistique = type de loi qui gouverne le phénomène
une v.a. $X \sim F_\theta$, paramètre θ
ex : $X \sim$ loi Normale $\mathcal{N}(\mu, \sigma^2)$

Estimation $\hat{\theta}$
Intervalle de Confiance Décision : tests

Modèles souvent limités à p petit

Genèse : Les origines de R

- Initié par **R**oss Ihaka et **R**obert Gentleman (1993)
  (University of Auckland, New Zealand)
- Implémentation du langage "S" (Chambers et al. 1984–1998)
  `S-PLUS` est une autre implémentation, commerciale, du langage S (Insightful Corporation)
- ACM Software System Award, 1998.
- De facto devenu :
  - le standard de la recherche en "computational statistics" et exploration de données
  - l'environnement idéal pour populariser les nouvelles méthodologies en statistique

Quelques avantages techniques de R

- Gratuit (GPL2) et Open Source (écrit en C)
- Multi-plateforme (UNIX, LINUX, MacOS X, Windows...)
- Développé et maintenu par les meilleurs experts en "Statistical Computing" : The R Core Team
  (B. Ripley, L. Tierney, J. Chambers...)
- Langage interactif, orienté objet, extensible
- Pensé pour l'exploration et la modélisation de données
  (à la différence de e.g. MATLAB, Scilab,...)
- Interface simple vers du code `C`, `Fortran` si besoin
- Outils de calcul parallèle accessibles

"Philosophie" de R

Une analyse statistique implique :

- Exploration des données (résumés numériques, graphiques, modèles)
- Choix des outils guidés par la visualisation des données
- Possibilité d'adapter les outils existants : l'écriture de fonctions est naturelle

Packages R et sites CRAN

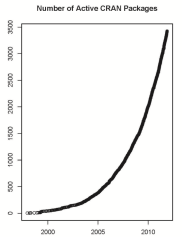
■ **Package :** ensemble de fonctions liées à un thème

- R inclut un environnement de développement
  - code R, C, ...
  - écritude de documentation "LaTeX-like"
- package testé et documenté avant mise en ligne
- un package peut *dépendre* ("hériter") d'autres packages
- Les packages "validés" sont accessibles via les sites miroirs CRAN = **C**omprehensive **R** **A**rchive **N**etwork

Comprehensive R Archive Network

`cran.r-project.org/web/packages`

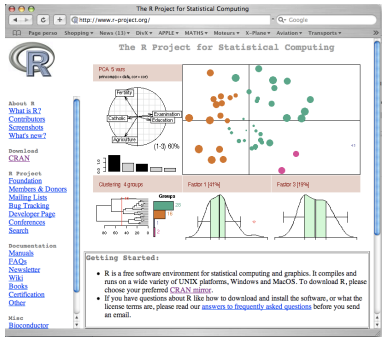
Nombre de packages actifs



K. Hornik, R core team 2012.

- Indicateur de l'activité en *statistical computing*
- Problèmes : qualité des packages, metadata,...

Homepage `www.r-project.org`



Comprehensive **R** **A**rchive **N**etwork et packages

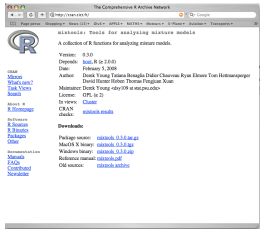
CRAN

`cran.r-project.org/web/packages`



Exemple : Package mixtools

`cran.cict.fr/web/packages/mixtools`



Pour s'y retrouver : CRAN Task Views

CRAN Task Views

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
Finance	Empirical Finance
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics
OfficialStatistics	Official Statistics & Survey Methodology
Optimization	Optimization and Mathematical Programming
Pharmacokinetics	Analysis of Pharmacokinetic Data
Phylogenetics	Phylogenetics, Especially Comparative Methods
Psychometrics	Psychometric Models and Methods
ReproducibleResearch	Reproducible Research
Robust	Robust Statistical Methods
SocialSciences	Statistics for the Social Sciences
Spatial	Analysis of Spatial Data
SpatioTemporal	Handling and Analyzing Spatio-Temporal Data

D'autres extensions/outils construits sur

RStudio : GUI (voir TPs) `https://www.rstudio.com`

Shiny : Interface web dynamique de visualisation
`http://shiny.rstudio.com/gallery/`

R Markdown : générateur de documents (pdf, html,...) incluant code LaTeX et code dynamique

hadoop MapReduce : package RHadoop pour le BigData

Objets

Structures comme tous les langage "de haut niveau" :

Vecteur, matrice, tableau multidim, objet structuré...

