



TD 4 à 7 Matrices et data frame dans R

ING1 EC503 algorithmique et programmation

Galharret Jean-Michel
département MSC

https://galharret.github.io/WEBSITE/cours_ONIRIS.html

Les matrices

Les matrices sont comme en mathématiques des tableaux de données. Attention comme pour les vecteurs les matrices ne contiendront que des éléments du même type (numérique, texte, booléen).

Définition et fonctions de base

```
A=matrix(c(1,2,3,  
          2,1,-1,  
          -3,-1,1),nrow=3)
```

Quelques fonctions utiles la dimension de la matrice

```
dim(A)
```

Le résultat est donc un vecteur ayant deux éléments le nombre de ligne et le nombre de colonnes

Indexation des éléments d'une matrice

Les éléments d'une matrice sont identifiés par un numéro de ligne et de colonne entre crochets $A[i,j]$

```
A[1,2]
# première ligne de A
A[1,]
# première colonne de A
A[,1]
```

Exercice :

1. Générer un vecteur fillmat contenant 250 valeurs tirées selon une loi uniforme continue de paramètres 0 et 10 grâce à la fonction *runif*.
2. Créer une matrice Mesures de 50 lignes et 5 colonnes contenant les valeurs de fillmat. Les colonnes seront nommées Mesure1,..., Mesure5.
3. Créer un vecteur Groupe de taille 50 contenant des valeurs entières entre 1 et 5 (fonction *sample*). On interprétera Groupe comme le groupe d'appartenance des 50 lignes de Mesures. Ajouter cette colonne à la matrice Mesures.
4. Combien de lignes appartiennent au groupe n°2 ?
5. Afficher le numéro des lignes appartenant au groupe n°2.
6. Afficher les lignes de Mesures correspondant au groupe n°2.
7. Déterminer combien Mesures contient de valeurs inférieures à 5 (on exclura la colonne groupe du comptage).
8. Calculer la moyenne des valeurs de Mesures correspondant au groupe n°4.
9. Même question mais pour les 3 premières colonnes uniquement.
10. Appliquer à Mesures les fonctions dim, ncol et nrow. A quoi correspondent ces nombres ?

Les data frames

C'est l'objet le plus important et le plus utilisé dans R il s'agit d'un tableau de données mais contrairement aux matrices on peut avoir différents types de données dans un dataframe.

Création d'un data frame

On crée deux variables x et y contenant $n = 100$ nombres distribués selon une loi uniforme continue sur $[0,20]$ (fonction runif) et une variable gr (expliquer le résultat du code correspondant).

La fonction **round** permet d'arrondir.

```
x<-round(runif(100,0,20),1)
y<-round(runif(100,0,20),1)
gr<-sample(c("ING1","ING2"),size=100,replace=T,prob=c(0.6,0.4))
df<-data.frame(NoteF=x,NoteM=y,gr=gr)
```

L'indexation des éléments dans un data-frame est la même que dans une matrice :

```
df[1,1]
```

On peut également sélectionner une colonne du data frame via \$

```
df$NoteF
```

Les fonctions utiles pour les data frame

summary

Cette fonction permet de résumer l'ensemble des variables du data frame.

```
summary(df)
```

apply

La fonction apply permet d'appliquer une fonction sur les lignes (1) ou les colonnes d'un data frame : par exemple si on souhaite appliquer la fonction moyenne (**mean**) sur les colonnes du data frame df on utilise :

```
apply(df[,1:2],2,mean)
```

Remarque : on n'a utilisé que les deux premières colonnes du data frame car la troisième n'est pas numérique (donc on ne peut pas calculer sa moyenne).

