Statistical Matching and Imputation of Survey Data with the Package StatMatch for the R Environment[*]

Marcello D'Orazio
Italian National Institute of Statistics (Istat), Rome, Italy
E-mail: madorazi@istat.it

April 4, 2012

1 Introduction

Statistical matching techniques aim at integrating two or more data sources, usually data from sample surveys carried out on the same target population. In the basic statistical matching framework, there are two data sources A and B sharing a set of variables X then the variable Y is available only in A while the variable Z is observed just in B. In practice the X variables are common to both the data sources, while the variables Y and Z are not jointly observed. The objective of statistical matching (hereafter denoted as SM) consists in integrating A and B in order to investigate the relationship between Y and Z. It is worth noting that the units in the two data sources come without an identifying code that permits to discover whether the same units has been observed in both the surveys; generally, in sample surveys, the chance the same units is included in both the surveys is close to zero.

The objective of SM can be achieved through a "micro" or a "macro" approach (D'Orazio *et al.*, 2006b). In the micro approach SM aims at creating a "synthetic" data source in which all the variables, X, Y and Z, are available (usually A filled in with the values of Z). In the macro approach the data sources are used to derive an estimate of the parameter of interest, e.g. the correlation coefficient between Y and Z or the contingency table $Y \times Z$. SM can be performed in a parametric or in a nonparametric framework. The parametric approach requires the explicit adoption of a model for (X, Y, Z); obviously if the model is misspecified then the results will not be reliable. The nonparametric approach is more flexible in handling complex situations (mixed type

[*]This work was developed in the framework of the ESSnet project on Data Integration, partly funded by Eurostat (December 2009–December 2011). For more information on the project visit http://www.essnet-portal.eu/di/data-integration

Table 1: Objectives and approaches to Statistical matching.

Objectives of Statistical matching	Approaches to statistical Matching		
	Parametric	Nonparametric	Mixed
MAcro	yes	yes	no
MIcro	yes	yes	yes

variables). The two approaches can be mixed: first a parametric model is assumed and its parameters are estimated then a completed synthetic data set is derived through a nonparametric micro approach. In this manner the advantages of both parametric and nonparametric approaches are maintained: the model is parsimonious while nonparametric techniques offer protection against model misspecification. An interesting comparison of some mixed methods that deal with continuous X, Y and Z variables is carried out by (Moriarity and Scheuren, 2001, 2003). A further comparison is available in (D'Orazio *et al.*, 2005). Table 1 provides a summary of the objectives and approaches to SM (D'Orazio *et al.*, 2008).

It is worth noting that in the traditional SM framework, when only A and B are available, all the SM methods (parametric, nonparametric and mixed) that use the set of common variables X to match A and B, implicitly assume the *conditional independence* (CI) of Y and Z given X:

$$f(x,y,z) = f(y|x) \times f(z|x) \times f(x)$$

This assumption is particularly strong and seldom holds in practice. In order to avoid the CI assumption the SM should incorporate some auxiliary information concerning the relationship between Y and Z (see Chap. 3 in D'Orazio *et al.* 2006b). The auxiliary information can be at micro level (a new data source in which Y and Z or X, Y and Z are jointly observed) or at macro level (e.g. an estimate of the correlation coefficient ρ_{XY} or an estimate of the contingency table $Y \times Z$, etc.) or simply consist of some logic constraints about the relationship between Y and Z (structural zeros, etc.; for further details see D'Orazio *et al.*, 2006a).

An alternative approach to SM consists in evaluating the *uncertainty* concerning an estimate of the parameter of interest. This uncertainty is due to the lack of joint information concerning Y and Z. For instance, let us consider a SM application whose target consists in estimating the correlation matrix of the trivariate normal distribution holding for (X,Y,Z); in the basic SM framework the available data allow to estimate all the components of the correlation matrix with the exception of ρ_{YZ}; in this case, due to the properties of the correlation matrix (has to be semidefinite positive), it is possible to conclude that:

$$\rho_{XY}\rho_{XZ} - \sqrt{\left(1-\rho_{YX}^2\right)\left(1-\rho_{XZ}^2\right)} \leq \rho_{YZ} \leq \rho_{XY}\rho_{XZ} + \sqrt{\left(1-\rho_{YX}^2\right)\left(1-\rho_{XZ}^2\right)}$$

The higher is the correlation between X and Y and between X and Z, the shorter will be the interval and consequently the lower will be the uncertainty. In practical applications, by substituting the unknown correlation coefficient with the corresponding estimates it is possible to derive a “range” of admissible values of the unknown ρ_{YZ}. The topic of the uncertainty will be covered in the Section 6.

Section 2 will be discuss some practical aspects concerning the preliminary steps necessary to apply SM techniques, with a particular emphasis on the choice of the marching variables; moreover some example data will be introduced in Section 2.1. In Section 3 some approaches to SM at micro level based on nonparametric methods will be shown. Section 4 is devoted to the mixed approaches to SM when dealing with continuous variables. Section 5 will discuss approaches to SM when dealing with data arising from complex sample surveys from finite populations.

2 Practical steps in an application of statistical matching

Before applying SM methods in order to integrate two or more data sources some decisions and preprocessing steps are required (Scanu, 2008). In practice, given two data sources A and B the following steps are necessary:

- Choice of the target variables Y and Z, i.e. of the variables observed distinctly in two sample surveys.
- Identification of all the common variables X shared by A and B. In this step some harmonization procedures may be required because of different definitions and/or classifications. Obviously, if two similar variables can not be harmonized they have to be discarded. The common variables should not present missing values and the observed values should be accurate (low or absent measurement error). It is worth noting that the common variables in the two data sources should have the same marginal/joint distribution, if A and B are representative samples of the same population.
- Potentially all the X variables can be used as matching variables but actually, not all them are used in the SM. Section 2.2 will provide more details concerning this topic.
- The choice of the matching variables is strictly related to the choice concerning the matching framework, i.e. micro or macro objective, parametric, nonparametric or a mixed approach etc.
- Once decided the framework, a SM technique is used to match the samples.
- Finally the results of the matching, whereas possible, should be evaluated.

2.1 Example data

The next Sections will provide a series of examples of application of some SM techniques in the R environment (R Development Core Team, 2011) by using the functions provided by the package **StatMatch** (D’Orazio, 2011). These examples will refer to data derived from the data set `eusilcS` contained in the package **simPopulation** (Alfons and Kraft, 2011). This is an artificial data set generated from real Austrian EU-SILC (European Union Statistics on Income and Living Conditions) survey data containing 11 725 observations on 18 variables (see `eusilcS` help pages for details):

```
> library(simPopulation) #loads pkg simPopulation
> data(eusilcS)
> str(eusilcS)
```

```
'data.frame':        11725 obs. of  18 variables:
 $ db030    : int  1 1 2 3 4 4 4 5 5 5 ...
 $ hsize    : int  2 2 1 1 3 3 3 5 5 5 ...
 $ db040    : Factor w/ 9 levels "Burgenland","Carinthia",..: 4 4 7 5 7 7 7 4 4 4 ...
 $ age      : int  72 66 56 67 70 46 37 41 35 9 ...
 $ rb090    : Factor w/ 2 levels "male","female": 1 2 2 2 2 1 1 1 2 2 ...
 $ pl030    : Factor w/ 7 levels "1","2","3","4",..: 5 5 2 5 5 3 1 1 3 NA ...
 $ pb220a   : Factor w/ 3 levels "AT","EU","Other": 1 1 1 1 1 1 3 1 1 NA ...
 $ netIncome: num  22675 16999 19274 13319 14366 ...
 $ py010n   : num  0 0 19274 0 0 ...
 $ py050n   : num  0 0 0 0 0 ...
 $ py090n   : num  0 0 0 0 0 ...
 $ py100n   : num  22675 0 0 13319 14366 ...
 $ py110n   : num  0 0 0 0 0 0 0 0 0 NA ...
 $ py120n   : num  0 0 0 0 0 0 0 0 0 NA ...
 $ py130n   : num  0 16999 0 0 0 ...
 $ py140n   : num  0 0 0 0 0 0 0 0 0 NA ...
 $ db090    : num  7.82 7.82 8.79 8.11 7.51 ...
 $ rb050    : num  7.82 7.82 8.79 8.11 7.51 ...
```

In order to use these data to show how SM works, some manipulations are needed to discard not relevant units (obs. with `age<16`, whose income and personal economic status are missing), to categorize some variables, etc.

```
> # discard units with age<16
> silc.16 <- subset(eusilcS, age>15) # units
> nrow(silc.16)
```

```
[1] 9522
```

```
> #
> # categorize age
> silc.16$c.age <- cut(silc.16$age, c(16,24,49,64,100), include.lowest=T)
> #
> # truncate hsize
> aa <- as.numeric(silc.16$hsize)
> aa[aa>6] <- 6
> silc.16$hsize6 <- factor(aa, ordered=T)
> #
> # recode personal economic status
> aa <- as.numeric(silc.16$pl030)
> aa[aa<3] <- 1
> aa[aa>1] <- 2
> silc.16$work <- factor(aa, levels=1:2, labels=c("working","not working"))
> #
> # categorize personal net income
> silc.16$c.netI <- cut(silc.16$net/1000,
+                        breaks=c(-6,0,5,10,15,20,25,30,40,50,200))
```

