(Re)découverte de R… en 1h45

Ecole de Bioinformatique AVIESAN-IFB – Roscoff – Octobre 2020

Thomas Denecker – thomas.denecker@france-bioinformatique.fr
Jacques van Helden – jacques.van-helden@univ-amu.fr
Stevenn Volant - stevenn.volant@pasteur.fr

Slides d'Hugo Varet – hugo.varet@pasteur.fr

aviesan
alliance nationale
pour les sciences de la vie et de la santé

ifb

CNRS UPMC
Station Biologique
Roscoff

R en quelques mots

Langage de programmation qui permet de :

1. manipuler des données : importer, transformer, exporter
2. faire des analyses statistiques plus ou moins complexes : description, exploration, modélisation...
3. créer des (jolies) figures

Disponible sur RCRAN 

Historique :

- 1993 : début du projet R
- 2000 : sortie de R 1.0.0
- 2020 : R 4.0.2

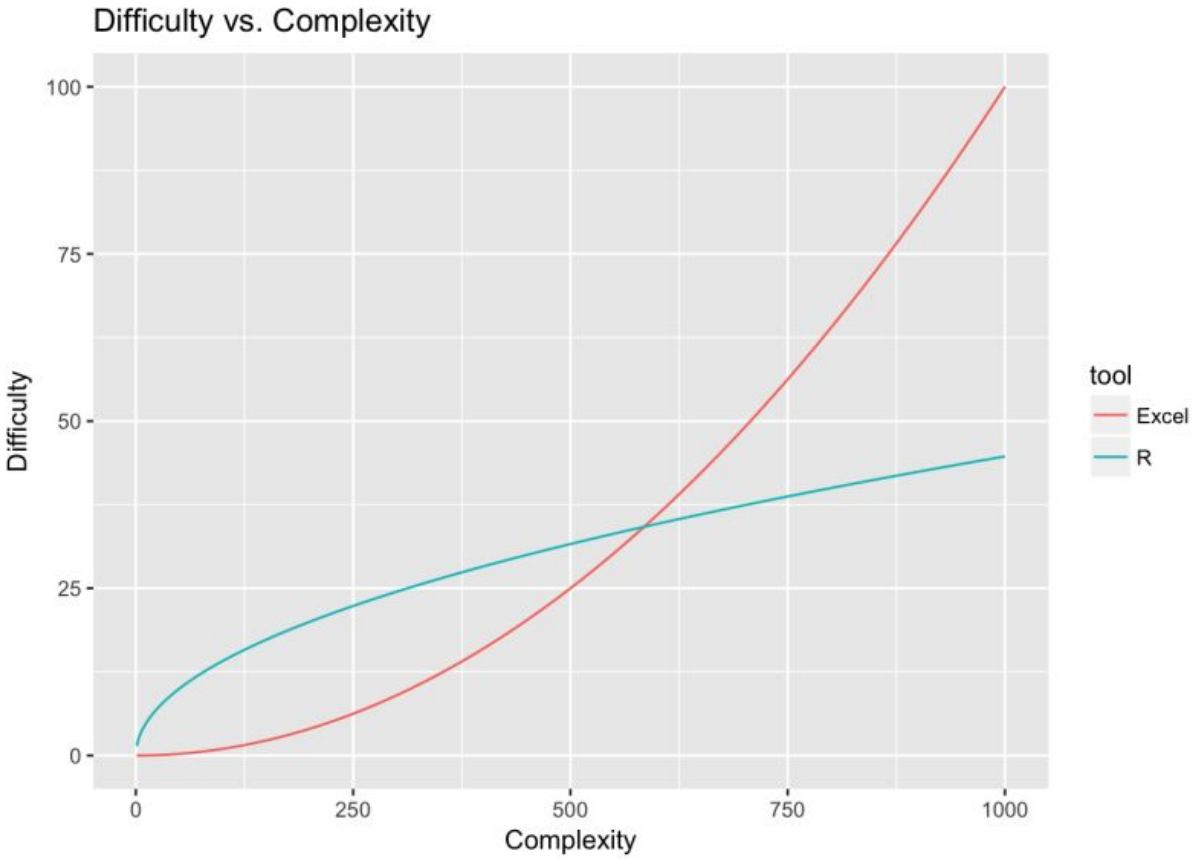
Avantages et inconvénients

Avantages :

- Souplesse d'utilisation pour réaliser des analyses statistiques
- Libre et gratuit, même s'il existe maintenant des versions payantes de RStudio (shiny et/ou server)
- Reproductibilité des analyses en écrivant/sauvegardant les commandes R dans des scripts
- Large communauté d'utilisateurs/aide en ligne
- Grand nombre de packages spécifiques

Inconvénients :

R vs Excel



Covid : le Royaume-Uni passe à côté de milliers de cas à cause... d'un fichier Excel arrivé à saturation

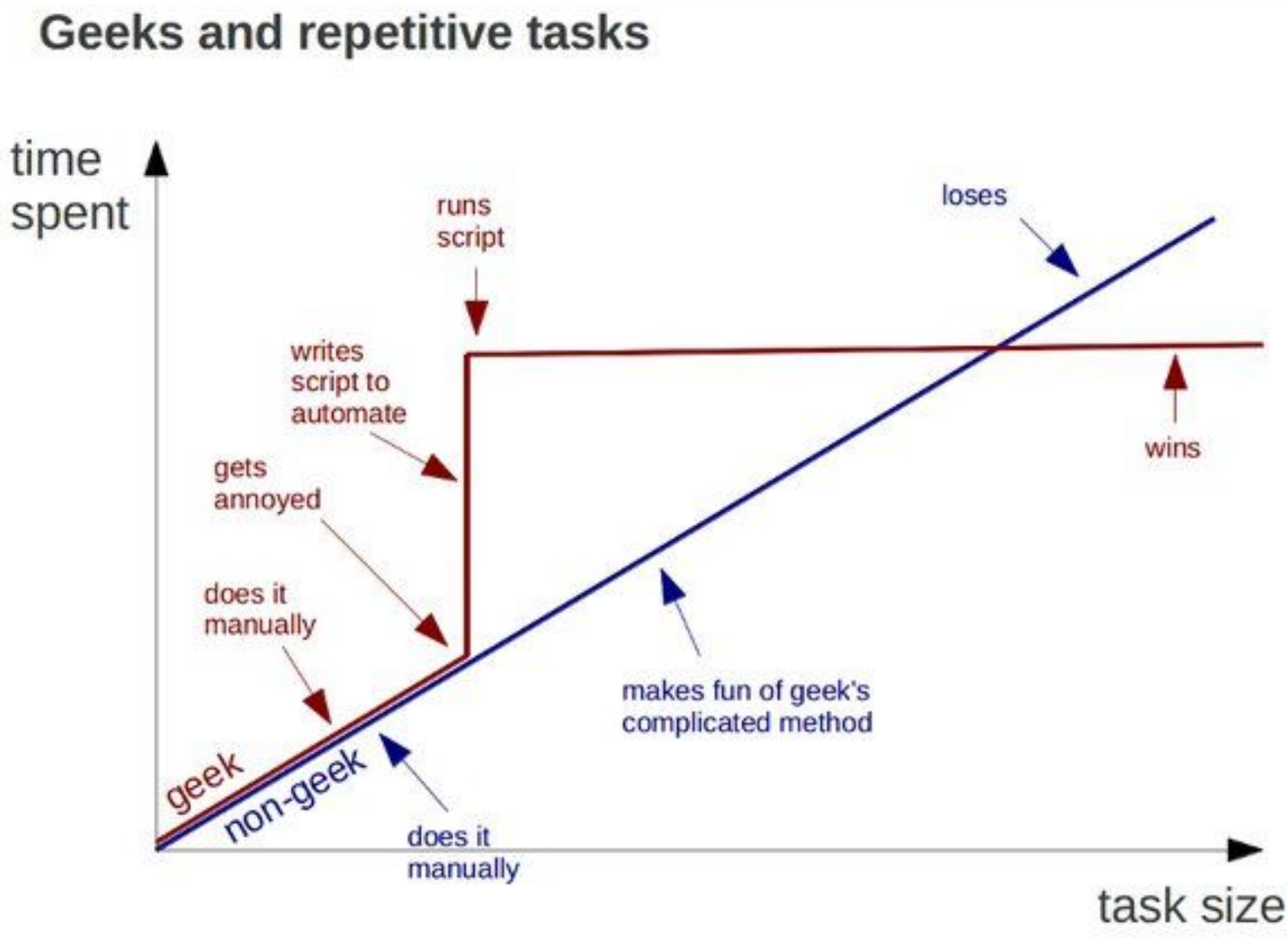
Les autorités sanitaires britanniques ont reconnu que près de 16.000 cas de coronavirus en Angleterre sont passés sous le radar au cours de la semaine écoulée à cause d'un problème dans le chargement des données.

Lire plus tard | Europe | Partager | Commenter

[Alexandre Counis, Les Echos, 5 oct. 2020]

Source: R-bloggers

Geeks and repetitive tasks



R sait tout faire

Lire un tableau de données	`read.table()`
Fusionner deux tableaux	`merge()`
Filtrer des lignes	`data[data$x > 10]`
Sélectionner des colonnes	`data[,c("x","y")]`
Rechercher une chaîne de caractères	`grep()`
Réaliser une ACP	`prcomp()`
Calculer une moyenne	`mean()`
Additionner deux matrices	`mat1 + mat2`
Exporter un tableau de données	`write.table()`
Calculer une variance	`var()`
Régression linéaire	`lm()`
Tracer une courbe	`plot()`
Tester une hypothèse	`t.test()`
Dessiner un histogramme	`hist()`
Convertir des données	`as.matrix()`

Modes d'utilisation (liste non exhaustive)

Localement via le terminal

Localement via RStudio (utilisation classique)

