```r
# Installer les packages nécessaires s'ils ne sont pas déjà installés
packages <- c("FactoMineR", "missMDA", "mice", "VIM", "ggplot2", "gridExtra", "missForest", "naniar", "Amelia", "bnstruct", "lme4", "MCMCglmm", "glmnet")
new_packages <- packages[!(packages %in% installed.packages()[,"Package"])]

if (length(new_packages)) {
  install.packages(new_packages, dependencies = TRUE)
}

# Charger les bibliothèques
lapply(packages, library, character.only = TRUE)

```

Afficher la sortie masquée

```r
library(lme4)          # for lmer
library(MCMCglmm)      # for MCMCglmm
library(Amelia)        # for amelia
library(ggplot2)       # for plotting
library(gridExtra)     # for arranging plots
library(naniar)        # for missing data analysis
library(VIM)
library(naniar)
library(UpSetR)

```

Donnée générée

Dans cette cellule, après avoir lu l'article, j'ai recherché les algorithmes des méthodes de traitement des données manquantes mentionnées sur le site de Julie Josse et je les ai d'abord testés avec des données générées.

```r
# Générer des données avec valeurs manquantes
set.seed(123)
data <- data.frame(
  X1 = c(rnorm(10, mean = 5), rep(NA, 5)),
  X2 = c(rnorm(10, mean = 3), rep(NA, 5)),
  X3 = c(rnorm(10, mean = 8), rep(NA, 5)),
  Group = factor(rep(1:3, each = 5))
)
print("Données avec valeurs manquantes :")
print(data)
```

```
[1] "Données avec valeurs manquantes :"
         X1       X2       X3 Group
1  4.439524 4.224082 6.932176     1
2  4.769823 3.359814 7.782025     1
3  6.558708 3.400771 6.973996     1
4  5.070508 3.110683 7.271109     1
5  5.129288 2.444159 7.374961     1
6  6.715065 4.786913 6.313307     2
7  5.460916 3.497850 8.837787     2
8  3.734939 1.033383 8.153373     2
9  4.313147 3.701356 6.861863     2
10 4.554338 2.527209 9.253815     2
11       NA       NA       NA     3
12       NA       NA       NA     3
13       NA       NA       NA     3
14       NA       NA       NA     3
15       NA       NA       NA     3
```

SECTION 1: IMPUTATION SIMPLE ET MULTIPLE

```r
# 1. Imputation Simple (Moyenne)
data_imputed_mean <- data
for(i in 1:3) {  # Sur les colonnes numériques uniquement
  data_imputed_mean[is.na(data_imputed_mean[, i]), i] <- mean(data_imputed_mean[, i], na.rm = TRUE)
}
print("Imputation par la moyenne :")
print(data_imputed_mean)
```

```
[1] "Imputation par la moyenne :"
         X1       X2       X3 Group
1  4.439524 4.224082 6.932176     1
2  4.769823 3.359814 7.782025     1
3  6.558708 3.400771 6.973996     1
4  5.070508 3.110683 7.271109     1
5  5.129288 2.444159 7.374961     1
6  6.715065 4.786913 6.313307     2
7  5.460916 3.497850 8.837787     2
8  3.734939 1.033383 8.153373     2
9  4.313147 3.701356 6.861863     2
10 4.554338 2.527209 9.253815     2
11 5.074626 3.208622 7.575441     3
12 5.074626 3.208622 7.575441     3
13 5.074626 3.208622 7.575441     3
14 5.074626 3.208622 7.575441     3
15 5.074626 3.208622 7.575441     3
```

```r
# 2. Imputation Simple (Médiane)
data_imputed_median <- data
for(i in 1:3) {  # Sur les colonnes numériques uniquement
  data_imputed_median[is.na(data_imputed_median[, i]), i] <- median(data_imputed_median[, i], na.rm = TRUE)
}
print("Imputation par la médiane :")
print(data_imputed_median)
```

```
[1] "Imputation par la médiane :"
         X1       X2       X3 Group
1  4.439524 4.224082 6.932176     1
2  4.769823 3.359814 7.782025     1
3  6.558708 3.400771 6.973996     1
4  5.070508 3.110683 7.271109     1
5  5.129288 2.444159 7.374961     1
6  6.715065 4.786913 6.313307     2
7  5.460916 3.497850 8.837787     2
8  3.734939 1.033383 8.153373     2
9  4.313147 3.701356 6.861863     2
10 4.554338 2.527209 9.253815     2
11 4.920165 3.380293 7.323035     3
12 4.920165 3.380293 7.323035     3
13 4.920165 3.380293 7.323035     3
14 4.920165 3.380293 7.323035     3
15 4.920165 3.380293 7.323035     3
```

```r
# 3. Imputation Multiple avec 'mice'

data_imputed_mice <- mice(data[,1:3], m = 5, maxit = 50, method = 'pmm', seed = 500)
data_imputed_mice_complete <- mice::complete(data_imputed_mice, 1)
print("Imputation multiple avec 'mice' :")
print(data_imputed_mice_complete)
```

```
      27   5  X1  X2  X3
      28   1  X1  X2  X3
      28   2  X1  X2  X3
      28   3  X1  X2  X3
      28   4  X1  X2  X3
      28   5  X1  X2  X3
      29   1  X1  X2  X3
      29   2  X1  X2  X3
      29   3  X1  X2  X3
      29   4  X1  X2  X3
      29   5  X1  X2  X3
      30   1  X1  X2  X3
      30   2  X1  X2  X3
      30   3  X1  X2  X3
```

```r
# 4. PCA avec Valeurs Manquantes (missMDA)
nb_comp <- estim_ncpPCA(data[,1:3], ncp.max = 5)
pca_result <- imputePCA(data[,1:3], ncp = nb_comp$ncp)
print("Données après PCA avec imputation :")
print(pca_result$completeObs)
```

```
[1] "Données après PCA avec imputation :"
            X1       X2       X3
 [1,] 4.439524 4.224082 6.932176
 [2,] 4.769823 3.359814 7.782025
 [3,] 6.558708 3.400771 6.973996
 [4,] 5.070508 3.110683 7.271109
 [5,] 5.129288 2.444159 7.374961
 [6,] 6.715065 4.786913 6.313307
 [7,] 5.460916 3.497850 8.837787
 [8,] 3.734939 1.033383 8.153373
 [9,] 4.313147 3.701356 6.861863
[10,] 4.554338 2.527209 9.253815
[11,] 5.074626 3.208622 7.575441
[12,] 5.074626 3.208622 7.575441
[13,] 5.074626 3.208622 7.575441
[14,] 5.074626 3.208622 7.575441
[15,] 5.074626 3.208622 7.575441
```

```r

# 5. Imputation par 'missForest'
data_imputed_mf <- missForest(data[,1:3])$ximp
print("Données après imputation avec missForest :")
print(data_imputed_mf)