In order to reproduce the basic SM framework, the data frame `silc.16` is split randomly in two data sets: `rec.A` consisting of 4 000 observations and `don.B` with the remaining 5 522 units. The two data frames `rec.A` and `don.B` share the variables `X.vars`; the person's economic status (`y.var`) is available only in `rec.A` while the net income (`z.var`) is available in `don.B`.

```
> set.seed(123456)
> obs.A <- sample(nrow(silc.16), 4000, replace=F)
> X.vars <- c("hsize","hsize6","db040","age","c.age",
+             "rb090","pb220a","rb050")
> y.var <- c("pl030","work")
> z.var <- c("netIncome", "c.netI")
> rec.A <- silc.16[obs.A, c(X.vars, y.var)]
> don.B <- silc.16[-obs.A, c(X.vars, z.var)]
> #
> # determine a rough weighting
> # compute N, the est. size of pop(age>16)
> N <- round(sum(silc.16$rb050))
> N

[1] 67803

> #rescale origin weights
> rec.A$wwA <- rec.A$rb050/sum(rec.A$rb050)*N
> don.B$wwB <- don.B$rb050/sum(don.B$rb050)*N
```

2.2 The choice of the matching variables

In SM A and B, may share many common variables. In practice, not all the common variables are used in SM but just the most relevant ones. The selection of the common variables to be used in SM, usually called *matching variables*, should be performed through opportune statistical methods (descriptive, inferential, etc.) and by consulting subject matter experts.

From a statistical point of view, the choice of the marching variables X_M ($X_M \subseteq X$) should be carried out in a "multivariate sense" in order to identify the subset of the X_M variables connected at the same time with Y and Z (Cohen, 1991); unfortunately this would require the availability of an auxiliary data source in which all the variables (X, Y, Z) are observed. In the basic SM framework the data in A permit to explore the relationship between Y and X, while the relationship between Z and X can be investigated in the file B. Then the results of the two separate analyses have to be combined in some manner; usually the subset of the matching variables is obtained as $X_M = X_Y \cup X_Z$, being X_Y the subset of the common variables that better explains Y ($X_Y \subseteq X$), while X_Z is the subset of the common variables that better explain Z ($X_Z \subseteq X$).

The simplest procedure to identify X_Y consists in fitting regression models to data available in A. The same procedure is carried out with data in B in order to identify the subset of the common variables X_Z that better explains Z.

The following R examples provide an idea of how to proceed. In particular, in the example data, the variable Y in A is a categorical binary variable (`work`), therefore a logistic regression model is fitted in order to identify X_Y.

```
> # analyses on A
> # logistic regression
> work.glm <- glm(work~hsize+db040+age+rb090+pb220a, data=rec.A,
+                  family=binomial(link = "logit"))
> summary(work.glm)

Call:
glm(formula = work ~ hsize + db040 + age + rb090 + pb220a, family = binomial(link = "log
    data = rec.A)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7327  -0.9511  -0.5405   0.9211   2.2492

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)        -3.105822   0.259524 -11.967   <2e-16 ***
hsize               0.010677   0.025941   0.412   0.6807
db040Carinthia     -0.065318   0.229838  -0.284   0.7763
db040Lower Austria -0.207209   0.209499  -0.989   0.3226
db040Salzburg      -0.028442   0.236368  -0.120   0.9042
db040Styria        -0.102106   0.212338  -0.481   0.6306
db040Tyrol         -0.258700   0.225449  -1.147   0.2512
db040Upper Austria -0.135302   0.210093  -0.644   0.5196
```

```
db040Vienna        -0.193611   0.214358  -0.903   0.3664
db040Vorarlberg     0.008169   0.250389   0.033   0.9740
age                 0.055375   0.002434  22.750   <2e-16 ***
rb090female         0.880804   0.072125  12.212   <2e-16 ***
pb220aEU           -0.237560   0.239145  -0.993   0.3205
pb220aOther         0.242078   0.141839   1.707   0.0879 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 5513.8  on 3999  degrees of freedom
Residual deviance: 4607.6  on 3986  degrees of freedom
AIC: 4635.6

Number of Fisher Scoring iterations: 3
```

In order to identify X_Y an automatic selection procedure can be used (forward, backward or stepwise selection):

```
> # stepwise selection
> new.work.glm <- step(work.glm, trace=0)
> summary(new.work.glm)

Call:
glm(formula = work ~ age + rb090 + pb220a, family = binomial(link = "logit"),
    data = rec.A)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7394  -0.9536  -0.5341   0.9137   2.1955

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.198019   0.123036 -25.993   <2e-16 ***
age          0.055118   0.002277  24.212   <2e-16 ***
rb090female  0.879327   0.072009  12.211   <2e-16 ***
pb220aEU    -0.276207   0.236205  -1.169   0.2423
pb220aOther  0.244811   0.139171   1.759   0.0786 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 5513.8  on 3999  degrees of freedom
Residual deviance: 4612.5  on 3995  degrees of freedom
AIC: 4622.5

Number of Fisher Scoring iterations: 3

> # X_Y
> X.Y.glm <- c("age","rb090", "pb220a")
```

According to the automatic selection procedure it comes out that X_Y is composed by the three variables "age", "rb090" and "pb220a".

In complex situations when a nonlinear relationship is supposed to exist, the selection of the subset X_Y can be demanded to nonparametric procedures such as *Classification And Regression Trees* (Breiman *et al.*, 1984). Instead of fitting a single tree it may be better to fit a *random forest* (Breiman, 2001) by using the functions available in the package **randomForest** (Liaw and Wiener, 2002). This technique provides a measure of importance for the predictors (that has to be used with caution). Alternatively *conditional inference trees* (Hothorn *et al.*, 2006) can be fitted; in this case the tree fitting is based on hypothesis testing and by considering explicitly the association between responses and covariates. In R these procedures are made available by the functions in the package **party** (Hothorn *et al.*, 2006).

The same analysis carried out in A should be carried out in B. In the example data, the target variable Z in B is the netIncome (better to consider its log).

```
> summary(don.B$netIncome)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  -4373    6080   14130   14810   20810  185500

> don.B$lognetI <- log(don.B$netIncome+4373+1)
> summary(don.B$lognetI)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.2877  9.2550  9.8260  9.6520 10.1300 12.1500

> #regression
> lnetI.lm <- lm(lognetI~hsize+db040+age+rb090+pb220a, data=don.B)
> summary(lnetI.lm)

Call:
lm(formula = lognetI ~ hsize + db040 + age + rb090 + pb220a,
    data = don.B)

Residuals:
    Min      1Q  Median      3Q     Max
```

```
-9.7620 -0.3785  0.0878  0.4253  2.2285

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)         9.7753007  0.0551851 177.137  < 2e-16 ***
hsize              -0.0596160  0.0062136  -9.594  < 2e-16 ***
db040Carinthia      0.0451692  0.0498676   0.906  0.36509
db040Lower Austria  0.0121853  0.0446170   0.273  0.78478
db040Salzburg       0.1303552  0.0520001   2.507  0.01221 *
db040Styria         0.0381493  0.0452166   0.844  0.39887
db040Tyrol          0.0701096  0.0486102   1.442  0.14928
db040Upper Austria  0.0844992  0.0446713   1.892  0.05860 .
db040Vienna         0.1364211  0.0455115   2.998  0.00273 **
db040Vorarlberg     0.0722760  0.0551028   1.312  0.18969
age                 0.0060719  0.0005054  12.015  < 2e-16 ***
rb090female        -0.5218040  0.0169144 -30.850  < 2e-16 ***
pb220aEU           -0.0897179  0.0584009  -1.536  0.12454
pb220aOther        -0.2975103  0.0353512  -8.416  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6268 on 5508 degrees of freedom
Multiple R-squared: 0.2029,       Adjusted R-squared: 0.201
F-statistic: 107.9 on 13 and 5508 DF,  p-value: < 2.2e-16

> # stepwise selection
> new.lnetI.lm <- step(lnetI.lm, trace=0)
> summary(new.lnetI.lm)

Call:
lm(formula = lognetI ~ hsize + db040 + age + rb090 + pb220a,
    data = don.B)

Residuals:
    Min      1Q  Median      3Q     Max
-9.7620 -0.3785  0.0878  0.4253  2.2285

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)         9.7753007  0.0551851 177.137  < 2e-16 ***
hsize              -0.0596160  0.0062136  -9.594  < 2e-16 ***
db040Carinthia      0.0451692  0.0498676   0.906  0.36509
db040Lower Austria  0.0121853  0.0446170   0.273  0.78478
db040Salzburg       0.1303552  0.0520001   2.507  0.01221 *
```

```
db040Styria         0.0381493  0.0452166   0.844  0.39887    
db040Tyrol          0.0701096  0.0486102   1.442  0.14928    
db040Upper Austria  0.0844992  0.0446713   1.892  0.05860 .  
db040Vienna         0.1364211  0.0455115   2.998  0.00273 ** 
db040Vorarlberg     0.0722760  0.0551028   1.312  0.18969    
age                 0.0060719  0.0005054  12.015  < 2e-16 ***
rb090female        -0.5218040  0.0169144 -30.850  < 2e-16 ***
pb220aEU           -0.0897179  0.0584009  -1.536  0.12454    
pb220aOther        -0.2975103  0.0353512  -8.416  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6268 on 5508 degrees of freedom
Multiple R-squared: 0.2029,	Adjusted R-squared: 0.201 
F-statistic: 107.9 on 13 and 5508 DF,  p-value: < 2.2e-16

> # X_Z
> X.Z.lm <- c("hsize","db040","age","rb090", "pb220a")
```

