Biologie et Modélisation

Introduction et éléments de base du langage 

M. Bailly-Bechet

Université Claude Bernard Lyon I – France

Document disponible à :
`http://pbil.univ-lyon1.fr/members/mbailly`

Table des matières

Table des matières

R en trois lignes

- R est un environnement d'analyse statistique basé sur le langage de programmation S.
- R est un logiciel libre.
- R est un logiciel professionnel.

S est interactif

S incite fortement l'utilisateur à examiner et analyser ses données de manière interactive, au contraire des logiciels classiques, tels que SAS, qui implémentent un modèle d'analyse en *différé* :

- L'utilisateur soumet une tâche en fournissant les données et les instructions correspondant à l'analyse à effectuer
- Le logiciel effectue l'analyse et imprime toutes les informations susceptibles d'intéresser l'utilisateur (et elles peuvent être très nombreuses)
- L'utilisateur scrute ensuite les résultats pour extraire l'information qui l'intéresse.

L'approche de S est devenue très populaire dans les milieux académiques.

S est prestigieux

En 1998, l'Association for Computing Machinery (ACM) récompense John Chambers de son prix prestigieux pour les logiciels pour :

- *the S system, which has forever altered the way people analyze, visualize, and manipulate data . . .*
- *le système S, qui a révolutionné la manière dont on analyse, visualise et manipule les données . . .*

C'est le seul logiciel de statistique à avoir jamais eu ce prix. Les autres lauréats sont, en autres, les créateurs d'UNIX, du WWW et du langage de programmation Java. En 2014 ? Les concepteurs de Mach.

R est le descendant de S

- Encouragés par Martin Mächler, Ross et Robert décidèrent de diffuser R en tant que *logiciel libre* en juin 1995.
- La disponibilité de R en tant que logiciel libre permet aux utilisateurs d'examiner, de modifier et d'améliorer le code source de R, puis de partager ces changements avec les autres.
- La seule restriction est que ces modifications doivent rester dans le domaine du libre. C'est ce qui assure, de manière pérenne, que tous les travaux futurs basés sur R resteront libres.

Avantages du modèle de développement de R

- Avec une communauté active d'utilisateurs et de développeurs les bugs sont identifiés rapidement.
- Comme le code source est disponible, c'est souvent les utilisateurs eux-même qui localisent les bugs dans le code source et proposent des solutions.
- Ces points, ainsi que d'autre problèmes de développement, sont discutés dans un forum publique de sorte que tous les utilisateurs ont la possibilité de contribuer.
- Une forme d'aide pour R est fournie sous la forme d'une liste de diffusion, où les utilisateurs peuvent poser des questions et d'autres y répondre. Bien que ce soit un forum informel, il y a tant d'abonnés qu'il n'est pas rare d'avoir une réponse dans la minute qui suit.

Table des matières

Qu'allons nous faire avec R ?

- R est un environnement permettant de faire des analyses statistiques et de produire des graphiques. C'est également un langage de programmation complet.
- Nous allons utiliser ici R comme une boîte à outils pour faire des analyses statistiques standard. L'aspect de programmation ne sera pas abordé, même s'il est sous-jacent.

Les informations sur R sont disponibles sur la homepage du projet R : `http://www.r-project.org`, c'est le premier résultat pour la recherche de la lettre "R" avec le moteur de recherche google.

Objectif de ce premier cours ...

- Apprendre les bases du langage
- Apprendre à manipuler des données
- Apprendre à faire un graphique
- Apprendre à utiliser la documentation et le système d'aide

Lancer et quitter

- *Unix/Linux* : entrer R dans un terminal
- *Mac OS X* : double-click sur R
- *Windows* : double-click sur R

Pour quitter , entrer `q()` sur la ligne de commande.