Sur un serveur via le terminal et une connexion ssh

Sur un serveur via un navigateur web pour accéder à RStudio server

Ouverture ou connexion à RStudio

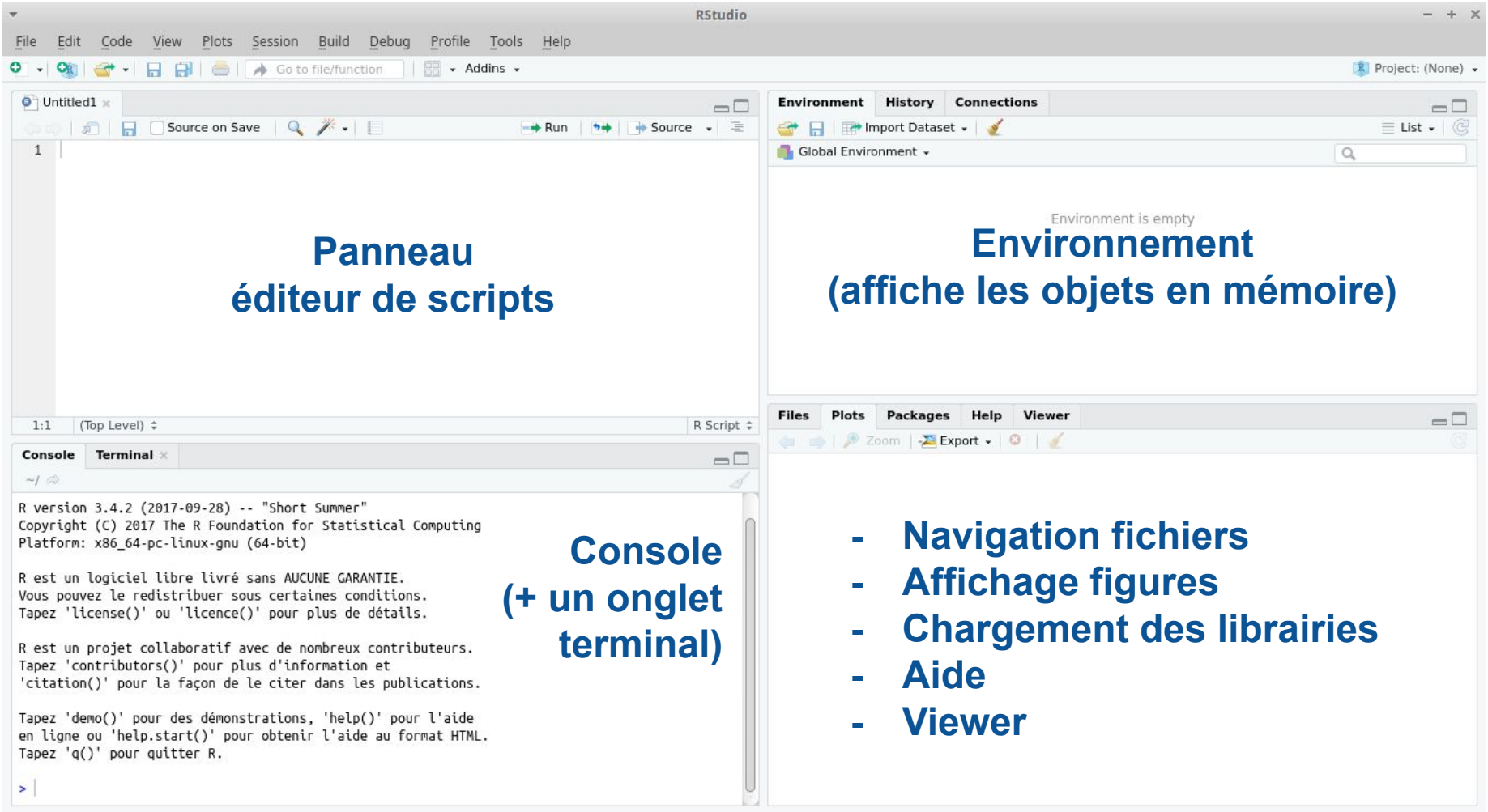
3 alternatives :

1. Ouvrir RStudio sur votre propre ordinateur (si installé)
2. Vous connecter au **serveur Web RStudio de l'IFB**
   https://rstudio.cluster.france-bioinformatique.fr
   puis vous identifier

3. Vous connecter via **Jupyter lab de l'IFB**
   https://jupyterhub.cluster.france-bioinformatique.fr
   puis cliquer sur l'icône RStudio

RStudio

- Disponible depuis 2011
- Logiciel facilitant l'utilisation de R via 4 panneaux
- Chaque panneau présente plusieurs onglets (fonctionnalités complémentaires)



R sait tout faire : il compte

Tapez les commandes suivantes dans le panneau Console de RStudio

```
2 + 3
```

```
4 * 5
```

```
6 / 4
```

```
1:10
```

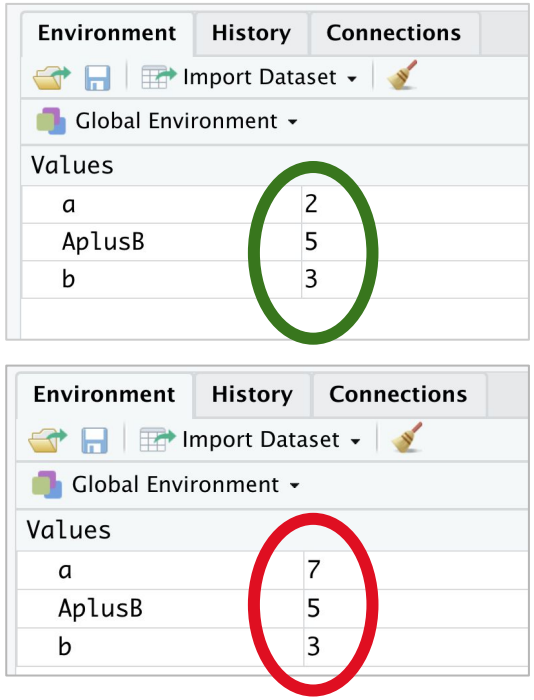
```
8:-9
```

```
1,2
```

```
1.2
```

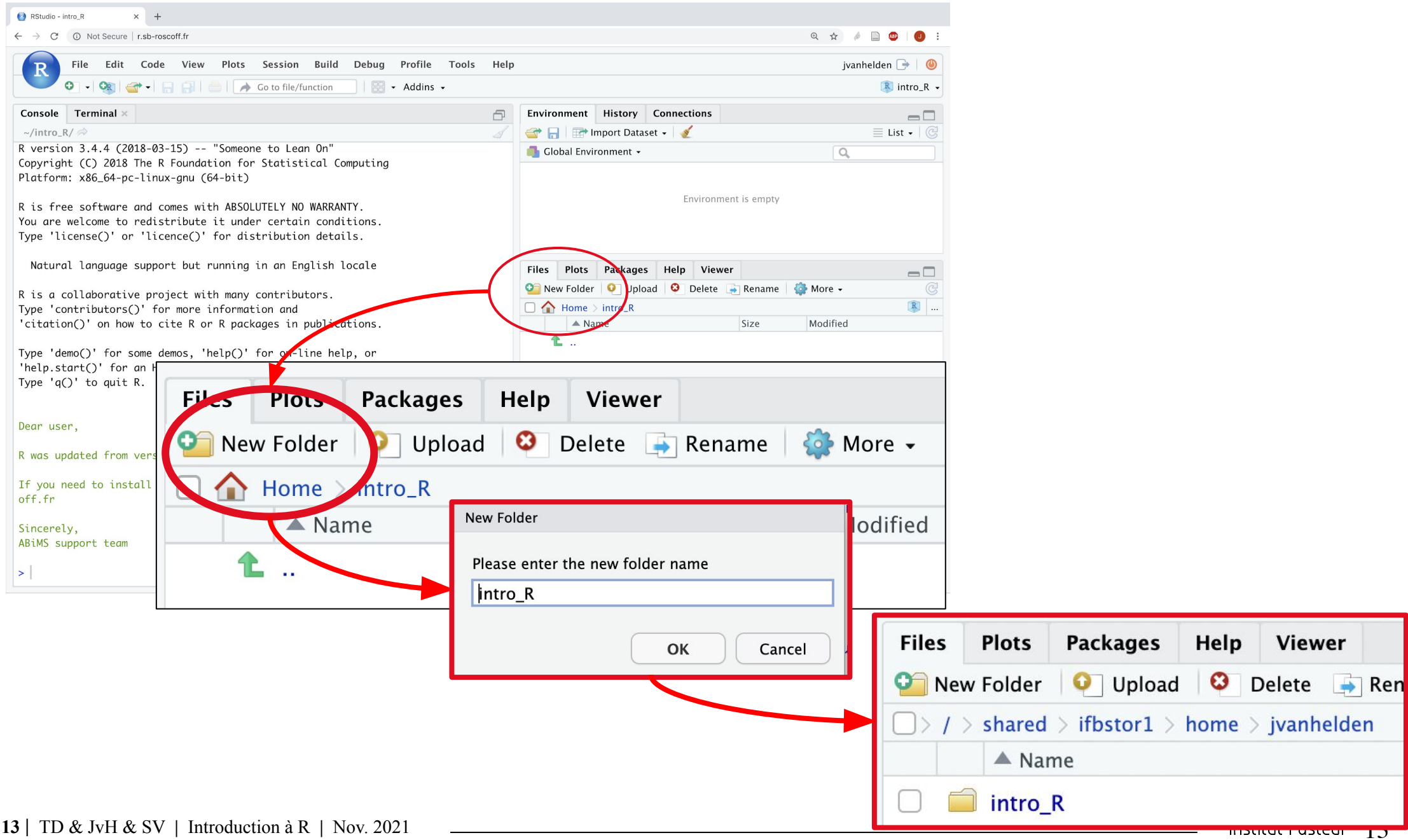
Notion de variable/objet

```
a <- 2              ## Créer une variable nommée a et lui assigner une valeur
print(a)            ## Afficher la valeur de la variable a
a                   ## Même résultat: si on évoque le nom de variable, R l'imprime


b <- 3              ## Assigner une valeur à une seconde variable
a_plus_b <- a + b   ## Effectuer un calcul avec 2 variables
print(a_plus_b)     ## Afficher le contenu de la variable a_plus_b

a <- 7              ## Changer la valeur de a
print(a_plus_b)     ## Note: le contenu de a_plus_b n'est pas modifié

a_plus_b <- a + b   ## On recalcule a_plus_b
print(a_plus_b)     ## La nouvelle valeur tient compte de la modification de a

vec1 <-  c(1,10)    ## Créer un vecteur
vec2 <-  1:10       ## Créer un vecteur contenant une séquence d'entiers de 1 à 10
vec2 + a            ## Somme d'un vecteur et d'un nombre
vec3 <-  c("riri", "fifi", "loulou")    ## Vecteur de chaînes de caractères
vec2 / 2                                ## Diviser un vecteur de nombres par un nombre
vec3  / 2                               ## Diviser des chaînes de caractères par un nombre

## Noms de variables interdits: TRUE, FALSE, T, F, c, t, pi, data, LETTERS, letters, ...
```



