

Introduction au logiciel R

Laurence Viry

MaiMoSiNE - Collège des écoles doctorales Grenoble

7-16 Février 2017

Plan du cours

- 1 Introduction
- 2 Installer R
- 3 Utiliser R
- 4 RStudio
- 5 Objets de R

Introduction au logiciel R

Laurence Viry

MaiMoSiNE - Collège des écoles doctorales Grenoble

7-16 Février 2017

Qu'est ce que R?

- R est un système d'**analyse statistique**, de **gestion de données** et de **visualisation** (R. Ihaka, R. Gentleman).
- R est à la fois un **logiciel** et un **langage interprété** dont la conception dérive du **langage S** développé au sein des laboratoires AT&T Bell dans les années 80.
- Le langage R possède aujourd'hui une **communauté mature d'utilisateurs** et de développeurs qui **partagent des milliers de package** via le Comprehensive R Archive Network (**CRAN**).
- R est un **langage de programmation simple** et efficace qui permet à l'utilisateur d'écrire ses propres algorithmes.
- R offre les possibilités d'**améliorer ses performances** en convertissant le haut niveau d'interprétation du code R en un langage compilé écrit en C, C++ ou Fortran (Rcpp,...).
- R permet de **s'adapter à l'architecture des processeurs multi-coeurs et des clusters distribués** en fournissant des outils adaptés aux techniques de programmation parallèle (parallel, multicore, snow,...)
- ...

R a des avantages majeurs

- Il est **gratuit** et le restera puisqu'il est distribué librement sous les termes de la **GNU General Public Licence**.
- Il est **disponible sur Internet**, via un grand réseau de miroirs et facile à installer sur toutes les plateformes.
- Il est le produit d'une **collaboration internationale entre statisticiens et informaticiens R development Core Team**.
- Il ne limite pas l'utilisateur à un ensemble de procédures ou d'options, il permet le développement d'algorithmes grâce à son **langage de programmation simple**.
- Les **packages développés par la communauté** sont accessibles via le **CRAN**.
- Les codes sources sont publiés, les **algorithmes utilisés peuvent être consultés**.
- L'**importation** et l'**exportation** de données et de résultats vers d'**autres progiciels** est possible (MS-Excel, SAS, SPSS, Matlab,...).
- Il existe des interfaces de développement (IDE) R efficaces et faciles à utiliser.
- ...

Inconvénients et alternatives

➤ Inconvénients

- L'utilisateur doit **définir lui-même la séquence des analyses** et les exécuter pas à pas. Il pourra aussi écrire des scripts dans un éditeur souvent fourni par les interfaces de développement R utilisées (dans ce cours RStudio).
- L'utilisateur doit **apprendre un langage**.
- Il devra **penser autrement la gestion de ses données**, des méthodes appliquées à ces données pour bénéficier des possibilités de programmation objets du langage R.

➤ Alternatives: **autres progiciels** permettent le traitement statistique, **pas open-source**, il faudra faire confiance aux développeurs.

- **S-PLUS**: implémentation commerciale du langage S diffusée par TIBCO.
- Progiciels de statistiques (**MINITAB**, **SPSS**, **Statistica**, **Systat**, **GenStat**, and **BMDP**,...).
- **SAS**: un concurrent de S-PLUS, très utilisé dans l'industrie, programmable .
- Des **programmes statistiques spécifiques**:
<http://www.stata.com/links/statistical-software-providers/>.
- **Tableurs** : Microsoft Excel (Data Analysis. . .),...

Installer R

R est **distribué librement** sous les termes de la **GNU General Public Licence**, son développement et sa distribution sont assurés par plusieurs statisticiens rassemblés dans le **R Development Core Team**.

R est distribué sous plusieurs formes à partir de sites internet du **CRAN**:

- **d'exécutables précompilés** pour Windows, Linux et OSX (Macintosh).
- **de sources**, écrit principalement **en C** et certaines routines en Fortran) qu'il faudra compiler avant de les utiliser.
- Les **instructions à suivre pour l'installation** sur chaque système sont fournis avec la distribution.
- Plusieurs **packages sont fournis avec l'installation de base**.
- De **nouveaux packages** peuvent être facilement **ajoutés et créés** en cours de développement.
- L'installation de **nouveaux packages** peut se faire **en local** ou pour **l'ensemble des utilisateurs**.