```

```
[1] "Données après imputation avec missForest :"
         X1       X2       X3
1  4.439524 4.224082 6.932176
2  4.769823 3.359814 7.782025
3  6.558708 3.400771 6.973996
4  5.070508 3.110683 7.271109
5  5.129288 2.444159 7.374961
6  6.715065 4.786913 6.313307
7  5.460916 3.497850 8.837787
8  3.734939 1.033383 8.153373
9  4.313147 3.701356 6.861863
10 4.554338 2.527209 9.253815
11 4.993705 2.857675 7.549812
12 4.993705 2.857675 7.549812
13 4.993705 2.857675 7.549812
14 4.993705 2.857675 7.549812
15 4.993705 2.857675 7.549812
```

SECTION 2: IMPUTATIONS AVANCÉES

```r

# 1. Imputation KNN
data_knn <- kNN(data[,1:3], k = 3)
print("Imputation KNN :")
print(data_knn)
```

```
[1] "Imputation KNN :"
         X1       X2       X3 X1_imp X2_imp X3_imp
1  4.439524 4.224082 6.932176  FALSE  FALSE  FALSE
2  4.769823 3.359814 7.782025  FALSE  FALSE  FALSE
3  6.558708 3.400771 6.973996  FALSE  FALSE  FALSE
4  5.070508 3.110683 7.271109  FALSE  FALSE  FALSE
5  5.129288 2.444159 7.374961  FALSE  FALSE  FALSE
6  6.715065 4.786913 6.313307  FALSE  FALSE  FALSE
7  5.460916 3.497850 8.837787  FALSE  FALSE  FALSE
8  3.734939 1.033383 8.153373  FALSE  FALSE  FALSE
9  4.313147 3.701356 6.861863  FALSE  FALSE  FALSE
10 4.554338 2.527209 9.253815  FALSE  FALSE  FALSE
11 4.769823 3.359814 7.271109   TRUE   TRUE   TRUE
12 4.769823 3.359814 7.271109   TRUE   TRUE   TRUE
13 4.769823 3.359814 7.271109   TRUE   TRUE   TRUE
14 4.769823 3.359814 7.271109   TRUE   TRUE   TRUE
15 4.769823 3.359814 7.271109   TRUE   TRUE   TRUE
```

```r
# 2. Imputation Expectation-Maximization (EM)
data_em <- missMDA::imputePCA(data[,1:3], method = "EM")
print("Imputation EM :")
print(data_em$completeObs)

```

```
[1] "Imputation EM :"
            X1       X2       X3
 [1,] 4.439524 4.224082 6.932176
 [2,] 4.769823 3.359814 7.782025
 [3,] 6.558708 3.400771 6.973996
 [4,] 5.070508 3.110683 7.271109
 [5,] 5.129288 2.444159 7.374961
 [6,] 6.715065 4.786913 6.313307
 [7,] 5.460916 3.497850 8.837787
 [8,] 3.734939 1.033383 8.153373
 [9,] 4.313147 3.701356 6.861863
[10,] 4.554338 2.527209 9.253815
[11,] 5.074626 3.208622 7.575441
[12,] 5.074626 3.208622 7.575441
[13,] 5.074626 3.208622 7.575441
[14,] 5.074626 3.208622 7.575441
[15,] 5.074626 3.208622 7.575441
```

```r

# 3. Imputation pour données qualitatives (MCA)
data_qual <- as.data.frame(lapply(data[,1:3], function(x) as.factor(cut(x, breaks=3))))
data_qual[is.na(data_qual)] <- NA
data_mca <- imputeMCA(data_qual, ncp = 2)
print("Imputation MCA pour données qualitatives :")
print(data_mca$completeObs)

```

```
[1] "Imputation MCA pour données qualitatives :"
          X1          X2          X3
1  (3.73,4.73] (3.54,4.79] (6.31,7.29]
2  (4.73,5.72] (2.28,3.54] (7.29,8.27]
3  (5.72,6.72] (2.28,3.54] (6.31,7.29]
4  (4.73,5.72] (2.28,3.54] (6.31,7.29]
5  (4.73,5.72] (2.28,3.54] (7.29,8.27]
6  (5.72,6.72] (3.54,4.79] (6.31,7.29]
7  (4.73,5.72] (2.28,3.54] (8.27,9.26]
8  (3.73,4.73] (1.03,2.28] (7.29,8.27]
9  (3.73,4.73] (3.54,4.79] (6.31,7.29]
10 (3.73,4.73] (2.28,3.54] (8.27,9.26]
11 (3.73,4.73] (2.28,3.54] (6.31,7.29]
12 (3.73,4.73] (2.28,3.54] (6.31,7.29]
13 (3.73,4.73] (2.28,3.54] (6.31,7.29]
14 (3.73,4.73] (2.28,3.54] (6.31,7.29]
15 (3.73,4.73] (2.28,3.54] (6.31,7.29]
```

