Formation
Logiciel R

Institut Pasteur
de Nouvelle-Calédonie

1

DEUXIÈME SÉANCE

2

RAPPEL

R orienté objet

- Objet simple
  >x<-4
- Vecteur
  >Chiffre<-c(1:15)
- Matrice
  >Matrice1<-
  matrix(Chiffre,ncol=3)
- Liste
  >Mlist<-list(x,Chiffre,Matrice.1)
- Table de donnée
  >Data<-data.frame(Matrice1)

- Fonctions
  - sum()
  - c()
  - sep()
  - seq()
  - help()
  - ……..

2° séance

GESTION DES DONNÉES AVEC LE LOGICIEL R

5

Organisation des données

Objectif : décrire les données

6

Table de données – dataframe

Variables | Variables | Variables

Observations | Observations | Observations

row.names	CODSEX	SEXE	POIDS	GROUPAGE	POIDS20	POIDSENG	POIDS15	GROUPOIDS	Nivscolmer
VR 1	2	Homme	10	0-4	0	10000	1	(9,15]	Primaire
VR 2	2	Femme	10.5	0-4	0	10500	1	(9,15]	Primaire
VR 3	2	Homme	11	0-4	0	11000	1	(9,15]	Primaire
VR 4	2	Homme	11.5	0-4	0	11500	1	(9,15]	Primaire
VR 5	1	Femme	12	0-4	0	12000	1	(9,15]	Primaire
VR 6	1	Femme	12.5	0-4	0	12500	1	(9,15]	Primaire
VR 7	1	Femme	13	0-4	0	13000	1	(9,15]	Primaire
VR 8	2	Homme	13.5	0-4	0	13500	1	(9,15]	Primaire
VR 9	2	Homme	14	5-9	0	14000	1	(9,15]	Primaire
VR 10	1	Femme	14.5	5-9	0	14500	1	(9,15]	Primaire
VR 11	2	Homme	15	5-9	0	15000	0	(9,15]	Primaire
VR 12	1	Femme	15.5	5-9	0	15500	0	(15,20]	Primaire
VR 13	1	Femme	16	5-9	0	16000	0	(15,20]	Secondaire
VR 14	1	Femme	16.5	5-9	0	16500	0	(15,20]	Secondaire
VR 15	1	Femme	17	5-9	0	17000	0	(15,20]	Secondaire
VR 16	2	Homme	17.5	5-9	0	17500	0	(15,20]	Secondaire
VR 17	2	Homme	18	5-9	0	18000	0	(15,20]	Secondaire

7

Action sur les variables

- Trier les données
- Regroupement en classe
- Transformation de variable

XE	POIDS	GROUPAGE	POIDS20	POIDSENG	POIDS15	GROUPOIDS	Nivs
mme	10	0-4	0	10000	1	(9,15]	Prim
mme	10.5	0-4	0	10500	1	(9,15]	Prim
mme	11	0-4	0	11000	1	(9,15]	Prim
mme	11.5	0-4	0	11500	1	(9,15]	Prim
mme	12	0-4	0	12000	1	(9,15]	Prim
mme	12.5	0-4	0	12500	1	(9,15]	Prim
mme	13	0-4	0	13000	1	(9,15]	Prim
mme	13.5	0-4	0	13500	1	(9,15]	Prim
mme	14	5-9	0	14000	1	(9,15]	Prim
mme	14.5	5-9	0	14500	1	(9,15]	Prim
mme	15	5-9	0	15000	0	(9,15]	Prim
mme	15.5	5-9	0	15500	0	(15,20]	Prim
mme	16	5-9	0	16000	0	(15,20]	Seco
mme	16.5	5-9	0	16500	0	(15,20]	Seco
mme	17	5-9	0	17000	0	(15,20]	Seco
mme	17.5	5-9	0	17500	0	(15,20]	Seco

Exemple de commande avec R

Ifelse(Matable$POIDS<=15, "(9,15]",ifelse(Matable$POIDS>15&Matable$POIDS<=20,"(15,20]","(20,25]"))

8

Importer des données

Définir votre répertoire de travail

Rappel:

Au démarrage de R penser à définir votre répertoire de travail

```
>setwd(dirname(file.choose()))
```

```
>getwd()
```

#pour vérifier la prise en compte du nouveau répertoire de travail

Passage d'Excel/Open-Office à R

- Souvent les données sont saisies sous Excel/ OpenOffice. La première ligne contient les noms des variables
- Enregistrer depuis Excel/OpenOffice la table de données au format csv (séparateur : point virgule)
- Pour l'importer, utiliser dans R la fonction **read.table ()**

11

Importer des données Format texte

- Si vous ne connaissez pas le format du fichier texte des données :

```
file.show(file.choose())
#pour  afficher le contenu d'un fichier texte
```

Comment sont codées les données manquantes?

12

Importer des données
Format texte

Mydata<-read.table(file.choose(), header=T, na.string= "999" ,dec=",", sep="; ",row.names="NUMIDENT")

- Demander d'afficher les noms des variables **header=TRUE**
- Indiquer les données manquantes codées 999 dans le fichier d'origine **na.string**
- Séparateur décimal est une virgule dans le fichier d'origine **dec=","**
- Séparateur des colonnes est un point virgule **sep = "; "**
- Le nom des lignes est transféré depuis la colonne NUMIDENT
- row.names="NUMIDENT"

13

Fonction dim()

- **Les dimensions du data.frame**

>dim(Mydata)

>dim(Mydata)[1] # nombre de lignes

>dim(Mydata)[2] # nombre de colonnes

- **Le nombre de colonnes**

>length(Mydata)

14

Fonction names()

- **Connaître le nom des variables du data.frame**

```
> names(Mydata)
```

15

Fonction row.names()

- Connaître le nom des lignes
- Changer le nom de la ligne 5

```
>row.names(Mydata)[5]<-  "AtR123456"  #peut-
être utile quand on trouve des doublons
```

Fonction dimnames()

- Connaitre le noms des lignes et des colonnes

```
>dimnames(Mydata)
```

16

Fonction str()

>str(Mydata)

Nombre d'enregistrements

Nombre de variables

Nom des variables

Type des données de chaque variable

17

Fonction mode()

- **Connaître le mode des variables du data.frame**

> mode(Mydata$AGE)

Exemples:

- > Aa <- " Surveillance"; Compar <- TRUE; Zz <- 1i; Xx<-3
- > mode(Aa); mode(Compar); mode(Zz);mode(Xx)
- [1] "character"
- [1] "logical"
- [1] "complex "
- [1] "numeric"

18

Fonction sapply()

- Connaître la nature des colonnes

>sapply(Mydata,is.numeric)

19

Modifier des données

20

Fonction factor()

- Permet de transformer une variable numérique en variable catégorielle de façon définitive

```
>summary(Mydata$SEXE)
>Sexe<-factor(Mydata$SEXE)
>summary(Sexe)
```

(attention la variable sexe factorisée n'est pas intégrée dans Mydata)
On doit l'intégrer:

```
Mydata<-data.frame(Mydata, Sexe)
```

On aurait pu écrire directement:

```
Mydata$SEXE<-factor(Mydata$SEXE)
```

21

Fonction factor()

- Transformer plusieurs variables en variables catégorielles

```
>names(Mydata) #pour retrouver les variables à transformer
>Transvar <- c( 4, 8:13, 16) #créer un vecteur pour choisir les numéros de colonnes des variables à transformer
>for (i in 1:8) {Mydata[, Transvar[i]] <- factor(Mydata[, Transvar[i]])}
```

22

Fonction as.factor()

- Permet de considérer qu'une variable numérique est une variable catégorielle
  Mais pas de façon définitive

>summary(Mydata$SEXE)

>summary(as.factor(Mydata$SEXE))

>summary(Mydata$SEXE)

```
>Exemple:
 a<-c(1,2,2,3,4,5)
> summary(a)
   Min. 1st Qu.  Median    Mean
3rd Qu.    Max.
  1.000   2.000   2.500   2.833
3.750   5.000
> b<-a
> b<-as.factor(b)
> b
[1] 1 2 2 3 4 5
Levels: 1 2 3 4 5
> summary(b)
1 2 3 4 5
1 2 1 1 1
> plot(b)
> plot(a)
> hist(a)
```