Cas pratique : données d'expression

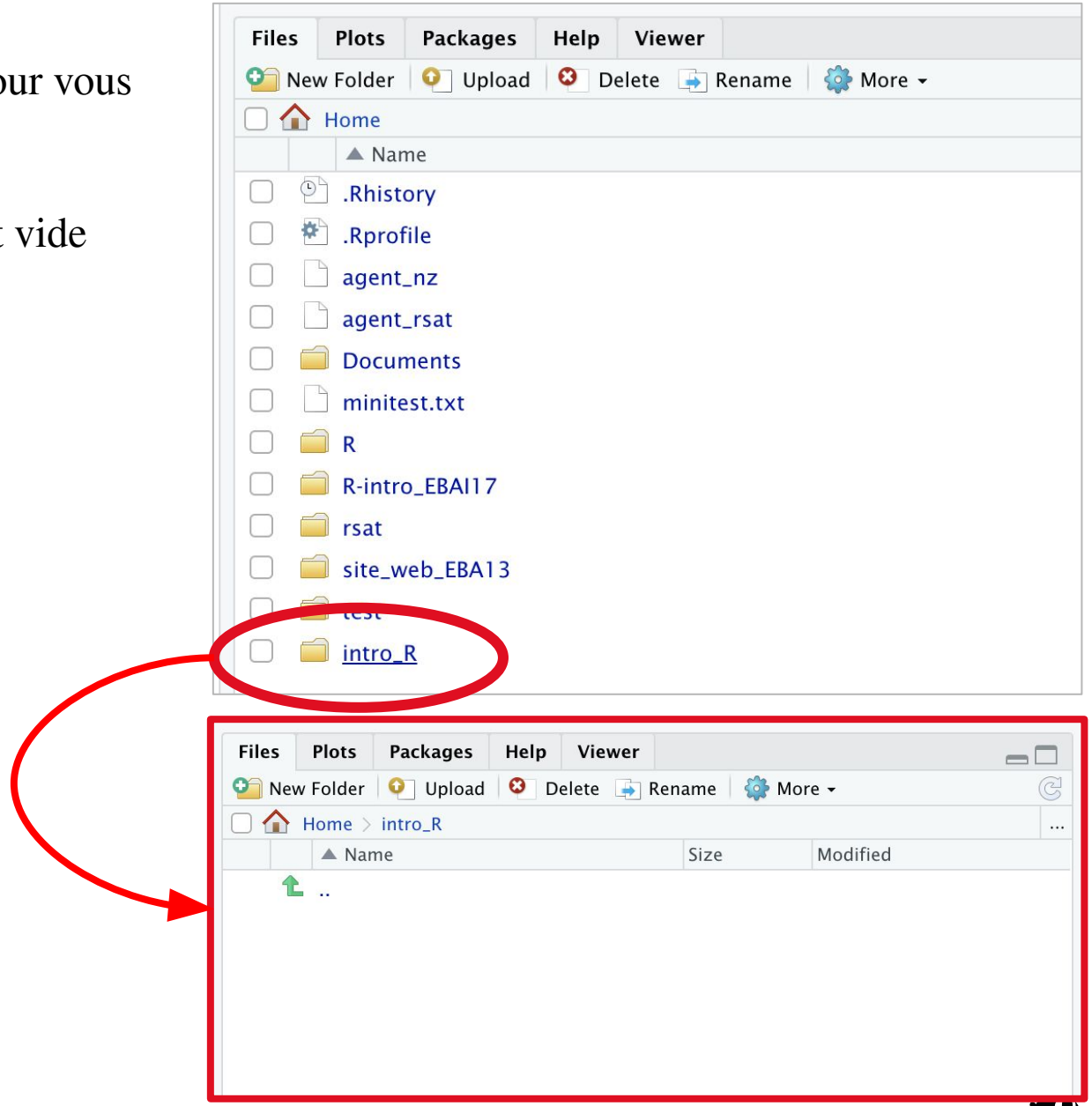
Création d'un dossier `intro_R` pour vos résultats de ce TP



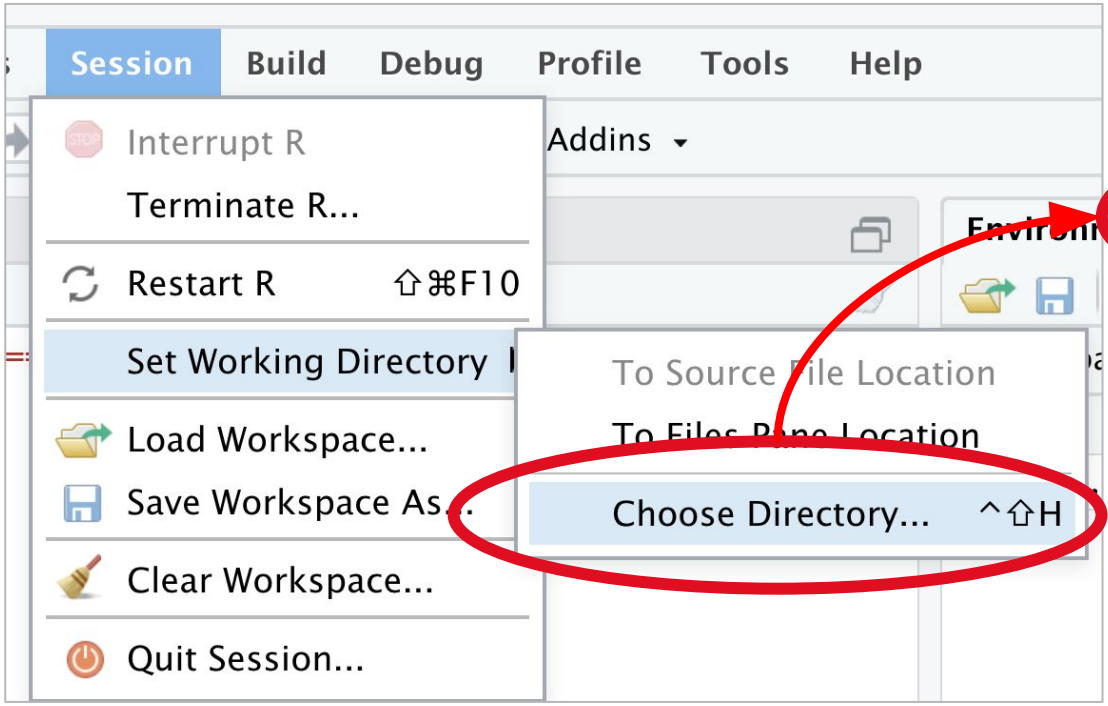
Déplacement dans le dossier “intro_R”

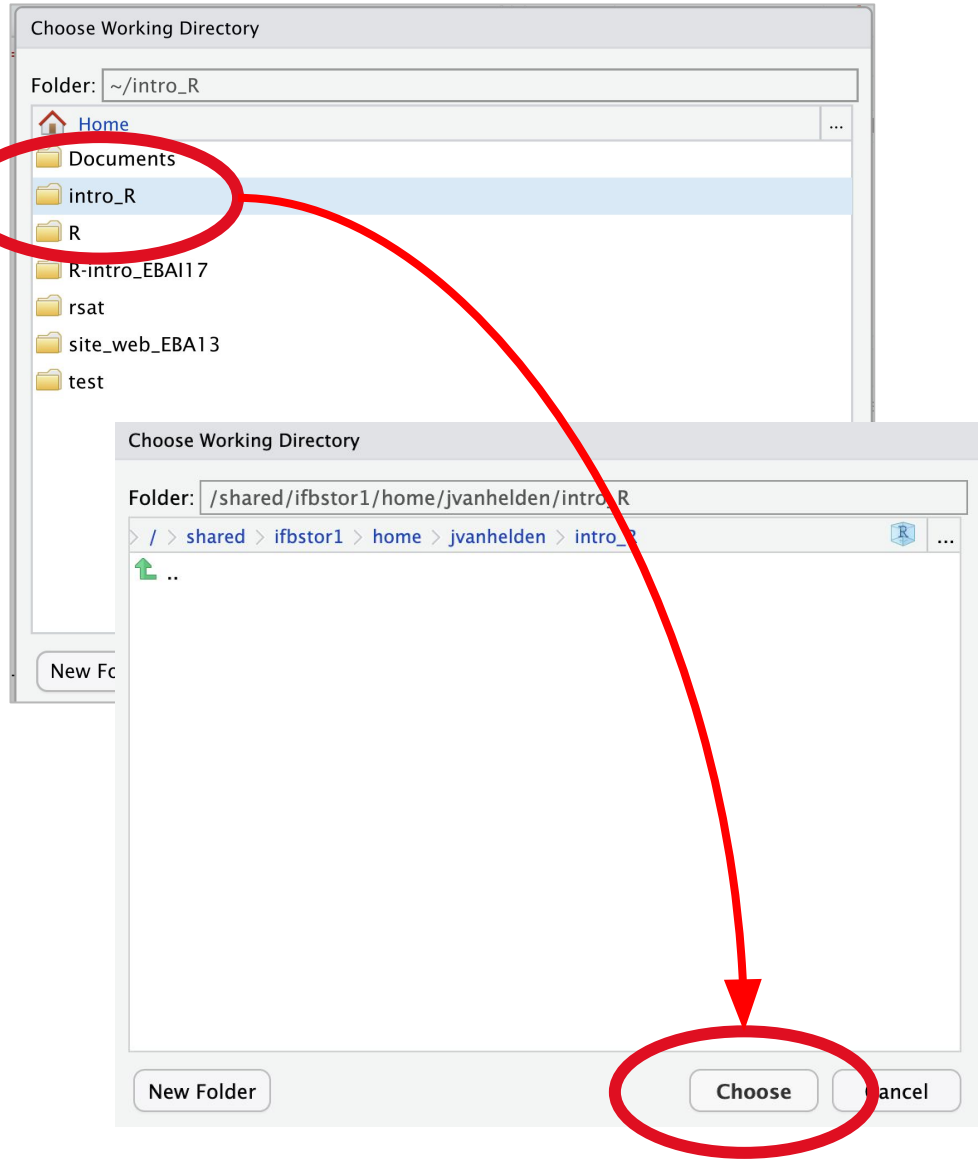
Double-cliquez sur le dossier “intro_R”, pour vous y déplacer.

Puisque vous venez de créer le dossier il est vide (image du bas).



