MULTIPLE IMPUTATION WITH PRINCIPAL COMPONENT METHODS: A USER GUIDE

In this chapter, we present how to use the multiple imputation methods described previously: the BayesMIPCA method, allowing multiple imputation of continuous data using PCA and MIMCA, allowing multiple imputation for categorical data using MCA. Both are available in the R package missMDA, which contains also functions to tune the parameters of the MI methods and functions to make diagnostics. These functions are presented through real incomplete data sets. First, we present how to investigate an incomplete data set and point out the utility of principal component methods to achieve this goal. Then, we explain how to perform MI using PCA if the data set is continuous and MI using MCA when data are categorical. Finally, we show how to perform analysis and how to pool the analysis results.

CONTENTS

1 Exploratory analysis of incomplete data

Like in the standard statistical framework without missing values, the exploration of data before performing a statistical method is required. However, exploring incomplete data does not only relate to the observed values, but also to the missing data pattern, and to the mechanism that connects values and pattern as well. Principal component methods offer interesting ways to perform it.

In this part, we will principally use the data set *sleep* from the R package VIM (Templ et al., 2015) as an illustrative example where the missing data mechanism is unknown. From Allison and Chichetti (1976), this data set deals with 62 mammal species on the interrelationship between sleep, ecological, and constitutional variables. The data set contains missing values on five variables. The three last variables are five-point scales.

```
> library(VIM)
> data(sleep, package = "VIM")
> don<-sleep
> summary(don,digits=5)
```

```
    BodyWgt            BrainWgt            NonD             Dream
 Min.   :   0.005   Min.   :   0.14   Min.   : 2.1000   Min.   :0.000
 1st Qu.:   0.600   1st Qu.:   4.25   1st Qu.: 6.2500   1st Qu.:0.900
 Median :   3.342   Median :  17.25   Median : 8.3500   Median :1.800
 Mean   : 198.790   Mean   : 283.13   Mean   : 8.6729   Mean   :1.972
 3rd Qu.:  48.203   3rd Qu.: 166.00   3rd Qu.:11.0000   3rd Qu.:2.550
 Max.   :6654.000   Max.   :5712.00   Max.   :17.9000   Max.   :6.600
                                      NA's   :14        NA's   :12
     Sleep             Span              Gest             Pred
 Min.   : 2.600   Min.   :  2.000   Min.   : 12.00   Min.   :1.000
 1st Qu.: 8.050   1st Qu.:  6.625   1st Qu.: 35.75   1st Qu.:2.000
 Median :10.450   Median : 15.100   Median : 79.00   Median :3.000
 Mean   :10.533   Mean   : 19.878   Mean   :142.35   Mean   :2.871
 3rd Qu.:13.200   3rd Qu.: 27.750   3rd Qu.:207.50   3rd Qu.:4.000
 Max.   :19.900   Max.   :100.000   Max.   :645.00   Max.   :5.000
 NA's   :4        NA's   :4         NA's   :4
      Exp              Danger
 Min.   :1.0000   Min.   :1.0000
 1st Qu.:1.0000   1st Qu.:1.0000
 Median :2.0000   Median :2.0000
 Mean   :2.4194   Mean   :2.6129
 3rd Qu.:4.0000   3rd Qu.:4.0000
 Max.   :5.0000   Max.   :5.0000
```

1.1 Missing data pattern

The exploration of incomplete data requires the investigation of the missing data pattern. First, this investigation is important because it can define the method to use to deal with

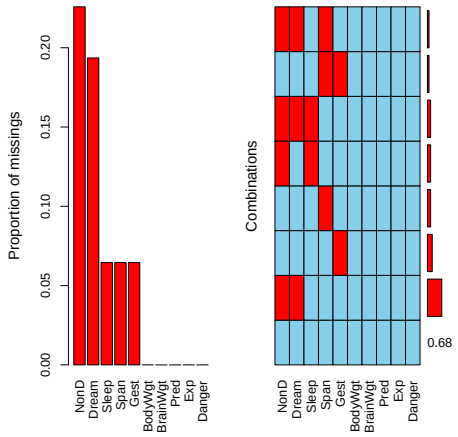

Figure 1: Visualisation of the missing data pattern: Aggregation graphic of the incomplete data set *sleep*

the missing values. Indeed, if the number of missing values is very small compared to the number of individuals, MI is not necessarily required and listwise deletion, which is a simpler method, could be preferred. In addition, although the BayesMIPCA and MIMCA methods are suitable for monotone or non-monotone pattern, it could be interesting for other MI method to identify such a pattern. For instance, chained equations can be tuned to deal with a monotone one (Rubin, 1987). Secondly, the proportion of missing values generally affects the convergence of iterative algorithms used in MI, such as Expectation-Maximisation (EM) algorithm or Data-Augmentation (DA) algorithm. If the proportion of missing values is large, these algorithms can require a larger number of iterations than usual. In addition, more imputed data sets can be required if a large part of the values of the analysis model are missing. Thirdly, the study of the frequencies of combinations of missing values on several variables can highlight a MCAR mechanism. Indeed, if the mechanism is MCAR, missing values occur independently on each incomplete variable, which implies that each combination of missing values is equally probable.

The R package VIM provides many tools to visualise a missing data pattern. The function `aggr` plots the amount of missing values in each variable and the amount of missing values in certain combinations of variables (Figure 1). Missing values occurs on 5 variables through a non-monotone pattern (right-hand side of Figure 1). The percentage of missing values is moderate, but only 68% of the individuals are complete, which justify the use of multiple imputation. The plot of the combinations (Figure 1) indicates that missing values often occur simultaneously on the variables Dream and NonD for instance. Nevertheless, this plot quickly becomes unreadable when the number of incomplete variables is large.