Importation d'un data frame inclus dans un package

Pour accéder aux data frames de R :

```
data()
```

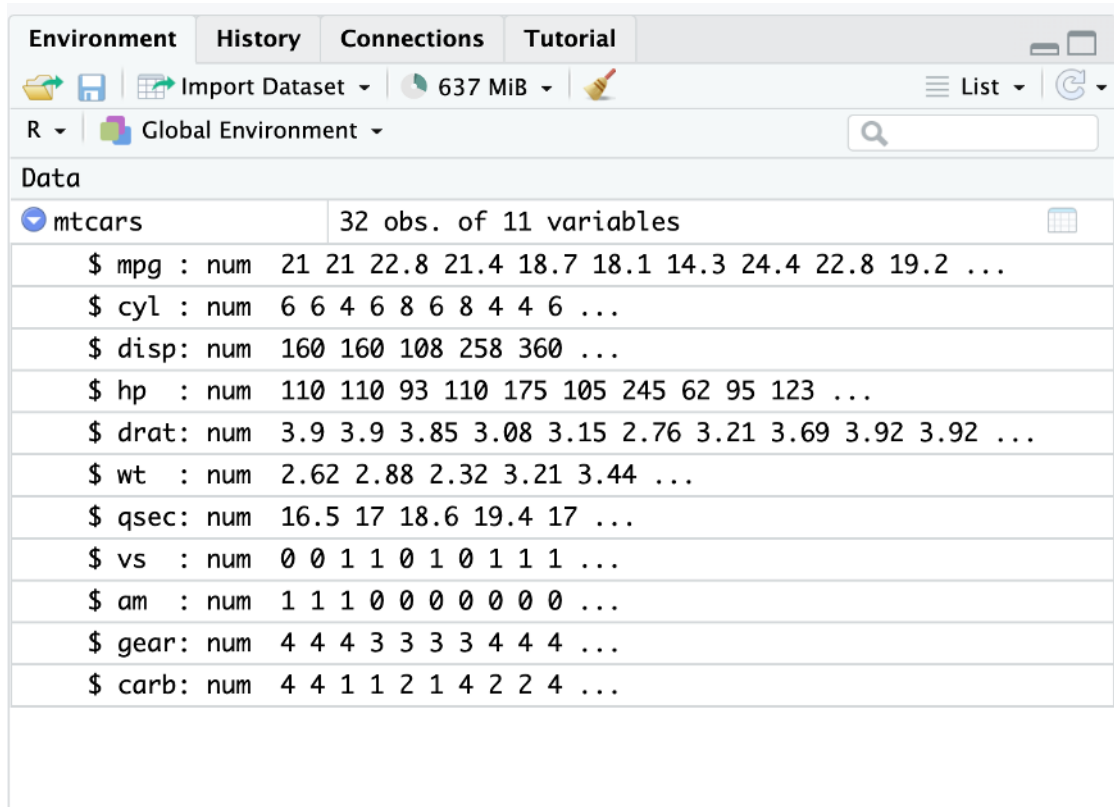
ensuite on peut grâce à la fonction help obtenir de l'aide sur l'un des data frame :

```
help("mtcars")
```

Ensuite on charge le jeu de données via

```
data("mtcars")
```

Une fois le data frame importé on constate qu'il est présent dans la fenêtre environnement



The screenshot shows the R Environment window with the following content:

Environment	History	Connections	Tutorial
R - Global Environment			
Data			
mtcars 32 obs. of 11 variables			
\$ mpg :	num	21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...	
\$ cyl :	num	6 6 4 6 8 6 8 4 4 6 ...	
\$ disp:	num	160 160 108 258 360 ...	
\$ hp :	num	110 110 93 110 175 105 245 62 95 123 ...	
\$ drat:	num	3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...	
\$ wt :	num	2.62 2.88 2.32 3.21 3.44 ...	
\$ qsec:	num	16.5 17 18.6 19.4 17 ...	
\$ vs :	num	0 0 1 1 0 1 0 1 1 1 ...	
\$ am :	num	1 1 1 0 0 0 0 0 0 0 ...	
\$ gear:	num	4 4 4 3 3 3 3 4 4 4 ...	
\$ carb:	num	4 4 1 1 2 1 4 2 2 4 ...	

Figure 1: Fenêtre environnement

La fonction factor()

Cette fonction va être très importante pour la suite et en particulier lorsque l'on va manipuler des jeux de données en statistique.

Un *facteur* est une variable qualitative. Les valeurs prises par cette variable sont appelées *modalités*. Il peut arriver que ces modalités soient codées. Par exemple dans certains jeux de données on connaîtra le statut professionnel des participants (modalités : en activité, en recherche d'emploi, étudiant, à la retraite) mais on peut avoir choisi de coder les modalités par des nombres (ici de 1 à 4) mais cette variable demeure un facteur.

