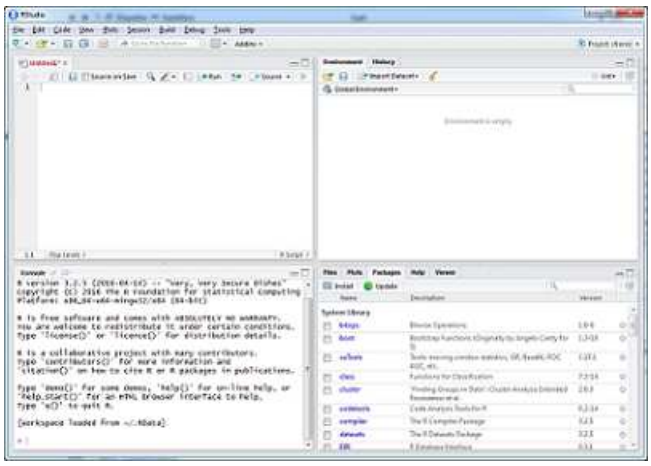
Chapitre 3 Présentation de l'interface et premières manipulations

3.1 Une interface particulière : RStudio



- Environnement de développement conçu spécialement pour R
- Interface utilisateur simple, conviviale, configurable et intégrant plusieurs outils
- Disponible sur http://www.rstudio.com

L'interface RStudio est composée de différents panneaux, dont l'arrangement peut être reconfiguré, incluant une console, un navigateur de fichiers et graphiques, l'espace de travail et l'historique des commandes. D'autres environnements graphiques existent, Rstudio semble le plus approprié à nos métiers

Lancer le script exemple

Démonstration : "./Presentation RStudio"

3.2 Session et répertoire de travail

- Session de travail R : commence à l'ouverture de RStudio et se termine en le quittant
- Répertoire de travail R : Dossier dans lequel le logiciel va chercher les fichiers de scripts et de données
- Tout ce qui a été fait au cours d'une session peut être enregistré dans le répertoire de travail :
  - données

- historique des fonctions …

Nouvelles fonctions	Attention
`setwd()` pour définir un répertoire de travail	Seulement / et pas \
`help()` et ? pour afficher l'aide	.
`dir()` pour lister un répertoire	.

Attention avec le répertoire de travail : si on l'écrit en dur `setwd('...')` , les anti-slash windows (\) doivent être remplacés par des slash(/)

Concernant le répertoire de travail, quelques conseils :

- Créer un nouveau répertoire pour un projet particulier qui sera votre répertoire de travail
- Créer des sous-répertoires dans ce répertoire : "./Data", "./Figures", "./Redaction"…
- Rassembler autant que possible les fichiers qui seront utilisés dans le cadre de ce projet et éviter d'aller chercher des fichiers ailleurs

3.3 Exercice

- Sur le poste de travail, créer un dossier devant servir de dossier de travail au cours de la formation.
- Dans Rstudio, définir ce dossier comme répertoire de travail directement en utilisant `setwd('')`
- Appeler l'aide en ligne par `?setwd` ou `help(setwd)`
- Faire une recherche dans la partie Help de RStudio
- Lire le contenu du répertoire de travail avec `dir()`

3.4 Prise en main de la console

- Effectuer et afficher les résultats de calculs de base `(+, -, *, /, **, ...)`
- Utiliser des fonctions spécifiques : `sum` , `abs` , `round` …
- On peut remonter dans l'historique des fonctions pour en rappeler une

Nouvelles fonctions	Attention
`sum()` pour sommer un résultat	Séparateur décimal point (.)
`abs()` pour retourner la valeur absolue	.
`round()` pour arrondir un nombre	.
flèches pour naviguer dans l'historique	.

3.5 Création d'une variable

- Assignation d'une variable

```
ma_variable <- 2
ma_variable <- "Toulouse"
ma_variable <- c ("Toulouse", "Nantes", "Strasbourg")
ma_variable <- 1:10
```

- Peuvent être numériques ou textes
- Peuvent être réutilisées

Nouvelles fonctions	Attention pour le nommage des variables
`ls()` pour faire lister les variables existantes	Casse -> seulement minuscules
`paste()` pour concaténer des variables textes	Pas de caractères spéciaux ni accentués
`rm()` pour supprimer une variable	Pas d'espaces ni tirets (-) ; préférer (_)

3.6 Exercice

- Créer plusieurs variables numériques par assignation `a<-5` , `b<-4` ,
- Regarder l'onglet Environnement
- Afficher la liste des variables avec la fonction `ls()`
- Faire un calcul avec ces variables et voir le résultat `(a+b)` etc.
- Créer une troisième variable à partir des deux premières `c<-a+b*3`
- Ré-assigner une variable : `a<-10` et vérifier l'onglet environnement
- Créer une variable chaîne de caractère (utilisation des simples quotes et des double-quotes) `t<-'chaine'`
- Concaténer a et t avec `paste(a,t)`
- Expérimenter la casse des noms de variables : créer `A<-15` et

`B<-12` et vérifier l'onglet environnement

- Supprimer les variables A et B avec la fonction `rm()`
- Aide en ligne *?ls* et *?rm*
- Attention : Pour supprimer toutes les variables `rm (list = ls ())`

3.7 Utilité des scripts

- Garder la trace d'une longue succession de lignes de code
- Pouvoir exécuter ce code (pas à pas ou en entier)
- Le modifier plus rapidement pour l'adapter
- Pouvoir le réutiliser avec de nouvelles données

Nouvelles fonctions	Attention
`#` pour des commentaires explicatifs du code	Importance de bien commenter
`print()` pour afficher sur la console	.
`Ctrl+R` pour lancer un script ligne par ligne	.

Utilisation simple de R $\rightarrow$ mode console. Chaque ordre, bout de code est rentré et exécuté au fur et à mesure sur la console.

Actions plus complexes, longues, nécessitant une maintenance, des modifications, $\rightarrow$ travailler à partir de la fenêtre éditeur. Les parties de code sont enchaînées et enregistrées sous la forme d'un fichier texte réutilisable par la suite.

Toujours utiliser des commentaires. Permet de pouvoir mieux comprendre ce que l'on a fait lorsqu'on reprend un programme plus tard ou lorsqu'on le donne à quelqu'un.

3.8 Exercice : utiliser un script

- Recopier le script ci-dessous et changer les paramètres pour le re-exécuter plusieurs fois
- Sauvegarder ce script dans votre répertoire de travail
- Fermer RStudio, le rouvrir, et réexécuter le script

```r
# Supprimer toutes les variables existantes
rm (list = ls())

# Création de mes variables taille et poids ; attention au point déci
poids <- 91
taille <- 1.87

# Calcul de l'IMC : poids sur taille au carré
imc <- poids/(taille^2)

# Affichage du résultat
print (imc)
```

```
## [1] 26.02305
```