Université Sorbonne Paris Nord, Institut Galilée
Année universitaire 2020–2021
mallein@math.univ-paris13.fr

TP 1 – Introduction à R

1 Introduction

Le logiciel R, sous licence libre GNU, est facile à installer à partir de la page du CRAN. Cette page continent toutes les ressources nécessaires à l'utilisateur de R, débutant ou expérimenté (fichiers d'installation, mises à jours, librairies, FAQ, newsletter, documentation...). De très nombreuses ressources et tutoriels sont disponible en ligne pour se former et se perfectionner, l'objectif de ce premier TP est de fournir une base minimale permettant l'utilisation de ce logiciel.

1.1 Pourquoi utiliser R

Le logiciel R est le logiciel le plus utilisé dans la communauté statistique académique. Il est également de plus en plus utilisé dans les services de R&D des entreprises industrielles, en concurrence avec les logiciels commerciaux. Il dispose d'une très riche librairie de quasiment toutes les procédures et méthodes statistiques de la littérature. Plus précisément, les recherches récentes sont d'abord développées et diffusées à l'aide de ce logiciel par la communauté scientifique.

Dans sa structure, R est un langage de programmation d'une syntaxe voisine de celle du langage C, et capable de manipuler des objets complexes sours forme de matrices, scalaires, vecteurs, listes, facteurs et *data frame*, qui est une structure centrale de R. Il propose une programmation matricielle et offre de nombreuses fonctionalités analogues à celles de Matlab/Octave.

1.2 Utilisation de R et RStudio

On conseille d'utiliser ici l'environnement de programmation RStudio, qui est relativement efficace. Mais il également possible d'utiliser R en ligne de commande. Dans cet environnement, on fera attention de se placer dans un répertoire approprié pour stocker du code. L'interface RStudio est décomposée en fenêtres telles que :

- une console dans laquelle il est possible d'exécuter des commandes R en direct,
- un éditeur de texte permettant d'écrire des scripts qu'il sera possible de lancer plusieurs fois,
- un explorateur de dossiers permettant de choisir le dossier courant,
- une liste des variables et objets (environnement) actuellement définies,
- plusieurs autres fenêtres accessibles (visualisation de graphes, terminal, gestionnaire de paquets, aide en ligne, etc.).

1.3 Références et bibliographie

Pour obtenir à la fois des informations plus précises sur R ainsi que les méthodes statistiques vues dans les différents cours, on pourra se rendre sur le site wikistat, qui représente une base de données importantes d'exemples et explications de techniques utilisées en statistique et en machine learning. Pour aller plus loin, n'importe laquelle des fiches présentes sur ce texte peut constituer un TP complet à tester de façon autonome.

2 Premières commandes

Exercice 1 (Nombres, classe `numeric`, variables)**.**

1. Taper les instructions suivantes dans la console. Que renvoient-elles ?

```
#Ceci est un commentaire
2+2
sqrt(2)

a = exp(2)
b = a + pi
a;b;(1+sqrt(5))/2

# liste des variables
ls()
# classe des données
class(a)
#effacer une variable
rm(a)
```

2. Calculer les quantités suivantes

$$\frac{2}{137+63} \quad \frac{\frac{2}{3}}{7} \quad \frac{2}{\frac{3}{7}} \quad (2^3)^2 \quad 2^{3^2}.$$

3. Calculer

$$\sin(\pi) \quad \sqrt{2}^2 - 2 \quad 2^{1023} \quad 2^{1024} \quad 1/0 \quad -1/0,$$

$$2 + \texttt{Inf} \quad \texttt{Inf} \times -\texttt{Inf} \quad \texttt{Inf} - \texttt{Inf} \quad \sin(\texttt{Inf}) \quad \texttt{Nan} + \texttt{Inf}.$$

4. En utilisant une variable auxiliaire et la touche ↑, afficher les dix premières puissances de 2.
5. Calculez $\frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}-1}+\frac{1}{\sqrt{2}+1}\right)$ en utilisant une variable intermédiaire. Vérifier le résultat obtenu.