This regression analysis ends with a model including all the available predictors. Then, according to the results of the two separate analyses on the example data, if it is decided that the matching variables are obtained as $X_M = X_Y \cup X_Z$, it comes out that all the available common variables should be used in the SM application.

```
> # the matching variables
> union(X.Y.glm, X.Z.lm)

[1] "age"    "rb090"  "pb220a" "hsize"  "db040" 
```

If a smaller subset of the matching variables is necessary, then by considering $X_M = X_Y \cap X_Z$, it would result a subset consisting of just three X variables:

```
> #the smallest subset of matching variables
> intersect(X.Y.glm, X.Z.lm)

[1] "age"    "rb090"  "pb220a"
```

The approach to SM based on the study of uncertainty offers the possibility of choosing the matching variables in a better way: by selecting just those common variables with the highest contribution to the reduction of the uncertainty. The function `Fbwidths.by.x` in **StatMatch** permits to explore the reduction of uncertainty when all the variables (X, Y, Z) are categorical. In particular, in the basic SM framework it is possible to show that

$$P_{j,k}^{(low)} \leq P(Y = j, Z = k) \leq P_{j,k}^{(up)},$$

being

$$
\begin{aligned}
P_{j,k}^{(low)} &= \sum_i P(X=i) \max\{0; P(Y=j|X=i) + P(Z=k|X=i) - 1\} \\
P_{j,k}^{(up)} &= \sum_i P(X=i) \min\{P(Y=j|X=i); P(Z=k|X=i)\}
\end{aligned}
$$

for $j = 1, \dots, J$ and $k = 1, \dots, K$, being J and K the categories of Y and Z respectively.

The function `Fbwidths.by.x` estimates $\left[P_{j,k}^{(low)}, P_{j,k}^{(up)}\right]$ for each cell in the contingency table $Y \times Z$ in correspondence of all the possible combinations of the X variables; then the reduction of uncertainty is measured naively by considering the average widths of the intervals:

$$\bar{d} = \frac{1}{J \times K} \sum_{j=1}^{J} \sum_{k=1}^{K} (\hat{P}_{j,k}^{(up)} - \hat{P}_{j,k}^{(low)})$$

```
> xx <- xtabs(~db040+hsize6+c.age+rb090+pb220a, data=rec.A)
> xy <- xtabs(~db040+hsize6+c.age+rb090+pb220a+work, data=rec.A)
> xz <- xtabs(~db040+hsize6+c.age+rb090+pb220a+c.netI, data=don.B)
> #
> library(StatMatch) #loads StatMatch
> out.fbw <-  Fbwidths.by.x(tab.x=xx, tab.xy=xy, tab.xz=xz)
> # average widhts of uncertainty bounds
> out.fbw$av.widths
```

```
                                n.vars   av.width
|db040                               1 0.10000000
|hsize6                              1 0.10000000
|pb220a                              1 0.10000000
|rb090                               1 0.10000000
|c.age                               1 0.08439346
|hsize6+rb090                        2 0.10000000
|db040+rb090                         2 0.10000000
|rb090+pb220a                        2 0.09997147
|db040+hsize6                        2 0.09985849
|hsize6+pb220a                       2 0.09977896
|db040+pb220a                        2 0.09970130
|c.age+pb220a                        2 0.08432518
|db040+c.age                         2 0.08374131
|hsize6+c.age                        2 0.08282457
|c.age+rb090                         2 0.07738444
|hsize6+rb090+pb220a                 3 0.09918992
|db040+rb090+pb220a                  3 0.09869455
|db040+hsize6+pb220a                 3 0.09768879
```

```
|db040+hsize6+rb090                       3 0.09688779
|db040+c.age+pb220a                       3 0.08171158
|hsize6+c.age+pb220a                      3 0.08137625
|db040+hsize6+c.age                       3 0.07711730
|c.age+rb090+pb220a                       3 0.07642623
|hsize6+c.age+rb090                       3 0.07623534
|db040+c.age+rb090                        3 0.07580610
|db040+hsize6+rb090+pb220a                4 0.09232628
|hsize6+c.age+rb090+pb220a                4 0.07360024
|db040+hsize6+c.age+pb220a                4 0.07197990
|db040+c.age+rb090+pb220a                 4 0.07175106
|db040+hsize6+c.age+rb090                 4 0.06411282
|db040+hsize6+c.age+rb090+pb220a          5 0.05775534
```

The results show that there is a marked reduction of the average width when passing from three to four common variables; the model with the matching variables `"db040"`, `"hsize6"`, `"c.age"` and `"rb090"` seems a good compromise among reduction of uncertainty and the necessity of not having too many matching variables. If there is the need of having less matching variables, in this example one should consider just two variables: `"c.age"` and `"rb090"`; in fact, the models with three matching variables do not provide a great decrease of average uncertainty with respect to the ones with just two.

The choice of the matching variables is a crucial phase in most of the SM applications. Choosing too many variables increases the complexity of the problem and may affect negatively the results of SM. In particular, in the micro approach it may introduce additional undesired variability and bias in estimating the joint (marginal) distribution of X_M and Z.

3 Nonparametric micro techniques

Nonparametric approach is very popular in SM when the objective is micro, i.e. the creation of a synthetic data set. Most of the nonparametric micro approaches consist in filling in the data set chosen as the *recipient* with the values of the variable which is available only in the other data set, the *donor* one. In this approach it is important to decide which data set plays the role of the recipient. Usually this is the data set which should be used ad the basis for further statistical analysis; a natural choice would appear that of using the larger ones because it would provide more accurate results. Unfortunately, such a way of working may provide inaccurate SM results, especially when the sizes of the two data sources are very different. The reason is quite simple, the larger is the recipient with respect to the donor, the more times a unit in the latter could be selected as a donor; in this manner, there is a high risk that the distribution of the imputed variable does not reflect the original one (estimated form the donor data set).

In the following it will be assumed that A is the recipient while B is the donor, being $n_A \leq n_B$ (n_A and n_B are the sample sizes of A and B respectively). Hence the objective of SM will be that of filling in A with values of Z (variable available only in B).

In **StatMatch** the following nonparametric micro techniques are available: *random hot deck*, *nearest neighbor hot deck* and *rank hot deck* (see Section 2.4 in D'Orazio *et al.*, 2006b; Singh *et al.*, 1993).

3.1 Nearest neighbor distance hot deck

The nearest neighbor distance hot deck techniques is implemented in the function `NND.hotdeck`. This function searches in `data.don` the nearest neighbor of each unit in `data.rec` according to a distance computed on the matching variables X_M specified with the argument `match.vars`. By default the Manhattan (city block) distance is considered (`dist.fun="Manhattan"`). In order to reduce the computation effort due to computation of distances it is preferable to define some donation classes (argument `don.class`): for a record in given imputation class it will be selected a donor in the same class (the distances are computed only among units belonging to the same class). Usually, the donation classes are defined according to one or more categorical common variables (geographic area, etc.). In the following, a simple example of usage of `NND.hotdeck` is reported.

```
> group.v <- c("rb090","db040")
> X.mtc <- c("hsize","age")
> out.nnd <- NND.hotdeck(data.rec=rec.A, data.don=don.B,
+                          match.vars=X.mtc, don.class=group.v,
+                          dist.fun="Manhattan")

Warning: The  Manhattan  distance is being used
All the categorical matching variables in rec and don
 data.frames, if present are recoded into dummies
```

The function `NND.hotdeck` does not create the synthetic data set; for each unit in A the corresponding closest donor in B is identified according to the chosen distance function; the recipient-donor units' identifiers are saved in the data.frame `mtc.ids` stored in the output list returned by `NND.hotdeck`. The output list provides also the distance between each couple recipient-donor (saved in the `dist.rd` component of the output list) and the number of available donors at the minimum distance for each recipient (component `noad`). Note that when there are more donors at the minimum distance then one of them is picked up at random.

```
> summary(out.nnd$dist.rd) # summary distances rec-don

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 0.0000  0.0000  0.0000  0.3278  0.0000  8.0000 

> summary(out.nnd$noad) # summary available donors at min. dist.
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.000   1.000   2.000   2.882   4.000  16.000 

> table(out.nnd$noad)

   1    2    3    4    5    6    7    8    9   10   11   12   13 
1288  921  638  421  275  192  113   49   39   22   11   18    4 
  14   16 
   6    3 
```

