Initiation à R via RStudio

Université Assane SECK de Ziguinchor

UFR des Sciences et Techniques

Département Informatique

Licence 2 : Ingénierie Informatique

Sommaire

Sommaire

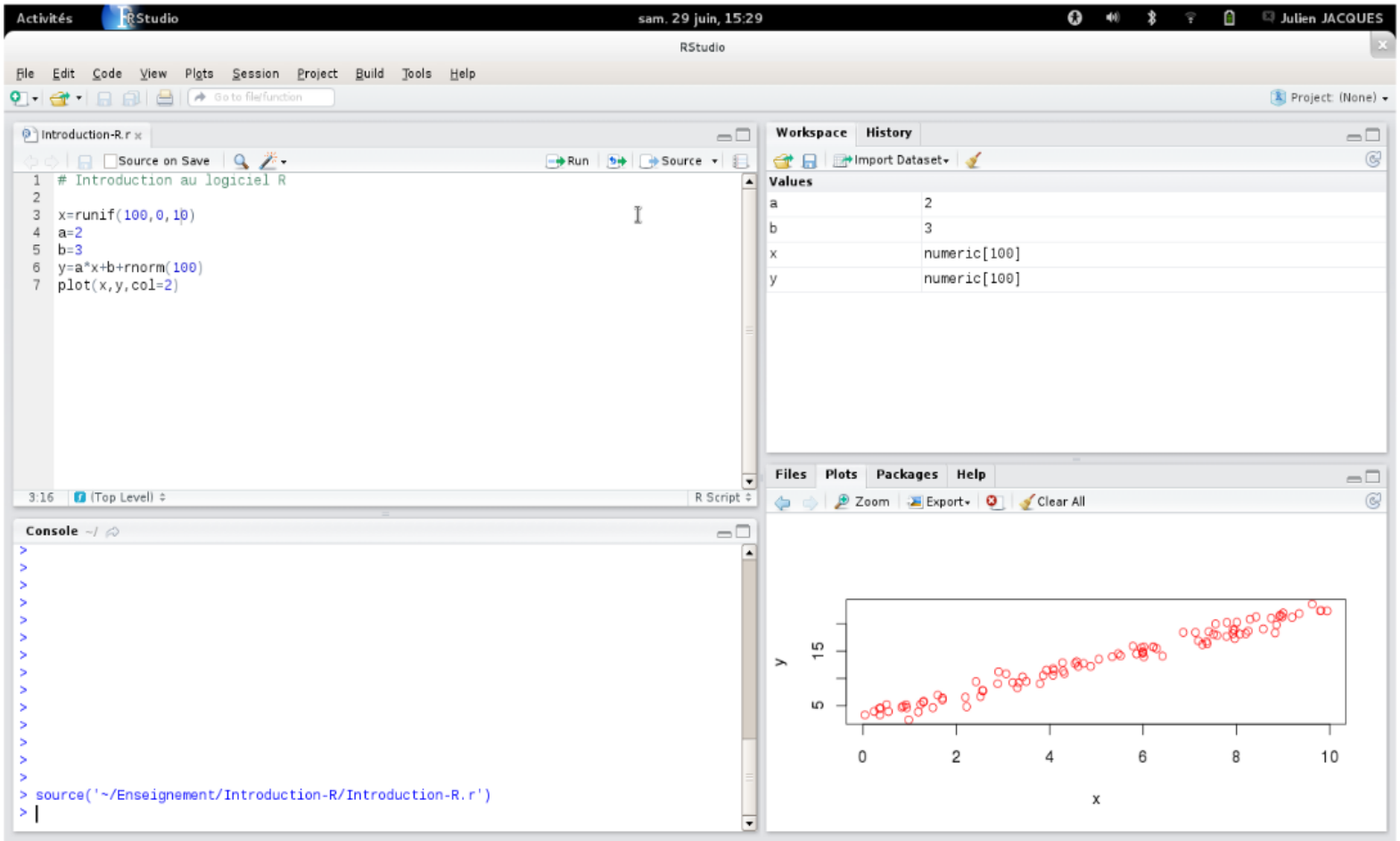
Le logiciel R (disponible sur http ://www.r-project.org/) est un logiciel libre qui permet de réaliser des analyses statistiques, et ayant un certain nombre d'atouts :

- il permet l'utilisation des méthodes statistiques classiques à l'aide de fonctions prédéfinies,
- il permet de créer ses propres programmes dans un langage de programmation assez simple d'utilisation.
- il permet d'utiliser des techniques statistiques innovantes et récentes à l'aide de package développés par les chercheurs et mis à disposition sur le site du Comprehensive R Archive Network (CRAN) (http ://cran.r-project.org/).
- Il existe des versions : Windows 95 et plus, Linux, MacOS

Le logiciel R fonctionne initialement en ligne de commande, mais des interfaces permettent désormais une utilisation plus conviviale.

