



Généralités sur le langage R

Alison PATOU

Patou.alison@gmail.com



Programme

- Présentation du langage R
- Les types de données
- Importer – Exporter des données
- Les techniques pour tracer des courbes et créer des graphiques



A savoir

- Tout est sur mon GitHub :
<https://github.com/apatou>
- (L'essentiel à retenir du cours, les dataset, exercices, ...)
- Merci de m'envoyer à chaque fin de séance vos TPs : patou.alison@gmail.com
- 1 projet final à rendre à la fin.

1

Introduction

Introduction

Présentation du logiciel R et de ses fonctionnalités

Avantages et Inconvénients

Accès au site de téléchargement de l'outil et installation

Utilisation de la console

1. Présentation du logiciel R et de ses fonctionnalités

- **Le langage S** a été créé dans les années 70 pour « programmer avec des données ». Il a été développé chez *Bell Laboratories* par une équipe de chercheurs menée par John M. Chambers.
 - Dès la fin des années 1980 et pendant près de vingt ans, le S a principalement été popularisé par une **mise en œuvre commerciale nommée S-PLUS**.
 - Inspirés à la fois par le S et par Scheme (un dérivé du Lisp), Ross Ihaka et Robert Gentleman proposent un langage pour l'analyse de données et les graphiques qu'ils nomment R. Le langage a été intégré, en 1997, au projet GNU, faisant de **R un logiciel libre**.
-

Présentation du logiciel R et de ses fonctionnalités



- ✓ Le langage R est un **langage interprété**. Cela signifie que l'on peut écrire seulement une ligne de code, la valider et en voir le résultat. Il n'y a donc pas besoin d'une étape préalable de compilation du code, celui-ci est interprété à la volée.

| | Avantage | Inconvénient |
|--------------------|-------------|--------------|
| Langage interprété | portabilité | Lenteur |
| Langage compilé | rapidité | Portabilité |

- ✓ Pour écrire du code en R on peut donc simplement lancer ce que l'on appelle la console et taper du code.

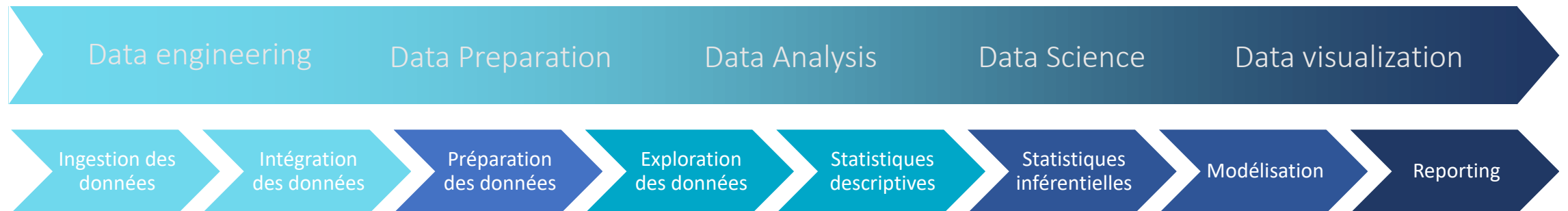
1. Présentation du logiciel R et de ses fonctionnalités



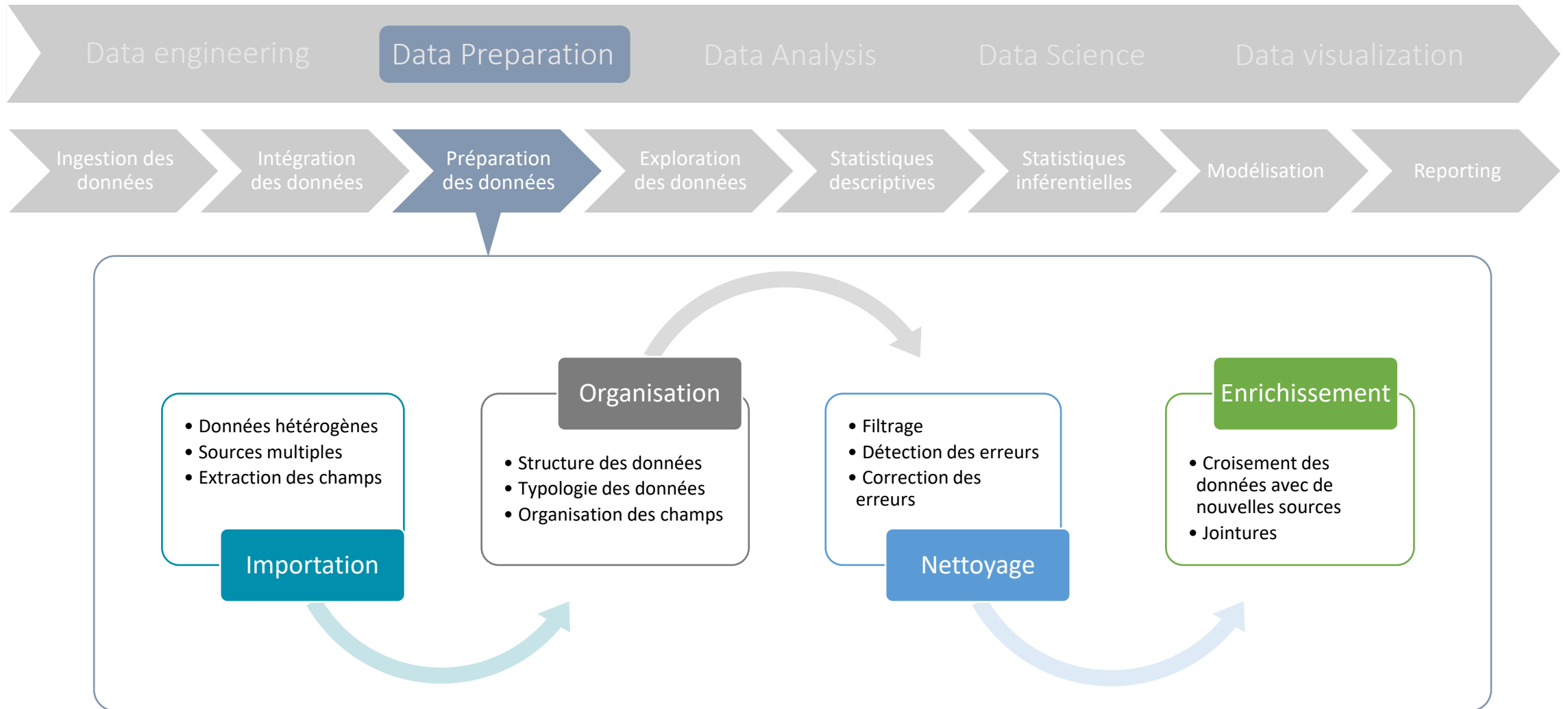
- ✓ L'objet de base est un **vecteur de données**.
- ✓ « vrai » langage de programmation
 - types de données
 - branchements conditionnels
 - boucles
 - ...
- ✓ Mode d'exécution : transmettre à R le fichier script « .r »
- ✓ Il est extensible (quasiment) à l'infini *via* le système des packages.

1. Présentation du logiciel r et de ses fonctionnalités

R est un langage particulièrement utilisé en **Data Science** mais il permet d'intervenir à plusieurs étapes du processus de traitement de la donnée :



1. Présentation du logiciel R et de ses fonctionnalités



The diagram illustrates the data science process flow, organized into two rows of chevron-shaped boxes pointing from left to right.

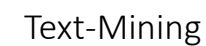
The top row contains five boxes:

- Data engineering
- Data Preparation
- Data Analysis
- Data Science
- Data visualization

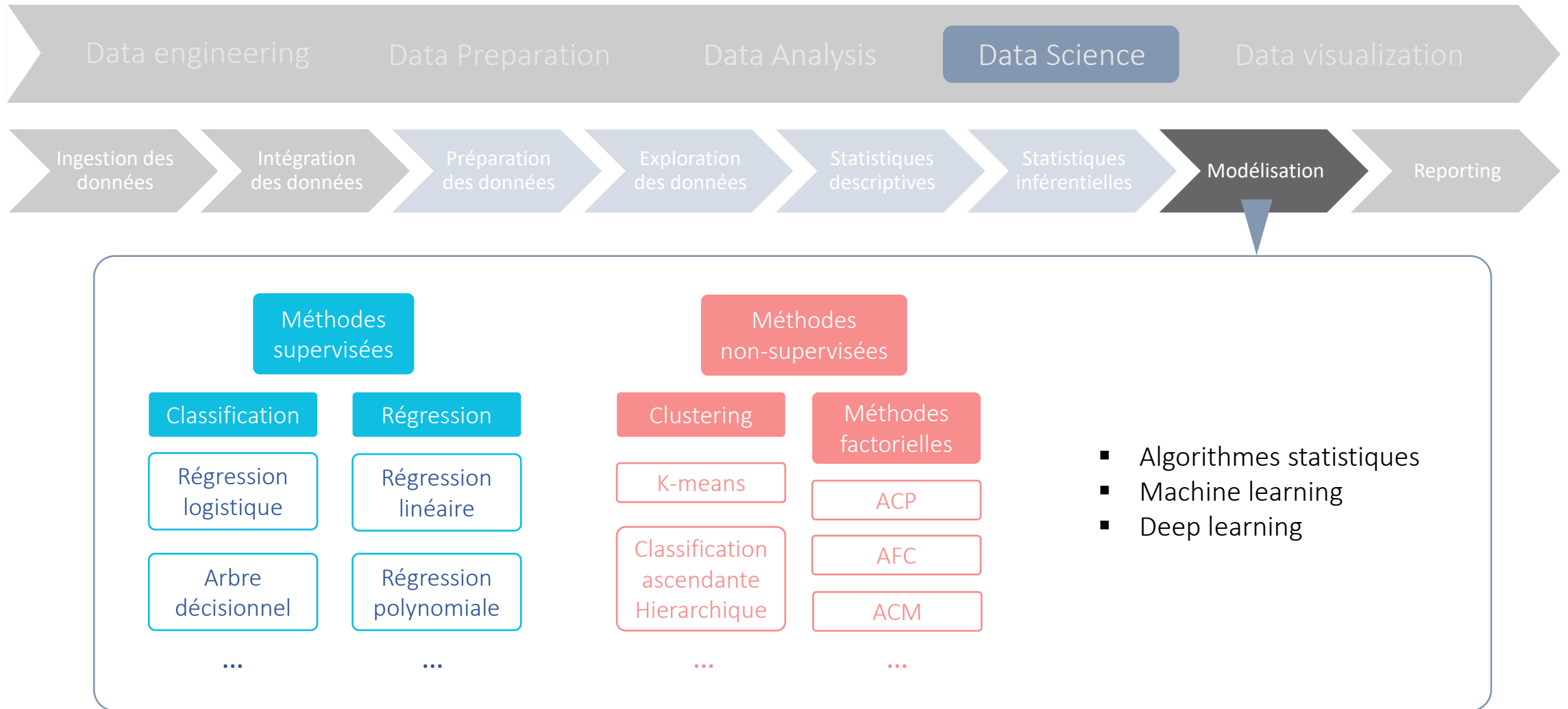
The bottom row contains eight boxes:

- Ingestion des données
- Intégration des données
- Préparation des données
- Exploration des données
- Statistiques descriptives
- Statistiques inférentielles
- Modélisation
- Reporting

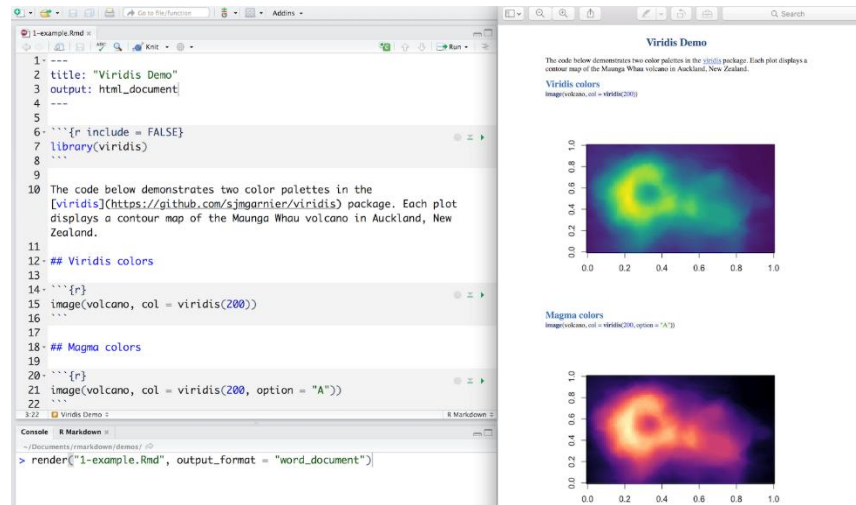
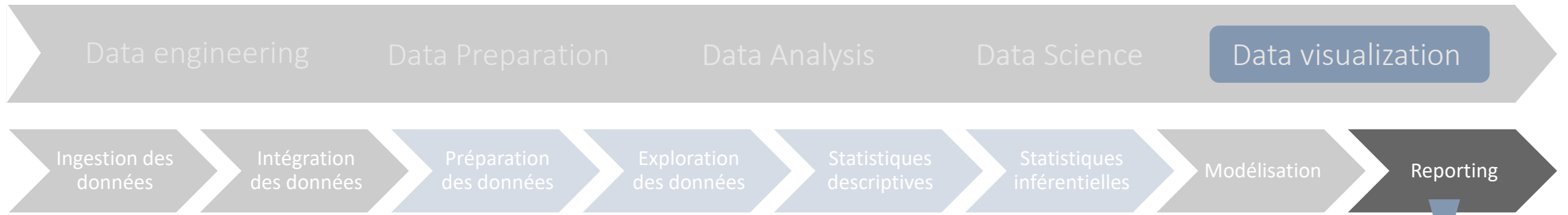
Vertical lines connect the top row to the bottom row, indicating a sequential flow. The boxes for 'Data Analysis', 'Data Science', 'Exploration des données', 'Statistiques descriptives', and 'Statistiques inférentielles' are highlighted in a darker blue color, while the others are in a lighter blue or grey color.



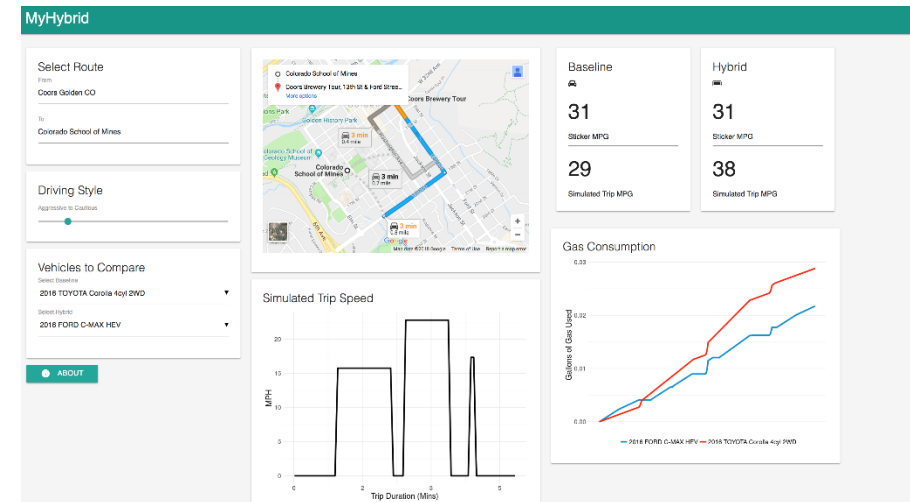
1. Présentation du logiciel R et de ses fonctionnalités



1. Présentation du logiciel R et de ses fonctionnalités



Confection de rapports (R Markdown)



Confection de Dashboards (R Shiny)

1. Présentation du logiciel R et de ses fonctionnalités

R est généralement à travers un **environnement de développement intégré** (IDE) dont :



1. Présentation du logiciel r et de ses fonctionnalités

R est généralement à travers un **environnement de développement intégré** (IDE) dont :



1. Présentation du logiciel R et de ses fonctionnalités



| | Open Source Edition | RStudio Desktop Pro |
|----------|--|---|
| Overview | <ul style="list-style-type: none">• Access RStudio locally• Syntax highlighting, code completion, and smart indentation• Execute R code directly from the source editor• Quickly jump to function definitions• Easily manage multiple working directories using projects• Integrated R help and documentation• Interactive debugger to diagnose and fix errors quickly• Extensive package development tools | <p>All of the features of open source; plus:</p> <ul style="list-style-type: none">• A commercial license for organizations not able to use AGPL software• Access to priority support• RStudio Professional Drivers |
| Support | Community forums only | <ul style="list-style-type: none">• Priority Email Support• 8 hour response during business hours (ET) |
| License | AGPL v3 | RStudio License Agreement |
| Pricing | Free | \$995/year |
| | DOWNLOAD RSTUDIO DESKTOP | DOWNLOAD FREE RSTUDIO DESKTOP PRO TRIAL |

1. Présentation du logiciel R et de ses fonctionnalités



| | Open Source Edition | RStudio Server Pro |
|---------------|--|---|
| Overview | <ul style="list-style-type: none">• Access via a web browser• Move computation closer to the data• Scale compute and RAM centrally | <p>All of the features of open source; plus:</p> <ul style="list-style-type: none">• Administrative Tools• Enhanced Security and Authentication• Metrics and Monitoring• Advanced Resource Management• Use RStudio, Python, and Jupyter |
| Documentation | Getting Started with RStudio Server | RStudio Server Professional Admin Guide |
| Support | Community forums only | <ul style="list-style-type: none">• Priority Email Support• 8 hour response during business hours (ET) |
| License | AGPL v3 | RStudio License Agreement |
| Pricing | Free | <p>Starting at \$4,975 / 5 named users per year</p> <p>Academic and Small Business discounts available</p> |
| | DOWNLOAD SERVER | DOWNLOAD FREE RSTUDIO SERVER PRO TRAIL |

2. Avantages et inconvénients

- R est un langage puissant **pour les applications mathématiques et statistiques** puisque développé dans ce but
- Il possède de nombreux avantages mais aussi quelques inconvénients



Avantages

| |
|---|
| langage basé sur la notion de vecteur (= simplification les calculs mathématiques) |
| Langage interprété : portabilité du code |
| Programmes courts ... |
| communauté importante en ligne ... |
| pas de typage ni de déclaration obligatoire des variables |
| Extensible <i>via</i> les packages |



Inconvénients

| |
|--|
| les variables passées comme arguments sont dupliquées en mémoire |
| Non compilable: performances pouvant être limitées |
| ... Temps de calcul parfois long |
| ... mais pas de support |
| Gestion du parallélisme |
| Gestion de la concurrence |

