Handling missing values with R

Koji Mizumura

Table of Contents

[1) Regression with NA (quantitative) for ozone 2](#_Toc521055456)

[1.1) Descriptive statistics, visualization with missing values 4](#_Toc521055457)

[1.2.1) Tabulation and Summaries 5](#_Toc521055458)

[1.2.2) Visualization 6](#_Toc521055459)

[1.2.3) Visualize missings across cases and variables 11](#_Toc521055460)

[1.3) PCA with missing values 21](#_Toc521055461)

[1.4) Multiple imputation 25](#_Toc521055462)

[Generate multiple data sets. 25](#_Toc521055463)

[Inspect the imputed values 32](#_Toc521055464)

[Perform regression 36](#_Toc521055465)

[1.5) Ecological example 39](#_Toc521055466)

[2) Categorical/mixed/multi-block data with missing values 56](#_Toc521055467)

[2.1) Single imputation of categorical data with MCA/MCA with missing values 56](#_Toc521055468)

[2.3) Multiple imputation for categorical data: Mu;tinomial regression with missing values 83](#_Toc521055469)

[2.3) Imputation with groups of variables/multiple factor analysis with missing values. 85](#_Toc521055470)

[3) Contingency tables with count data and missing values 86](#_Toc521055471)

[4) Multilevel (mixed) data with missing values 86](#_Toc521055472)

[Alternative approach for visualizing missingness 86](#_Toc521055473)

# 1) Regression with NA (quantitative) for ozone

First of all you will need to install the following packages

install.packages("VIM")  
install.packages("devtools")  
library(devtools)  
install\_github("njtierney/naniar")  
install.packages("naniar")  
  
install.packages("missMDA")  
install.packages("Amelia")  
install.packages("mice")  
install.packages("missForest")  
install.packages("FactoMineR")  
install.packages("tidyverse")

Air pollution is currently one of the most serious public health worries worldwide. Many epidemiological studies have proved the influence that some chemical compounds, such as sulfur dioxide (SO2), nitrogen dioxide(NO2), ozone(O3) can have on our health. Associations set up to monitor air quality are active all over the world to measure the concentration of these pollutants. They also keep a record of meteorological conditions such as temperature, cloud cover, wind, etc.

We have at our disposal 112 observations collected during the summer or 2001 in Rennes. The variables available are: \* maxO3 (maximum daily ozone) \* maxO3v (maximum daily ozone the previous day) \* T12 (temperature at midday) \* T9 \* T15 (Temp at 3pm) \* Vx12 (projection of the wind speed vector on the east-west axis at midday) \* Vx9 and Vx15 as well as the Nebulosity (cloud) Ne9, Ne12, Ne15

Here the final aim is to analyze the relationship between the maximum daily ozone(max03) level and the other meteorological variables. To do so we will perform regression to explain maxO3 in function of all the other variables. This data is incomplete - there are missing values. Indeed, it occurs frequently to have machines that fail one day, leading to some information not recorded. We will therefore perform regression with missing values via multiple imputation.

* Importing the data

ozo <- read.table("C:/Users/kojikm.mizumura/Desktop/Data Science/8. UseR2018/1 NA Treatment/ozoneNA.csv",header=T,sep=",",row.names=1)  
  
WindDirection <- ozo[,12]  
don <- ozo[,1:11] #### keep the continuous variables  
  
# dataset summary  
summary(don)

## maxO3 T9 T12 T15   
## Min. : 42.00 Min. :11.30 Min. :14.30 Min. :14.90   
## 1st Qu.: 71.00 1st Qu.:16.00 1st Qu.:18.60 1st Qu.:18.90   
## Median : 81.50 Median :17.70 Median :20.40 Median :21.40   
## Mean : 91.24 Mean :18.22 Mean :21.46 Mean :22.41   
## 3rd Qu.:108.25 3rd Qu.:19.90 3rd Qu.:23.60 3rd Qu.:25.65   
## Max. :166.00 Max. :25.30 Max. :33.50 Max. :35.50   
## NA's :16 NA's :37 NA's :33 NA's :37   
## Ne9 Ne12 Ne15 Vx9   
## Min. :0.000 Min. :0.000 Min. :0.00 Min. :-7.8785   
## 1st Qu.:3.000 1st Qu.:4.000 1st Qu.:3.00 1st Qu.:-3.0000   
## Median :5.000 Median :5.000 Median :5.00 Median :-0.8671   
## Mean :4.987 Mean :4.986 Mean :4.60 Mean :-1.0958   
## 3rd Qu.:7.000 3rd Qu.:7.000 3rd Qu.:6.25 3rd Qu.: 0.6919   
## Max. :8.000 Max. :8.000 Max. :8.00 Max. : 5.1962   
## NA's :34 NA's :42 NA's :32 NA's :18   
## Vx12 Vx15 maxO3v   
## Min. :-7.8785 Min. :-9.000 Min. : 42.00   
## 1st Qu.:-3.6941 1st Qu.:-3.759 1st Qu.: 70.00   
## Median :-1.9284 Median :-1.710 Median : 82.50   
## Mean :-1.6853 Mean :-1.830 Mean : 89.39   
## 3rd Qu.:-0.1302 3rd Qu.: 0.000 3rd Qu.:101.00   
## Max. : 6.5778 Max. : 3.830 Max. :166.00   
## NA's :10 NA's :21 NA's :12

head(don)

## maxO3 T9 T12 T15 Ne9 Ne12 Ne15 Vx9 Vx12 Vx15 maxO3v  
## 20010601 87 15.6 18.5 NA 4 4 8 0.6946 -1.7101 -0.6946 84  
## 20010602 82 NA NA NA 5 5 7 -4.3301 -4.0000 -3.0000 87  
## 20010603 92 15.3 17.6 19.5 2 NA NA 2.9544 NA 0.5209 82  
## 20010604 114 16.2 19.7 NA 1 1 0 NA 0.3473 -0.1736 92  
## 20010605 94 NA 20.5 20.4 NA NA NA -0.5000 -2.9544 -4.3301 114  
## 20010606 80 17.7 19.8 18.3 6 NA 7 -5.6382 -5.0000 -6.0000 94

dim(don)

## [1] 112 11

* Load the libraries

library(VIM)

## Warning: package 'VIM' was built under R version 3.5.1

library(FactoMineR)

## Warning: package 'FactoMineR' was built under R version 3.5.1

library(missMDA)

## Warning: package 'missMDA' was built under R version 3.5.1

## 1.1) Descriptive statistics, visualization with missing values

**Q1** When could it be a good idea to delete rows or columns with missing values to work with a complete data set? Could you do it here?

dim(na.omit(don))

## [1] 13 11

Deleting rows or columns is possible as long as there is enough data left and the missing values of the MCAR type so that the sample is a subsample of the original data. We will obtain unbiased estimators but with more variance. Deleting observations with missing data for ozone data leads to a table with 13 rows.

First, we perform some descriptive statistics (how many missing? how many variables, individuals with missing?) and try to **insepect and visualizae the pattern of missing entiries and get hints on the mechanism**. For this purpose, we use the R package **naniar** as well as Multiple Correspondence Analysis (FactoMineR package). An alternative would to use VIM (Visualization and Imputation of Missing Values - MAthias Templ)

naniar provides principled, tidy ways to summarize, visualize, and manipulate missing data with minimal deviations from the workflows in ggplot2 and tidy data.

We can start off with some quick summaries of the amount of missing and complete data in don using: - pct\_miss() to give us the percentage of missings in the data - n\_miss() to give the number of missings,

and their complete equivalents: - pct\_complete() to give us the percentage of completes in the data - n\_complete() to give the number of complete values

library(naniar)

## Warning: package 'naniar' was built under R version 3.5.1

library(tidyverse)

## Warning: package 'tidyverse' was built under R version 3.5.1

## -- Attaching packages ------------------------------------------------------------------------------------------------------------------- tidyverse 1.2.1 --

## √ ggplot2 3.0.0 √ purrr 0.2.5  
## √ tibble 1.4.2 √ dplyr 0.7.6  
## √ tidyr 0.8.1 √ stringr 1.3.1  
## √ readr 1.1.1 √ forcats 0.3.0

## Warning: package 'ggplot2' was built under R version 3.5.1

## Warning: package 'purrr' was built under R version 3.5.1

## Warning: package 'dplyr' was built under R version 3.5.1

## -- Conflicts ---------------------------------------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::between() masks data.table::between()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::first() masks data.table::first()  
## x dplyr::lag() masks stats::lag()  
## x dplyr::last() masks data.table::last()  
## x purrr::transpose() masks data.table::transpose()

pct\_miss(don) # percentage of missing value in the data.

## [1] 23.7013

n\_miss(don) # number of missing values in the

## [1] 292

n\_complete(don) # without missing value

## [1] 940

pct\_complete(don) # without missing value

## [1] 76.2987

This is useful, but would be repetitive if you wanted to repeat this for every variable. We can instead look at summaries across the **variables** and **cases**

### 1.2.1) Tabulation and Summaries

You can find the number and percentage missing in each variable and case using miss\_case\_summary and miss\_var\_summary.

miss\_var\_summary(don)

## # A tibble: 11 x 4  
## variable n\_miss pct\_miss n\_miss\_cumsum  
## <chr> <int> <dbl> <int>  
## 1 Ne12 42 37.5 199  
## 2 T9 37 33.0 53  
## 3 T15 37 33.0 123  
## 4 Ne9 34 30.4 157  
## 5 T12 33 29.5 86  
## 6 Ne15 32 28.6 231  
## 7 Vx15 21 18.8 280  
## 8 Vx9 18 16.1 249  
## 9 maxO3 16 14.3 16  
## 10 maxO3v 12 10.7 292  
## 11 Vx12 10 8.93 259

miss\_case\_summary(don)

## # A tibble: 112 x 4  
## case n\_miss pct\_miss n\_miss\_cumsum  
## <int> <int> <dbl> <int>  
## 1 18 8 72.7 44  
## 2 56 8 72.7 167  
## 3 25 7 63.6 69  
## 4 47 7 63.6 138  
## 5 110 7 63.6 286  
## 6 24 6 54.5 62  
## 7 31 6 54.5 91  
## 8 45 6 54.5 131  
## 9 22 5 45.5 55  
## 10 33 5 45.5 97  
## # ... with 102 more rows

This shows us there are two variables with exactly 37 missings in both, but that each individual variable seems to have unique number of missings. We also note that there are no variables with zero missings.

For the cases, we see that there are 13 cases with no missings, 24 with 1 missing, 22 with 2 missing and so on.

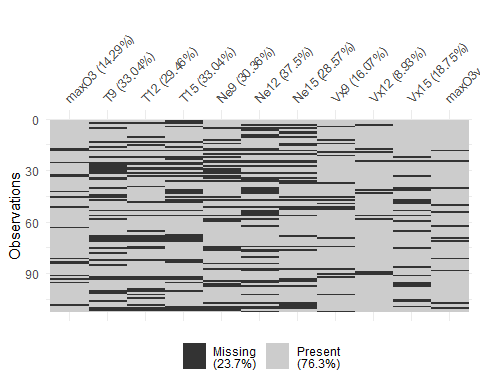
### 1.2.2) Visualization

A quick way to get a look at the missingness in the data is to use vis\_miss. This visualizes the missingness across the entire data set.

library(visdat)

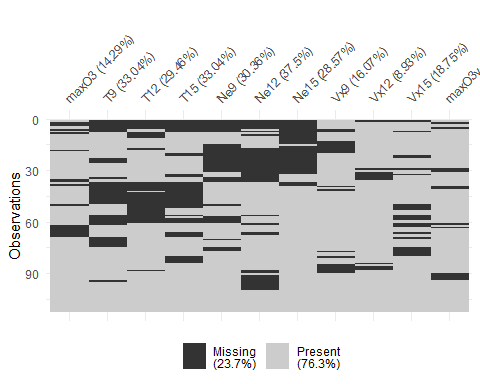
## Warning: package 'visdat' was built under R version 3.5.1

vis\_miss(don)



You can also apply clustering to find similar missingness groups by setting cluster=TRUE.

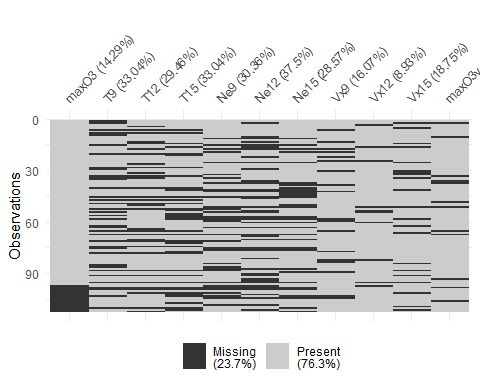
vis\_miss(don,cluster=TRUE)



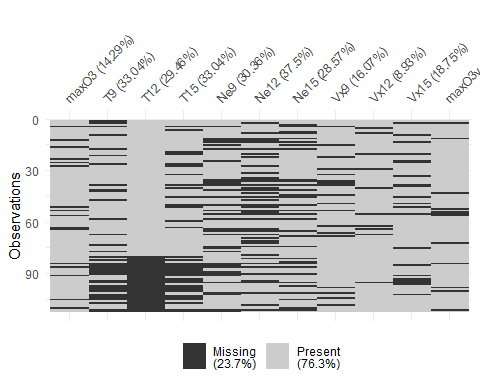
There are a lot of different clusters here, it is difficult to clear relationships here.

Another technique is to try arranging by different variables using arrange().

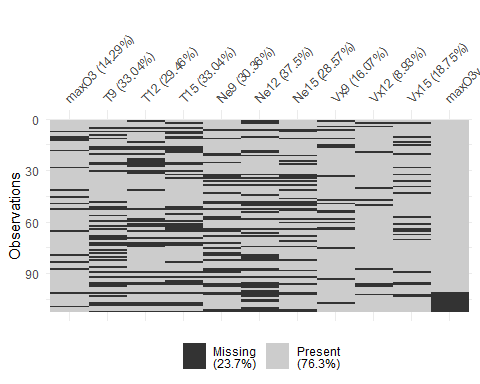
don %>%   
 arrange(maxO3) %>%   
 vis\_miss()



don %>%   
 arrange(T12) %>%   
 vis\_miss()



don %>%   
 arrange(maxO3v) %>%   
 vis\_miss



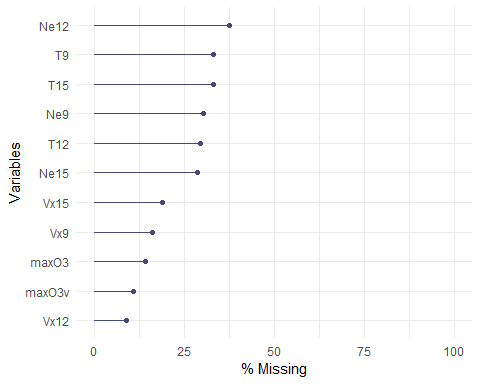
### 1.2.3) Visualize missings across cases and variables

Another way to look at missings is to visualize them by variables and cases. To visualize the missings for each variable, we usegg\_miss\_var:

library(naniar)  
gg\_miss\_var(don)

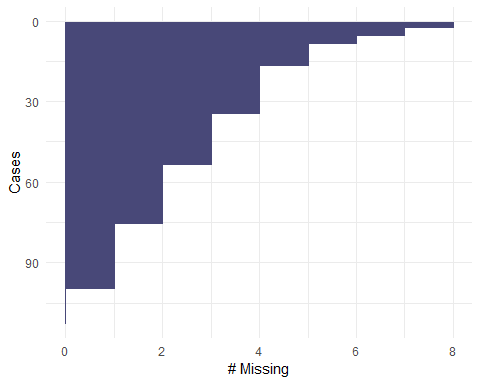
And show the percent missing by setting show\_pct=TRUE, and set the ylimits to be between 0 and 100.

library(tidyverse)  
gg\_miss\_var(don,  
 show\_pct=TRUE)+  
 ylim(0,100)



We can look at the missings across cases using gg\_miss\_case:

gg\_miss\_case(don)



And we can look at the combination and patterns of missingness by looking at an upset plot of the missingness - with **gg\_miss\_upset**.