In order to derive the synthetic data set it is necessary to run the function `create.fused`:

```
> head(out.nnd$mtc.ids)

     rec.id don.id
[1,] "401"  "376" 
[2,] "71"   "118" 
[3,] "92"   "106" 
[4,] "225"  "253" 
[5,] "364"  "288" 
[6,] "370"  "380" 

> fA.nnd.m <- create.fused(data.rec=rec.A, data.don=don.B,
+                          mtc.ids=out.nnd$mtc.ids,
+                          z.vars=c("netIncome","c.netI"))
> head(fA.nnd.m) #first 6 obs.

    hsize hsize6      db040 age    c.age rb090 pb220a   rb050
401     5      5 Burgenland  45  (24,49]  male     AT 4.545916
71      2      2 Burgenland  65 (64,100]  male     AT 6.151409
92      2      2 Burgenland  81 (64,100]  male     AT 6.151409
225     3      3 Burgenland  51  (49,64]  male     AT 5.860364
364     4      4 Burgenland  18  [16,24]  male     AT 6.316554
370     5      5 Burgenland  50  (49,64]  male     AT 4.545916
    pl030        work      wwA netIncome  c.netI
401     1     working 10.85782  47159.21 (40,50]
71      5 not working 14.69250  21316.32 (20,25]
92      5 not working 14.69250  21667.53 (20,25]
225     1     working 13.99734  34166.20 (30,40]
364     1     working 15.08694  10228.02 (10,15]
370     1     working 10.85782   9456.25  (5,10]
```

As far as distances are concerned (argument `dist.fun`), all the distance functions in the package **proxy** (Meyer and Butchta, 2011) are available. Anyway, for some particular distances it was decided to write specific R functions. In particular, when dealing with continuous matching variables it is possible to use the *maximum distance* (L^∞ norm) implemented in `maximum.dist`; this function works on the true observed values (continuous variables) or on transformed ranked values (argument `rank=TRUE`) as suggested in (Kovar *et al.*, 1988); the transformation (ranks divided by the number of units) removes the effect of different scales and the new values are distributed in the interval $[0, 1]$. The Mahalanobis distance can be computed by using `mahalanobis.dist` which allows an external estimate of the covariance matrix (argument `vc`). When dealing with mixed type matching variables, the *Gowers's dissimilarity* (Gower, 1981) can be computed (function `gower.dist`): it is an average of the distances computed on the single variables according to different rules, depending on the type of the variable. All the distances are scaled to range from 0 to 1, hence the overall distance cat take a value in $[0, 1]$. It is worth noting that when dealing with mixed types matching variables it is still possible to use the distance functions for continuous variables but `NND.hotdeck` transforms factors into dummies (by means of the function `fact2dummy`).

```
> group.v <- c("rb090","db040")
> X.mtc <- c("hsize","age","pb220a")
> out.nnd.g <- NND.hotdeck(data.rec=rec.A, data.don=don.B,
+                          match.vars=X.mtc, don.class=group.v,
+                          dist.fun="Gower")
> summary(out.nnd.g$dist.rd) # summary distances rec-don

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.000000 0.000000 0.000000 0.005218 0.004329 0.148500

> fA.nnd.g <- create.fused(data.rec=rec.A, data.don=don.B,
+                     mtc.ids=out.nnd.g$mtc.ids,
+                     z.vars=c("netIncome","c.netI"))
```

By default `NND.hotdeck` does not pose constraints on the "usage" of donors: a record in the donor data set can be selected many times as a donor. The multiple usage of a donor can be avoided by resorting to a *constrained hot deck* (argument `constrained=TRUE` in `NND.hotdeck`); in such a case, a donor can be used just once and all the donors are selected in order to minimize the overall matching distance (assuming that $n_A \leq n_B$). In practice, the donors are identified by solving a traveling salesperson problem; two alternative algorithms are available the classic one (argument `constr.alg="lpSolve"`) (Berkelaar *et al.*, 2011) and the RELAX-IV algorithm (Bertsekas and Tseng, 1994) (argument `constr.alg="relax"`). This latter one is much faster but there are some restrictions on its license.

```
> group.v <- c("rb090","db040")
> X.mtc <- c("hsize","age")
> out.nnd.mc <- NND.hotdeck(data.rec=rec.A, data.don=don.B,
+                           match.vars=X.mtc, don.class=group.v,
+                            dist.fun="Manhattan", constrained=T,
+                            constr.alg="lpSolve")

Warning: The  Manhattan  distance is being used
All the categorical matching variables in rec and don
 data.frames, if present are recoded into dummies

> fA.nnd.mc <- create.fused(data.rec=rec.A, data.don=don.B,
+                     mtc.ids=out.nnd.mc$mtc.ids,
+                     z.vars=c("netIncome","c.netI"))
```

The constrained matching requires a higher computational effort but preserves better the marginal distribution of the variable imputed in the synthetic data set. Obviously the overall matching distance tends to be greater than the one in the unconstrained case.

```
> # comparing marginal distribution of C.netI
> tt.0 <- prop.table(xtabs(~c.netI, data=don.B))
> tt.m <- prop.table(xtabs(~c.netI, data=fA.nnd.m))
> tt.g <- prop.table(xtabs(~c.netI, data=fA.nnd.g))
> tt.mc <- prop.table(xtabs(~c.netI, data=fA.nnd.mc))
> tt <- cbind(origin=c(tt.0), m4.unc=c(tt.m),
+             g5.unc=c(tt.g), m4.constr=c(tt.mc))
> round(tt*100, 2)

         origin m4.unc g5.unc m4.constr
(-6,0]    12.89  13.18  12.38     12.78
(0,5]      9.25   9.18   9.10      9.50
(5,10]    14.03  13.80  13.85     13.95
(10,15]   17.35  17.38  17.40     17.15
(15,20]   18.60  18.68  18.95     18.65
(20,25]   13.27  12.68  12.62     13.25
(25,30]    6.61   7.00   7.50      6.53
(30,40]    5.11   5.22   4.98      5.22
(40,50]    1.32   1.25   1.52      1.38
(50,200]   1.56   1.65   1.70      1.60

> # distance wrt to the origin distr.
> 1/2*colSums(abs(tt-tt[,1]))

     origin      m4.unc      g5.unc   m4.constr
0.000000000 0.009848243 0.016385639 0.005115357

> #overall matching distances
> sum(out.nnd$dist.rd)  #unconstrained

[1] 1311

> sum(out.nnd.mc$dist.rd)  #constrained

[1] 2906
```

3.2 Random hot deck

The function `RANDwNND.hotdeck` carries out the random selection of each donor from a suitable subset of all the available donors. This subset can be formed in different ways, e.g. by considering all the donors sharing the same characteristics of the recipient (defined according to some X_M variables, such as geographic region, etc.) or simply the closest donors according to a particular rule. The traditional *random hot deck* within imputation classes (Singh *et al.*, 1993) is performed by simply specifying the donation classes via the argument `don.class` (the classes are formed by crossing the categories of the categorical variables being considered). For each record in the recipient data set in a given donation class, a donor is picked up completely at random within the same donation class.

```
> group.v <- c("db040","rb090")
> rnd.1 <- RANDwNND.hotdeck(data.rec=rec.A, data.don=don.B,
+                           match.vars=NULL, don.class=group.v)
> fA.rnd <- create.fused(data.rec=rec.A, data.don=don.B,
+                        mtc.ids=rnd.1$mtc.ids,
+                        z.vars=c("netIncome", "c.netI"))
```

As for `NND.hotdeck`, the function `RANDwNND.hotdeck` does not create the synthetic data set; the recipient-donor units' identifiers are saved in the component `mtc.ids` of the list returned in output by `RANDwNND.hotdeck`. The number of donors available in each donation class are saved in the component `noad`.

It is worth noting that the donors can also be selected with probability proportional to a weight (specified with the argument `weight.don`); an example will be provided in Section 5.

`RANDwNND.hotdeck` implements various alternative methods to restrict the subset of the potential donors. These methods are based essentially on a distance measure computed on the matching variables provided via the argument `match.vars`. In practice, when `cut.don="k.dist"` only the donors whose distance from the recipient is less or equal to threshold `k` are considered (see Andridge and Little, 2010). By setting `cut.don="exact"` the `k` ($0 < k \leq n_D$) closest donors are retained (n_D is the number of available donors for a given recipient). With `cut.don="span"` a proportion `k` ($0 < k \leq 1$) of the closest available donors it is considered while with `cut.don="rot"` only the subset reduces to the $[\sqrt{n_D}]$ closest donors. Finally, when `cut.don="min"` only the donors at the minimum distance from the recipient are retained.

```
> # random choiches of a donor among the closest k=20 wrt age
> group.v <- c("db040","rb090")
> X.mtc <- "age"
> rnd.2 <- RANDwNND.hotdeck(data.rec=rec.A, data.don=don.B,
+                           match.vars=X.mtc, don.class=group.v,
+                           dist.fun="Manhattan",
+                            cut.don="exact", k=20)
```

```
Warning: The  Manhattan  distance is being used
All the categorical matching variables in rec and don data.frames,
 if present, are recoded into dummies

> fA.knnd <- create.fused(data.rec=rec.A, data.don=don.B,
+                          mtc.ids=rnd.2$mtc.ids,
+                          z.vars=c("netIncome", "c.netI"))
```

