

## Chapitre 2. (re)découverte de R

Andreas Zanzoni ([orcid.org/0000-0002-4818-6161](https://orcid.org/0000-0002-4818-6161))

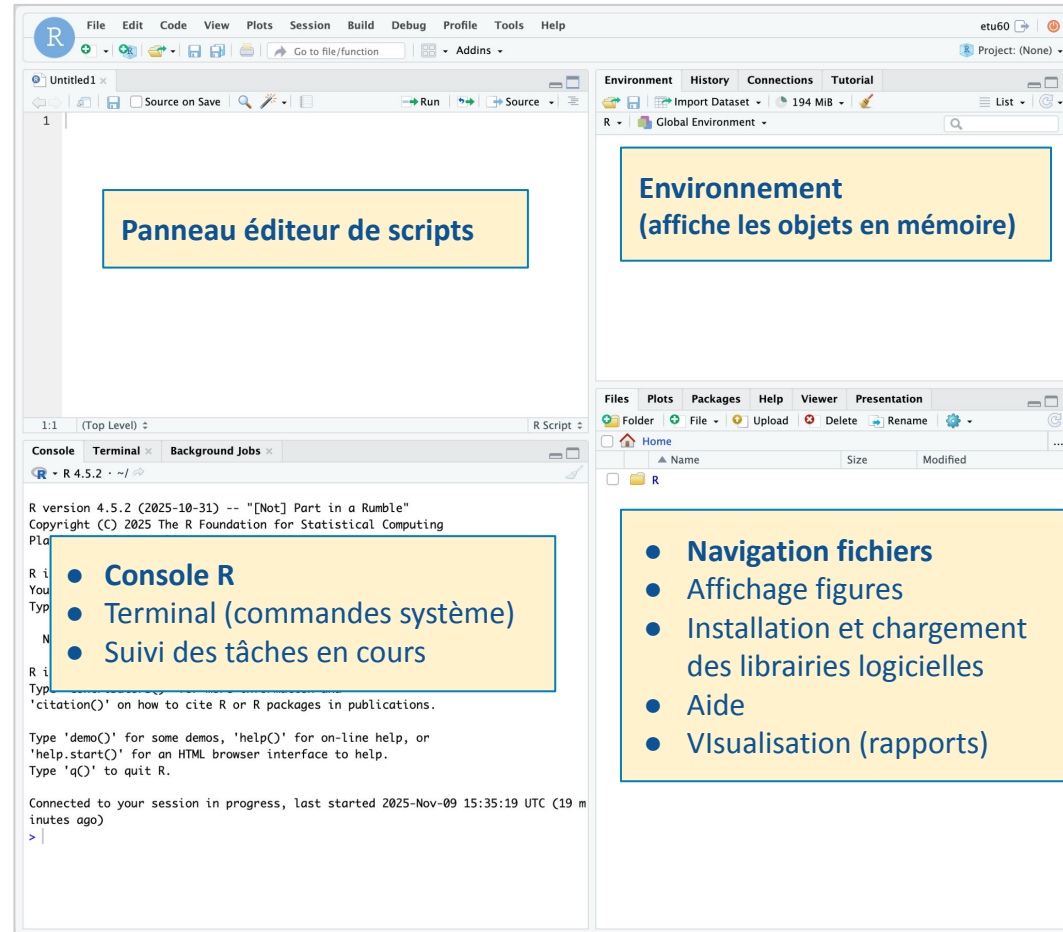
Jacques van Helden ([orcid.org/0000-0002-8799-8584](https://orcid.org/0000-0002-8799-8584))

Lou BERGOGNE ([orcid.org/0009-0004-9409-0154](https://orcid.org/0009-0004-9409-0154))

# L'environnement RStudio

RStudio fournit un environnement complet pour

- Exécuter des commandes R
- Implémenter du code R (scripts)
- Rédiger des rapports scientifiques qui encapsulent du code R (R markdown)
- Effectuer des analyses
- Afficher les résultats
- Gérer les versions du code
- ...



# Vous pouvez compter sur R

Testez chacune des commandes suivantes dans la console R

```
2 + 3
```

```
4 * 5
```

```
6 / 4
```

```
1,2
```

```
1.2
```

```
9^2
```

```
123456789*987654321
```

```
1:10
```

```
8:-9
```

```
127:231
```

```
1:8 + 5
```

```
1:8 * 8:1
```

```
1:8 * 1:2
```

```
(1:10)^2
```

```
1:10^2
```

Interprétez les résultats

- Quel est le séparateur de décimales ?
- A quoi correspond l'opérateur `:` ?
- A quoi correspond le nombre entre crochets qui précède une ligne de résultat ?
- Comment s'appliquent les opérations entre deux vecteurs de même longueur ?
- Comment s'appliquent les opérations entre deux vecteurs de longueurs différentes ?
- Comment s'affichent les longs vecteurs de nombres ?
- Quel est l'ordre de priorité entre `:` et `^` ?