Dans R :

```
statut_pro<-sample(1:4,100,replace=T)
## R ne voit pas qu'il s'agit d'un facteur car les modalités sont numériques
class(statut_pro)
```

```
[1] "integer"
```

```
table(statut_pro)
```

```
statut_pro
 1  2  3  4
20 25 24 31
```

On utilise la fonction `factor` pour dire à R qu'il s'agit d'une variable qualitative. On voit le changement à travers `class` ou `levels` :

```
statut_pro<-factor(statut_pro)
class(statut_pro)
```

```
[1] "factor"
```

```
levels(statut_pro)
```

```
[1] "1" "2" "3" "4"
```

```
levels(statut_pro)<-c("en activité", "en recherche d'emploi",
                    "étudiant", "à la retraite")
table(statut_pro)
```

```
statut_pro
      en activité en recherche d'emploi      étudiant
      20          25          24
à la retraite
      31
```

Remarque : Les variables binaires (facteurs à deux modalités) sont souvent codées 0/1.

On peut être amené à vouloir avoir les modalités d'un facteur classées dans un certain ordre pour ce faire on va utiliser la fonction `labels` avec l'ordre choisi :

```
statut_pro<-factor(statut_pro,labels=c("étudiants","en activité",
                                       "en recherche d'emploi","à la retraite"))
table(statut_pro)
```

```
statut_pro
  étudiants          en activité en recherche d'emploi
      20                25                24
  à la retraite
      31
```

Attention les labels doivent être exactement écrits de la même façon que les levels si les modalités sont déjà écrites et sinon on peut directement agir sur les modalités via labels :

```
data("mtcars")
help("mtcars")
mtcars$vs<-factor(mtcars$vs,levels=0:1,labels=c("V-shaped","straight"))
mtcars$am<-factor(mtcars$am,levels=0:1,labels=c("automatic","manual"))
```

Importation d'une base de données externe

On peut utiliser une interface graphique pour ouvrir les fichiers contenant le data frame. Les plus courants sont :

- *.csv : (Comma Separated Values) ce sont des fichiers de type tableaux sans mise en forme
- *.xlsx : fichiers produits à partir du logiciel excel
- *.ods : fichiers produits à partir du logiciel Calc de LibreOffice.

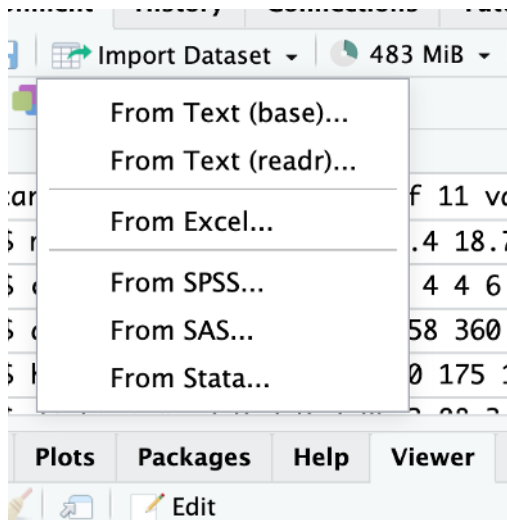


Figure 2: Menu Importer

On va importer le data frame contenu dans le fichier bordeaux.csv disponible sur Connect (vous devez le télécharger).

Vous avez deux solutions : ***From text (base)***

From text (readr)

Heading : permet d'utiliser la première comme identifications des colonnes du data frame (`colnames()`)
Row names : on peut aussi utiliser la première colonne comme identification des lignes du data frame.
Separator : on peut utiliser la virgule (comma), le point-virgule (semicolon), la tabulation (Tab) l'espace (Whitespace)
Decimal : soit la virgule (comma) soit le point (Period).