Exercice 2 (Booléens, classe logical, `==`, `!=`, `<=`, `&`, `&&`, `|`, `||`, `!`)**.**

1. Tester les propositions suivantes :

$$2+2=4 \quad 2+2=5 \quad 2\leq 2 \quad 2\leq 3 \quad 2<3 \quad (1<2) \text{ et } (1<\texttt{Inf}) \quad (1<2) \text{ et } (3\leq \texttt{Nan}),$$
$$(1<2) \text{ ou } (0=1) \quad (1<2) \text{ ou } (3\leq 3) \quad \text{non(Faux et Faux)} \quad \text{(non Faux) et Faux.}$$

2. Tester les codes suivants et expliquer les résultats obtenus.

```
(2 + 2 == 4) * 5    2.5 & 15    3 | print("hello")    3 || print("hello")
      0 & NaN    1 & NaN    class(TRUE)    class(NA)    class(NaN)
```

Exercice 3 (Fonctions `nomDeFonction = function(args){ instructions; return(sortie)}`)**.**

1. Il est possible de définir des fonctions grâce à la syntaxe suivante. Une bonne pratique consiste à définir les fonctions dans un fichier à part portant le nom de la fonction.
   (a) Copier le code suivant dans un fichier appelé `factorielle.R`

```
factorielle = function(n) {
  if (n==1) resultat = 1 # arrêt de la récursion
  else resultat = factorielle(n-1)*n # appel récursif
  return(resultat)
}
```

   (b) Exécuter l'instruction `source("factorielle.R")` . Que renvoie `factorielle(10)` ?
2. Définir la fonction f qui à x associe x^3.
3. Définir la fonction g qui à x associe le couple $(\cos(x), \sin(x))$. On pourra utiliser la commande `list` qui transforme une série de données en liste.

Exercice 4 (`if cond1 { exp1 } else if cond2 { exp2 } else { exp3 }`).

1. Programmer la fonction `absolue` qui renvoie la valeur absolue d'une quantité mesurée.
2. Programmer la fonction `h` qui à x associe $x+1$ si $x \leq 1$, $2x$ si $1 \leq x \leq 3$, x^2 sinon.

Exercice 5 (`for (var in deb:fin) { expr }`).

1. Calculer la somme $1^2 + 2^2 + \cdots + 30^2$.
2. Calculer la somme des cubes des nombres impairs inférieurs à 100.
3. Écrire une fonction (itérative) prenant en paramètre n et retournant la valeur u_n définie de la façon suivante
$$u_0 = 1 \quad \text{et } u_{n+1} = \sqrt{1+u_n}.$$
4. Écrire une fonction qui calcule la somme suivante $\sum_{i=1}^{n}\sum_{j=1}^{n} j$.

Exercice 6 (`while (cond) { expr }`).

1. Calculer le plus petit entier N tel que $\ln\ln N > 2$.
2. On pose $\ell = \frac{1+\sqrt{5}}{2}$. Calculer le plus petit entier N tel que $|\ell - u_N| < 10^{-6}$, avec u la fonction définie dans l'exercice précédent.

Exercice 7 (Les vecteurs).

1. Exécutez l'instruction `x = 1:10` . Quelle est la classe de `x` ?
2. Exécutez l'instruction `x = c(1,2,3,4,5,6,7,8,9,10)` . Quelle est la classe de `x` ?
3. Exécutez l'instruction `y = 2 *x + 3` . Quel est son effet ?
4. Que renvoient `y[5]` ? `y[1:3]` ? `y[-4]` ? `y[11]` ?
5. Testez les instructions `y < 10` , `y - 10 < x` , `sin(y)` , `sin(y < 10)` . Commentez.
6. Que renvoie `y[y > 9 &y < 20]` ? Et `which(y > 9 &y < 20)` ?
7. Que pensez-vous du code `for (i in 1:length(x)) {y[i] = cos(x[i])}` ?
8. Testez les commandes `seq(a,b,t)` et `rep(x,n)` permettant de créer des vecteurs. Les utiliser pour calculer à nouveau la somme des nombres impairs inférieurs à 100 grâce à l'instruction `sum` .

Exercice 8 (Les matrices).

1. Exécutez l'instruction suivante `A = matrix(1:15,ncol=5)` . Que vaut `A` ? Quelle est sa classe ?
2. Exécutez l'instruction `B = matrix(1:15,nc=5,byrow=TRUE) ; B` . Qu'obtenez-vous ?
3. Que renvoient `A[1,3]` ? `A[,2]` ? `A[2,]` ? `A[1:3,1:3]` ?
4. Que renvoie `dim(A)` ?