4. Accès au site et téléchargement


1. Télécharger R : <https://www.r-project.org/>



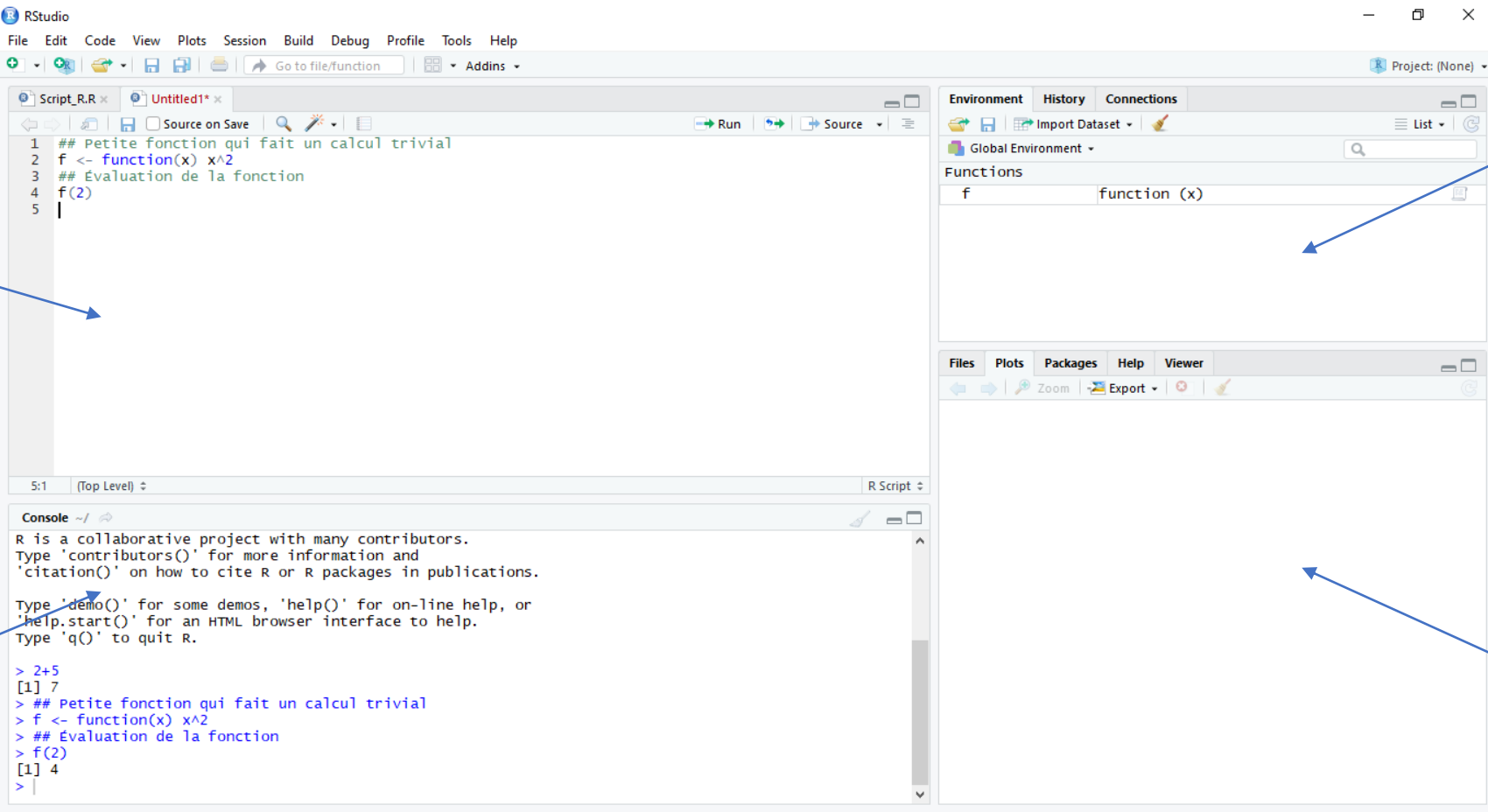
News

- **R version 3.4.4 (Someone to Lean On)** has been released on 2018-03-15.
- **useR! 2018** (July 10 - 13 in Brisbane) is open for registration at <https://user2018.r-project.org>
- **The R Journal Volume 9/2** is available.
- **R version 3.3.3 (Another Canoe)** has been released on Monday 2017-03-06.
- **useR! 2017** took place July 4 - 7 in Brussels <https://user2017.brussels>
- The **R Logo** is available for download in high-resolution PNG or SVG formats.

2. Télécharger Rstudio : <https://www.rstudio.com/products/RStudio/>

| | |
|---|-----------------------|
| Support | Community forums only |
| License | AGPL v3 |
| Pricing | Free |
| <div><div>DOWNLOAD RSTUDIO DESKTOP</div></div> | |

Environnement de base



The screenshot shows the RStudio IDE interface. The main editor window on the left contains a script with the following R code:

```
1 ## Petite fonction qui fait un calcul trivial
2 f <- function(x) x^2
3 ## Evaluation de la fonction
4 f(2)
5 |
```

The console window at the bottom left shows the output of the script:

```
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.

> 2+5
[1] 7
> ## Petite fonction qui fait un calcul trivial
> f <- function(x) x^2
> ## Evaluation de la fonction
> f(2)
[1] 4
> |
```

The Environment pane on the right shows the Global Environment with a list of functions:

| Object | Class |
|--------|--------------|
| f | function (x) |


Annotations with blue arrows point to the following components:

- Scripts**: Points to the main editor window.
- Console**: Points to the console window.
- Vue des fonctions et données**: Points to the Environment pane.
- Aide**: Points to the Help pane.
- Graphiques**: Points to the Plots pane.
- Fichiers**: Points to the Files pane.
- Packages**: Points to the Packages pane.

IDE RStudio : environnement de développement gratuit, libre et multiplateforme pour R

Utilisation de la console

Calcul simple directement dans la ligne de commande : instruction -> résultat

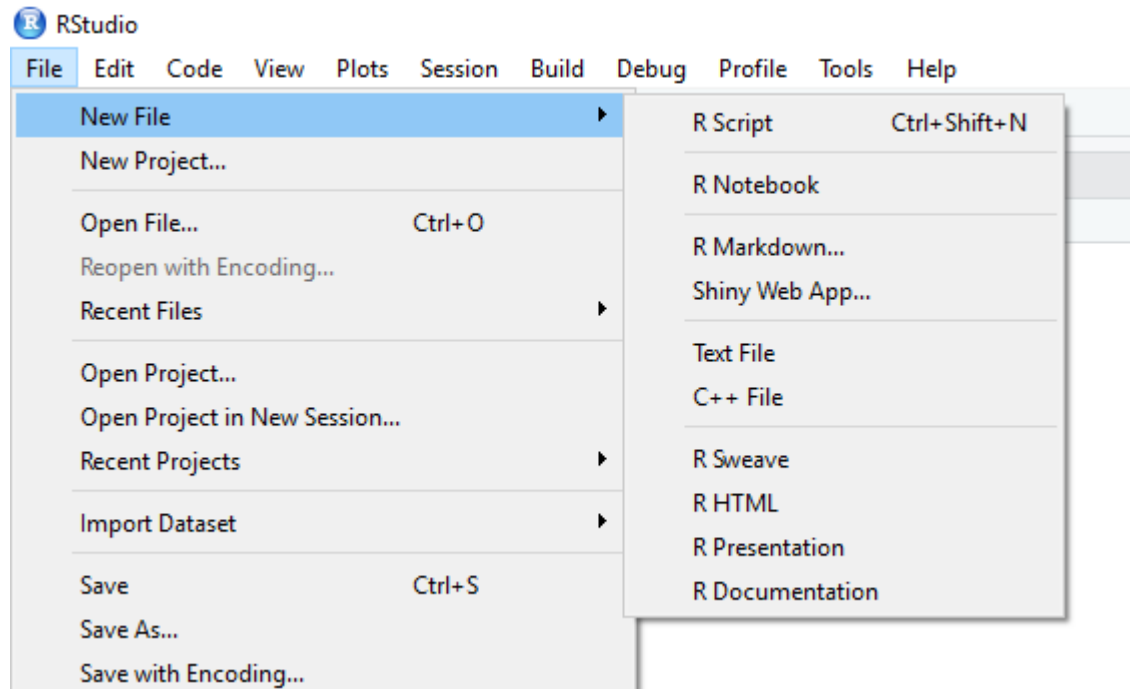
```
Console ~/   
  
R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.  
  
R is a collaborative project with many contributors.  
Type 'contributors()' for more information 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.  
  
> 2+5  
[1] 7
```



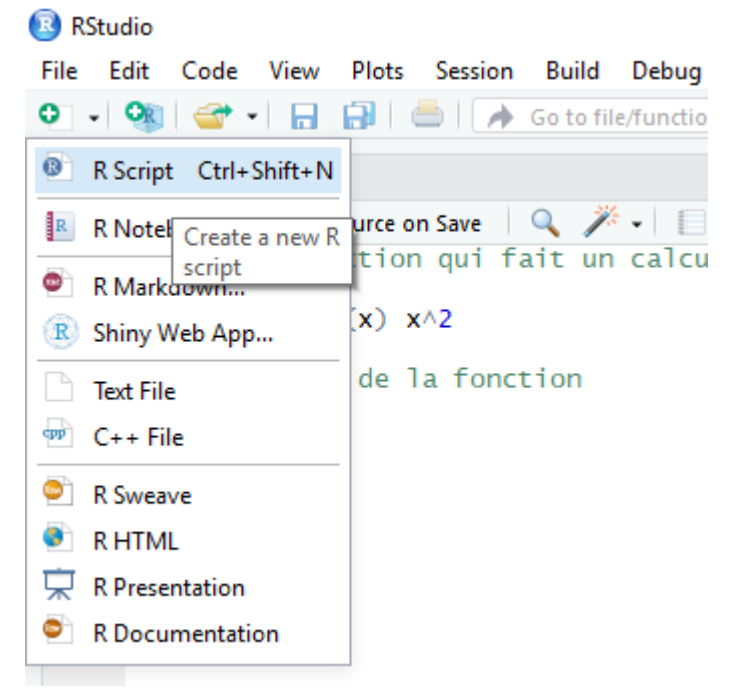
Le résultat ne sera pas enregistré en mémoire

