INITIATION AU LOGICIEL R - TP 1

Commandes et fonctions de bases, manipulation de vecteurs et matrices

---

Exercice 1 : Anticiper ce que font les 6 instructions suivantes, puis vérifier vos réponses en les saisissant dans la console.

1. `1:3^2`
2. `(1:5)*2`
3. `x<-2` et `2x<-2*x`
4. `x<-1` et `x< -1`
5. `var<-3` et `Var*2`
6. `TRUE+T+FALSE*F+T*FALSE+F`

Exercice 2 : Vecteurs

1. Créer les 3 vecteurs ci-dessous á l'aide de la fonction **rep**. Vous les nommerez respectivement `vec1`, `vec2` et `vec3`.
   `(1, 2, 3, 4, 5, 1, 2, 3, 4, 5, 1, 2, 3, 4, 5)`
   `(1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4, 5, 5, 5)`
   `(1, 1, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4)`
2. Créer le vecteur `vec4` ci-dessous avec la fonction **paste** :
   `"A1)" "A2)" "A3)" "A4)" "A5)" "A6)" "A7)" "A8)" "A9)" "A10)"`
3. Le vecteur `letters` contient les 26 lettres de l'alphabet. A l'aide d'une instruction R, déterminer la coordonnée de la lettre `q` dans l'alphabet.
4. Donner une instruction permettant de remplacer les valeurs manquantes d'un vecteur `x` quelconque par des zéros.
5. Donner une instruction permettant de remplacer les éléments à valeur négative d'un vecteur de nombres `y` par leur opposé.
6. Soit `x` un vecteur de nombres. Que font les instructions `which.min(x)` et `which(x==min(x))` ?

Exercice 3 : Quelques calculs

1. Programmer la somme suivante en une ligne de code : $\sum_{i=10}^{100}(i^3+4i^2)$.
2. Soient $\mathbf{x}=(x_1,x_2,\ldots,x_n)$ et $\mathbf{y}=(y_1,y_2,\ldots,y_n)$ deux vecteurs de taille n (prendre par exemple `x=sample(0:99,100,replace=TRUE)` et `y=sample(0:99,100,replace=TRUE)`.)
   (a) Créer le vecteur $(y_2-x_1,\ldots,y_n-x_{n-1})$.
   (b) Créer le vecteur $\left(\frac{\sin(y_1)}{\cos(x_2)},\frac{\sin(y_2)}{\cos(x_3)},\ldots,\frac{\sin(y_{n-1})}{\cos(x_n)}\right)$.
   (c) Créer le vecteur $(x_1+2x_2-x_3,x_2+2x_3-x_4,\ldots,x_{n-2}+2x_{n-1}-x_n)$.

(d) Calculer $\displaystyle\sum_{i=1}^{n-1} \frac{e^{-x_{i+1}}}{x_i + 10}$.

Exercice 4 : Valeurs manquantes

1. Fixer la graine du générateur de nombres aléatoires (**set.seed**) à la valeur 7 et créer un vecteur `vec1` de 100 nombres tirés uniformément entre 0 et 7 (**runif**). Calculer la moyenne empirique (**mean**) et la variance empirique des valeurs contenues dans `vec1`.

   Rappel : Soit $\mathbf{x} = (x_i)_{1 \leq i \leq n}$ un vecteur et $\overline{x}$ la moyenne empirique de ses composantes. La variance empirique des x_i est donnée par :

$$\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2 - \overline{x}^2.$$

2. Créer une copie du vecteur `vec1` nommée `vec2`. Fixer la graine du générateur de nombres aléatoires é la valeur 8. Sélectionner 10 indices au hasard parmi les 100 indices des composantes du vecteur `vec2` (**sample**). Remplacer la valeur des composantes correspondant à ces indices par `NA` (valeur manquante).
3. Calculer les moyenne et variance empiriques des éléments du vecteur `vec2` : directement (sans gestion des `NA`) puis en utilisant l'argument `na.rm=TRUE` (en gérant les `NA`).
4. Créer un vecteur `vec3` en supprimant les composantes de `vec2` qui contiennent une valeur manquante et calculer les moyenne et variance empiriques des composantes de ce vecteur `vec3`. Commenter l'action de `na.rm=TRUE`.
5. Créer une copie du vecteur `vec2` nommée `vec4`. Remplacer les valeurs manquantes de `vec4` par la moyenne empirique des composantes du vecteur `vec3`.
6. Créer une copie du vecteur `vec2` nommée `vec5`. Remplacer les valeurs manquantes de `vec5` par un tirage aléatoire avec remise parmi les valeurs non manquantes du vecteur `vec2` (**sample**).