Premier pas : Interaction avec R

On utilise généralement R interactivement, selon un cycle question-et-réponse :

- Vous entrez une commande et tapez la touche "Retour à la ligne".
- R exécute cette commande (avec affichage d'un résultat si besoin est)
- R attend une autre commande

Quelques exemples simples

Dans les exemples suivants, ce qui est entré par l'utilisateur figure en rouge, et la réponse de R est en bleu. Par exemple :

```
2 + 2
```

```
[1] 4
```

Quelques exemples simples

```
exp(-2) ##  la fonction exponentielle
```

```
[1] 0.1353353
```

```
log(100, base = 10)
```

```
[1] 2
```

```
runif(10)
```

```
 [1] 0.45484783 0.15774286 0.05865066 0.55151885 0.28279239 0.03008109
 [7] 0.74912356 0.94879005 0.35267319 0.15306775
```

La dernière commande produit 10 nombres pseudo-aléatoires compris entre 0 et 1. Le résultat affiché est un vecteur de 10 nombres. Les nombres entre crochets au debut de chaque ligne donnent l'indice du premier nombre de la ligne.

Les fonctions

- `exp()`, `log()` et `runif()` sont des fonctions.
- Les appels aux fonctions sont indiqués par la présence de parenthèses.
- La plupart des choses utiles sous R sont faites par des fonctions.

Table des matières

Variables et Affectations

Comme la plupart des langages de programmation, R a des variables auxquelles on peut affecter une valeur. Pour cela on utilise l'opérateur '<-' ou '->'. L'opérateur classique '=' marche aussi.

```
x <- 2
y <- x + 3
"ceci est une chaine de caracteres" -> s
x
```

```
[1] 2
```

```
y
```

```
[1] 5
```

```
s
```

```
[1] "ceci est une chaine de caracteres"
```

```
x + x
```

```
[1] 4
```

```
x ^ y
```

```
[1] 32
```

Noms des variables

Les noms de variables sont très flexibles. N'importe quelle variable peut stocker n'importe quelle valeur (il n'y a pas besoin de déclarer les variables). Cependant, il faut savoir que :

- Les noms de variables ne peuvent pas commencer par un chiffre ou un caractère spécial
- Les noms sont sensibles à la casse des caractères (un caractère minuscule comme `x` est différent d'un caractère majuscule comme `X`)
- Il est très fortement recommandé d'utiliser des noms explicites pour vos variables.

Vecteurs

- Les types élémentaires dans R sont tous des vecteurs
- Même un simple nombre est un vecteur de longueur 1

La construction `c(...)` peut être utilisée pour générer un nouveau vecteur :

```
poids <- c(60, 72, 57, 90, 95, 72)
poids
```

```
[1] 60 72 57 90 95 72
```

"poids" est donc un vecteur de 6 éléments.
Pour accéder à un élément particulier d'un vecteur, il vous suffit d'utiliser l'opérateur d'indexation `[]` :

```
poids[3]
```

```
[1] 57
```

Arithmétique vectorielle

Les opérations arithmétiques usuelles

- + pour faire des additions
- - pour faire des soustractions
- * pour faire des multiplications
- / pour faire des divisions
- ^ pour élever à la puissance

et les fonctions mathématiques travaillent élément par élément sur les vecteurs et produisent un autre vecteur :

```
taille <- c(1.75, 1.80, 1.65, 1.90, 1.74, 1.91)
taille^2
```

```
[1] 3.0625 3.2400 2.7225 3.6100 3.0276 3.6481
```

```
imc <- poids/taille^2
imc
```

```
[1] 19.59184 22.22222 20.93664 24.93075 31.37799 19.73630
```

```
log(imc)
```

```
[1] 2.975113 3.101093 3.041501 3.216102 3.446107 2.982460
```

Arithmétique vectorielle : recyclage

Quand deux vecteurs ne sont pas de même longueur, le plus court est recyclé. La commande suivante ajoute 0 à tous les éléments impairs et 2 à tous les éléments pairs de la variable `imc` :

```
 imc
[1] 19.59184 22.22222 20.93664 24.93075 31.37799 19.73630
 c(0,2)
[1] 0 2
 imc + c(0, 2)
[1] 19.59184 24.22222 20.93664 26.93075 31.37799 21.73630
```

Fonctions vectorisées

Beaucoup de fonctions résument un vecteur de données en produisant un nombre à partir d'un vecteur. Par exemple :

```
sum(poids)
```

```
[1] 446
```

```
length(poids)
```

```
[1] 6
```

```
poids.moy <- sum(poids) / length(poids)
poids.moy
```

```
[1] 74.33333
```

La dernière commande calcule la moyenne de `poids` qui vaut donc ici 74.3.

