« Systèmes Intelligents & Multimédia » TP N°1 et 2

Jean-Daniel Zucker

23 mai 2012

1 Démarrer R

R est un logiciel pour l'analyse statistique. C'est un logiciel libre ; il est disponible gratuitement et tourne sur différent système (PC Linux, PC Windows, Mac). Vous pouvez le télécharger ici : `http://cran.r-project.org/` si vous voulez l'installer chez vous.

Pour démarrer R sous Windows ou Mac, double-cliquez sur l'icône correspondante. Pour démarrer R sous Linux, double-cliquez sur l'icône du terminal (icône avec un petit écran noir), puis dans la fenêtre qui s'affiche tapez "R" et validez.

Utilisez RStudio de préférence.

Le signe ">" en début de ligne est "l'invite de commande" de R : le programme vous demande d'entrer une commande.

La commande suivante (à taper au clavier ; le ">" est l'invite de commande et ne doit pas être tapé) quitte R :

```
> q()
```

Astuce lorsque l'on utilise R, il est possible de reprendre la ligne de commande taper précédemment en appuyant sur la flèche du haut du clavier. Appuyer plusieurs fois permet de récupérer des lignes plus anciennes.

2 Syntaxe et manipulation de données

2.1 Commentaires

Dans R, tout ce qui suit le caractère # (= dièse) est un commentaire et n'est pas pris en compte par R :

```
> # Ceci est du baratin qui n'est pas pris en compte par R !
```

2.2 Variables

R étant prévu pour faire des calculs statistiques, il ne manipule que des tableaux de données. Ces tableaux sont stockés dans des *variables*, ce qui permet de leur donner un nom. Le nom des variables doit commencer par une lettre et peut contenir des lettres, des chiffres, des points et des caractères de soulignement (_), mais surtout pas d'espace.

L'opérateur = est utilisé pour donner une valeur à une variable ; il peut se lire "prend la valeur de" (NB : on trouve aussi l'opérateur <- ("flèche") qui a exactement la même signification).

```
> age = 28
```

Pour afficher la valeur de la variable âge, il suffit de taper le nom de la variable :

```
> age
[1] 28
```

la variable "age" est ici un tableau avec une seule case, qui contient le chiffre 28. Le 1 entre crochet indique qu'il s'agit de la case n°1. R numérote les cases des tableaux en commençant à 1.

2.3 Types de donnée

R peut manipuler des nombres entiers, des *flottants* (= nombres à virgule), des chaînes de caractère et des *booléens* (valeur vraie ou fausse) :

```
> age        = 28 # Entier
> poids      = 64.5 # Flottant
> nom        = "Ngo bo" # Chaîne de caractère
> enseignant = TRUE  #  booléen
> etudiant   = FALSE #  booléen
> telephone  = "01 48 38 73 34" # Chaîne de caractère !
```

Enfin, la valeur spéciale NA (*non available*) est utilisée lorsqu'une donnée est manquante.

2.4 Tableaux

2.4.1 Vecteurs

Un vecteur est un tableau à une dimension. Toutes les cases du vecteur doivent contenir des données du même type (des entiers, des chaînes de caractère,...). La fonction c() permet de créer un vecteur :

```
> ages = c(28, 25, 23, 24, 26, 23, 21, 22, 24, 29, 24, 26, 31, 28, 27, 24, 23, 25, 27, 25,
24, 21, 24, 23, 25, 31, 28, 27, 24, 23)
> ages
[1] 28 25 23 24 26 23 21 22 24 29 24 26 31 28 27 24 23 25 27 25 24 21 24 23 25
[26] 31 28 27 24 23
```

"[1]" indique que ce qui suit est la première valeur du vecteur, et "[26]" que la deuxième ligne commence à la 26ème valeur ; "[1]" et "[26]" ne sont pas des éléments du vecteur.

