TP R : Imputation multiple

C. Preda / V. Vandewalle

6 Mars 2018

Objectif du TP

L'objectif de ce TP est de comprendre l'imputation multiple à travers l'algorithme MICE. On utilisera le package **mice** et notamment la fonction *mice*. Nous allons illustrer les principales fonctionnalités du package à l'aide de la base de données `airquality` dans le contexte de la régression linéaire.

Avant tout

Installer le package **mice** et jetez un coup d'oeil sur la fonction *mice*

```
install.packages("mice") # installe le package mice
library(mice)            # charge la librairie mice
help(mice)               # donnee des informations sur mice
```

Les données

Chargement de la base de données `airquality` dans R:

```
d = data.frame(airquality)  #c'est une base de donnée pré-installée avec R
str(d)                      # affiche la structure (le type) des données
```

```
'data.frame':   153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
 $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
 $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
 $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

On déclare les variables `Month` et `Day` qualitatives :

```
d$Month = as.factor(d$Month)
d$Day = as.factor(d$Day)
```

Statistiques univariées des données :

```
summary(d)
```

Ozone	Solar.R	Wind	Temp	Month	Day
Min. : 1.00	Min. : 7.0	Min. : 1.700	Min. :56.00	5:31	1 : 5
1st Qu.: 18.00	1st Qu.:115.8	1st Qu.: 7.400	1st Qu.:72.00	6:30	2 : 5
Median : 31.50	Median :205.0	Median : 9.700	Median :79.00	7:31	3 : 5
Mean : 42.13	Mean :185.9	Mean : 9.958	Mean :77.88	8:31	4 : 5
3rd Qu.: 63.25	3rd Qu.:258.8	3rd Qu.:11.500	3rd Qu.:85.00	9:30	5 : 5
Max. :168.00	Max. :334.0	Max. :20.700	Max. :97.00	NA	6 : 5
NA's :37	NA's :7	NA	NA	NA	(Other):123

Les variables `Ozone` et `Solar.R` présentent des valeurs manquantes.

Les données manquantes

Voici les profils des valeurs manquantes :

```
library(mice)
```

```
md.pattern(d)
```

```
    Wind Temp Month Day Solar.R Ozone
111    1    1     1   1       1     1  0
 35    1    1     1   1       1     0  1
  5    1    1     1   1       0     1  1
  2    1    1     1   1       0     0  2
       0    0     0   0       7    37 44
```

Ce tableau se lit ainsi (on lit d'abord les marges extremes - gauche et droite - du tableau) : il y a 111 observations ('individus') avec 0 données manquantes. Il y a 35 observations avec une valeur manquante pour la variable `Ozone`. Il y a 5 observations avec une valeur manquante pour la variable `Solar.R`. Il y a 2 observations avec 2 valeurs manquantes (ces deux valeurs correspondent aux variables `Solar.R` et `Ozone`). Il y a 44 données manqunates en tout dont 37 pour la variable `Ozone` et 7 pour la variable `Solar.R`

Un profil (pattern) de données manquantes est donc un vecteur avec les elements 0 ou 1 avec 0 = donnée manquante, 1 = donnée présente.

MCAR, MAR ou MNAR

Afin de mettre un modèle sur les données manquantes, les croisements deux a deux des variables est utile. Le package **VIM** et sa fonction *marginplot* fait cette chose très bien :

```
library(VIM)
```

Par exemple :

