1 L'environement RStudio

1.1 Éléments de RStudio

Présentation des différentes fenêtres de RStudio et personnalisation.
Par défaut La fenêtre de RStudio est composé de 4 panneau principaux
De gauche à droite puis de haut en bas

Le panneau à droite ...(images)

2 Premiers Pas

2.1 R est une calculette

```
###Calcul###

# Une opération simple
10 + 3
# Une Opération plus complexe
10 / (3+8) * 78
# les différents opérateur sont :
# multiplication *
# addition +
# division /
# soustraction -


###Stockage###

#stockage d'une valeur dans une variable
nombreX <- 50.8
#Accès à la valeur stockée dans la variable nombreX
#Utiliser l'auto-complétion. Commencer par écrire nomb puis appuyer sur la
    touche de tabulation.
nombreX



# Utilisation de plusieurs variables
# stockage de la variable nombreY (le symbole <- est identique à =)
nombreY <- 7.4
# Utiliser l'auto-complétion, si plusieurs solutions existent vous pouvez
# continuer de taper le nom de la variable pour que le choix soit plus
# restreint puis utiliser les flèches du clavier pour choisir la bonne
# variable. Une fois sélectionnée, appuyer soit sur entrée.
nombreX + nombreY
# Stockage du résultat de l'opération dans la variable sommeXY
# Conseil : utiliser l'historique en appuyant sur la 'flèche haut'
# on peut retrouver des commandes déjà écrites.
sommeXY <- nombreX + nombreY
# Opération avec variable et constante déjà existante.
sommeXY
```

2.2 Créer des objets et les utiliser

2.2.1 Vecteurs

```
####Création###

#création d'un vecteur numérique
```

```r
monVecteur1 <- c(20, 45, 78, 12)

# Une suite de nombre de 1 à 30
maSuite <- seq(from = 1, to = 30)

#Une suite de nombre paire
maSuitePaire <- seq(from = 2, to = 20, by = 2)


# Nous avons utiliser la fonction seq()
# Pour trouver à quoi serve les arguments taper la commande
?seq

# La documentation de cette fonction s'affiche sur la panneau en bas à droite de RStudio


#Création d'un vecteur avec chaine de caractère.
monVecteurA <- c("Mut_1", "Mut_2", "Mut_3")


###S'informer sur les objets####

##Taille de l'objet##

##Exercice 1 ####################
# Indiquer la longueur de tous les vecteurs crées au moyen de la fonction legnth()
#
# Longueur des vecteurs :
#
#
#
#
##
#Remarque :
# Pour connaitre toutes les vecteurs créés il faut regarder l'onglet Workspace
# dans le panneau en haut à droite de R studio.
# Il est également possible d'uiliser la fonction ls() dans la console R.

################################

##Accéder au élément##

#Accès à l'élément d'indice 1
monVecteur1[1]
#Accéder aux élément d'indice  3 , 4 puis 5.
# Remarques.


#Accès aux éléments d'indice de 1 à 3
monVecteur1[1:3]

##Calcul##

#opération sur le vecteur
monVecteur1 + 5.5

#addition de deux vecteurs
monVecteur2 <- c(10, 100, 5, 2)
```

```
monVecteur1 + monVecteur2
```

2.2.2 Matrices et data frame

```
###Creation###

##Matrice avec des nombres aléatoires##
maMatrice <- matrix(rnorm(100), ncol = 10)

##Expliquer cette ligne de commande##
#A quoi correspondent les différents mots composants cette ligne de commande
    .

##S'informer sur la matrice
#voir les n premières lignes d'une matrice
head(maMatrice)

#Combien de lignes la fonction head() affiche t'elle par defaut
#Comment afficher plus ou moins de ligne ? Utiliser la documentation de la
    fonction head pour trouver.
#Comment voir la matrice à partir des derniere lignes ?

#Dimension de la matrice : nombre de lignes et nombre de colones
dim(maMatrice)
#Que renvoit la fonction dim()

#La fonction ncol() permet de nous renseigner sur le nombre de colones
#Trouver une fonction similaire pour trouver le nombre de ligne

#Avec la fonction dim() afficher uniquement le nombre de lignes

###Acceder aux valeurs###

#Une matrice est un ensemble de vecteur.
#Chaque colone est un vecteur ainsi que chaque lignes.

#Recupper des lignes, des colones des sous matrices.
#Recuperer la 9eme colone.
col9 <- maMatrice[, 9]

#De la même façon récuperer une ligne au choix.

#Pour recuperer plusieurs ligne ou colone il faut indiquer les indices à
    recuperer sous forme de vecteurs.

#Pour les colones 1,3 et 7
#On indique les indices des colones à reccuperer
colones <- c(1, 3, 7)
#Puis
mesColones137 <- maMatrice[, colones]
#Il est possible de tout faire en une seule ligne de commande
mesColones137 <- maMatrice[, c(1, 3, 7)]

##Nom de ligne et de colones
#IMPORTANT les noms de lignes sont toujours uniques.

##Quelques fonctions à connaitre##
#Utiliser les fonctions suivante à la fois sur les des vecteur comme col9 et
    sur la matrice entière