When distances are computed on the matching variables, then the output of `RANDwNND.hotdeck` provides some information concerning the distances of the possible available donors for each recipient observation in the data.frame.

```
> head(rnd.2$sum.dist)

     min max       sd cut dist.rd
[1,]   0  47 11.02087   5       3
[2,]   0  49 14.54555   4       1
[3,]   0  65 19.01027   9       2
[4,]   1  41 10.09283   6       5
[5,]   1  74 19.53088  11       2
[6,]   0  42 10.16749   5       2
```

In particular, `"min"`, `"max"` and `"sd"` columns report respectively the minimum, the maximum and the standard deviation of the distances (all the available donors are considered), while `"cut"` refers to the distance of the `k`th closest donor; `"dist.rd"` is distance existing among the recipient and the randomly chosen donor.

```
> tt.0 <- prop.table(xtabs(~c.netI, data=don.B))
> tt.rnd <- prop.table(xtabs(~c.netI, data=fA.rnd))
> tt.knnd <- prop.table(xtabs(~c.netI, data=fA.knnd))
> tt <- cbind(origin=c(tt.0), rnd=c(tt.rnd), k.nnd=c(tt.knnd))
> round(tt*100, 2)

         origin   rnd k.nnd
(-6,0]    12.89 13.53 13.70
(0,5]      9.25  9.72  9.47
(5,10]    14.03 13.58 13.53
(10,15]   17.35 16.35 17.80
(15,20]   18.60 18.35 18.50
(20,25]   13.27 13.15 12.47
(25,30]    6.61  6.78  6.90
(30,40]    5.11  5.83  4.88
(40,50]    1.32  1.43  1.38
(50,200]   1.56  1.30  1.38
> # distance wrt to the origin distr.
> 1/2*colSums(abs(tt-tt[,1]))

    origin        rnd      k.nnd
0.00000000 0.02088473 0.01821532
```

When selecting a donor among those available in the subset identified by `cut.don` and `k` it is possible to use a weighted selection by specifying a weighting variable via `weight.don` argument. The aspects related to the usage of weights will be covered in Section 5.

3.3 Rank hot deck

The *rank hot deck distance* method has been introduced by Singh *et al.* (1993). It searches for the donor at a minimum distance from the given recipient record but, in this case, the distance is computed on the percentage points of the empirical cumulative distribution function of the unique (continuous) common variable X_M being considered. The empirical cumulative distribution function is estimated by:

$$\hat{F}(x_k) = \frac{1}{n}\sum_{i=1}^{n} I\left(x_i \leq x_k\right)$$

being $I() = 1$ if $x_i \leq x_k$ and 0 otherwise. This transformation provides values uniformly distributed in the interval $[0, 1]$; moreover, it can be useful when the values of X_M can not be directly compared because of measurement errors which however do not affect the "position" of a unit in the whole distribution (D'Orazio *et al.*, 2006b). This method is implemented in the function `rankNND.hotdeck`. In the following just a simple example of usage is reported.

```
> rnk.1 <- rankNND.hotdeck(data.rec=rec.A, data.don=don.B,
+                          var.rec="age", var.don="age")
> #create the synthetic data set
> fA.rnk <- create.fused(data.rec=rec.A, data.don=don.B,
+                        mtc.ids=rnk.1$mtc.ids,
+                        z.vars=c("netIncome", "c.netI"),
+                        dup.x=TRUE, match.vars="age")
> head(fA.rnk)

      hsize hsize6         db040 age    c.age  rb090 pb220a
4547      2      2     Carinthia  45  (24,49]   male     AT
9819      4      4      Salzburg  35  (24,49] female     AT
4461      2      2     Carinthia  57  (49,64]   male     AT
10222     2      2         Tyrol  69 (64,100] female     AT
8228      4      4 Upper Austria  25  (24,49] female     AT
3361      3      3        Vienna  22  [16,24]   male  Other
```

```
        rb050 pl030        work      wwA age.don netIncome
4547  6.863162     1     working 16.39250      45  14337.87
9819  6.089967     1     working 14.54575      35   9350.56
4461  6.863162     1     working 16.39250      58   4331.63
10222 6.857877     5 not working 16.37988      70  16343.10
8228  6.945309     4 not working 16.58871      25  13167.76
3361  8.374000     1     working 20.00110      22  11626.29
        c.netI
4547  (10,15]
9819   (5,10]
4461    (0,5]
10222 (15,20]
8228  (10,15]
3361  (10,15]
```

The function `rankNND.hotdeck` allows defining some donation classes with the argument `don.class`; in this case the empirical cumulative distribution is estimated separately class by class.

```
> rnk.2 <- rankNND.hotdeck(data.rec=rec.A, data.don=don.B, var.rec="age",
+                          var.don="age", don.class="db040")
> fA.grnk <- create.fused(data.rec=rec.A, data.don=don.B,
+                          mtc.ids=rnk.2$mtc.ids,
+                          z.vars=c("netIncome", "c.netI"),
+                          dup.x=TRUE, match.vars="age")
> head(fA.grnk)
```

```
    hsize hsize6      db040 age    c.age  rb090 pb220a    rb050
401     5      5 Burgenland  45  (24,49]   male     AT 4.545916
245     3      3 Burgenland  64  (49,64] female     AT 5.860364
71      2      2 Burgenland  65 (64,100]   male     AT 6.151409
92      2      2 Burgenland  81 (64,100]   male     AT 6.151409
86      2      2 Burgenland  27  (24,49] female     AT 6.151409
113     2      2 Burgenland  58  (49,64] female     AT 6.151409
    pl030        work      wwA age.don netIncome  c.netI
401     1     working 10.85782      39  15308.91 (15,20]
245     5 not working 13.99734      63  21639.69 (20,25]
71      5 not working 14.69250      64   8988.24  (5,10]
92      5 not working 14.69250      77  15648.92 (15,20]
86      2     working 14.69250      26  16607.60 (15,20]
113     1     working 14.69250      55  21745.87 (20,25]
```

In estimating the empirical cumulative distribution it is possible to consider the weights of the observations (arguments `weight.rec` and `weight.don`). This topic will be covered in Section 5.

3.4 Using functions in StatMatch to impute missing values in a survey

All the functions in **StatMatch** that implement the hot deck imputation techniques can be used to impute missing values in a single data set. In this case it is necessary to separate the observations in two data sets: the file A plays the role of recipient and will contain the units with missing values on the target variable while the file B is the donor and will contain all the available donors (units with non missing values for the target variable). In the following a simple example with the `iris` data.frame is reported.

```
> # introduce missing values in iris
> set.seed(1324)
> miss <- rbinom(150, 1, 0.30) #generates randomly missing
> iris.miss <- iris
> iris.miss$Petal.Length[miss==1] <- NA
> summary(iris.miss$Petal.L)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
    1.1     1.6     4.3     3.8     5.1     6.9      46 

> # separate units in two data sets
> rec <- subset(iris.miss, is.na(Petal.Length), select=-Petal.Length)
> don <- subset(iris.miss, !is.na(Petal.Length))
```

Once the starting data set has been split in two (recipient and donor) a nonparametric imputation technique can be used to search for a donor for each observation in the recipient data set. In the following example the nearest neighbor hotdeck is considered.

```
> # search for closest donors
> X.mtc <- c("Sepal.Length", "Sepal.Width", "Petal.Width")
> nnd <- NND.hotdeck(data.rec=rec, data.don=don,
+                          match.vars=X.mtc, don.class="Species",
+                          dist.fun="Manhattan")

Warning: The  Manhattan  distance is being used
All the categorical matching variables in rec and don
 data.frames, if present are recoded into dummies
```

At this point the function `create.fused` is used to fill in the target variable (`"Petal.Length"` in the example) in the recipient data set. Then the filled recipient and the donor one can be concatenated to obtain the origin dataset with imputed values.

```
> # fills rec
> imp.rec <- create.fused(data.rec=rec, data.don=don,
+                         mtc.ids=nnd$mtc.ids, z.vars="Petal.Length")
> imp.rec$imp.PL <- 1 # flag for imputed
> #
> # re-aggregate data sets
> don$imp.PL <- 0
> imp.iris <- rbind(imp.rec, don)
> summary(imp.iris)

  Sepal.Length    Sepal.Width     Petal.Width          Species  
 Min.   :4.300   Min.   :2.000   Min.   :0.100   setosa    :50  
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:0.300   versicolor:50  
 Median :5.800   Median :3.000   Median :1.300   virginica :50  
 Mean   :5.843   Mean   :3.057   Mean   :1.199                  
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:1.800                  
 Max.   :7.900   Max.   :4.400   Max.   :2.500                  
  Petal.Length       imp.PL      
 Min.   :1.100   Min.   :0.0000  
 1st Qu.:1.500   1st Qu.:0.0000  
 Median :4.250   Median :0.0000  
 Mean   :3.736   Mean   :0.3067  
 3rd Qu.:5.100   3rd Qu.:1.0000  
 Max.   :6.900   Max.   :1.0000  

> #summary stat of imputed and non imputed Petal.Length
> tapply(imp.iris$Petal.Length, imp.iris$imp.PL, summary)

$`0`
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    1.1     1.6     4.3     3.8     5.1     6.9 

$`1`
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.300   1.425   4.200   3.591   5.100   6.700 
```

4 Mixed methods

A SM mixed method consists of two steps: (1) a model is fitted and all its parameters are estimated, then (2) a nonparametric approach is used to create the synthetic data set. The model is more parsimonious while the nonparametric approach offers "protection" against model misspecification. The proposed mixed approaches for SM are based essentially on *predictive mean matching* imputation methods (see D'Orazio *et al.* 2006b, Section 2.5 and 3.6). The function `mixed.mtc` in **StatMatch** implements two similar mixed methods that deal with variables (X_M, Y, Z) following the the multivariate normal distribution. The main difference consists in the estimation of the parameters of the two regressions Y vs. X_M and Z vs. X_M. By default the parameters are estimated through maximum likelihood (argument `method="ML"`); in alternative a method proposed by Moriarity and Scheuren (2001, 2003) (argument `method="MS"`) is available. D'Orazio *et al.*

(2005) compared these methods in an extensive simulation study: in general ML tends to perform better, moreover it permits to avoid some incoherencies in the estimation of the parameters that are possible with the Moriarity and Scheuren approach.