SECTION 3: IMPUTATION PAR MODÈLES STATISTIQUES

```r
# 1. Imputation par Modèles Mixtes
data_complete <- data[!(data$Group %in% c("levels_to_exclude")),]
fit_mixed <- lmer(X1 ~ X2 + (1 | Group), data = data_complete, REML = FALSE)
data$Group <- factor(data$Group, levels = levels(data_complete$Group))
data$X1_imputed <- ifelse(is.na(data$X1), predict(fit_mixed, newdata = data, allow.new.levels = TRUE), data$X1)
print("Données après imputation par modèle mixte :")
print(data$X1_imputed)
```

```
boundary (singular) fit: see help('isSingular')

[1] "Données après imputation par modèle mixte :"
 [1] 4.439524 4.769823 6.558708 5.070508 5.129288 6.715065 5.460916 3.734939
 [9] 4.313147 4.554338       NA       NA       NA       NA       NA
```

```
# 2. Imputation par Chaînes de Markov Monte Carlo (MCMC)
data$X2 <- as.numeric(as.character(data$X2))
data$X3 <- as.numeric(as.character(data$X3))
data$X2[is.na(data$X2)] <- mean(data$X2, na.rm = TRUE)
data$X3[is.na(data$X3)] <- mean(data$X3, na.rm = TRUE)
fit_mcmc <- MCMCglmm(X1 ~ X2 + X3, random = ~Group, data = data, nitt = 13000, burnin = 3000, pr = TRUE)
data$X1_imputed_mcmc <- ifelse(is.na(data$X1), predict(fit_mcmc, newdata = data), data$X1)
print("Données après imputation avec MCMC :")
print(data$X1_imputed_mcmc)
```

```
                        MCMC iteration = 0

                        MCMC iteration = 1000

                        MCMC iteration = 2000

                        MCMC iteration = 3000

                        MCMC iteration = 4000

                        MCMC iteration = 5000

                        MCMC iteration = 6000

                        MCMC iteration = 7000

                        MCMC iteration = 8000

                        MCMC iteration = 9000

                        MCMC iteration = 10000

                        MCMC iteration = 11000

                        MCMC iteration = 12000

                        MCMC iteration = 13000
[1] "Données après imputation avec MCMC :"
 [1] 4.439524 4.769823 6.558708 5.070508 5.129288 6.715065 5.460916 3.734939
 [9] 4.313147 4.554338 5.066539 5.066539 5.066539 5.066539 5.066539
```

```

# 3. Joint Modeling avec Amelia
data_joint <- amelia(data, m = 5, idvars = "Group")$imputations[[1]]
print("Imputation avec Joint Modeling (Amelia) :")
print(data_joint)


```

```
Warning message in amcheck(x = x, m = m, idvars = numopts$idvars, priors = priors, :
"The variables (or variable with levels) X1_imputed, X1_imputed_mcmc are perfectly collinear with another variable in the data.
"
Warning message in amelia_prep(x = x, m = m, idvars = idvars, empri = empri, ts = ts, :
"You have a small number of observations, relative to the number, of variables in the imputation model.  Consider removing some variables, or reducing the order of time polynomials to reduce the number of parameters."
-- Imputation 1 --

  1  2  3  4  5  6  7

-- Imputation 2 --

  1  2  3  4  5  6  7  8  9 10

-- Imputation 3 --

  1  2  3  4  5  6  7

-- Imputation 4 --

  1  2  3  4  5  6  7  8

-- Imputation 5 --

  1  2  3  4  5  6  7

[1] "Imputation avec Joint Modeling (Amelia) :"
         X1       X2       X3 Group X1_imputed X1_imputed_mcmc
1  4.439524 4.224082 6.932176     1   4.439524        4.439524
2  4.769823 3.359814 7.782025     1   4.769823        4.769823
3  6.558708 3.400771 6.973996     1   6.558708        6.558708
4  5.070508 3.110683 7.271109     1   5.070508        5.070508
5  5.129288 2.444159 7.374961     1   5.129288        5.129288
6  6.715065 4.786913 6.313307     2   6.715065        6.715065
7  5.460916 3.497850 8.837787     2   5.460916        5.460916
8  3.734939 1.033383 8.153373     2   3.734939        3.734939
9  4.313147 3.701356 6.861863     2   4.313147        4.313147
10 4.554338 2.527209 9.253815     2   4.554338        4.554338
11 5.067279 3.208622 7.575441     3   5.067763        5.066539
12 5.067142 3.208622 7.575441     3   5.066800        5.066539
13 5.070663 3.208622 7.575441     3   5.068018        5.066539
14 5.072480 3.208622 7.575441     3   5.065280        5.066539
15 5.065396 3.208622 7.575441     3   5.065064        5.066539
```

SECTION 4: VALIDATION DE L'IMPUTATION

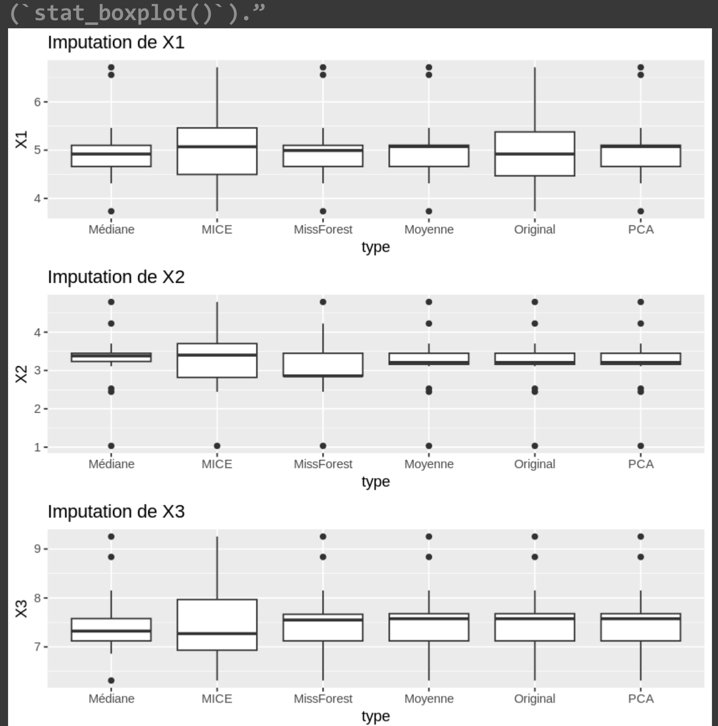
```

# Calcul de la RMSE pour évaluer la qualité de chaque méthode
rmse <- function(true, predicted) {
  sqrt(mean((true - predicted)^2, na.rm = TRUE))
}
rmse_mixed <- rmse(data$X1, data$X1_imputed)
rmse_mcmc <- rmse(data$X1, data$X1_imputed_mcmc)
rmse_lasso <- rmse(data$X1, data$X1_imputed_lasso)
print(paste("RMSE Modèle Mixte:", rmse_mixed))
print(paste("RMSE MCMC:", rmse_mcmc))
print(paste("RMSE LASSO:", rmse_lasso))