Exercice 9 (Manipulation de matrices, `matrix`). 1. Créez les matrices

$$A = \begin{bmatrix} 3 & -1 & 2 & 5 \\ 1 & 0 & -10 & 3 \\ -1 & 1 & 3 & 10 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 10 & 0 & 2 \\ 3 & -1 & 1 & 4 \\ 2 & -1 & 0 & 2 \end{bmatrix}, \quad C = \begin{bmatrix} 2 \\ 4 \\ -1 \\ 1 \end{bmatrix}.$$

2. Expliquer le retour de `A[A[,1] > 0,]`
3. Que renvoient `A * B` et `A * C` ?
4. Que renvoient `A %*% B` et `A %*% C` ? Que renvoie `D = A %*%t(B)` ?
5. Testez les commandes `solve(D)` , `diag(D)` , `eigen(D)` .

Exercice 10 (Les listes `list`). La commande `list(nom1=el1,nom2=el2,...)` permet de créer une liste. La liste est la généralisation d'un vecteur, dont les élements peuvent être de différent types, et appelés de différentes façons.

1. On pose `x = list("toto",1:8)` . Que renvoit `x` ? `x[[1]]` ? `x[[2]]+10` ?
2. On pose `y = list(matrice=D,vecteur=f,texte="toto",scalaire=8)` . Que renvoie `names(y)` ? Et `y$vecteur` ? Et `y$vec` ? Proposer plusieurs méthodes pour accéder la quantité scalaire stockée dans `y` . Exécutez `y$test="coucou"` . Que devient `y` ? Que renvoie `y$t` ?

On utilisera souvent les listes pour renvoyer les résultats d'une fonction.

Exercice 11 (Appliquer une fonction à une liste)**.** La commande `apply(mat, margin, fun)` permet d'appliquer une même fonction `fun` à toutes les lignes (si `margin` vaut 1) ou colonnes (si `margin` vaut 2) d'une matrice.

1. Testez les expressions `apply(A,1,sum)` et `apply(A,2,sum)` .
2. Caculer la somme des valeurs de la matrice `B` sans faire appel à une boucle `for` .

Exercice 12 (Les data.frame)**.** Un *data frame* est analogue à la matrice de la même façon qu'une liste est analogue à un vecteur. C'est un tableau dont les colonnes peuvent être hétérogènes, et auxquelles un nom peut être donné. On peut transformer une matrice en data frame en utilisant la commande *as.data.frame()*.

1. On exécute le code suivant.

```
taille = c (147,132,156,167,156,140)
poids = c(50,46,47,62,58,45)
sexe = c("M","F","F","M","M","F")
H = data.frame(taille,poids,sexe)
```

2. Que renvoie la commande `summary(H)` ? Et `H$taille` ? Et `H[1,]` ?
3. On pose `MH = as.matrix(H)` . Quelle est la différence entre `MH` et `H` ?
4. Que renvoie `H[H$taille>150,2]` ? Et `H[H$taille>150,]` ?
5. Extraire de `H` la taille et le poids des individus de sexe masculin de taille supérieure à 150.
6. Testez les commades `plot(H$poids,H$taille)` et `plot(H$sexe,H$taille)` .

Un data frame `iris` est disponible pour faire des tests. Vous pouvez extraire la longueur des pétales des iris de genre setosa, ou la largeur des sépales des iris dont la largeur des pétales est inférieure à 1.5.

Exercice 13 (Gammes en R)**.** On essaira de trouver les méthodes les plus élégantes permettant de répondre aux questions suivantes, qui ne nécessitent normalement pas d'appels à une boucle `for` .

1. Calculer sans boucle la somme de tous les entiers entre 1 et 10000 qui sont multiples de 3 ou de 5.
2. (a) Écrire les tables de multiplications sous forme d'une matrice 10×10.
   (b) Changer la 3e colonne pour qu'elle indique la table de 12 à la place.
3. Construisez les vecteurs suivants

$$(1/k^2, 1 \le k \le 10) \quad (\exp(1+k/10), 0 \le k \le 20) \quad (2k+\sin(k), 0 \le k \le 100).$$

Exercice 14 (Arpèges en R)**.**

1. Écrire une fonction `est_premier(n)` renvoyant vrai si et seulement si n est un entier naturel premier.
2. Écrire une fonction `triangle_pascal(n)` qui renvoie une matrice de taille $(n, n+1)$ telle que l'entrée (n,k) corresponde au coefficient binomial $\binom{n}{k-1}$.
3. Écrire une fonction `eratosthene(n)` renvoyant le vecteur des nombres premiers inférieurs à n grâce à la méthode d'Erathostène.
4. Écrire la suite des quotients $n^n e^{-n}\sqrt{2\pi n}/n!$ pour $1 \le n \le 1000$ et représenter graphiquement ce vecteur avec `plot` .