Création d'un script

Solution 1:



Solution 2:



Création d'un script

Ecriture dans le script (calcul, fonction, algorithme, ...)

```
1 ## Petite fonction qui fait un calcul trivial
2 f <- function(x) x^2
3 ## Évaluation de la fonction
4 f(2)
```



Affichage du résultat dans la console

```
Console ~/
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

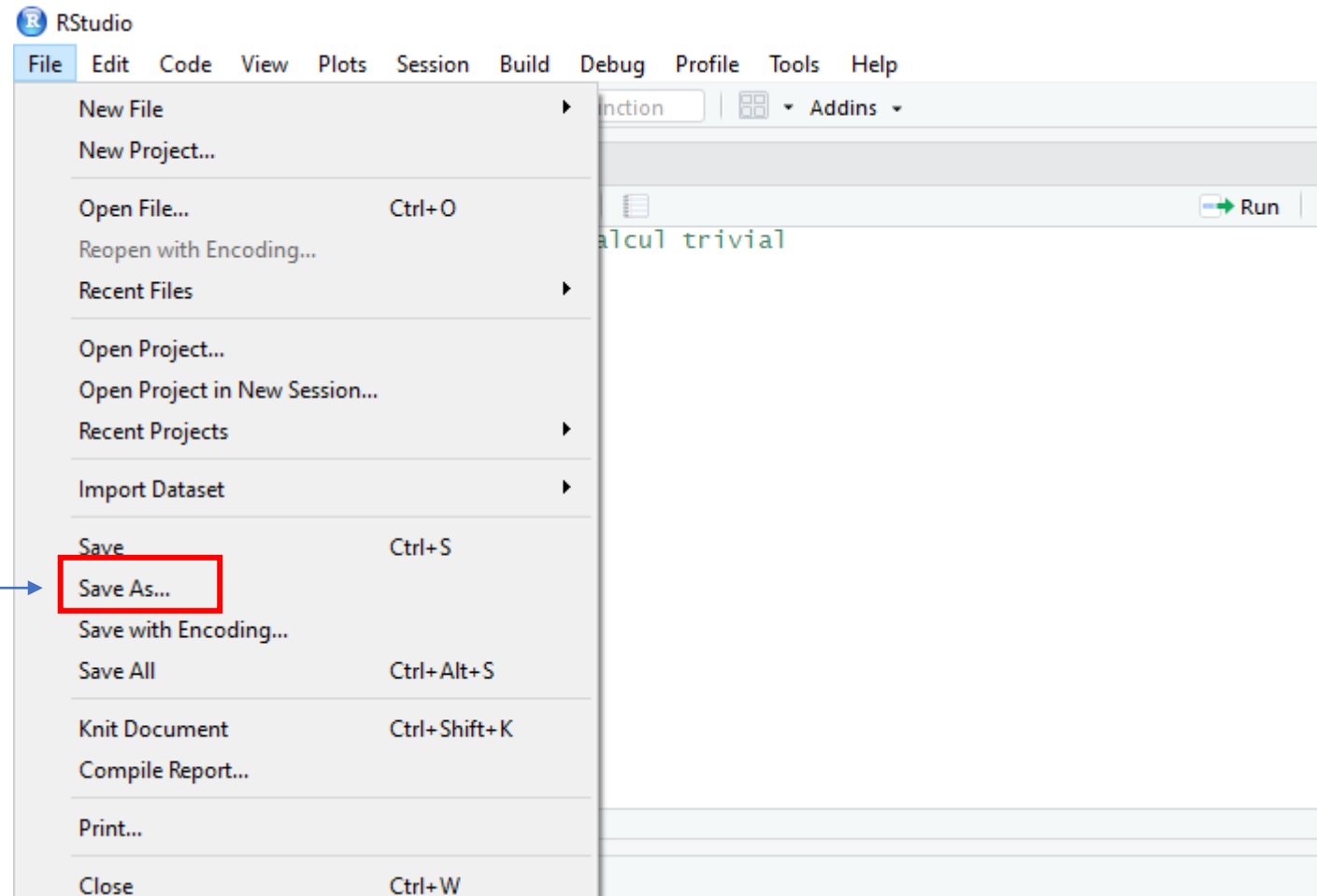
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

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.

> f <- function(x) x^2
> ## Evaluation de la fonction
> f(2)
[1] 4
```

Création d'un script

Enregistrement via
Save/Save As...



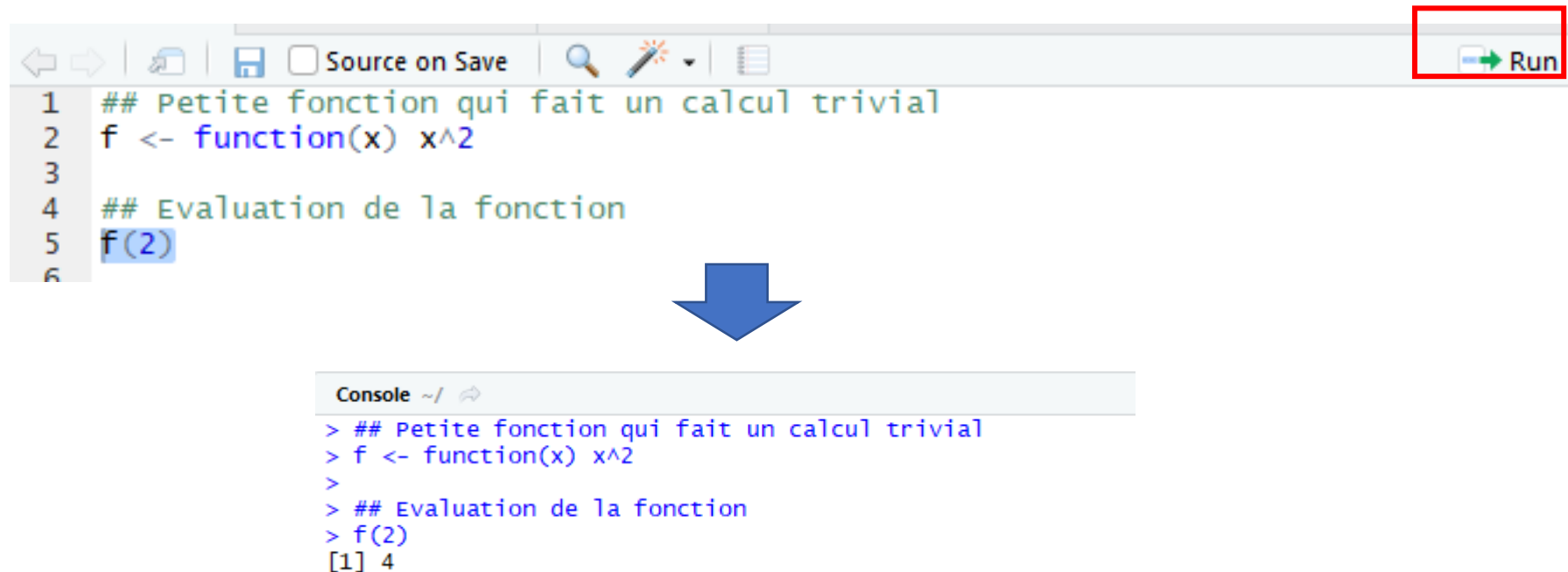
Création d'un script

Enregistrer votre script R avec un .R

Ex : vous souhaitez appeler votre fichier exemple_1
vous l'enregistrer directement en « exemple_1.R »

Exécution du code

Pour exécuter votre commande, il suffit de sélectionner ce que vous souhaitez exécuter puis **appuyer sur 'Run'** ou avec les touches **CTRL+ENTER**



The image shows a code editor interface. At the top, there is a toolbar with various icons. A red rectangle highlights the 'Run' button, which is represented by a green arrow pointing right and the word 'Run'. Below the toolbar, the code editor contains the following R code:

```
1 ## Petite fonction qui fait un calcul trivial
2 f <- function(x) x^2
3
4 ## Evaluation de la fonction
5 f(2)
6
```

A large blue arrow points from the code editor to a console window below it. The console window has a title bar that says 'Console ~/ ↶'. It displays the output of the code execution:

```
> ## Petite fonction qui fait un calcul trivial
> f <- function(x) x^2
>
> ## Evaluation de la fonction
> f(2)
[1] 4
```

Le Répertoire sous r

Gestion des chemins :

```
> getwd()  
[1] "C:/Users/alison.patou/Documents"
```

La fonction **getwd()** affiche la localisation du répertoire de travail sous la forme d'un chemin absolu.