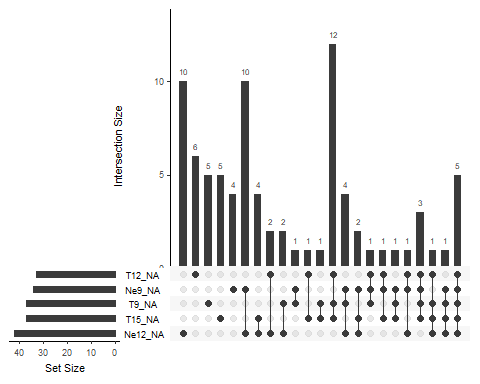
The upset shows the combination of missings, by default choosing the 5 variables with the most missings, and then orders by the size of the missings in that set.

We set order.by="freq" to order the missiness by their frequency.

# gg\_miss\_upset(don,  
# order.by="freq")  
# install.packages("UpSetR")  
library(UpSetR)

## Warning: package 'UpSetR' was built under R version 3.5.1

don %>%   
 as\_shadow\_upset() %>%   
 upset()

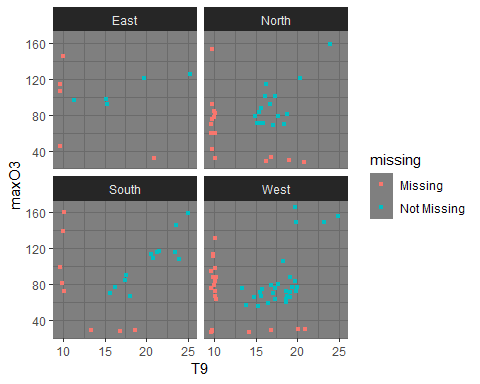


We can then explore the missingness by some categorical variable using facet:

head(don)

## maxO3 T9 T12 T15 Ne9 Ne12 Ne15 Vx9 Vx12 Vx15 maxO3v  
## 20010601 87 15.6 18.5 NA 4 4 8 0.6946 -1.7101 -0.6946 84  
## 20010602 82 NA NA NA 5 5 7 -4.3301 -4.0000 -3.0000 87  
## 20010603 92 15.3 17.6 19.5 2 NA NA 2.9544 NA 0.5209 82  
## 20010604 114 16.2 19.7 NA 1 1 0 NA 0.3473 -0.1736 92  
## 20010605 94 NA 20.5 20.4 NA NA NA -0.5000 -2.9544 -4.3301 114  
## 20010606 80 17.7 19.8 18.3 6 NA 7 -5.6382 -5.0000 -6.0000 94

ggplot(don,  
 aes(x=T9,  
 y=maxO3))+  
 geom\_miss\_point()+  
 facet\_wrap(~ozo$WindDirection)+  
 theme\_dark()



To take a closer look at the distribution of missings we add some missingness indicator information to the data. We call this indicator infrmation a “shadow matrix”, and it gets added to the data with bind\_shadow. This creates a copy of the data with the name “Variable\_NA”, and the values “NA” and “!NA” for missing, and not missing, respectively.

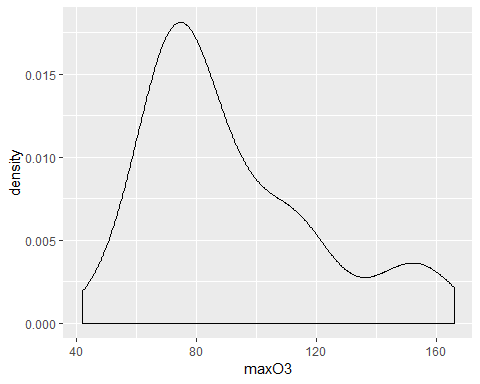
don %>% bind\_shadow() %>% glimpse()

## Observations: 112  
## Variables: 22  
## $ maxO3 <int> 87, 82, 92, 114, 94, 80, 79, 79, 101, 106, 101, 90, ...  
## $ T9 <dbl> 15.6, NA, 15.3, 16.2, NA, 17.7, 16.8, 14.9, 16.1, 18...  
## $ T12 <dbl> 18.5, NA, 17.6, 19.7, 20.5, 19.8, 15.6, 17.5, 19.6, ...  
## $ T15 <dbl> NA, NA, 19.5, NA, 20.4, 18.3, 14.9, 18.9, 21.4, 22.9...  
## $ Ne9 <int> 4, 5, 2, 1, NA, 6, 7, 5, 2, 5, 7, NA, 7, NA, 8, 6, N...  
## $ Ne12 <int> 4, 5, NA, 1, NA, NA, 8, 5, NA, NA, 7, 6, 5, 7, 7, 5,...  
## $ Ne15 <int> 8, 7, NA, 0, NA, 7, NA, NA, 4, NA, 3, 8, 6, NA, NA, ...  
## $ Vx9 <dbl> 0.6946, -4.3301, 2.9544, NA, -0.5000, -5.6382, -4.33...  
## $ Vx12 <dbl> -1.7101, -4.0000, NA, 0.3473, -2.9544, -5.0000, -1.8...  
## $ Vx15 <dbl> -0.6946, -3.0000, 0.5209, -0.1736, -4.3301, -6.0000,...  
## $ maxO3v <int> 84, 87, 82, 92, 114, 94, 80, 99, 79, 101, 106, 101, ...  
## $ maxO3\_NA <fct> !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !N...  
## $ T9\_NA <fct> !NA, NA, !NA, !NA, NA, !NA, !NA, !NA, !NA, !NA, !NA,...  
## $ T12\_NA <fct> !NA, NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, NA, !NA,...  
## $ T15\_NA <fct> NA, NA, !NA, NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, ...  
## $ Ne9\_NA <fct> !NA, !NA, !NA, !NA, NA, !NA, !NA, !NA, !NA, !NA, !NA...  
## $ Ne12\_NA <fct> !NA, !NA, NA, !NA, NA, NA, !NA, !NA, NA, NA, !NA, !N...  
## $ Ne15\_NA <fct> !NA, !NA, NA, !NA, NA, !NA, NA, NA, !NA, NA, !NA, !N...  
## $ Vx9\_NA <fct> !NA, !NA, !NA, NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA...  
## $ Vx12\_NA <fct> !NA, !NA, NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA...  
## $ Vx15\_NA <fct> !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !N...  
## $ maxO3v\_NA <fct> !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !NA, !N...

This allows us to think about the “missingness” of a variable as its own variable. So we can look at a density plot of maxO3 in ggplot2:

ggplot(don,  
 aes(x=maxO3))+  
 geom\_density()

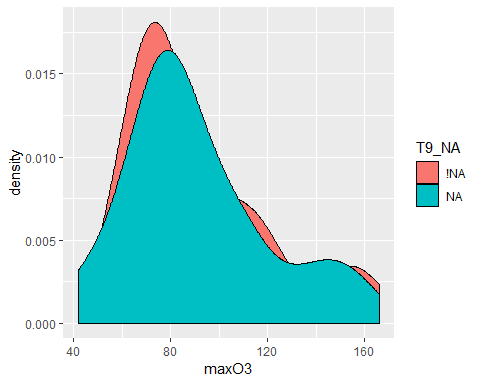
## Warning: Removed 16 rows containing non-finite values (stat\_density).



We can use the “shadow matrix” to allow us to look at the density according to whether T9\_NA is missing:

don %>%   
 bind\_shadow() %>%   
 ggplot(aes(x=maxO3,  
 fill=T9\_NA))+  
 geom\_density()

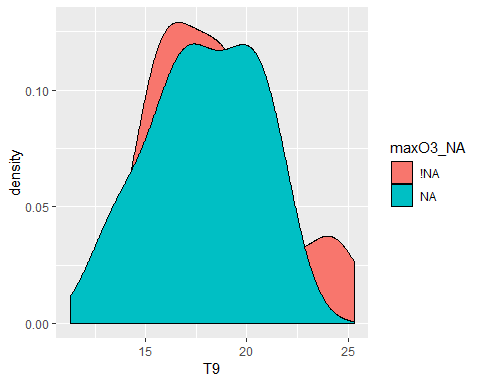
## Warning: Removed 16 rows containing non-finite values (stat\_density).



Or equivalently look at the variable T9 according to whether maxO3 is missing:

don %>%   
 bind\_shadow() %>%   
 ggplot(aes(x=T9,  
 fill=maxO3\_NA))+  
 geom\_density()

## Warning: Removed 37 rows containing non-finite values (stat\_density).



We can see that the distribution of T9 is the same when maxO3 is observed and when max03 is missing. If the two densities (red and blue) were very different, it would imply that when maxO3 is missing the value of T9 can be very high or very low which lead to suspect the MAR hypothesis.

We can usebind\_shadow to then group by the missingness of a variable and perform some summary statistics on T9 for when maximum daily ozone level is present, and when it is missing.

don %>%   
 bind\_shadow() %>%   
 group\_by(maxO3\_NA) %>%   
 summarise\_at(  
 .vars=vars(T9),  
 .funs=funs(mean,sd,var,min,max),  
 na.rm=T  
 )

## # A tibble: 2 x 6  
## maxO3\_NA mean sd var min max  
## <fct> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 !NA 18.3 3.06 9.36 11.3 25.3  
## 2 NA 17.9 2.61 6.83 13.3 21

**Q2** Do you observe any associations bwtween the missing entries? When values are missing on a variable, does it correspond to small or large values on another one?

We observed that the temperature variables T9, T12, and T15 tend to be missing together (probably indicating that thermometers failed) [as well as the Ne9 Ne12 and Ne15 variables.]

We see more “red” values. WE do not see more black or white values which should imply that T9 is missing it would have corresponded to high or low values in everything points to MCAR values.

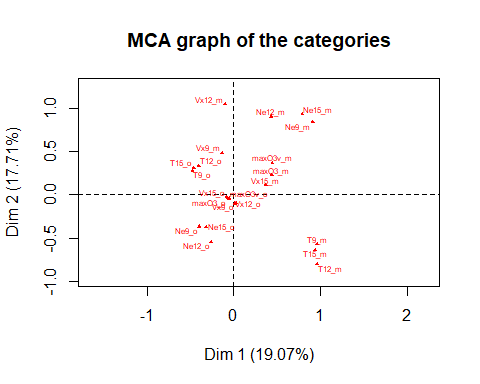
**R1** Create a categorical data set with “o” when the value of the cell is observed and “m” when it is missing, and with the same row and column names as in the original data. Then, you can perform Multiple Correspondence Analysis with the MCA function of the FactoMineR package.

library(FactoMineR)  
?MCA

MCA can be seen as the counterpart of PCA for categorical data and here is used to study associations between missing and observed entries. MCA is a straightforwardly tool to visualize the missing data pattern even if the number of variable is large.

It shows if missing values simultaneously occur in several variables or if missing values occur when some other variables are observed

library(FactoMineR)  
  
data\_miss <- data.frame(is.na(don))  
data\_miss <- apply(X=data\_miss, FUN=function(x) if(x) "m" else "o", MARGIN=c(1,2))  
  
# data\_miss <- as\_shadow(don) with the naniar package  
res.mca <- MCA(data\_miss,graph=F)  
plot(res.mca,invis="ind",title="MCA graph of the categories",cex=0.5)



## 1.3) PCA with missing values

Then before modeling the data, we perform a *PCA with missing values* to explore the correlation between variables. Using the R package missMDA dedicated to perform principal component methods with missing values and to impute data with PC methods.

* Perform PCA with missing values using the *imputePCA* functions, with the number of components determined by the **estim\_ncpPCA**. Then plot the variable circle.

library(missMDA)

?estim\_ncpPCA  
?imputePCA

estim\_ncpPCA: Estimate the number of dimensions for the Principal Component Analysis by cross-validation.

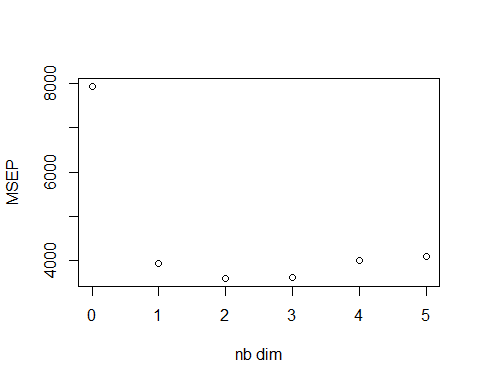
ImputePCA: Impute the missing entries of a contingency table using Correspondence Analysis (CA). Can be used as a preliminary step before performing CA on an incomplete data set.

The package missMDA allows the use of principal component methods for an incomplete data set. To achieve this goal in the case of PCA, the missing values are predicted using the iterative PCA algorithm for a predefined number of dimensions. Then, PCA is performed on the imputed data set. The single imputation step requires tuning the number of dimensions used to impute the data.

nb <- estim\_ncpPCA(don,method.cv="Kfold",verbose=F)  
# estimate the number of components from incomplete data  
  
nb$ncp

## [1] 2

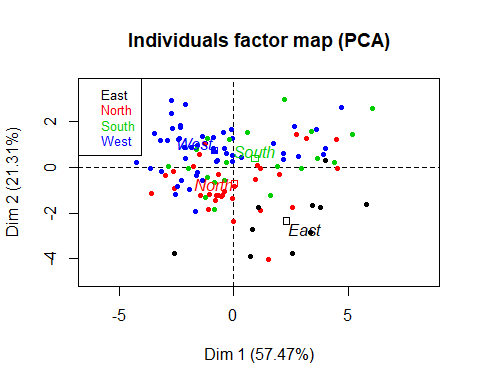
#2  
  
plot(0:5,nb$criterion,xlab="nb dim",ylab="MSEP")



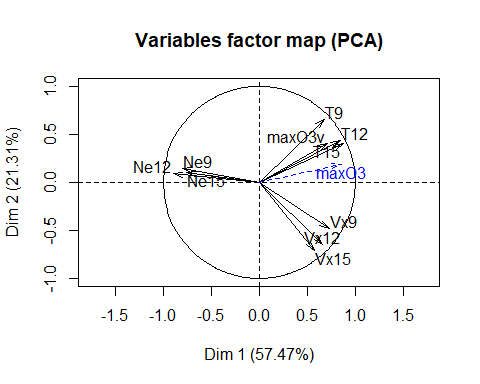
res.comp <- imputePCA(don,ncp=nb$ncp)  
# iterateivePCA algorithm  
  
res.comp$completeObs[1:3,]

## maxO3 T9 T12 T15 Ne9 Ne12 Ne15 Vx9  
## 20010601 87 15.6000 18.50000 20.47146 4 4.000000 8.000000 0.6946  
## 20010602 82 18.5047 20.86986 21.79932 5 5.000000 7.000000 -4.3301  
## 20010603 92 15.3000 17.60000 19.50000 2 3.984066 3.812104 2.9544  
## Vx12 Vx15 maxO3v  
## 20010601 -1.710100 -0.6946 84  
## 20010602 -4.000000 -3.0000 87  
## 20010603 1.950563 0.5209 82

# the imputed data set  
  
imp <- cbind.data.frame(res.comp$completeObs,WindDirection)  
  
  
res.pca <- PCA(imp,quanti.sup=1,quali.sup=12,ncp=nb$ncp,graph=FALSE)  
plot(res.pca,hab=12,lab="quali");



plot(res.pca,choix="var")



head(res.pca$ind$coord)

## Dim.1 Dim.2  
## 20010601 -0.6604580 -1.2048271  
## 20010602 -1.2317545 1.0465411  
## 20010603 0.7984643 -2.7299508  
## 20010604 2.5423205 -1.7435774  
## 20010605 -0.4047517 0.8406578  
## 20010606 -2.6701824 1.6934864

# scores (principal components)

The incomplete data set can be imputed using the function imputePCA performing the iterative PCA algorithm, specifying the number of dimensions through the argument ncp=2.