```
marginplot(airquality[, c("Ozone","Wind")], col = mdc(c("obs", "mis")), cex = 1.2, cex.lab = 1.2,pch=19)
```

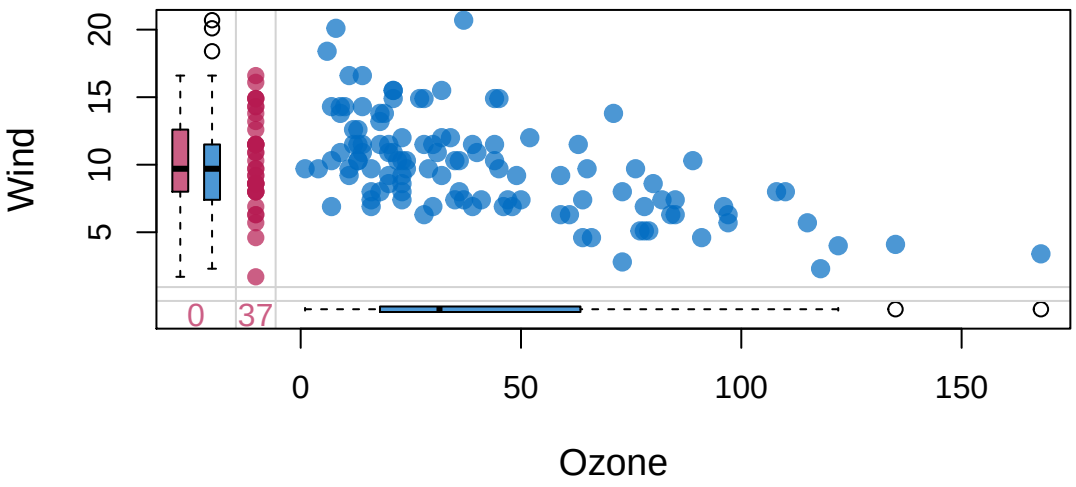


Figure 1: Quel modèle pour les données manquantes ?

ou encore, quand les deux variables ont des valeurs manquantes :

```
marginplot(airquality[, c("Ozone","Solar.R")], col = mdc(c("obs", "mis")), cex = 1.2, cex.lab = 1.2,pch=
```

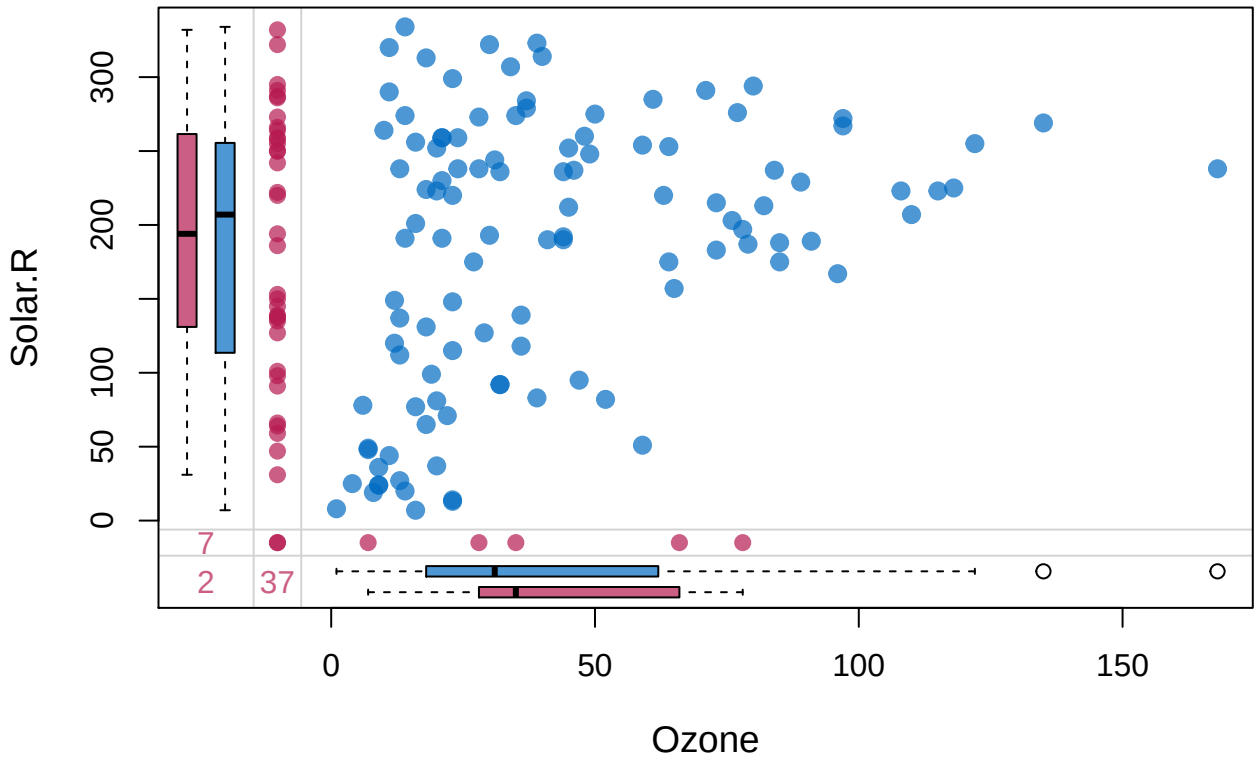


Figure 2: Quel modèle pour les données manquantes ?

Imputation multiple des données manquantes avec MICE

On réalise l'imputation afin de pouvoir réaliser un modèle de régression linéaire avec comme variable réponse `Ozone`. Puisque les variables qualitatives (mois et jour du mois) ne sont pas pertinantes (point de vue "metier"), on les considère pas dans le processus d'imputation :

