Introduction à R

Programmation & Logiciels Statistiques

TD 1

Salim Lardjane - Université de Bretagne-Sud

Commencer

- Lancer R

Aide

- help.start() : Lance l'aide au format html

- help(fonction ou commande) ou ?fonction ou commande : fournit de l'aide sur l'utilisation de la fonction ou de la commande à laquelle on s'intéresse

Aide

- help.search("mot-clé") : fournit une liste des fonctions et commandes pouvant être associées au mot-clé spécifié
- apropos("chaîne de caractères") : fournit une liste des fonctions et commandes dont le nom contient la chaîne de caractères spécifiée

Aide

```
> help.start()
> help(integer)
> help.search("integer")
> apropos("help")
```

- On peut également utiliser le menu *Aide*.

Quitter

```
> q()
```

- On peut également utiliser *Sortir* dans le menu *Fichier*.

Répertoire de travail (répertoire courant de R)

```
> getwd()
> setwd("d:/Work/cours/Tutorial")
> getwd()
```

Historique des commandes

- La liste de commandes R passées peut être obtenue à l'aide de la commande history().

```
> history()
```

Calcul numérique

- En mode interactif, R peut être utilisé comme une calculatrice scientifique. On dispose des opérations arithmétiques et d'un ensemble de fonctions de calcul numérique et de visualisation graphique.

Constantes prédéfinies

- pi : 3.141593
- Inf (Infinite) : nombre infini
- NaN (Not a Number) : n'est pas un nombre, exprime une indétermination
- NA (Not Available) : Non disponible, exprime une valeur manquante.

Constantes prédéfinies

```
> pi
> 1/0
> 0/0
```

- Des commandes peuvent être saisies sur la même ligne à condition d'être séparées par un point-virgule.

```
> pi; 1/0; 0/0
```

Opérations arithmétiques

- +, -, *, /, , %%, %/%
- / : division usuelle
- ^ : élévation à une puissance
- %% : modulo
- %/% : division entière

Opérations arithmétiques

```
> 5+7-3+2*5
> 4/3
> 4 %/% 3
> 3^4
> 5 %% 3
```

Opérations arithmétiques

- Les opérations peuvent être composées en respectant les priorités uselles et en utilisant des parenthèses.

```
> 4*(-5)+12
> 2.3*(4-6)/(3+15)
```

- L'élévation à une puissance est prioritaire par rapport aux autres opérations. Comparer

```
> 9^1/2
```

et

```
> 9^(1/2)
```

Saisie

↑	*passer à l'instruction précédente*
↓	*passer à l'instruction suivante*
←	*aller vers la gauche*
→	*aller vers la droite*
BackSpace	*effacer le caractère à gauche*
Suppr	*effacer le caractère à droite*
Début, Fin	*se positionner au début, à la fin*

Saisie

- Saisir

> 4*(4.5-23+2.5)/(8.5-3.2)

- puis remplacer 23 par 2.3

Vecteurs numériques

- *Saisie d'un vecteur x*

```
> x<-c(6, 4, 7)
> x
```

ou

```
> assign("y",c(6, 4, 7))
> get("y")
> y
```

Saisie

- On peut saisir successivement au clavier les éléments d'un vecteur à l'aide de la commande scan().

```
> x<-scan()
> x
```

Vecteurs

- La longueur d'un vecteur est sa dimension.

```
> length(x)
```

- Lors de la saisie à la console d'un vecteur, on peut spécifier directement le nombre de composantes.

```
> x<-scan(n=3)
> x
```

- R fait la distinction entre majuscules et minuscules.

```
> x
> X
```

Construction d'un vecteur à partir d'un autre

```
> v <- c(x,1,2,8)
> w <- c(1,v)
> w <- c(1,x,1,2,8)
```

Récupération d'une composante d'un vecteur

- Dans R, les indices des coordonnées d'un vecteur commencent à 1.

```
> w
> w[3]
> w[0]
```

Récupération de plusieurs composantes d'un vecteur

```
> w<-c(4,2,5)
> w
> w[c(1,3)]
> w[c(3,1)]
> w[0:2]
> w[c(1,4)]
> w[c(0:2,NA)]
```

Récupération des composantes d'un vecteur sauf certaines

```
> w<-c(4,2,5)
> w
> w[-3]
> w[c(-1,-2)]
```

Ajout d'une composante

```
> w<-c(w,7)
> w
> w<-append(w,7)
> w
> w<-append(w,8,after=2)
> w
```

Remplacement de composantes

```
> w[2]<-3
> w
> w<-replace(w,1,7)
> w
> w<-replace(w,c(2,4),c(8,9))
> w
> w[-c(1,4)]<-0
> w
```