At convergence the algorithm provides both an estimation of the scores and leading as well as a completed data set. The imputePCA function outputs the imputed data set. The completed data set is in the object ocompleteObs. The imputePCA function also outputs the fitted matrix in the object fitted.

**Q3** Could you guess cross-validation is performed to select the number of components?

The cross-validation is performed with the Kfold method for the Kfold. A percentage PNA of missing values is inserted and predicted with a PCA model using ncp.min to ncp.max dimensions. This process is repeated nbsim times. The number of components which leads to the smallest MSEP(Mean Standard Error of Prediction) is retained.

Through the argument method.cv, the function estim\_ncpPCA proposes several cross-validation procedures to choose this number. The default method is the generalized cross-validation method (method.cv="gcv"). It consists in searching the number of dimensions which minimizes the generalized cross-validation criterion, which can be seen as an approximation of the leave-one-out cross-validation criterion. The procedure is very fast, because it does not require adding explicitly missing values and predicting them for each cell of the data set.

However, the number of dimensions minimizing the criterion can sometimes be obviously when several local minimum occur. In such a case, more computationally intensive methods, those performing explicit cross-validation, can be sued, such as Kfold (method.cv="Kfold) or leave-one-out (method.cv="loo").

The Kfold cross-validation suggests to retain 2 dimensions for the imputation of the data set.

## 1.4) Multiple imputation

### Generate multiple data sets.

We perform multiple imputation either assuming 1) Joint modeling (one joint probabilistic model for the variables all together) - we use the R package Amelia, which is by default consider Gaussian distribution 2) Conditional modeling (one model per variable) approach - we use the R package mice which by default consider one model of linear regression per variable 3) a PCA baseds model - we use the R package missMDA

For each approach, we generate 100 imputed data sets.

library(Amelia)

## Warning: package 'Amelia' was built under R version 3.5.1

## Loading required package: Rcpp

## Warning: package 'Rcpp' was built under R version 3.5.1

## ##   
## ## Amelia II: Multiple Imputation  
## ## (Version 1.7.5, built: 2018-05-07)  
## ## Copyright (C) 2005-2018 James Honaker, Gary King and Matthew Blackwell  
## ## Refer to http://gking.harvard.edu/amelia/ for more information  
## ##

?amelia

res.amelia <- amelia(don,m=5)

## -- Imputation 1 --  
##   
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20  
## 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40  
## 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57  
##   
## -- Imputation 2 --  
##   
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20  
## 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40  
## 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60  
## 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80  
## 81 82  
##   
## -- Imputation 3 --  
##   
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20  
## 21 22 23 24 25 26 27 28 29 30 31  
##   
## -- Imputation 4 --  
##   
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20  
## 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35  
##   
## -- Imputation 5 --  
##   
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20  
## 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40  
## 41 42 43 44 45 46 47 48 49 50 51 52 53

names(res.amelia$imputations)

## [1] "imp1" "imp2" "imp3" "imp4" "imp5"

res.amelia$imputations$imp1

## maxO3 T9 T12 T15 Ne9 Ne12  
## 20010601 87.00000 15.60000 18.50000 18.17613 4.0000000 4.000000  
## 20010602 82.00000 18.30995 19.75505 19.81190 5.0000000 5.000000  
## 20010603 92.00000 15.30000 17.60000 19.50000 2.0000000 5.672168  
## 20010604 114.00000 16.20000 19.70000 23.45268 1.0000000 1.000000  
## 20010605 94.00000 18.23216 20.50000 20.40000 2.7724501 3.857982  
## 20010606 80.00000 17.70000 19.80000 18.30000 6.0000000 2.649196  
## 20010607 79.00000 16.80000 15.60000 14.90000 7.0000000 8.000000  
## 20010610 79.00000 14.90000 17.50000 18.90000 5.0000000 5.000000  
## 20010611 101.00000 16.10000 19.60000 21.40000 2.0000000 3.266562  
## 20010612 106.00000 18.30000 23.31953 22.90000 5.0000000 5.418568  
## 20010613 101.00000 17.30000 19.30000 20.20000 7.0000000 7.000000  
## 20010614 90.00000 17.60000 20.30000 17.40000 7.1158649 6.000000  
## 20010615 72.00000 17.09325 18.48305 20.62282 7.0000000 5.000000  
## 20010616 70.00000 17.10000 18.20000 18.00000 6.7798175 7.000000  
## 20010617 83.00000 15.40000 17.05061 16.60000 8.0000000 7.000000  
## 20010618 88.00000 14.16928 19.10000 19.05281 6.0000000 5.000000  
## 20010620 118.19414 21.00000 24.60000 26.90000 1.2930390 2.263485  
## 20010621 131.99774 18.96065 19.58601 22.68643 4.6809715 5.291643  
## 20010622 121.00000 19.70000 24.20000 26.90000 2.0000000 1.000000  
## 20010623 146.00000 23.60000 28.60000 28.40000 3.8835956 4.605655  
## 20010624 121.00000 20.40000 25.20000 27.70000 1.0000000 0.000000  
## 20010625 146.00000 16.76230 23.33353 25.23818 0.6003430 0.000000  
## 20010626 108.00000 24.00000 23.50000 28.46804 4.0000000 4.000000  
## 20010627 83.00000 19.70000 22.90000 24.80000 9.3028853 8.402636  
## 20010628 83.91571 17.82024 21.38375 24.32371 3.7457913 1.180040  
## 20010629 81.00000 15.25531 17.83278 18.67561 3.0000000 4.000000  
## 20010630 67.00000 20.09824 23.40000 23.70000 4.7477587 6.553301  
## 20010701 70.00000 12.18129 21.19178 23.53295 5.0000000 2.000000  
## 20010702 106.00000 18.34066 24.92562 26.92414 -1.7746239 0.000000  
## 20010703 139.00000 23.02141 30.10000 31.90000 -2.5342575 1.000000  
## 20010704 79.00000 17.52062 19.38764 20.31708 8.4348306 7.110911  
## 20010705 84.98868 16.80000 18.20000 22.00000 8.0000000 8.000000  
## 20010706 130.01682 20.80000 23.79017 24.08252 -0.5791964 3.000000  
## 20010707 113.00000 17.50453 18.20000 22.70000 4.8290825 6.037958  
## 20010708 72.00000 19.86980 21.20000 23.90000 7.0000000 6.037891  
## 20010709 88.00000 19.20000 22.00000 24.09586 4.1183641 3.386069  
## 20010710 77.00000 19.40000 20.70000 22.50000 7.0000000 8.000000  
## 20010711 71.00000 19.20000 21.00000 22.40000 6.0000000 4.000000  
## 20010712 56.00000 13.80000 16.63385 18.50000 8.0000000 8.000000  
## 20010713 45.00000 13.70192 14.50000 15.20000 8.0000000 9.495169  
## 20010714 67.00000 15.60000 18.60000 17.69351 5.0000000 4.727439  
## 20010715 85.00920 16.90000 19.10000 19.57269 5.0000000 8.175964  
## 20010716 84.00000 17.40000 20.40000 20.00723 3.0000000 4.420985  
## 20010717 63.00000 16.82550 20.50000 20.60000 8.0000000 6.000000  
## 20010718 34.32591 16.25654 15.60000 15.79381 7.9843925 8.000000  
## 20010719 92.00000 16.70000 19.10000 19.30000 7.0000000 6.000000  
## 20010720 88.00000 18.84518 20.30000 19.37937 4.8517912 4.690952  
## 20010721 66.00000 18.00000 20.23090 22.23844 8.0000000 6.000000  
## 20010722 72.00000 18.60000 21.90000 23.60000 4.0000000 7.000000  
## 20010723 81.00000 18.80000 22.50000 23.90000 6.0000000 3.000000  
## 20010724 107.91157 19.00000 22.50000 24.10000 3.5174711 4.884450  
## 20010725 149.00000 19.90000 26.90000 29.00000 3.0000000 4.000000  
## 20010726 153.00000 23.49919 29.69507 31.14402 1.0000000 1.381271  
## 20010727 159.00000 24.00000 28.30000 26.50000 2.0000000 2.235960  
## 20010728 149.00000 23.30000 27.60000 28.80000 4.0000000 5.258465  
## 20010729 160.00000 24.32455 26.87250 32.21063 -0.4021709 3.533112  
## 20010730 156.00000 24.90000 30.50000 32.20000 0.0000000 1.000000  
## 20010731 84.00000 20.80324 26.30000 27.80000 8.1585304 4.720933  
## 20010801 126.00000 25.30000 29.50000 31.20000 -0.6409641 4.000000  
## 20010802 116.00000 21.30000 23.80000 22.10000 7.0000000 7.000000  
## 20010803 77.00000 20.00000 18.20000 23.60000 5.0000000 7.000000  
## 20010804 63.00000 18.70000 20.60000 20.30000 6.0000000 4.128179  
## 20010805 57.36770 18.60000 18.70000 17.80000 8.0000000 8.000000  
## 20010806 65.00000 19.20000 23.00000 22.70000 8.0000000 7.000000  
## 20010807 72.00000 19.90000 21.64871 20.40000 7.0000000 7.000000  
## 20010808 60.00000 18.70000 21.40000 21.70000 7.0000000 7.000000  
## 20010809 70.00000 18.40000 17.10000 21.20346 3.0000000 6.000000  
## 20010810 77.00000 16.84753 18.93996 18.36214 4.0000000 5.000000  
## 20010811 98.00000 18.21068 21.63018 25.23713 1.0000000 1.000000  
## 20010812 111.00000 23.08934 24.94755 26.24832 1.0000000 5.000000  
## 20010813 75.00000 16.44613 16.89360 21.58306 8.0000000 7.000000  
## 20010814 116.00000 23.50000 29.80000 31.70000 1.0000000 3.000000  
## 20010815 109.00000 20.80000 23.70000 26.60000 8.0000000 5.000000  
## 20010819 67.00000 18.80000 20.95924 18.90000 10.8677260 10.052587  
## 20010820 76.00000 19.99446 22.58462 24.00000 5.1582702 5.000000  
## 20010821 113.00000 20.60000 24.80000 27.00000 -1.3071353 1.049195  
## 20010822 117.00000 21.60000 26.90000 28.60000 6.0000000 6.960820  
## 20010823 131.00000 23.25329 28.40000 30.10000 5.0000000 3.000000  
## 20010824 166.00000 19.80000 27.20000 30.80000 4.0000000 0.000000  
## 20010825 159.00000 25.00000 33.50000 35.50000 1.0000000 -1.014421  
## 20010826 116.92320 20.10000 22.90000 27.60000 8.0000000 8.000000  
## 20010827 114.00000 15.01154 21.30487 19.70154 7.0000000 4.000000  
## 20010828 115.93108 21.00000 24.40000 25.01540 1.0000000 6.000000  
## 20010829 99.73522 16.90000 17.80000 20.60000 5.6708003 6.164512  
## 20010830 76.00000 18.86392 18.60000 18.70000 7.0000000 7.000000  
## 20010831 59.00000 16.50000 20.30000 20.30000 5.0000000 7.000000  
## 20010901 78.00000 17.70000 20.20000 21.50000 4.3710439 7.340308  
## 20010902 76.00000 17.30000 22.70000 24.60000 4.0000000 3.984471  
## 20010903 55.00000 15.30000 16.80000 19.20000 8.0000000 7.000000  
## 20010904 71.00000 15.90000 19.20000 19.50000 7.0000000 5.000000  
## 20010905 81.10047 16.20000 18.90000 19.30000 2.0000000 5.000000  
## 20010906 59.00000 15.53781 18.20898 16.48117 7.0000000 7.000000  
## 20010907 78.61994 14.24367 18.80216 20.94593 6.0000000 5.000000  
## 20010908 63.00000 17.30000 19.80000 19.40000 9.3037884 6.976746  
## 20010912 76.58166 14.20000 22.20000 20.80796 5.0000000 1.135365  
## 20010913 74.00000 15.80000 18.70000 19.10000 6.1360412 7.000000  
## 20010914 71.00000 15.20000 17.90000 18.60000 2.9659386 3.272843  
## 20010915 69.00000 17.10000 17.70000 17.50000 6.0000000 7.000000  
## 20010916 71.00000 15.40000 16.79446 16.60000 4.0000000 5.000000  
## 20010917 60.00000 14.81809 14.90115 18.43321 4.0000000 5.000000  
## 20010918 42.00000 12.34030 14.30000 14.90000 8.0000000 7.000000  
## 20010919 65.00000 14.80000 17.00164 15.90000 7.0000000 7.336874  
## 20010920 71.00000 15.50000 18.00000 17.40000 7.0000000 7.000000  
## 20010921 96.00000 11.30000 16.86030 20.20000 3.0000000 3.000000  
## 20010922 98.00000 15.20000 19.70000 20.30000 2.0000000 2.000000  
## 20010923 92.00000 13.48448 17.60000 18.20000 1.0000000 4.000000  
## 20010924 76.00000 13.30000 17.70000 17.70000 6.0848269 5.623298  
## 20010925 92.68653 13.30000 18.10410 17.80000 3.0000000 5.000000  
## 20010927 77.00000 16.20000 20.80000 21.86436 5.3605786 3.701125  
## 20010928 99.00000 17.39457 19.00495 19.65117 4.7425880 5.386496  
## 20010929 83.00000 21.30604 24.71655 27.24385 6.0344318 5.000000  
## 20010930 70.00000 15.70000 18.60000 20.70000 7.0000000 5.467206  
## Ne15 Vx9 Vx12 Vx15 maxO3v  
## 20010601 8.0000000 0.69460000 -1.7101000 -0.6946000 84.00000  
## 20010602 7.0000000 -4.33010000 -4.0000000 -3.0000000 87.00000  
## 20010603 2.6322340 2.95440000 0.2770959 0.5209000 82.00000  
## 20010604 0.0000000 0.88981826 0.3473000 -0.1736000 92.00000  
## 20010605 4.9874793 -0.50000000 -2.9544000 -4.3301000 114.00000  
## 20010606 7.0000000 -5.63820000 -5.0000000 -6.0000000 94.00000  
## 20010607 5.3896471 -4.33010000 -1.8794000 -3.7588000 80.00000  
## 20010610 5.5929526 0.00000000 -1.0419000 -1.3892000 99.00000  
## 20010611 4.0000000 -0.76600000 -1.0261000 -2.2981000 79.00000  
## 20010612 9.4226799 1.28560000 -2.2981000 -3.9392000 101.00000  
## 20010613 3.0000000 -1.50000000 -1.5000000 -0.8682000 106.00000  
## 20010614 8.0000000 2.75649058 -1.0419000 -0.6946000 101.00000  
## 20010615 6.0000000 -0.86820000 -2.7362000 -6.8944000 90.00000  
## 20010616 6.7497561 -4.08462332 -7.8785000 -5.1962000 72.00000  
## 20010617 7.8909771 -4.33010000 -2.0521000 -3.0000000 70.00000  
## 20010618 4.0000000 0.52090000 -2.9544000 -1.0261000 83.00000  
## 20010620 1.0000000 -0.34200000 0.6908361 -0.6840000 121.00000  
## 20010621 3.9848829 0.00000000 0.3473000 -2.5712000 121.35598  
## 20010622 0.0000000 -1.70024007 -0.1132129 2.0000000 81.00000  
## 20010623 6.0022975 1.00000000 -1.9284000 -1.2155000 121.00000  
## 20010624 0.0000000 -2.79531646 -0.5209000 1.0261000 146.00000  
## 20010625 0.0000000 2.95440000 6.5778000 6.1517283 121.00000  
## 20010626 0.0000000 -2.57120000 -3.8567000 -4.6985000 146.00000  
## 20010627 5.2973781 -2.59810000 -6.6816360 -5.4329681 92.84556  
## 20010628 1.4700491 -5.63820000 -3.8302000 -4.5963000 83.00000  
## 20010629 4.0000000 -1.92840000 -2.5712000 -4.3301000 57.00000  
## 20010630 3.9504142 -1.53210000 -3.0642000 -0.8682000 81.00000  
## 20010701 1.0000000 0.68400000 0.0000000 1.3681000 67.00000  
## 20010702 1.0000000 2.81910000 3.9392000 3.4641000 70.00000  
## 20010703 4.0000000 1.87940000 2.0000000 1.3681000 106.00000  
## 20010704 4.7277580 0.69460000 -0.8660000 -1.0261000 139.00000  
## 20010705 6.0000000 0.00000000 0.0000000 1.2856000 79.00000  
## 20010706 4.0000000 0.00000000 1.7101000 0.9818756 93.00000  
## 20010707 5.2150357 -3.75880000 -3.9392000 -4.6985000 97.00000  
## 20010708 4.0000000 -2.59810000 -3.9392000 -3.7588000 113.00000  
## 20010709 1.8647691 -1.96960000 -3.0642000 -4.0000000 72.00000  
## 20010710 6.4387967 -2.81185538 -5.6382000 -9.0000000 88.00000  
## 20010711 6.0000000 -7.87850000 -6.8937000 -6.8937000 77.00000  
## 20010712 6.0000000 1.50000000 -3.8302000 -2.0521000 71.00000  
## 20010713 8.0000000 0.68400000 4.0000000 0.6626391 42.54133  
## 20010714 5.0000000 -3.21390000 -5.5982527 -4.9019260 45.00000  
## 20010715 6.0000000 -2.29810000 -3.7588000 0.0000000 67.00000  
## 20010716 6.0000000 0.00000000 -1.5297587 -2.5981000 67.00000  
## 20010717 6.0000000 2.00000000 -5.3623000 -6.1284000 84.00000  
## 20010718 10.3569505 -4.89258081 -3.8302000 -4.3301000 63.00000  
## 20010719 4.0000000 -2.05210000 -4.4995000 -2.7362000 69.00000  
## 20010720 5.9809787 -4.55231878 -3.4641000 -4.2908289 92.00000  
## 20010721 5.0000000 -3.00000000 -3.5000000 -2.9469776 88.00000  
## 20010722 6.0000000 -1.79852001 -1.9696000 -0.6715423 66.00000  
## 20010723 2.0000000 0.52090000 -1.0000000 -2.0000000 108.28321  
## 20010724 3.4694738 -1.29460832 -1.0261000 0.5209000 81.00000  
## 20010725 2.7503646 -0.87504463 -0.9397000 -0.6428000 83.00000  
## 20010726 4.0000000 0.93970000 1.5000000 -1.9779075 149.00000  
## 20010727 7.0000000 -0.34200000 1.2856000 -2.0000000 121.98777  
## 20010728 3.0000000 0.86600000 -1.5321000 -0.1736000 159.00000  
## 20010729 0.8664583 1.53210000 6.3519775 4.4360356 149.00000  
## 20010730 4.0000000 -0.50000000 -1.8794000 -1.2856000 160.00000  
## 20010731 2.0000000 -1.36810000 -2.7970010 0.0000000 156.00000  
## 20010801 4.0000000 3.00000000 3.7588000 1.1859718 84.00000  
## 20010802 8.0000000 0.00000000 -2.3941000 -1.3892000 126.00000  
## 20010803 6.0000000 -3.46410000 -2.5981000 -3.7588000 116.00000  
## 20010804 7.0000000 -5.00000000 -4.9240000 -5.6382000 63.51007  
## 20010805 8.0000000 -4.69850000 -2.5000000 -0.8682000 63.00000  
## 20010806 7.0000000 -3.83020000 -4.9240000 -5.6382000 54.00000  
## 20010807 8.0000000 -3.00000000 -4.5963000 -4.0521083 65.00000  
## 20010808 7.0000000 -5.63820000 -6.0622000 -6.8937000 72.00000  
## 20010809 3.0000000 -5.90880000 -3.2139000 -4.4995000 60.00000  
## 20010810 3.8636494 -1.92840000 -1.0261000 0.5209000 70.00000  
## 20010811 0.0000000 -0.83526428 -1.5321000 -1.0000000 95.06478  
## 20010812 2.0000000 -1.02610000 -3.0000000 -2.2981000 98.00000  
## 20010813 1.0000000 -0.86600000 0.0000000 0.0000000 109.66995  
## 20010814 5.0000000 1.87940000 1.3681000 0.6946000 75.00000  
## 20010815 4.0000000 -1.02610000 -1.7101000 -3.2139000 116.00000  
## 20010819 9.3353163 -4.11595910 -5.3623000 -2.5000000 86.00000  
## 20010820 5.0000000 -3.06420000 -2.2981000 -1.9489916 67.00000  
## 20010821 2.0025183 1.36810000 0.8682000 -2.2981000 76.00000  
## 20010822 4.0000000 1.53210000 1.9284000 1.9284000 113.00000  
## 20010823 3.0000000 0.17360000 -1.9696000 -1.9284000 117.00000  
## 20010824 1.0000000 0.64280000 -0.8660000 0.6840000 131.00000  
## 20010825 1.0000000 1.00000000 0.6946000 -1.7101000 166.00000  
## 20010826 6.0000000 1.28560000 -1.7321000 -0.6840000 131.12304  
## 20010827 5.0000000 3.06420000 2.8191000 1.3681000 100.00000  
## 20010828 3.0000000 4.00000000 4.0000000 3.7588000 114.00000  
## 20010829 7.0000000 -2.00000000 -0.5209000 1.8794000 112.00000  
## 20010830 7.0000000 -3.46410000 -4.0000000 -1.7321000 101.00000  
## 20010831 6.0000000 -4.33010000 -5.3623000 -1.4906798 76.00000  
## 20010901 6.3378610 -1.66838775 0.5209000 0.0000000 59.00000  
## 20010902 6.0000000 -2.95440000 -2.9544000 -2.0000000 76.16657  
## 20010903 5.0000000 -1.87940000 -4.5078295 -2.3941000 76.00000  
## 20010904 3.0000000 1.36790444 -1.7465061 -1.3892000 55.00000  
## 20010905 6.0000000 -1.36810000 -0.8682000 0.9575946 71.00000  
## 20010906 7.0000000 0.02537611 -1.9284000 -1.7101000 66.00000  
## 20010907 5.8004939 -1.50000000 -3.4641000 -3.0642000 59.00000  
## 20010908 7.7714034 -4.59630000 -6.0622000 -4.3301000 68.00000  
## 20010912 6.0000000 -0.86600000 -5.0000000 -3.3703805 62.00000  
## 20010913 7.0000000 -4.59630000 -6.8937000 -2.9810563 78.00000  
## 20010914 3.2844159 -1.04190000 -1.3681000 -1.5268286 74.00000  
## 20010915 8.0000000 -5.19620000 -2.7362000 -1.0419000 71.00000  
## 20010916 5.0000000 -3.83020000 0.0000000 1.3892000 69.00000  
## 20010917 4.0000000 0.00000000 3.2139000 0.0000000 71.00000  
## 20010918 7.0000000 -2.50000000 -3.2139000 -2.5000000 60.00000  
## 20010919 7.0000000 0.11938909 -6.0622000 -5.1962000 42.00000  
## 20010920 6.0000000 -3.93920000 -3.0642000 0.0000000 65.00000  
## 20010921 3.0000000 -0.17360000 3.7588000 3.8302000 71.00000  
## 20010922 2.0000000 4.00000000 5.0000000 2.7885922 96.00000  
## 20010923 6.0000000 5.19620000 5.1423000 4.6941194 98.00000  
## 20010924 7.8155362 -0.93970000 -0.7660000 -0.5000000 67.75141  
## 20010925 4.0626529 0.00000000 -1.0000000 -1.2856000 76.00000  
## 20010927 4.1355468 -0.69460000 -2.0000000 -2.7440205 71.00000  
## 20010928 3.6104475 1.50000000 0.8682000 0.8682000 87.02102  
## 20010929 3.0000000 -4.00000000 -3.7588000 -4.0000000 99.00000  
## 20010930 7.0000000 -2.22004557 -1.0419000 -4.0000000 83.00000