Définissez votre dossier espace de travail (working directory)





1. Dans le menu "**Session**", lancez "**Choose Directory …**"
2. **Naviguez** jusqu'à votre dossier `intro_R`
3. **Double-cliquez** dessus pour l'ouvrir
4. Cliquez **Choose**

Téléchargez les fichiers sur votre machine

A partir d'un navigateur Web, téléchargez et enregistrez **sur votre ordi** les fichiers de données

- `expression.txt`: données d'expressions pour 4 échantillons
- `annotation.csv`: informations sur les gènes (id, name, chr, start, stop)

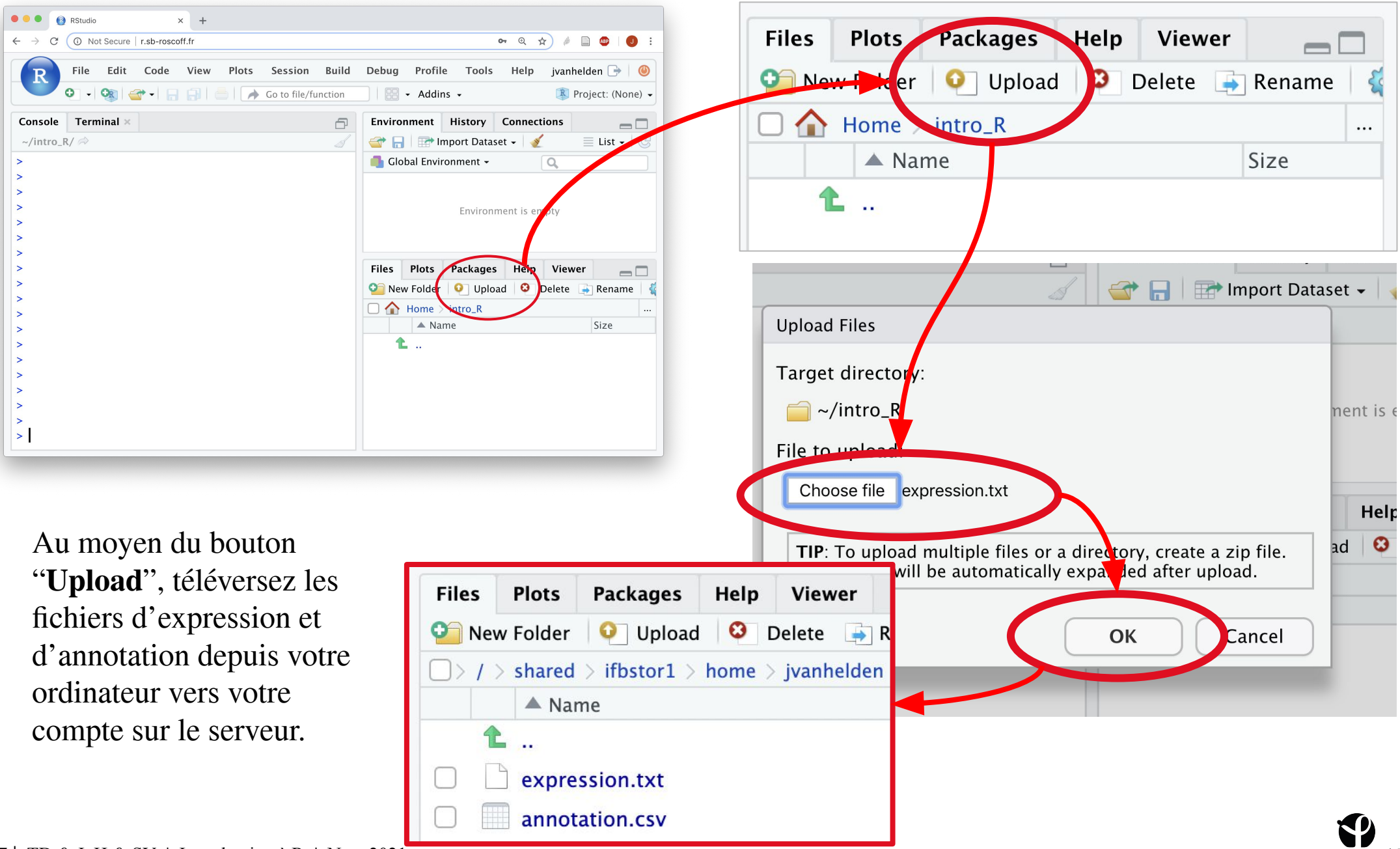
https://tinyurl.com/ebaii-expression-txt

https://tinyurl.com/ebaii21-annotation-csv

Attention: veillez à sauvegarder les fichiers

- sous leur nom original,
- avec les extensions .txt et .csv respectives (certains navigateurs omettent l'extension, ce qui poserait problème pour la suite du TP)

Téléversement (“upload”) des données

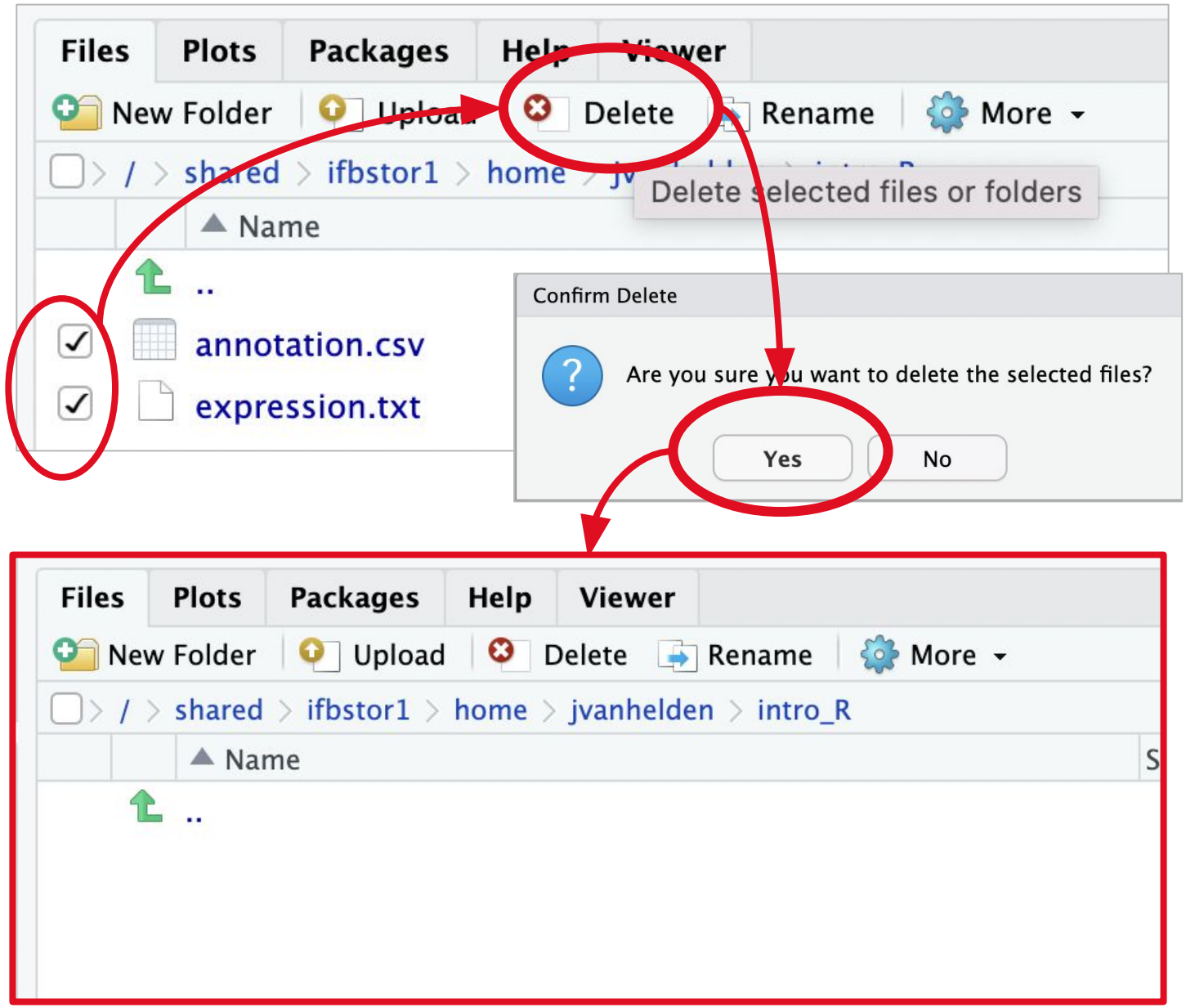


Au moyen du bouton “**Upload**”, téléversez les fichiers d’expression et d’annotation depuis votre ordinateur vers votre compte sur le serveur.

On efface tout et on recommence

1. Sélectionnez les deux fichiers
2. Effacez-les sans pitié

(nous allons vous montrer deux autres façons de les téléverser)



The “R geek” way (V2, directement depuis Rstudio)

Revenir à la maison (note: R interprète le caractère “~" comme le “HOME” de Linux; et cela marche aussi pour Windows!)

```
setwd("~")
```

Définir une variable qui indique le chemin du dossier de travail (working directory).

```
my_work_dir <- "~/intro_R"
```

S’il n’existe pas encore, créer le dossier de travail. (Commande Unix équivalente: "mkdir -p ~/intro_R")

```
dir.create(my_work_dir, recursive = TRUE, showWarnings = FALSE)
```

Où suis-je ? (Commande Unix équivalente: "pwd")

```
getwd()
```

Aller dans ce dossier de travail (Commande Unix équivalente: "cd ~/intro_R")

```
setwd(my_work_dir)
```

Et maintenant, où suis-je ?

```
getwd()
```

Qu'y a-t-il par ici ? (Commande Unix équivalente: "ls")

```
list.files()
dir()        ## Un autre nom pour la même commande
```

Télécharger un fichier : the “geek” way (V2)

Nous avons montré ci-dessus comment télécharger des fichiers en utilisant l’interface graphique de RStudio.

Alternativement, on peut télécharger des fichiers au moyen de la commande R **download.file**.

Les deux commandes suivantes permettent de télécharger les fichiers utilisés pour les exercices.

```
download.file(url = "https://tinyurl.com/ebaii-expression-txt", destfile = "expression.txt")
```

```
download.file(url = "https://tinyurl.com/ebaii21-annotation-csv", destfile = "annotation.csv")
```

Note : équivalent de la commande “wget” sous Unix.