```
d = d[,c(1:4)]
dm  = mice(d, m = 5, seed=10, print = FALSE, maxit = 50)
summary(dm)
```

```
Multiply imputed data set
Call:
mice(data = d, m = 5, maxit = 50, printFlag = FALSE, seed = 10)
Number of multiple imputations:  5
Missing cells per column:
  Ozone Solar.R    Wind    Temp
     37       7       0       0
Imputation methods:
  Ozone Solar.R    Wind    Temp
  "pmm"   "pmm"      ""      ""
VisitSequence:
  Ozone Solar.R
      1       2
PredictorMatrix:
```

```
        Ozone Solar.R Wind Temp
Ozone       0       1    1    1
Solar.R     1       0    1    1
Wind        0       0    0    0
Temp        0       0    0    0
Random generator seed value:  10
```

Explorer l'objet `dm` (avec la fonction `str`) et regarder la composante `imp` : on a les 5 imputations pour chaque des valeurs manquantes.

```
print(dm$imp)
```

Jeux de données completes apres imputation des données manquantes.

On utilisera la fonction *complete*. Par exemple, le premier jeu de données complétes on l'obtient avec :

```
d1 = complete(dm,1)
```

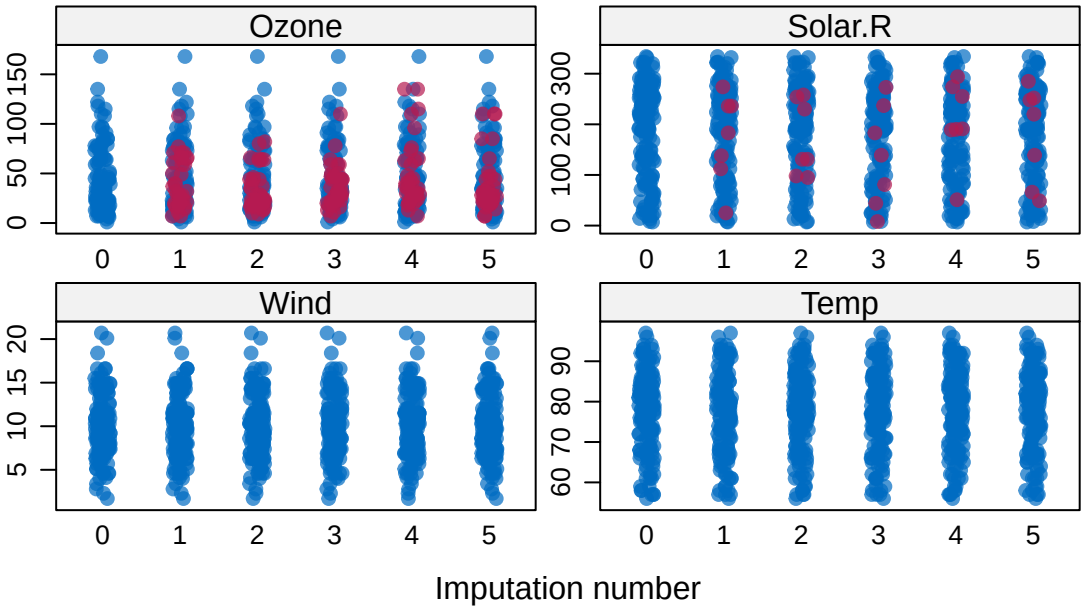
et avec `complete(dm,2)` le deuxième jeu, et ainsi de suite jusqu'au 5-eme jeu: `complete(dm,5)`

On pourra comparer le jeu de données d1 à la base initiale à travers les statisiques univariés :