```

```r
# sum()
# mean()
# median()
# length()
# summary()

#Calculer la moyenne d'un vecteur de votre choix sans utiliser la fonction mean()

###Nom des colones##
#la fonction colnames() permet d'obtenir le nom des colones d'une matrice ou d'un data.frame
colnames(maMatrice)

#Que retourne cette fonction ?

#Donner un nom au colones
colnames(maMatrice) <- LETTERS[1:ncol(maMatrice)]
colnames(maMatrice)

##Type et classe d'objet##
typeof(maMatrice)
class(maMatrice)

#Transformer une matrice en data.frame
monDataFrame <- as.data.frame(maMatrice)

#Autre moyen d'acceder au colone avec un data.frame
coloneC <- monDataFrame$C
```

3 Erreurs, importer exporter des données

3.1 5 erreurs courrantes

```r
#Ligne de commande erroné.
matriceBug <- read.delin("Rmatrix.txt" ,  rown.names = 1  head = true)

#Erreur 1 :
#Erreur 2 :
#Erreur 3 :
#Erreur 4 :
#Erreur 5 :

#Bonne ligne de commande :
```

3.2 Importer et exporter des données

```r
# Importation de données

#La fonction essentiel read.delim
#Sans aucun argument
mesEchantillons <- read.delim("samples.txt")

#Avec des arguments
mesEchantillons <- read.delim("samples.txt", row.names = 1, dec = ",", stringsAsFactor = FALSE, header = TRUE)

#Observer le nom des colones
```

```
11
12
13 mesEchantillons <- read.delim("samples2.txt", row.names = 1, dec = ",",
       stringsAsFactor = FALSE, header = TRUE)
14
15 #Possible d'importer des données avec R studio mes touts les arguments ne
       sont pas disponibles
16
17 #Sauvegarde dans un fichier
18 #La fonction essentiel write.table
19 write.table(x = mesEchantillons, file = "mesEchantillons.txt", col.names =
       NA , row.names = TRUE, quote = FALSE, sep = "\t")
```

4 Graphiques

```
1  #la fonction plot()
2  x<-1:20
3  plot(x, x^2)
4
5  #Il existe plusieurs fonction de base pour les graphiques.
6  # plot()
7  # hist()
8  # barplot()
9  # barplot2()
10 # boxplot()
11 # Utilisation basique
12 boxplot(maMatrice)
13
14
15 #Arguments communs à toutes les fonctions
16 plot(x, x^2, xlim=c(0, 30), ylim=c(-100, 500), xlab="Variable x", ylab="
       Variable x au carré", main="Carré des valeurs de 1 à 20", cex.axis=1.5,
       cex.lab=1.5, cex.main=2, bty="l", pch=16)
17
18
19 # Sauvegarde dans un fichier image
20 # Dans l'onglet Plots : Export-'Save Plot As Image'
21 # File name :  boxPlot
22 # La même chose avec la commande :
23 jpeg("boxPlot.jpeg")
24 plot(x, x^2, xlim=c(0, 30), ylim=c(-100, 500), xlab="Variable x", ylab="
       Variable x au carré", main="Carré des valeurs de 1 à 20", cex.axis=1.5,
       cex.lab=1.5, cex.main=2, bty="l", pch=16)
25
26 #fermer la fenêtre graphique en cours et enregistre le fichier pour l'
       ocasion
27 dev.off()
28
29 #Il existe 71 paramétres pour affiner les graphiques
30 #liste des 71 paramétre de la fonction par()
31 par()
```

5 Écrire et utiliser une fonction et l'utiliser dans un script

5.0.1 Préparation

Il est plus facile d'écrire les fonctions dans des fichiers. Créer un nouveau fichier et le nommer fonction.R La synthaxe pour écrire une fonction est la suivante :

```r
maFonction <- function(argument1, argument2 = valeurParDefaut){

  #On utilise le nom des argument comme variable pour faire des calcul, appeler des fonctions.
  resultatTemporaire <- argument1  + argument2

  #On continue différent traitement avec d'autre variable créer dans la fonction
  resultatTraiter <- uneAutreFonction(resultatTemporaire)

  #On retourne le resultats avec la fonction return()
  return(resultatTraiter)


}
```

Ennoncé : Créer la fonction *ingredientsPateAPizza* qui prend comme argument le nombre de pizza, par défaut 1. Elle doit retourner un vecteur indiquant les quantités pour les ingrédient.

Les ingrédients pour une pizza

- farine : 500
- eau : 250
- levure : 20

Aide : Attribuer un nom au indice d'un vecteur avec la fonction names() qui fonctione de la même façon que colnames pour les matrices et data.frame.

Tester sa fonction : La fonction est écrite dans le fichier fonction.R, mais elle encore inconnu pour R. Pour la faire connaitre il faut executer la commande. Il existe plusieurs façon de faire.

Avec la fonction *source*.

```r
source("fonction.R")
```

Si il n'y pas d'erreur de syntax, la fonction sera reconnu par R ce que l'on remarque que sur la fenetre Workspace.

On peut maintenant utiliser notre fonction de la même façon que n'importe quelles autres.