Structures spécifiques :

Classes d'objets spécialisés pour les statistiques, eg :

- `factor` : vecteur à valeurs *modalités* sans ordre
  = facteur qualitatif (e.g. Région, CSP, sexe,...)
- `data.frame` : liste de vecteurs de même longueur, de classes quelconques
  = table individus-caractères

Classes et méthodes

Langage orienté objet

Les fonctions peuvent disposer de méthodes adaptées aux classes de leur argument

Une fonction se comporte différemment suivant la classe de l'argument avec lequel on l'utilise.
Exemple : méthodes définies pour la fonction `summary` qui résume un objet :

```
> methods(summary)
[1] summary.aov          summary.aovlist     summary.aspell*
[4] summary.connection   summary.data.frame  summary.Date
[7] summary.default      summary.ecdf*       summary.factor
 ...
```


Objets

Comme tout langage de haut niveau :

- `numeric` : valeur numérique
- `vector` : collection d'objets de même `mode` (type)
- `matrix` : tableau de dimension 2
- `array` : tableau de dimension $d \geq 2$

Mais aussi :

- `factor` : vecteur à valeurs *modalités*
- `list` : collection d'objets de types quelconques
- `data.frame` : liste de vecteurs de même longueur

Langage spécialisé pour les statistiques

data frame = structure d'une table individus-caractères
factor = facteur qualitatif (sexe, groupe sanguin, CSP...)

Pour commencer

Informations générales

```
# ceci est un commentaire
R.version  # infos techniques...
citation() # pour bibliographie d'articles...

citation   # sans "()", définition de la fonction!
```

aide en ligne

```
help()      # aide de l'aide
?citation   # aide de la fonction citation
?"+"        # aide pour un opérateur
```

Chargement de packages et démo de fonctions

```
install.packages("ade4") # download & install
library(ade4)            # charge ce package
example(dudi.pca)        # essayer! (data mining: ACP)
```


Éléments et vecteurs

Calculette, affectations

```
2 + 2
x <- 1; y <-2 # ";" sépare les commandes
x = 4         # "presque toujours" equivalent à x <- 4
rm(x)         # suppression de x du "workspace"
```

Création de vecteurs

```
a <- c(1,2,5,8,9)     # "c"ombine: différent de MATLAB!
c <- c("toto","tu")   # vecteur de caracteres (deux c...)
length(a)             # longueur du vecteur

a = rep(1,10)         # répétition, voir ?rep
b = rep(c(10,12),5)   # idem sur (10,12)
s = 1:10              # boucle (cf Matlab, Scilab)
s = seq(1,10,by=2)    # séquences, voir ?seq
seq(1,10,length=20)   # discrétisation en 20 points

seq(1,10,len=20)      # idem, "Partial Matching" !!!
```

Règles concernant les arguments de fonctions

Exemple : définition de la fonction

```
seq(from = 1, to = 1, by = ((to-from)/(length.out-1)),
    length.out = NULL, ...)
```

- pour les arguments fournis sans nom l'ordre compte :
  `seq(1,10,2)` ⇔ `seq(from=1, to=10, by=2)`
- pour les arguments nommés l'ordre ne compte pas :
  `seq(to=10, by=2, from=1)` est valide
- les arguments non précisés prennent les valeurs par défaut indiquées dans l'aide de la fonction :
  `seq(to=10, by=2)` est ⇔ à `seq(1,10,2)`
- le "partial matching" permet de ne spécifier que partiellement le nom d'un paramètre
  "`len =`" ⇔ "`length.out =`", car pas d'ambiguité

Manipulations sur les vecteurs (1)

Opérations

```
a+b; a*b; a/b   # "élément-wise", même longueur
a<b             # test element-wise
log(b)          # ft math element-wise

a+s             # pas de même longueur: "recycling" !
```

Quelques fonctions statistiques élémentaires

```
min(a)          # minimum
mean(a)         # moyenne empirique
sd(a)           # écart-type (standard deviation)
sort(a)         # tri
```

Échantillonnage dans un ensemble

```
a=sample(1:10) # par défaut permutation de 1:10
?sample        # aide, arguments...
sample(a,2)    # tirage de 2 éléments de a
```