MCA can be straightforwardly used to visualise the missing data pattern even if the number of variable is large. Indeed, the missing data pattern can be viewed as a data set

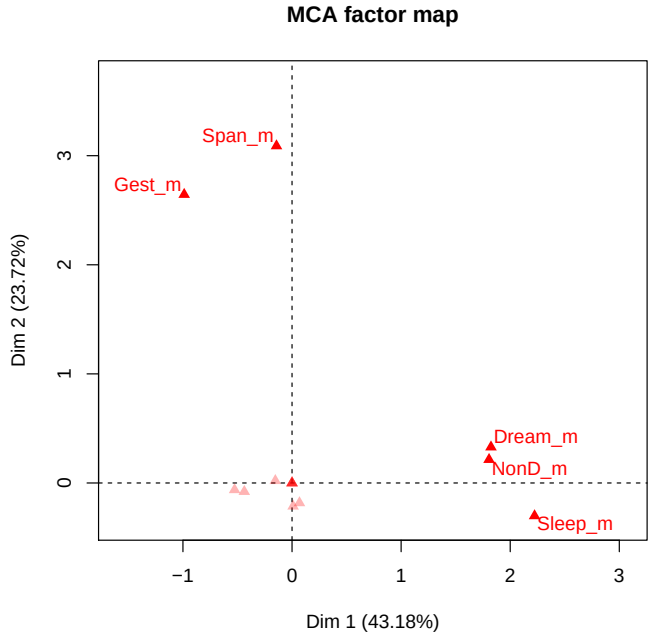

Figure 2: Visualisation of the missing data pattern: Graph of the categories from the MCA on the missing data pattern for the data set *sleep*

where all variables have two categories: *mis*, indicating a missing value and *obs* for an observed one. MCA will highlight associations between pairs of categories by searching the common dimension of variability between the corresponding variables. Thus, it will show if missing values simultaneously occur in several variables or if missing values occur when some other variables are observed. To perform MCA on the missing data pattern and visualise the associations between categories, the following lines can be run:

```
> library(FactoMineR)
> # Creation of a categorical data set with "o" when observed and "m" when missing
> pattern <- matrix("o",nrow=nrow(don),ncol=ncol(don))
> pattern[is.na(don)] <- "m"
> pattern<-as.data.frame(pattern)
> dimnames(pattern) <- dimnames(don)
> # MCA
> res.mca<-MCA(pattern,graph=F)
> plot(res.mca,selectMod=grep("_m",rownames(res.mca$var$coord)),invisible="ind")
```

The graph of the categories is represented in Figure 2. It highlights two groups of categories:

- group 1: Sleep_m, NonD_m, Dream_m
- group 2: Gest_m, Span_m

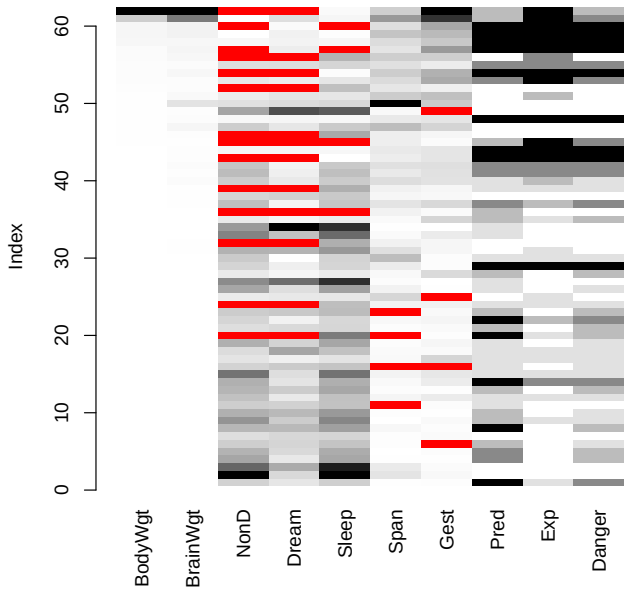

Figure 3: Visualisation of the missing data mechanism: Matrix plot of the incomplete *sleep* data set

The categories of the first group have large coordinates (positive) on the first axis (horizontal) whereas the categories of the second group have large coordinates on the second axis (vertical) only. It means that missing values tend to occur simultaneously on the variables Span and Gest on the one hand, and on the variables Dream, NonD and Sleep on the other hand. Consequently, MCA suggests that the missing data are not missing completely at random. To go further in the understanding of the mechanism, it can be useful to make the link between missing values and observed values.

1.2 Missing data mechanism

Even if the MAR assumption cannot be checked, the simultaneous investigation of the missing data pattern and the observed values can allow a better understanding of the missing data mechanism. To obtain some clues about its nature, first, it can be useful to visualise the data matrix (see Figure 3). This representation can be obtained by using the function `matrixplot` from the package VIM: all cells of a data matrix are visualised by rectangles. Available data are coded according to a grey scale scheme, while missing data are in red. By ordering the lines according to one variable, the link between missing values and this variable can be highlighted.