```
summary(d1)
```

Représentation graphique des données imputées

```
library(lattice)
stripplot(dm, pch = 20, cex = 1.2)
```




```
xyplot(dm, Ozone ~ Solar.R|.imp, pch = 20, cex = 1.4)
```

On peut aussi comparer les distributions avant et apres imputations:

```
par(mfrow=c(1,2))
hist(d[,1], col="lightblue", probability = TRUE, xlab = "Ozone", main = "Ozone")
hist(d1[,1], density = 2, add= TRUE, border = "red", probability = TRUE, col ="red")
```

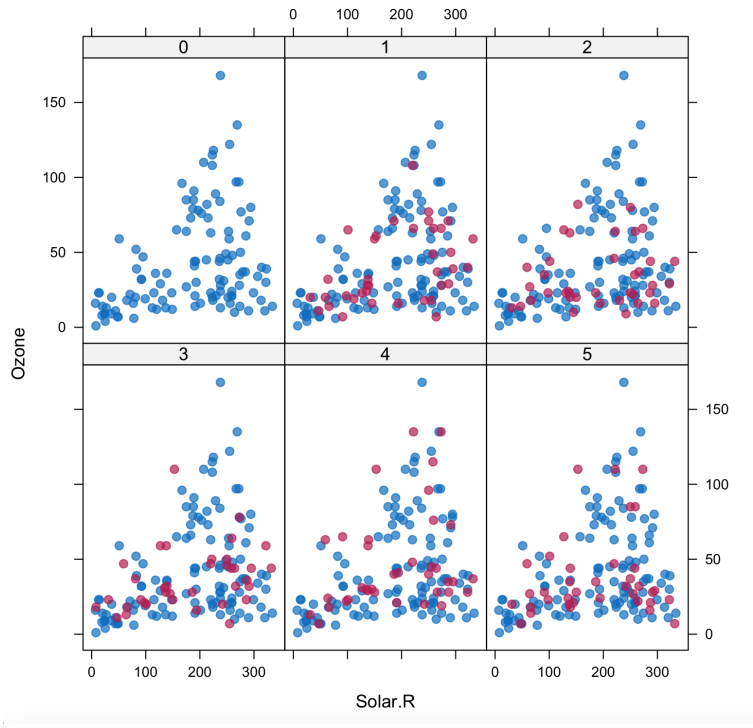
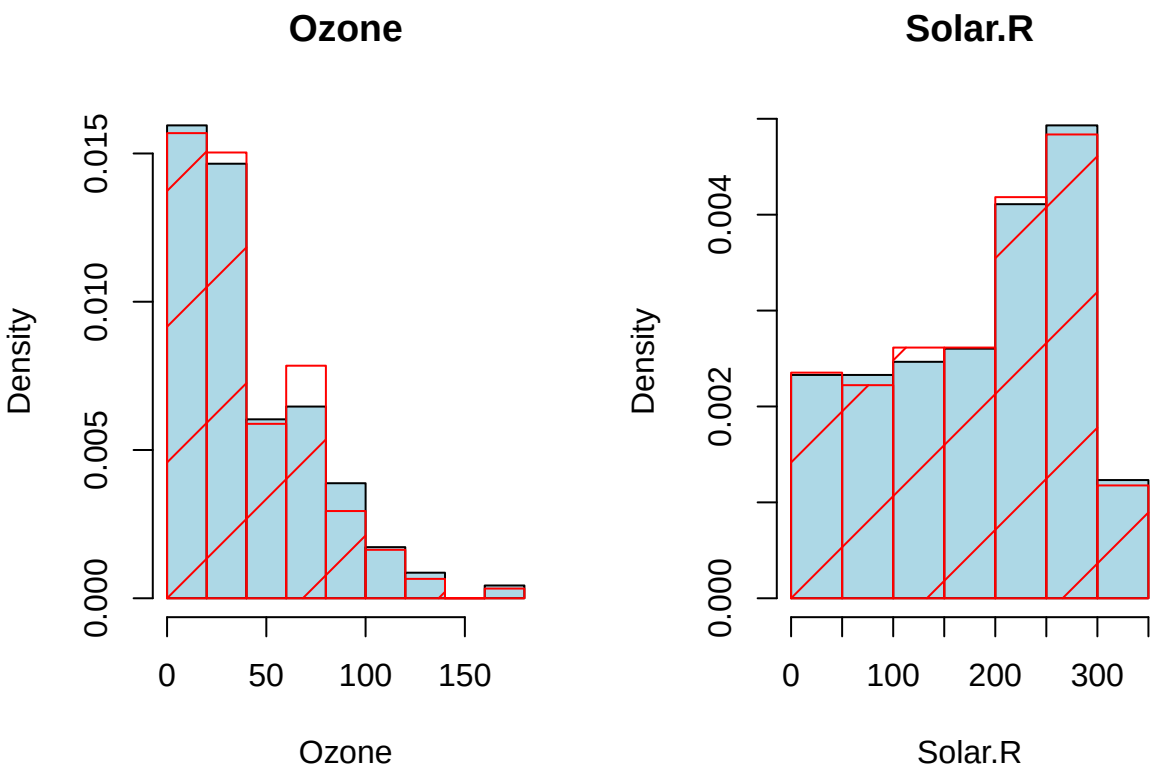


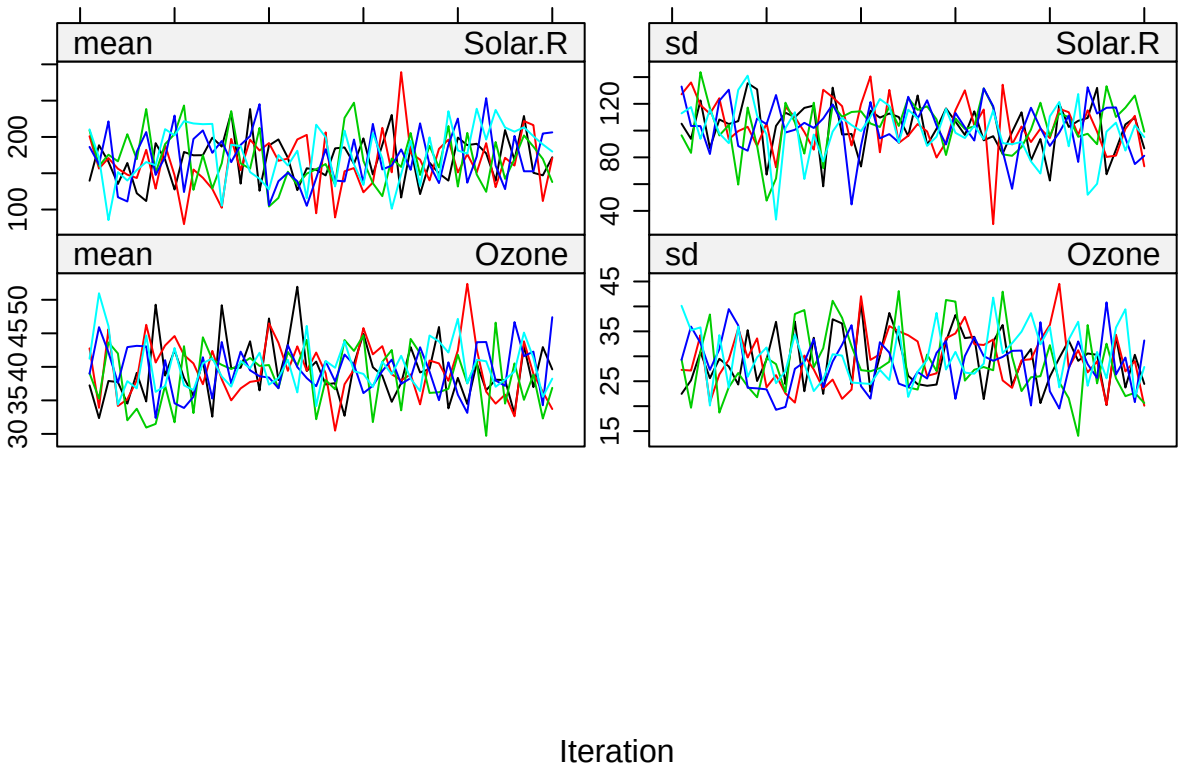
Figure 3: Données manquantes et imputations

```
hist(d[,2], col="lightblue", probability = TRUE, xlab = "Solar.R", main = "Solar.R")
hist(d1[,2], density = 2, add= TRUE, border = "red", probability = TRUE, col ="red")
```



Pour chaque jeu de données parmi les 5, on peut regader l'evolution de la moyenne et l'ecart-type lors du processus iteratif.