Manipulations sur les vecteurs (2)

Extraction d'éléments :

```
e = seq(1,20,by=2)
e[3]            # élément d'un vecteur
e[1:5]          # 1:5 vaut [1,2,3,4,5] (cf scilab)
e[c(3,6,8)]     # sous-vecteur explicite
e[-3]           # tout sauf e[3]
e[1:5][4]       # élément 4 du vecteur e[1:5]
e[1:5][2:4][1] # devinez les résultats! (QCM...)
```

Opérateurs logiques

```
a <- c(T,T,F,F)      # T ou TRUE, F ou FALSE
!a                   # not a
b <- c(T,F); a & b   # ET logique, b recyclé ici
# extraction de sous-vecteurs par tests
e[e > 5]             # seules les valeurs testées à TRUE
e[e != 5]            # sauf les e[i]=5
```


Manipulations sur les matrices

Construction et extraction

```
a=1:10               # on reprend
?matrix              # méthode par défaut?
m = matrix(a,nrow=4) # avec recycling (warning)
dim(m)               # voir aussi nrow(m) et ncol(m)
m[,3]                # 3ème colonne de m
m[2,3]               # élément m(2,3)
class(m)             # différent de mode(m)
```

Opérations

```
p = matrix(a,ncol=2)  # par défaut byrow = FALSE
q = matrix(a,nrow=2)
q %*% p               # produit de matrices
q %*% m               # pb dimensions
diag(m)               # diagonale, même si pas carrée
t(m)                  # transpose
apply(m,1,sum)        # opération (sum) par ligne (1)
```


Exercices

1. Construire une matrice M à 10 lignes et 5 colonnes constituée d'entiers tirés au hasard dans $\{0,\ldots,9\}$
2. Calculer la moyenne des colonnes de M
3. Calculer l'écart-type des lignes de M

Manipulations sur les listes

Une liste est une collection d'objets

- ordonnés
- de types (`mode`, `class`) quelconques

Construction et extraction

```
lsn <- list(1:5, "toto", T)       # éléments sans noms
ls <- list(x=1:5,nom="toto",z=T) # idem avec noms
summary(ls) # default method: noms, classes et modes
ls          # liste du contenu
lsn[[1]]    # extraction élément 1 de la liste
ls$x        # opérateur $ lorsque le nom est connu
```


Fonctions génériques, méthodes et classes

Fonctions qui s'adaptent à leurs arguments

Certaines fonctions ont des méthodes dont le résultat dépend de la classe de leurs arguments.

Exemple : résumé de structures

```
class(a); summary(a) # a est "numeric"
class(c); summary(c) # c est "character"
summary(ls)          # résumé d'une liste (default)
methods(summary)     # méthodes associées
```

Exemple : graphiques élémentaires (et beaucoup d'autres...)

```
plot(a)              # basic: série des observations
x <- rnorm(100); y <- x + rnorm(100) # x iid ~ N(0,1)...
plot(x, y, pch=20)   # nuage de points
```


data.frame = table individus-caractères

`data.frame` = liste particulière cad collection de (p) vecteurs :

- de types (`mode`, `classe`) quelconques
- de même longueur (n)

Les data.frame peuvent se manipuler comme des listes, mais aussi comme des tableaux

C'est une structure fondamentale pour la finalité de R : manipuler et analyser des tableaux de données.

Data.frame = table individus-caractères

Construction à la main

```
a=sample(1:10); b=rep(c(1,2),5);
c = c(rep("M",6),rep("F",4)) # facteur qualitatif
x = data.frame(a,b,c)        # création de la structure
class(x)                     # = data.frame
class(c); class(x$c)         # conversion par défaut
```

Manipulations de base

```
names(x)        # noms des colonnes = variables
row.names(x)    # noms des lignes = individus
x; x$a; x$a[2] # affiche x, la variable a, extraction
x[[1]]          # idem car c'est une liste!
```

→ Poursuivre avec des données réelles...