Sommaire

Nous proposons ici de travailler avec l'interface RStudio, téléchargeable sur le site http ://www.rstudio.com/



L'interface est composée de 4 fenêtres :

- Fenêtre d'édition (en haut à gauche) : dans cette fenêtre apparaissent les fichiers contenant les scripts R que l'utilisateur est en train de développer. En entête de cette fenêtre, des icônes permettent de sauvegarder le fichier, d'exécuter un morceau de code sélectionné (icône run) ou l'intégralité du code contenu dans le fichier (icône source).

- Fenêtre de commande (en bas à gauche) : cette fenêtre contient une console dans laquelle les codes R sont saisis pour être exécutés.

- Fenêtre espace de travail / historique (en haut à droite) : contient les objets en mémoire, que l'on peut consulter en cliquant sur leur noms, ainsi que l'historique des commandes exécutées,

- Fenêtre explorateur / graphique / package / aide (en bas à droite)

Sommaire

☞ Le répertoire de travail est celui à partir duquel vous avez lancer l'interface RStudio. Pour connaître ce répertoire on utilise la commande getwd().

☞ Il sera pratique de se placer dans un répertoire de travail bien défini. Pour ce faire, vous pouvez soit utiliser la commande setwd pour vous déplacer dans l'arborescence des répertoires, soit utiliser le menu de l'interface :

- Session
  - Set Working Directory
    - Choose Directory

Par la suite, lorsque vous serez amené à charger des jeux de données, si ceux-ci sont placés dans le répertoire courant dans lequel vous vous êtes placé, vous n'aurez pas à saisir le chemin complet de ce répertoire.

Sommaire

☞ Toutes les fonctions de R sont stockées dans une grande bibliothèque. Cette bibliothèque contient des packages (ou librairies) de fonctions. Toutes les librairies ne sont pas chargées au lancement du logiciel. On peut charger un package à l'aide de la commande :

R > library("nom librairie")

☞ Il est possible d'installer des packages supplémentaires. Pour cela, lorsque vous disposez d'une connexion internet, il suffit d'utiliser la commande suivante en indiquant le nom du package que l'on veut installer :

R > install.packages("nom librairie")
RStudio vous proposera alors de choisir le serveur à utiliser pour télécharger le package et procédera ensuite à l'installation. Il faudra ensuite charger le package à l'aide de la commande library().

Sommaire

Sommaire

- R peut être utilisé pour réaliser des opérations élémentaires :

$R > ((1 + sqrt(5))/2)^2$

- dont le résultat peut être stocké dans une variable

$R > a = ((1 + sqrt(5))/2)^2$

gardée en mémoire (a apparaît alors dans la fenêtre espace de travail), et qui peut être ré-utilisée par la suite :

$R > b = sqrt(a)$

- Pour effacer les variables en mémoire dans la session R, il faut taper la commande suivante :

R> rm(list=ls())

Sommaire

☞ La variable a définie ci-dessus est un scalaire. R gère également des vecteurs et matrices. Un des avantages de R est qu'un grand nombre d'opérations et de fonctions sont applicables directement sur des vecteurs (voir sur des matrices). Ainsi, le recours au boucle de type for peut être évitée, et doit généralement l'être pour des raisons de rapidité d'exécution.

☞ Pour créer un vecteur, il est possible d'utiliser la fonction de concaténation c :

```
R > x = c(7, 8, 9)
R > y = 1 :3
R > z = rep(x, y)
```

- La commande matrix permet de créer une matrice

```
R > M1 = matrix(z, 2, 3)
```

- ce qui peut également être fait en concaténant des vecteurs en ligne (rbind) ou en colonne (cbind) :

```
R > M2 = cbind(x,y)
```

```
R > M3 = rbind(x,y)
```

- Les tableaux à plus de 2 dimensions, appelés array en R, sont également utilisables :

```
R> T = array(0,dim=c(2,3,4)
```

Extraire les éléments d'une matrice

☞ Soit Mat la matrice de dimensions $m \times n$. La commande $Mat[i, j]$ permet d'extraire l'élément qui est à la i-ième ligne et j-ième colonne.