At the end of the first step, after the estimation of the parameters of the two regression models, the data set A is filled in with the "intermediate" values $\tilde{z}_a = \hat{z}_a + e_a$ ($a = 1, \ldots, n_A$) obtained by adding a random residual term e_a to the predicted values $\hat{z}_a$. The same happens in B which is filled in with the values $\tilde{y}_b = \hat{y}_b + e_b$ ($b = 1, \ldots, n_B$). Finally, in the step (2) each record in A is filled in with the value of Z observed on the donor found in B according to a constrained distance hot deck; the Mahalanobis distance is computed by considering the intermediate and live values: couples $(y_a, \tilde{z}_a)$ in A and $(\tilde{y}_b, z_b)$ in B.

Such a two steps procedure offers various advantages: it offers protection against model misspecification and also reduces the risk of bias in the marginal distribution of the imputed variable because the distances are computed on intermediate and truly observed values of the target variables, Y and Z, instead of the matching variables X_M. In fact when computing the distances by considering all the matching variables, variables with low predictive power on the target variable may influence negatively the distances.

In the following example the `iris` data set is used just to show how `mixed.mtc` works.

```
> # uses iris data set
> iris.A <- iris[101:150, 1:3]
> iris.B <- iris[1:100, c(1:2,4)]
> X.mtc <- c("Sepal.Length","Sepal.Width") # matching variables
> # parameters estimated using ML
> mix.1 <- mixed.mtc(data.rec=iris.A, data.don=iris.B, match.vars=X.mtc,
+                    y.rec="Petal.Length", z.don="Petal.Width",
+                    method="ML", rho.yz=0,
+                    micro=TRUE, constr.alg="lpSolve")
> mix.1$mu #estimated means

Sepal.Length  Sepal.Width Petal.Length  Petal.Width
    5.843333     3.057333     4.996706     1.037109

> mix.1$cor #estimated cor. matrix

             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000  -0.1175698    0.9131794   0.8490516
Sepal.Width    -0.1175698   1.0000000   -0.0992586  -0.4415012
Petal.Length    0.9131794  -0.0992586    1.0000000   0.7725288
Petal.Width     0.8490516  -0.4415012    0.7725288   1.0000000

> head(mix.1$filled.rec) # A filled in with Z

    Sepal.Length Sepal.Width Petal.Length Petal.Width
101          6.3         3.3          6.0         1.0
```

```
102          5.8         2.7          5.1         1.3
103          7.1         3.0          5.9         1.5
104          6.3         2.9          5.6         1.0
105          6.5         3.0          5.8         1.5
106          7.6         3.0          6.6         1.6
```

```
> cor(mix.1$filled.rec)
```

```
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000   0.4572278    0.8642247   0.5049387
Sepal.Width     0.4572278   1.0000000    0.4010446   0.1100799
Petal.Length    0.8642247   0.4010446    1.0000000   0.4052548
Petal.Width     0.5049387   0.1100799    0.4052548   1.0000000
```

When using `mixed.mtc` the synthetic data set is provided in output as the component `filled.rec` of the list returned by calling it with the argument `micro=TRUE`. When `micro=FALSE` the function `mixed.mtc` returns just the estimates of the parameters (parametric macro approach).

The function `mixed.mtc` by default performs mixed SM under the CI assumption ($\rho_{YZ|X_M} = 0$ argument `rho.yz=0`). When some additional auxiliary information about the correlation between Y and Z is available (estimates from previous surveys or form external sources) then it can be exploited in SM by specifying a value ($\neq 0$) for the argument `rho.yz`; it represents guess for $\rho_{YZ|X_M}$ when using the ML estimation or a guess for ρ_{YZ} when estimating the parameters by using the Moriarity and Scheuren method.

```
> # parameters estimated using ML and rho_YZ|X=0.85
> mix.2 <- mixed.mtc(data.rec=iris.A, data.don=iris.B, match.vars=X.mtc,
+                    y.rec="Petal.Length", z.don="Petal.Width",
+                    method="ML", rho.yz=0.85,
+                    micro=TRUE, constr.alg="lpSolve")
> mix.2$cor
```

```
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000  -0.1175698    0.9131794   0.8490516
Sepal.Width    -0.1175698   1.0000000   -0.0992586  -0.4415012
Petal.Length    0.9131794  -0.0992586    1.0000000   0.9113867
Petal.Width     0.8490516  -0.4415012    0.9113867   1.0000000
```

```
> head(mix.2$filled.rec)
```

```
    Sepal.Length Sepal.Width Petal.Length Petal.Width
101          6.3         3.3          6.0         1.5
102          5.8         2.7          5.1         1.3
103          7.1         3.0          5.9         1.6
```

```
104           6.3          2.9           5.6          1.4
105           6.5          3.0           5.8          1.5
106           7.6          3.0           6.6          1.5
```

Special attention is required when specifying a guess for ρ_{YZ} under the Moriarity and Scheuren estimation approach (`method="MS"`); in particular it may happen that the specified value for ρ_{YZ} is not compatible with the given SM framework (remember that the correlation matrix must be positive semidefinite). If this is the case, then the `mixed.mtc` substitutes the input value `rho.yz` by its closest admissible value, as shown in the following example.

```
> mix.3 <- mixed.mtc(data.rec=iris.A, data.don=iris.B, match.vars=X.mtc,
+                    y.rec="Petal.Length", z.don="Petal.Width",
+                    method="MS", rho.yz=0.75,
+                    micro=TRUE, constr.alg="lpSolve")

input value for rho.yz is 0.75
low(rho.yz)= -0.1662
up(rho.yz)= 0.5565
Warning: value for rho.yz is not admissible: a new value is chosen for it
The new value for rho.yz is  0.5465

> mix.3$rho.yz

  start low.lim  up.lim    used
 0.7500 -0.1662  0.5565  0.5465
```

5 Statistical matching of data from complex sample surveys

The SM techniques presented in the previous Sections implicitly or explicitly assume that the observed values in A and B are i.i.d. Unfortunately, when dealing with samples selected from a finite population by means of complex sampling designs (with stratification, clustering, etc.) it is difficult to maintain the i.i.d. assumption (it would mean that the sampling design can be ignored) and the sampling design and the weights assigned to the units (usually design weights corrected for unit nonresponse, frame errors, etc.) have to be considered when making inferences (see Särndal *et al.*, 1992, Section 13.6).

When matching data from complex samples surveys, two different alternatives ways can be followed: *naive* approaches which essentially consist in using SM methods developed for i.i.d. samples or *ad hoc* approaches that explicitly take into account the sampling design and the corresponding sampling weights. In this latter case, three different SM methods are available: Renssen's approach based on weights' *calibrations* (Renssen, 1998); Rubin's *file concatenation* (Rubin, 1986) and the approach based on the *empirical likelihood* proposed by Wu (2004). A comparison among these approaches can be found in D'Orazio (2010).

5.1 Naive approaches

Most of these approaches consists in applying SM nonparametric micro methods (nearest neighbour distance, random or rank hotdeck) without considering the design nor the units weights. Once obtained the synthetic dataset (recipient filled in with the missing variables) the successive statistical analyses are carried out by considering the sampling design underlying the recipient data set and the corresponding survey weights. In the following a simple SM example based on nearest neighbor hotdeck is reported.

```
> # summary info on the weights
> sum(rec.A$wwA) # estimated pop size from A

[1] 67803

> sum(don.B$wwB) # estimated pop size from B

[1] 67803

> summary(rec.A$wwA)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  8.538  14.470  16.510  16.950  19.370  29.920

> summary(don.B$wwB)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  6.149  10.580  11.890  12.280  13.950  21.550

> # NND unconstrained hotdeck
> group.v <- c("rb090","db040")
> X.mtc <- c("hsize","age")
> out.nnd <- NND.hotdeck(data.rec=rec.A, data.don=don.B,
+                        match.vars=X.mtc, don.class=group.v,
+                        dist.fun="Manhattan")

Warning: The  Manhattan  distance is being used
All the categorical matching variables in rec and don
 data.frames, if present are recoded into dummies

> fA.nnd.m <- create.fused(data.rec=rec.A, data.don=don.B,
+                     mtc.ids=out.nnd$mtc.ids,
+                     z.vars=c("netIncome","c.netI"))
> # estimating average net income
> weighted.mean(fA.nnd.m$netIncome, fA.nnd.m$wwA) # imputed in A

[1] 15190.13
```

```
> weighted.mean(don.B$netIncome, don.B$wwB) # ref. estimate in B

[1] 15073.95

> # comparing marginal distribution of C.netI using weights
> tt.0w <- prop.table(xtabs(wwB~c.netI, data=don.B))
> tt.fw <- prop.table(xtabs(wwA~c.netI, data=fA.nnd.m))
> tt <- cbind(origin=c(tt.0w), nnd.naive=c(tt.fw))
> round(tt*100, 2)

         origin nnd.naive
(-6,0]    11.95     12.91
(0,5]      9.04      8.53
(5,10]    14.10     13.94
(10,15]   17.52     17.63
(15,20]   18.97     18.68
(20,25]   13.54     12.41
(25,30]    6.75      7.10
(30,40]    5.17      5.39
(40,50]    1.33      1.49
(50,200]   1.63      1.92

> # distance wrt to the origin distr.
> 1/2*colSums(abs(tt-tt[,1]))

    origin  nnd.naive
0.00000000 0.02085371
```