# Les variables R

Testez les commandes suivantes, et observez l'évolution du panneau "Environment"

```
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 écrit un nom de variable, R imprime sa valeur

b = 2           ## Créer une variable nommée b et lui assigner une valeur
b == a          ## Test d'égalité de deux variables

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

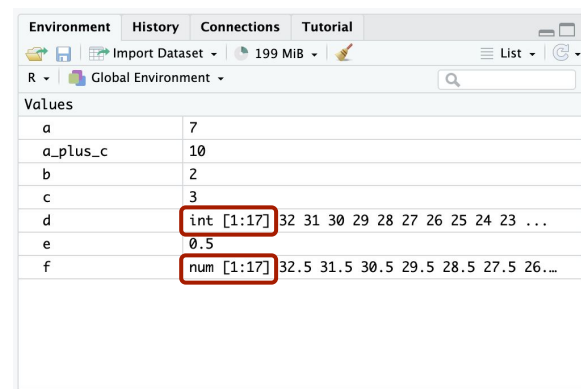
a <- 7          ## Changer la valeur de a
print(a_plus_c) ## Mince alors, le contenu de a_plus_c n'est pas mis à jour

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

d <- 32:16      ## Assigner un vecteur de plusieurs valeurs à une variable
e <- 0.5        ## Assigner un vecteur de plusieurs valeurs à une variable
f <- d + e       ## Effectuer un calcul avec deux vecteurs
print(f)        ## Afficher le résultat
```

Interprétez le résultat

- Quelle est la différence entre `=` et `<-` ?
- Quelle est la différence entre `=` et `==` ?
- Pourquoi faut-il recalculer `a_plus_c` si la valeur de `a` ou de `c` change ?
- Pour les variables contenant un vecteur à plusieurs valeurs, à quoi correspondent les indications qui précèdent les valeurs dans la fenêtre Environnement (encadrées ci-dessous) ?



Environment	History	Connections	Tutorial
R	Global Environment	199 MiB	
Values			
a	7		
a_plus_c	10		
b	2		
c	3		
d	int [1:17]	32 31 30 29 28 27 26 25 24 23 ...	
e	0.5		
f	num [1:17]	32.5 31.5 30.5 29.5 28.5 27.5 26.5 ...	

# Cas d'étude : données protéomiques d'une étude immunologique

Canale, F. P. et al. Proteomics of immune cells from liver tumors reveals immunotherapy targets. *Cell Genomics* 3, 100331 (2023).

[doi.org/10.1016/j.xgen.2023.100331](https://doi.org/10.1016/j.xgen.2023.100331)

- Tableaux de données et autre matériel supplémentaire sur le site Web de Cell  
[www.cell.com/cell-genomics/fulltext/S2666-979X\(23\)00099-X](https://www.cell.com/cell-genomics/fulltext/S2666-979X(23)00099-X)
- Données brutes (pas pour ce cours) :  
[www.ebi.ac.uk/pride/archive/projects/PXD040957](https://www.ebi.ac.uk/pride/archive/projects/PXD040957)

Nous utiliserons le tableau de données protéomiques

- Couverture protéomique : 8192 protéines
- Liver** : 35 échantillons de foie (non-cancéreux)
- HCC** : 34 échantillons de carcinome hépatocellulaire (HCC)
- Dont 31 échantillons appariés (pris sur le même patient)

## Objectifs

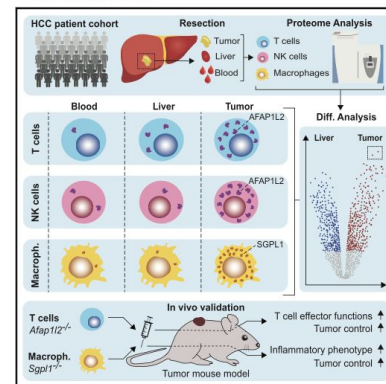
- Détecter les protéines dont l'abondance diffère entre les deux tissus
- Analyser la fonction de ces protéines

## Cell Genomics

Article

### Proteomics of immune cells from liver tumors reveals immunotherapy targets

#### Graphical abstract



#### Authors

Fernando P. Canale, Julia Neumann, Janusz von Renesse, ..., Nuh N. Rahbari, Giorgio Ercolani, Roger Geiger

#### Correspondence

roger.geiger@irb.usl.ch

#### In brief

Canale et al. employed mass spectrometry-based proteomics to analyze immune cells from blood, liver, and tumor tissues of 48 hepatocellular carcinoma patients. They identified significant phenotype alterations in tumor-infiltrating immune cells, including SGPL1 upregulation in macrophages and AFAP1L2 in T cells. Genetic deletion of these genes in murine macrophages and T cells, respectively, enhanced anti-tumor activity, indicating their potential as immunotherapy targets.