Fonctions vectorisées

Beaucoup de fonctions résument un vecteur de données en produisant un nombre à partir d'un vecteur. Par exemple :

```
sum(poids)
```

```
[1] 446
```

```
length(poids)
```

```
[1] 6
```

```
poids.moy <- sum(poids) / length(poids)
poids.moy
```

```
[1] 74.33333
```

La dernière commande calcule la moyenne de `poids` qui vaut donc ici 74.3.

```
mean(poids)
```

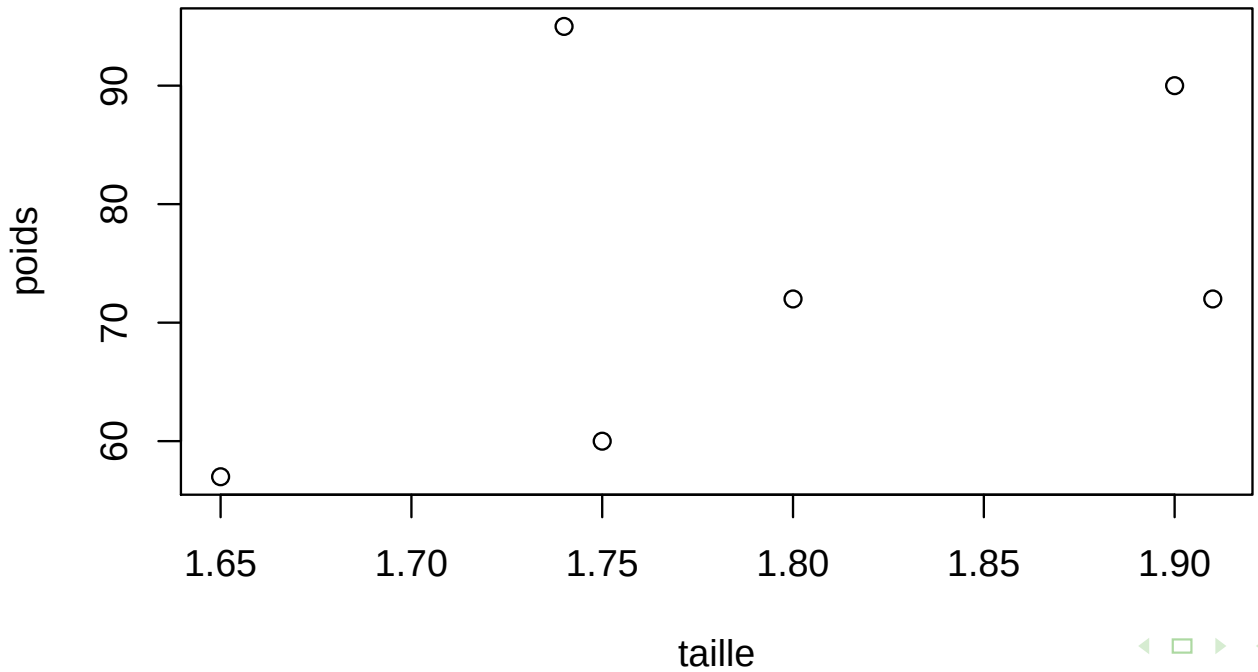
```
[1] 74.33333
```

Table des matières

plot()

La manière la plus simple de produire des graphiques sous R est d'utiliser la fonction `plot()` :

```
plot(x = taille, y = poids)
```



Options et retouche

Les fonctions graphiques de R comportent de nombreuses options qui permettent de contrôler de façon très fine les graphiques. Par exemple, les paramètres de la fonction `plot` utilisée par défaut sont :

```
args(plot.default)
```

```
function (x, y = NULL, type = "p", xlim = NULL, ylim = NULL,
    log = "", main = NULL, sub = NULL, xlab = NULL, ylab = NULL,
    ann = par("ann"), axes = TRUE, frame.plot = axes, panel.first = NULL,
    panel.last = NULL, asp = NA, ...)
NULL
```

L'argument `...` signifie qu'il y a encore d'autre paramètres graphiques possibles. Ils sont contrôlés par la fonction `par()`.