# the first imputed dataset

library(mice)

## Warning: package 'mice' was built under R version 3.5.1

## Loading required package: lattice

##   
## Attaching package: 'lattice'

## The following object is masked from 'package:UpSetR':  
##   
## histogram

##   
## Attaching package: 'mice'

## The following object is masked from 'package:tidyr':  
##   
## complete

## The following objects are masked from 'package:base':  
##   
## cbind, rbind

imp.mice <- mice(don,m=100,defaultMethod = "norm.boot")  
# the variability of the parameter is obtained

1. Now generate 100 imputed data sets with MIPCA method and 2 components. Store the result in a variable called res.MIPCA.

library(missMDA)  
?MIPCA  
?plot.MICPA

res.MIPCA <- MIPCA(don,ncp=2,nboot=100)  
# MI with PCA using 2 dimensions

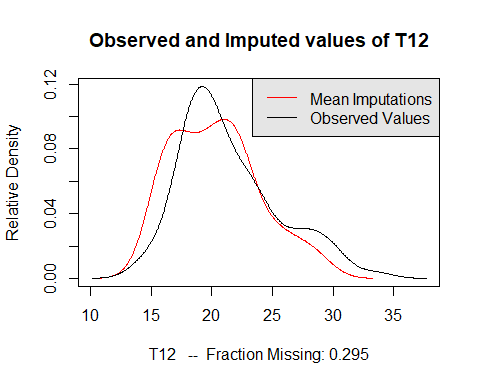
The function MIPCA gives as output the data set imputed by the iterative PCA algorithm (in res.imputedPCA), and the other data sets generated by the MIPCA algorithm(in res.MI). The number of data sets generated by this algorithm is controlled by the nboot argument, equal to 100 by default. The other arguments of this function are same as those for the imputePCA function.

### Inspect the imputed values

Exploratory analysis is very important and even at this stage of the analysis.

We will **inspect the imputed values created** to explorer the correlation between variables. Usr the R package missMDA dedicated to perform principal components methods with missing values to impute data with PC methods.

library(mice)  
library(Amelia)  
compare.density(res.amelia,var="T12")



**Q** Do both distributions need to be close? Could the missing values differ from the observed ones both in spread and in location?

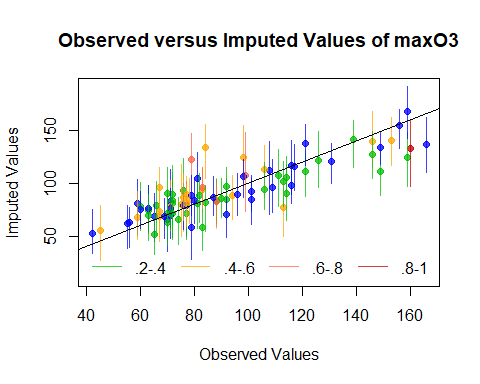
Note that a difference between these distributions does not mean that the model is unsuitable.

Indeed, when the missing data mechanism is not MCAR, it could make sense to observe differences between the distribution of imputed values and the distribution of observed values. However, if differences occur, more investigations would be required to try to explain them.

The quality of imputation can also be assessed with cross-validation using the **overimpute** function. Each observed value is deleted and for each one 100 values are predicted (using the same MI method) and the mean and 90% confidence intervals are computed for these 100 values.

Then, we inspect whether the observed value falls within the obtained interval. On the graph, the y=x line is plotted (where the imputations should fall if they were perfect), as well as the mean (dots) and intervals (lines) for each value. Around ninety percent of these confidence intervals should contain the y = x line, which means that the true observed value falls within this range. The color of the line (as coded in the legend) represents the fraction of missing observations in the pattern of missingness for that observation (ex: blue=0-2 missing entries).

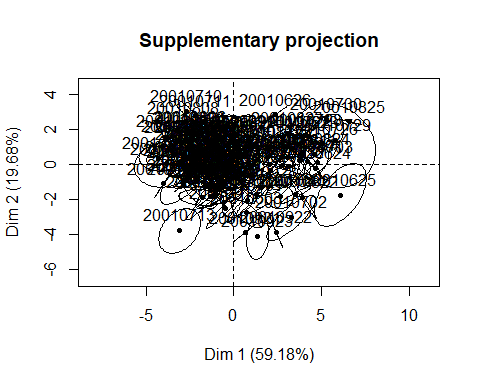
overimpute(res.amelia,var="maxO3")



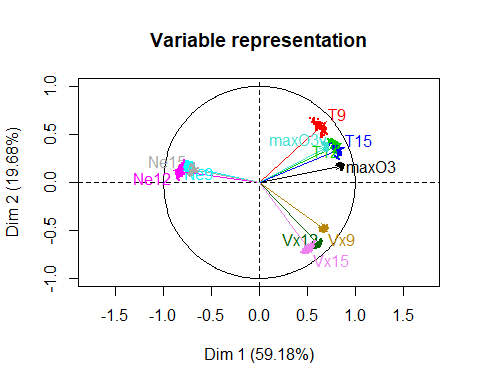
* Comment the quality of the imputation.

We can also examine the variability by projecting as supplementary tables the imputed data sets on the PCA configuration (plot the results of MI with PCA).

plot(res.MIPCA,choice="ind.supp")



plot(res.MIPCA,choice="var")



The plots represent the projection of the individuals (top) and variables (bottom) of each imputed data set as supplementary elements onto the reference configuration obtained with the iterative PCA algorithm. For the individuals, a confidence area is constructed for each, and if one has no missing entries, its confidence area is restricted to a point. All the plots show that the variability across different imputations is small and a user can interpret the PCA results with confidence.