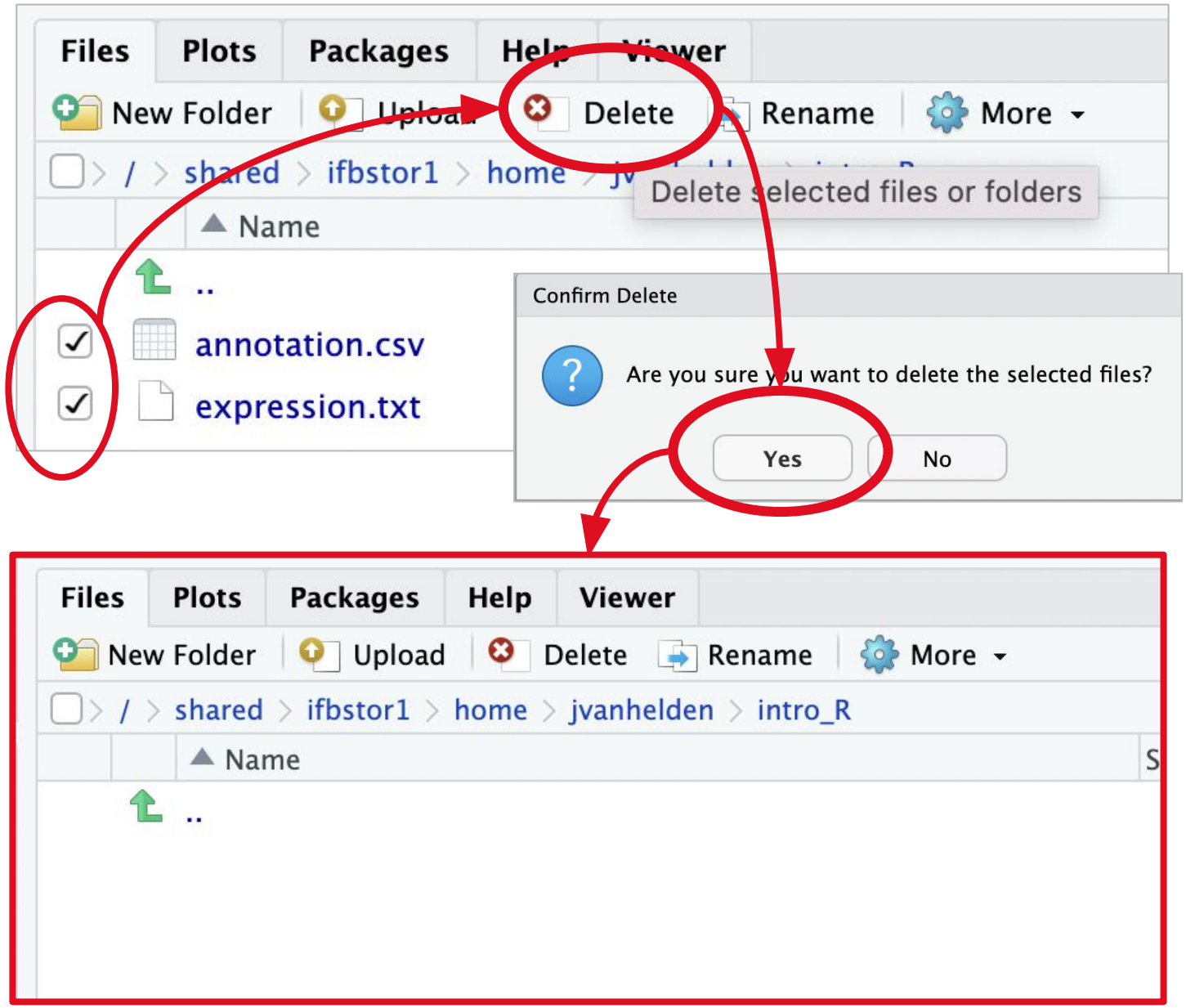
Qu'y a-t-il par ici ? (Commande Unix équivalente: "ls")

```
list.files()
```

On efface tout et on recommence

1. Sélectionnez les deux fichiers
2. Effacez-les sans pitié

(nous allons vous montrer deux autres façons de les téléverser)



The “bash geek” way (V3, directement de votre home du cluster)

Alternative: dans le terminal du cluster, téléchargez et enregistrez dans votre home les fichiers de données:

- `expression.txt`: données d’expressions pour 4 échantillons
- `annotation.csv`: informations sur les gènes (id, name, chr, start, stop)

Ouvrez un connection ssh

```
ssh [votre_login]@core.cluster.france-bioinformatique.fr
```

Où suis-je ?

```
pwd
```

Créez un répertoire “intro_R”

```
mkdir -p ~/intro_R
```

Déplacez-vous dans votre dossier

```
cd ~/intro_R
```

Où suis-je maintenant ?

```
pwd
```

Téléchargez les données

```
wget https://tinyurl.com/ebaii-expression-txt --output-document=expression.txt
wget https://tinyurl.com/ebaii21-annotation-csv -O annotation.csv
```

Qu’y a-t-il ici ?

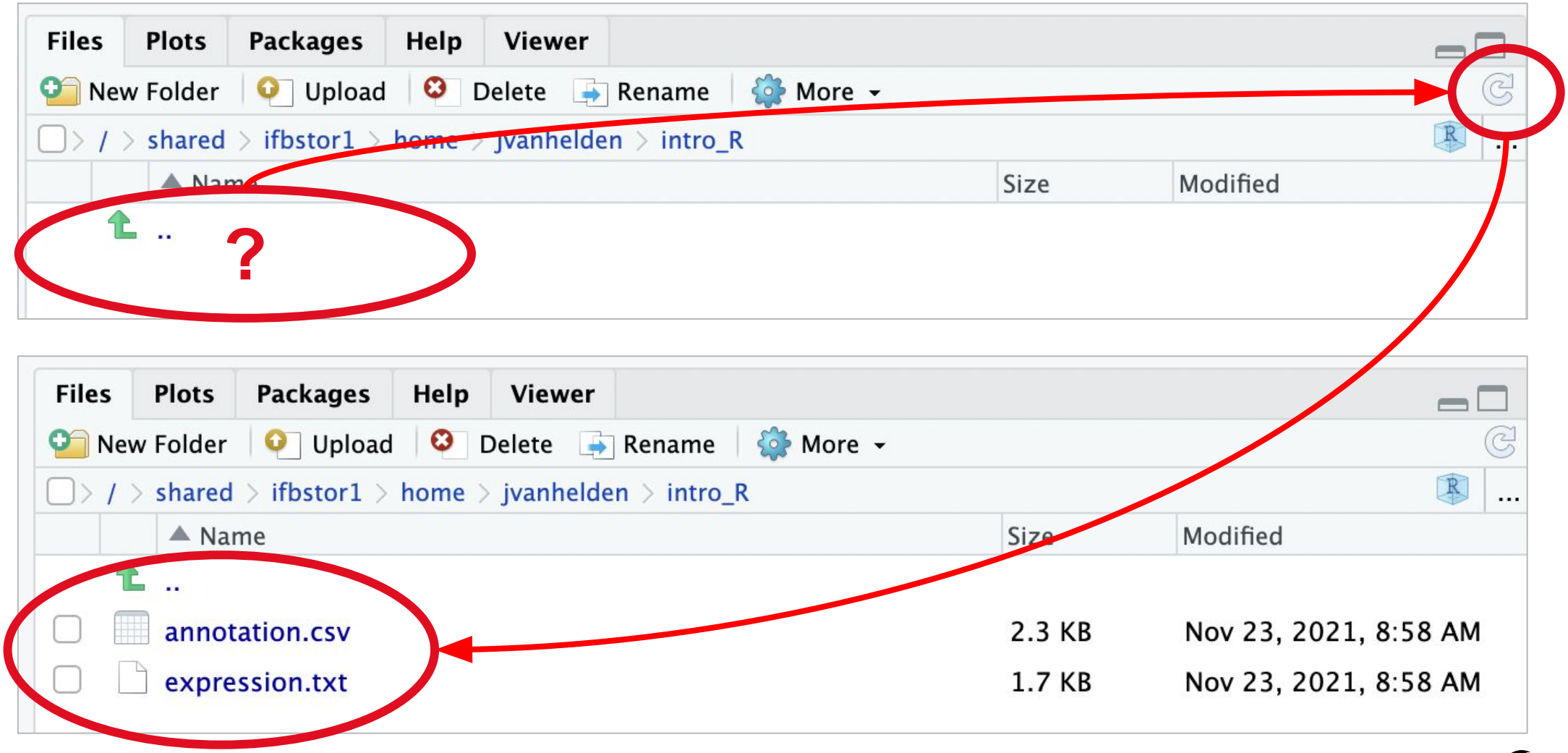
```
ls -l
```

A quoi ressemblent ces fichiers ?

```
less expression.txt
less annotation.csv
```

Actualisation du dossier

Dans certains cas, il faut actualiser le contenu du dossier pour pouvoir voir le nouveau sous-dossier.
Vérifiez ensuite si `intro_R` apparaît bien dans le contenu de votre dossier principal.



Chargement des données (dans la mémoire de R)

Charger le contenu du fichier "expression.txt" dans une variable nommée "exprs".

```
exprs <- read.table(file = "expression.txt", header = TRUE, sep = "\t")
```

Accéder à l'aide d'une fonction

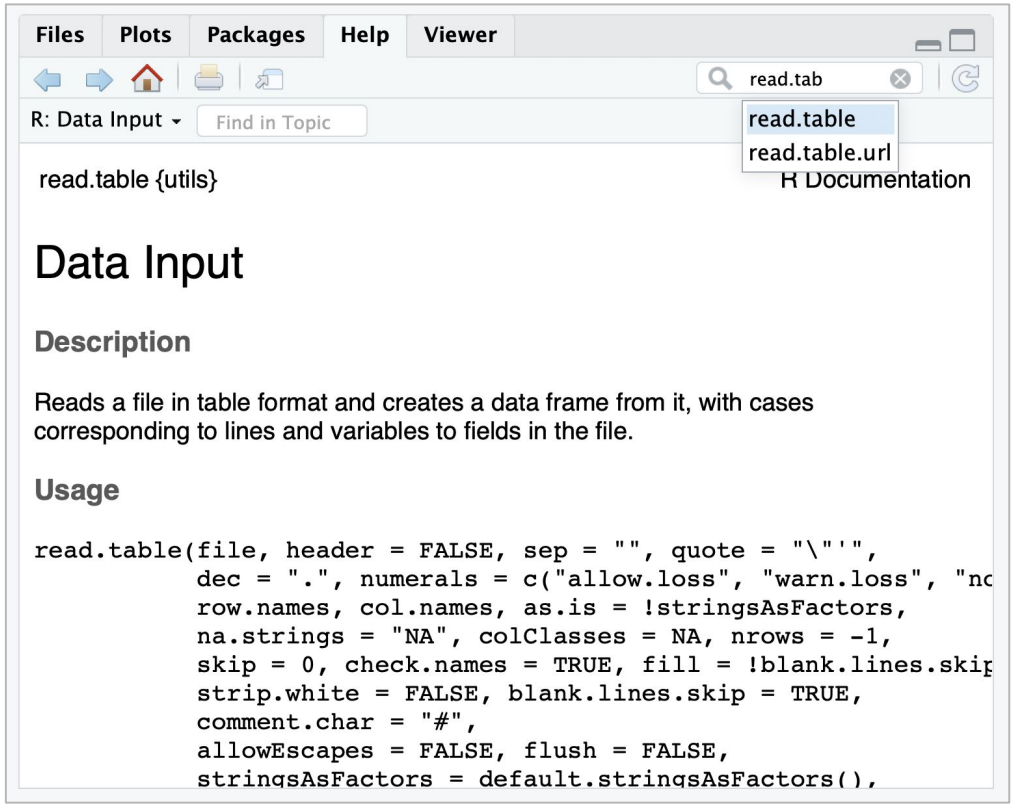
```
help(read.table)
```

Notation alternative

```
?read.table
```

Approche pratique :