23

Fonction names()

- **Renommer une variable (une colonne)**

>names(Mydata)[2]<- "AGENJOUR" #modifier le nom de la troisième colonne

24

Effacer le nom des lignes

>row.names( )<-NULL

Effacer le nom des colonnes

>Mydata1<-Mydata

>names(Mydata1)<- NULL

#attention le nom de colonnes sera remplacée par des valeurs par défaut

Redonner les noms aux variables de Mydata1

>Nomvar<-names(Mydata)

>names(Mydata1)<-Nomvar

25

Voir les données d'une colonne

>Mydata[,3] #extrait les données de la 3ième colonne

Voir les données d'une ligne

>Mydata[2,] #permet de voir les données d'un enregistrement

Voir l'information d'une variable pour un enregistrement

>Mydata[2,3] #permet de voir les données de la colonne 3 pour l'enregistrement 2

26

Voir une partie des données

>Mydata1<-Mydata[1:5,1:6]

#extrait les données de la 1ière à la 5ième ligne et de la 1ière à la 6ième colonne

>head(Mydata)

27

Fonction subset()

- Extraire une partie de la base de donnée

>Mydata.3<-subset(Mydata,Mydata$VALEURF > 38)

#insère dans une nouvelle table les données filtrées

Autre méthode

>Mydata.3<-Mydata[Mydata$VALEURF> 38,]

#la virgule après 38 indique que la recherche porte sur les lignes

28

Extraire des données (suite)

- Les 5 premières lignes avec les variables 2 à 5

> Mydata.4<-Mydata[1:5, 2:5]

- Les 5 premières lignes mais avec les variables 1, 3 et 6

>Mydata.5<-Mydata[1:5, c(1,3,6)]

- Toutes les lignes (sauf la 3ème), toutes les variables (sauf la 1ère)

>Mydata.6<-Mydata[-3, -1]

- Les 5 premières lignes, toutes les variables sauf les n° 1, 3, et 5

>Mydata.7<-Mydata[1:5, c(-1,-3,-5)]

29

Modifier les données d'un tableau

- **Créer une nouvelle variable**

> Agenanne<-Mydata$AGENJOUR/365

>Agenanne<-round(Agenanne, digits = 0)

- **L'ajouter dans le tableau de données**

>Mydata<-data.frame(Mydata,Agenanne)

30

Modifier les données d'un tableau(2)

- Création d'une variable
- Affecter des valeurs en fonction des données d'une autre variable

>Mydata$FIEVRE<-ifelse(Mydata$VALEURF < 38,0,1)

31

Fonction transform()

- Ajouter une colonne (Une variable) à partir des données d'une variable existante

> Mydata<-transform(Mydata,AGENAN=AGENJOUR/365)

ou mieux

>Mydata<- transform(Mydata,AGENAN=round(AGENJOUR/365,digit=0))

NB : dans la fonction transform(), le signe = ne peut pas être remplacé par <-

32

Fonction cut()

Pour créer des classes à partir d'une variable quantitative

Classes de type:]a,b]

```
summary(Mydata$AGENAN)
```

Exemple:

```
Mydata$TRANCHAGE<-cut(Mydata$AGENAN,
   breaks=c(0,3,5,10,max(Mydata$AGENAN)))

Mydata$TRANCHAGE<-cut(Mydata$AGENAN,
   breaks=c(-0.1,3,5,10,max(Mydata$AGENAN)))
levels(Mydata$TRANCHAGE)[1]<-"[0,3]"
```

33

Fonction which()

- Retrouver l'emplacement d'un élément d'après sa valeur

```
>which(Mydata$VALEURF> 41 |Mydata$VALEURF<35 ,
   arr.ind=TRUE)

>which(is.na(Mydata),arr.ind=TRUE) #pour repérer les valeurs
   manquantes
```

Nb:arr.ind pour array indices, ceci permet de connaitre les couples ligne x colonne