Options et retouche

```
names(par())
```

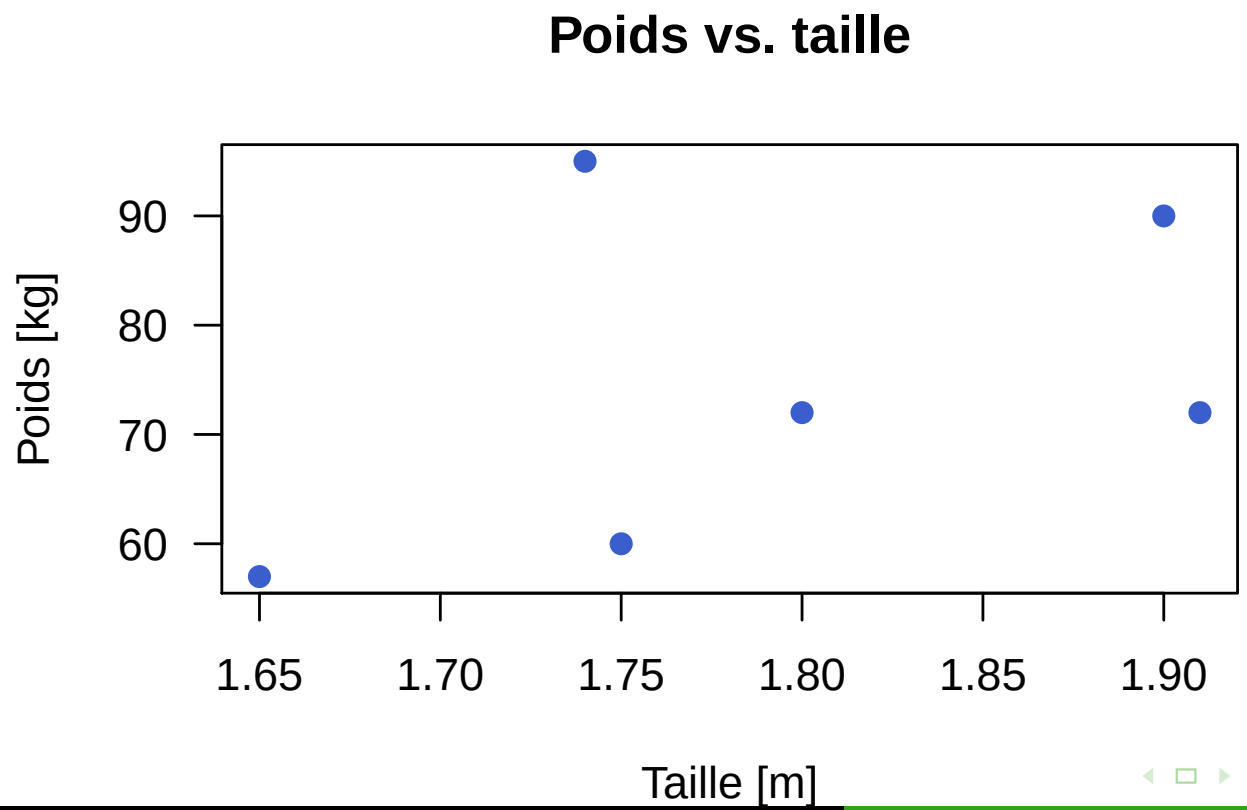
```
 [1] "xlog"      "ylog"      "adj"       "ann"       "ask"       "bg"
 [7] "bty"       "cex"       "cex.axis"  "cex.lab"   "cex.main"  "cex.sub"
[13] "cin"       "col"       "col.axis"  "col.lab"   "col.main"  "col.sub"
[19] "cra"       "crt"       "csi"       "cxy"       "din"       "err"
[25] "family"    "fg"        "fig"       "fin"       "font"      "font.axis"
[31] "font.lab"  "font.main"  "font.sub"  "lab"       "las"       "lend"
[37] "lheight"   "ljoin"     "lmitre"    "lty"       "lwd"       "mai"
[43] "mar"       "mex"       "mfcol"     "mfg"       "mfrow"     "mgp"
[49] "mkh"       "new"       "oma"       "omd"       "omi"       "page"
[55] "pch"       "pin"       "plt"       "ps"        "pty"       "smo"
[61] "srt"       "tck"       "tcl"       "usr"       "xaxp"      "xaxs"
[67] "xaxt"      "xpd"       "yaxp"      "yaxs"      "yaxt"      "ylbias"
```