Modification des chemins :

```
> setwd("C:/Users/alison.patou/Pictures"); getwd()  
[1] "C:/Users/alison.patou/Pictures"
```

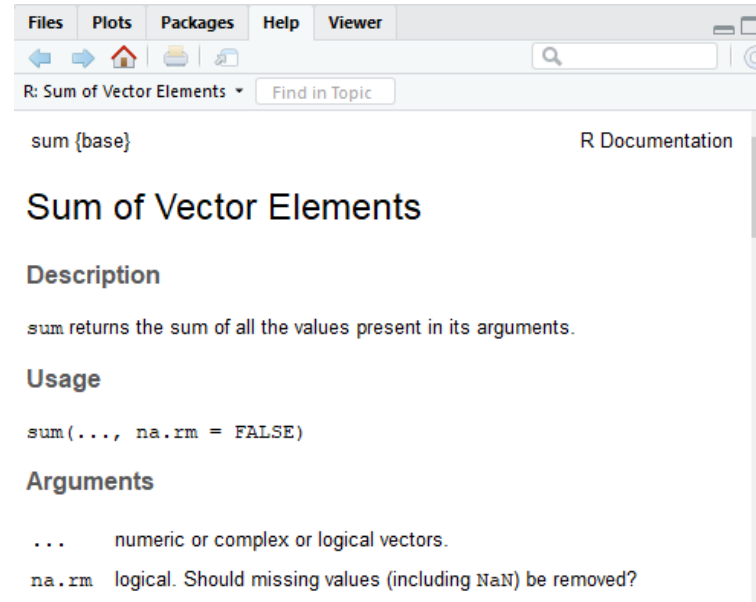
On peut modifier la localisation du répertoire de travail avec la fonction **setwd()**

Aide

Lorsque l'on souhaite accéder à l'aide intégrée de R sur une fonction :

```
> help("sum")
```

```
> ?sum
```

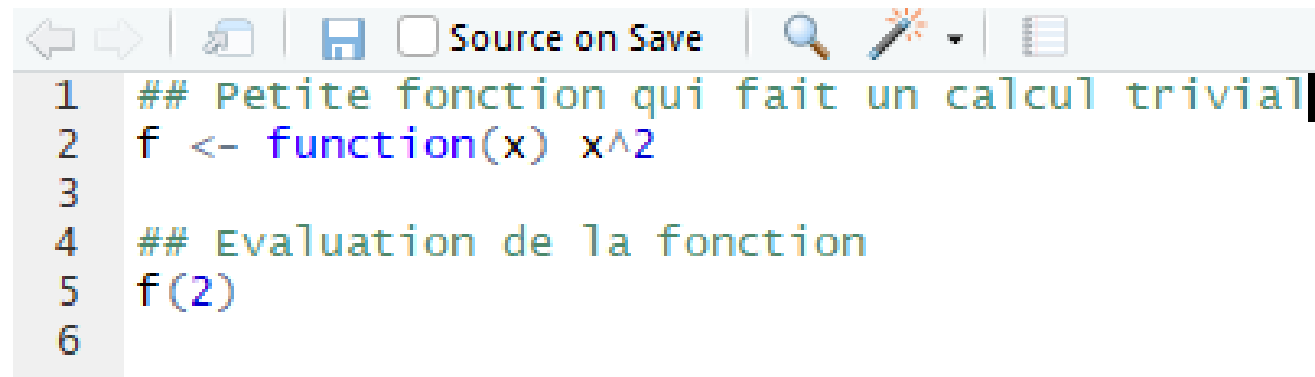


Ces deux commandes affichent une page (en anglais) décrivant la fonction, ses paramètres, son résultat, le tout accompagné de diverses notes, références et exemples.

Commentaires

Les commentaires sont essentiels dans un script : il s'agit d'un texte libre, commençant par un ou plusieurs # et ignoré par R.

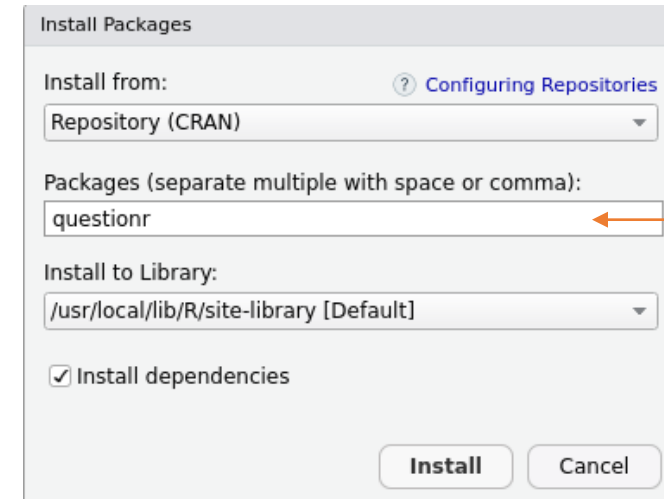
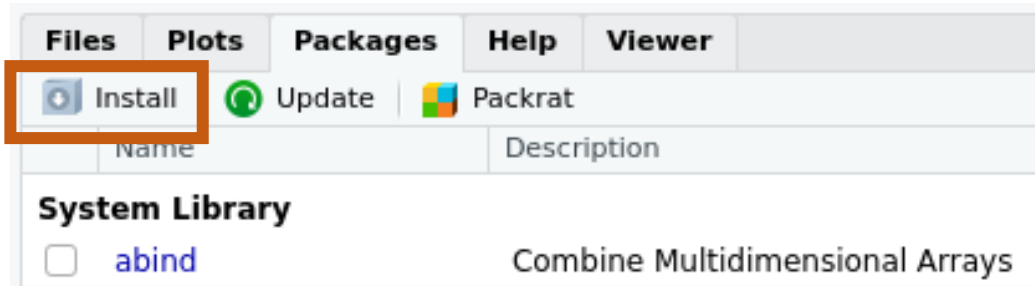
Ils décrivent les étapes d'un script, permettent de documenter, d'explicitier ce que l'on fait afin de faciliter la compréhension du code :



```
1 ## Petite fonction qui fait un calcul trivial
2 f <- function(x) x^2
3
4 ## Evaluation de la fonction
5 f(2)
6
```

Les packages

- R étant un logiciel libre, il bénéficie d'un développement communautaire riche et dynamique.
- L'installation de base de R permet de faire énormément de choses, mais le langage dispose en plus d'un système d'extensions permettant **d'ajouter facilement de nouvelles fonctionnalités grâce à des packages**.
- Installation d'un package :



Le nom du package
(auto-complétion
possible)

- Une fois le package installé, il faut le “charger” avant de pouvoir utiliser les fonctions qu’il propose. Ceci se fait avec la fonction **library(nomdupackage)**

2.

Les types de données

Les types de données sous R

Il existe 3 grandes classes de données :

- Vecteurs simples: Ce sont des simples listes ordonnées de valeurs simples
- Données composites (listes, data frames, etc.)
- Données "spéciales" pour programmeurs

Il existe 6 types de valeurs simples:

"logical", "integer", "double", "complex",
"character", "raw"

✓ **Typeof(AAA)** : renvoie le type de AAA

Les objets

Qu'est-ce qu'une variable sous R ?

- ✓ Permettent de stocker une valeur (un chiffre, du texte, ...) ou un objet (une fonction, une matrice, un vecteur, ...)
- ✓ L'affectation d'une valeur à une variable se fait à l'aide de l'opérateur <- ou =
- ✓ On retrouve les opérateurs mathématiques

```
# Assigner la valeur 4 à my_var
my_var <- 4

# Afficher la valeur de la variable my_var
my_var

# Assigner la valeur 6 à la variable my_muffins
my_muffins <- 6

# Assigner la valeur 4 à la variable my_cupcakes
my_cupcakes <- 4

# La variable my_cakes contiendra le nombre total de gâteaux que vous avez
my_cakes <- my_muffins + my_cupcakes

# Affiche 10
my_cakes
```

Les objets

Tout dans le langage R est un objet : les variables contenant des données, les fonctions, les opérateurs, ...

Les objets possèdent au minimum :

- une longueur : `length()`
- un mode : `mode()`

```
# Création vecteur
> v <- c(4,2,1,3)

# Afficher le mode de l'objet v
> mode(v)
[1] « numeric »

# Afficher la longueur de l'objet v
> Length(v)
[1] 4
```

Les objets

**c(a=,b=,...)
vector()**

```
# Construction de deux vecteurs
> Vector_A <- c(1,2,3)
> Vector_B <- c(4,5,6)

# Faire la somme de ces deux vecteurs
> total_vector = Vector_A + Vector_B

# Afficher cette somme
> total_vector
[1] 5 7 9
```

```
# Initialisation de vecteurs :
> vector(« character », 5)
[1] «» «» «» «» «» «»

> vector(« logical », 3)
[1] FALSE FALSE FALSE

> vector(« numeric », 10)
[1] 0 0 0 0 0 0 0 0 0 0
```

Array : tableau à n dimensions de valeurs de même type

array()

- ✓ Nommer les lignes et les colonnes **rownames**(matrice) et **colnames**(matrice) ou **dimnames** = list(vecteur1, vecteur2)
- ✓ **Cbind** / **Rbind** permet de combiner plusieurs matrices ou vecteurs (soit ajout de colonnes, soit ajout de lignes).
- ✓ **ColSums** / **RowSums** pour sommer sur les colonnes ou sur les lignes
- ✓ Sélection avec [1,2], possibilité avec des plages de données [1:2,6], possibilité de seulement des lignes ou des colonnes [,1] ou [2,]
- ✓ Possibilité de multiplier toutes les valeurs par un chiffre ou par une autre matrice

```
# Création d'une matrice
> matrix(1:9, nrow=3)
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

Les objets

Array : tableau à n dimensions de valeurs de même type

`array()`

**Matrice : collection d'éléments de même types de données
= tableau à 2 dimensions**

`matrix()`

- ✓ Nommer les lignes et les colonnes `rownames(matrice)` et `colnames(matrice)` ou `dimnames = list(vecteur1, vecteur2)`
- ✓ Fonction `rowSums` pour sommer les lignes
- ✓ `Cbind` / `Rbind` permet de combiner plusieurs matrices ou vecteurs (soit ajout de colonnes, soit ajout de lignes).
- ✓ `ColSums` / `RowSums` pour sommer sur les colonnes ou sur les lignes
- ✓ Sélection avec `[1,2]`, possibilité avec des plages de données `[1:2,6]`, possibilité de seulement des lignes ou des colonnes `[,1]` ou `[2,]`
- ✓ Possibilité de multiplier toutes les valeurs par un chiffre ou par une autre matrice