Séquences de valeurs

- Si l'on souhaite définir un vecteur de valeurs comprises entre deux valeurs fixées et espacées d'un pas constant, on peut procéder de l'une des façons suivantes

```
> debut<-0
> fin<-1
> pas<-0.25
> t<-seq(debut,fin,pas)
> t
> t<-seq(-0.5,1,0.25)
> t
```

Séquences de valeurs

```
> t<-seq(from=-0.5,to=1,by=0.25)
> t
```

- Par défaut, le pas est de 1 et la première valeur vaut 1.

```
> seq(10)
> seq(-0.5,10)
```

Séquences de valeurs

- Cas particulier: la commande : permet de générer un vecteur de nombres compris entre la première et la dernière valeur pour un pas de 1.

```
> t<-0:10
> t
> t<-0.5:10.8
> t
```

- La longueur du vecteur obtenu peut être obtenue à l'aide de la fonction length.

```
> length(seq(0,1,.1))
```

Séquences de valeurs

- On peut également spécifier directement le nombre d'éléments à obtenir au lieu du pas.

```
> t<-seq(from=-0.5,to=1,length=9)
> t
```

Répétition et ordre

- Afin de répéter une même valeur ou un même vecteur un nombre de fois, on peut utiliser la fonction rep.

```
> rep(2,10)
> rep(c(4,2),3)
```

- Afin d'inverser l'ordre d'un vecteur, on peut utiliser la fonction rev.

```
> rev(1:5)
```

Unique

- Afin d'extraire des composantes sans répétition :

```
> x <-c(rep(1,3),seq(1,5,by=2),rev(seq(1,5,length
                =3)),rep(2,3))
> x
> unique(x)
```

Tri et rangs

- Afin d'ordonner les composantes par ordre croissant ou décroissant :

```
> sort(x)
> rev(sort(x))
```

- Afin d'obtenir les rangs des composantes (en cas d'ex-aequos, le rang moyen est utilisé) :

```
> rank(x)
```

Tri et rangs

- Afin d'obtenir les indices des composantes ordonnées par ordre croissant :

```
> order(x)
```

- sort(x) est équivalent à x[order(x)].

Opérations arithmétiques sur les vecteurs

- Les opérateurs +,-,*, /, , %%, %/% permettent de réaliser des opérations sur les vecteurs élément par élément. On dit que R est *vectorisé*.

```
> x<-c(0,6,3)
> y<-c(2,5,7)
> x+y
> x-y
> 3+x
> x-2
> 2*x
```

Opérations arithmétiques sur les vecteurs

```
> x/4
> x*y
> x^2
> x/y
> x%%y
> x%/%y
```

Fonctions

- *Exemple explicite de fonction*

```
> f<-function(x,y)
{
return(x+y-2)
}

> x
> y
> z<-f(x,y)
> z
```

Fonctions

- La définition d'une fonction peut faire intervenir des variables locales.

```
 > f<-function(x,y)
{
ans<-x+y-2
return(ans)
}
 > f(x,y)
```

Fonctions mathématiques

- Les fonctions mathématiques usuelles sont prédéfinies dans R : *abs, sqrt, sin, cos, exp, tan, asin, acos, atan, log, log10, log2, sinh, cosh, tanh, asinh, acosh, atanh...* ainsi que les fonctions spéciales *gamma, digamma, beta, bessel,...*etc.
- R étant *vectorisé*, toutes ces fonctions peuvent prendre un vecteur comme argument. Les fonctions sont alors appliquées élément par élément.

Fonctions mathématiques

```
> x
> y
> z<- -y
> sqrt(x)
> abs(z)
```

Fonctions numériques usuelles

ceiling(x)	plus petit entier supérieur à x (composante par composante)
floor(x)	plus grand entier inférieur - - -
trunc(x)	composante entière de x - - -
round(x)	arrondi à la plus proche valeur entière (valeurs .5 arrondies au plus proche entier pair) - - -
round(x,n)	arrondi à n chiffres après la virgule - - -

Fonctions numériques usuelles

```
> x<- c(-1.9069,0.76018,-0.26556,-1.89828,0.08571,NA)
> ceiling(x)
> floor(x)
> trunc(x)
> round(x)
> round(x,2)
```

Fonctions statistiques usuelles

sum(x)	somme des composantes d'un vecteur x
prod(x)	produit des - - -
mean(x)	moyenne des - - -
mean(x,trim=p)	moyenne trimmée - - - (p est compris entre 0 et 0.5)
median(x)	médiane des - - -
quantile(x,probs=p)	quantiles des - - - de niveaux spécifiés dans le vecteur p
var(x)	variance des - - -
sd(x)	écart-type des - - -
min(x)	minimum des - - -
max(x)	maximum des - - -
cumsum(x)	sommes cumulées des - - -
cumprod(x)	produits cumulés des - - -
rank(x)	rangs des - - -

Fonctions statistiques usuelles

```
> x<-scan()
> sum(x)
> prod(x)
> mean(x)
> mean(x,trim=0.2)
> median(x)
> quantile(x,probs=c(0,0.1,0.9)))
> var(x)
> sd(x)
> min(x)
> max(x)
> cumsum(x)
> cumprod(x)
> rank(x)
```

Nouvelle fonction

- *Définir une fonction permettant de centrer et réduire un vecteur donné*.