Pour une exploration systématique des paramètres graphiques, voir la fiche `http://pbil.univ-lyon1.fr/R/fichestd/tdr75.pdf`.

Options et retouche

Un exemple de graphique utilisant quelques options :

```
plot(x = taille, y = poids, pch = 19, col = "royalblue3", las = 1,
main = "Poids vs. taille", xlab = "Taille [m]", ylab = "Poids [kg]")
```



Options et retouche

Dans le graphe précédent :

- `pch` permet de choisir le type de point avec lequel on fait le graphe (rond, carré, triangle...)
- `col` permet de choisir la couleur des points
- `las=1` permet d'écrire les poids et les tailles sur les axes à l'horizontale
- `main` définit le titre du graphique
- `xlab` définit le nom de l'axe des abcisses
- `ylab` définit le nom de l'axe des ordonnées

Options et retouche

Il existe de nombreuses fonctions permettant de retoucher un graphique, par exemple :

```
plot(x = taille, y = poids, pch = 19, col = "royalblue3", las = 1,
main = "Poids vs. taille", xlab = "Taille [m]", ylab = "Poids [kg]")
range_val <- seq(from = min(taille), to = max(taille), length = 100)
lines(x = range_val, y = 22.5 * range_val^2, col = "red")
legend("bottomright", inset = 0.01, legend = expression(y == 22.5*x^2),
lty = 1, col = "red")
```

La variable `x`, générée par la commande `seq`, est la suite arithmétique de 100 nombres allant de la plus petite à la plus grande des valeurs du vecteur `taille`, et la commande `lines` permet de tracer une ligne suivant une équation `y=f(x)`. `legend` insère une légende dans la figure principale.