```
# Création d'une matrice
> matrix(1:9, nrow=3)
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

> Le facteur contient la liste + indication des valeurs distinctes

- ✓ **Levels** permet de changer le nom des éléments d'un facteur. Attention, l'ordre d'un facteur, s'il n'est pas spécifié est l'ordre alphabétique.
- ✓ **Summary** compte le nombre d'éléments d'un facteur. Attention, à faire sur un facteur et non sur un vecteur
- ✓ Lors de la création d'un facteur, il est possible de l'ordonner avec les mots clés **ordered = TRUE** et **levels =** liste ordonnée

```
# Construction de deux vecteurs
> temperature_vector <- c(«High», «Low», «High», «Low»,
«Medium»)
> Factor_temperature_vector <- factor(temperature_vector,
order=TRUE, levels=c(«High», «Low», «Medium»))
> Factor_temperature_vector
[1] High Low High Low Medium
Levels : Low<Medium<High
```

list()

Liste : Liste de données qui peut contenir D'autres types de données complexes

- ✓ La fonction List permet de créer une liste
- ✓ Fonction Names pour renommer (ou list(name1=objet1, name2=objet2,...))
- ✓ Afficher un élément entre double crochets. On peut récupérer un élément d'un objet listé en rajoutant la sélection
- ✓ Pour rajouter un élément, utiliser c(list, élément)
- ✓ Pour sélectionner un élément : List[[1]]

```
# Regardons les éléments de la liste

> Actor <- list(«Jack Nicholson», «Shelley Duvall»,
«Danny Lloyd», «Barry Nelson», «Scatman Crothers»)
[[1]]
[1] «Jack Nicholson»

[[2]]
[1] «Shelley Duvall»

[[3]]
[1] «Danny Lloyd»

[[4]]
[1] «Barry Nelson»

[[5]]
[1] «Scatman Crothers»
```

DataFrame : Tableau composé de type de données différentes

`data.frame()`

- ✓ Fonctions `head()` et `tail()` pour récupérer les premières et les dernières lignes d'un data frame.
- ✓ `str()` permet de voir la structure d'un data frame
- ✓ Fonction `data.frame()` avec une liste de vecteurs de la même taille Chaque vecteur sera une colonne du data frame/
- ✓ On peut utiliser le raccourci `$` pour appeler toute une colonne

ex : `Clients$Noms` affiche la colonne Noms du data frame Clients

- ✓ La fonction `Subset` permet de ne récupérer qu'une partie des données, par rapport à un filtre
- ✓ On peut sélectionner une colonne en tapant **`Clients[numéro ou nom de colonne]`**

```
# Structure de la table Clients
> str(Clients)
'data.frame': 312 obs. of 4 variables
 $ Noms      : char  DUPONT BESSE CARBONE CESAR ...
 $ Prenoms   : num   Marc Antoine Luc Marie ...
 $ Age       : num   12 35 38 75 25 43 61 8 96 55 ...
 $ Dep       : num   69 38 13 75 92 18 37 74 58 21 ...
```


3.

Importer – Exporter des
données

Lecture des données

- ✓ Les fonctions **read.table()** et **read.csv()** permettent de lire et importer des fichiers *.txt* et *.csv*. Le résultat va se trouver dans une structure de type "data frame".

```
# Le fichier data.txt est lu et stocké dans un nouveau objet R nommé Database
Database <- read.table("data.txt", header = TRUE)

# Fichier de type CSV depuis un serveur web (ce fichier contient des stats de
# google webmaster tools pour edutechwiki ...)
Database_webmaster <- read.csv("http://tecfa.unige.ch/guides/R/data/edutechwiki-
fr-gw-oct-6-2014.csv", header = TRUE, sep= ",")
```

- ✓ Les fonctions **fread()** du package `data.table` permet de lire les fichiers avec un délimiteur régulier, similaire à `read.table()` la fonction est bien plus rapide.

Visualiser les données

Visualiser les tableaux que vous avez importé ou créé :

- Dans RStudio, cliquer sur le variable (par ex. "Database" ci-dessus) dans le panneau Environment
- Utilisez : **summary**(DB), **dim**(Database), etc.
- Pour afficher une colonne ou d'autres détails utilisez la syntaxe "\$" ou "[..]"

Export des données

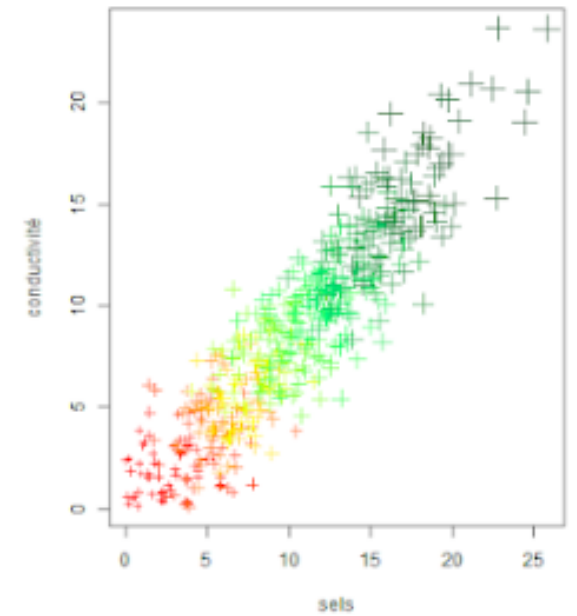
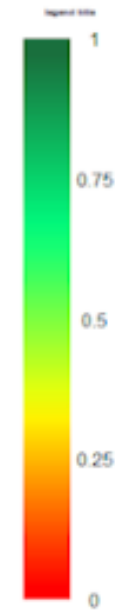
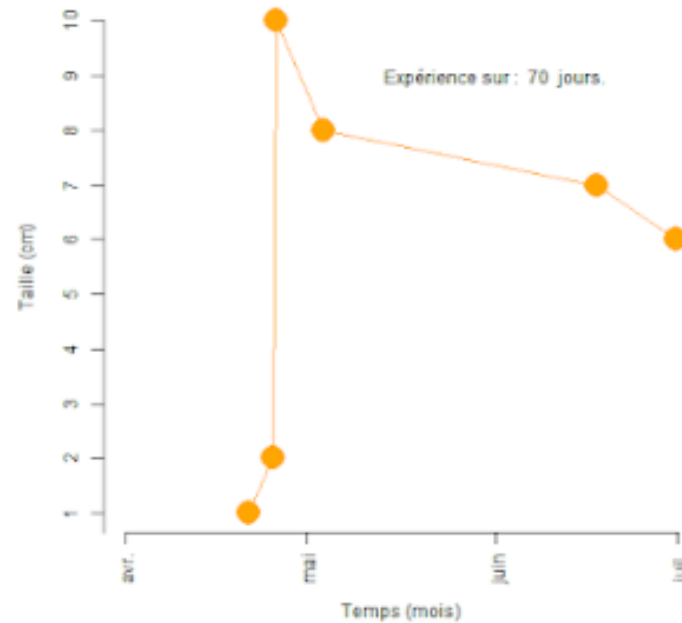
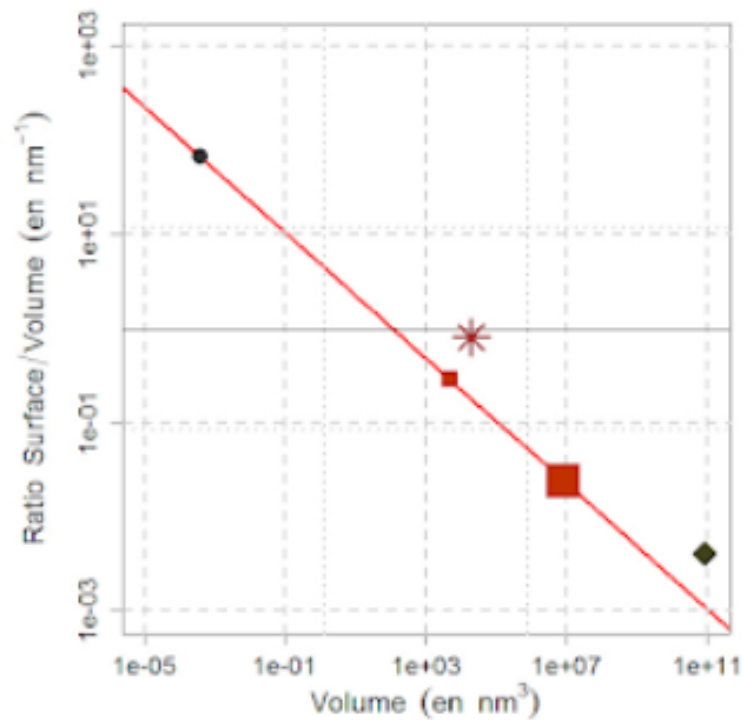
- ✓ Les fonctions **write_csv**, **write_delim**, **write_tsv** permettent d'enregistrer un *data frame* ou un tibble dans un fichier au format texte délimité

```
# Le fichier est enregistré sous nomFichier.csv et le numéro des lignes n'est pas conservé  
write_csv(nomFichier, file="nomFichier.csv", row.names = FALSE)
```

4.