La fonction c() permet aussi de *concaténer* (= mettre bout à bout) des vecteurs :

```
> poids_groupe_temoin       = c(75.0, 69.2, 75.4, 87.3)
> poids_groupe_intervention = c(70.5, 64.2, 76.4, 81.6)
> poids = c(poids_groupe_temoin, poids_groupe_intervention)
> poids
[1] 75.0 69.2 75.4 87.3 70.5 64.2 76.4 81.6
```

Il est possible d'accéder à un élément du vecteur avec des crochets. Par exemple pour accéder au second élément du vecteur poids :

```
> poids[2]
[1] 69.2
```

Il est aussi possible d'accéder à l'ensemble des poids répondant à une condition, par exemple l'ensemble des poids supérieurs à 70.0 :

```
> poids[poids > 70.0]
[1] 75.0 75.4 87.3 70.5 76.4 81.6
```

Enfin, la fonction length() permet de récupérer le nombre d'éléments d'un tableau :

```
> length(poids)
[1] 8
> length(poids[poids > 70.0])
[1] 6
```

Il est possible de créer un vecteur contenant une suite de nombres entiers avec la fonction seq :

```
> seq(1, 10)
[1]  1  2  3  4  5  6  7  8  9 10
> seq(1, 10, by = 0.5)
[1]  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5  7.0  7.5  8.0
[16]  8.5  9.0  9.5 10.0
```

2.4.2 Matrices

Une matrice est un tableau à deux dimensions, c'est à dire avec des lignes et des colonnes. Comme pour les vecteurs, toutes les cases d'une matrice doivent contenir des données du même type.

Une matrice est créée à partir d'un vecteur contenant les valeurs, et d'un nombre de ligne (nr, pour Number of Row) et de colonne (nc, pour Number of Column) :

```
> ma_matrice = matrix(c(1.5, 2.1, 3.2, 1.6, 1.4, 1.5),nr=3, nc=2)
> ma_matrice
     [,1] [,2]
[1,]  1.5  1.6
[2,]  2.1  1.4
[3,]  3.2  1.5
```

Les éléments de la matrice peuvent être accédé en donnant entre crochets le numéro de la ligne puis celui de la colonne :

```
> ma_matrice[1, 1]
[1] 1.5
```

Il est aussi possible de récupérer une ligne ou une colonne entière, en omettant le numéro correspondant :

```
> ma_matrice[1,]
[1] 1.5 1.6
```

2.4.3 Listes

Une liste est un tableau à une dimension, qui peut contenir des données de différents types (contrairement au vecteur).

```
> ma_liste = list("JB", 28)
```

2.4.4 Tableaux de donnée (*data frame* en anglais)

Un tableau de donnée est un tableau où chaque colonne correspond à un attribut différents (âge, taille, poids par exemple) et chaque ligne à un individu différent. Il est possible de créer des tableaux de données dans R, cependant il est beaucoup plus facile de les charger à partir d'un fichier. On utilise pour cela la fonction read.table(). Voici le fichier taille_poids.csv :

```
nom,taille,poids
Ngo bo,1.70,64.0
Bo  bo,1.80,63.0
M. X,1.67,70.5
M. Y,1.69,95.0
M. Z,1.75,NA
```

Ce fichier peut être chargé ainsi dans R :

```
t = read.table("taille_poids.csv", sep=",", header=TRUE)
```

sep="," indique que dans le fichiers les différentes colonnes sont séparées par des virgules, et header=TRUE indique que la première ligne du fichier contient les noms des colonnes. Ces valeurs correspondent aux fichiers CSV (*Comma-Separated Value file* : fichier dont les valeurs sont séparées par des virgules) qui peuvent être généré facilement, par exemple en exportant depuis un tableur comme OpenOffice ou Excel.

```
> t
                nom taille poids
1 Ngo bo   1.70  64.0
2       Bo  bo   1.80  63.0
3              M. X   1.67  70.5
4              M. Y   1.69  95.0
5              M. Z   1.75    NA
```

Les noms des colonnes sont disponibles via la fonction names() :

```
> names(t)
[1] "nom" "taille" "poids"
```