However, as previously, it can be difficult to highlight in this way relationships between missing data pattern and observed values when the number of variables is large. MCA can also be used to point out these relationships, but contrary to the previous case, the

principal component method need to be performed on a data set containing information on observed data and the missing data pattern as well. To take both into account, we consider observed data and include missing values as a category of the variables. Thus, if there is a link between the missing values and the observed values, then MCA can highlight it. Note that MCA is not dedicated to continuous variables. If the data set contains such variables, they need to be split into categories. Continuous variables of the *sleep* data set can be recoded as follows:

```
> don.cat<-as.data.frame(don)
> for(i in 1:7){
+   breaks<-c(-Inf,quantile(don.cat[[i]],na.rm=T)[-1])
+   don.cat[[i]]<-cut(don.cat[[i]],breaks=breaks,labels=F)
+   don.cat[[i]]<-addNA(don.cat[[i]],ifany=T)
+ }
> for(i in 8:10){
+   don.cat[[i]]<-as.factor(don.cat[[i]])
+ }
> summary(don.cat)
```

BodyWgt	BrainWgt	NonD	Dream	Sleep	Span	Gest	Pred	Exp	Danger
1:16	1:16	1 :12	1 :14	1 :15	1 :15	1 :15	1:14	1:27	1:19
2:15	2:15	2 :12	2 :13	2 :14	2 :14	2 :14	2:15	2:13	2:14
3:15	3:15	3 :14	3 :10	3 :15	3 :14	3 :14	3:12	3: 4	3:10
4:16	4:16	4 :10	4 :13	4 :14	4 :15	4 :15	4: 7	4: 5	4:10
		NA:14	NA:12	NA: 4	NA: 4	NA: 4	5:14	5:13	5: 9

Note that the first variables are split into 4 categories with the same number of individuals per category, whereas the last ones are only recoded since they are five-point scales.

Then, MCA can be applied on the recoded data set:

```
> res.MCA<-MCA(don.cat,graph=F)
> plot(res.MCA,choix="ind",invisible="ind")
```

The Figure 4 sums up the relationships between observed categories and missing categories. For instance, the category SleepNA has high coordinates on the first axis. Missing values on the variables Sleep can be associated with the observed categories having the largest coordinates on the first axis, such as Exp_5, Danger_5 BodyWgt_4 and BrainWgt_4. Thus, MCA shows a link between missing values on the variables Sleep and the values of the variables Exp, Danger, BodyWgt and BrainWgt. Many other comments could be done from this graph. It would be also interesting to analyse the following dimensions by specifying the argument `axes=c(3,4)`:

```
> plot(res.MCA,choix="ind",invisible="ind",axes=c(3,4))
```

Note that the graph can be sensitive to the splitting. If the number of individuals is large, it can be interesting to increase the number of categories to obtain a finer understanding

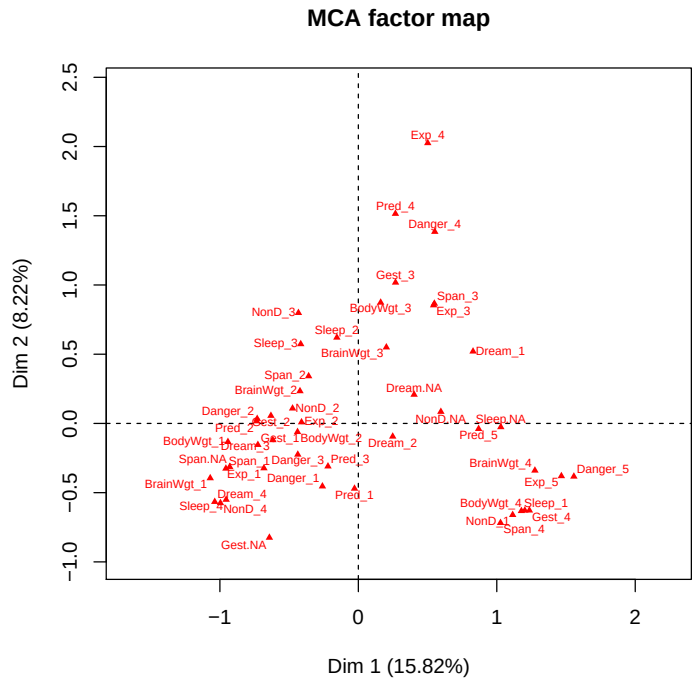

Figure 4: Visualisation of the missing data mechanism: Graph of the categories from MCA performed on the data set *sleep* where continuous data are split into categories and missing values are recoded as a category

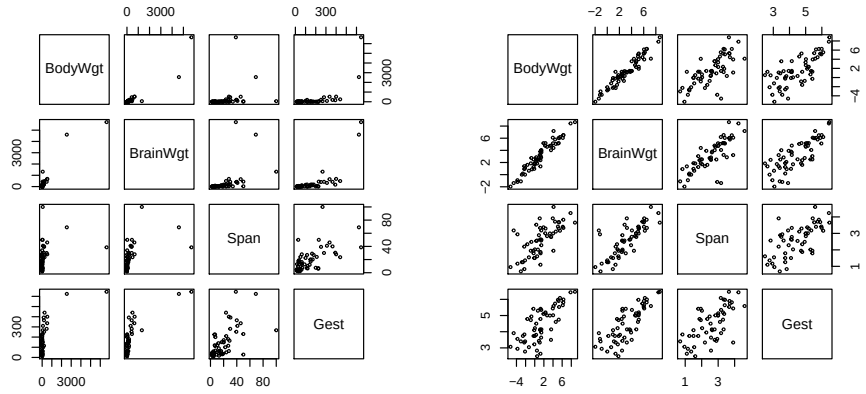

Figure 5: Bivariate plots for four variables of the *sleep* data set. The graph on the left corresponds to the original variables, and the graph on the right to the variables after logarithm transformations.

of the relationships between categories, and therefore of the mechanism. In addition, the splitting is performed according to the quantiles, but other ways could be used. The advantage of quantiles is to avoid rare categories that can have a great influence on the representation.