```

```
[1] "RMSE Modèle Mixte: 0"
[1] "RMSE MCMC: 0"
[1] "RMSE LASSO: NaN"
```

SECTION 5: VISUALISATION DES DONNÉES IMPUTÉES

```

data_long <- rbind(
  data.frame(type = "Original", data[,1:3]),
  data.frame(type = "Moyenne", data_imputed_mean[,1:3]),
  data.frame(type = "Médiane", data_imputed_median[,1:3]),
  data.frame(type = "MICE", data_imputed_mice_complete),
  data.frame(type = "PCA", pca_result$completeObs),
  data.frame(type = "MissForest", data_imputed_mf)
)
plot_list <- list()
for (col in names(data[,1:3])) {
  p <- ggplot(data_long, aes_string(x = 'type', y = col)) +
    geom_boxplot() + labs(title = paste("Imputation de", col))
  plot_list[[col]] <- p
}
do.call(grid.arrange, plot_list)

```

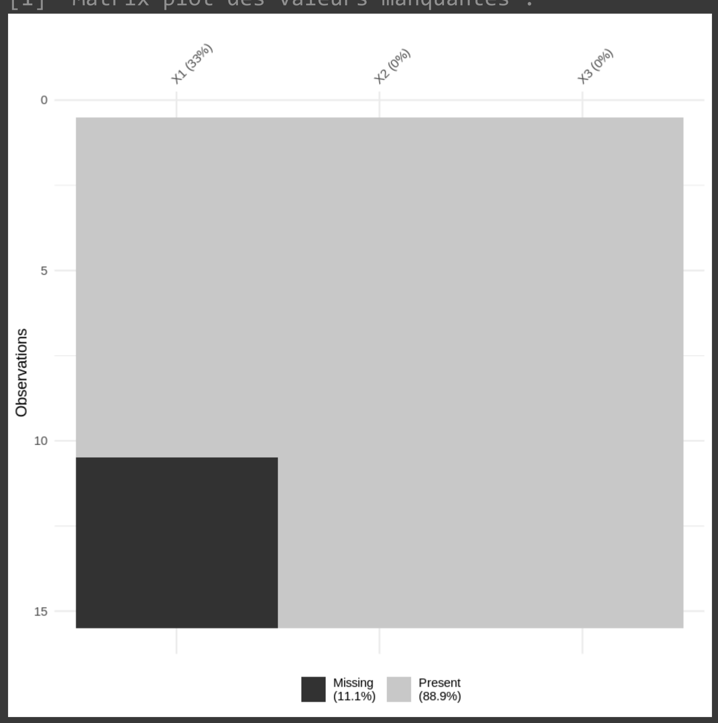
```
Warning message:
"`aes_string()` was deprecated in ggplot2 3.0.0.
ℹ Please use tidy evaluation idioms with `aes()`.
ℹ See also `vignette("ggplot2-in-packages")` for more information."
Warning message:
"Removed 5 rows containing non-finite outside the scale range
(`stat_boxplot()`)."
```



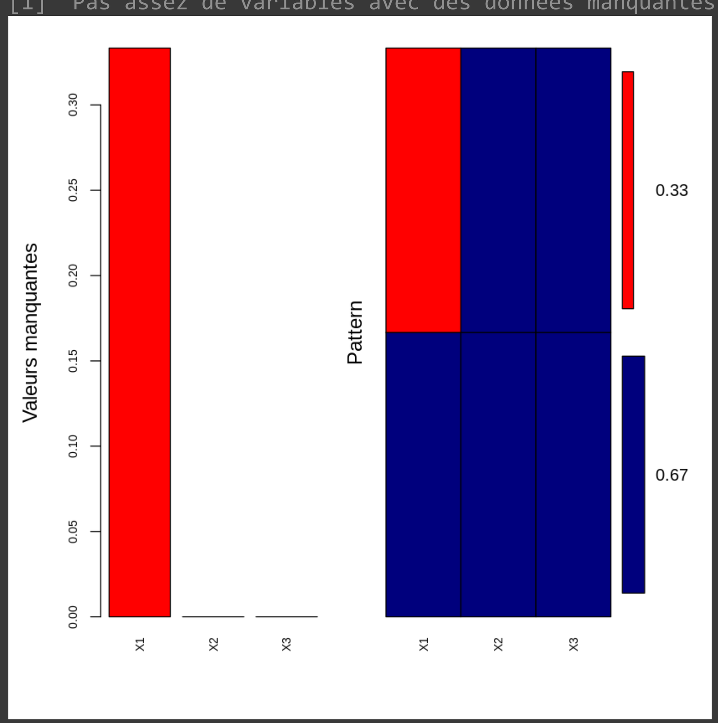
SECTION 6: ANALYSE DES VALEURS MANQUANTES

```
# Visualisation des patrons de valeurs manquantes
print("Heatmap des valeurs manquantes :")
vis_miss(data[, 1:3])

# Visualisation avec VIM (Matrix plot)
print("Matrix plot des valeurs manquantes :")
aggr(data[, 1:3], col = c("navyblue", "red"), numbers = TRUE, sortVars = TRUE,
     labels = names(data[, 1:3]), cex.axis = 0.7, gap = 3, ylab = c("Valeurs manquantes", "Pattern"))

# Cartographie des valeurs manquantes par paires de variables
if (sum(colSums(is.na(data[, 1:3])) > 0) > 1) {  # Vérifie que plus d'une variable contient des données manquantes
    print("Visualisation par paires de variables :")
    gg_miss_upset(data[, 1:3])
} else {
    print("Pas assez de variables avec des données manquantes pour créer un upset plot.")
}
```

```
[1] "Heatmap des valeurs manquantes :"
[1] "Matrix plot des valeurs manquantes :"
```