#### Highlights

- Proteomes of FACS-purified T cells, NK cells, and macrophages from 48 HCC patients
- Tumor-infiltrating CD8 T cells upregulate AFAP1L2 in response to chronic stimulation
- Deletion of AFAP1L2 in CD8 T cells enhances their anti-tumor activity
- Tumor macrophages upregulate SGPL1, which inhibits inflammatory anti-tumor functions



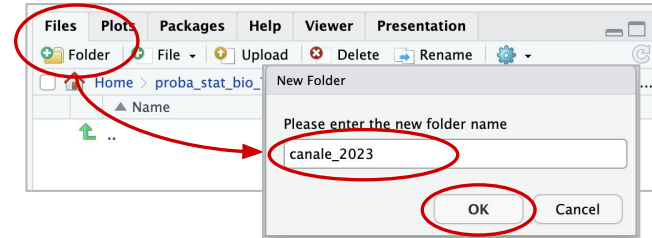
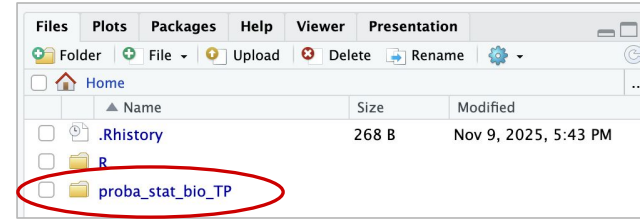
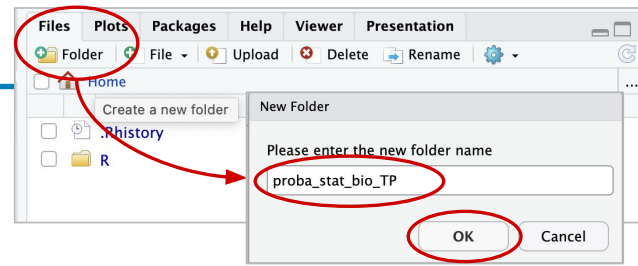
Canale et al., 2023, *Cell Genomics* 3, 100331  
June 14, 2023 © 2023 The Author(s).  
<https://doi.org/10.1016/j.xgen.2023.100331>

# Créer votre dossier de travail

## Deux alternatives

- Via l'interface graphique de RStudio
  - Sur votre ordinateur, créez un dossier `proba_stat_bio_TP` pour ces travaux pratiques
  - Double-cliquez sur le dossier créer pour vous y déplacer
  - Un sous-dossier `canale_2023` pour travailler sur les données Canale (nous créerons d'autres sous-dossiers au fil des TP)
- En 1 commande R
  - Fonction `dir.create()`
  - Argument `file="~/proba_stat_bio_TP/canale_2023"` pour indiquer le chemin complet du dossier qu'on veut créer
  - Argument `recursive = T` : quand le chemin inclut des dossiers imbriqués, créer les dossiers intermédiaires
  - Argument `showWarnings = F` : ne pas afficher les avertissements si le dossier existe déjà

```
dir.create(path="~/proba_stat_bio_TP/canale_2023",  
          recursive = T,  
          showWarnings = F)
```



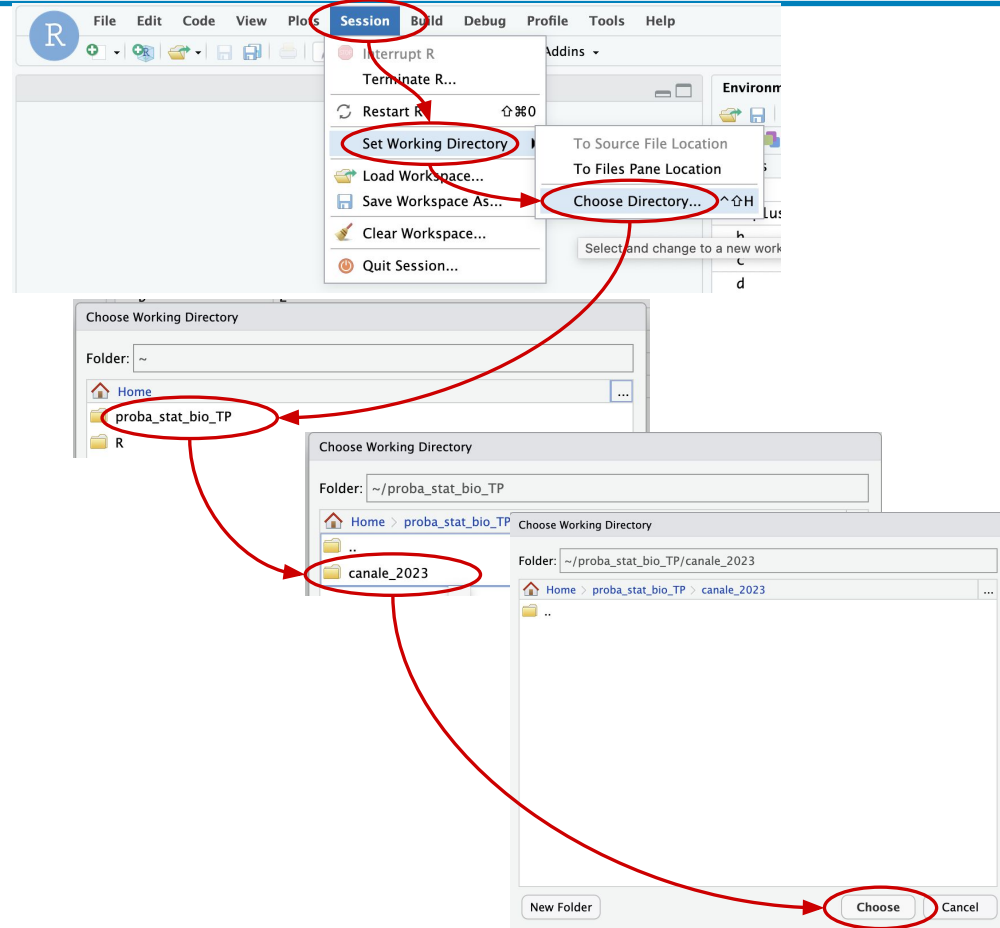
# Vous déplacer dans votre dossier de travail

## Deux alternatives

- Via l'interface graphique de RStudio
  - Dans le menu de RStudio, choisissez la fonction **Session** → **Set Working Directory** → **Choose Directory...**,
- En 1 commande R
  - `setwd("~/proba_stat_bio_TP/canale_2023")`

Le dossier de travail (work directory, **wd**) servira de base pour toutes les analyses liées à ce projet :

- Données téléchargées
- Fichiers de résultats (généralement tabulaires)
- Figures
- ...