- En **mode console** (avec une interface GUI ou pas).
- Avec l'éditeur **Tinn-R** et la **console R**.
- A partir du module **ESS** de l'éditeur **Emacs**
- Avec un **environnement de développement intégré (IDE)**
L'objectif étant d'augmenter la productivité des programmeurs **en automatisant une partie des activités** et en **simplifiant les opérations**.

L'IDE le plus couramment utilisé est (**RStudio**), elle est utilisable sur la plupart des plates-formes (Windows, Linux, MacOS), **c'est celui qu'on utilisera dans ce cours**.

La console R

Le langage R est **un langage interprété**.

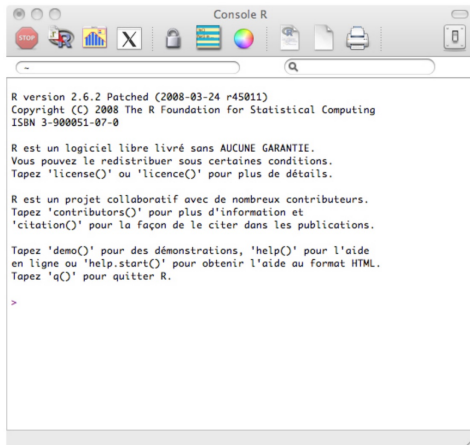
On écrit **une ligne de code**, on la **valide** (<Entrée>) et on **observe le résultat** (pas besoin d'une étape préalable de compilation du code).

Pour écrire du code en R on peut simplement **lancer une console**, obtenir un prompt "<" et **taper du code**. Pour lancer une console:

- **sous Windows**, lancer le programme **Rxxx** (xxx correspondant à la version) dont un raccourci a été créé sur le bureau après l'installation.
- **sous Mac OS X**, lancer le programme R présent dans le dossier Applications.
- **sous Linux** (et plus généralement tout système Unix), ouvrir un terminal et lancer la commande **R**.

On essaie et on observe... on est sous linux

La console R sous MACOS



```
R version 2.6.2 Patched (2008-03-24 r45011)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

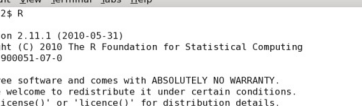
R est un logiciel libre livré sans AUCUNE GARANTIE.
Vous pouvez le redistribuer sous certaines conditions.
Tapez 'license()' ou 'licence()' pour plus de détails.

R est un projet collaboratif avec de nombreux contributeurs.
Tapez 'contributors()' pour plus d'information et
'citation()' pour la façon de le citer dans les publications.

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

>
```

Console R sous Mac OS X



```
bash-3.2$ R

R version 2.11.1 (2010-05-31)
Copyright (C) 2010 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

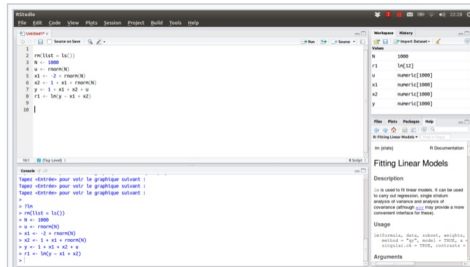
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> █
```

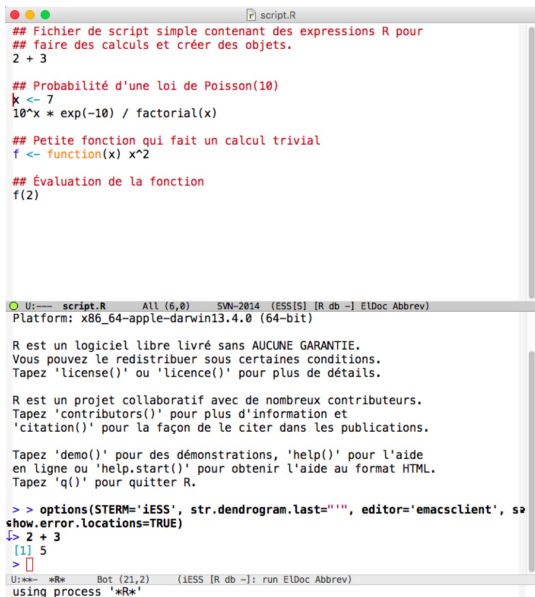
R dans un Terminal sous Linux.