```
plot(dm, c("Solar.R", "Ozone"))
```



Regression linéaire dans le contexte de l'imputation multiple

```
fit <- with(dm, lm(Ozone ~ Solar.R + Wind + Temp))
```

Résultats sous forme de liste :

```
print(fit)
```

Examiner les élements l'objet `fit` :

```
print(fit[1])
print(fit[2])
print(fit[3])
print(fit[4])
```

fit[4] contient les 5 jeux de coefficients. En voici le premier :

```
print(fit[4]$analyses[1])
```

```
[[1]]

Call:
lm(formula = Ozone ~ Solar.R + Wind + Temp)

Coefficients:
(Intercept)      Solar.R         Wind         Temp
  -63.56542      0.08058     -2.78207      1.51331
```

Agréger les résulatats : la phase de "pooling" :

```
fitpool = pool(fit)
summary(fitpool)
```

```
                    est          se         t        df     Pr(>|t|)
(Intercept) -58.63539376 21.20230072 -2.765520  58.98914 7.574954e-03
Solar.R       0.06894261  0.02228545  3.093615  53.97258 3.129438e-03
Wind         -3.01010955  0.56779862 -5.301368 121.62826 5.215342e-07
Temp          1.50536786  0.24351214  6.181901  43.22603 1.948854e-07
                   lo 95       hi 95 nmis        fmi     lambda
(Intercept) -101.06126297 -16.2095245   NA 0.21177816 0.18549934
Solar.R        0.02426245   0.1136228    7 0.22831739 0.20024220
Wind          -4.13415804  -1.8860611    0 0.07731531 0.06226683
Temp           1.01435309   1.9963826    0 0.27132062 0.23836810
```

La statistique `fmi` represent la part (proportion) de variance de l'estimateur due à l'imputation. Une valeur trop grande montre que l'imputation est probablement inappropriée dans ce cas. Une valeur < 0.3 est généralement tolerée.

De manière plus synthétique :

```
fitpool$qhat
```

```
  (Intercept)    Solar.R      Wind     Temp
1   -63.56542 0.08057929 -2.782067 1.513307
2   -52.72874 0.06352161 -3.099659 1.437012
3   -47.79415 0.06878361 -3.059093 1.367108
4   -68.50701 0.07452613 -3.039255 1.646708
5   -60.58165 0.05730240 -3.070473 1.562704
```

```
fitpool$qba
```

```
 (Intercept)      Solar.R         Wind         Temp
-58.63539376   0.06894261  -3.01010955   1.50536786
```

A vous maintenant!

Note : Les conclusions et les resultats de votre analyse seront rédigés sous la forme d'un document pdf.

Chargez le fichier des données : http://math.univ-lille1.fr/~preda/GIS4/tibetan_skull_missing.csv

Il s'agit d'un jeu de données sur de 32 cranes de deux types (A = mongolian et B = Indian) pour lesquels on connait 5 caractéristiques numériques. Le but est de construire un modèle prédictif du type de crane à partir des caractéristiques numériques (X1 - X5) à l'aide de la régression logistique binaire.

L'analyse doit contenir:

1. statistiques univariées descriptives pour toutes les variables.
2. statistiques bivariées: étudier le lien entre la variable type et les variables explicatives, prises une par une. Réaliser le test statistique de comparaison des deux groupes et illustraire graphiquement les couples (Xi, type), i=1, . . . , 5.
3. Réaliser le régression logistique en présence des données manquantes. Ecrivez le modèle obtenu.
4. Tracer la courbe ROC du modèle logistique obtenu. Quelle est le taux de mal-classés en considérant le seuil de décision = 0.5?