# Téléchargement des données à analyser

- Dans le sous-dossier `proba_stat_bio_TP/canale_2023`, créez un sous-sous-dossier `data` pour y déposer les données (utilisez l'interface graphique de RStudio ou bien adaptez la ligne de commande R `dir.create()` précédente)
- Sur un navigateur web, connectez-vous au cours sur Ametice

[ametice.univ-amu.fr/course/view.php?id=140161](https://ametice.univ-amu.fr/course/view.php?id=140161)

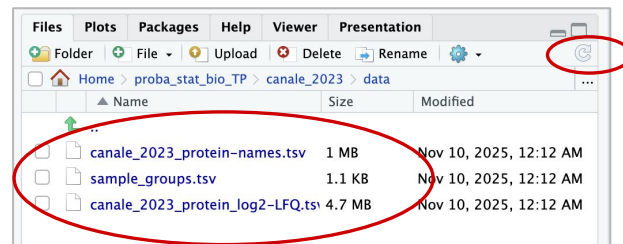
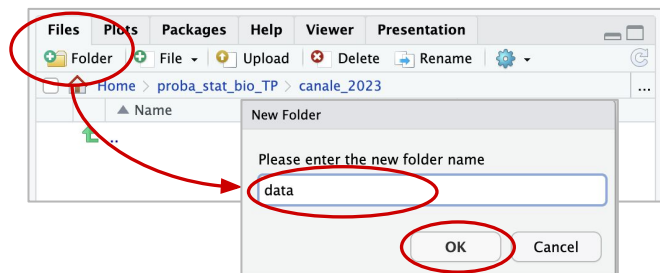
- Dans la section **Jeux de données** d'Ametice, téléchargez les 3 tableaux de données de Canale et déposez-les dans le sous-dossier `data` de votre dossier de travail (en le localisant dans l'interface graphique de votre système opérateur, Windows, Mac OS ou Linux).



- Dans le panneau **Files**, double-cliquez sur le dossier `data` pour vérifier son contenu.
- Vérifiez que les 3 fichiers se trouvent au bon endroit
- Si les jeux de données ne s'affichent pas, vérifiez qu'ils soient dans ce dossier sur votre système opérateur. Cliquez ensuite sur l'icône "refresh".

## Note

- Si vous travaillez sur un serveur RStudio à distance (par exemple sur le cloud ou sur le cluster de l'IFB), vous pouvez utiliser le bouton **Upload** du panneau **Files** pour téléverser les fichiers depuis votre ordinateur vers le serveur.





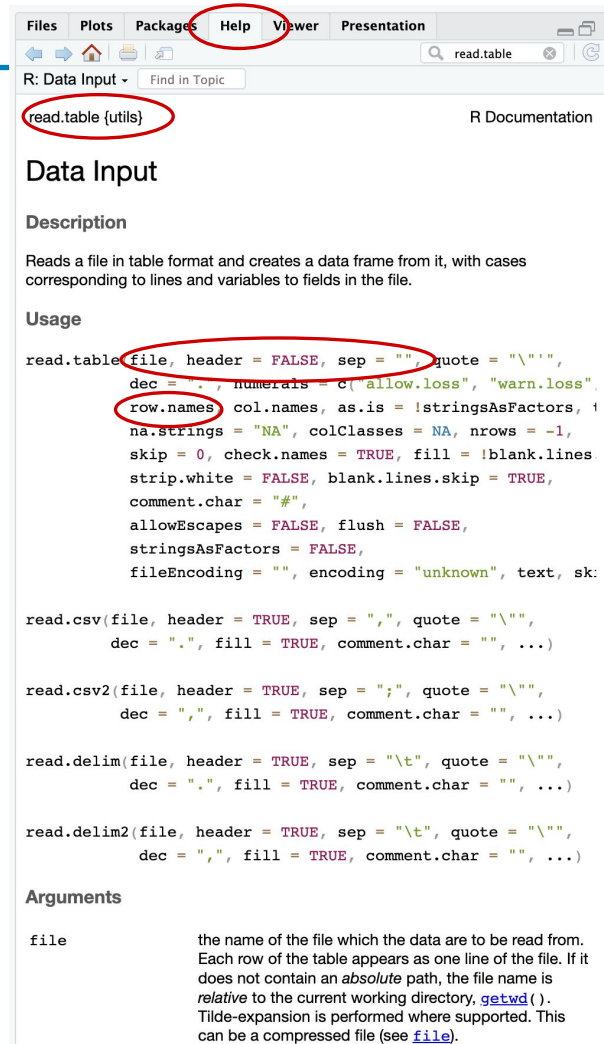
# Chargement des données (dans la mémoire de R)