R avec l'environnement de développement intégré

RStudio.

Pour les utilisateurs d'**emacs**, installer "**ESS**" (Emacs Speaks Statistics) pour avoir la **coloration syntaxique**, la **complétion**, etc ...



```
## Fichier de script simple contenant des expressions R pour
## faire des calculs et créer des objets.
2 + 3

## Probabilité d'une loi de Poisson(10)
k <- 7
10^x * exp(-10) / factorial(x)

## Petite fonction qui fait un calcul trivial
f <- function(x) x^2

## Évaluation de la fonction
f(2)
```

U:~ script.R All (6,0) SVN-2014 (ESS[S] [R db -] ElDoc Abbrev)
Platform: x86_64-apple-darwin13.4.0 (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
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'information et
'citation()' pour la façon de le citer dans les publications.

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

```
> > options(STERM='iESS', str.dendrogram.last="", editor='emacsclient', s
show.error.locations=TRUE)
[1] 5
>
```

U:~ *R* Bot (21,2) (iESS [R db -]: run ElDoc Abbrev)
using process '*R*'

Pour lancer R:

- **sous Windows**, lancer le programme Rxxx (xxx correspondant à la version) dont un raccourci a été créé sur le bureau après l'installation.
- **sous Mac OS X**, lancer le programme R présent dans le dossier Applications.
- **sous Linux** (et plus généralement tout système Unix), ouvrir un terminal et lancer la commande "R" (sans les guillemets).

Les **interfaces graphiques** sont différentes entre les différents systèmes d'exploitation mais cela ne change rien au **langage qui sera compatible avec toutes les plates-formes**.

Pour quitter le logiciel R:

> q()

- Vous pouvez **sauvegarder la session**: R demandera "**Save workspace image?**"; répondre **y** (Yes)
Il créera deux fichiers dans le répertoire de travail: .Rhistory et .RData.
- **Réouvrir une session**, vous retrouver **l'environnement** et **l'historique** de la session sauvegardée.

> ls() # vérifier votre espace de travail

R dans une console - comme une calculatrice

```
> 42.1 ; 39 + 3 ; 8 / 3 # Faire des opérations
[1] 42.1
[1] 42
[1] 2.666667
> 4<7 # valeur logique
[1] TRUE
> log(0.3/(1-0.3)) ; pi/sqrt(3) # Utiliser des fonctions intrinsèques
[1] -0.8472979
[1] 1.813799
> 10 %% 3 # Reste de la division (modulo:  $10 = (3 \times 3) + 1$ )
[1] 1
> 10 %/% 3 # Partie entière de la division
[1] 3
> 5^3 # Des puissances
[1] 125
> ((10 + 15) / 5) - 3*2 # Combiner plusieurs opérateurs
[1] -1
```

R language de programmation

Comme dans tout **langage de programmation**, les valeurs peuvent être stockées dans une **variable** et réutilisées.

Dans R **tout est un objet** : entier, réel, chaîne de caractères, fonctions, tableaux,...

Un objet peut être créé avec l'**opérateur d'assignation** qui s'écrit `<-` ou `->`

```
> n <- 4 # Créer un objet
```

```
> n # afficher
```

```
[1] 4
```

```
> 3*n+2 -> nx # utiliser l'objet stocké
```

```
> x <- runif(8) # 8 réels au hasard dans [0;1]
```

```
> x
```

```
[1] 0.2675082 0.2186453 0.5167968 0.2689506 0.1811683 0.5185761 0.56
```

```
[8] 0.1291569
```

```
> mean(x)
```

```
[1] 0.3329481
```

Espace de travail(workspace)

Les **objets** sont stockés dans **un espace de travail** pendant la session.

```
> # Créer des objets
> n <- 4
> nx <- 3*n+2
> x <- runif(8) # 8 réels au hasard dans [0;1]
> ls()
[1] "n"  "nx" "x"
> ls.str() # affiche des details sur les objets en mémoire
n :  num 4
nx :  num 14
x :  num [1:8] 0.256 0.718 0.961 0.1 0.763 ...
> rm(x)      # supprimer un objet
> ls(pattern="n")
[1] "n"  "nx"
> myfunc <- function() {y <- 1; ls()}
> myfunc()      # shows "y"
[1] "y"
```