Exemple jouet : Données *State Facts*

Source : *Bureau of the Census, US (1977).*

Sélection de 7 variables :

- `Etat` Noms des "individus" = 50 états (code à 2 lettres)
- `Pop` Population estimée en 1975
- `Revenu` Revenu moyen par habitant
- `Apb` taux d'analphabétisme (en % de population)
- `Meurtre` taux de criminalité pour 100 000 habitants
- `Diplome` % de population de niveau équivalent au Bac
- `Region` région d'appartenance des états (Northeast, South, North Central, West)

L'un des nombreux jeux de données inclus dans R

Importation de données (1)

Fichier texte "brut" des données State Facts

```
Etat    Pop     Revenu  Apb     Meurtre Diplome Region
AL      3615    3624    2.1     15.1    41.3    South
AK      365     6315    1.5     11.3    66.7    West
AZ      2212    4530    1.8     7.8     58.1    West
...
```

1. Récupérer les données "texte" en local ou en ligne
   `StateFacts_tuto.txt`
2. Sauver le fichier dans votre répertoire de travail
3. Indiquer à R le répertoire de travail, cf menu
   `Fichier → Changer de répertoire de travail...`

Importation de données (2)

Importation sous forme de data.frame

```
states <- read.table("StateFacts_tuto.txt",
          header=T,     # ligne 1 = noms des variables
          row.names=1) # labels individus colonne 1

ls()            # liste des objects chargés/existants
head(states)    # affiche "le début" d'un objet
```

Méthode de travail après import de données :
Sauvegarde d'un dataframe au format R binaire compressé

```
save(states, file="StateFacts_tuto.Rdata")
```

Chargement d'un dataframe au format R binaire compressé

```
load("StateFacts_tuto.Rdata")
```


Premières manipulations

Taille de la table, attributs...

```
dim(states)      # p=6 variables + labels individus
nrow(states)     # n, aussi par dim(states)[1]
attach(states)   # rend "visible" les variables de states
detach(states)   # opération inverse

colnames(states)  # noms des variables
row.names(states) # labels individus
```

Extractions...

```
states$Pop       # car c'est aussi une "list"
states[[2]]      # car structure ordonnée
states[1,3]      # car c'est aussi une "matrix"

class(states$Region) # facteur créé au chargement
```


Description d'un caractère qualitatif

Variable X, à K modalités $X \in E = \{x_1, \ldots, x_K\}$
sans relation d'ordre
Ex : Region $\in$ {North_Central, Northeast, South, West}

- **Résumé numérique :** Table de *fréquences* = comptages

$$n_i = \sum_{\ell=1}^{n} \mathbb{I}_{\{X_\ell = x_i\}} = \#\text{ réponses } x_i \quad \text{et} \quad f_i = \frac{n_i}{n}, \quad i = 1, \ldots, K$$

- **Interprétation** : Le vecteur $(f_1, \ldots, f_K)$ est une probabilité sur E, construite à partir des observations

$$0 \le f_i \le 1, \quad \text{et} \quad \sum_{i=1}^{K} f_i = 1$$

c'est la loi empirique de X : $f_i \equiv \mathbb{P}(X = x_i)$

Une variable qualitative (facteur)

Table de fréquence d'un facteur

```
# tables de comptages à 1, 2,... facteurs
table(Region)        # table des n_j

t <- table(Region)   # objet de classe "table"
prop.table(t)        # table des f_j
```

Représentation graphique

```
plot(Region)              # method plot.factor
plot(Region, col=2)       # voir options de plot()
pie(table(Region))        # camembert, peu conseillé!
```


Exercices

En une seule commande à chaque fois :

1. Afficher la moyenne empirique de la variable `Revenu`
2. Afficher l'écart-type de `Meurtre`
3. Combien y-a-t-il d'États de Revenu moyen > 5000 ?
4. Afficher les moyennes empiriques des variables quantitatives de la table `states`
5. changer dans la table le nom de la variable `Pop` en `Population` (attention cf. `attach()...`)
6. Calculer $\sum_{i=1}^{n} X_i^2$ pour la variable X =`Population`

Exemples de statistiques (résumés) numériques

Fonctions agissant sur les variables

```
mean(Apb)
var(Meurtre)
max(Revenu) # statistiques numériques...
```

Fonctions agissant sur le data.frame

```
summary(states)  # min, max, quantiles...
sd(states)       # pas défini pour un facteur!
sd(states[,-6])  # ni pour un data.frame...
cor(states[,-6]) # corrélations, sauf facteur Region
```

Statistiques par niveaux de facteur(s)

```
tapply(Meurtre,Region,mean)  # mean(Meurtre) par Region
by(states[,-6], Region, colMeans) # pour le dataframe
```