Les graphiques

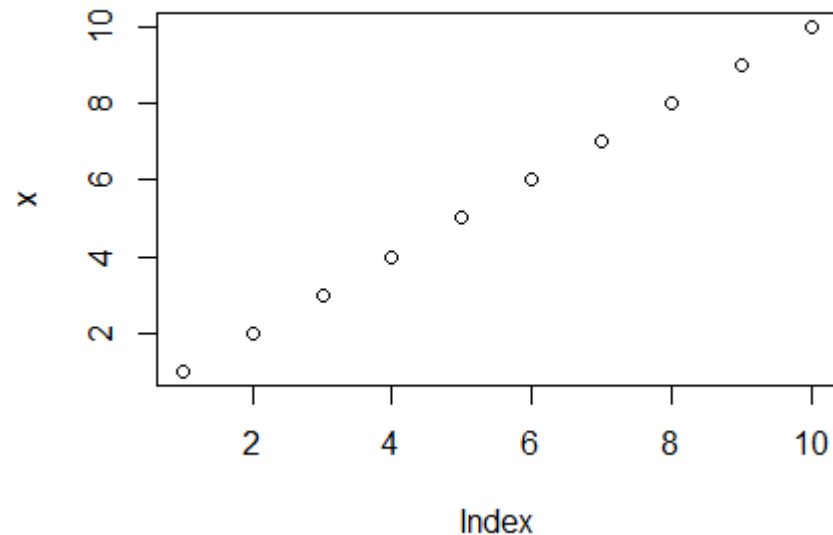
1. Courbes et nuages de point



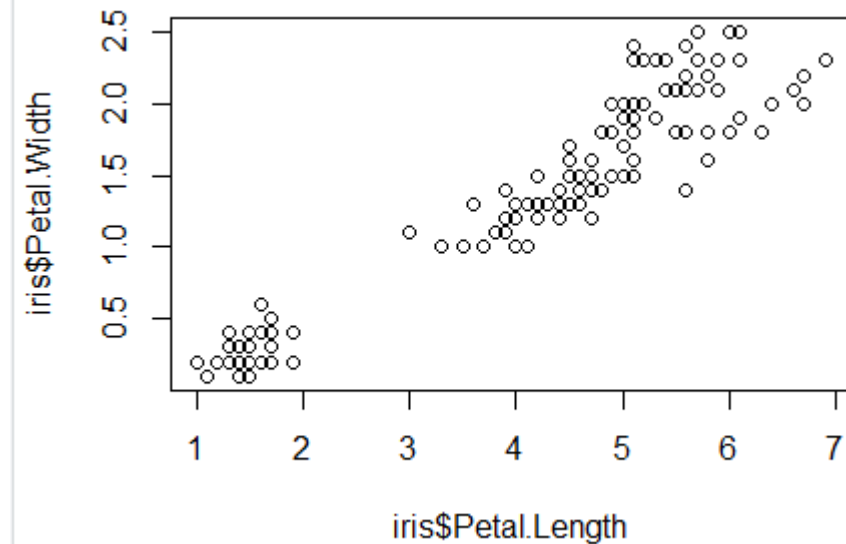
1. Scatter plot

Tracer un nuage de point (*scatter plot*) illustrant toutes les valeurs de x dans l'ordre et à intervalles réguliers

```
# Tracer un graphique d'un vecteur  
x = c(1:10)  
plot(x)
```



```
# Tracer un graphique du dataset iris  
  
Plot(iris$Petal.Length, iris$Petal.Width)
```



1. Scatter plot

Les options de mise en forme des graphiques

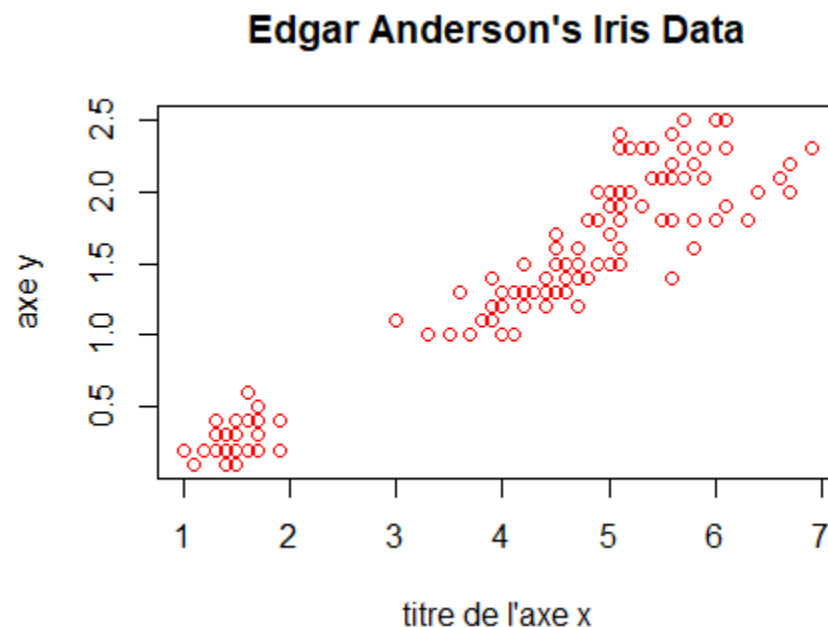
```
# main pour le titre
plot(iris$Petal.Length, iris$Petal.Width, main="Edgar Anderson's Iris Data")

# col pour changer la couleur
plot(iris$Petal.Length, iris$Petal.Width,col="red")

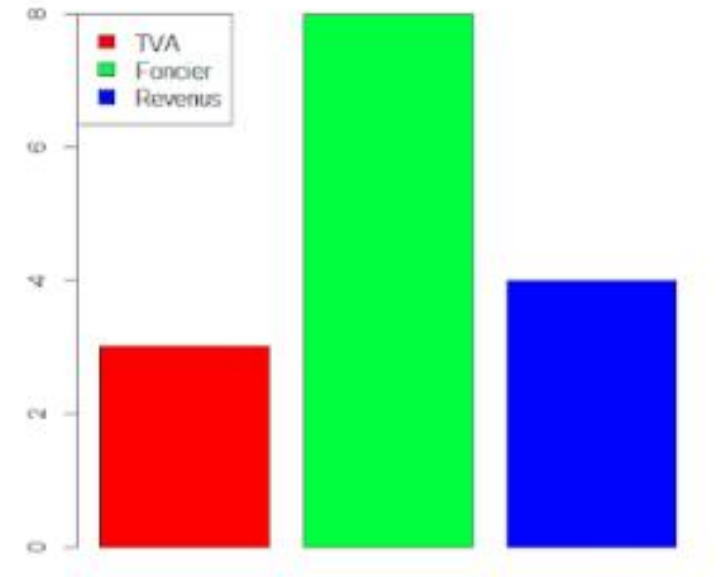
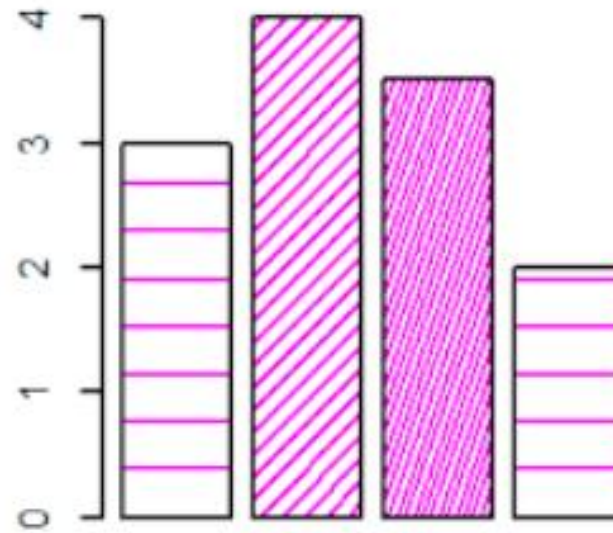
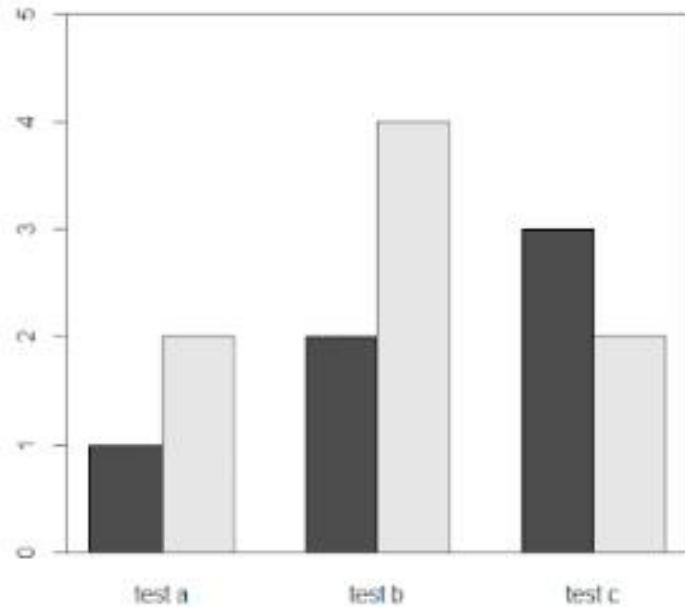
# type pour changer le type de point ('l' pour ligne, 'p' pour point, 'b' point-ligne, ...)
plot(iris$Petal.Length, iris$Petal.Width,type="b")

# xlab, ylab pour renommer les axes
plot(iris$Petal.Length, iris$Petal.Width,xlab = "titre de l'axe x",ylab = "axe y")

# xlim et ylim pour paramétrer les limites
plot(iris$Petal.Length, iris$Petal.Width,xlim=c(0,10),ylim=c(2,5))
```