- Fonction centrale **plot**:

```
> xx <- seq(0,1,length=50) ; fx <- sin(2*pi*xx)
> plot(x=xx,y=fx)
```

- Variables **qualitatives**

```
> xx <- c(1,4,4,3,1,3,4,4,1,1,2,3,4)
> barplot(xx) # diagramme bâton
```

- Variables **quantitatives**

```
> x <- rnorm(100)
> hist(x,main="histogramme de x") # histogramme de x
> boxplot(x) # boîte à moustache
```

- **Graphiques 3D**

```
> x <- seq(-10,10,length=30) ; y<- x # définition du maillage
> f <- function(x,y) 10*sin(sqrt(x^2 + y^2))/sqrt(x^2+y^2)
> z <- outer(x,y,f) # évaluation de f sur le maillage (x,y)
> image(x,y,z) # forme d'image
> persp(x,y,z) # forme de nappe
> contour(x,y,z) # les contours
```

- ...

✧ **Aide en ligne** En cours de session, vous pouvez obtenir de l'aide en ligne.

- **Sur des fonctions**

- > `help(solve)` # ou `?solve`

- **Sur des détails de programmation**

- > `help("[")` # encadré d'une simple ou double quote

- > `help("if"); help("for")`

- Sur la plupart des implémentations de R, **une aide au format HTML**

- > `help.start()`

- Informations sur l'implémentation de algorithmes dans les méthodes

- > `help.search("binomial")` # ou `??binomial`

Liste toutes les méthodes qui font références au mot spécifié.

✧ **Tutoriaux**



Bibliothèques ou package

Un **package** ou **bibliothèque de programmes externes**, est un **ensemble de programme R**, qui complète les fonctionnalités de R.

- **Liste des package installés**

```
> library()  
> mat <- available.packages() # Packages disponibles (au CRAN)
```

- **Charger un package** (si la librairie est déjà chargée, ne fait rien)

```
> library(grDevices)  
> sessionInfo()      # packages chargés en cours de session
```

- **Liste des fonctions d'un package**

```
> library(help="stats") #fonctions du package "stats"
```

- **Liste des datasets disponibles comme exemples sous R**

```
> data()  
> data(package="cluster") # Datasets pour le package cluster
```

Installer package

- **Installation d'un package en tant qu'administrateur**

```
> install.packages("gplots") # sous root
```

- **Mise à jour d'un package** Certains packages sont en constantes évolution avec de nouvelles versions disponibles. Il est préférable de les mettre à jour de temps en temps.

```
> update.packages()
```

Attention, la mise à jour d'un package est fonction de la version de R. Il faudra mettre à jour R régulièrement.

L'utilisateur peut créer ses propres packages (R avancé), les diffuser sur le CRAN ou pour l'usage de sa propre communauté.

Installer d'un package en local

- Utilisation de la variable d'environnement **R_LIBS** qui fournit le/les répertoire(s) d'installation en local des packages

```
export R_LIBS="/home/your_username/R/lib
```

```
> install.packages("gplots") # en local dans R_LIBS  
> .libPaths() # liste des répertoires de recherche des librairies
```

- Si vous ne voulez pas mettre à jour la variable à chaque installation
Placer la commande export dans le fichier **.bashrc** de **votre home directory**, elle sera exécutée à chaque session.

ou, dans un fichier **.Renvi** de votre home directory, la ligne

```
export R_LIBS="/home/your_username/R/lib
```

Vous pouvez indiquer plusieurs répertoires, séparés par **":"**

- Exécutez la fonction **.libPaths()** et observez.