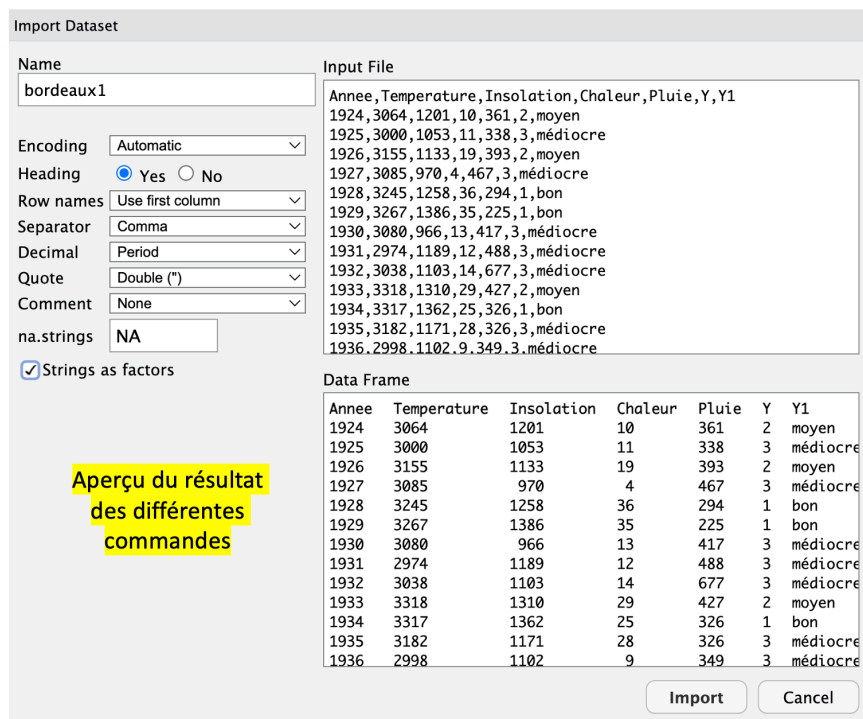


Figure 3: Importation avec la librairie base

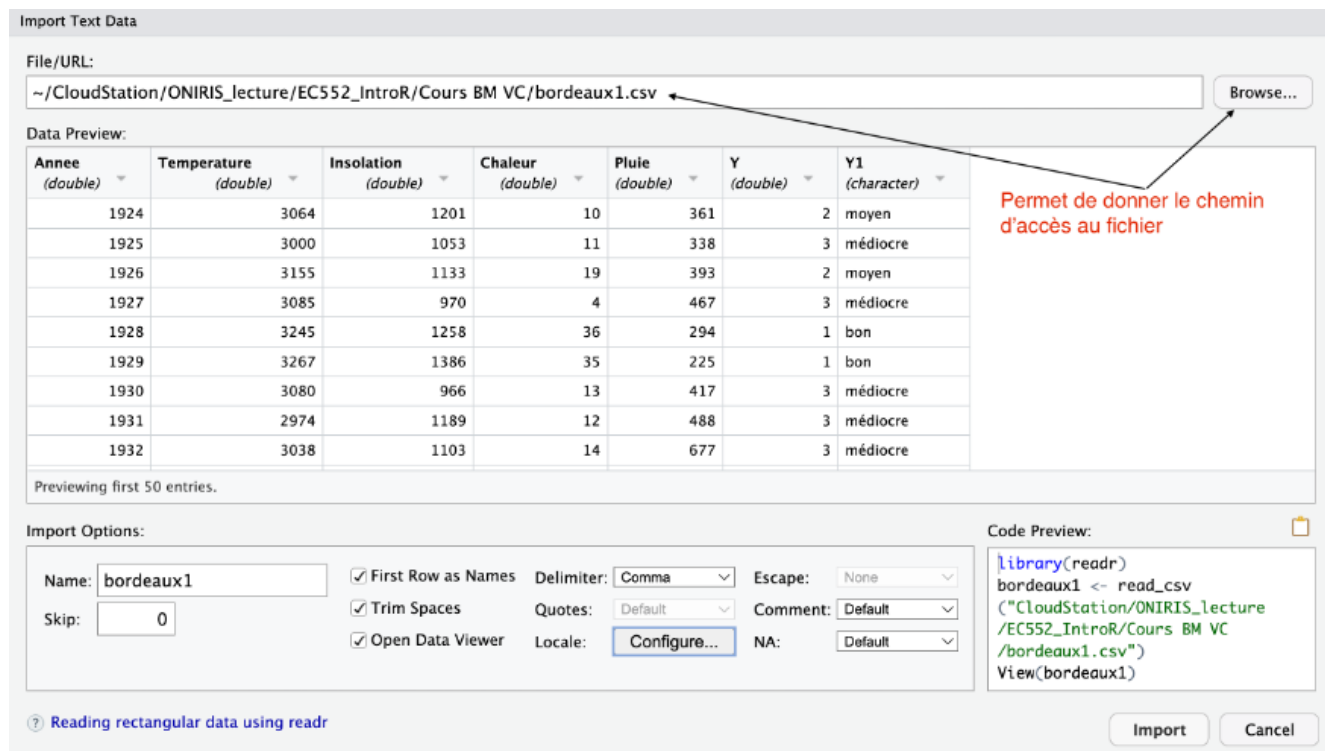


Figure 4: Importation avec la librairie readr

Ensuite il est recommandé de copier/coller la ligne de commande dans le script R afin de pas à avoir à reproduire la manipulation à chaque session de travail sur le fichier bordeaux.csv

The screenshot shows the RStudio interface. The top pane displays a data frame named 'bordeaux1' with 10 rows and 7 columns. The columns are: Annee, Temperature, Insolation, Chaleur, Pluie, Y, and Y1. The bottom pane shows the R console with the following code and output:

```

> library(readr)
> bordeaux1 <- read_csv("CloudStation/ONIRIS_lecture/EC552_IntroR/Cours BM VC/bordeaux1.csv")
Rows: 34 Columns: 7
— Column specification —————
Delimiter: ","
chr (1): Y1
dbl (6): Annee, Temperature, Insolation, Chaleur, Pluie, Y

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
> View(bordeaux1)
>

```

A red arrow points from the console code to the data frame header. A red text annotation in the console says: "Recopier cette ligne de code dans le script".

Figure 5: La fonction read_csv()

Exercice sur le data frame iris

1. Charger le fichier de données iris. Lire l'aide de iris pour comprendre le jeu de données.
2. Quel est le type de iris ? Quelles sont les dimensions de iris ?
3. Appliquer la fonction str à iris. A quoi correspondent les informations renvoyées ?

4. Appliquer la fonction `summary` à `iris`. A quoi correspondent les informations renvoyées ?
5. Utiliser les fonctions `colnames()` et `rownames()`. Quelles sont les informations renvoyées ?
6. A l'aide la fonction de texte `paste` remplacer le nom des lignes par fleur 1, ..., fleur 150.