As far as imputation of missing values is concerned, a way of taking into account the sampling design can consist in forming the donation classes by using the design variables (stratification and/or clustering variables) jointly with the most relevant common variables (Andridge and Little, 2010). Unfortunately this operation can increase the complexity of SM or may be unfeasible because the design variables may not be available (or partly available). In particular, when the two sample surveys have quite different designs it is common that the design variables characterizing one survey are not available in the other one and vice versa.

When imputing missing values in a survey, another possibility consists in using sampling weights (design weights) to form the donation classes (Andridge and Little, 2010). But again, in SM applications the problem can be slightly more complex even because the sets of weights can be quite different from one survey to the other (usually the available weights are the design weights corrected to compensate for unit nonresponse, to satisfy some given constraints etc.). The same Authors (Andridge and Little, 2010) indicate that in the imputation framework, the selection of the donors can be carried out with probability proportional to weights associated to the donors (*weighted random* *hot deck*). Andridge and Little (2009) found that such a usage of the the sample weights can give poor results.

The properties of the weighted random hot deck in the SM applications have not been investigated, nevertheless the function `RANDwNDD.hotdeck` permits to select the donors with probability proportional to weights specified via the `weight.don` argument.

```
> rnd.2 <- RANDwNND.hotdeck(data.rec=rec.A, data.don=don.B,
+                           match.vars=NULL, don.class=group.v,
+                           weight.don="wwB")
> fA.wrnd <- create.fused(data.rec=rec.A, data.don=don.B,
+                         mtc.ids=rnd.2$mtc.ids,
+                         z.vars=c("netIncome","c.netI"))
> head(fA.wrnd)

    hsize hsize6      db040 age    c.age rb090 pb220a    rb050
401     5      5 Burgenland  45  (24,49]  male     AT 4.545916
71      2      2 Burgenland  65 (64,100]  male     AT 6.151409
92      2      2 Burgenland  81 (64,100]  male     AT 6.151409
225     3      3 Burgenland  51  (49,64]  male     AT 5.860364
364     4      4 Burgenland  18  [16,24]  male     AT 6.316554
370     5      5 Burgenland  50  (49,64]  male     AT 4.545916
    pl030        work      wwA netIncome  c.netI
401     1     working 10.85782  24069.74 (20,25]
71      5 not working 14.69250  12850.50 (10,15]
92      5 not working 14.69250  15430.68 (15,20]
225     1     working 13.99734  11346.32 (10,15]
364     1     working 15.08694      0.00  (-6,0]
370     1     working 10.85782  24069.74 (20,25]

> weighted.mean(fA.wrnd$netIncome, fA.wrnd$wwA) # imputed in A

[1] 15060

> weighted.mean(don.B$netIncome, don.B$wwB) # ref. estimate in B

[1] 15073.95

> # comparing marginal distribution of C.netI using weights
> tt.0w <- prop.table(xtabs(wwB~c.netI, data=don.B))
> tt.fw <- prop.table(xtabs(wwA~c.netI, data=fA.wrnd))
> tt <- cbind(origin=c(tt.0w), nnd.naive=c(tt.fw))
> round(tt*100, 2)

         origin nnd.naive
(-6,0]    11.95     12.73
(0,5]       9.04      8.18
(5,10]     14.10     14.20
(10,15]    17.52     18.32
(15,20]    18.97     19.20
(20,25]    13.54     12.33
(25,30]     6.75      6.95
(30,40]     5.17      4.60
(40,50]     1.33      1.76
(50,200]    1.63      1.74
```

```
> # distance wrt to the origin distr.
> 1/2*colSums(abs(tt-tt[,1]))

   origin nnd.naive
0.0000000 0.0264548
```

Perhaps, a better usage of the units weights is achieved in the rank hot deck SM where the the weights, w_i, of the units in A and in B can be used in estimating the percentage points of the the empirical cumulative distribution function by means of the expression:

$$\hat{F}(x_k) = \frac{\sum_{i=1}^{n} w_i I\left(x_i \leq x_k\right)}{\sum_{i=1}^{n} w_i}$$

Such a procedure can provide quite good results in terms of preservation of the marginal distribution of the imputed variables in the synthetic data set. In the following it is reported an very simple example.

```
> rnk.w <- rankNND.hotdeck(data.rec=rec.A, data.don=don.B,
+                          don.class="db040", var.rec="age",
+                          var.don="age", weight.rec="wwA",
+                           weight.don="wwB")
> #
> #create the synthetic data set
> fA.wrnk <- create.fused(data.rec=rec.A, data.don=don.B,
+                        mtc.ids=rnk.w$mtc.ids,
+                        z.vars=c("netIncome", "c.netI"),
+                        dup.x=TRUE, match.vars="age")
> #
> weighted.mean(fA.wrnk$netIncome, fA.wrnk$wwA) # imputed in A

[1] 15047.15

> weighted.mean(don.B$netIncome, don.B$wwB) # ref. estimate in B

[1] 15073.95
```

```
> # comparing marginal distribution of C.netI using weights
> tt.0w <- prop.table(xtabs(wwB~c.netI, data=don.B))
> tt.fw <- prop.table(xtabs(wwA~c.netI, data=fA.wrnk))
> tt <- cbind(origin=c(tt.0w), nnd.naive=c(tt.fw))
> round(tt*100, 2)

         origin nnd.naive
(-6,0]    11.95     12.40
(0,5]      9.04      9.23
(5,10]    14.10     13.21
(10,15]   17.52     16.93
(15,20]   18.97     20.33
(20,25]   13.54     13.49
(25,30]    6.75      6.40
(30,40]    5.17      4.87
(40,50]    1.33      1.29
(50,200]   1.63      1.85

> # distance wrt to the origin distr.
> 1/2*colSums(abs(tt-tt[,1]))

    origin  nnd.naive
0.00000000 0.02225742
```

5.2 Renssen's statistical matching through weights calibrations

This approach consists in a series of calibration steps of the survey weights of A and B in order to achieve consistency between estimates (mainly totals) computed separately from them. Calibration is a technique very common in sample surveys for deriving new survey weights, as close as possible to the starting ones, which fulfill a series of constraints concerning totals for a set of auxiliary variables (for further details on calibration see Särndal, 2005). The Renssen's approach works well when dealing with categorical variables or in a mixed case in which the number of continuous variables is very limited. In the following it will be assumed that all the variables (X_M, Y, Z) are categorical. The procedure and the functions developed in **StatMatch** permit to handle one or more continuous variables (better just one) in the subset of the matching variables X_M, while Y and Z are assumed to be categorical (when this is not the case it is necessary to categorize the continuous variables).

The first step in the Renssen's procedure consists in calibrating weights in A and in B such that the new weights when applied to the set of the matching variables X_M allow to reproduce some known (or estimated) population totals. In **StatMatch** the harmonization step can be performed by using `harmonize.x`. This function performs weights calibration (or post-stratification) by means of functions available in the R package **survey** (Lumley, 2011). When the population totals are already known then they have to be passed to `harmonize.x` via the argument `x.tot`; on the contrary, when they are unknown (`x.tot=NULL`) they are estimated by a weighted average of the totals estimated on the two surveys before the harmonization step:

$$\tilde{t}_{X_M} = \lambda \hat{t}^{(A)}_{X_M} + (1-\lambda)\, \hat{t}^{(B)}_{X_M}$$

being $\lambda = n_A/(n_A + n_B)$ (n_A and n_B are the sample sizes of A and B respectively) (Korn and Graubard, 1999, pp. 281–284).