RStudio est **environnement de développement** intégré pour **R**. Il forme une interface utilisateur simple structurée autour d'**une barre de menu** et de **diverses fenêtres**, dont l'**arrangement peut être reconfiguré**. Il propose des outils:

- **Console R**.
- **Editeur**: écrire des scripts, les sauvegarder et les exécuter.
- **Files**: un navigateur de fichiers.
- **Espace de travail**: affiche les objets utilisés en cours de session.
- **Historique**: fournit l'historique des commandes.
- **Packages**: gestion des installations et des packages pour la session en cours.
- **Help**: affichage des help, accès aux tutoriaux,...
- **Plot**: gestions des sorties graphiques.
- **Viewer**:

RStudio est **Open-Source**(libre), il est disponible <http://www.rstudio.com/>

Interface RStudio

The screenshot displays the RStudio IDE interface. The main window shows a script file named 'CalculIndices_new.R' with R code for loading data, setting the working directory, and transforming objects. The console at the bottom shows the R startup message and the current working directory. The sidebar on the right contains the Environment, History, Files, Plots, and Packages panels, as well as a Help/Viewer panel with links to R resources and manuals.

```
# Lecture de l'objet de type sobolroalhs et des quantites d'interet
#
setwd("~/sensibilite/ModEcoMar/dev/RLHS/AS/data")
load("sobol_rlhs_74_1e+5.RData")
load("QI_RLHS_1e+5.RData")
#
# Chargement des scripts (donnees manquantes traitees)
# |
library("sensitivity", lib.loc="/Library/Frameworks/R.framework/Versions/3.2/Resources")
# transformer l'objet sobolroalhs pour la nouvelle version
obj_rlhs_new <- sobolroalhs(model=NULL, factors=obj_rlhs$factors, runs=obj_rlhs$levels,
obj_rlhs_new$X <- obj_rlhs$X
obj_rlhs_new$permu1 <- obj_rlhs$permu1
obj_rlhs_new$permu2 <- obj_rlhs$permu2
#
# Quantites d'interet
# Remplace les valeurs aberrantes par NaN (boxplot)
# nInvalid : vecteur des quantites d'interet. Les valeurs aberrantes
8:3 (Top Level) ±
```

Console: ~/formation/cours-introR/Presentation/ ↵
Platform: x86_64-apple-darwin16.1.0 (64-bit)

R est un logiciel libre livré sans AUCUNE GARANTIE.
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'information et
'citation()' pour la façon de le citer dans les publications.

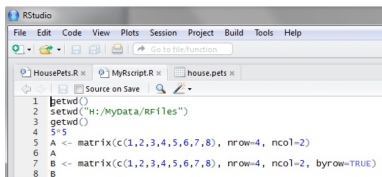
Tapez 'demo()' pour des démonstrations, 'help()' pour l'aide
en ligne ou 'help.start()' pour obtenir l'aide au format HTML.
Tapez 'q()' pour quitter R.

```
> setwd("~/formation/cours-introR/Presentation")
> |
```

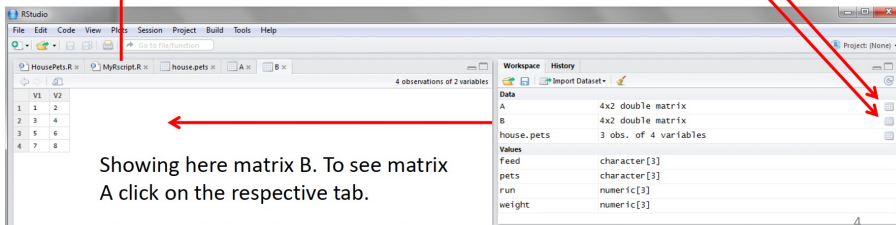
Environment History Files Plots Packages
New Folder Delete Rename More
Home > formation > cours-introR > TP
Name Size Modified
..
Données-R
Environnement-R
Graphiques-R
Programmer-R
Help Viewer
Home > Find in Topic
R Resources
Learning R Online
CRAN Task Views
R on StackOverflow
Getting Help with R
RStudio
RStudio IDE Support
RStudio Cheat Sheets
RStudio Tip of the Day
RStudio Packages
RStudio Products
Manuals
An Introduction to R
Writing R Extensions
R Data Import/Export
The R Language Definition
R Installation and Administration
R Internals
Reference
Packages
Search Engine & Keywords
Miscellaneous Material

Onglet "Environnement"

Tous les objets créés pendant la session R sont stockés dans l'espace de travail et peuvent être visualisés.



```
1 getwd()
2 setwd("H:/Mydata/Rfiles")
3 getwd()
4 5*5
5 A <- matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2)
6 A
7 B <- matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2, byrow=TRUE)
8 B
```



4 observations of 2 variables

	V1	V2
1	1	2
2	3	4
3	5	6
4	7	8

Showing here matrix B. To see matrix A click on the respective tab.