> vect=c(1.5 :9.5)
> Mat=matrix(vect,ncol=3,nrow=3)
> Mat[2,3] affiche l'élément qui est à la 2ième ligne et 3ième colonne.
> Mat[,1] affiche la première colonne
> Mat[3,] affiche la troisième ligne

☞ Les opérations arithmétiques et les fonctions mathématiques fonctionnent éléments par éléments.

Quelques fonctions sur les matrices

- Le produit matriciel est obtenu avec $\%*\%$
- Calcul du déterminant : det()
- t(A) retourne la transposée de la matrice A
- solve(A) retourne l'inverse de la matrice A
- solve(A,b) retourne x tel que $Ax = b$
- dim(), ncol(), nrow() retournent respectivement la dimension de la matrice, le nombre de lignes et le nombre de colonnes.

Sommaire

☞ R dispose d'un grand nombre de fonctions prédéfinies, utilisables en appelant la fonction par son nom suivi de ses arguments entre parenthèses :

```
R> mean(x)
R> rnorm(10)
R> rnorm(10,mean=1,sd=2)
```

☞ Astuce : lorsque vous commencez à taper le nom de la fonction, vous pouvez en appuyant sur la touche tabulation voir les différentes fonctions commençant par les lettres déjà saisies. Lorsque le nom de la fonction est totalement saisi, la tabulation permet de voir les arguments attendus par la fonction.

☞ Il est également possible de créer ses propres fonctions.

Structure générale pour créer des fonctions

- La structure générale d'une fonction est
  mafonction = function (listes des paramètres)
  {
  commandes
  objets retournés
  }
- Les accolades { et } définissent le début et la fin de la fonction.
- La dernière instruction contient le ou les objets retournés par la fonction.
- Exécuter la fonction : mafonction(...)

Structure générale pour créer des fonctions

Exemple

Ecrire une fonction qui prend en arguments deux variables x et y, et qui renvoie leur somme et leur produit.

```
myfunction = function(x,y)
{
som = x+y
prodt = x*y
return(list("somme"=som,"produit" = prodt))
}
```

Exécuter cette fonction pour $x = 4$ et $y = 5$

```
R > myfunction(4,5)
$ somme
[1] 9
$ produit
[1] 20
```

Sommaire

☞ Il est possible d'obtenir de l'aide pour une commande particulière en tapant la commande :

> ? nom-commande ou ?"nom-commande"

> help(nom-commande) ou help("nom-commande")

☞ De plus, il est possible de réaliser une recherche de fonctions à l'aide de mots-clés grâce à la fonction apropos().

> apropos(mean)

☞ Les démos :

> demo() # pour obtenir la liste des demos

> demo(graphics)

Astuce : un bon moyen pour trouver de l'aide et des exemples sur une fonction consiste simplement à taper le nom de la fonction sous Google.

Sommaire

Sommaire

La fonction read.table()

☞ La fonction read.table() permet de lire des fichiers de données au format texte (ASCII). Cette fonction est destinée à lire les tableaux de données, les données issues d'un tableur qui auraient été préalablement transformées en fichier texte, ...

Exemple :
base =read.table(" /Bureau/IAM/courslogicielR/data/wsd2010data.txt", header=TRUE)

☞ Il existe plusieurs variantes de cette fonction pour lesquelles la valeur par défaut des arguments change : read.csv(), read.csv2(), read.delim() et read.delim2()

Sommaire

Les fonctions write(), write.table()

Les fonctions write() et write.table() permettent d'exporter des objets R en fichiers textes. La fonction write() sert pour les vecteurs et les matrices, la fonction write.table(), elle, exporte les dataframes avec les noms de ligne et de colonne.