Comme pour les matrices, il est possible d'accéder aux cases, lignes et colonnes d'un tableau. Les noms des colonnes peuvent être utilisés à la place de leurs index :

```
> t[1, 2] # Première ligne, deuxième colonne : taille du premier individu
[1] 1.7
> t[1, "taille"] # Première ligne, colonne taille: pareil
[1] 1.7
> t[1,] # Première ligne
                nom taille poids      IMC
1 NgoBaoCho    1.7    64 22.14533
> t[,"taille"] # Colonne taille
[1] 1.70 1.80 1.67 1.69 1.75
> t$taille # Notation raccourci pour la colonne taille
[1] 1.70 1.80 1.67 1.69 1.75
```

Il est aussi possible d'indexer avec une condition : par exemple, pour obtenir un tableau avec seulement les individus dont la taille est supérieure à 1m70 (ne pas oublier la virgule, qui sert à indiquer que l'on veut récupérer toutes les colonnes !) :

```
> t[t$taille > 1.7,]
           nom taille poids      IMC
2 NgoBaoCho   1.80    63 19.44444
5         M. Z   1.75    NA       NA
```

Enfin, la fonction write.table() permet d'enregistrer un tableau de donnée (row.names = FALSE permet de désactivé les numéros de ligne) :

```
write.table(t, "taille_poids_2.csv", sep = ",", row.names = FALSE)
```

Ce fichier pourra ensuite être rechargé avec read.table() comme ci-dessus.

2.5 Opérateur et calcul

2.5.1 Opérations

R permet de réaliser la plupart des opérations courantes à l'aide des opérateurs suivants : + (addition), - (soustraction), * (multiplication), / (division), ^ (puissance).

```
> 2 * 3 + 1
[1] 7
```

Les opérations sont réalisées sur chaque élément des tableaux. Par exemple, pour calculer l'Indice de Masse Corporelle (IMC) sur chaque individu du tableau de donnée chargé précédemment, en appliquant la formule $IMC = \frac{poids}{taille^2}$:

```
> t$poids / (t$taille ^ 2)
[1] 22.14533 19.44444 25.27878 33.26214 NA
```

R a calculé l'IMC pour chacun des 5 individus ! Notez que la donnée manquante (NA) se "propage", ce qui est logique : si le poids de M. Z est manquant, il n'est pas possible de calculer son IMC, qui est donc manquant lui aussi.

Il est possible d'ajouter une quatrième colonne avec l'IMC à notre tableau de la manière suivante :

```
> t$IMC = t$poids / (t$taille ^ 2)
> t
                 nom taille poids      IMC
1 NgoBaocho   1.70  64.0 22.14533
2       NgoBaoCho   1.80  63.0 19.44444
3               M. X   1.67  70.5 25.27878
4               M. Y   1.69  95.0 33.26214
5               M. Z   1.75    NA       NA
```

2.5.2 Comparaison

Les comparaisons se font avec les opérateurs <, >, <= (inférieur ou égal) , >= (suppérieur ou égal), == (égal), != (différent de). Ils retournent une (ou plusieurs) valeur(s) booléenne(s).

```
> 1 < 3
[1] TRUE
> t$IMC > 25
[1] FALSE FALSE  TRUE  TRUE    NA
```

Il est possible de combiner plusieurs comparaisons avec des & (et) ou des | (ou).

2.6 Fonctions

R définit un grand nombre de fonctions ; nous en avons déjà vu quelques unes. La fonction help() permet d'obtenir de l'aide sur une fonction :

```
help(mean)
```