1. demande à Google "*Comment lire une table en R ?*"
2. adapte l'exemple



Recherche interactive sous RStudio

- Sélectionner l'onglet "Help" du panneau inférieur droit.
- Taper "read.table" dans la boîte de recherche.

Affichage de l'objet "exprs"

Imprimer toutes les valeurs.

print(exprs)

Affichage des premières lignes de l'objet

head(exprs)

Affichage des dernières lignes de l'objet

tail(exprs)

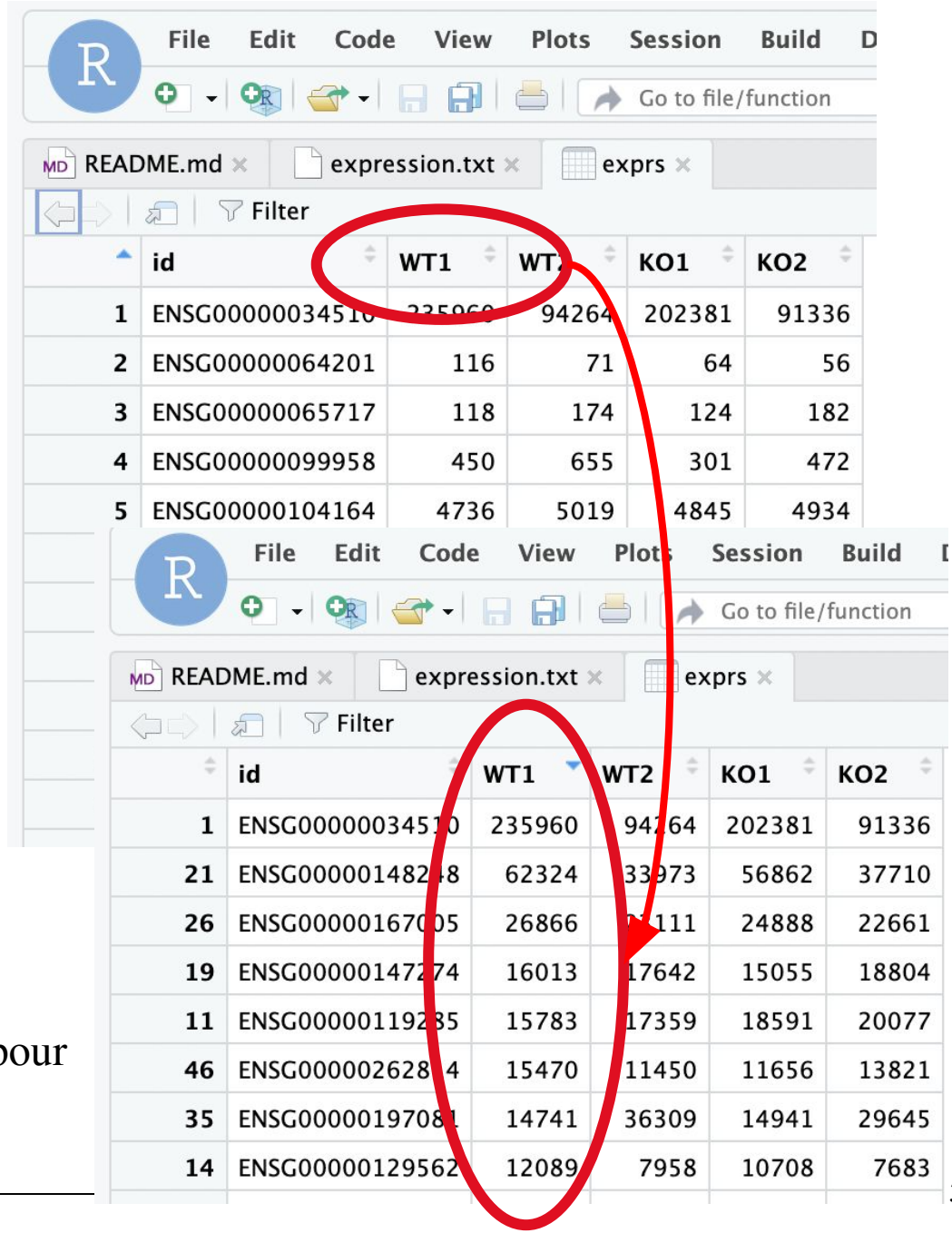
Un peu plus de lignes

head(exprs, n = 15)

Explorer le tableau dans un panneau de visualisation

View(exprs)

Note: vous pouvez cliquer sur une en-tête de colonne pour trier les données



Caractéristiques d'un tableau de données

Dimensions

```
ncol(exprs)      ## Nombre de colonnes
nrow(exprs)      ## Nombre de lignes
dim(exprs)       ## Dimensions
```

Noms des colonnes et des lignes

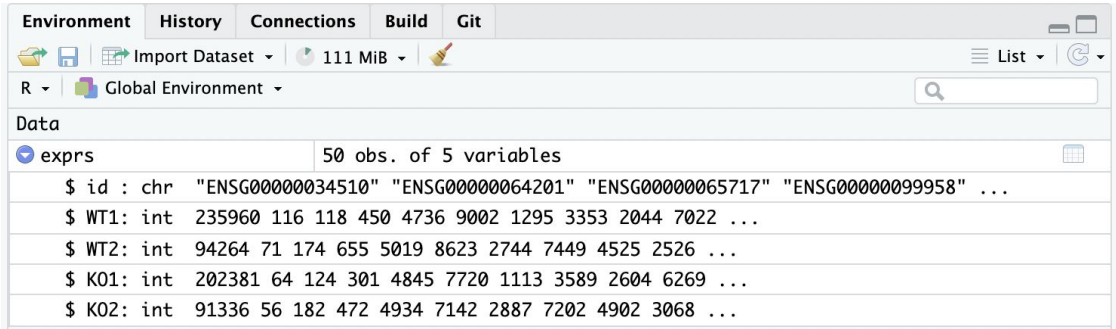
```
colnames(exprs)  ## Noms des colonnes
names(exprs)     ## idem
rownames(exprs)  ## noms des lignes
```

Résumé rapide des données par colonne

```
summary(exprs)   ## Statistiques par colonne
```

```
str(exprs)       ## Structure de la variable
```

Même résultat que dans le panneau "Environment"



Sélection de colonnes d'un tableau

Afficher les noms des colonnes

```
colnames(exprs)
```

Valeurs stockées dans la colonne nommée "WT1"

```
exprs$WT1
```

Notation alternative

```
exprs[ , "WT1"]
```

Sélection de plusieurs colonnes.

```
exprs[ , c("WT1", "WT2")]
```

Sélection de colonnes par leur indice

```
exprs[ , 2]
exprs[ , c( 3, 2)]
```

Histogrammes

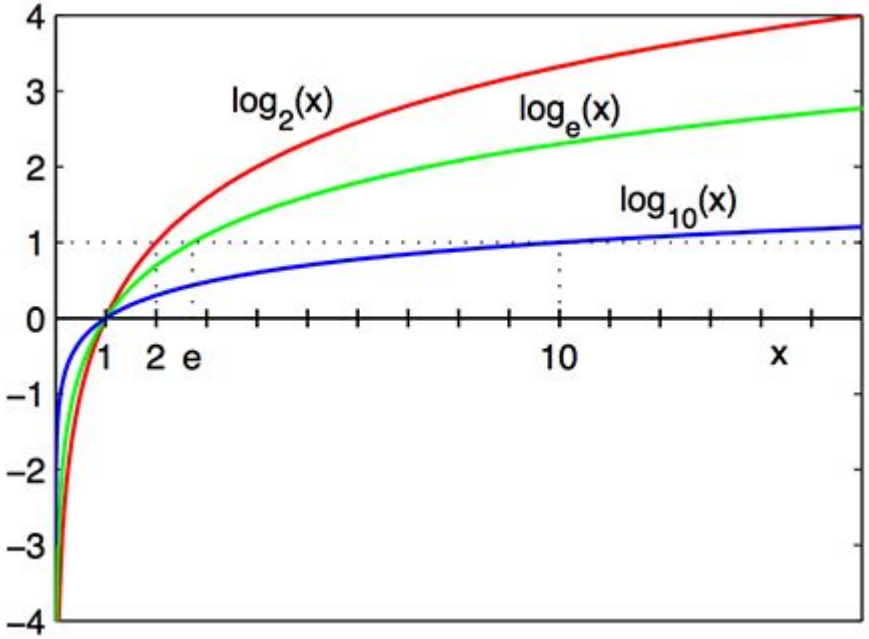
Histogramme des valeurs d'expression pour l'échantillon WT1