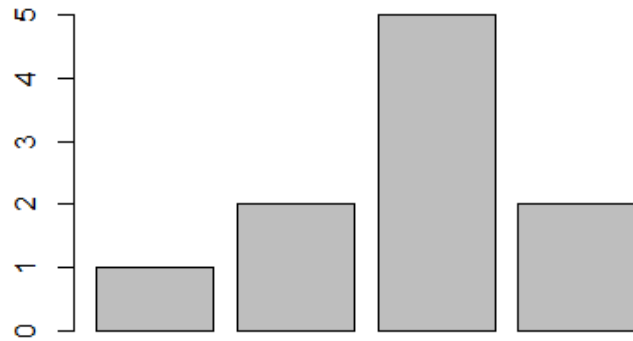
2. Diagrammes en barre



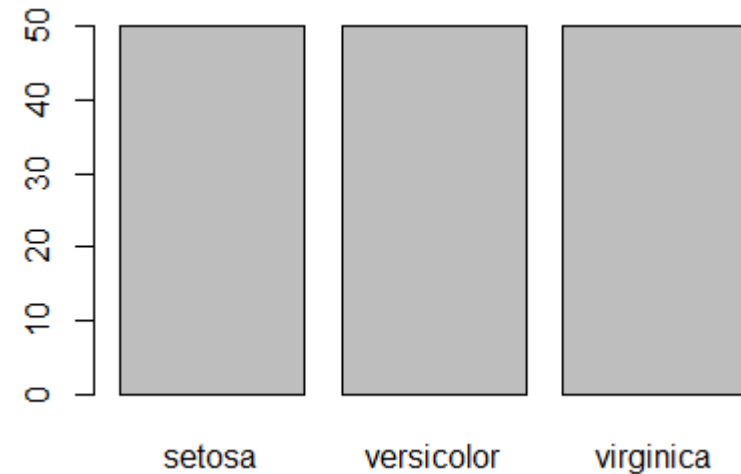
2. Diagramme en barre

La fonction `barplot()` est la plus commune pour les diagrammes en barre

```
# Tracer un graphique d'un vecteur  
x = c(1,2,5,2)  
barplot(x)
```



```
# Tracer un barplot du dataset iris  
  
Barplot(table(iris$Species))
```

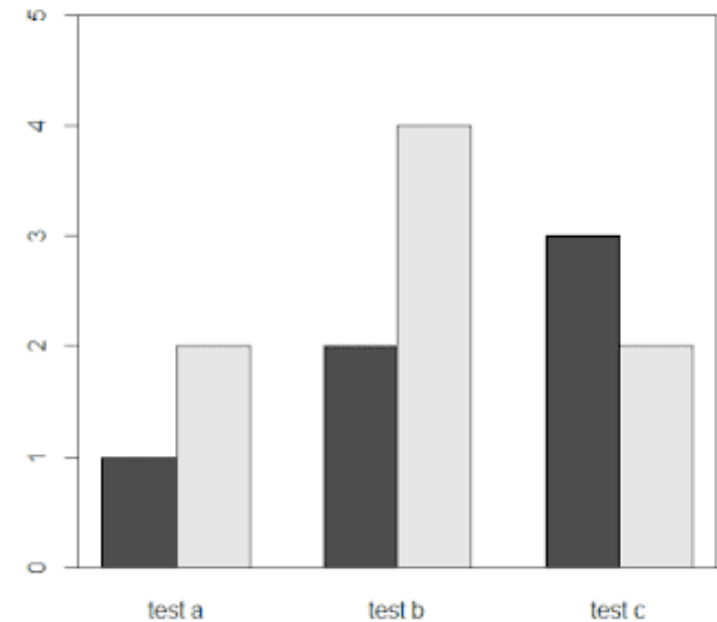


2. Diagramme en barre

Diagrammes à barres regroupées

```
# Tracer un graphique d'un vecteur

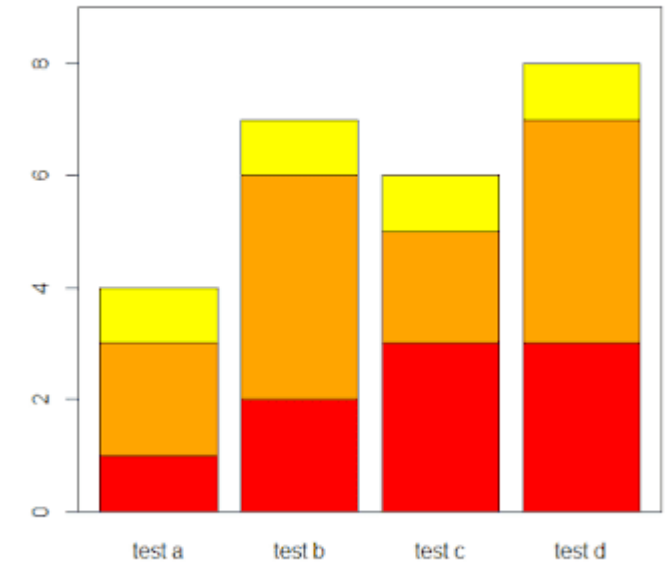
x = c(1,2,3)
y = c(2,4,2)
type = c("test a","test b","test c") moyennes =
c(x,y)
moyennes = matrix(moyennes,nc=3, nr=2, byrow=T) #
nc : nombre de tests - nr : nombre de barres
accolées (ici par paire) colnames(moyennes) = type
barplot(moyennes,beside=T)
```



2. Diagramme en barre

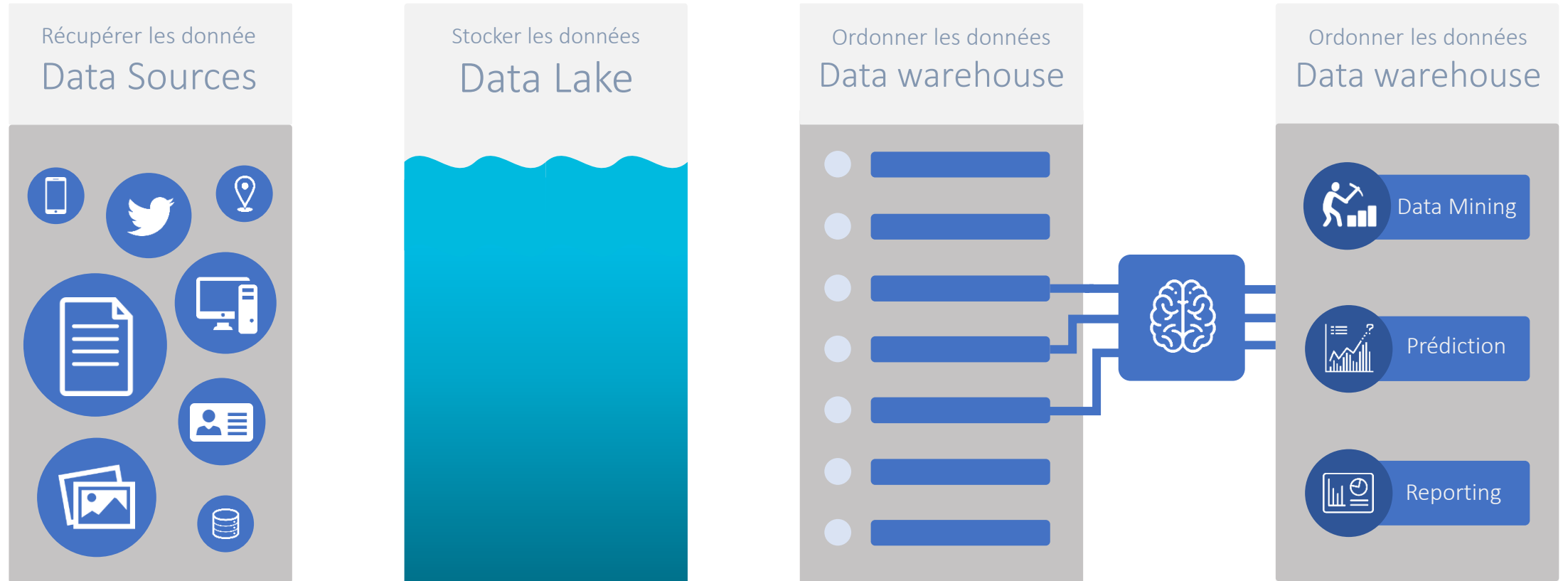
Diagrammes à barres empilées

```
x = c(1,2,3,3)
y = c(2,4,2,4)
z = c(1,1,1,1)
type = c("test a","test b","test c","test d")
moyennes = c(x,y,z)
moyennes = matrix(moyennes,nc=4, nr=3, byrow=T) # nc : nombre de
tests - nr : nombre de barres accolées (ici par 3)
colnames(moyennes) = type
barplot(moyennes,beside=F,col=c("red","orange","yellow","pink"),ylim=
c(0,9)) ; box()
```



3. Exemple de cas d'usage

- Utilisation de R dans la brique analytique d'une **architecture Big Data**



3. Exemple de cas d'usage

Data Science avec Microsoft

Pour la gestion de la volumétrie

Côté Azure

- Azure HDFS
- Azure DataLake
- Azure SQL Datawarehouse

Côté SQL Server

- Base OLTP / OLAP / InMemory
- Polybase

Pour la visualisation des données

- Power BI

Pour la puissance de calcul

- Azure Machine Learning
- SQL Server R Services

Microsoft a racheté la société **Revolution Analytics** et intégré son moteur dans SQL Server. Ce moteur permet d'exploiter la puissance d'un serveur centralisé et de faciliter la manipulation de données avec R depuis SQL Server

Pourquoi un serveur R ?

- *Pour alléger le travail à réaliser côté client*
- *Parce qu'un serveur aura plus de ressources qu'un poste client*
- *Pour automatiser, centraliser, partager son travail*

Exemple sur d'autres technos

Infrastructure Enedis

Données

- Oracle
- Hadoop

Calcul

- Serveur R

Visualisation des données

- Tableau

Client

- RStudio

3. Exemple de cas d'usage

Visualisation dynamique avec RShiny – étude de la qualité de l'eau sur les 6 dernières années