### Perform regression

MI aims to apply a statistical method on an incomplete data set. We now apply a regression model on each imputed data set of the amelia method and MIPCA methods.

resamelia <- lapply(res.amelia$imputations,as.data.frame)  
head(resamelia$imp1)

## maxO3 T9 T12 T15 Ne9 Ne12 Ne15  
## 20010601 87 15.60000 18.50000 18.17613 4.00000 4.000000 8.000000  
## 20010602 82 18.30995 19.75505 19.81190 5.00000 5.000000 7.000000  
## 20010603 92 15.30000 17.60000 19.50000 2.00000 5.672168 2.632234  
## 20010604 114 16.20000 19.70000 23.45268 1.00000 1.000000 0.000000  
## 20010605 94 18.23216 20.50000 20.40000 2.77245 3.857982 4.987479  
## 20010606 80 17.70000 19.80000 18.30000 6.00000 2.649196 7.000000  
## Vx9 Vx12 Vx15 maxO3v  
## 20010601 0.6946000 -1.7101000 -0.6946 84  
## 20010602 -4.3301000 -4.0000000 -3.0000 87  
## 20010603 2.9544000 0.2770959 0.5209 82  
## 20010604 0.8898183 0.3473000 -0.1736 92  
## 20010605 -0.5000000 -2.9544000 -4.3301 114  
## 20010606 -5.6382000 -5.0000000 -6.0000 94

# a regression on each imputed dataset  
fitamelia <- lapply(resamelia,lm, formula="maxO3~T9+T12+Ne9+Ne12+Ne15+Vx9+Vx12+Vx15+maxO3v")  
  
# fitamelia <- lapply(resamelia, with, lm(maxO3 ~T9+T12+T15+Ne9+Ne12+Ne15+Vx9+Vx12+Vx15+maxO3v))

library(mice)  
imp.mice <- mice(don,m=100,defaultMethod="norm.boot")  
# the variability of the parameters is obtained  
  
lm.mice.out <- with(imp.mice,lm(maxO3~T9+T12+T15+Ne9+Ne12+Ne15+Vx9+Vx12+Vx15+maxO3v))

res.MIPCA <- lapply(res.MIPCA$res.MI,as.data.frame)  
fitMIPCA <- lapply(res.MIPCA,lm,formula="maxO3~T9+T12+T15+Ne9+Ne12+Ne15+Vx9+Vx12+Vx15+maxO3v")

* Aggregate the results of regression with multiple imputation according to Rubin’s rule for MI with amelia, and with PCA with the **pool** function from the mice package.

poolamelia <- pool(as.mira(fitamelia))  
summary(poolamelia)

## estimate std.error statistic df p.value  
## (Intercept) 26.5207819 15.09845441 1.75652297 13.408985 8.275302e-02  
## T9 -1.6669811 1.91925885 -0.86855458 17.018876 3.876397e-01  
## T12 3.8497870 1.44271852 2.66842555 20.397490 9.191726e-03  
## Ne9 -1.8554017 1.53522880 -1.20855062 6.884181 2.303282e-01  
## Ne12 -2.3732135 2.31240329 -1.02629743 7.027553 3.077860e-01  
## Ne15 0.4353628 1.06002585 0.41070958 13.209919 6.823627e-01  
## Vx9 -0.5127813 1.15345581 -0.44456083 20.895909 6.578148e-01  
## Vx12 1.0988270 1.14290744 0.96143127 63.021995 3.391774e-01  
## Vx15 -0.1007391 1.07123314 -0.09404029 24.866019 9.253078e-01  
## maxO3v 0.3613850 0.07816011 4.62365053 81.550756 1.397694e-05

poolMIPCA <- pool(as.mira(fitMIPCA))  
summary(poolMIPCA)

## estimate std.error statistic df p.value  
## (Intercept) 13.1534322 17.62415929 0.7463296 68.73423 0.45794881  
## T9 0.8627325 1.38646785 0.6222521 39.84176 0.53578262  
## T12 1.6069351 1.09421026 1.4685798 45.79516 0.14639286  
## T15 0.8856658 0.86455787 1.0244147 59.44769 0.30914020  
## Ne9 -1.2251962 1.17825374 -1.0398407 55.16545 0.30196548  
## Ne12 -1.6388678 1.50893817 -1.0861067 52.60697 0.28112996  
## Ne15 0.3316824 1.13020419 0.2934713 64.65598 0.77002378  
## Vx9 0.7165621 1.06049162 0.6756886 70.53044 0.50144786  
## Vx12 0.8927738 1.17961832 0.7568328 64.85822 0.45167089  
## Vx15 0.3744586 1.17412661 0.3189252 59.65199 0.75072583  
## maxO3v 0.2508858 0.09126871 2.7488691 68.29076 0.00758891

# pool.mice <- pool(lm.mice.out)  
# summary(pool.mice)

* Write a function that removes the variables with the largest p-values step by step (each time a variable is removed th regression model is performed again) until all variables are significant.

don2 <- don  
reg <- lm(maxO3~.,data=don2)  
while(any(summary(reg)$coeff[-1,4]>0.05)){  
 don2 <- don2[,!(colnames(don2)%in%names(which.max(summary(reg)$coeff[-1,4])))]  
 reg <- lm(maxO3~.,data=don2)  
}

We combine the results and perform the **regression with missing values**

#submodel to compare  
  
library(mice)  
  
fitMIPCA <- lapply(res.MIPCA,lm,formula="maxO3~T12+Ne9+Vx12+maxO3v")  
poolMIPCA <- pool(as.mira(fitMIPCA))  
summary(poolMIPCA)

## estimate std.error statistic df p.value  
## (Intercept) 9.9668321 14.08147055 0.7077977 67.04595 4.812414e-01  
## T12 2.9635292 0.60616069 4.8890157 69.17399 5.533895e-06  
## Ne9 -1.8903643 1.05347329 -1.7944112 62.39923 7.673427e-02  
## Vx12 1.8041858 0.72345186 2.4938575 74.52189 1.481514e-02  
## maxO3v 0.3240862 0.08180795 3.9615494 75.80784 1.668874e-04

# lm.mice.out <- with(imp.mice,lm(maxO3))  
# pool.mice <- pool(lm.mice.out)  
# summary(pool.mice)  
fitamelia <- lapply(resamelia,lm, formula="maxO3~T12+Ne9+Vx12+maxO3v")  
poolamelia <- pool(as.mira(fitamelia))  
summary(poolamelia)

## estimate std.error statistic df p.value  
## (Intercept) 14.3779751 12.08991980 1.189253 21.32678 0.2474129462  
## T12 2.9610541 0.59537504 4.973427 20.25828 0.0000611678  
## Ne9 -3.0529870 0.87612316 -3.484655 18.10336 0.0021731479  
## Vx12 0.9297542 0.76890235 1.209197 16.57584 0.2398223057  
## maxO3v 0.3319225 0.08398547 3.952141 14.99262 0.0007110601

## 1.5) Ecological example

Studies in community ecology aim to understand how and why individuals of different species co-occur in the same location at the same time. Hence, ecologists usually collect and store data on species distribution as tables containing the abundances of different species in several sampling sites. Additional information such as measures of environmental variables or species traits can also be recorded to examine the effects of abiotic features (characteristics, i.e. due to physico-chemical action and no biological action) and biotic features.

Several projects compile data from preexisting databases. Due to the wide heterogeneity of measurement methods and research objectives, these huge data sets are often characterized by a high number of missing values. Hence, in addition to ecological questions, such data sets also present some important methodological and technical challenges for multivariate analysis.

The GLOPNET data set contains 6 traits measured for 2494 plant species: LMA (leaf mass per area), LL (leaf lifes-pan), Amass (photosynthetic assimilation), Nmass (leaf nitrogen), Pmass (leaf phosphorus), Rmass (dark respiration rate). The last four variables are expressed per leaf dry mass. GLOPNET is a compilation of several existing data sets and thus contains a large proportion of missing values. All traits were log-normally distributed and log-transformed before analysis.

Ecolo <- read.csv("C:/Users/kojikm.mizumura/Desktop/Data Science/8. UseR2018/1 NA Treatment/ecological.csv",header=T,sep=";",dec=",")

Lets delete species with only missing values for continuous variables

ind <- which(rowSums(is.na(Ecolo[,-1]))==6)  
biome <- Ecolo[-ind,1] ### Keep a categorical variables  
Ecolo <- Ecolo[-ind,-1] ### Select continuous variables  
dim(Ecolo)

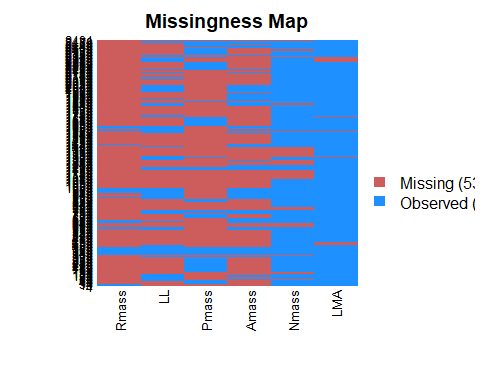
## [1] 2494 6

# proportion of missing values  
sum(is.na(Ecolo))/(nrow(Ecolo)\*ncol(Ecolo))

## [1] 0.5338145

## 55% of missing values

# delete species with missing values  
library(Amelia)  
missmap(Ecolo)



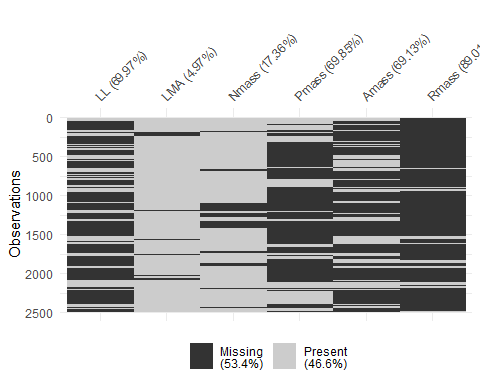
dim(na.omit(Ecolo))

## [1] 72 6

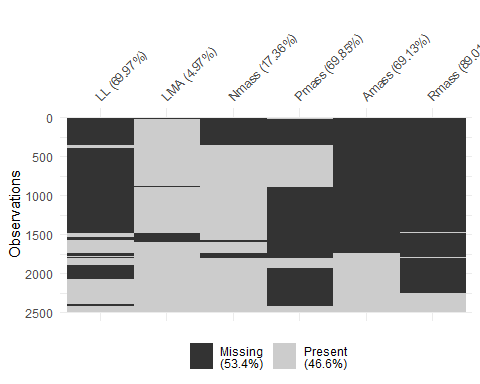
# only 72 remining species..

of the entires in the GLOPNET data set are missing. Only 72 species have complete information for the 6 traits and the proportion of missing values varied between 4.97% (LMA) to 89.01%

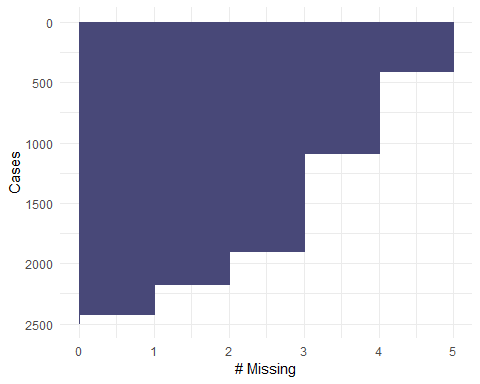
vis\_miss(Ecolo)



vis\_miss(Ecolo,cluster=T)



gg\_miss\_case(Ecolo)



gg\_miss\_var(Ecolo)

library(VIM)  
library(devtools)

## Warning: package 'devtools' was built under R version 3.5.1

library(naniar)  
library(missMDA)  
library(Amelia)  
library(mice)  
library(missForest)

## Warning: package 'missForest' was built under R version 3.5.1

## Loading required package: randomForest

## Warning: package 'randomForest' was built under R version 3.5.1

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:ggplot2':  
##   
## margin

## Loading required package: foreach

##   
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':  
##   
## accumulate, when

## Loading required package: itertools

## Warning: package 'itertools' was built under R version 3.5.1

## Loading required package: iterators

library(FactoMineR)  
library(tidyverse)  
library(missForest)  
  
# gg\_miss\_upset(Ecolo,  
# order.by = "freq")

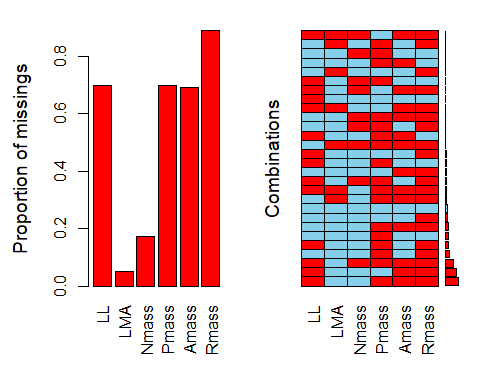
miss\_case\_table(Ecolo)

## # A tibble: 6 x 3  
## n\_miss\_in\_case n\_cases pct\_miss  
## <int> <int> <dbl>  
## 1 0 72 2.89  
## 2 1 248 9.94  
## 3 2 277 11.1   
## 4 3 807 32.4   
## 5 4 685 27.5   
## 6 5 405 16.2

miss\_var\_table(Ecolo)

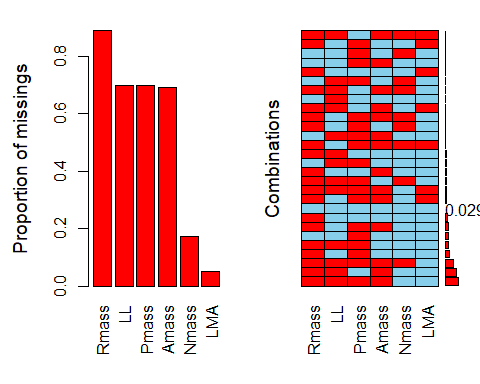
## # A tibble: 6 x 3  
## n\_miss\_in\_var n\_vars pct\_miss  
## <int> <int> <dbl>  
## 1 124 1 16.7  
## 2 433 1 16.7  
## 3 1724 1 16.7  
## 4 1742 1 16.7  
## 5 1745 1 16.7  
## 6 2220 1 16.7

# visualize the pattern  
library(VIM)  
aggr(Ecolo)



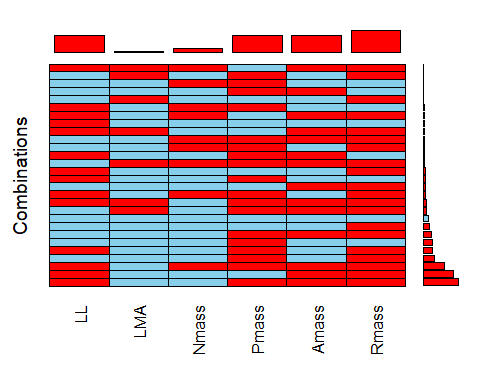
aggr(Ecolo,only.miss=T,numbers=T,sortVar=T)

## Warning in plot.aggr(res, ...): not enough vertical space to display  
## frequencies (too many combinations)



##   
## Variables sorted by number of missings:   
## Variable Count  
## Rmass 0.89013633  
## LL 0.69967923  
## Pmass 0.69847634  
## Amass 0.69125902  
## Nmass 0.17361668  
## LMA 0.04971933

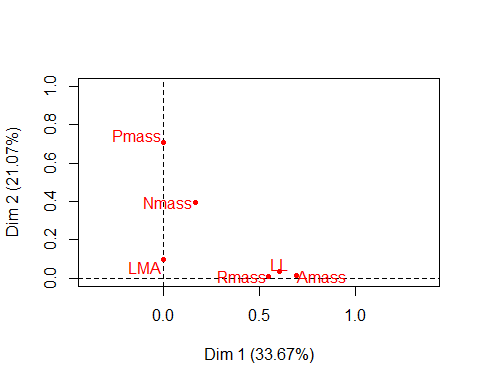
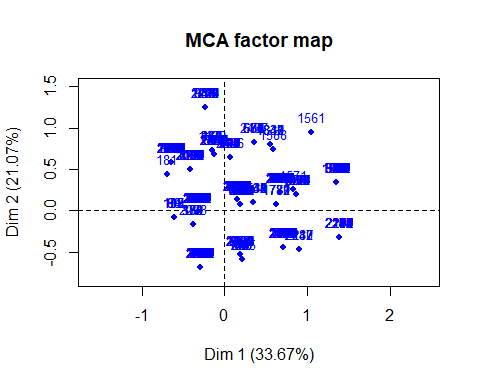
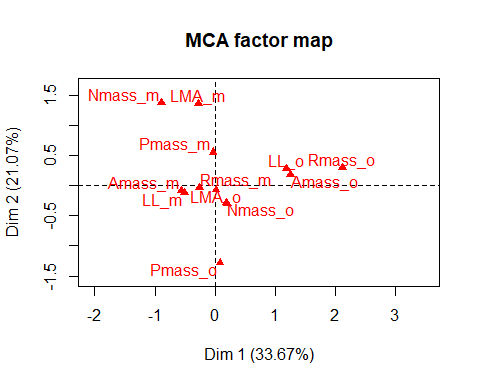
res <- summary(aggr(Ecolo,prop=TRUE,combined=TRUE))$combinations



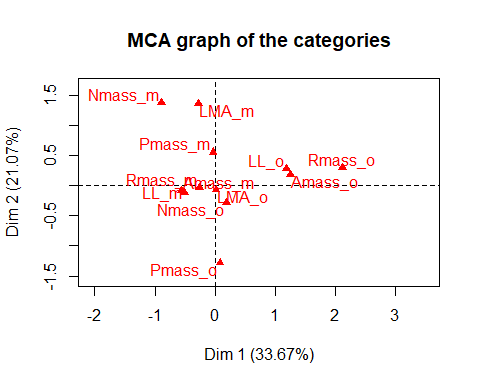
res[rev(order(res[,2])),]