7. Pour un data frame on peut appeler une colonne par son nom en utilisant `$` : `iris$Sepal.Length`. En utilisant cette information donner la classe de la colonne `Species`.
8. Quels sont les niveaux du facteur "Species" (fonction `levels`) ?
9. Créer une nouvelle colonne nommée `groupe` (on pensera à `$`) dans le data frame `iris` identique à la colonne `Species`. Quelle est la classe de cette nouvelle colonne ?
10. Renommer les niveaux de la colonne `groupe` en A, B, C (A pour `setosa`, B pour `versicolor`, C pour `virginica`). Afficher les indices des lignes de `iris` correspondant au groupe B.
11. Créer l'ensemble des numéros de lignes correspondantes aux fleurs du groupe A. Afficher les lignes de `iris` correspondant au groupe A. Proposer une solution alternative.
12. En adoptant la même logique que la question précédente, afficher uniquement les lignes de `iris` où « `Sepal.Length` » est inférieur à 5.
13. Combien y a-t-il d'individus ayant la longueur des sépales inférieure à 5 ?

Exercice sur le data frame `mtcars`

1. Charger le data frame `mtcars`. Combien de véhicules et de caractéristiques sur les véhicules sont contenus dans le data frame ?
2. Deux caractéristiques sont mal identifiées dans le data frame car ce sont des variables qualitatives (facteurs), rectifier en utilisant la fonction `as.factor`.
3. En utilisant l'aide du data frame `mtcars` renommer les niveaux des deux facteurs précédents.
4. Etablir la table de contingence de ces deux facteurs (fonction `table`)
5. Calculer la moyenne des 7 premières variables (on utilisera les fonctions `mean` et `apply`).
6. Reprendre la question pour les voitures avec un moteur à plat. De même pour les voitures ayant un moteur en V et qui sont automatiques.
7. Calculer le nombre de voitures ayant un moteur en V qui ont au moins 3 carburateurs.