Voici une liste des principales fonctions :

summary(tableau) affiche un résumé du tableau

length(vecteur) retourne le nombre de case d'un vecteur

ncol(tableau) retourne le nombre de colonne d'un tableau

nrow(tableau) retourne le nombre de ligne d'un tableau

q() quitte R

min(vecteur) retourne la plus petite valeur d'un vecteur

max(vecteur) retourne la plus grande valeur d'un vecteur

sort(vecteur) retourne une copie d'un vecteur après l'avoir trié

mean(vecteur) calcule la moyenne

median(vecteur) calcule la médianne

var(vecteur) calcule la variance

sd(vecteur) calcule la déviation standard

sqrt(nombre) retourne la racine carré d'un nombre

sum(vecteur) retourne la somme de toutes les valeurs du vecteur

round(flottant) arrondit un nombre

rank(vecteur) retourne un vecteur avec les rangs (c'est à dire avec 1 pour la plus petite valeur, 2 pour la seconde, etc)

quantile(vecteur) affichage par quantile

Nous avons vu que les valeurs NA se propagent. Cela est parfois gênant, par exemple lorsque l'on calcule une moyenne :

```
> mean(t$poids)
[1] NA
```

Dans ce cas, il faut demander à R de ne pas tenir compte des valeurs NA pour ce calcul :

```
> mean(t$poids, na.rm = TRUE)
[1] 73.125
```

2.7 Exercice

Nous allons étudier une base de prescription, afin d'étudier l'inertie thérapeutique dans l'hypertension. L'inertie thérapeutique correspond à la situation où le traitement anti-hypertenseur actuel du patient est insuffisant, et où le médecin ne modifie pas ce traitement.

1. Charger le fichier table_inertie.csv et afficher les données. Ce tableau contient les colonnes suivantes :

   medecin identifiant du médecin,

   patient identifiant du patient,

   sexe du patient,

   age du patient,

   diabetique vrai si le patient est diabétique,

   date2 date de la consultation,

   pas2 pression artérielle systolique lors de la dernière consultation,

   pad2 pression artérielle diastolique lors de la dernière consultation,

   type2 type de traitement en sortie de la dernière consultation (1=monothérapie, 2=bithérapie,...),

   date1 date de la précédente consultation,

   pas1, pad1, type1 comme pas2, pad2, type2 mais pour la consultation précédente,

   traitement_change vrai si le traitement a été changé lors de la dernière consultation,

   changement_recent vrai si le traitement a été changé moins de 3 mois avant la dernière consultation.
2. Calculer la moyenne de la pression artérielle systolique, sa variance, son écart type, ainsi que les valeurs minimum et maximum.
3. Combien y a-t-il de patients diabétiques ?
4. Ajouter une colonne "personne_agee" qui vaut vrai si l'âge est supérieur à 65 ans.
5. Extraire les lignes du tableau qui correspondent aux patients diabétiques et les mettre dans un nouveau tableau que l'on appellera d. De même pour les patients non-diabétiques, dans le tableau nd.
6. Dans les tableaux d et nd, ajouter une nouvelle colonne "traitement_satisfaisant" indiquant si le traitement de l'HTA est satisfaisant, c'est-à-dire si la tension artérielle est inférieure ou égale à 140/90mmHg (130/80 mmHg chez le diabétique).
7. Quel est le taux de patients diabétiques ayant un traitement satisfaisant ? de patients non diabétiques ?
8. Assembler les deux tableaux d et nd, afin de disposer d'un tableau global avec la nouvelle colonne "traitement_satisfaisant" :

   ```
   > t = merge(d, nd, all=TRUE)
   ```
9. Extraire les lignes du tableau qui correspondent aux patients ayant un traitement non satisfaisant et les mettre dans un nouveau tableau que l'on appellera ns.

3 Graphiques

3.1 Rappels de vocabulaire

Données quantitatives, données numériques lorsques ces variables sont des nombres sur lesquels les opérations arithmétiques (addition, soustraction, moyenne,...) ont un sens : par exemple, des températures, des concentrations, des poids, des volumes,... mais pas des numéros de département.

Données qualitatives dans le cas contraire : valeurs booléennes (vrai ou faux, oui ou non), ou définies par un ensemble de valeurs possible (par exemple une variable "fréquence" pouvant prendre les valeurs "rare", "fréquent", "très fréquent" ; ou bien une variable "souche d'Aspergillus" pouvant prendre les valeurs "Fumigatus", "Niger", "Lentullus" ; ou des numéros de département).