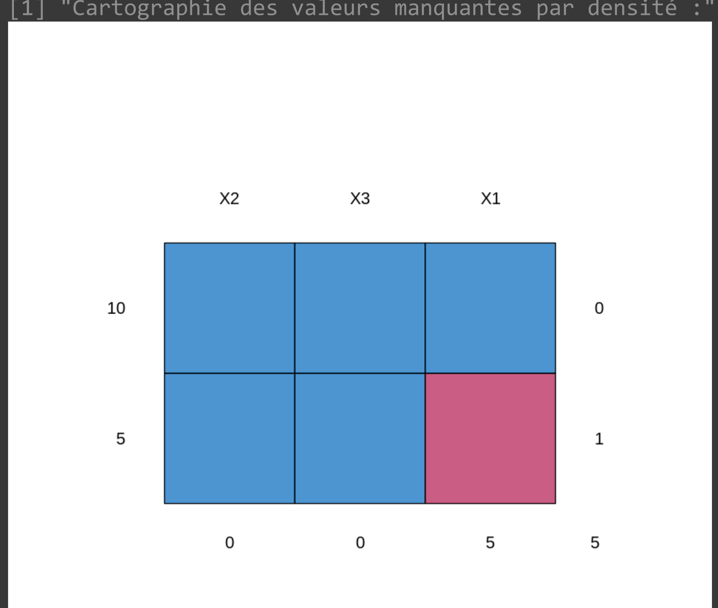


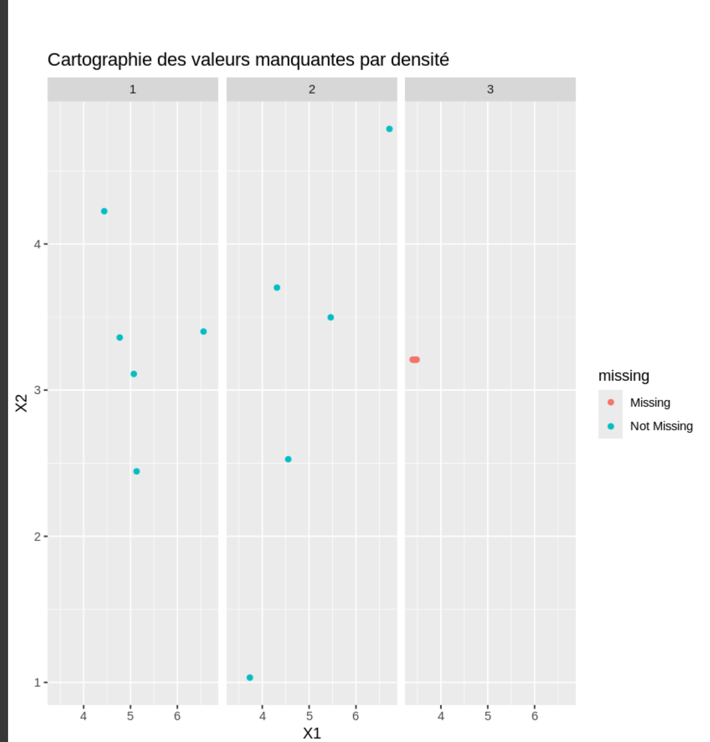

```
 Variables sorted by number of missings:
 Variable     Count
       X1 0.3333333
       X2 0.0000000
       X3 0.0000000
[1] "Pas assez de variables avec des données manquantes pour créer un upset plot."
```




```


# 2. Détection des mécanismes de valeurs manquantes

# a) Test MCAR de Little avec mice
# Installer et charger naniar si nécessaire
if (!require("naniar")) {
  install.packages("naniar", dependencies = TRUE)
  library(naniar)
}

# Utiliser mcar_test de naniar
mcar_test_result <- naniar::mcar_test(data[, 1:3])
print("Test de Little pour MCAR :")
print(mcar_test_result)


# b) Patron des valeurs manquantes avec md.pattern
print("Patron des valeurs manquantes :")
pattern <- md.pattern(data[, 1:3])
print(pattern)

# c) Analyse de la distribution des valeurs manquantes
print("Cartographie des valeurs manquantes par densité :")
ggplot(data, aes(x = X1, y = X2)) +
  geom_miss_point() +
  facet_wrap(~ Group) +
  labs(title = "Cartographie des valeurs manquantes par densité")

```

```
[1] "Test de Little pour MCAR :"
# A tibble: 1 × 4
  statistic    df p.value missing.patterns
      <dbl> <dbl>   <dbl>            <int>
1         0     2       1                2
[1] "Patron des valeurs manquantes :"
   X2 X3 X1
10  1  1  1 0
5   1  1  0 1
    0  0  5 5
[1] "Cartographie des valeurs manquantes par densité :"
```





donne reel

Dans cette cellule, jles ai testé avec des données réelles.

```
# Charger les jeux de données en R
data("airquality")
data <- airquality
print("Jeu de données airquality avec valeurs manquantes :")
print(head(data))
```

```
[1] "Jeu de données airquality avec valeurs manquantes :"
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

SECTION 1: IMPUTATION SIMPLE ET MULTIPLE

```
# 1. Imputation Simple (Moyenne)
data_imputed_mean <- data
for(i in 1:ncol(data_imputed_mean)) {  # Sur toutes les colonnes
  data_imputed_mean[is.na(data_imputed_mean[, i]), i] <- mean(data_imputed_mean[, i], na.rm = TRUE)
}
print("Imputation par la moyenne :")
print(head(data_imputed_mean))
```

```
[1] "Imputation par la moyenne :"
     Ozone  Solar.R Wind Temp Month Day
1 41.00000 190.0000  7.4   67     5   1
2 36.00000 118.0000  8.0   72     5   2
3 12.00000 149.0000 12.6   74     5   3
4 18.00000 313.0000 11.5   62     5   4
5 42.12931 185.9315 14.3   56     5   5
6 28.00000 185.9315 14.9   66     5   6
```

```
# 2. Imputation Simple (Médiane)
data_imputed_median <- data
for(i in 1:ncol(data_imputed_median)) {
  data_imputed_median[is.na(data_imputed_median[, i]), i] <- median(data_imputed_median[, i], na.rm = TRUE)
}
print("Imputation par la médiane :")
print(head(data_imputed_median))
```

```
[1] "Imputation par la médiane :"
  Ozone Solar.R Wind Temp Month Day
1  41.0     190  7.4   67     5   1
2  36.0     118  8.0   72     5   2
3  12.0     149 12.6   74     5   3
4  18.0     313 11.5   62     5   4
5  31.5     205 14.3   56     5   5
6  28.0     205 14.9   66     5   6
```

```
# 3. Imputation Multiple avec 'mice'

# Imputation multiple avec mice
data_imputed_mice <- mice(data, m = 5, maxit = 50, method = 'pmm', seed = 500)

# Obtenir le premier jeu de données imputé complet
data_imputed_mice_complete <- mice::complete(data_imputed_mice, 1)
print("Imputation multiple avec 'mice' :")
print(head(data_imputed_mice_complete))

```