1.3 Observed data

The analysis of the observed values is also important for understanding the data. Principal component methods are powerful tool for this purpose. However, applying principal component methods on an incomplete data set is not straightforward since the most of statistical methods, they cannot be used directly on incomplete data. The package missMDA (Husson and Josse, 2015; Josse and Husson, 2015) enables the use of principal components on an incomplete data set.

1.3.1 Preliminary transformations

Principal component methods provide better representation when relationships between continuous variables are linear. To apply PCA or FAMD to incomplete data (continuous or mixed respectively), it can be first useful to check that a linear trend occurs between variables. If linearity between them is not verified, then transformations of the variables (such as logarithm, square root, logistic) can be performed. For instance, to apply PCA on the incomplete data set *sleep*, we begin by generating the bivariate plots. The Figure 5 illustrates that a log transformation of some variables of the data set *sleep* can improve linearity.

```
> don.log<-sleep
> don.log[,c(1:2,6,7)]<-log(don.log[,c(1:2,6,7)])
```

Logarithm transformation can be useful for skew variables, while square root transformation will be generally suitable for count data, and logistic transformation for proportions. Note that these transformations will be also useful to perform MI. To apply BayesMIPCA on an incomplete data set, it will be more suitable to work with the transformed data set, which could be eventually back-transformed.

1.3.2 Principal component methods with missing values

Principal component methods are particularly useful to better understand the structure of the data. This allows the understanding of the relationships between variables as well as the similarities between individuals. In particular, This can be useful to detect outliers. For a continuous data set, PCA is the suitable method to use, whereas MCA will be appropriate for a categorical data set and FAMD for a mixed one. However, missing values make it difficult to be applied. 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 (Josse and Husson, 2012) for a predefined number of dimensions. Then, PCA is performed on the imputed data set. The rationale is the same for MCA and FAMD.

The single imputation step requires tuning the number of dimensions used to impute the data. Through the argument `method.cv`, the function `estim_ncpPCA` proposes several cross-validation procedures to choose this number. The rationale of these methods is to search the number of components minimising the prediction error for observed values. The default method is the generalised cross-validation method (`method.cv="gcv"`). It consists in searching the number of dimensions which minimises the generalised cross-validation criterion, which can be seen as an approximation of the leave-one-out cross-validation criterion (Josse and Husson, 2011). 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 minimising the criterion can sometimes be unobvious when several local minimum occur. In such a case, more computationally intensive methods, those performing explicit cross-validation, can be used, such as Kfold (`method.cv="Kfold"`) or leave-one-out (`method.cv="loo"`). More precisely, the number of dimensions can be estimated as follows on the *sleep* data set:

```
> library(missMDA)
> res.ncp<-estim_ncpPCA(don.log,method.cv="Kfold")
> plot(res.ncp$criterion~names(res.ncp$criterion),xlab="number of dimensions")
> ncp<-4
```

The Kfold cross-validation suggests to retain 4 dimensions for the imputation of the data set *sleep* (Figure 7). Thus, the incomplete data set can be imputed using the function `imputePCA`, specifying the number of dimensions through the argument `ncp=4`. The function returns the imputed data set through the output `completeObs` on which PCA can be performed to summarise the relationships between variables and similarities between individuals:

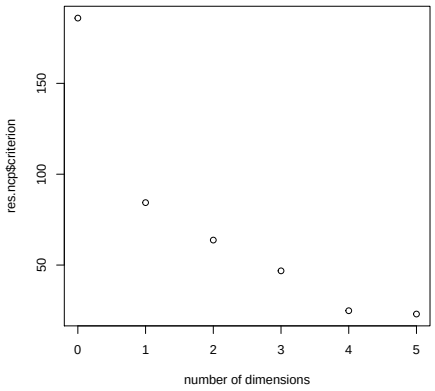

Figure 6: Cross-validation error according to the number of dimensions used for the data set *sleep*

```
> #single imputation
> res.imp<-imputePCA(don.log,ncp = ncp)
> #PCA on the imputed data set
> res.PCA<-PCA(res.imp$completeObs,graph=F)
> #Graph of the variables and graph of the individuals
> plot(res.PCA,choix="var")
> plot(res.PCA,choix="ind")
```

The correlation circle summarises the relationships between variables (left-hand side of Figure 7). The coordinates of the variable on one axis is given by its correlation with the corresponding component. This graph shows several groups of linked variables. For instance, the variables Dream, Sleep and NonD are linked and quite negatively correlated with the others. The graph of the individuals summarises similarities between individuals. The individuals are mapped so that individuals having similar values within the data set of all variables are also close on the map. For instance, Figure 7 shows similarities between some individuals, such as the individuals 37 and 38, and shows oppositions between others, such as between the individuals 5 and 61. It does not highlight outliers.