Données ordonnées lorsqu'il est possible de classer les valeurs de la variable selon un ordre. Les données quantitatives sont toujours ordonnées, et certaines données qualitatives le sont aussi (par exemple il est possible d'ordonner des fréquences mais pas des souches d'Aspergillus).

3.2 Graphique à 1 variable

3.2.1 boites à moustaches simple : 1 variable numérique

Pour afficher une boîte à moustache simple :

```
> boxplot(variable_numérique)
```

Lorsque l'on a plusieurs variables INDÉPENDANTES, il est possible de placer plusieurs boîtes à moustaches les unes à côté des autres, en séparant les variables par des virgules :

```
> boxplot(variable1_numérique, variable2_numérique, variable3_numérique,...)
```

3.2.2 Histogramme : 1 variable numérique

C'est une représentation plus classique et très utile ; on l'obtient avec :

```
> hist(variable_numérique)
```

Par défaut R applique la loi de Sturges (cf cours) pour déterminer le nombre de barres ; il est possible de préciser le nombre de barres manuellement (ici, 3), ou bien les valeurs auquelles auront lieu les coupures :

```
> hist(variable_numérique, breaks = 3)
> hist(variable_numérique, breaks = c(0.0, 0.2, 0.5, 0.6, 1.0))
```

3.2.3 Camembert : 1 variable qualitative

Un camembert se fait avec la commande pie :

```
> pie(variable_numérique)
```

Pour faire un camembert à partir d'une donnée qualitative, il faut d'a bord stratifier celle-ci avec la commande summary :

```
> pie(summary(factor(variable_qualitative)))
```

3.3 Graphique à 2 variables

Il est fréquent d'étudier une variable en fonction d'une (ou plusieurs) autres variables : par exemple la réponse biologique d'un organisme à une substance en fonction de la dose de substance,...

3.3.1 boites à moustaches : 1 variable numérique + 1 variables qualitatives

Pour afficher une boîte à moustache de la variable 1 pour chaque valeur possible de la variable 2 (la variable 1 est numérique et la variable 2 est qualitative) :

```
> boxplot(variable1_numérique ~ variable2_qualitative)
```

3.3.2 Graphique à deux dimensions (X, Y) : 2 variables numériques

La première variable est placée en X, la seconde en Y. Il est possible d'utiliser des variables numériques ou qualitatives, cependant ce type de graphique est surtout utile avec des variables qualitatives.

```
> plot(variable_numerique1, variable_numerique2)
```

3.3.3 Barre cumulée : 2 variables qualitative

Un diagramme en "barre cumulée" permet d'étudier 2 variables qualitatives :

```
> plot(factor(variable_qualitative1), factor(variable_qualitative2))
```

3.4 Graphique à 3 variables et plus

3.4.1 boites à moustaches : 1 variable numérique + 2 variables qualitatives

Pour afficher une boîte à moustache sur trois variables (ou plus) : ici on affiche une boîte à moustache pour la variable 1 pour chaque combinaison des variables 2 et 3 :

```
> boxplot(variable1_numerique ~ (variable2_qualitative + variable3_qualitative))
```

Astuce Le nombre de boîtes peut vite devenir important ; l'ensemble est souvent plus lisible à l'horizontal, et en mettant les étiquettes des axes à l'horizontal (las = 2) :

```
> boxplot(variable1_numerique ~ (variable2_qualitative + variable3_qualitative), horizontal = TRUE, las
```

3.4.2 Graphique de niveau : 1 variable numérique + 2 variables qualitatives

Avant d'utiliser ce type de graphique, il faut importer le module d'extention "lattice" (il suffit de le faire une seule fois) :

```
> require(lattice)
```

Les graphiques de niveau permettent d'étudier une variable numérique en fonction de deux variables qualitatives :

```
> levelplot(variable_numérique1 ~ variable_qualitative2 * variable_qualitative3)
```

Chaque "case" du graphique obtenue correspond à une valeur de la variable 2 et une valeur de la variable 3 ; la couleur de la case indique la valeur moyenne de la variable 1.