- Utilisez la fonction R `getwd()` pour vérifier si vous êtes bien dans le dossier de travail  
~/proba\_stat\_bio\_TP/canale\_2023
- Si ce n'est pas le cas, déplacez-vous y en utilisant soit la fonction R `getwd()` soit le menu RStudio  
**Session** → **Set Working Directory** → **Choose Directory...**
- Lisez l'aide de la fonction R `read.table()`  
`help(read.table)`
- Une page d'aide apparaît dans le panneau **Help**. Lisez attentivement l'aide pour les options *file*, *header*, *sep*, *row.names*.

Astuce: le caractère "tabulation" s'écrit \t

- Utilisez la fonction `read.table()` pour charger le fichier `canale_2023_protein_log2-LFQ.tsv` dans une variable nommée `log2_LFQ`
- Utilisez les options *header* et *row.names* pour indiquer que la première ligne du fichier contient les en-têtes de colonnes (identifiants d'échantillons dans notre cas), et que la première colonne contient les noms de lignes (identifiants des protéines dans notre cas).
- Note : **LFQ = label-free quantification**, méthode classique de spectrométrie de masse, qui revient à estimer la quantité d'une molécule sur base de la surface sous le pic

Solution sur la diapo suivante



The screenshot shows the RStudio interface with the **Help** menu highlighted. The **read.table (utils)** help page is displayed in the **Help** pane. The **Data Input** section is visible, along with the **Description** and **Usage** sections. The **Usage** section shows the function signature: `read.table(file, header = FALSE, sep = "\t", quote = "\"", dec = ".", numerals = c("allow.loss", "warn.loss", "no.loss"), row.names = NULL, col.names = NULL, as.is = FALSE, stringsAsFactors = TRUE, na.strings = "NA", colClasses = NULL, nrows = NULL, skip = 0, check.names = TRUE, fill = FALSE, blank.lines.skip = TRUE, strip.white = FALSE, blank.lines.skip = TRUE, comment.char = "#", allowEscapes = FALSE, flush = FALSE, stringsAsFactors = FALSE, fileEncoding = "", encoding = "unknown", text, sk`. The **Arguments** section is also visible, showing the **file** argument: "the name of the file which the data are to be read from. Each row of the table appears as one line of the file. If it does not contain an absolute path, the file name is relative to the current working directory, `getwd()`. Tilde-expansion is performed where supported. This can be a compressed file (see `file`).

# Commande R pour charger les données

Nous utilisons la fonction `read.delim()` pour charger un tableau de nombres délimités par un caractère particulier (le séparateur).

- File: le chemin du fichier à partir du dossier courant (working directory)
- Sep : nous disposons d'un fichier tsv, les colonnes sont donc séparées par des tabulations (symbolisées par la chaîne de caractères `\t`)
- La première ligne du fichier de données contient les en-têtes des colonnes (dans notre cas, elles indiquent l'identifiant de chaque échantillon)
- La première colonne du fichier de données contient les noms de lignes (dans notre cas, elles indiquent l'identifiant de la protéine)

Note: pour la lisibilité, les arguments d'une commande R peuvent s'écrire sur plusieurs lignes.

```
## (re)set your working directory
setwd("~/proba_stat_bio_TP/canale_2023")
getwd()      ## Check that you are in the correct directory
list.files()  ## Check the files present in this directory
list.files("data")  ## Check the files in the subdirectory data

## Display the manual of the function read.table()
help(read.table)
## Don't forget to read if in the Help panel

## Load the data file
log2_LFQ = read.delim(
  file = "data/canale_2023_protein_log2-LFQ.tsv",
  sep="\t",
  header=TRUE,
  row.names=1)
```

# Affichage d'un tableau de données

Utilisez les commandes R `print()`, `head()` et `tail()` pour afficher le contenu de la variable `log2_LFQ`

Notez que pour des données trop volumineuses, la commande `print()` n'affiche que le début des résultats et imprime ensuite un avertissement: `[ reached 'max' /getOption("max.print") -- omitted 8168 rows ]`

```
Console Terminal Background Jobs
R - R 4.5.2 - ~/proba_stat_bio_TP/canale_2023/
> print(log2_LFQ)
```