Options et retouche

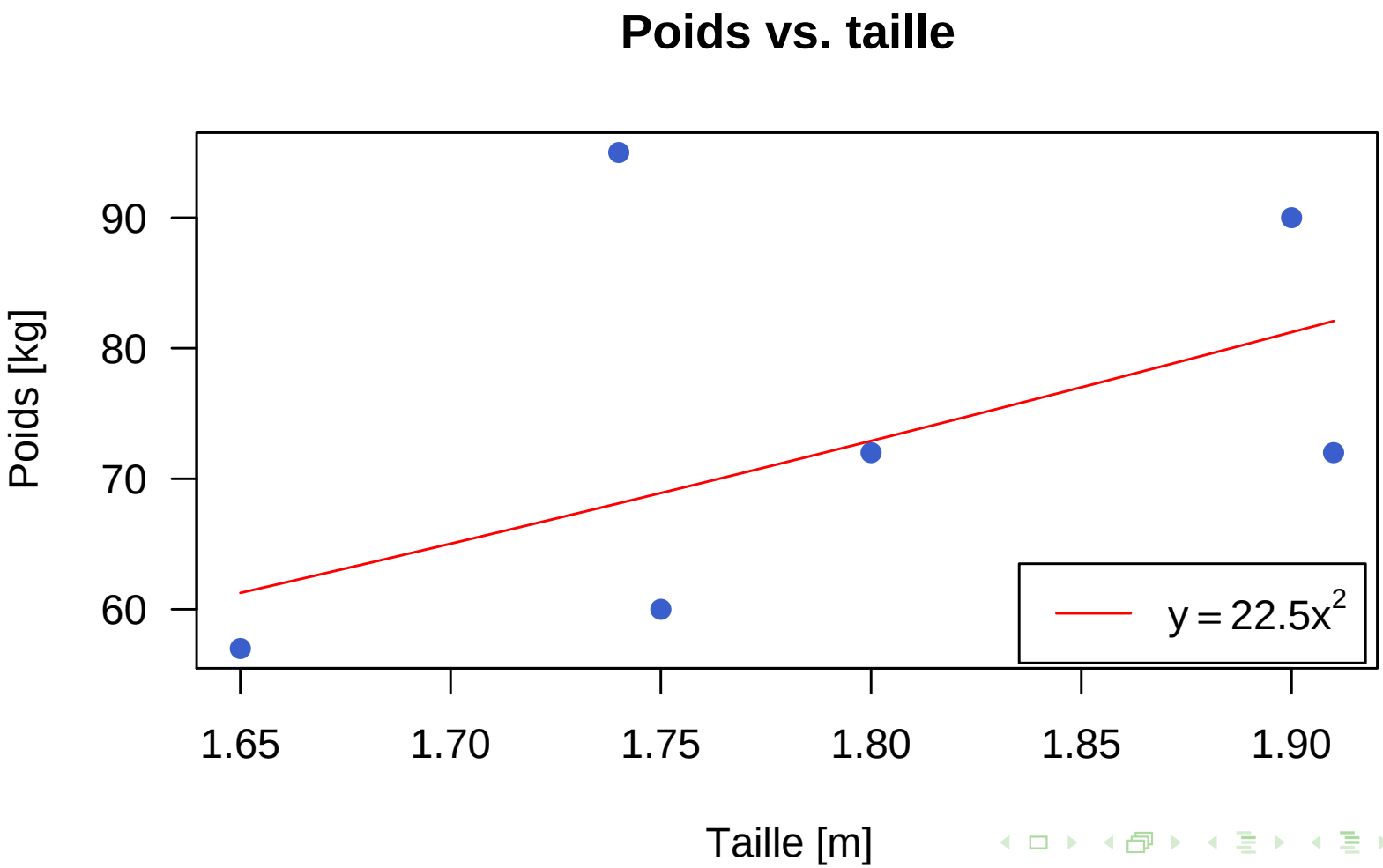


Table des matières

help.start()

R a beaucoup trop d'outils pour que quiconque puisse les retenir tous, il est donc très important de savoir comment retrouver les informations pertinentes en utilisant le système d'aide.
`help.start()` ouvre une fenêtre avec une interface pour l'aide de type HTML. Il y a un lien vers un manuel très détaillé pour les débutants appelé "An Introduction to R", ainsi que des listes par sujets.

RSiteSearch()

La fonction `RSiteSearch()` fait une recherche dans l'ensemble des documents (manuels, documentation, archives des listes de diffusion) du site de R.

help.search()

Quand vous voulez obtenir de l'aide sur un sujet donné, mais que vous ne savez pas quelle est la bonne page d'aide, la fonction `help.search()` est très utile. Elle vous renvoie toutes les fonctions dont la description contient le mot que vous cherchez :

```
help.search("logarithm")
```

```
Help files with alias or concept or title matching fuzzy matching:


VGAM::Log                  Logarithmic Distribution
VGAM::logff                Logarithmic Distribution
base::log                  Logarithms and Exponentials
nlme::logDet               Extract the Logarithm of the Determinant
```

apropos()

Un autre outil utile est la fonction `apropos()` qui donne une liste de tous les sujets contenant (exactement) l'argument :

```
apropos("plot")[1:10]
```

```
 [1] "assocplot"       "barplot"         "barplot.default" "biplot"
 [5] "boxplot"         "boxplot.default" "boxplot.matrix"  "boxplot.stats"
 [9] "cdplot"          "coplot"
```