## Combinations Count Percent  
## 21 1:0:0:1:1:1 580 23.25581395  
## 17 1:0:0:0:1:1 495 19.84763432  
## 25 1:0:1:1:1:1 339 13.59262229  
## 5 0:0:0:1:0:1 178 7.13712911  
## 19 1:0:0:1:0:1 147 5.89414595  
## 4 0:0:0:1:0:0 143 5.73376103  
## 7 0:0:0:1:1:1 131 5.25260626  
## 2 0:0:0:0:0:1 99 3.96952686  
## 1 0:0:0:0:0:0 72 2.88692863  
## 27 1:1:0:1:1:1 45 1.80433039  
## 13 0:1:0:1:1:1 45 1.80433039  
## 24 1:0:1:1:0:1 38 1.52365678  
## 18 1:0:0:1:0:0 31 1.24298316  
## 3 0:0:0:0:1:1 31 1.24298316  
## 16 1:0:0:0:0:1 30 1.20288693  
## 14 0:1:1:1:1:1 20 0.80192462  
## 20 1:0:0:1:1:0 14 0.56134723  
## 9 0:0:1:1:0:1 13 0.52125100  
## 10 0:0:1:1:1:1 9 0.36086608  
## 26 1:1:0:0:1:1 7 0.28067362  
## 23 1:0:1:1:0:0 6 0.24057739  
## 22 1:0:1:0:1:1 6 0.24057739  
## 15 1:0:0:0:0:0 6 0.24057739  
## 11 0:1:0:0:0:1 5 0.20048115  
## 28 1:1:1:0:1:1 1 0.04009623  
## 12 0:1:0:1:0:1 1 0.04009623  
## 8 0:0:1:1:0:0 1 0.04009623  
## 6 0:0:0:1:1:0 1 0.04009623

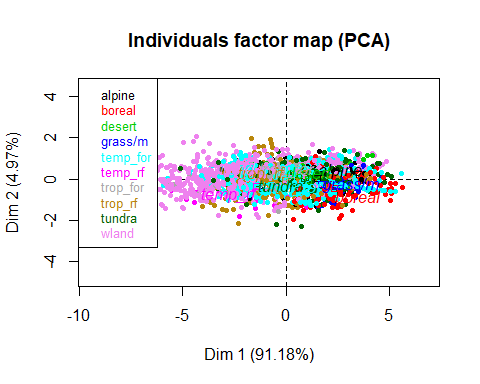
mis.ind <- matrix("o",nrow=nrow(Ecolo),ncol=ncol(Ecolo))  
mis.ind[is.na(Ecolo)] <- "m"  
dimnames(mis.ind) <- dimnames(Ecolo)  
library(FactoMineR)  
resMCA <- MCA(mis.ind)



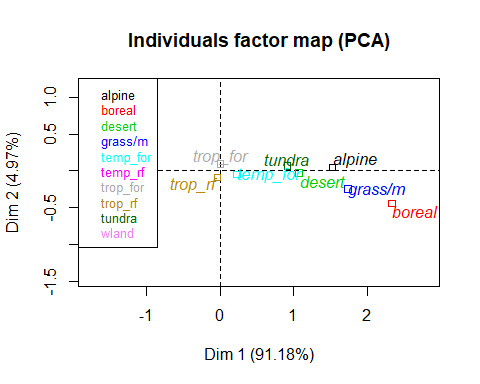
plot(resMCA,invis="ind",title="MCA graph of the categories")



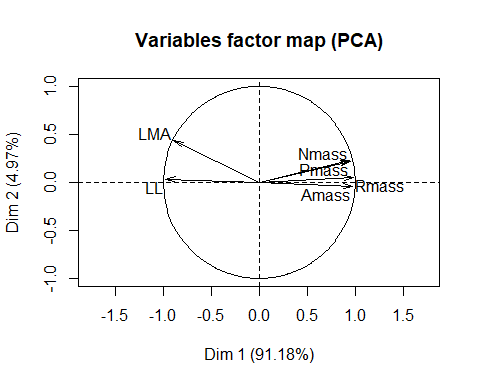
library(missMDA)  
### nb <- estim\_ncpPCA(Ecolo,method.cv="Kfold",nbsim=100) ### Time consuming!  
res.comp <- imputePCA(Ecolo,ncp=2)  
  
#Perform a PCA on the completed data set  
imp <- cbind.data.frame(res.comp$completeObs,biome)  
res.pca <- PCA(imp,quali.sup=7,graph=FALSE)  
plot(res.pca, hab=7, lab="quali")



plot(res.pca, hab=7, lab="quali",invisible="ind")

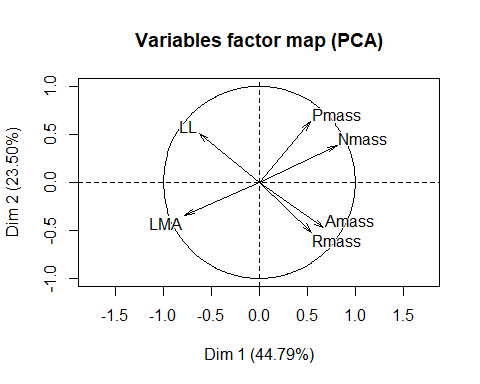
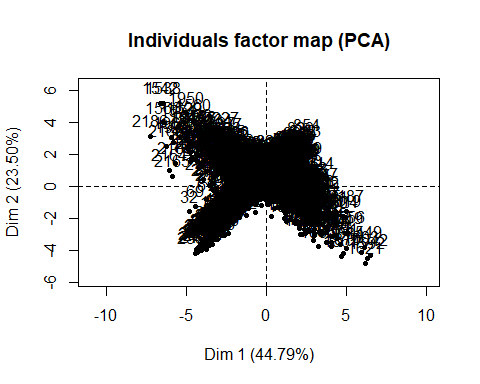


plot(res.pca, choix="var")



# Compare with PCA on the data imputed by the mean  
PCA(Ecolo)

## Warning in PCA(Ecolo): Missing values are imputed by the mean of the  
## variable: you should use the imputePCA function of the missMDA package



## \*\*Results for the Principal Component Analysis (PCA)\*\*  
## The analysis was performed on 2494 individuals, described by 6 variables  
## \*The results are available in the following objects:  
##   
## name description   
## 1 "$eig" "eigenvalues"   
## 2 "$var" "results for the variables"   
## 3 "$var$coord" "coord. for the variables"   
## 4 "$var$cor" "correlations variables - dimensions"  
## 5 "$var$cos2" "cos2 for the variables"   
## 6 "$var$contrib" "contributions of the variables"   
## 7 "$ind" "results for the individuals"   
## 8 "$ind$coord" "coord. for the individuals"   
## 9 "$ind$cos2" "cos2 for the individuals"   
## 10 "$ind$contrib" "contributions of the individuals"   
## 11 "$call" "summary statistics"   
## 12 "$call$centre" "mean of the variables"   
## 13 "$call$ecart.type" "standard error of the variables"   
## 14 "$call$row.w" "weights for the individuals"   
## 15 "$call$col.w" "weights for the variables"

This first axis corresponding to the “leaf economic spectrum” separates species with potential for quick returns for investment with high values for Nmass, Amass, Rmass and Pmass and low values for LL and LMA (right part) from species with slow returns on the left part. Scores for the traits are very consistent between methods, to a lessert extent for the Mean.

This representation can be used to add external information: grouping species by major biomes illustrates the universality of the leaf economic spectrum but also some specificities (e.g., Desert and Boreal forest mainly contain species of the quick-return end).

The graphical representation obtained by the Mean imputation highlights a very particular shape indicating that results are not reliable.

# 2) Categorical/mixed/multi-block data with missing values

## 2.1) Single imputation of categorical data with MCA/MCA with missing values

We use the survey data set health concerning students’ health. 320 students answered 20 questions on their consumption of products (drugs, alcohol), on their psychological state and their sleeping condition. In addition, we have information regarding their gender, age and accommodation.

The aim is to study the principal dimensions of variability of this data and to see if there are relationships between alcohol consumption and psychological state for instance. Then, after grouping individuals with the same profile, one can “label” them and see if there are relationships with the socio-economic questions.

Missing values are inserted to illustrate the methods.

# load FactoMineR package  
library(FactoMineR)  
library(magrittr)

##   
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':  
##   
## set\_names

## The following object is masked from 'package:tidyr':  
##   
## extract

# raw data check  
health <- read.csv("C:/Users/kojikm.mizumura/Desktop/Data Science/8. UseR2018/1 NA Treatment/sante.tex",sep=";",header=T)  
dim(health)

## [1] 327 20

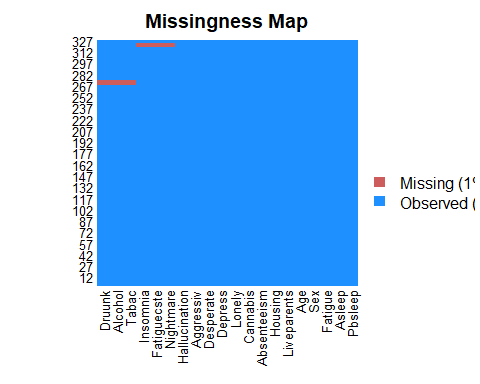
summary(health)

## Pbsleep Asleep Fatigue Nightmare   
## Never:136 Never : 66 Never : 41 Never :172   
## Often: 70 QuiteOften: 86 QuiteOften:159 QuiteOften: 28   
## Rare :121 Rare :142 Rare : 91 Rare :119   
## Veryoften : 33 Veryoften : 36 Veryoften : 8   
##   
##   
##   
## Fatiguecste Insomnia Sex Age   
## Never : 73 Never :250 Boy :120 18yrsorless:147   
## QuiteOften: 98 QuiteOften: 16 Girls:207 19yrs : 99   
## Rare :139 Rare : 54 20yrs : 49   
## Veryoften : 17 Veryoften : 7 21yrsetplus: 32   
##   
##   
##   
## Liveparents Housing Absenteeism Tabac   
## No :186 Campus : 31 Exceptionally:112 Frequent : 81   
## Yes:141 Flat\_alone : 59 Never :129 Never :208   
## FurnishedRoom: 18 Often : 21 Occasional: 38   
## Hostel : 10 Sometimes : 57   
## Other : 8 Veryoften : 8   
## Parents :141   
## Roomate : 60   
## Alcohol Druunk Cannabis Lonely Depress   
## Frequent : 27 Never:122 Frequent : 22 Never:128 Never:119   
## Never : 42 Yes :205 Never :165 Often: 55 Often: 54   
## Ocasional:258 Occasional:140 Rare :144 Rare :154   
##   
##   
##   
##   
## Desperate Aggressiv Hallucination  
## Never:184 Never:182 Never :314   
## Often: 52 Often: 31 RareorOften: 13   
## Rare : 91 Rare :114   
##   
##   
##   
##

healthNA <- health  
  
# Omitting values  
healthNA[5:10,4:6] <- NA  
healthNA[55:60,12:14] <- NA  
  
# check the updated data  
head(healthNA)

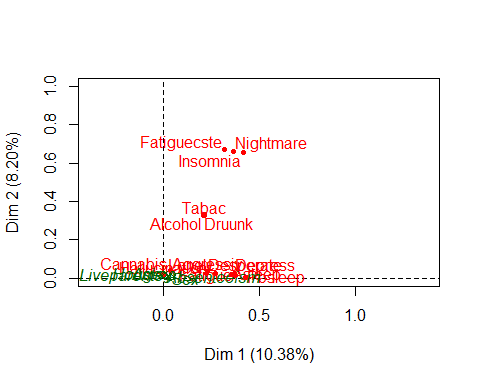
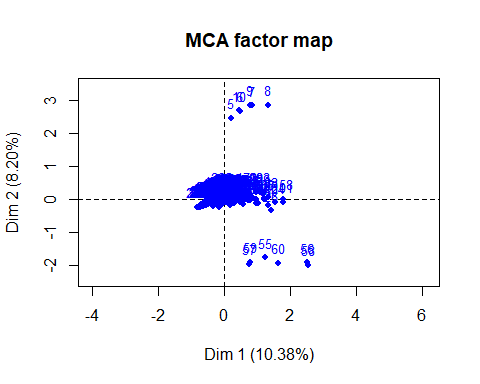
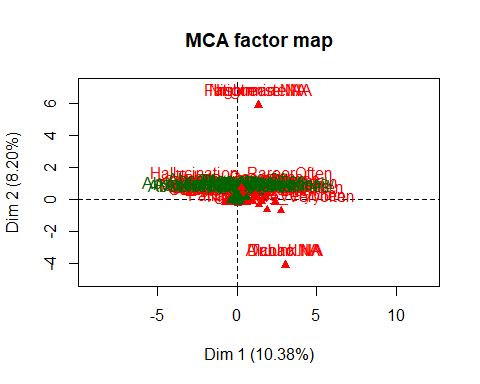
## Pbsleep Asleep Fatigue Nightmare Fatiguecste Insomnia Sex  
## 1 Never Rare Rare Never Never Never Boy  
## 2 Rare Never QuiteOften Never Never Never Girls  
## 3 Rare Never Never Rare Never Never Girls  
## 4 Often Veryoften QuiteOften Never QuiteOften Never Girls  
## 5 Often Never QuiteOften <NA> <NA> <NA> Girls  
## 6 Never Rare QuiteOften <NA> <NA> <NA> Girls  
## Age Liveparents Housing Absenteeism Tabac Alcohol  
## 1 20yrs No Other Never Frequent Ocasional  
## 2 20yrs Yes Parents Exceptionally Frequent Ocasional  
## 3 21yrsetplus No Other Veryoften Occasional Ocasional  
## 4 18yrsorless No Flat\_alone Never Occasional Ocasional  
## 5 19yrs Yes Parents Never Never Never  
## 6 19yrs No Campus Never Never Ocasional  
## Druunk Cannabis Lonely Depress Desperate Aggressiv Hallucination  
## 1 Yes Frequent Never Never Never Rare Never  
## 2 Yes Never Never Rare Never Never Never  
## 3 Yes Occasional Never Rare Never Never Never  
## 4 Never Never Often Rare Often Rare Never  
## 5 Never Never Never Never Never Never Never  
## 6 Yes Occasional Rare Often Never Never Never

Amelia::missmap(healthNA)



First, we can explorer the pattern of missing using MCA (by default it codes a missing values as a new category):

res.mcaNA <- MCA(healthNA,quali.sup=c(7:11))



We can also explorer some of the healthNA missingness using tools from naniar: *vis\_miss* gg\_miss\_var \*gg\_miss\_case

Then, we can study the similarities between the students and the associations between categories performing MCA while skipping the missing values. We carry-out the following steps:

res.impute <- imputeMCA(health[,c(1:6,12:20)],ncp=5)  
res.impute$tab.disj[1:10,10:21]

## Rare Veryoften Never QuiteOften Rare Veryoften Never QuiteOften Rare  
## 1 1 0 1 0 0 0 1 0 0  
## 2 0 0 1 0 0 0 1 0 0  
## 3 0 0 0 0 1 0 1 0 0  
## 4 0 0 1 0 0 0 0 1 0  
## 5 0 0 0 1 0 0 0 1 0  
## 6 0 0 0 0 1 0 0 0 1  
## 7 0 0 0 0 0 1 0 1 0  
## 8 0 1 1 0 0 0 0 1 0  
## 9 0 0 0 0 1 0 0 0 1  
## 10 0 0 1 0 0 0 0 0 1  
## Veryoften Never QuiteOften  
## 1 0 1 0  
## 2 0 1 0  
## 3 0 1 0  
## 4 0 1 0  
## 5 0 0 1  
## 6 0 1 0  
## 7 0 0 1  
## 8 0 1 0  
## 9 0 0 0  
## 10 0 1 0

apply(res.impute$tab.disj[1:10,12:15],1,sum)