	HCC_1	HCC_10	HCC_101	HCC_102	HCC_103	HCC_104	HCC_109
Q13286	NA	24.66709	23.99559	25.94235	26.09573	24.79417	25.99378
A0A024QZW4	NA	NA	NA	22.89160	23.39780	22.32989	NA
A0A024R161	22.64654	25.27728	NA	24.32065	25.58683	NA	24.32382
A0A024R1R8	NA	NA	25.91020	24.21156	23.14141	25.47482	25.32529
A0A024R216	NA	NA	26.98533	25.13101	26.01144	24.68437	25.09716
A0A024R442	28.48212	28.96747	29.78475	30.56265	30.04388	29.37368	30.63910
A0A024R4E5	31.07833	31.54347	31.38134	30.96829	31.65301	31.80597	31.16117
A0A087WZA9	26.61681	27.32732	26.24621	27.90484	27.41691	26.55157	26.47849
A0A024R4M0	33.12663	32.93243	32.10818	31.34599	31.90038	31.74763	31.52818
A0A024R571	30.01651	29.47146	28.47591	28.77115	28.53731	29.41597	29.81851
A0A024R5I4	26.47517	24.36760	26.06846	25.64724	25.46248	NA	NA
A0A0A0MRU5	24.17790	26.20680	27.40698	25.48806	25.62628	28.97221	28.66492
A0A024R6I7	NA	24.78292	21.80947	28.51431	29.58796	30.49047	NA
A0A024RA52	28.95283	28.42770	30.19484	29.73925	30.15726	29.58628	29.71875
	HCC_11	HCC_110	HCC_111	HCC_12	HCC_14	HCC_15	HCC_16
Q13286	NA	24.64741	24.57045	24.16906	NA	NA	NA
A0A024QZW4	NA	23.62990	NA	23.33661	NA	NA	NA
A0A024R161	NA	NA	25.54262	22.89253	NA	25.36533	NA
A0A024R1R8	26.16907	25.97189	26.19336	27.38837	NA	NA	28.90176
A0A024R216	29.69949	NA	27.43237	28.74381	28.57272	26.97978	NA
A0A024R442	26.77693	30.05168	29.59918	28.55644	28.63986	28.61758	29.92719
A0A024R4E5	30.52916	32.02921	31.25490	31.22992	28.65236	28.31038	31.36646
A0A087WZA9	29.15750	27.01776	26.86242	26.33118	29.07526	NA	26.00894
A0A024R4M0	30.22642	31.74435	31.79048	33.28967	31.52122	31.47358	31.93789
A0A024R571	26.94168	28.26977	28.30003	29.33569	27.85007	26.33024	30.45454

# View()

La fonction R `View()` permet d'afficher le contenu d'un tableau de façon interactive : vous pouvez

- Naviguer dans les lignes et les colonnes
- Trier les lignes selon les valeurs d'une colonne, en cliquant sur l'en-tête
- Appliquer des filtres sur une ou plusieurs colonnes pour sélectionner un sous-ensemble des lignes

Notez que certaines cellules de ce tableau d'abondance des protéines contiennent le symbole `NA`, qui signale une donnée manquante. Nous verrons ultérieurement comment traiter ces valeurs.

En-têtes de colonnes  
(identifiants des échantillons)

	HCC_1	HCC_10	HCC_101	HCC_102	HCC_103	HCC_104	HCC_109	HCC_11	HCC_110	HCC_111
Q13286	NA	24.66709	23.99559	25.94235	26.09573	24.79417	25.99378	NA	24.64741	24.57
A0A024QZW4	NA	NA	NA	22.89160	23.39780	22.32989	NA	NA	23.62990	
A0A024R161	22.64654	25.27728	NA	24.32065	25.58683		24.32382	NA	NA	25.54
A0A024R1R8	NA	NA	25.91020	24.21156	23.14141	25.47482	25.32529	26.16907	25.97189	26.19
A0A024R216	NA	NA	26.98533	25.13101	26.01144	24.68437	25.09716	29.69949	NA	27.43
A0A024R442	28.48212	28.96747	29.78475	30.56265	30.04388	29.37368	30.63910	26.77693	30.05168	29.59
A0A024R4E5	31.07833	31.54347	31.38134	30.96829	31.65301	31.80597	31.16117	30.52916	32.02921	31.25
A0A087WZA9	26.61681	27.32732	26.24621	27.90484	27.41691	26.55157	26.47849	29.15750	27.01776	26.86
A0A024R4M0	33.12663	32.93243	32.10818	31.34599	31.90038	31.74763	31.52818	30.22642	31.74435	31.79
A0A024R571	30.01651	29.47146	28.47591	28.77115	28.53731	29.41597	29.81851	26.94168	28.26977	28.30
A0A024R5I4	26.47517	24.36760	26.06846	25.64724	25.46248	NA	NA	28.47619	26.94537	25.59
A0A0A0MRU5	24.17790	26.20680	27.40698	25.48806	25.62628	28.97221	28.66492	23.60975	25.84816	25.65
A0A024R6I7	NA	24.78292	21.80947	28.51431	29.58796	30.49047	NA	NA	NA	
A0A024RA52	28.95283	28.42770	30.19484	29.73925	30.15726	29.58628	29.71875	27.53170	29.67570	29.70
A0A024RAC6	22.71127	21.81503	23.35649	22.12818	23.11321	24.83779	NA	NA	23.79848	23.99
E9PRM7	26.12530	28.11748	24.46482	27.16710	26.87608	26.44126	25.54537	28.00715	26.13975	27.25
F2Z2M1	23.02063	NA	NA	NA	NA	NA	NA	NA	NA	
P46379-3	27.22453	27.20546	28.34456	28.39750	27.28065	27.44079	27.97989	27.59548	29.35195	27.69
A0A067XG54	25.48867	26.15238	26.31432	25.35844	25.02276	24.98971	NA	27.63305	26.52245	27.29
A0A075B6I0	24.08521	NA	24.09402	NA	NA	NA	NA	24.06612	NA	
A0A075B6K5	NA	NA	NA	23.84833	24.17448	24.42073	NA	NA	NA	
A0A075B6P5	27.50605	27.30765	26.42222	28.62167	29.41615	29.13097	29.08583	27.24627	26.85270	27.26
A0A075B6R2	25.46862	24.54292	24.04485	26.78019	27.22426	24.02808	27.20452	NA	26.77894	26.29
A0A0C4DH68	25.44579	23.30841	24.52247	25.54315	26.51991	27.04124	24.46594	26.02568	NA	23.83
A0A075B6S2	27.02570	24.53368	25.35659	26.25832	NA	NA	NA	25.51782	24.78057	23.84
A0A075B6S5	NA	NA	NA	NA	23.50736	NA	23.38995	NA	NA	
A0A075B730	25.11724	24.47537	31.59917	29.34600	29.05038	28.26772	25.26807	23.66633	21.48974	22.67
O00623	23.92515	24.55368	22.31047	23.78357	22.70142	21.47788	25.02039	24.62795	20.70691	24.47