Note that for a categorical data set, MCA can be performed in the same way: first, the number of components is estimated, next single imputation with iterative MCA algorithm is used (Josse et al., 2012), and then MCA is performed on the imputed disjunctive table. It will be also the same for FAMD on mixed data. For instance, MCA can be applied on the incomplete categorical data set *TitanicNA* from the package missMDA.

```
> data(TitanicNA)
> summary(TitanicNA)

  CLASS        AGE          SEX          SURV
 0   :733   0   :  85   0   : 388   0   :1183
```

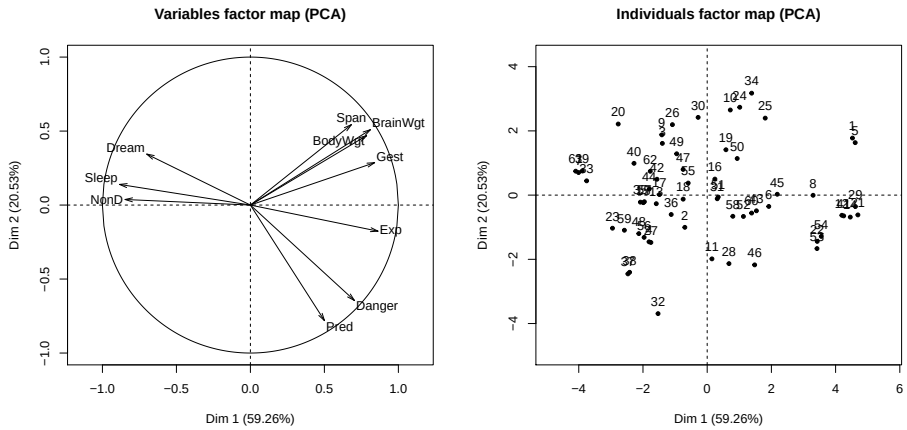

Figure 7: Correlation circle and graph of individuals from the principal component analysis on the data set *sleep*

```
 1   :256   1   :1662   1   :1393   1   : 566
 2   :225   NA's: 454   NA's: 420   NA's: 452
 3   :553
NA's:434
```

This data set provides information on the fate of passengers on the fatal maiden voyage of the ocean liner Titanic, summarised according to economic status (class), sex, age and survival. Twenty percent of values are missing completely at random on each variable. To perform MCA on such an incomplete categorical data set, we can run:

```
> #number of components
> estim_ncpMCA(TitanicNA)
> #single imputation
> res.imp<-imputeMCA(TitanicNA,ncp = 5)
> #MCA on the imputed data set
> res.MCA<-MCA(TitanicNA,res.imp$tab.disj,graph=F)
> #Graph of the categories and individuals
> plot(res.MCA,choix="ind",invisible="ind")
> plot(res.MCA,choix="ind")
```

To sum up, the exploration of an incomplete data set allows the user to understand the missing data pattern, the mechanism, as well as the observed values. The principal component methods are very suitable tool in this goal. On one hand, MCA can be used to analyse the missing data pattern. On the other hand, the mechanism can be studied by analysing simultaneously data and missing data pattern with MCA: if data are continuous, then variables must be split and a new category for the missing values added, otherwise, if data are categorical, MCA can be used adding a category for missing values. Lastly, PCA, MCA, or even FAMD, can be used to understand the structure of the observed values, continuous, categorical, or mixed, respectively.

2 Multiple imputation for continuous data

2.1 Multiple Imputation

The BayesMIPCA method can be used to perform multiple imputation of continuous variables through PCA (Audigier et al., 2015a). The algorithm is based on the Bayesian treatment of the PCA model proposed by Verbanck et al. (2013). When missing values occur in the data set, a classical way to draw parameters from a posterior distribution consists in using a DA algorithm. Then, the imputation of the data set is performed from each parameter obtained by the DA algorithm. This requires alternating `Lstart` times imputation of the missing values and draw of the parameters in the posterior distribution (burn-in period). Note that the imputation step is called *step I* and the step of draw in the posterior is called *step P*. After convergence (to the posterior distribution), imputed data sets are kept each `L` iterations to ensure independence between successive imputations. The function `MIPCA` performs multiple imputation according to PCA from a DA algorithm. The BayesMIPCA method can be used by specifying the argument `method.mi="Bayes"`. Otherwise, a bootstrap version is performed (Josse and Husson, 2011). Note that this bootstrap method has been assessed to evaluate uncertainty in PCA, rather than to apply a statistical method on an incomplete data set. The function `MIPCA` requires as inputs the incomplete data set `X`, the number of imputed data sets `nboot` and the number of dimensions `ncp`. The number of dimensions can be chosen by cross-validation as described previously (Section 1.3). The values of `Lstart` and `L` are fixed to 1000 and 100, respectively. The function returns the multiply imputed data set through the value `res.MI`.

```
> #estimate the number of dimensions
> res.ncp<-estim_ncpPCA(don.log,method.cv="Kfold")
> plot(res.ncp$criterion~names(res.ncp$criterion),xlab="number of dimensions")
> ncp<-4

> #Multiple imputation
> res.BayesMIPCA<-MIPCA(X=don.log,nboot=100,ncp=ncp,method.mi="Bayes")

> #Extract the first data set
> imp1<-res.BayesMIPCA$res.MI[[1]]
> summary(imp1)
```

2.2 Diagnostics

2.2.1 BayesMIPCA algorithm

The number of iterations for the burn-in period (`Lstart`), and the number of iterations between each retained imputed data set (`L`) play a role in the BayesMIPCA algorithm. Typically, their values are empirically chosen through graphical investigations.