```
 iter imp variable
  1   1  Ozone  Solar.R
  1   2  Ozone  Solar.R
  1   3  Ozone  Solar.R
  1   4  Ozone  Solar.R
  1   5  Ozone  Solar.R
  2   1  Ozone  Solar.R
  2   2  Ozone  Solar.R
  2   3  Ozone  Solar.R
  2   4  Ozone  Solar.R
  2   5  Ozone  Solar.R
  3   1  Ozone  Solar.R
  3   2  Ozone  Solar.R
  3   3  Ozone  Solar.R
  3   4  Ozone  Solar.R
  3   5  Ozone  Solar.R
  4   1  Ozone  Solar.R
  4   2  Ozone  Solar.R
  4   3  Ozone  Solar.R
  4   4  Ozone  Solar.R
  4   5  Ozone  Solar.R
  5   1  Ozone  Solar.R
  5   2  Ozone  Solar.R
  5   3  Ozone  Solar.R
  5   4  Ozone  Solar.R
  5   5  Ozone  Solar.R
  6   1  Ozone  Solar.R
  6   2  Ozone  Solar.R
  6   3  Ozone  Solar.R
  6   4  Ozone  Solar.R
  6   5  Ozone  Solar.R
  7   1  Ozone  Solar.R
  7   2  Ozone  Solar.R
  7   3  Ozone  Solar.R
  7   4  Ozone  Solar.R
  7   5  Ozone  Solar.R
  8   1  Ozone  Solar.R
  8   2  Ozone  Solar.R
  8   3  Ozone  Solar.R
  8   4  Ozone  Solar.R
  8   5  Ozone  Solar.R
  9   1  Ozone  Solar.R
  9   2  Ozone  Solar.R
  9   3  Ozone  Solar.R
  9   4  Ozone  Solar.R
  9   5  Ozone  Solar.R
  10   1  Ozone  Solar.R
  10   2  Ozone  Solar.R
  10   3  Ozone  Solar.R
  10   4  Ozone  Solar.R
  10   5  Ozone  Solar.R
  11   1  Ozone  Solar.R
  11   2  Ozone  Solar.R
  11   3  Ozone  Solar.R
  11   4  Ozone  Solar.R
  11   5  Ozone  Solar.R
  12   1  Ozone  Solar.R
```

```
# 4. PCA avec Valeurs Manquantes (missMDA)
nb_comp <- estim_ncpPCA(data, ncp.max = 5)
pca_result <- imputePCA(data, ncp = nb_comp$ncp)
print("Données après PCA avec imputation :")
print(head(pca_result$completeObs))
```

```
[1] "Données après PCA avec imputation :"
        Ozone  Solar.R Wind Temp Month Day
[1,] 41.00000 190.0000  7.4   67     5   1
[2,] 36.00000 118.0000  8.0   72     5   2
[3,] 12.00000 149.0000 12.6   74     5   3
[4,] 18.00000 313.0000 11.5   62     5   4
[5,] 42.12931 185.9315 14.3   56     5   5
[6,] 28.00000 185.9315 14.9   66     5   6
```

```
# 5. Imputation par 'missForest'
data_imputed_mf <- missForest(data)$ximp
print("Données après imputation avec missForest :")
print(head(data_imputed_mf))
```

```
[1] "Données après imputation avec missForest :"
     Ozone  Solar.R Wind Temp Month Day
1 41.00000 190.0000  7.4   67     5   1
2 36.00000 118.0000  8.0   72     5   2
3 12.00000 149.0000 12.6   74     5   3
4 18.00000 313.0000 11.5   62     5   4
5 18.20667 147.2600 14.3   56     5   5
6 28.00000 261.1633 14.9   66     5   6
```

SECTION 2: IMPUTATIONS AVANCÉES

```
# 1. Imputation KNN
data_knn <- kNN(data, k = 3)
print("Imputation KNN :")
print(head(data_knn))
```

```
[1] "Imputation KNN :"
  Ozone Solar.R Wind Temp Month Day Ozone_imp Solar.R_imp Wind_imp Temp_imp
1    41     190  7.4   67     5   1     FALSE       FALSE    FALSE    FALSE
2    36     118  8.0   72     5   2     FALSE       FALSE    FALSE    FALSE
3    12     149 12.6   74     5   3     FALSE       FALSE    FALSE    FALSE
4    18     313 11.5   62     5   4     FALSE       FALSE    FALSE    FALSE
5    18      99 14.3   56     5   5      TRUE        TRUE    FALSE    FALSE
6    28     299 14.9   66     5   6     FALSE        TRUE    FALSE    FALSE
  Month_imp Day_imp
```

```
1     FALSE   FALSE
2     FALSE   FALSE
3     FALSE   FALSE
4     FALSE   FALSE
5     FALSE   FALSE
6     FALSE   FALSE
```

```
# 2. Imputation Expectation-Maximization (EM)
data_em <- missMDA::imputePCA(data, method = "EM")
print("Imputation EM :")
print(head(data_em$completeObs))
```

```
[1] "Imputation EM :"
         Ozone  Solar.R Wind Temp Month Day
[1,] 41.000000 190.0000  7.4   67     5   1
[2,] 36.000000 118.0000  8.0   72     5   2
[3,] 12.000000 149.0000 12.6   74     5   3
[4,] 18.000000 313.0000 11.5   62     5   4
[5,] -5.172391 244.6936 14.3   56     5   5
[6,] 28.000000 288.3683 14.9   66     5   6
```

```

# 3. Imputation pour données qualitatives (MCA)
data_qual <- as.data.frame(lapply(data[,1:3], function(x) as.factor(cut(x, breaks=3))))
data_qual[is.na(data_qual)] <- NA
data_mca <- imputeMCA(data_qual, ncp = 2)
print("Imputation MCA pour données qualitatives :")
print(data_mca$completeObs)
```

