TP1 : Introduction au logiciel R

Dans cette première séance de travaux pratiques, nous introduisons le logiciel libre R que nous utiliserons dans les séances suivantes. En particulier, nous illustrons certains aspects de R en nous basant sur les notions de statistique vues en cours.

1 Mise en place

Il est possible d'utiliser R en mode interactif grâce à la commande `R` (ou `R --vanilla` pour éviter de charger et de sauver une session). Une fois cette commande validée, vous avez la possibilité d'entrer des commandes directement dans l'invite de commande de R (symbole `>`).

Le logiciel R propose de nombreuses fonctions de base. Pour quitter le mode interactif, il faut utiliser `q()`. Les fonctions `help` et `help.search` permettent d'obtenir de l'aide,

```
help(q) # Affiche l'aide sur la fonction q
?q # Comme help(q)

help.search("quit") # Cherche le mot quit dans l'aide
??quit # Comme help.search("quit")

q() # Quitte le mode interactif
```

- Que se passe-t-il si vous oubliez les parenthèses dans la dernière ligne ?
- Lisez l'aide de la fonction `mean`.

Le mode interactif est utile pour consulter l'aide et faire des essais mais il est moins adapté pour une utilisation avancée de R. Pour cela, nous utilisons un fichier (extension ".r" pour profiter de la coloration syntaxique) dans lequel nous entrons les commandes que R devra exécuter. Pour cela, un simple copier-coller des lignes de code dans le mode interactif suffit ou vous pouvez charger un fichier avec la commande `source`,

```
source("mon_fichier.r")
```

Dans ce fichier, vous pouvez ajouter des commentaires avec le caractère `#`. Tout ce qui est entre ce caractère et le prochain retour à la ligne n'est pas interprété par R (voir le premier exemple). Les commentaires sont particulièrement utiles lorsque nous reprenons un ancien fichier, par exemple.

Bien entendu, il existe de nombreuses autres façons d'exécuter un code R. Vous pouvez créer un script exécutable, utiliser une interface graphique comme Rcmdr, ... Voir sur internet si vous souhaitez plus de renseignements sur tout cela.

2 Des données

Pour charger un jeu de données, il faut utiliser une fonction adaptée selon le type de fichier qui contient ce jeu. Une telle fonction très commune est `read.table` qui permet de lire un fichier formaté en tableau et retourne une liste des valeurs. Comme tous les objets de R, cette liste

peut être stockée dans une variable grâce à l'opérateur d'affectation `<-` (il est possible d'utiliser également l'opérateur `=` mais cela est déconseillé pour éviter la confusion avec l'opérateur d'égalité `==`).

Pour cette séance, nous allons utiliser deux jeux de données extraits de l'article *Modeling wine preferences by data mining from physicochemical properties* de Cortez *et al.* paru dans *Decision Support Systems* en 2009. Ces jeux de données contiennent différentes mesures chimiques sur plusieurs vins rouges et blanc.

```
rouge <- read.table(
  "http://www.math.univ-toulouse.fr/~xgendre/ens/data/WineRed",
  header=TRUE,sep=";")
blanc <- read.table(
  "http://www.math.univ-toulouse.fr/~xgendre/ens/data/WineWhite",
  header=TRUE,sep=";")
```

- Avec l'aide sur la fonction `read.table`, expliquez à quoi servent les deux options `header` et `sep` que nous avons utilisées.
- Que fait la fonction `typeof` ? Que vous retourne-t-elle pour les listes `rouge` et `blanc` ?
- Que fait la fonction `names` ? Que vous retourne-t-elle pour les listes `rouge` et `blanc` ?
- Que fait la fonction `dim` ? Que vous retourne-t-elle pour les listes `rouge` et `blanc` ?
- A l'aide des trois questions précédentes, précisez les informations que contiennent nos deux jeux de données.

Afin d'accéder à une colonne particulière d'une liste, nous pouvons utiliser son nom ou le numéro de la colonne (elles sont numérotées à partir de 1). L'objet retourné est un vecteur contenant les valeurs de la colonne.

```
pH_rouge <- rouge$pH # Avec le nom de la colonne
pH_rouge <- rouge[[9]] # Avec le numéro de la colonne

blanc_density <- blanc$density # Avec le nom de la colonne
blanc_density <- blanc[[8]] # Avec le numéro de la colonne
```

3 Quelques fonctions pour manipuler les données

Pour calculer la moyenne uniforme des éléments d'un vecteur `v`, nous pouvons utiliser la fonction `mean`. Voici, par exemple, comment calculer les moyennes des pH des vins rouges et blancs,

```
moy_pH_rouge <- mean(rouge$pH)
moy_pH_blanc <- mean(blanc$pH)
```

Nous allons voir comment calculer des moyennes pondérées avec R et vérifier la formule de décomposition de la moyenne.

- Que fait la fonction `c` ? Utilisez-la pour calculer directement la moyenne uniforme des pH des vins rouges et blancs avec `mean`.
- Quel est le poids du groupe des vins rouges ? Et celui du groupe des vins blancs ?