Exercice 5 : Création et inversion de matrice, sélection dans une matrice

1. Créer la matrice `mat` suivante avec les noms de lignes et de colonnes :

$$\begin{array}{l} \\ \texttt{ligne-1} \\ \texttt{ligne-2} \\ \texttt{ligne-3} \\ \texttt{ligne-4} \end{array}
\begin{array}{c} \begin{array}{cccc} \texttt{colonne 1} & \texttt{colonne 2} & \texttt{colonne 3} & \texttt{colonne 4} \end{array} \\ \left(\begin{array}{cccc} 1 & 5 & 5 & 0 \\ 0 & 5 & 6 & 1 \\ 3 & 0 & 3 & 3 \\ 4 & 4 & 4 & 2 \end{array}\right) \end{array}$$

2. Créer un vecteur contenant les éléments diagonaux de la matrice `mat`.
3. Créer une matrice contenant uniquement les deux premières lignes de la matrice `mat`.
4. Créer une matrice contenant uniquement les deux dernières colonnes de la matrice `mat`.
5. Créer une matrice contenant toutes les colonnes de la matrice `mat` sauf la troisième.
6. Calculer le déterminant puis inverser la matrice `mat` en utilisant les fonctions appropriées. Vérifier par un calcul approprié que la matrice obtenue est bien l'inverse de `mat`.
7. Créer un vecteur dont les éléments sont obtenus en sommant ligne par ligne les éléments de `mat` (utiliser **apply**).
8. Créer un vecteur dont les éléments sont obtenus en prenant le plus grand élément de chaque colonne (utiliser **apply**).
9. Créer une matrice contenant les colonnes de la matrice `mat` dont tous les éléments sont plus petits que 6 (utiliser **apply** et **all** pour sélectionner ces colonnes).
10. Créer une matrice contenant les lignes de la matrice `mat` qui ne présentent aucun zéro.

Exercice 6 : Calcul matriciel et modèle de régression linéaire

On considère le jeu de données `mtcars` disponible sous `R`. Ce jeu de données contient les mesures de 11 variables qui décrivent les performances de 32 voitures (modéles 1973-1974). Soumettre successivement les commandes `data(mtcars)`, `dim(mtcars)`, `head(mtcars)` et `help(mtcars)`. Commenter les sorties. On considère le modèle :

$$\text{mpg} = \beta_0 + \beta_1 \text{hp} + \beta_2 \text{wt} + \varepsilon,$$

où mpg désigne le vecteur contenant la consommation de carburant des 32 véhicules (en gallons par miles), hp le vecteur contenant leur puissance (en nombre de chevaux) et wt le vecteur contenant leur poids (en milliers de livres). ε est un vecteur d'erreurs dont les termes sont indépendants et identiquement distribués suivant une loi normale $\mathcal{N}(0, \sigma^2)$. On cherche à estimer les paramètres β_0, β_1 et β_2 par la méthode des moindres carrés ordinaires (MCO).

1. A partir de `mtcars`, créer le vecteur $\mathbf{y}$ contenant les valeurs de consommation de carburant (`mpg`).
2. Créer une matrice $\mathbf{X}$ qui aura autant de lignes que $\mathbf{y}$ et 3 colonnes. La première colonne ne contiendra que des 1, la seconde contiendra le vecteur `hp` et la troisième le vecteur `wt` (on pourra utiliser la fonction **cbind**).
3. Calculer la transposée $\mathbf{X}^T$ de $\mathbf{X}$.
4. Calculer de deux façons différentes le produit matriciel $\mathbf{X}^T\mathbf{X}$ (pour l'une, on utilisera la fonction **crossprod**).
5. Calculer l'inverse $(\mathbf{X}^T\mathbf{X})^{-1}$ de ce produit matriciel.
6. Calculer le produit matriciel $\mathbf{X}^T\mathbf{y}$.
7. Calculer les estimations MCO des paramètres, c'est-à-dire calculer $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$. Vérifier le résultat obtenu en soumettant l'instruction `lm(mpg~1+hp+wt,data=mtcars)` dans la console.

Exercice 7 : Data-frame

1. Charger le package `MASS`.
2. A partir du jeu de données `iris` disponible sous R (utiliser **data**`(iris)` pour le charger puis `head(iris)` pour en visualiser les premières lignes), créer un jeu de données `iris2` ne contenant que les données pour lesquelles la variable `Species` prend la modalité `versicolor`.
3. Trier par ordre décroissant les données de `iris2` en fonction de la variable `Sepal.Length` (utiliser la fonction **order**).
4. A quel type d'objet appartient `iris$Species` ?
5. Soumettre la commande `by(iris$Sepal.Width, iris$Species,mean)` et commenter la sortie obtenue.

Exercice 8 : Utilisation des fonctions apply, lapply

1. (a) Calculer les statistiques de base (moyenne, min, max, etc) des trois variables du jeu de données `ethanol` (disponible sous `R` dans le package `lattice`).
   (b) Calculer les quartiles de chacune des trois variables (utiliser pour cela la fonction **apply** avec la fonction **quantile**).
   (c) Toujours avec la fonction **apply**, calculer tous les déciles de chacune des trois variables en utilisant l'argument `probs` de la fonction **quantile**.
2. Charger le jeu de données `Aids2` du package `MASS` et le résumer (**summary**). Soumettre le code suivant et commenter les actions réalisées :
```
indice=!unlist(lapply(Aids2,is.numeric))
Aids2.qual=Aids2[,indice]
lapply(Aids2.qual,levels)
```

Exercice 9 : Importation de données et fusion de tables

1. Importer dans `R` les fichiers `test1.csv`, `test2.csv` et `test3.csv` après en avoir examiné le contenu (voir par exemple la ligne 130 des fichiers `test2.csv` et `test3.csv`).
2. (a) Importer les fichiers `etat1.csv`, `etat2.csv` et `etat3.csv`.
   (b) Fusionner les trois data-frames en un seul en utilisant les colonnes communes.