Nous n'avons donné que les 10 premiers éléments, la liste complète est trop longue.

help(sujet) ou ?sujet

`help(sujet)` que l'on peut aussi écrire `?sujet` affiche la page d'aide pour le sujet ou la fonction `sujet`. Toutes les fonctions de R ont une page d'aide. Quand on connaît le nom de la fonction ou du sujet qui nous intéresse, c'est en général le meilleur moyen d'apprendre à l'utiliser.

```
help(log)
```

```
Description
log computes logarithms, by default natural logarithms. The general form
log(x, base) computes logarithms with base base.

Usage
log(x, base = exp(1))

Arguments
x       a numeric or complex vector.
base    a positive or complex number: the base with respect to which
        logarithms are computed. Defaults to e=exp(1)
```

example()

Les pages d'aide sont généralement très détaillées. Elles contiennent souvent, entre autres :

- Une section "See Also" qui donne les pages d'aide sur des sujets apparentés
- Une section "Description" de ce que fait la fonction
- Une section "Examples" avec du code illustrant ce que fait la fonction documentée. Ces exemples peuvent être exécutés directement en utilisant la fonction `example()`, essayez par exemple :

```
example(plot)
```

args()

La fonction `args()` donne la liste des arguments d'une fonction :

```
args(plot.default)
```

```
function (x, y = NULL, type = "p", xlim = NULL, ylim = NULL,
    log = "", main = NULL, sub = NULL, xlab = NULL, ylab = NULL,
    ann = par("ann"), axes = TRUE, frame.plot = axes, panel.first = NULL,
    panel.last = NULL, asp = NA, ...)
NULL
```

Pour en savoir plus

- Pour un public francophone, un très bon point de départ est le manuel d'Emmanuel Paradis, R pour les débutants. Vous pouvez le récupérer, avec d'autres manuels, à cette adresse : `http://pbil.univ-lyon1.fr/members/mbailly/Biologie_Modelisation/docs`
- Plusieurs milliers de pages d'enseignement en français de statistiques sous  sont disponibles ici : `http://pbil.univ-lyon1.fr/R/`. Les niveaux vont de l'initiation au niveau post-doctoral, à vous d'explorer.

Table des matières

library()

R utilise un système de bibliothèques, les *packages*.

- Chaque bibliothèque est une collection regroupant des outils d'une même thématique.
- R est lui même une bibliothèque appelée `base`
- Certaines bibliothèques sont automatiquement disponibles lorsque R est lancé, d'autres doivent être chargées avec la fonction `library()`

installed.packages()

Certaines bibliothèques sont pré-installées avec R. La liste des bibliothèques installées est donnée par la fonction :

```
installed.packages()
```

Il y a beaucoup d'autre bibliothèques développées par des utilisateurs de R disponibles sur le site du CRAN (Comprehensive R Archive Network).

search()

Certaines bibliothèques sont automatiquement disponibles lorsque R est lancé. À n'importe quel moment, la liste des bibliothèques chargées est donnée par la fonction `search()` :

```
search()
```

```
[1] ".GlobalEnv"        "package:stats"     "package:graphics"
[4] "package:grDevices" "package:utils"     "package:datasets"
[7] "Autoloads"         "package:base"
```

install.packages()

D'autres bibliothèques peuvent être chargées par l'utilisateur. Nous allons charger la bibliothèque `stats4` qui contient des fonctions statistiques avancées. Ceci peut être fait avec :

```
library(stats4)
```

De nouvelles bibliothèques peuvent être téléchargées et installées avec la fonction `install.packages()`. Par exemple, pour installer la bibliothèque `stats4` (si elle n'est pas déjà installée), on peut utiliser :

```
install.packages("stats4", lib = getwd())
library(help = stats4, lib = getwd()))
```

La dernière commande donne la liste de toutes les pages d'aide de la bibliothèque.