## 1 2 3 4 5 6 7 8 9 10   
## 1 1 1 1 1 1 1 1 1 1

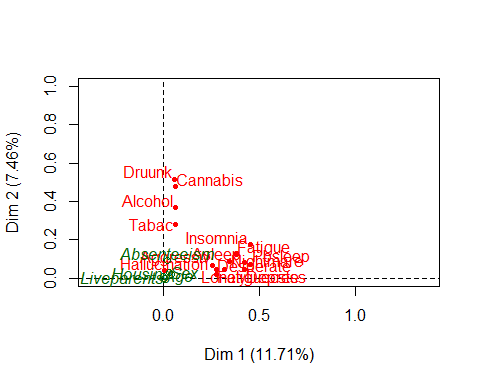
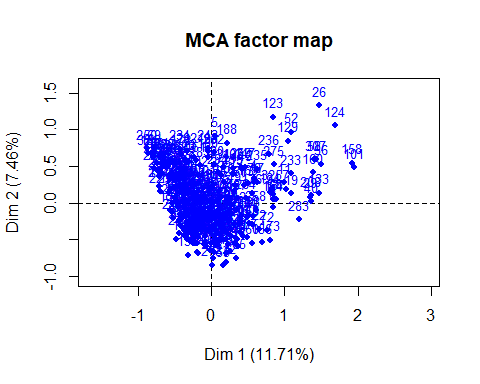
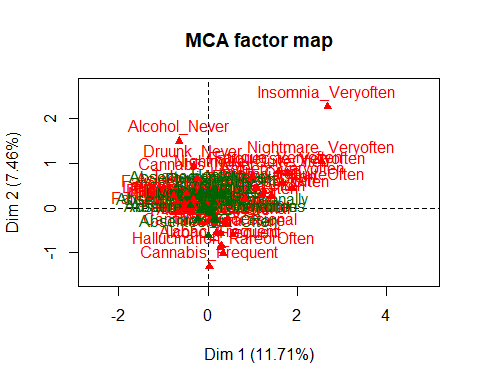
# sum to 1 per variable  
res.impute$comp[5:10,4:6]

## Nightmare Fatiguecste Insomnia  
## 5 QuiteOften QuiteOften QuiteOften  
## 6 Rare Rare Never  
## 7 Veryoften QuiteOften QuiteOften  
## 8 Never QuiteOften Never  
## 9 Rare Rare Rare  
## 10 Never Rare Never

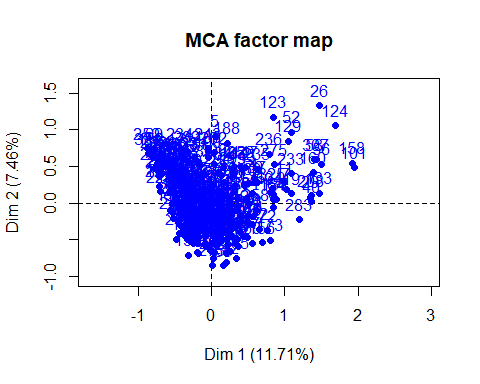
# the completed dataset with the most plausible category  
health[5:10,4:6]

## Nightmare Fatiguecste Insomnia  
## 5 QuiteOften QuiteOften QuiteOften  
## 6 Rare Rare Never  
## 7 Veryoften QuiteOften QuiteOften  
## 8 Never QuiteOften Never  
## 9 Rare Rare Rare  
## 10 Never Rare Never

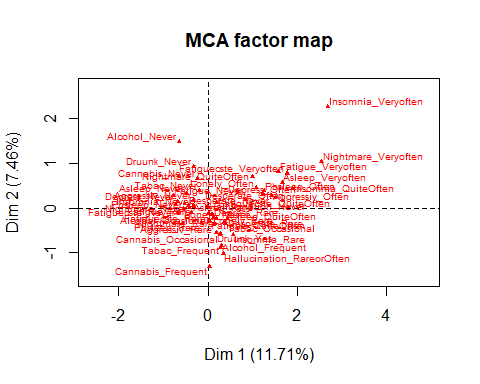
## The imputed indicator matrix can be used as an input of the MCA function of the FactoMineR to perform the MCA on the incomplete data  
res.mca <- MCA(healthNA,tab.disj=res.impute$tab.disj,quali.sup=7:11)



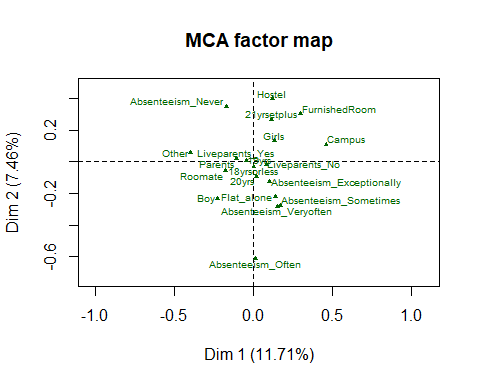
plot(res.mca, invisible=c("var","quali.sup"))



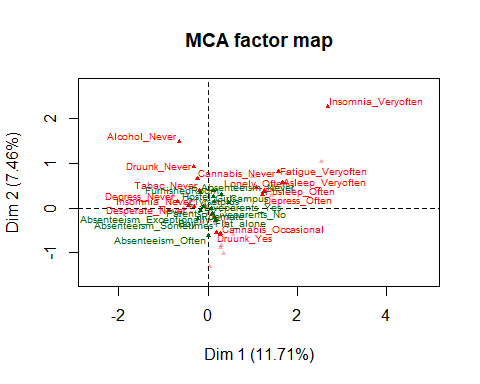
plot(res.mca, invisible=c("ind","quali.sup"), cex = 0.6)



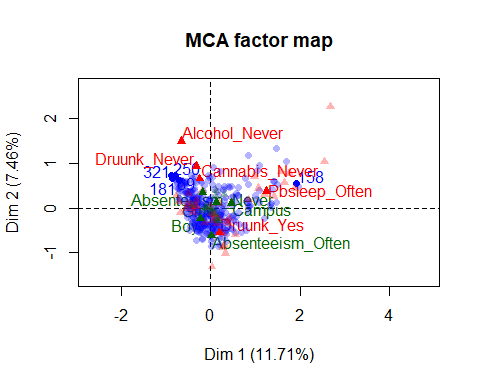
plot(res.mca, invisible=c("ind","var"), cex = 0.6)



plot(res.mca,invisible=c("ind"),autoLab="yes", selectMod="cos2 15", cex = 0.6)



plot(res.mca,autoLab="yes", selectMod="cos2 5", select="cos2 5")



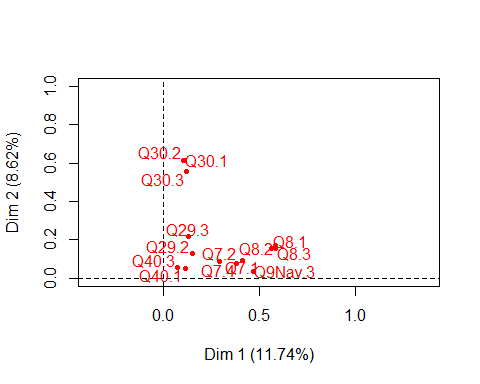
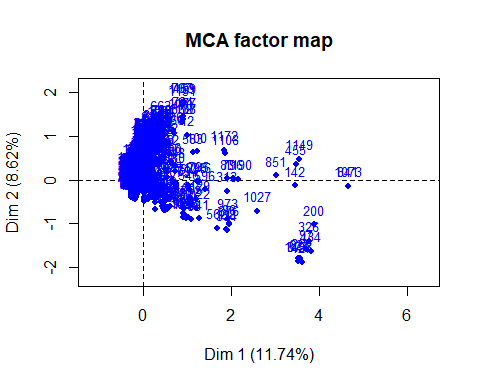
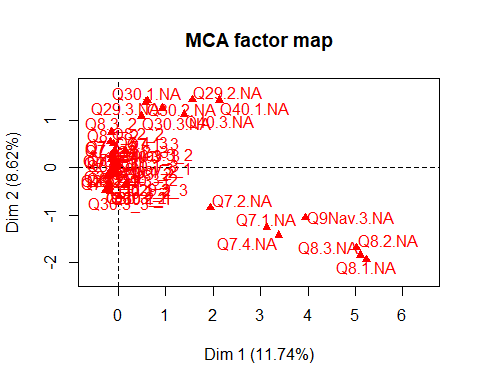
res.mca

## \*\*Results of the Multiple Correspondence Analysis (MCA)\*\*  
## The analysis was performed on 327 individuals, described by 20 variables  
## \*The results are available in the following objects:  
##   
## name   
## 1 "$eig"   
## 2 "$var"   
## 3 "$var$coord"   
## 4 "$var$cos2"   
## 5 "$var$contrib"   
## 6 "$var$v.test"   
## 7 "$ind"   
## 8 "$ind$coord"   
## 9 "$ind$cos2"   
## 10 "$ind$contrib"   
## 11 "$quali.sup"   
## 12 "$quali.sup$coord"   
## 13 "$quali.sup$cos2"   
## 14 "$quali.sup$v.test"  
## 15 "$call"   
## 16 "$call$marge.col"   
## 17 "$call$marge.li"   
## description   
## 1 "eigenvalues"   
## 2 "results for the variables"   
## 3 "coord. of the categories"   
## 4 "cos2 for the categories"   
## 5 "contributions of the categories"   
## 6 "v-test for the categories"   
## 7 "results for the individuals"   
## 8 "coord. for the individuals"   
## 9 "cos2 for the individuals"   
## 10 "contributions of the individuals"   
## 11 "results for the supplementary categorical variables"  
## 12 "coord. for the supplementary categories"   
## 13 "cos2 for the supplementary categories"   
## 14 "v-test for the supplementary categories"   
## 15 "intermediate results"   
## 16 "weights of columns"   
## 17 "weights of rows"

## Another example of imputation of categorical data  
data(vnf)  
head(vnf)

## Q7.1 Q7.2 Q7.4 Q8.1 Q8.2 Q8.3 Q9Nav.3 Q29.2 Q29.3 Q30.1 Q30.2 Q30.3  
## 1 1 1 3 1 2 2 2 1 2 <NA> <NA> 2  
## 2 1 1 1 2 1 1 2 1 3 2 <NA> <NA>  
## 3 1 1 1 2 2 2 2 1 2 1 1 2  
## 4 1 1 1 1 1 1 1 1 1 1 1 1  
## 5 1 3 1 1 1 1 1 2 2 1 2 2  
## 6 1 1 1 1 2 1 1 1 1 <NA> <NA> <NA>  
## Q40.1 Q40.3  
## 1 3 2  
## 2 2 2  
## 3 3 3  
## 4 3 1  
## 5 2 2  
## 6 2 1

# Look at the pattern of missing values with MCA  
MCA(vnf)



## \*\*Results of the Multiple Correspondence Analysis (MCA)\*\*  
## The analysis was performed on 1232 individuals, described by 14 variables  
## \*The results are available in the following objects:  
##   
## name description   
## 1 "$eig" "eigenvalues"   
## 2 "$var" "results for the variables"   
## 3 "$var$coord" "coord. of the categories"   
## 4 "$var$cos2" "cos2 for the categories"   
## 5 "$var$contrib" "contributions of the categories"   
## 6 "$var$v.test" "v-test for the categories"   
## 7 "$ind" "results for the individuals"   
## 8 "$ind$coord" "coord. for the individuals"   
## 9 "$ind$cos2" "cos2 for the individuals"   
## 10 "$ind$contrib" "contributions of the individuals"  
## 11 "$call" "intermediate results"   
## 12 "$call$marge.col" "weights of columns"   
## 13 "$call$marge.li" "weights of rows"

#1) Select the number of components  
# nb <- estim\_ncpMCA(vnf,ncp.max=5)   
# Time-consuming, nb=$  
  
#2) Impute the indicator matrix  
res.impute <- imputeMCA(vnf,ncp=4)  
res.impute$tab.disj[1:5,1:5]

## Q7.1.1 Q7.1.2 Q7.1.3 Q7.2.1 Q7.2.2  
## 1 1 0 0 1 0  
## 2 1 0 0 1 0  
## 3 1 0 0 1 0  
## 4 1 0 0 1 0  
## 5 1 0 0 0 0

res.impute$comp[1:5,1:5]

## Q7.1 Q7.2 Q7.4 Q8.1 Q8.2  
## 1 1 1 3 1 2  
## 2 1 1 1 2 1  
## 3 1 1 1 2 2  
## 4 1 1 1 1 1  
## 5 1 3 1 1 1

#2.2) Single imputation for mixed data with FAMD and with Forest  
res.ncp <- estim\_ncpFAMD(ozo)

##   
 |   
 | | 0%  
 |   
 |= | 1%  
 |   
 |= | 2%  
 |   
 |== | 3%  
 |   
 |=== | 4%  
 |   
 |=== | 5%  
 |   
 |==== | 6%  
 |   
 |===== | 7%  
 |   
 |===== | 8%  
 |   
 |====== | 9%  
 |   
 |======= | 10%  
 |   
 |======= | 11%  
 |   
 |======== | 12%  
 |   
 |========= | 13%  
 |   
 |========= | 14%  
 |   
 |========== | 15%  
 |   
 |=========== | 16%  
 |   
 |=========== | 17%  
 |   
 |============ | 18%  
 |   
 |============ | 19%  
 |   
 |============= | 20%  
 |   
 |============== | 21%  
 |   
 |============== | 22%  
 |   
 |=============== | 23%  
 |   
 |================ | 24%  
 |   
 |================ | 25%  
 |   
 |================= | 26%  
 |   
 |================== | 27%  
 |   
 |================== | 28%  
 |   
 |=================== | 29%  
 |   
 |==================== | 30%  
 |   
 |==================== | 31%  
 |   
 |===================== | 32%  
 |   
 |====================== | 33%  
 |   
 |====================== | 34%  
 |   
 |======================= | 35%  
 |   
 |======================== | 36%  
 |   
 |======================== | 37%  
 |   
 |========================= | 38%  
 |   
 |========================== | 39%  
 |   
 |========================== | 40%  
 |   
 |=========================== | 41%  
 |   
 |============================ | 42%  
 |   
 |============================ | 43%  
 |   
 |============================= | 44%  
 |   
 |============================== | 45%  
 |   
 |============================== | 46%  
 |   
 |=============================== | 47%  
 |   
 |================================ | 48%  
 |   
 |================================ | 49%  
 |   
 |================================= | 51%  
 |   
 |================================= | 52%  
 |   
 |================================== | 53%  
 |   
 |=================================== | 54%  
 |   
 |=================================== | 55%  
 |   
 |==================================== | 56%  
 |   
 |===================================== | 57%  
 |   
 |===================================== | 58%  
 |   
 |====================================== | 59%  
 |   
 |======================================= | 60%  
 |   
 |======================================= | 61%  
 |   
 |======================================== | 62%  
 |   
 |========================================= | 63%  
 |   
 |========================================= | 64%  
 |   
 |========================================== | 65%  
 |   
 |=========================================== | 66%  
 |   
 |=========================================== | 67%  
 |   
 |============================================ | 68%  
 |   
 |============================================= | 69%  
 |   
 |============================================= | 70%  
 |   
 |============================================== | 71%  
 |   
 |=============================================== | 72%  
 |   
 |=============================================== | 73%  
 |   
 |================================================ | 74%  
 |   
 |================================================= | 75%  
 |   
 |================================================= | 76%  
 |   
 |================================================== | 77%  
 |   
 |=================================================== | 78%  
 |   
 |=================================================== | 79%  
 |   
 |==================================================== | 80%  
 |   
 |===================================================== | 81%  
 |   
 |===================================================== | 82%  
 |   
 |====================================================== | 83%  
 |   
 |====================================================== | 84%  
 |   
 |======================================================= | 85%  
 |   
 |======================================================== | 86%  
 |   
 |======================================================== | 87%  
 |   
 |========================================================= | 88%  
 |   
 |========================================================== | 89%  
 |   
 |========================================================== | 90%  
 |   
 |=========================================================== | 91%  
 |   
 |============================================================ | 92%  
 |   
 |============================================================ | 93%  
 |   
 |============================================================= | 94%  
 |   
 |============================================================== | 95%  
 |   
 |============================================================== | 96%  
 |   
 |=============================================================== | 97%  
 |   
 |================================================================ | 98%  
 |   
 |================================================================ | 99%  
 |   
 |=================================================================| 100%

res.famd <- imputeFAMD(ozo,ncp=2)  
res.famd$completeObs[1:5,1:5]

## maxO3 T9 T12 T15 Ne9  
## 20010601 87 15.60000 18.500 20.17593 4.000000  
## 20010602 82 17.20488 19.579 20.42794 5.000000  
## 20010603 92 15.30000 17.600 19.50000 2.000000  
## 20010604 114 16.20000 19.700 24.34365 1.000000  
## 20010605 94 18.89175 20.500 20.40000 5.395526

# # install.packages("missForecast")  
# library(missForecast)  
# res.rf <- missForest(ozo)  
# res.rf$ximp[1:5,1:5]

## 2.3) Multiple imputation for categorical data: Mu;tinomial regression with missing values

To perform a mutinomial with missing values, we can use multiple imputation.