Showing 1 to 28 of 8,182 entries, 69 total columns

Noms de lignes  
(identifiants des protéines)

# Caractéristiques d'un tableau

---

Testez les commandes suivantes et interprétez les résultats

## Dimensions

`ncol(log2_LFQ)`

`nrow(log2_LFQ)`

`dim(log2_LFQ)`

## Nombre de colonnes

## Noms des colonnes et des lignes

`colnames(log2_LFQ)`

`names(log2_LFQ)`

`rownames(log2_LFQ)`

## Résumé rapide des données par colonne

`summary(log2_LFQ)`

`str(log2_LFQ)`

# Affichage de sous-ensembles d'un tableau

On peut sélectionner une cellule d'un tableau

- Par l'indice de la ligne et de la colonne

```
log2_LFQ[31, 12]
```

- Par les noms de la ligne et de la colonne :

```
log2_LFQ["A0A087WU53", "HCC_14"]
```

On peut sélectionner une ou quelques lignes d'un tableau

- Par l'indice de la ligne : `log2_LFQ[31, ]`
- Par le nom de ligne : `log2_LFQ["A0A075B7D0", ]`

Il existe plusieurs façons de sélectionner une ou quelques colonnes d'un tableau.

- Par le nom de colonne : `log2_LFQ[, "HCC_31"]`
- Par le nom de colonne (notation alternative):  
`log2_LFQ$HCC_31`
- Par l'indice de colonne: `log2_LFQ[, 31]`

Comment sélectionner plusieurs colonnes ? Par ex, HCC\_31, HCC\_104 et Liver\_104 ?

```
log2_LFQ[, c("HCC_31", "HCC_104", "Liver_104")]
```

Sélection de colonnes par leur indice

```
log2_LFQ[, 31]
```

Comment sélectionner les colonnes 2 et 3 ?

```
log2_LFQ[, c(3, 2)]
```

et les 5 premières colonnes ?

```
log2_LFQ[, 1:5]
```

et les colonnes 10 à 16 sur les lignes 2000 à 2020 ?

```
log2_LFQ[2000:2020, 10:16]
```

# Chargement des métadonnées

Les métadonnées sont des données qui décrivent un jeu de données.

Pour le tableau du protéome, nous disposons de deux fichiers de métadonnées

- Description des échantillons : fichier `sample_groups.tsv`, indiquant l'identifiant de chaque échantillon, le type de cellules (groupe) et le numéro de patient
- Description des protéines : fichier `sample_groups.tsv` fournissant les identifiants, les noms de protéines et les noms de gène de chaque protéine du tableau de protéome.

## Exercice

- Utilisez la fonction R `read.delim()` pour charger ces deux fichiers dans les variables R nommées respectivement `sample_groups` et `protein_names`.
- Vérifiez les dimensions des deux `data.frames`
  - `sample_groups` :
  - `protein_names` :
- Utilisez la fonction `View()` pour vérifier le contenu des deux variables.
  - Assurez-vous que l'en-tête a bien été pris en compte et que les contenus des colonnes correspond aux en-têtes.
  - Vérifiez que le nombre de lignes de métadonnées correspond aux nombres attendus.

## Solution

```
## Set the working directory
setwd("~/proba_stat_bio_TP/canale_2023")

## Load the metadata table "sample_groups.tsv"
sample_groups <- read.delim(
  file = "data/sample_groups.tsv",
  sep = "\t", header = TRUE, row.names = 1)
View(sample_groups)

## Load the protein names from a separate table
protein_names <- read.delim(
  file = "data/canale_2023_protein-names.tsv",
  sep = "\t", header = TRUE, row.names = 1)
View(protein_names)
```

# R markdown

## Avec R Markdown

- meilleure traçabilité
- meilleure reproductibilité
- possibilité de visualiser
  - code
  - résultats textuels
  - Résultats graphiques
  - Vos ajouts textuels (présentation de la problématique, objectifs, interprétation des résultats, ...)



## Ressources

- [bookdown.org/yihui/rmarkdown/](http://bookdown.org/yihui/rmarkdown/)
- [rmarkdown.rstudio.com/](http://rmarkdown.rstudio.com/)

Pendant les travaux pratiques, nous utiliserons R markdown pour mettre en oeuvre les analyses statistiques avec R, conserver une trace du code implémenté, afficher les résultats, et ajouter des commentaires d'interprétation.