To check if the number of iterations for the burn-in period is sufficiently high, the successive values of some summaries (*e.g.* mean or correlation coefficients) are investigated

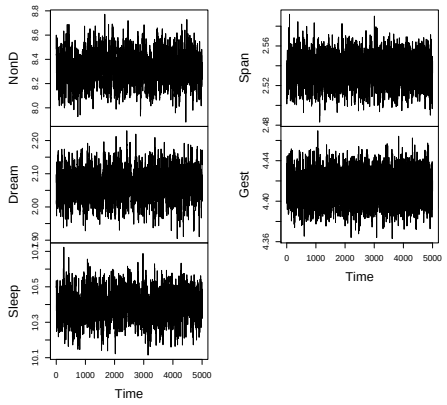

Figure 8: Successive values of the mean through the imputed data set for the imputed variables of the data set *sleep*

through the successive iterations of the algorithm. However, the imputed data sets given by the BayesMIPCA algorithm (with default parameters) are not the imputed data sets from each step (I), but only a sample of them because a data set is kept each `L` iterations from the $\texttt{Lstart}^{th}$. To obtain the successive imputed data sets, the algorithm need to be run by specifying the arguments `Lstart=1` and `L=1` for a large number of imputed data sets `nboot`. The successive values for the mean of each incomplete variables can be plotted with the time series function `ts`:

```
> res.conv<-MIPCA(X=don.log,nboot=5000,ncp=ncp, method.mi="Bayes",Lstart=1,L=1)
> res.mean<-lapply(res.conv$res.MI,colMeans)
> X<-ts(do.call(rbind,res.mean))
> plot(X[,3:7],main="")
```

If stationarity of the successive values seems to be verified after 1000 iterations, then `Lstart` can be preserved. Otherwise, it could be increased. Figure 8 shows the successive values for the means. The convergence seems to have been reached quickly, meaning that the default parameter `Lstart=1000` is suitable.

Concerning the number of iterations between each retained data set, it can be checked by visualising the autocorrelograms of the summaries (Figure 9). An autocorrelogram represents the correlation between the vector of the successive statistics and its shifted version for several lags. We aim finding a lag `L` sufficiently large to avoid correlation between the vectors of statistics. The autocorrelograms for the means can be obtained as follows:

```
> par(mfrow=c(3,2))
> Lstart<-1000
> X<-ts(X[Lstart:nrow(X),],start=Lstart)
> apply(X[,3:7],2,acf)
```

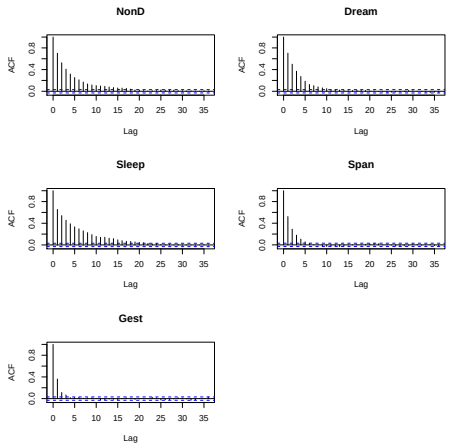

Figure 9: Autocorrelograms for the mean the imputed variables of the data set *sleep*

The Figure 9 shows the correlation between the mean of each incomplete variable, and its shift version for a lag smaller than 35. We see that a lag larger than 25 is sufficient to have a correlation close to zero. This indicates that `L` should be over 25 to expect independence between the imputed values from one data set to another. Therefore, the default parameter `L=100` is suitable.

These investigations are not foolproof and it can be useful to focus on others summaries, such as correlation coefficients that deal with the relationships between variables, whereas the means deal with marginal distribution only.

2.2.2 Fit of the model

The fit of the imputation model is interesting to evaluate the accuracy of the imputed values. A first way to assess accuracy consists in comparing the distribution of the imputed values and the distribution of the observed ones. The package VIM can be used for this purpose. 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 principal component methods can be very useful in this aim (Section 1).

Another way consists in comparing imputed values to their 'true' values. However, the values that are missing are not available, by definition. To assess the fit of the Bayesian PCA model, we propose to use *overimputation* (Blackwell et al., 2015). It consists in imputing each observed values from the parameters of the imputation model obtained from the MI procedure. The comparison between the "overimputed" values and the observed values is made by building a confidence interval for each observed value. If the model fits well the data, then the 90% confidence interval should contain the observed value