Nouvelle fonction

```
 > cr<-function(x)
{
ans=(x-mean(x))/sqrt(var(x))
return(ans)
}
 > v<-1:10
 > cr(v)
```

Autres fonctions statistiques

which.max(x)	indice du maximum des composantes de x
which.min(x)	indice du minimum des - - -
range(x)	identique à c(min(x),max(x))
var(x,y) ou cov(x,y)	covariance entre x et y
cor(x,y)	corrélation linéaire entre - - -
cummin(x)	minimums cumulés des composantes de x
cummax(x)	maximums cumulés des - - -
pmin(x,y,...)	vecteur dont la i-ème composante est min(x[i],y[i],...)
pmax(x,y,...)	vecteur dont la i-ème composante est max(x[i],y[i],...)
diff(x)	vecteur dont la i-ème composante est x[i+1]-x[i]
diff(x,lag=n)	vecteur dont la i-ème composante est x[i+n]-x[i]

Arguments par défaut

- Il est souvent souhaitable de spécifier des valeurs par défaut pour les arguments d'une fonction. Cela peut être fait lors de la déclaration de la fonction. Par exemple, la première ligne de la fonction mean est

```
mean <- function(x,trim=0,na.rm=FALSE)
```

Arguments par défaut

- Les arguments pour lesquels une valeur par défaut est spécifiée peuvent être omis lors de l'appel de la fonction.
- Par exemple :

```
> x<-1:10
> mean(x)
```

permet d'obtenir la moyenne standard de x.

Arguments par défaut

- Si l'on souhaite effectuer un trimming de 20%, il suffit de spécifier la valeur correspondante pour l'argument trim :

```
> mean(x,trim=0.2)
```

- Il en va de même pour l'argument na.rm, que nous discuterons plus loin.

Arguments par défaut

- *Exemple détaillé*

```
> fk<-function(x,y,k=2)
{
return(x+y-k)
}
 > x<-1:3
 > y<-5:7
 > fk(x,y)
 > fk(x,y,k=3)
```

Matrices numériques

- *Matrice nulle*

```
> matrix(0,nrow=2,ncol=3)
```

ou

```
> mat.or.vec(2,3)
```

- *Matrice unité*

```
> matrix(1,nrow=3,ncol=2)
```

Matrice identité

```
> x<-matrix(0,nrow=2,ncol=2)
> diag(x)<-1
> x
```

ou

```
> diag(1,nrow=2)
```

ou encore

```
> diag(rep(1,2))
```

Saisie d'une matrice quelconque

- *Eléments lus colonne par colonne*

```
> x<-matrix(c(0,1,2,3,4,5),nrow=2,ncol=3)
> x
```

Saisie d'une matrice quelconque

- *Eléments lus ligne par ligne*

```
> x<-matrix(c(0,1,2,3,4,5),nrow=2,ncol=3,
      byrow=TRUE)
> x
```

Saisie d'une matrice

- *Par redimensionnement ou répétition ("recyclage") des éléments d'un vecteur ou d'un scalaire*

```
> y<-matrix(c(0,5),nrow=2,ncol=3)
> y
> z<-matrix(5,nrow=2,ncol=3)
> z
```

Saisie d'une matrice

- *Par concaténation de lignes*

```
> rbind(c(0,2,4),c(1,3,5))
```

- *Par concaténation de colonnes*

```
> cbind(c(0,1),c(2,3),c(4,5))
```

Saisie d’une matrice

- *Saisie directe au clavier*

```
> x <-matrix(scan(n=2*3),nrow=2,ncol=3,byrow=TRUE)
> x
```

- *Saisie d'une matrice diagonale*

```
> u <- c(10,20)
> v <- diag(u)
> v
```

Saisie d’une matrice

- *Saisie d'un vecteur colonne*

```
> y<-matrix(c(6,4,7),nrow=3,ncol=1)
```

ou

```
> as.matrix(c(6,4,7))
```

Dimensions d'une matrice

```
> d<-dim(y)
> d
> d[1]
> d[2]
```

Transposée d'une matrice

```
> x
> tx<-t(x)
> tx
```

Eléments d'une matrice

```
> x[2,1]
```