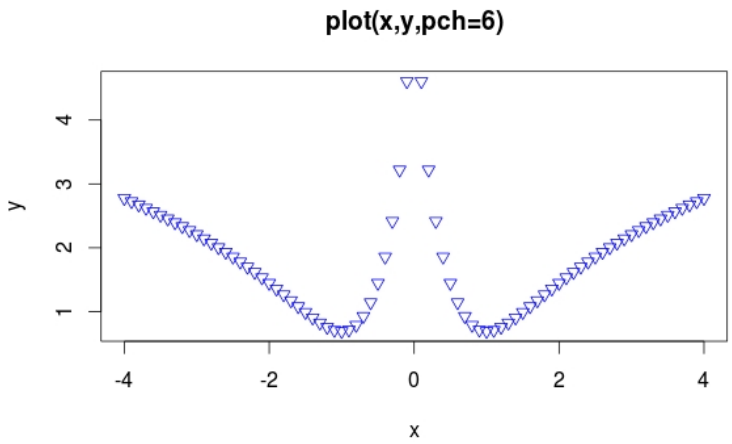
Sommaire

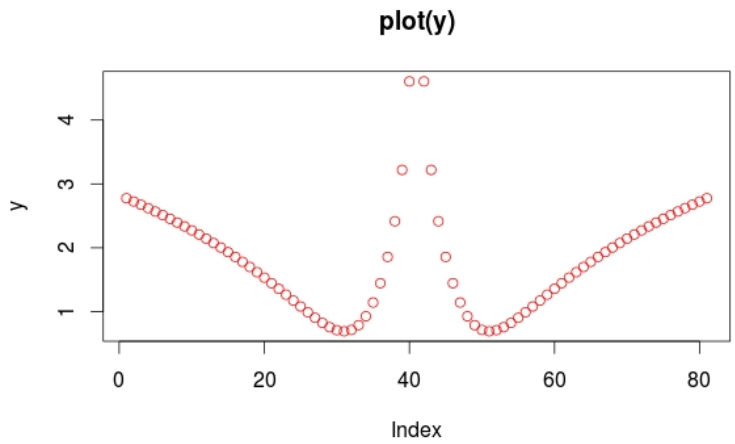
Les fonctions usuelles plot(), lines(), points()

plot() est la fonction centrale. Les fonctions points() ou lines() sont utilisées pour superposer des courbes ou des nuages de points.

> $x = seq(-4, 4, 0.1)$; $y = log(x^2 + 1/x^2)$

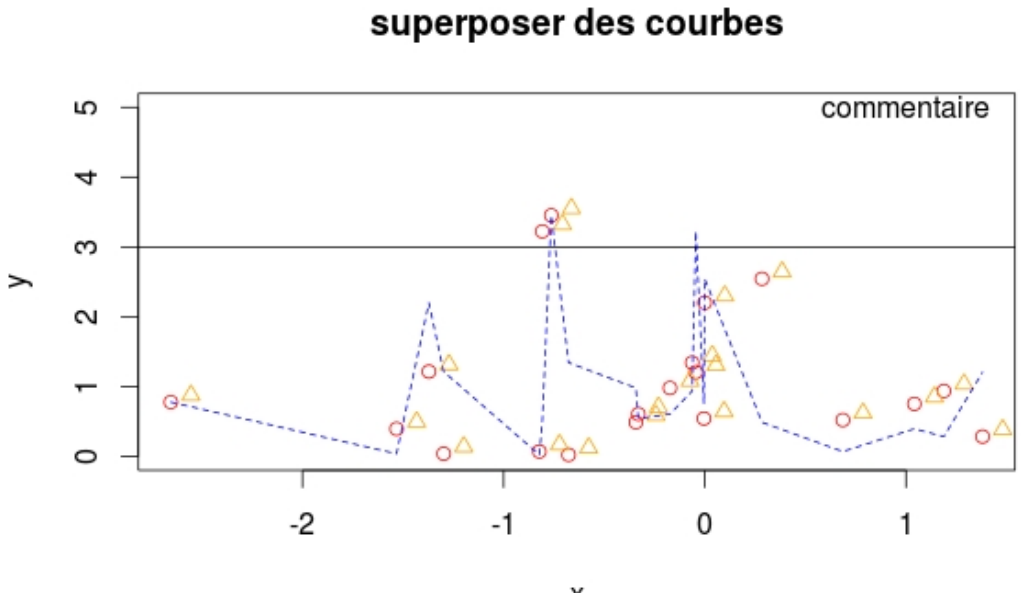
> plot(y) ; plot(x,y)





Superposition de courbes

```
> x=rnorm(20) ; y=rexp(20)
> plot(x,y)          # nuage de points
> points(x+.1,y+.1, pch=2)          # ajouter un nuage de points
> lines(sort(x),y, lty=2)          # ajouter une ligne
> abline(h=3)          # ajouter une ligne horizontale
> text(1,5,"commentaire")          # ajouter un texte
>title("superposer des courbes")          # ajouter un titre
```



Autres fonctions graphiques

- hist(x) : trace l'histogramme de x
- pie(x) : diagramme en secteurs
- barplot(x) : diagramme en barres verticales
- boxplot(x) : diagramme en boîte

Sommaire

Etude de cas

On considère un jeux de données fournissant l'Age, le Genre, la Faculté et la Note de 19 étudiants d'un institut.