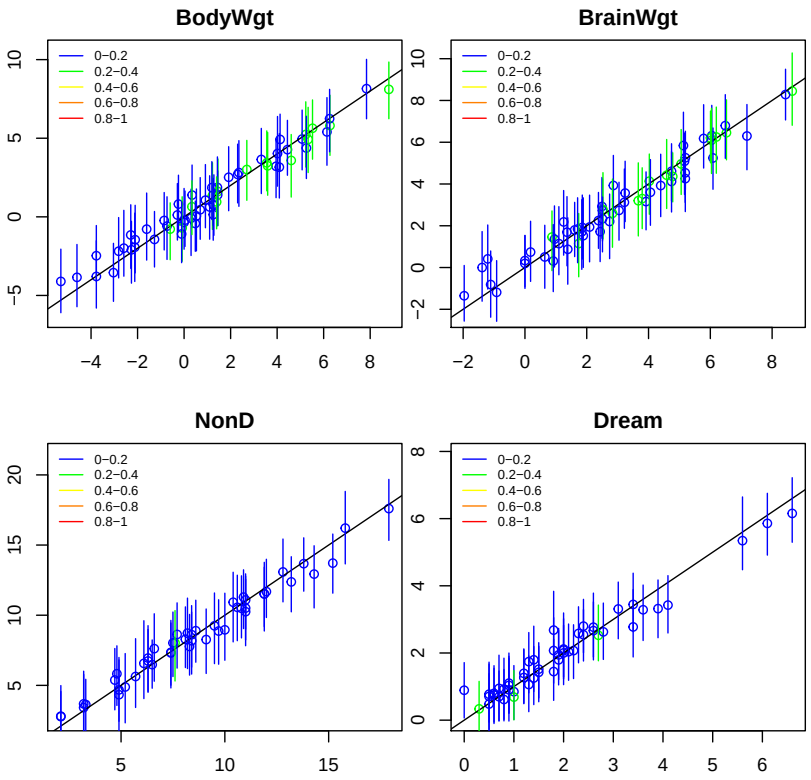

Figure 10: Assessment of the Bayesian PCA model. The dots represent the mean imputation and the vertical segments the confidence intervals for the missing value. Around 90 percent of these confidence intervals should contain the first bisector which corresponds to the true values of the missing values. The color of the line represents the proportion of missing observations in the missing data pattern for that observation.

in 90% of the cases. The fit can be assessed by the function `Overimpute` which takes as an input the output of the MIPCA function (`output`) and the variables that are plotted (`plotvars`). Note that this function is similar to the function `overimpute` from the Amelia package (Honaker et al., 2014).

For instance, to check the fit of the method on the first variables we can run

```
> Overimpute(res.BayesMIPCA,plotvars=1:4)
```

The function plots the predicted values and the confidence intervals as shown in Figure 10. The x-axis corresponds to the values that are suppressed, whereas the y-axis corresponds to the predicted values. Thus, the first bisector represents the line where the prediction is perfect. For each observation, the 90% confidence intervals is plotted with a color depending on the percentage of missing values on the covariates. The predicted values with a high number of missing covariates tend to have larger confidence intervals. If the multiple imputation performs well, then 90% of the confidence intervals should cross the first bisector. Note that the confidence intervals are construct according to the quantiles of

the overimputed values, therefore, a large number of imputed values is required, meaning that the output of MIPCA should be obtained for a parameter `nboot` greater than 100.

3 Multiple imputation for categorical data

The MIMCA method performs MI for categorical data using MCA (Audigier et al., 2015b). The rationale of the method consists in performing a non-parametric bootstrap of the individuals to reflect the uncertainty on the parameters of the imputation model; next in imputing the incomplete disjunctive table according to the MCA parameters estimated from each bootstrap replicate; and finally, in proposing categories from it by making a random draw for each missing entry. More precisely, several weightings are first defined for the individuals. Then, the iterative regularized MCA algorithm (Josse et al., 2012) is applied according to each weighting for a predefined number of dimensions. It leads to several imputations of the disjunctive table. These imputed tables are scaled to verify the constraint that the sum is equal to one per variable and per individual. Lastly, missing categories are drawn from the probabilities given by the imputed tables. Thus, the multiply imputed categorical data set is obtained.

The function `MIMCA` takes as an input the incomplete data set (`X`), the number of imputed data sets (`nboot`), and the number of dimensions (`ncp`). The imputed data sets are available in the output `res.MI`. The function can be applied on the incomplete data set *TitanicNA* as follows:

```
> ## Number of components
> res.ncp <- estim_ncpMCA(TitanicNA,method.cv="loo")
> plot(res.ncp$criterion~names(res.ncp$criterion),xlab="number of dimensions")
> ## Multiple Imputation
> res.MIMCA <- MIMCA(TitanicNA, ncp=5)
> ## First completed data matrix
> res.MIMCA$res.MI[[1]]
```

To assess the imputation of missing categorical values according to the MIMCA method, it is also possible to compare the distributions of the imputed values and the observed ones. However, the bivariate clouds cannot be represented for categorical variables. A way to achieve this goal consists in visualising the contingency table of one imputed data set, while distinguish the counts according to the fact whether missing data occurs on a variable (or a set of variables) or not. For a MCAR mechanism, equal counts for the imputed data set are expected if missing values occur or not on other variables. The function `mosaicMiss` from the R package VIM proposes such a diagnostic. Due to the fact that the output could be non-intuitive, it is rather recommended for experienced user.

4 Applying a statistical method

MI aims to apply a statistical method on an incomplete data set. To apply such a method on the multiply imputed data set obtained from the function `MIPCA` or `MIMCA`, one simple way is to use the R package mice (Van Buuren and Groothuis-Oudshoorn, 2014). The function `with.mids` performs analysis from an object of class `mids`, while the function `pool` pools the analysis results. The function `prelim` can be used to preliminary transform the output of `MIPCA` or `MIMCA` as an mids object.

For instance, to predict the survival status of the passengers of the ocean liner Titanic according to their age, sex and economic status, a logistic regression model can be applied as follows:

```
> #1/ transform the mimca output as a mids object
> imp<-prelim(res.MIMCA,TitanicNA)
> #2/ perform analysis
> fit <- with(data=imp,exp=glm(SURV~CLASS+AGE+SEX,family = "binomial"))
> #3/ pool the analysis results
> summary(pool(fit))
```

```
                   est        se           t       df     Pr(>|t|)      lo 95
(Intercept)  2.2847406 0.4748161   4.8118428 212.4985 2.833765e-06  1.3487876
CLASS2       0.8333108 0.1994424   4.1782034 514.5267 3.452037e-05  0.4414892
CLASS3      -0.1144325 0.2395882  -0.4776217 327.2261 6.332383e-01 -0.5857600
CLASS4      -0.9305057 0.2055680  -4.5265112 333.3670 8.355867e-06 -1.3348796
AGE2        -0.9646376 0.4045235  -2.3846270 188.3181 1.808946e-02 -1.7626172
SEX2        -2.6077296 0.1738127 -15.0031050 555.0386 0.000000e+00 -2.9491406
                 hi 95 nmis       fmi    lambda
(Intercept)  3.2206937   NA 0.5990257 0.5952694
CLASS2       1.2251323   NA 0.3530771 0.3505674
CLASS3       0.3568950   NA 0.4700070 0.4667775
CLASS4      -0.5261318   NA 0.4648453 0.4616443
AGE2        -0.1666580   NA 0.6380770 0.6342536
SEX2        -2.2663185   NA 0.3351248 0.3327334
```