```
[1] "Imputation MCA pour données qualitatives :"
           Ozone    Solar.R        Wind
1   (0.833,56.7]  (116,225] (1.68,8.03]
2   (0.833,56.7]  (116,225] (1.68,8.03]
3   (0.833,56.7]  (116,225] (8.03,14.4]
4   (0.833,56.7]  (225,334] (8.03,14.4]
5   (0.833,56.7]  (225,334] (8.03,14.4]
6   (0.833,56.7]  (225,334] (14.4,20.7]
7   (0.833,56.7]  (225,334] (8.03,14.4]
8   (0.833,56.7] (6.67,116] (8.03,14.4]
9   (0.833,56.7] (6.67,116] (14.4,20.7]
10  (0.833,56.7]  (116,225] (8.03,14.4]
11  (0.833,56.7]  (116,225] (1.68,8.03]
12  (0.833,56.7]  (225,334] (8.03,14.4]
13  (0.833,56.7]  (225,334] (8.03,14.4]
14  (0.833,56.7]  (225,334] (8.03,14.4]
15  (0.833,56.7] (6.67,116] (8.03,14.4]
16  (0.833,56.7]  (225,334] (8.03,14.4]
17  (0.833,56.7]  (225,334] (8.03,14.4]
18  (0.833,56.7] (6.67,116] (14.4,20.7]
19  (0.833,56.7]  (225,334] (8.03,14.4]
20  (0.833,56.7] (6.67,116] (8.03,14.4]
21  (0.833,56.7] (6.67,116] (8.03,14.4]
22  (0.833,56.7]  (225,334] (14.4,20.7]
23  (0.833,56.7] (6.67,116] (8.03,14.4]
24  (0.833,56.7] (6.67,116] (8.03,14.4]
25  (0.833,56.7] (6.67,116] (14.4,20.7]
26  (0.833,56.7]  (225,334] (14.4,20.7]
27  (0.833,56.7]  (116,225] (1.68,8.03]
28  (0.833,56.7] (6.67,116] (8.03,14.4]
29  (0.833,56.7]  (225,334] (14.4,20.7]
30     (112,168]  (116,225] (1.68,8.03]
31  (0.833,56.7]  (225,334] (1.68,8.03]
32  (0.833,56.7]  (225,334] (8.03,14.4]
33  (0.833,56.7]  (225,334] (8.03,14.4]
34  (0.833,56.7]  (225,334] (14.4,20.7]
35  (0.833,56.7]  (116,225] (8.03,14.4]
36  (0.833,56.7]  (116,225] (8.03,14.4]
37  (0.833,56.7]  (225,334] (8.03,14.4]
38  (0.833,56.7]  (116,225] (8.03,14.4]
39  (0.833,56.7]  (225,334] (1.68,8.03]
40    (56.7,112]  (225,334] (8.03,14.4]
41  (0.833,56.7]  (225,334] (8.03,14.4]
42  (0.833,56.7]  (225,334] (8.03,14.4]
43  (0.833,56.7]  (225,334] (8.03,14.4]
44  (0.833,56.7]  (116,225] (1.68,8.03]
45  (0.833,56.7]  (225,334] (8.03,14.4]
46  (0.833,56.7]  (225,334] (8.03,14.4]
47  (0.833,56.7]  (116,225] (14.4,20.7]
48  (0.833,56.7]  (225,334] (14.4,20.7]
49  (0.833,56.7] (6.67,116] (8.03,14.4]
50  (0.833,56.7]  (116,225] (8.03,14.4]
51  (0.833,56.7]  (116,225] (8.03,14.4]
52    (56.7,112]  (116,225] (1.68,8.03]
53  (0.833,56.7] (6.67,116] (1.68,8.03]
54  (0.833,56.7] (6.67,116] (1.68,8.03]
55  (0.833,56.7]  (225,334] (1.68,8.03]
56    (56.7,112]  (116,225] (1.68,8.03]
```

SECTION 3: IMPUTATION PAR MODÈLES STATISTIQUES

```
# Imputation par Modèles Mixtes
# Assumption: 'Month' can be used as a random effect group for illustration
data$Month <- factor(data$Month)
fit_mixed <- lmer(Ozone ~ Solar.R + Wind + Temp + (1 | Month), data = na.omit(data), REML = FALSE)
data$Ozone_imputed <- ifelse(is.na(data$Ozone), predict(fit_mixed, newdata = data, allow.new.levels = TRUE), data$Ozone)
print("Données après imputation par modèle mixte :")
print(head(data$Ozone_imputed))
```

```
[1] "Données après imputation par modèle mixte :"
[1] 41 36 12 18 NA 28
```

```
# Imputation par Chaînes de Markov Monte Carlo (MCMC)
data$Solar.R <- as.numeric(as.character(data$Solar.R))
data$Solar.R[is.na(data$Solar.R)] <- mean(data$Solar.R, na.rm = TRUE)
fit_mcmc <- MCMCglmm(Ozone ~ Solar.R + Wind + Temp, random = ~Month, data = data, nitt = 13000, burnin = 3000, pr = TRUE)
data$Ozone_imputed_mcmc <- ifelse(is.na(data$Ozone), predict(fit_mcmc, newdata = data), data$Ozone)
# Print the imputed data
print("Données après imputation avec MCMC :")
print(head(data$Ozone_imputed_mcmc))
```

```

                       MCMC iteration = 0

                       MCMC iteration = 1000

                       MCMC iteration = 2000

                       MCMC iteration = 3000

                       MCMC iteration = 4000

                       MCMC iteration = 5000

                       MCMC iteration = 6000

                       MCMC iteration = 7000

                       MCMC iteration = 8000

                       MCMC iteration = 9000

                       MCMC iteration = 10000

                       MCMC iteration = 11000

                       MCMC iteration = 12000

                       MCMC iteration = 13000
[1] "Données après imputation avec MCMC :"
[1] 41.000000 36.000000 12.000000 18.000000 -8.177573 28.000000
```

```
# Joint Modeling avec Amelia
data_joint <- amelia(data, m = 5, idvars = "Month")$imputations[[1]]
data$Ozone_imputed_amelia <- data_joint$Ozone
print("Données après imputation avec Joint Modeling (Amelia) :")
print(head(data$Ozone_imputed_amelia))
```

```
Warning message in amcheck(x = x, m = m, idvars = numopts$idvars, priors = priors, :
“The variables (or variable with levels) Ozone_imputed, Ozone_imputed_mcmc are perfectly collinear with another variable in the data.
”
-- Imputation 1 --