☞ Lecture des données :

```
> database =
read.table(" /Bureau/IAM/courslogicielR/data/wsd2010data.txt",
header=TRUE, quote="")

> names(database)
[1] "Etudiant" "Age" "Genre" "Faculte" "Note"

> attach(database) # permet à R de reconnaître les noms des variables
```

Statistique exploratoire unidimensionnelle

Variable qualitative (Genre)

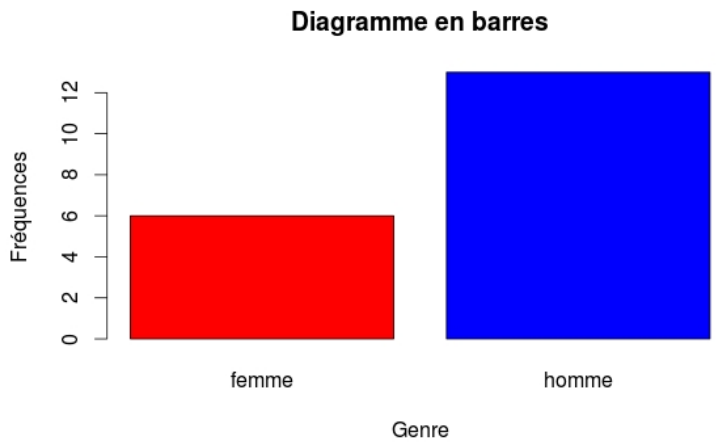
```
> TG = table(Genre)        # tableau de distribution de la variable Genre

> TG                       # affiche le tableau de distribution
```

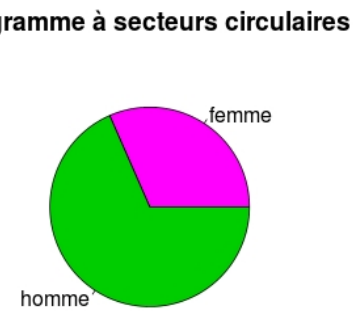
☞ Représentation graphique

```
> barplot(TG, col= c(2,4),main="Diagramme en barres", xlab="Genre",
ylab="Fréquences",axes=TRUE)

> pie(TG, col= c(6,3),main="Diagramme à secteurs circulaires")
```

Statistique exploratoire unidimensionnelle

Variable qualitative (Genre)





Statistique exploratoire unidimensionnelle

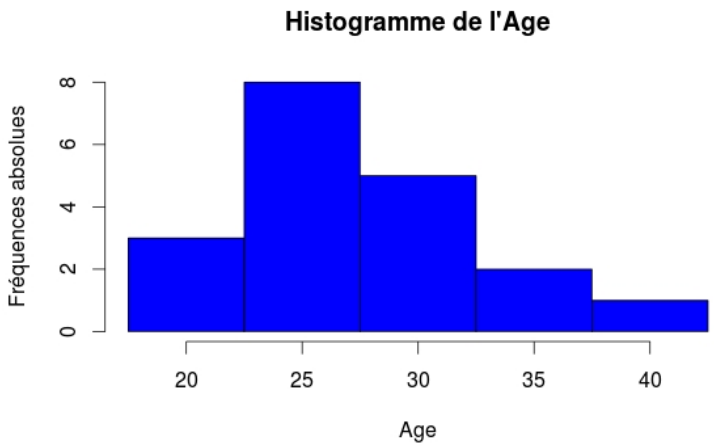
Variable quantitative (Age)