- Avec l'aide sur la fonction `weighted.mean`, calculez la moyenne pondérée des moyennes de chaque groupe et retrouvez le résultat précédent.

Pour calculer des quantiles d'un jeu d'observations stocké dans un vecteur `v`, nous utilisons la fonction `quantile`. Sans plus d'argument, celle-ci calcule les quantiles à 0%, 25%, 50%, 75%, 100%. Pour avoir les quantiles à d'autres ordres, il faut manipuler le paramètre `probs`.

```
v <- 1:10
quantile(v) # Calcul des quantiles de base
quantile(v, probs=seq(0, 1, 0.1)) # Calcul des déciles
```

- Est-ce que les quantiles retournés concordent avec la définition du cours ?
- A quoi sert le paramètre `type` de la fonction `quantile` ? Pour quelle valeur de ce paramètre retrouvons-nous la définition du cours ?
- Calculez les quartiles et la médiane du taux d'alcool dans les vins rouges et celui dans les vins blancs. Commentez les résultats obtenus.

4 Premiers graphiques

Une des forces du logiciel R est la grande variété de graphiques qu'il peut produire et la facilité avec laquelle l'utilisateur peut créer ses propres graphiques. Nous nous contenterons dans cette séance d'utiliser des fonctions de base et laissons les paramètres graphiques pour les séances suivantes.

Dans le cours, nous avons présenté les boîtes à moustaches (ou *boxplot* en anglais). Il s'agit d'un graphique synthétique très utilisé en pratique. Pour l'afficher, le logiciel R propose une fonction `boxplot` qui permet également d'afficher plusieurs diagrammes à moustaches simultanément.

```
boxplot(rouge$pH) # Diagramme pour le pH des vins rouges
boxplot(rouge$pH) # Diagramme pour le pH des vins blancs
boxplot(rouge$pH, rouge$pH) # Les deux simultanément
boxplot(rouge) # Pour toutes les variables de la liste
```

- Quels commentaires pouvez-vous faire sur le pH des vins rouges et celui des vins blancs à partir du troisième exemple ?
- Discutez de l'aspect des boîtes dans le quatrième exemple. Quel problème cela pose-t-il ?
- A quoi sert la fonction `scale` ? Discutez du résultat de la commande `boxplot(scale(rouge))` par rapport à la question précédente.
- Affichez la boîte à moustaches de `rouge$residual.sugar` et commentez son aspect. Comparez ce résultat à la boîte à moustaches des mêmes données auxquelles vous appliquerez la fonction $f(x) = \ln(x)/x^{1/4}$.
- Grâce à l'aide de la fonction `boxplot`, vous pouvez commencer à vous familiariser avec certaines options graphiques. Essayez, par exemple, de modifier la couleur de la boîte à moustaches.

Pour comparer la répartition de deux jeux de données, nous avons vu le diagramme quantile-quantile (ou *QQ-plot* en anglais). Ce diagramme se trace avec la fonction `qqplot` (le paramètre `type` permet de tracer les segments joignant les points),

```
qqplot(rouge$alcohol, blanc$alcohol, type="l")
qqplot(rouge$pH, blanc$pH, type="l")
```

- Comment se lisent ces graphiques ? Que nous disent-ils sur les répartitions des variables `alcohol` et `pH` ?
- A l'aide de la fonction `abline`, vous pouvez tracer une droite par dessus le graphique. Utilisez cette commande pour étayer vos réponses précédentes.

5 Vos propres fonctions

Comme dans tout langage informatique, il est possible d'écrire ses propres fonctions en R. Voici un exemple de fonction qui calcule la variance des observations contenues dans le vecteur `v` passé en paramètre avec les poids uniformes.

```
variance <- function(v) {
  moy <- mean(v)
  var <- 0
  for (i in 1:length(v)) {
    var <- var + (v[i]-moy)^2
  }
  return(var / length(v))
}
```

Notez, en particulier, la syntaxe de la boucle `for`. Nous l'utilisons ici pour mettre en évidence les calculs successifs mais, en pratique, ce genre de boucle n'est pas conseillé et il vaut mieux utiliser les fonctions natives de R.

- Transformez cette fonction pour permettre à l'utilisateur de spécifier les poids qu'il veut utiliser. Comment passer certains poids par défaut si l'utilisateur n'en fournit aucun (les poids uniformes, par exemple) ?
- *(Optionnel)* Adaptez votre fonction pour qu'elle n'utilise plus de boucle `for`. Utilisez pour cela la fonction `weighted.mean`.
- Calculez, avec votre fonction, la variance des pH des vins rouges et blancs ensemble, puis celle des pH des vins rouges uniquement et enfin celle des pH des vins blancs.
- Calculez les variances intra et inter. Vérifiez votre résultat à l'aide de la formule de décomposition de la variance.

6 Pour ceux qui ont fini

Écrivez une fonction qui affiche la fonction de répartition associée aux observations contenues dans un vecteur passé en paramètre. Adaptez votre fonction pour afficher simultanément deux fonctions de répartition associées à deux jeux de données passés en paramètres. Proposez à l'utilisateur différentes options graphiques (couleur, ...).