The screenshot displays the RStudio interface with two windows. The left window, titled 'chunks.Rmd', shows R code chunks for a plot. The right window, titled 'RStudio: Preview HTML', shows the rendered HTML output.

**Left Window (chunks.Rmd):**

```
1 R Code Chunks
2 =====
3
4 With R Markdown, you can insert R code
5 chunks including plots:
6
7 ```{r qplot, fig.width=4, fig.height=3,
8   message=FALSE}
9 # quick summary and plot
10 library(ggplot2)
11 summary(cars)
12 qplot(speed, dist, data=cars) +
13   geom_smooth()
```

**Right Window (RStudio: Preview HTML):**

### R Code Chunks

With R Markdown, you can insert R code chunks including plots:

```
# quick summary and plot
library(ggplot2)
summary(cars)
```

##	speed	dist
##	Min. : 4.0	Min. : 2
##	1st Qu.:12.0	1st Qu.: 26
##	Median :15.0	Median : 36
##	Mean :15.4	Mean : 43
##	3rd Qu.:19.0	3rd Qu.: 56
##	Max. :25.0	Max. :120

```
qplot(speed, dist, data = cars) + geom_smooth()
```



# Cheat sheets

**Base R Cheat Sheet**

**Getting Help**

**Advanced R Cheat Sheet**

Created by: Arienne Colton and Sean Chen

**Environment Basics**

Environment – Data structure (with two components below) that powers lexical scoping

1. Create environment: `env1 <- new.env()`

**Environments**

**Search Path** – mechanism to look up objects, particularly functions.

- Access with: `search()` – lists all parents of the global environment (see Figure 1)
- Access any environment on the search path: `as.environment("package:base")`

**Function Environments**

1. **Enclosing environment** – an environment where the function is created. It determines how function finds value.
2. **Binding environment** – all environments that the function has a binding to. It determines how we find

**R Markdown : : CHEAT SHEET**

**What is R Markdown?**

**Workflow**

1. Open a new Rmd file at File > New File > R Markdown. Use the wizard that opens to propose the file with a template.
2. Write document by editing template.
3. Run document in RStudio viewer, use knit button or `knitr::render()` to knit.
4. Preview (optional) to web server.
5. Examine build log in R Markdown console.
6. Use output file that is saved along side Rmd.

**render**

Use `knitr::render()` to render text at cmd line. Important args:

input: file to render  
output: file to output (can be blank)  
output\_dir: output directory (can be blank)

**Embed code with knitr syntax**

**INLINE CODE**

Insert with `"<code>"`. Results appear as text without code.

**CODE CHUNKS**

Code in Rmd is surrounded with `"<pre>"` and `"</pre>"`. Place chunk options within curly braces, after `"<pre>"`. Insert with `"<pre>"`.

**GLOBAL OPTIONS**

Set with `knitr::opts_chunk$set()`. A global option is `knitr::opts_chunk$set(echo = TRUE)`.

**IMPORTANT CHUNK OPTIONS**

- `cache`: cache results for future runs (default = FALSE)
- `cache.path`: directory to save cached results in (default = ".cache/").
- `child`: file(s) to knit and then include (default = FALSE)
- `collapse`: collapse output into single block (default = FALSE)
- `comment`: prefix for each line of results (default = "#")

**render options**

- `depend`: chunk dependencies for caching (default = FALSE)
- `echo`: display code in output document (default = TRUE)
- `engine`: code language used in chunk (default = "R")
- `error`: display error messages in doc (TRUE) or suppress when errors occur (FALSE) (default = TRUE)
- `file`: file to write chunk (default = TRUE)

**render options**

- `fig.align`: "left", "right", or "center" (default = "center")
- `fig.cap`: figure caption as character string (default = NULL)
- `fig.height`, `fig.width`: Dimensions of plots in inches
- `highlight`: highlight source code (default = TRUE)
- `html`: file to write for display (default = FALSE)
- `message`: display code messages in document (default = TRUE)
- `results` (default = "markup")
  - `asis`: plain-text results
  - `html`: HTML results
  - `knitr`: knitr results
  - `latex`: LaTeX results
  - `raw`: raw code
  - `source`: source code
  - `text`: text results
  - `word`: word results
- `warning`: display code warnings in document (default = TRUE)

**RStudio IDE : : CHEAT SHEET**

**Documents and Apps** **Write Code** **R Support** **Pro Features**



[www.r-project.org/](http://www.r-project.org/)

[rstudio.com/](http://rstudio.com/)

[www.r-bloggers.com/](http://www.r-bloggers.com/)

[thinkr.fr/](http://thinkr.fr/)

## Cheat sheets for R

- [stat33a.berkeley.edu/spring-2025/cheatsheets/base-r-cheatsheet.pdf](http://stat33a.berkeley.edu/spring-2025/cheatsheets/base-r-cheatsheet.pdf)

## Cheat sheets for Rmarkdown

- <https://posit.co/blog/the-r-markdown-cheat-sheet/>
- [rstudio.com/resources/cheatsheets/](http://rstudio.com/resources/cheatsheets/)