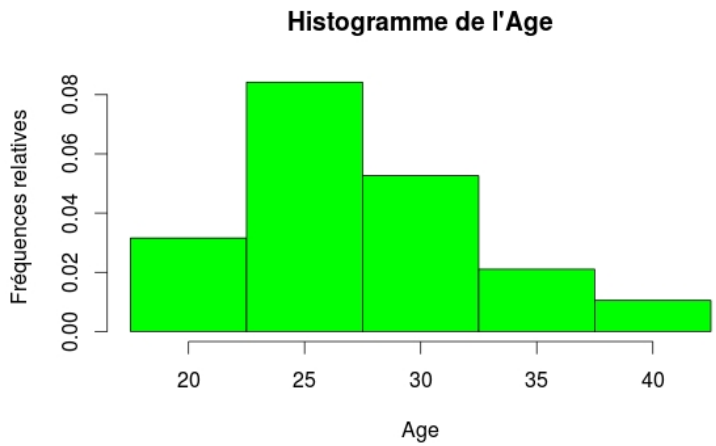
- Histogramme

```
> subdivision= c(17.5, 22.5, 27.5, 32.5, 37.5, 42.5) # classes d'amplitude 5

> hist(Age,subdivision, main="Histogramme de l'Age", xlab="Age", ylab="Effectifs",col="blue")

> hist(Age,subdivision, prob=TRUE, main="Histogramme de l'Age", xlab="Age", ylab="Fréquences relatives",col="green")
```

Statistique exploratoire unidimensionnelle

Variable quantitative (Age)





Statistique exploratoire unidimensionnelle

Moyenne, Ecart-type, coefficient de variation

```
> moy = mean(Age)

> variance = var(Age)

> ecarttype = sqrt(var(Age))

> CV = sqrt(var(Age))/abs(mean(Age))
```

Statistique exploratoire unidimensionnelle

Quantiles

```
> quantile(Age,0.25)          # premier quartile
> quantile(Age,0.50)          # deuxième quartile
> quantile(Age,0.75)          # troisième quartile
> quantile(Age, probs= c(0.25,0.50,0.75))      # donne les 3 quartiles
```

☞ Centile d'ordre t%, t compris entre 1 et 100

```
> quantile(Age,t/100)
```

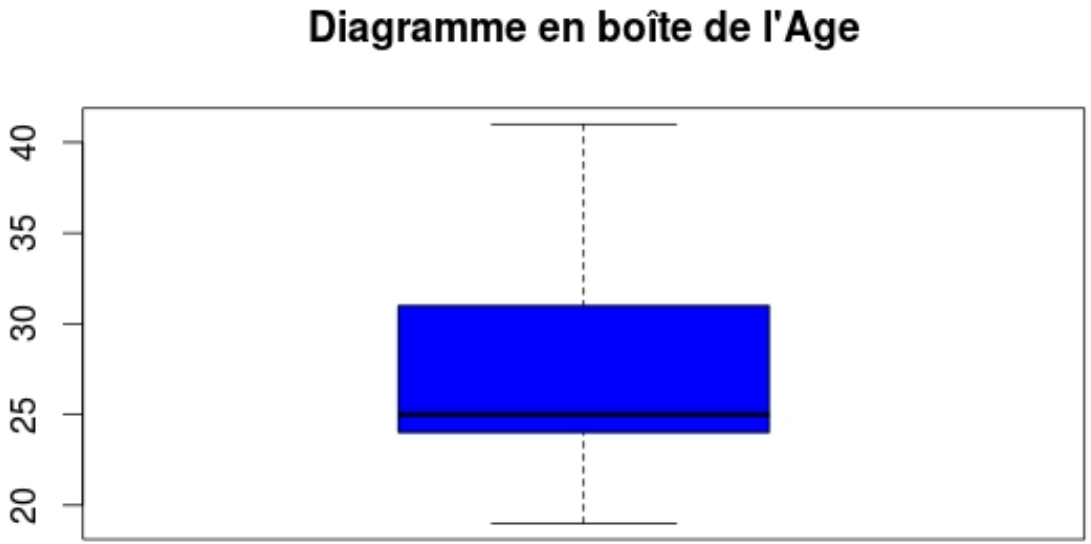
☞ Décile d'ordre t%, t compris entre 1 et 10

```
> quantile(Age,t/10)
```

Statistique exploratoire unidimensionnelle

Boîte à moustache

boxplot(Age, main="Diagramme en boîte de l'Age", col="blue")



Tableaux de Fréquences croisées

Diagrammes en barres multiples

```
> TFG = table(Faculte,Genre)
> barplot(TFG, legend = c("Ingenierie", "Management", "Sciences"),
col= c(2,3,5),main="Diagrammes en barres empilées", xlab="genre",
ylab="Facultés",ylim=c(0,30))
```