Workspace History

Import Dataset

Data

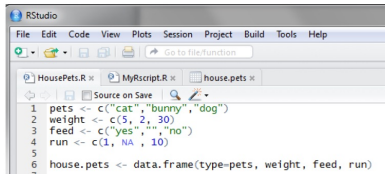
A	4x2 double matrix
B	4x2 double matrix
house.pets	3 obs. of 4 variables

Values

feed	character[3]
pets	character[3]
run	numeric[3]
weight	numeric[3]

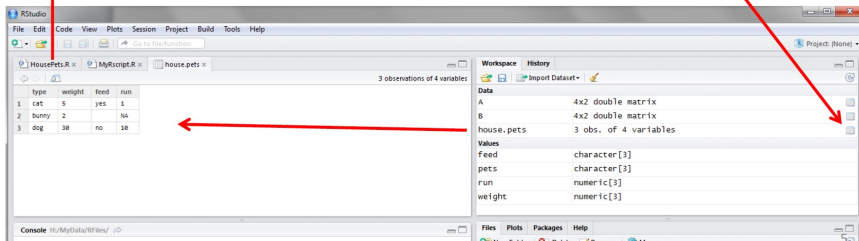
DSS/OTR

Onglet "Environnement" (suite)



```
1 pets <- c("cat", "bunny", "dog")
2 weight <- c(5, 2, 30)
3 feed <- c("yes", "", "no")
4 run <- c(1, NA, 10)
5
6 house.pets <- data.frame(type=pets, weight, feed, run)
7
```

Click on the dotted square to look at the dataset in a spreadsheet form.



The Environment pane displays the 'house.pets' data frame with 3 observations of 4 variables. A red arrow points from the text above to the dotted square icon next to 'house.pets'. Another red arrow points from the 'house.pets' entry to the Environment pane table below.

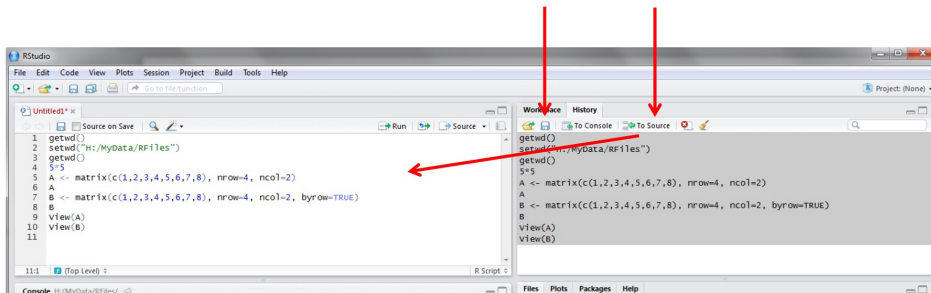
	type	weight	feed	run
1	cat	5	yes	1
2	bunny	2		NA
3	dog	30	no	10

DSS/OTR

Onglet "Historique"

L'onglet **Historique** conserve un enregistrement de **toutes les commandes précédentes**. On pourra:

- **sauvegarder** dans un script **R sous l'éditeur**, l'intégralité de la liste ou une partie, pour garder une trace du travail (cliquer sur "to Source").
- **exécuter** une partie de la liste ou l'intégralité dans **la console R** (cliquer sur "to Console").



R utilise des fonctions et des opérateurs qui agissent sur des objets.
Les principaux **modes** ou **types** de ces objets sont:

Mode	Contenu de l'objet
numeric	nombres réels
complex	nombres complexes
logical	valeurs booléennes (vrai/faux)
character	chaînes de caractères
function	fonction
list	données quelconques
expression	expressions non évaluées

La fonction **typeof** permet d'obtenir une description plus précise de la **re-présentation interne d'un objet**.

```
> f <- "R"
> mode(f)
> typeof(f)
> ff <- c(3, "E")
> ff[1]
> mode(3L)
> mode(3)
> typeof(3L)  # 3L : entier 3
> typeof(3)   # 3 : réel double précision
```