  1  2  3  4  5  6  7  8  9

-- Imputation 2 --

  1  2  3  4  5  6  7

-- Imputation 3 --

  1  2  3  4  5  6  7  8

-- Imputation 4 --

  1  2  3  4  5  6  7  8  9

-- Imputation 5 --

  1  2  3  4  5  6  7  8  9 10 11 12
```

```
[1] "Données après imputation avec Joint Modeling (Amelia) :"
[1] 41.000000 36.000000 12.000000 18.000000 -8.694116 28.000000
```

SECTION 4: VALIDATION DE L'IMPUTATION

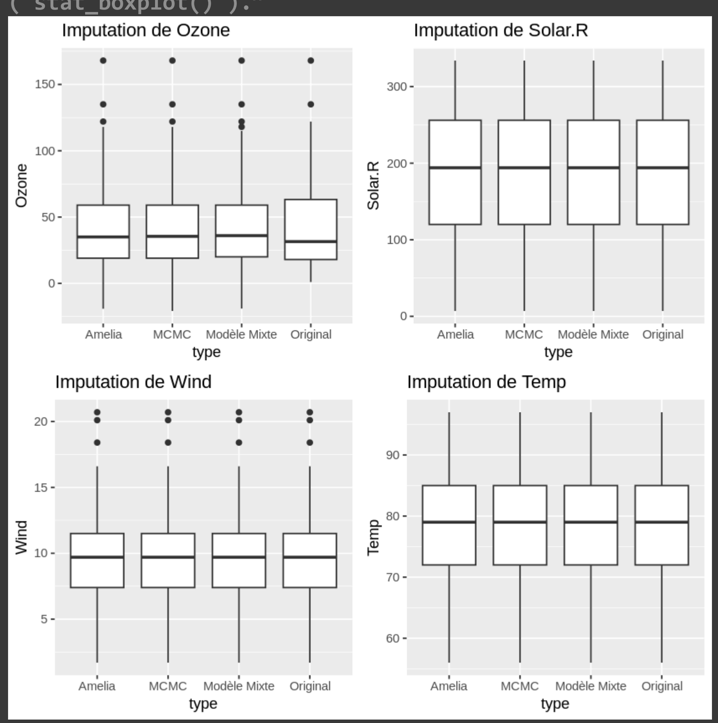
```
# Calcul de la RMSE pour évaluer la qualité de chaque méthode
rmse <- function(true, predicted) {
  sqrt(mean((true - predicted)^2, na.rm = TRUE))
}
rmse_mixed <- rmse(data$Ozone, data$Ozone_imputed)
rmse_mcmc <- rmse(data$Ozone, data$Ozone_imputed_mcmc)
rmse_amelia <- rmse(data$Ozone, data$Ozone_imputed_amelia)

# Print RMSE results
print(paste("RMSE Modèle Mixte:", rmse_mixed))
print(paste("RMSE MCMC:", rmse_mcmc))
print(paste("RMSE Amelia:", rmse_amelia))
```

```
[1] "RMSE Modèle Mixte: 0"
[1] "RMSE MCMC: 0"
[1] "RMSE Amelia: 0"
```

SECTION 5: VISUALISATION DES DONNÉES IMPUTÉES

```
# Preparing data for plotting
data_long <- rbind(
  data.frame(type = "Original", Ozone = data$Ozone, Solar.R = data$Solar.R, Wind = data$Wind, Temp = data$Temp),
  data.frame(type = "Modèle Mixte", Ozone = data$Ozone_imputed, Solar.R = data$Solar.R, Wind = data$Wind, Temp = data$Temp),
  data.frame(type = "MCMC", Ozone = data$Ozone_imputed_mcmc, Solar.R = data$Solar.R, Wind = data$Wind, Temp = data$Temp),
  data.frame(type = "Amelia", Ozone = data_joint$Ozone, Solar.R = data$Solar.R, Wind = data$Wind, Temp = data$Temp)
)

# Ensure column names are consistent for proper rbind operation


# Plotting
plot_list <- list()
for (col in names(data[, c("Ozone", "Solar.R", "Wind", "Temp")])) {
  p <- ggplot(data_long, aes_string(x = 'type', y = col)) +
    geom_boxplot() + labs(title = paste("Imputation de", col))
  plot_list[[col]] <- p
}
do.call(grid.arrange, plot_list)
```

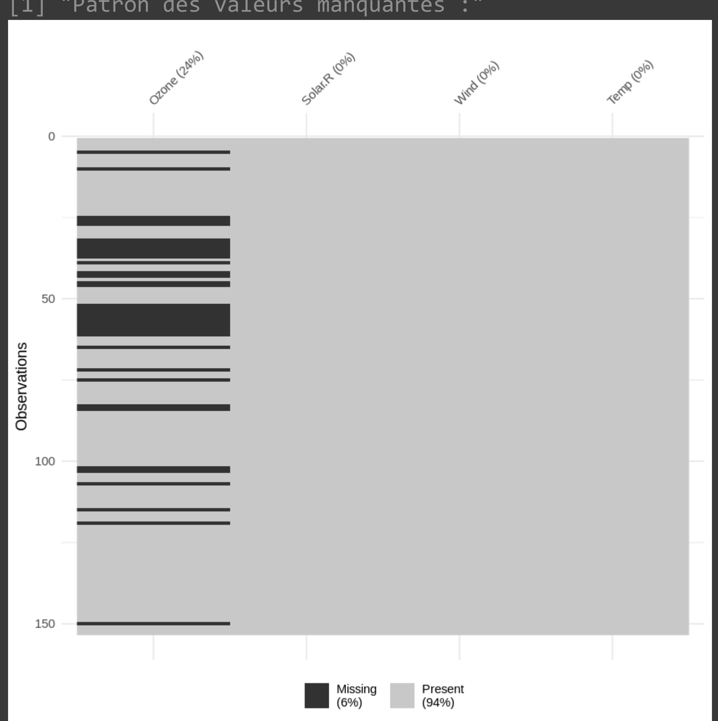
```
Warning message:
"Removed 39 rows containing non-finite outside the scale range
(`stat_boxplot()`)."
```



SECTION 6: ANALYSE DES VALEURS MANQUANTES

```
# Heatmap des valeurs manquantes
print("Heatmap des valeurs manquantes :")
vis_miss(data[, c("Ozone", "Solar.R", "Wind", "Temp")])

# Patron des valeurs manquantes avec md.pattern
print("Patron des valeurs manquantes :")
pattern <- md.pattern(data[, c("Ozone", "Solar.R", "Wind", "Temp")])
print(pattern)

# Test MCAR de Little
mcar_test_result <- mcar_test(data[, c("Ozone", "Solar.R", "Wind", "Temp")])
print("Test de Little pour MCAR :")
print(mcar_test_result)

```

```
[1] "Heatmap des valeurs manquantes :"
[1] "Patron des valeurs manquantes :"
```




```
    Solar.R Wind Temp Ozone
116       1    1    1     1  0
37        1    1    1     0  1
          0    0    0    37 37
[1] "Test de Little pour MCAR :"
# A tibble: 1 × 4
  statistic    df p.value missing.patterns
      <dbl> <dbl>   <dbl>            <int>
1     0.517     3   0.915                2
```