3.4.3 Nuage en 3D : 3 variables numériques

Avant d'utiliser ce type de graphique, il faut importer le module d'extention "lattice" (il suffit de le faire une seule fois) :

```
> require(lattice)
```

Les nuages de point en 3D permettent d'étudier le comportement de 3 variables numériques simultanément :

```
> cloud(variable_numérique1 ~ variable_numérique2 * variable_numérique3)
```

3.5 Mise en forme des graphiques

Il existe de nombreuses options pour mettre en forme les grahiques. En voici quelques-unes :

main titre du graphique

sub sous-titre du graphique

xlab titre de l'axe X

ylab titre de l'axe Y

D'autres options sont disponibles ; pour les obtenir, consultez l'aide, par exemple pour les histogrammes :

```
> help(hist)
```

3.6 Suite de l'exercice

1. Tracer un histogramme représentant la pression artérielle systolique.
2. Tracer un graphique représentant la pression artérielle systolique en fonction de la présence ou non d'un diabète.
3. Tracer un graphique représentant le changement de traitement en fonction du traitement satisfaisant ou non.
4. On considère qu'il y a "inertie thérapeutique" lorsque le traitement est insuffisant (tension trop élevé) et que le médecin n'a pas changé le traitement, et qu'il n'y a pas eu de changement récent. Extraire les lignes du tableau correspondant aux patients dont le traitement est insuffisant, et les placer dans un nouveau tableau "ns". Ajouter ensuite une colonne "inertie" indiquant s'il y a eu inertie thérapeutique.
5. Tracer un graphique pour l'inertie thérapeutique.
6. Tracer un graphique représentant l'inertie thérapeutique en fonction du nombre de médicament.
7. Combien y a-t-il de patients pour lequel l'inertie a pu être calculée ?

4 Comparaison de moyenne

4.1 Cas 1 : comparer deux moyennes observées dans des échantillons différents

Pour comparer 2 moyennes observées sur des échantillons (ou sous-échantillons) différents, on utilise le test de Welch Student :

```
> t.test(variable1_numerique, variable2_numerique)
```

Dans ce cas, on distingue en général deux groupes d'invidus : un groupe "témoin" et un groupe "expérimental", et l'on souhaite comparer la valeur moyenne d'une variable entre ces deux groupes.

4.2 Cas 2 : comparer deux moyennes observées sur un même échantillon

Pour comparer de deux moyennes observées sur un même échantillon, les valeurs étant appariées 2 à 2, on utilise le test T de student apparié :

```
> t.test(variable1_numerique, variable2_numerique, paired = TRUE)
```

Dans ce cas, les deux variables doivent avoir le même nombre de valeurs, chaque paire ($1^{\text{ère}}$ valeur de la variable 1, $1^{\text{ère}}$ valeur de la variable 2), ($2^{\text{ème}}$ valeur de la variable 1, $2^{\text{ème}}$ valeur de la variable 2),... correspondant à un seul individu. C'est notamment le cas des études du type "avant - après".

NB En général, ce type d'étude (sur des valeurs appariées) conduit à une meilleure sensibilité que le cas précédent (sur des valeurs non appariées), cependant il est parfois plus difficile à mettre en oeuvre !

4.3 Cas 3 : comparer une moyenne observée dans un échantillon à une moyenne théorique

Pour comparer une moyenne observée dans un échantillon à une moyenne théorique, on utilise le test T de student :

```
> t.test(variable_numerique, mu = moyenne_theorique)
```

4.4 Suite de l'exercice

1. Les patients diabétiques ont-ils un âge significativement différents des autres ?
2. La pression artérielle systolique moyenne augmente-t-elle significativement entre la dernière consultation et l'avant dernière ?
3. La tension artérielle systolique est-elle significativement supérieure chez les personnes âgées ?