# with mice  
library(mice)  
x.impmi <- mice(healthNA[,c(1:6,12:20)],m=5,printFlag=FALSE)  
  
# with MCA  
x.impmimca <- MIMCA(healthNA[,c(1:6,12:20)],ncp=5)

# Performing a model on each imputed data table  
lm.mice.out <- with(x.impmi,nnet::multinom(Alcohol~Pbsleep+Fatigue+Nightmare,trace=F))  
pool.mice <- pool(lm.mice.out) # combining the results  
summary(pool.mice)

## estimate std.error statistic  
## Never:(Intercept) 0.66040811 0.6343901 1.04101272  
## Never:PbsleepOften 0.31270969 0.7850324 0.39833985  
## Never:PbsleepRare 0.09442752 0.6195498 0.15241313  
## Never:FatigueQuiteOften -0.46690388 0.7456402 -0.62617855  
## Never:FatigueRare -0.33668204 0.7794162 -0.43196695  
## Never:FatigueVeryoften -0.63895599 1.0506438 -0.60815665  
## Never:NightmareQuiteOften 0.62959531 1.2559926 0.50127312  
## Never:NightmareRare 0.44392191 0.6332707 0.70099863  
## Never:NightmareVeryoften -14.38475668 448.7285831 -0.03205670  
## Ocasional:(Intercept) 1.48667479 0.5658228 2.62745670  
## Ocasional:PbsleepOften -0.04224835 0.6664860 -0.06338970  
## Ocasional:PbsleepRare 0.38274137 0.5072923 0.75447895  
## Ocasional:FatigueQuiteOften 0.39413180 0.6506987 0.60570553  
## Ocasional:FatigueRare 0.47813125 0.6774479 0.70578306  
## Ocasional:FatigueVeryoften 0.06464259 0.8804771 0.07341768  
## Ocasional:NightmareQuiteOften 1.13489106 1.0986137 1.03302107  
## Ocasional:NightmareRare 1.03550972 0.5356759 1.93309016  
## Ocasional:NightmareVeryoften -0.56506716 1.0942414 -0.51640085  
## df p.value  
## Never:(Intercept) 306.3932 0.29868880  
## Never:PbsleepOften 262.2995 0.69065659  
## Never:PbsleepRare 267.2444 0.87896123  
## Never:FatigueQuiteOften 306.7125 0.53166314  
## Never:FatigueRare 306.8724 0.66606868  
## Never:FatigueVeryoften 294.5741 0.54353325  
## Never:NightmareQuiteOften 218.2427 0.61653818  
## Never:NightmareRare 280.2627 0.48383487  
## Never:NightmareVeryoften 306.9883 0.97444767  
## Ocasional:(Intercept) 306.7563 0.00903467  
## Ocasional:PbsleepOften 253.2288 0.94949745  
## Ocasional:PbsleepRare 295.4251 0.45114005  
## Ocasional:FatigueQuiteOften 304.7111 0.54515786  
## Ocasional:FatigueRare 306.9564 0.48085829  
## Ocasional:FatigueVeryoften 290.9630 0.94152156  
## Ocasional:NightmareQuiteOften 211.4734 0.30240715  
## Ocasional:NightmareRare 291.7436 0.05414450  
## Ocasional:NightmareVeryoften 226.3909 0.60594625

imp<-prelim(x.impmimca,healthNA[,c(1:6,12:20)])  
fit <- with(data=imp,exp=nnet::multinom(Alcohol ~ Pbsleep + Fatigue +Nightmare, trace = FALSE))  
res.pool<-pool(fit)  
summary(res.pool)

## estimate std.error statistic  
## Never:(Intercept) 0.657815673 0.6345329 1.036692836  
## Never:PbsleepOften 0.290812508 0.7816127 0.372067297  
## Never:PbsleepRare 0.090807841 0.6130303 0.148129455  
## Never:FatigueQuiteOften -0.468252164 0.7485530 -0.625543132  
## Never:FatigueRare -0.334332876 0.7796307 -0.428834916  
## Never:FatigueVeryoften -0.637864152 1.0458072 -0.609925163  
## Never:NightmareQuiteOften 0.721530980 1.2545515 0.575130633  
## Never:NightmareRare 0.451843223 0.6331119 0.713686242  
## Never:NightmareVeryoften -12.177456142 382.3438827 -0.031849486  
## Ocasional:(Intercept) 1.486863215 0.5660798 2.626596610  
## Ocasional:PbsleepOften -0.003677097 0.6630918 -0.005545382  
## Ocasional:PbsleepRare 0.396410882 0.5064687 0.782695729  
## Ocasional:FatigueQuiteOften 0.395495631 0.6510444 0.607478782  
## Ocasional:FatigueRare 0.477812957 0.6777316 0.705018010  
## Ocasional:FatigueVeryoften 0.061352634 0.8749591 0.070120571  
## Ocasional:NightmareQuiteOften 1.165796007 1.0963912 1.063302998  
## Ocasional:NightmareRare 1.020307433 0.5371672 1.899422601  
## Ocasional:NightmareVeryoften -0.807663330 1.0998350 -0.734349555  
## df p.value  
## Never:(Intercept) 306.3998 0.300694987  
## Never:PbsleepOften 302.7091 0.710099466  
## Never:PbsleepRare 302.9538 0.882337860  
## Never:FatigueQuiteOften 304.6435 0.532079434  
## Never:FatigueRare 306.9199 0.668344152  
## Never:FatigueVeryoften 303.4326 0.542362615  
## Never:NightmareQuiteOften 290.2092 0.565624260  
## Never:NightmareRare 299.1618 0.475963481  
## Never:NightmareVeryoften 306.9648 0.974612781  
## Ocasional:(Intercept) 306.4082 0.009057055  
## Ocasional:PbsleepOften 301.0541 0.995579050  
## Ocasional:PbsleepRare 304.2085 0.434409196  
## Ocasional:FatigueQuiteOften 305.3989 0.543982335  
## Ocasional:FatigueRare 306.9354 0.481333623  
## Ocasional:FatigueVeryoften 304.5549 0.944143352  
## Ocasional:NightmareQuiteOften 293.5783 0.288480316  
## Ocasional:NightmareRare 299.6417 0.058446159  
## Ocasional:NightmareVeryoften 288.8656 0.463296293

## 2.3) Imputation with groups of variables/multiple factor analysis with missing values.

Let us consider the journal impact factors data from journalmetrics.com. We use a subset of 443 journals of the same sections than Journal of Statistical Software (Computer Science :: Software“, Decision Sciences :: Statistics, Probabilityand Uncertainty” and Mathematics :: Statistics and Probability“).

This data has 45 columns which correspond to three metrics recorded each year from 1999 to 2013:

1. IPP - impact per publication (it is closed to the ISI impact factor but for three rather than two years),
2. SNIP - source normalized impact per paper (tries to weight by the number of citationsper subject field to adjust for different citation cultures) and
3. the SJR - SCImago journal rank (tries to capture average prestige per publication). This data contains 31% of missing values.

We impute it with single imputation by Multiple Factor Analysis.

# install.packages("denoiseR")  
library(denoiseR)  
  
summary(impactfactor)  
year=NULL; for (i in 1: 15) year= c(year, seq(i,45,15))   
res.imp <- imputeMFA(impactfactor, group = rep(3, 15), type = rep("s", 15))  
  
## MFA on the imputed data set  
res.mfa <-MFA(res.imp$completeObs, group=rep(3,15), type=rep("s",15),   
name.group=paste("year", 1999:2013,sep="\_"),graph=F)  
  
plot(res.mfa, choix = "ind", select = "contrib 15", habillage = "group", cex = 0.7)  
points(res.mfa$ind$coord[c("Journal of Statistical Software", "Journal of the American Statistical Association", "Annals of Statistics"), 1:2], col=2, cex=0.6)  
text(res.mfa$ind$coord[c("Journal of Statistical Software"), 1],   
res.mfa$ind$coord[c("Journal of Statistical Software"), 2],cex=1,  
labels=c("Journal of Statistical Software"),pos=3, col=2)  
plot.MFA(res.mfa,choix="var", cex=0.5,shadow=TRUE, autoLab = "yes")  
plot(res.mfa, select="IEEE/ACM Transactions on Networking", partial="all", habillage="group",unselect=0.9,chrono=TRUE)

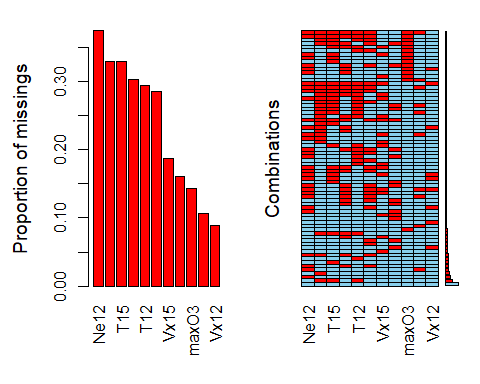
# 3) Contingency tables with count data and missing values

# 4) Multilevel (mixed) data with missing values

# Alternative approach for visualizing missingness

The function VIM **aggr** calculates and represents the number of missing entries in each variable and for certain combinations of variables (which tend to be missing simultaneously).

res <- summary(aggr(don,sortVar=TRUE))$combinations



##   
## Variables sorted by number of missings:   
## Variable Count  
## Ne12 0.37500000  
## T9 0.33035714  
## T15 0.33035714  
## Ne9 0.30357143  
## T12 0.29464286  
## Ne15 0.28571429  
## Vx15 0.18750000  
## Vx9 0.16071429  
## maxO3 0.14285714  
## maxO3v 0.10714286  
## Vx12 0.08928571

head(res[rev(order(res[,2])),])

## Combinations Count Percent  
## 1 0:0:0:0:0:0:0:0:0:0:0 13 11.607143  
## 45 0:1:1:1:0:0:0:0:0:0:0 7 6.250000  
## 10 0:0:0:0:0:1:0:0:0:0:0 5 4.464286  
## 35 0:1:0:0:0:0:0:0:0:0:0 4 3.571429  
## 41 0:1:0:0:1:1:1:0:0:0:0 3 2.678571  
## 28 0:0:1:0:0:0:0:0:0:0:0 3 2.678571

The VIM function **matrixplot** creates a matrix plot in which all cells of a data matrix are visualized by rectangles. Available data is coded according to a continuous color scheme (gray scale), while missing/imputed data is visualized by a clearly distinguishable color (red). If you use Rstudio the plot is not interactive (there are the warnings), but if you use R directly, you can click on a column of your choice: the rows are sorted (decreasing order) of the values of this column. This is useful to check if there is an association between the value of a variable and the missingness of another one.

matrixplot(don,sorttby=2)

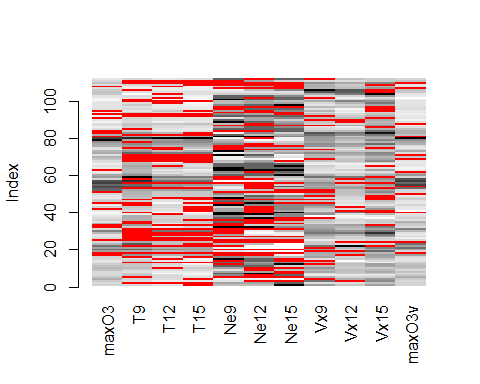
## Warning in plot.window(..., yaxs = "r"): "sorttby" はグラフィックスパラメー  
## タではありません

## Warning in plot.window(..., yaxs = if (is.null(dots$yaxs)) "i" else dots  
## $yaxs): "sorttby" はグラフィックスパラメータではありません

## Warning in (function (side, at = NULL, labels = TRUE, tick = TRUE, line =  
## NA, : "sorttby" はグラフィックスパラメータではありません

## Warning in axis(2, xpd = NA, ...): "sorttby" はグラフィックスパラメータでは  
## ありません

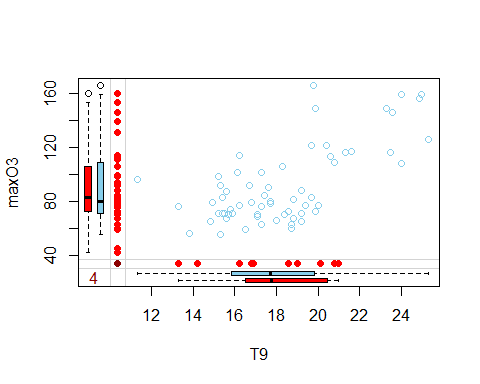
## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):  
## "sorttby" はグラフィックスパラメータではありません



# Here the variable selected is variable 2

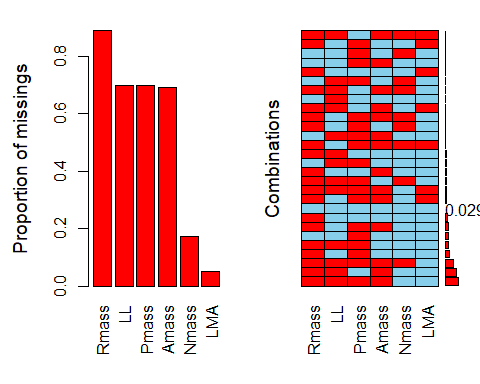
The VIM function **marginplot** creates a scatterplot with additional information on the missing values. If you plot the variables (x,y), the points with no missing values are represented as in a standard scatterplot. The points for which x (resp. y) is missing are represented in red along the y (resp. x) axis. In addition, boxplots of the x and y variables are represented along the axes with and without missing values (in red all variables x where y is missing, in blue all variables x where y is observed).

marginplot(don[,c("T9","maxO3")])



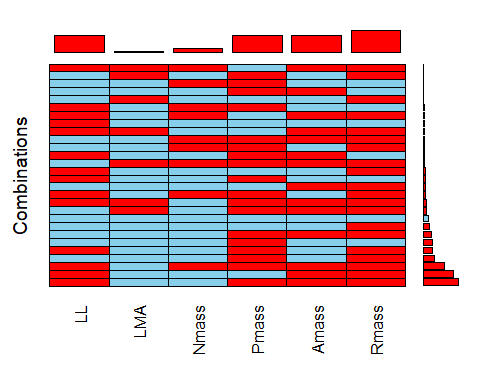
# visualize the pattern  
library(VIM)  
#aggr(Ecolo)  
aggr(Ecolo,  
 only.miss = TRUE,  
 numbers = TRUE,  
 sortVar = TRUE)

## Warning in plot.aggr(res, ...): not enough vertical space to display  
## frequencies (too many combinations)



##   
## Variables sorted by number of missings:   
## Variable Count  
## Rmass 0.89013633  
## LL 0.69967923  
## Pmass 0.69847634  
## Amass 0.69125902  
## Nmass 0.17361668  
## LMA 0.04971933

res <- summary(aggr(Ecolo, prop = TRUE, combined = TRUE))$combinations



#res[rev(order(res[,2])),]