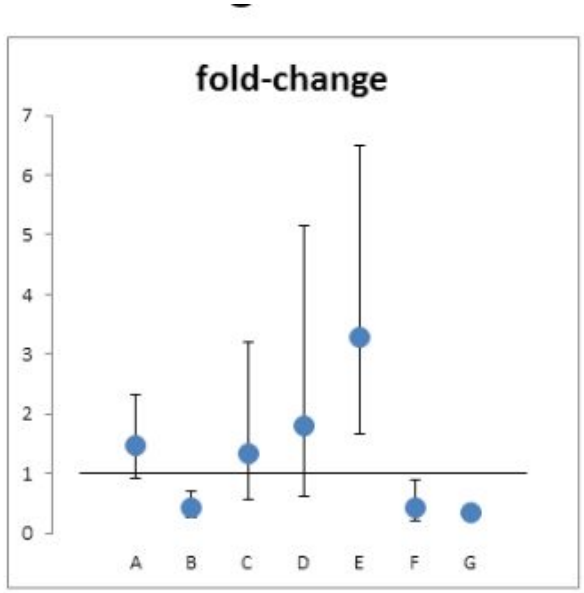
```
hist(exprs$WT1)
```

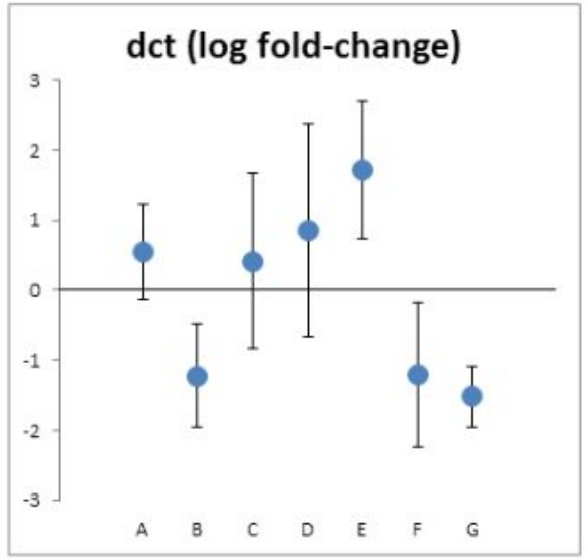
Histogramme du logarithme de ces valeurs

```
hist(log(exprs$WT1))
hist(log2(exprs$WT1))
hist(log10(exprs$WT1))
```

Logarithmes (pour rappel…)







Boîtes à moustaches (boxplots)

Boite à moustache des valeurs d'expression pour l'échantillon WT1

```
boxplot(exprs$WT1)
```

Boite à moustache du logarithme de ces valeurs

```
boxplot(log(exprs$WT1))
boxplot(log2(exprs$WT1))
boxplot(log10(exprs$WT1))
```

Boite à moustache des valeurs d'expression pour les 4 échantillons

```
boxplot(exprs)

boxplot(exprs[,-1])                ## ignorer la première colonne

boxplot(log2(exprs[,-1]))

boxplot(exprs[,-1], log = "y")

boxplot(exprs[,-1], log = "y", las = 1)        ## afficher les étiquettes des axes horizontalement
```

Nuage de points

Expressions KO1 vs WT1

```
plot(x = log(exprs$WT1), y = log(exprs$KO1))
```

Personnalisation des paramètres graphiques

```
plot(x = log(exprs$WT1),                  ## données pour l'abscisse
     y = log(exprs$KO1),                  ## données pour l'ordonnée
     main = "Expression KO1 vs WT1",      ## Titre principal
     xlab = "WT1",                        ## légende de l'axe X
     ylab = "KO1",                        ## légende de l'axe Y
     pch = 16,                            ## caractère pour marquer les points
     las = 1,                             ## écrire les échelles horizontalement
     col = "red")                         ## couleur des points
grid()                                    ## ajout d'une grille
abline(a = 0, b = 1)                      ## ajouter la droite X = Y (intercept a = 0, pente b = 1).
```

Sélection de lignes d'un tableau

Sélection des lignes 4 et 11 du tableau des expressions

```
exprs[c(4, 11), ]
```

Sélection des identifiants de deux gènes d'intérêt

```
mygenes <- c("ENSG00000253991", "ENSG00000099958")
```

Vecteur booléen indiquant si chaque ID du tableau fait partie des gènes d'intérêt

```
exprs$id %in% mygenes
```

Indices des lignes correspondant aux IDs des gènes d'intérêt

```
which(exprs$id %in% mygenes)
```

Afficher les lignes correspondantes

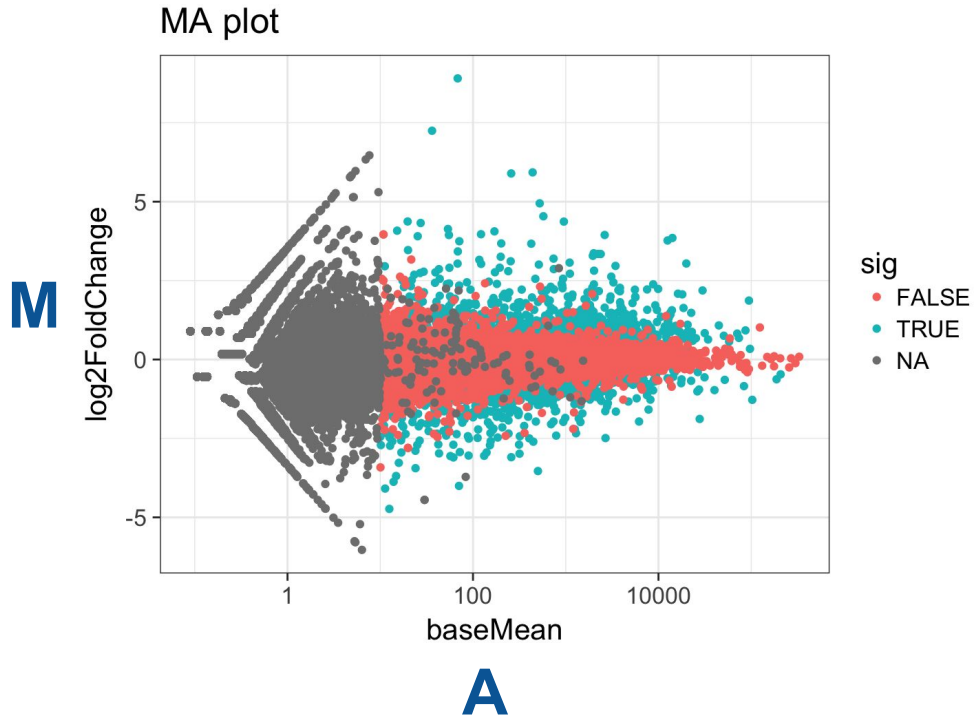
```
exprs[which(exprs$id %in% mygenes),  ]
subset(x = exprs, id %in% mygenes)          ## formulation plus intuitive
```

Approche plus moderne, avec le package dplyr

```
library(dplyr)                              ## charger la librairie dplyr
exprs %>% filter(id %in% mygenes)           ## envoyer le tableau exprs à la commande filter()
exprs %>% filter(id %in% mygenes) %>% mutate(mean_KO = (KO1 + KO2)/2)
## plus avancé : enchaîner plusieurs commandes
```

Analyse d'expression différentielle : MA-plot

```
> head(exprs, 10)
                 id    WT1   WT2    KO1   KO2
1  ENSG00000034510 235960 94264 202381 91336
2  ENSG00000064201    116    71     64    56
3  ENSG00000065717    118   174    124   182
4  ENSG00000099958    450   655    301   472
5  ENSG00000104164   4736  5019   4845  4934
6  ENSG00000104783   9002  8623   7720  7142
7  ENSG00000105229   1295  2744   1113  2887
8  ENSG00000105723   3353  7449   3589  7202
9  ENSG00000116199   2044  4525   2604  4902
10 ENSG00000118939   7022  2526   6269  3068
```



Le MA plot représente le lien entre différence d'expression et intensité moyenne.

- **M (magnitude)** est le logarithme en base 2 du rapport d'expression ("log2 fold-change")
- **A (average intensity)** est la moyenne des logarithmes des valeurs d'expression.

M = log2(KO/WT) = log2(KO) - log2(WT)	log2 fold change, "magnitude"
A = ½ log2(KO x WT) = ½ (log2(KO) + log2(WT))	average log2 value

Calculs sur les colonnes

Calcul de moyennes par ligne (`rowMeans`) pour un sous-ensemble donné des colonnes (WT1 et WT2).

```
rowMeans(exprs[   , c("WT1","WT2")])
```

Ajout de colonnes avec les expressions moyennes des WT et des KO

```
exprs$meanWT <- rowMeans(exprs[   , c("WT1","WT2")])
exprs$meanKO <- rowMeans(exprs[   , c("KO1","KO2")])
head(exprs)                              ## Check the result
```

Fold-change KO vs WT

```
exprs$FC <- exprs$meanKO / exprs$meanWT
head(exprs)                              ## Check the result
```

Moyenne de tous les échantillons

```
exprs$mean <- rowMeans(exprs[   , c("WT1", "WT2", "KO1", "KO2")])
head(exprs)                              ## Check the result
```

M = log2(KO/WT) = log2(KO) - log2(WT) log2 fold change, “magnitude”

A = ½ log2(KO x WT) = ½ (log2(KO) + log2(WT)) average log2 value

MA-plot : log_2FC vs intensité

Calcul de M et A

```
exprs$M <- log2(exprs$FC)
```

```
exprs$A <- (log2(exprs$meanKO) + log2(exprs$meanWT)) /2
```

On peut ensuite dessiner un nuage de points (en l'agrémentant un peu)

```
plot(x = exprs$A, y = exprs$M, main = "MA plot",
     col = "blue", pch = 16, xlab = "A = intensity", ylab = "M = log2FC")
grid(lty = "solid", col = "lightgray")
abline(h = 0)
```

M = log2(KO/WT) = log2(KO) - log2(WT)	log2 fold change, "magnitude"
A = ½ log2(KO x WT) = ½ (log2(KO) + log2(WT))	average log2 value

Appliquer une fonction sur les lignes/colonnes

Appliquer une **fonction** (moyenne, variance, ...) sur chaque **ligne** d'un tableau

```
mean_per_row <- apply(exprs[ , c("WT1", "WT2", "KO1", "KO2")], 1, mean)

mean_per_row <- apply(exprs[ , c(2, 3, 4, 5)], 1, mean)

mean_per_row <- apply(exprs[ , -1 ], 1, mean)

mean_per_row <- apply(exprs[ , which(sapply(exprs, class) != "factor")], 1, mean)

var_per_row <- apply(exprs[ , c("WT1", "WT2", "KO1", "KO2")], 1, var)
```

Appliquer une **fonction** (moyenne, variance, ...) sur chaque **colonne** d'un tableau

```
mean_per_col <- apply(exprs[ , c("WT1", "WT2", "KO1", "KO2")], 2, mean)

var_per_col <- apply(exprs[ , c("WT1", "WT2", "KO1", "KO2")], 2, var)
```