Distribution empirique d'une variable quantitative

Variable $X \in E \subseteq \mathbb{N}$, ou $\mathbb{R}$
Ex : Taux de criminalité `Meurtre` à valeur ≥ 0

Résumé graphique : l'histogramme = Discrétisation de l'intervalle $[\min(X_i); \max(X_i)]$ en k classes $C_1, \ldots, C_k$, et représentation des

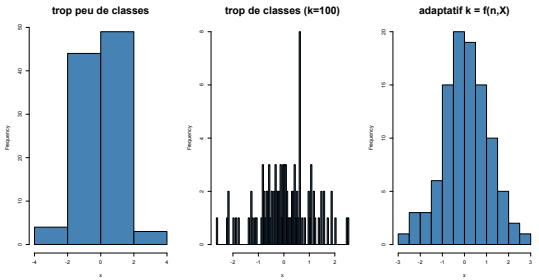
$$C_j \mapsto n_j = \sum_{i=1}^{n} \mathbb{I}_{X_i \in C_j} \quad \text{ou bien} \quad C_j \mapsto f_j = \frac{n_j}{n}$$

C'est une représentation de la loi empirique de X :

$$f_j \equiv \mathbb{P}(X \in C_j), \quad j = 1, \ldots, K$$

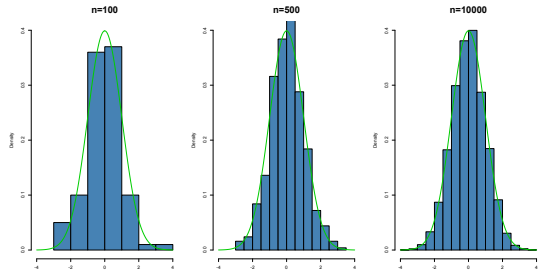
Distribution empirique d'une variable quantitative

Problème : choix du nombre de classes $k = k(n)$
Exemple sur $n = 100$ données gaussiennes $\sim \mathcal{N}(0,1)$:



Convergence de l'histogramme vers la densité

Si on admet que $X_1, \ldots, X_n$ est un n-échantillon d'une loi de densité f, on peut normaliser l'histogramme de sorte qu'il converge vers f.
Exemple simulé iid $\sim \mathcal{N}(0,1)$:



Histogrammes dans R

Variables quantitatives

```
hist(Meurtre)                  # tout par défaut

par(mfrow=c(1,2))              # 2 subplots
hist(Meurtre, col=2)           # ou col="red"
hist(Meurtre, col=3, freq=F)   # normalisé sur la densité

hist(Meurtre, col=8, freq=F,   # comparé avec autre algo
     breaks="Scott")           # de calcul nb de classes
```

Résumés par histogrammes pour plusieurs variables
→ exemple d'usage d'une boucle :

Usage de boucles et programmes. . .

Exemple : afficher les histogrammes des 5 variables quantitatives

```
par(mfrow=c(2,3))  # tableau de 2 x 3 graphiques
for (j in 1:5) {
  hist(states[,j], col=8, xlab="",  # j-ième colonne
       cex.main=2,cex.axis=1.5,     # taille caractères
       main=colnames(states)[j])    # nom j-ième var.
        }
```


Nuages de points *scatterplots* (1)

Quelques graphiques élémentaires

```
plot(Revenu)              # X(i) contre indices i=1,...
plot(Apb,Meurtre)         # nuage de points (scatterplot)
```

Quelques options (voir `par()`)

```
plot(Apb,Meurtre, pch=3,              # type de point
     xlab="Taux d'analphabétisme",    # label axe x
     ylab="Taux de criminalité",      # label axe y
     main="nuage de points")          # titre
```

D'autres méthodes de `plot`

```
plot(states[,-6])             # method plot.data.frame
plot(~Pop+Revenu+Meurtre)     # d'une liste explicite
par(mfrow=c(1,2))             # subplots
plot(Meurtre ~ Apb + Diplome) # method plot.formula
```


Nuages de points (2)

Nuages avec labels individus

```
plot(Apb,Meurtre,type="n",
     main="Meurtre par taux d'analphabétisme")
text(Apb,Meurtre,row.names(states))
```