34

Identifier les données aberrantes

```
>Mydata[6,8]<-3700
>boxplot(Mydata$VALEURF)
>result<-boxplot(Mydata$VALEURF) #affecte les
   résultats de boxplot dans un objet
>Aberrant<-result$out # identifie les valeurs aberrantes
>Aberrant #voir les valeurs aberrantes
>which(Mydata[,  "VALEURF"]%in%Aberrant)
#pour connaitre le numéro des lignes avec points aberrants
```

35

Fonction edit()

- **Le contenu du fichier peut être modifié**

```
>edit(Mydata)
```

Confirmer les modifications

```
>Mydatamodif<-edit(Mydata)
```

- **Ou utiliser directement la fonction fix()**

```
>fix(Mydata) #prend en compte les modifications
```

36

Enregistrer des données

- write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ", eol = "\n", na = "NA", dec = ".", row.names = TRUE, col.names = TRUE, qmethod = c("escape", "double"))
- write(x,file = "")
- write.csv () ou write.csv2()

- write.csv2(Mydata,file= "Cours2 .csv ")

- save(Mydata, file = "Mydata.Rdata")

37

2° séance

FONCTIONS SUPPLÉMENTAIRES À DÉCOUVRIR

38

Fonction attach()

- Permet d'écrire les noms des variables sans le nom du fichier $
- Charger avant le package « utils »

```
>library(utils)
>attach(Mydata)
```

Et maintenant au lieu d'écrire

> mean(MyData$AGEAN)

Je peux écrire

>mean(AGEAN)

Fonction detach()

39

Supprimer les données manquantes

- Supprimer les enregistrement ayant des données manquantes de tout le tableau

```
>MyData.8<-na.omit(Mydata)
   #nouvelle table sans les données manquantes
>dim(Mydata.8)
>dim(Mydata)
   #Pour comparer les deux tables de données
```

40

Rediriger les sorties textes vers un fichier – Fonction sink()

Le fichier texte est créé dans le répertoire de travail défini au début

```
>sink("analyse.txt", append=TRUE)
#append= TRUE permet d'écrire dans le fichier au fur et à mesure
```

Pour commenter le fichier texte

```
>getwd()
>cat("La liste des objets imprimés le : ",date(),"\n")
#la fonction cat() permet d'inscrire des commentaires avant les analyses
>ls()
```

Pour terminer la redirection

```
>sink()
```

41

Lire un fichier MS-Excel

- **On utilisera la bibliothèque de fonctions « xlsReadWrite »**

```
library(xlsReadWrite)
>read.xls(file.choose(),colNames= TRUE, sheet= 1,
   from= 1, rowNames=T)
```

42

Lire un fichier Stata

- **On utilisera la bibliothèque de fonctions « foreign »**

```
library(foreign)
read.dta(file.choose())
```

43

Fonction rbind()

- Ajouter un enregistrement (une ligne)

```
>rbind(Mydata, c( , , , , , ))
```

- Concaténer deux tables

NB: il faut que les noms de variables soient identiques

```
>rbind(Dataframe1,Dataframe2)
```

44

Fonction cbind()

- Ajouter une variable (une colonne)

>cbind(Mydata, c(, , , , ,))

- Concaténer deux tables par colonne

>cbind(Dataframe1,Dataframe2)

NB: cbind vérifie le nom des lignes et garde celle de la première table

45

Fonction merge()

- Concaténer deux tables par colonne avec une clef

  merge(Dataframe1,Dataframe2, by="NUMIDENT")

Exemple:

```
Numident<-paste(c(letters[1:10]),c(1:10),sep="-")
Age<-c(5,4,6,3,5,4,7,3,4,5)
Poids<-seq(5,7.25,by=0.25)
Dataframe1<-data.frame(Numident,Age)
Dataframe1
Dataframe2<-data.frame(Numident,Poids)
Dataframe2
Dataframe3<-merge(Dataframe1,Dataframe2,by="Numident")
Dataframe3
names(Dataframe3)<-c("NUMIDENT","AGE","POIDS")
Dataframe3
```

46