Charger les annotations des gènes

```
read.table(file = "annotation.csv")                          ## pas cool
read.table(file = "annotation.csv", sep = ";")               ## pas parfait
read.table(file = "annotation.csv", sep = ";", header = TRUE)  ## OK
```

```
annot <- read.table(file = "annotation.csv", sep = ";", header = TRUE)
dim(annot)                    ## Vérifier les dimensions
head(annot)                   ## Afficher quelques lignes
```

Combien de gènes par chromosome ?

```
table(annot$chr)
```

Question : combien de gènes sur le chromosome 8 ? Et sur le X ?

```
barplot(sort(table(annot$chr)), horiz = TRUE, las = 1,
        col = "lightblue",  xlab = "Number of genes")
```

Ma première bioinformatique intégrative

- 1ere étape : fusionner les tableaux d'expressions et d'annotations :

```
?merge
exprs_annot <- merge(exprs, annot, by = "id")
head(exprs_annot)
```

- 2eme étape : sous-ensemble des lignes pour lesquelles chr vaut 8 :

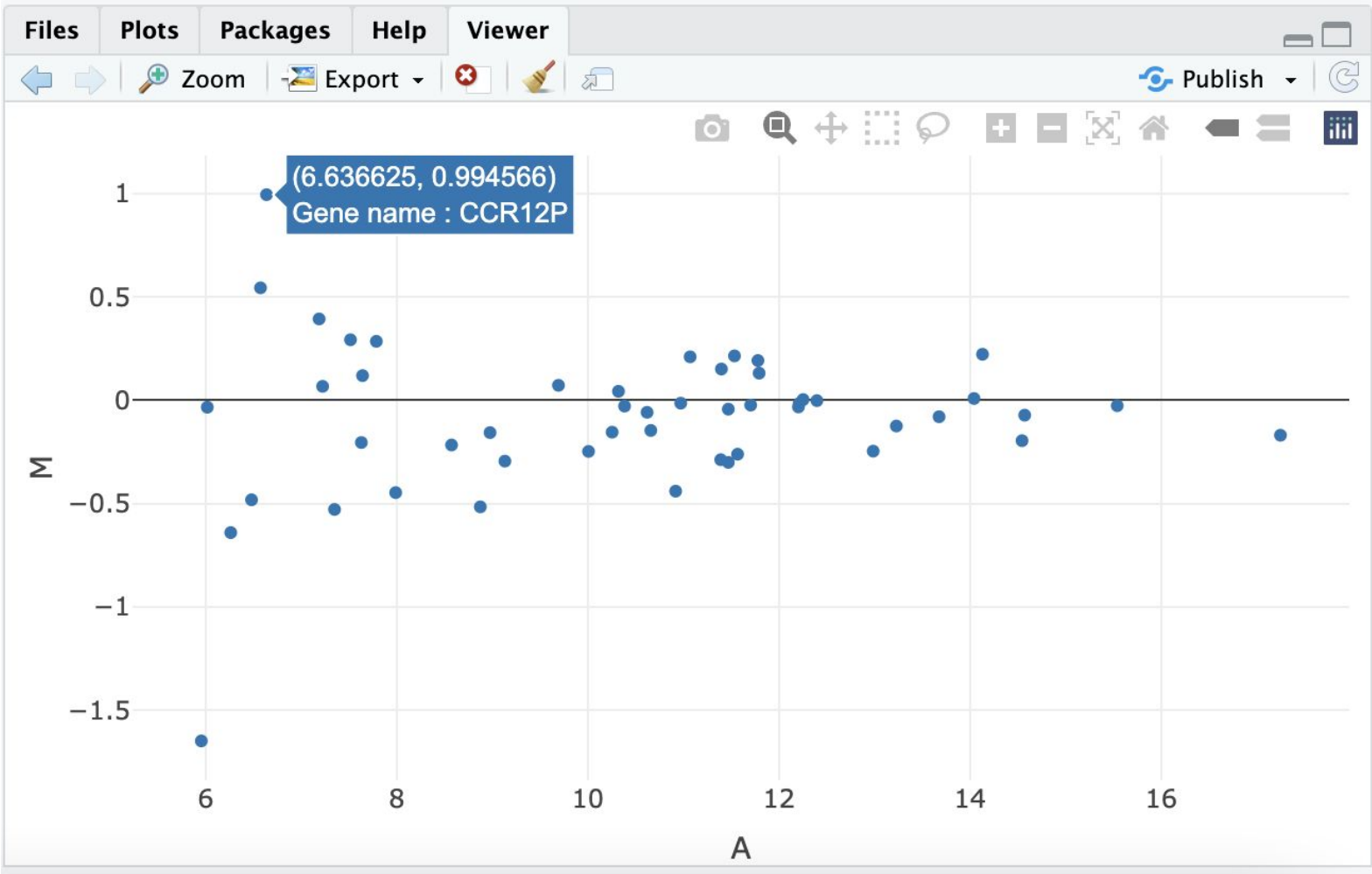
```
exprs_chr8 <- exprs_annot[which(exprs_annot$chr == 8),   ]
print(exprs_chr8)
```

- Exporter exprs_chr8 dans un fichier :

```
write.table(x = exprs_chr8, file = "exprs_chr8.txt",
  sep = "\t",
  row.names = FALSE,
  col.names = TRUE)
```

Mon premier graphique dynamique

```
library(plotly)
plot_ly(data = exprs_annot, x = ~A, y = ~M, text = paste("Gene name :", exprs_annot$name), type = 'scatter', mode = 'markers')
```



Take home messages

- Tout est faisable avec R
- **Définir et comprendre l'opération mathématique/statistique** avant de chercher la fonction R correspondante
- R est un langage :
  - plusieurs types et structures de données (out of scope)
  - énormément de commandes à découvrir (out of scope)
  - Google est votre ami
- Une infinité de :
  - ressources en ligne
  - tutoriels pour des analyses spécifiques (e.g. DESeq2 pour le RNA-Seq)
- Bonnes pratiques : https://style.tidyverse.org/syntax.html

Ressources IFB

Serveur **RStudio**

https://rstudio.cluster.france-bioinformatique.fr/

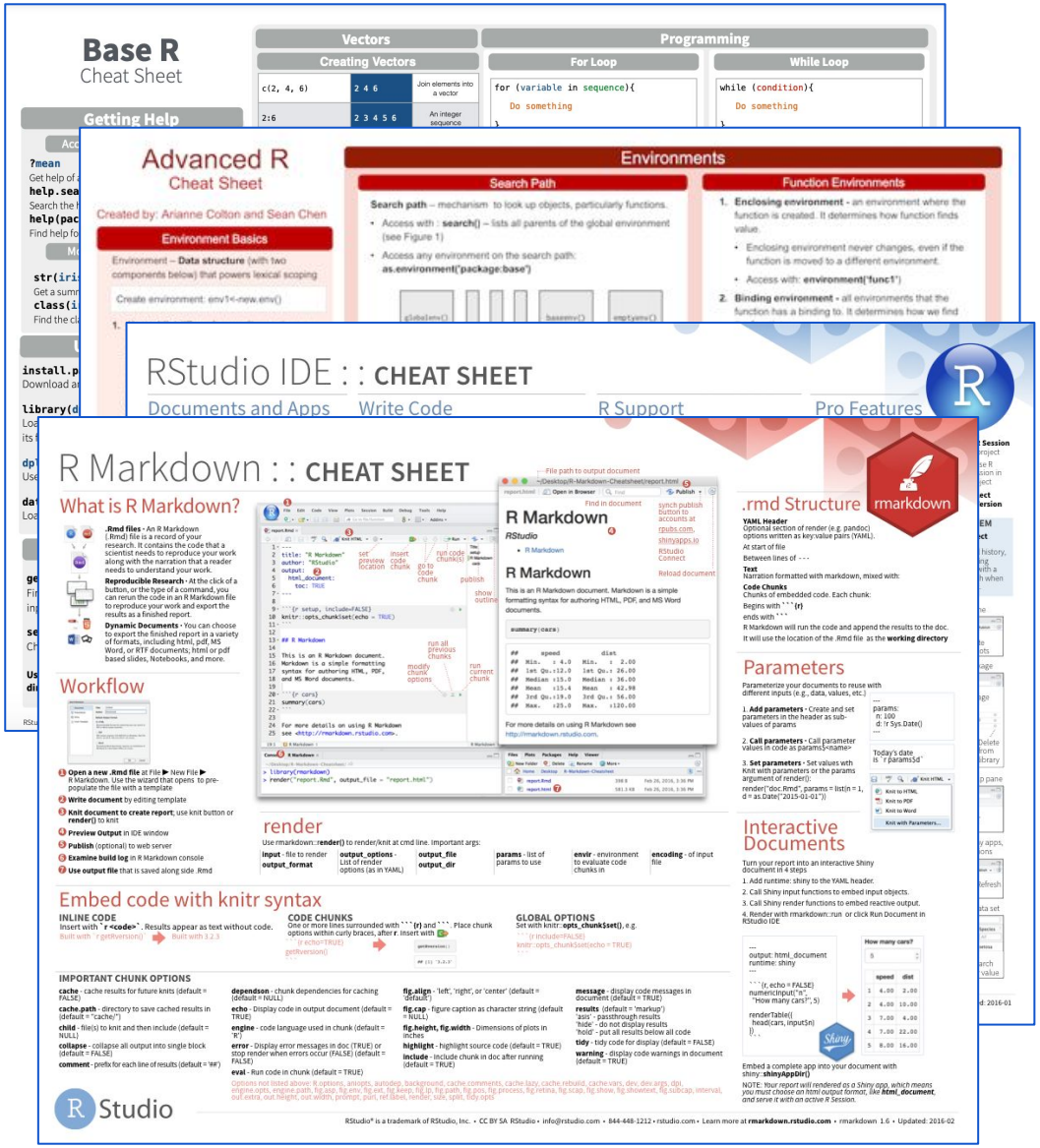
Jupyter lab (inclut un serveur RStudio et plein d'autres choses

http://jupyterhub.cluster.france-bioinformatique.fr/

Une question ? Un besoin ? Un problème ? **Contactez la communauté IFB**

https://community.france-bioinformatique.fr/

Ressources



R https://www.r-project.org/

RStudio https://rstudio.com/

R-bloggers https://www.r-bloggers.com/

THINKR https://thinkr.fr/

Rstudio Cheatsheets (un tas de thèmes):
https://rstudio.com/resources/cheatsheets/