Nuages avec modalité d'un facteur

```
plot(Apb,Meurtre,type="n") # points "invisibles"
text(Apb,Meurtre,Region)   # effet région ?
# conversion en modalités brèves
R2 <- factor(Region,labels=c("C","E","S","W"))
plot(Apb,Meurtre,type="n")
text(Apb,Meurtre,R2,col=as.numeric(R2))  # plus clair
```


Nuages de points (3)

Nuages avec barycentres et ellipses d'inertie par modalités d'un facteur :

```
library(ade4)       # package D. Chessel Lyon I

couleurs <- seq(1,length(levels(R2)))
s.class(data.frame(Apb,Meurtre),R2,
    cstar=0, cpoint=0, clabel=0.8, col=couleurs,
    axesell=F, add.plot=T)     # ajouter au graphique
```

Sous-nuages par modalités de facteur(s)

```
coplot(Meurtre ~ Apb | Region) # conditionne par facteur
coplot(Meurtre ~ Apb | Diplome, # | va quantitative
    number=4, overlap=0)        # en 4 classes
```


Tris de structures et dataframe dans R

Objectif : trier un tableau de données suivant un ou plusieurs critères.

Exemple jouet

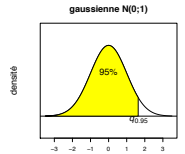
```
x <- c(3,10,9,4)
sort(x)
o <- order(x) # permutation
o
x[o] # x[o[i]] = la ième plus petite de x
```

Exercices :

1. Quelles sont les 10 plus fortes valeurs de Revenu ?
2. Quel est l'individu-ligne (État) de Revenu minimum ?

Quantiles empiriques

Rappel : quantile théorique Pour une variable aléatoire réelle $X \sim F$ loi à densité, le quantile d'ordre $\alpha \in]0,1[$ de X est q_α tel que $\mathbb{P}(X < q_\alpha) = \alpha$



Pour une suite d'observations $\mathbf{X} = (X_1, \ldots, X_n)$, on note $X_{(1)} < X_{(2)} < \cdots < X_{(n)}$ la suite des valeurs triées.

Définition
Le quantile empirique d'ordre $\alpha \in]0,1[$ d'une suite d'observations $\mathbf{X}$ est la $[n\alpha]$-ième valeur de la suite $\mathbf{X}$ triée.

On le note Q_α

Quantiles empiriques

Exemple : pour $n = 100$ observations :

- la *Médiane* est $M = Q_{0.5} = X_{(50)}$
- le 1er *quartile* est $Q_{0.25} = X_{(25)}$

Quelques dénominations usuelles :

- Quartiles : $Q_{0.25}, Q_{0.50}, Q_{0.75}$
- Ecart InterQuartile $IQR = Q_{0.75} - Q_{0.25}$
- Déciles : $Q_{0.10}, \ldots, Q_{0.90}$
- Percentiles : $Q_{0.01}, \ldots, Q_{0.99}$

Représentation : Boxplots

Représentations des quantiles : boxplots

Variables individuelles

```
boxplot(Meurtre)
boxplot(states[,1:3],
    main="boxplots de 3 variables")  # pb échelles!
```

Boxplots par modalité de facteur(s)

```
boxplot(Meurtre ~ Region);
plot(Region,Apb)                     # idem !!
```

Exemple de comparaisons

```
par(mfrow=c(2,1))                    # subplots
boxplot(Meurtre ~ Region,varwidth=T, # largeur variable
    main='Meurtre par Region')
boxplot(Diplome ~ Region,varwidth=T)
title('Diplome par Region')          # comme main=...
```


Graphiques “radar” ou “segments”

Exemple sur toutes les variables qualitatives

```
stars(states[,-6])              # basic, tous les individus
stars(states[1:10,-6],          # sélection d'individus
    key.loc=c(6,2))             # avec légende
```

Segment plots

```
stars(states[1:10,-6],          # sélection d'individus
    key.loc=c(6,2),             # avec légende
    draw.segments=T)            # "segment" plot
```

NB : peu utile dès que n grand car ce ne sont pas des résumés d'une population !

Quelques Références

- Chessel, D., Dufour, A.-.B., and Thioulouse J. (2004) : *The ade4 package-I- One-table methods*. R News. 4 : 5-10.
- Chambers, J.M. (1998), *Programming with Data*. New York : Springer.
- Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth edition. Springer.
- R Wiki `http://wiki.r-project.org`
- R graphics gallery `http://addictedtor.free.fr/graphiques`
- and much more !...

Utiliser 