5 Corrélation et régression linéaire

La fonction cor() permet de calculer le coefficient de corrélation linéaire :

```
> cor(variable1_numérique, variable2_numérique)
```

Ce coefficient indique si les deux variables sont liées de manière linéaire. Une valeur de 0 indique une absence de liaison, une valeur de 1 ou de -1 indique une linéarité parfaite.

En cas de relation linéaire, la fonction lm() permet d'effectuer une régression linéaire :

```
> lm(variable1_numérique ~ variable2_numérique)
Call:
lm(formula = variable1_numérique ~ variable2_numérique)
Coefficients:
(Intercept)          variable2_numérique
    Y0                       pente
# pente est la pente de la droite
# Y0 est son ordonnée à l'origine
```

La variable passée en premier à la fonction lm() doit être la variable dont on cherche à expliquer les variations, à partir de la seconde variable.

6 Exercice 2

Afin de tester la toxicité d'une variété de maïs OGM, 3 groupes de 10 rats a été nourri avec ce maïs. Le maïs OGM représentait 11% de la ration alimentaire dans le premier groupe, 22% dans le second, et 33% dand le troisième. Un quatrième groupe témoin de 60 rats a été nourri avec du maïs non-OGM. Après 90 jours, on mesure le poids du foie de chaque rat.

[Gilles-Eric Séralini *et al.*, New analysis of a rat feeding study with a genetically modified maize reveals signs of hepatorenal toxicity, 2007, Archives of Environmental Contamination and Toxicology, version française :
http ://www.criigen.org/full_article.pdf]

1. Le tableau de données est enregistré dans le fichier ogm.csv. Charger ce fichier. Quelles sont les variables dont nous disposons ? Sont-elles numériques ou qualitatives ?
2. Représenter graphiquement le nombre de rats dans chacun des 4 groupes (groupe à 0%, 11%, 22% et 33% d'OGM). Peut-on expliquer pourquoi le nombre de rats est plus important dans le groupe à 0% ?
3. Représenter graphiquement la répartition des rats mâles et femelles au sein des 4 groupes.
4. Représenter graphiquement la taille du foie en fonction de la quantité d'OGM présent dans l'alimentation des rats.
5. Créer les variables g0, g11, g22 et g33 contenant les lignes du tableau correspondant aux rats nourris avec 0%, 11%, 22% et 33% d'OGM, respectivement.
6. Chez cette espèce de rat, on considère que le poids du foie est anormal s'il dépasse 17g. Ajouter une nouvelle colonne au tableau indiquant pour chaque rat si le poids de son foie est anormal ou non.
7. Représenter graphiquement la répartition foie normal / anormal au sein de chacun des 4 groupes.
8. Comparer le poids moyen du foie des rats du groupe sans OGM, avec celui des rats du groupe à 11% d'OGM, puis à 22% et à 33%.
9. Pourquoi l'étude n'a-t-elle pas utilisé un protocole avec appariement des rats, type "avant consommation d'OGM" versus "après consommation d'OGM" (comme cela avait été fait pour les lapins, cf exercice précédent) ?
10. Calculer le coefficient de corrélation linéaire entre la taille du foie des rats et la quantité d'OGM absorbée. Y a-t-il un effet dose dans la toxicité de cet OGM ? Que peut-on en déduire sur le mécanisme d'action de cette toxicité ?
11. Effectuer une régression linéaire de la quantité d'OGM consommée sur le poids du foie. Qu'en déduit-on ?

7 Exercice 2 sur vos données

1. Exporter votre base en format csv et lisez la dans R
   (a) on écrira MesDonnees = read.table("mesdonnees.csv", sep=",", header=TRUE)
2. Définissez le vecteur des valeurs de la classe.
3. Représenter graphiquement le nombre de valeurs pour chaque classe.
4. Représenter graphiquement 3 variables que vous aurez sélectionné en fonction de la classe.
5. Calculer la corrélation entre vos variables numériques et la classe.
6. Faite une régression entre la variable ayant la meilleure corrélation et la classe