In addition to providing a punctual estimate and a estimate of its associated variability for each quantity of interest, the `pool` function provides interesting outputs such as the bounds of the 95% confidence intervals or the fraction of missing information (fmi). This last quantity can be interpreted as the part of variability due to missing values. Large values (*e.g.* fmi$> .5$) indicate that the results are sensitive to the MI method used. In such a case, the comparison of the obtained results with the ones obtained by listwise deletion is recommended. A smaller variance for estimator obtained by MI is expected. If differences occur in the point estimates, it makes sense to attach more trust to the MI results since MI is theoretically superior to Listwise deletion. However, it remains important to try to explain differences.

Listwise deletion is the default method of the `glm` function to deal with missing values. Inference from this method can be simply obtained as follows:

```
> summary(glm(SURV~CLASS+AGE+SEX,family = "binomial",data=TitanicNA))

Call:
glm(formula = SURV ~ CLASS + AGE + SEX, family = "binomial",
    data = TitanicNA)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1622  -0.6887  -0.6695   0.4506   2.1979

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.34395    0.52070   4.502 6.75e-06 ***
CLASS1       0.83232    0.23980   3.471 0.000519 ***
CLASS2       0.03716    0.27416   0.136 0.892176
CLASS3      -1.00372    0.24802  -4.047 5.19e-05 ***
AGE1        -0.94021    0.42788  -2.197 0.027993 *
SEX1        -2.72175    0.23895 -11.391  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1120.73  on 897  degrees of freedom
Residual deviance:  859.11  on 892  degrees of freedom
  (1303 observations deleted due to missingness)
AIC: 871.11

Number of Fisher Scoring iterations: 4
```

As expected, the variability of the estimators is larger with listwise deletion, while the point estimates are close to the ones obtained by MI since the mechanism is MCAR.

Note that other packages than mice can be used to make inference from the multiply imputed data set. For instance, we mention the `Zelig` package (Owen et al., 2013), which provides a large range of analysis models.

BIBLIOGRAPHY

T. Allison and D. Chichetti. Sleep in mammals: ecological and constitutional correlates. *Science*, 194(4266):732–734, 1976. (page 4)

V. Audigier, F. Husson, and J. Josse. Multiple imputation for continuous variables using a bayesian principal component analysis. *Journal of Statistical Computation and Simulation*, 2015a. doi: 10.1080/00949655.2015.1104683. (page 14)

V. Audigier, F. Husson, and J. Josse. MIMCA: Multiple imputation for categorical variables with multiple correspondence analysis. *ArXiv e-prints*, 2015b. (page 18)

M. Blackwell, J. Honaker, and G. King. A unified approach to measurement error and missing data: Overview and applications. *Sociological Methods and Research*, pages 1–39, 2015. (page 16)

J. Honaker, G. King, and M. Blackwell. *Amelia II: A Program for Missing Data*, 2014. R package version 1.7.2. (page 17)

F. Husson and J. Josse. *missMDA: Handling Missing Values with Multivariate Data Analysis*, 2015. URL `http://www.agrocampus-ouest.fr/math/husson,http://juliejosse.com/`. R package version 1.9. (page 10)

J. Josse and F. Husson. Multiple imputation in PCA. *Advances in data analysis and classification*, 5:231–246, 2011. (pages 11 et 14)

J. Josse and F. Husson. Handling missing values in exploratory multivariate data analysis methods. *Journal de la Société Française de Statistique*, 153 (2):1–21, 2012. (page 11)

J. Josse and F. Husson. missmda a package to handle missing values in principal component methods. *Journal of Statistical Software*, 2015. (page 10)

J. Josse, M. Chavent, B. Liquet, and F. Husson. Handling missing values with regularized iterative multiple correspondence analysis. *Journal of Classification*, 29:91–116, 2012. (pages 12 et 18)

M. Owen, K. Imai, G. King, and O. Lau. *Zelig: Everyone's Statistical Software*, 2013. URL http://CRAN.R-project.org/package=Zelig. R package version 4.2-1. (page 20)

D. B. Rubin. *Multiple Imputation for Non-Response in Survey*. Wiley, New-York, 1987. (page 5)

M. Templ, A. Alfons, A. Kowarik, and B. Prantner. *VIM: Visualization and Imputation of Missing Values*, 2015. URL http://CRAN.R-project.org/package=VIM. R package version 4.3.0. (page 4)

S. Van Buuren and K. Groothuis-Oudshoorn. *mice*, 2014. R package version 2.22. (page 19)

M. Verbanck, J. Josse, and F. Husson. Regularised PCA to denoise and visualise data. *Statistics and Computing*, pages 1–16, 2013. ISSN 0960-3174, 1573-1375. doi: 10.1007/s11222-013-9444-y. (page 14)