The following example shows how to harmonize the joint distribution of the gender and classes of age with the data from the previous example, assuming that the joint distribution of age and gender is not known.

```
> tt.A <- xtabs(wwA~rb090+c.age, data=rec.A)
> tt.B <- xtabs(wwB~rb090+c.age, data=don.B)
> (prop.table(tt.A)-prop.table(tt.B))*100

        c.age
rb090        [16,24]    (24,49]    (49,64]   (64,100]
  male    0.3661141  1.0995148 -0.9456418 -0.6383618
  female  0.0891681 -0.4970682  1.0772175 -0.5509426

> library(survey) # loads survey
> # creates svydesign objects
> svy.rec.A <- svydesign(~1, weights=~wwA, data=rec.A)
> svy.don.B <- svydesign(~1, weights=~wwB, data=don.B)
> #
> # harmonizes wrt to joint distr. of gender vs. c.age
> out.hz <- harmonize.x(svy.A=svy.rec.A, svy.B=svy.don.B,
+                 form.x=~c.age:rb090-1)
> #
> summary(out.hz$weights.A) # new calibrated weights for A

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  8.647  14.390  16.570  16.950  19.030  31.470

> summary(out.hz$weights.B) # new calibrated weights for B

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  6.279  10.540  11.840  12.280  13.910  22.400

> tt.A <- xtabs(out.hz$weights.A~rb090+c.age, data=rec.A)
> tt.B <- xtabs(out.hz$weights.B~rb090+c.age, data=don.B)
> (prop.table(tt.A)-prop.table(tt.B))*100

        c.age
rb090            [16,24]        (24,49]        (49,64]       (64,100]
  male    0.000000e+00 -2.775558e-15 -1.387779e-15 -1.387779e-15
  female  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
```

After the harmonization, the second step in the Renssen's procedure consists in estimating the two way contingency table $Y \times Z$. In absence of auxiliary information it is estimated under the CI assumption by means of:

$$\hat{P}^{(CIA)}(Y, Z) = \hat{P}^{(A)}(Y|X_M)\,\hat{P}^{(B)}(Z|X_M)\,\hat{P}(X_M)$$

In practice, $\hat{P}^{(A)}(Y|X_M)$ is computed from A; $\hat{P}^{(B)}(Z|X_M)$ is computed from data in B while $P(X_M)$ can be estimated indifferently from A or B (the data set are harmonized with respect to the X_M distribution). In **StatMatch** an estimate of the table $Y \times Z$ under the CIA is provided by the function `comb.samples`.

```
> # estimating c.pl030 vs. c.netI under the CI assumption
> out <- comb.samples(svy.A=out.hz$cal.A, svy.B=out.hz$cal.B,
+              svy.C=NULL, y.lab="work", z.lab="c.netI",
+              form.x=~c.age:rb090-1)
> #
> addmargins(t(out$yz.CIA))  # table estimated under the CIA

             working not working        Sum
(-6,0]     4203.9273   3929.4698  8133.3971
(0,5]      3212.7539   2941.5722  6154.3261
(5,10]     4436.4472   5108.0075  9544.4547
(10,15]    5648.5383   6199.2373 11847.7756
(15,20]    7129.6193   5716.1572 12845.7765
(20,25]    5391.3879   3802.7509  9194.1388
(25,30]    2877.6585   1696.1470  4573.8055
(30,40]    2249.5066   1256.9719  3506.4786
(40,50]     555.7829    345.2169   900.9998
(50,200]    688.8992    412.9481  1101.8473
Sum       36394.5210  31408.4790 67803.0000
```

When some auxiliary information represented by third data source C, containing all the variables (X_M, Y, Z) or just (Y, Z), is available Renssen's approach permits to exploit it in estimating $Y \times Z$. Two alternative methods are available: (a) *incomplete two way stratification*; and (b) *synthetic two way stratification*. In practice, both the methods estimate $Y \times Z$ from C after some further calibration steps (for further details see Renssen, 1998). The function `comb.samples` implements both the methods. In practice, the synthetic two way stratification (argument `estimation="synthetic"`) can be applied only when C contains all the variables of interest (X_M, Y, Z); on the contrary, when the data source C observes just Y and Z, only the incomplete two way stratification method can be applied (argument `estimation="incomplete"`). In the following a simple example is reported based on the artificial EU-SILC data introduced in Section 2.1; here a small sample C ($n_C = 200$) with all the variables of interest (X_M, Y, Z) is artificially created.

```
> # generating artificial sample C
> set.seed(43210)
> obs.C <- sample(nrow(silc.16), 200, replace=F)
> #
> X.vars <- c("hsize","hsize6","db040","age","c.age",
+             "rb090","pb220a", "rb050")
> y.var <- c("pl030","work")
> z.var <- c("netIncome","c.netI")
> #
> aux.C <- silc.16[obs.C, c(X.vars, y.var, z.var)]
> aux.C$wwC <- aux.C$rb050/sum(aux.C$rb050)*round(sum(silc.16$rb050)) # rough w
> svy.aux.C <- svydesign(~1, weights=~wwC, data=aux.C)
> #
> # incomplete two-way estimation
> out.inc <- comb.samples(svy.A=out.hz$cal.A, svy.B=out.hz$cal.B,
+                 svy.C=svy.aux.C, y.lab="work", z.lab="c.netI",
+                 form.x=~c.age:rb090-1, estimation="incomplete")
> #
> addmargins(t(out.inc$yz.est))

           working not working        Sum
(-6,0]    318.3646   7815.0325  8133.3971
(0,5]    3155.6684   2998.6577  6154.3261
(5,10]   3960.8064   5583.6483  9544.4547
(10,15]  4736.0014   7111.7742 11847.7756
(15,20]  9302.3226   3543.4539 12845.7765
(20,25]  6318.9931   2875.1457  9194.1388
(25,30]  4011.6435    562.1620  4573.8055
(30,40]  2587.8739    918.6047  3506.4786
(40,50]   900.9998      0.0000   900.9998
(50,200] 1101.8473      0.0000  1101.8473
Sum     36394.5210  31408.4790 67803.0000

> new.wwC <- weights(out.inc$cal.C) #new cal. weights for C
> #
> # marginal distributions of work
> m.work.cA <- xtabs(out.hz$weights.A~work, data=rec.A)
> m.work.cC <- xtabs(new.wwC~work, data=aux.C)
> m.work.cA-m.work.cC

work
    working not working
          0           0

> #
> # marginal distributions of c.netI
> m.cnetI.cB <- xtabs(out.hz$weights.B~c.netI, data=don.B)
> m.cnetI.cC <- xtabs(new.wwC~c.netI, data=aux.C)
> m.cnetI.cB-m.cnetI.cC

c.netI
       (-6,0]         (0,5]        (5,10]       (10,15] 
 7.275958e-12 -9.094947e-13  0.000000e+00  0.000000e+00 
      (15,20]       (20,25]       (25,30]       (30,40] 
 0.000000e+00  0.000000e+00  9.094947e-13 -4.547474e-13 
      (40,50]      (50,200] 
 1.136868e-13  2.273737e-13 

> # joint distribution of the matching variables
> tt.A <- xtabs(out.hz$weights.A~rb090+c.age, data=rec.A)
> tt.B <- xtabs(out.hz$weights.B~rb090+c.age, data=don.B)
> tt.C <- xtabs(new.wwC~rb090+c.age, data=aux.C)
> (prop.table(tt.A)-prop.table(tt.B))*100

        c.age
rb090          [16,24]       (24,49]       (49,64]      (64,100]
  male    0.000000e+00 -2.775558e-15 -1.387779e-15 -1.387779e-15
  female  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00

> (prop.table(tt.A)-prop.table(tt.C))*100

        c.age
rb090        [16,24]    (24,49]    (49,64]   (64,100]
  male    -3.5559464 -0.6181224  1.4472383 -0.8255670
  female   0.4223229  2.1410662 -0.3271726  1.3161809

> #distance tt.A-tt.C
> 1/2*sum(abs(prop.table(tt.A)-prop.table(tt.C)))

[1] 0.05326808
```

The incomplete two way stratification method estimates the table $Y \times Z$ from C by preserving the marginal distribution of Y and of Z estimated respectively from A and from B after the initial harmonization step; on the contrary, the joint distribution of the matching variables X_M (which is the basis of the harmonization step) is not preserved.

```
> # synthetic two-way estimation
> out.synt <- comb.samples(svy.A=out.hz$cal.A, svy.B=out.hz$cal.B,
+                  svy.C=svy.aux.C, y.lab="work", z.lab="c.netI",
+                  form.x=~c.age:rb090-1, estimation="synthetic")
> #
> addmargins(t(out.synt$yz.est))
```

```
            working not working        Sum
(-6,0]      351.6488   7781.7483  8133.3971
(0,5]      3610.2537   2544.0724  6154.3261
(5,10]     4052.7261   5491.7286  9544.4547
(10,15]    5384.8795   6462.8961 11847.7756
(15,20]    8542.0337   4303.7428 12845.7765
(20,25]    5971.5562   3222.5826  9194.1388
(25,30]    3781.3214    792.4840  4573.8055
(30,40]    2697.2545    809.2241  3506.4786
(40,50]     900.9998      0.0000   900.9998
(50,200]   1101.8473      0.0000  1101.8473
Sum       36394.5210  31408.4790 67803.0000

> new.wwC <- weights(out.synt$cal.C) #new cal. weights for C
> #
> # marginal distributions of work
> m.work.cA <- xtabs(out.hz$weights.A~work, data=rec.A)
> m.work.cC <- xtabs(new.wwC~work, data=aux.C)
> m.work.cA-m.work.cC

work
     working  not working
5.093170e-11 1.818989e-11

> # marginal distributions of c.netI
> m.cnetI.cB <- xtabs(out.hz$weights.B~c.netI, data=don.B)
> m.cnetI.cC <- xtabs(new.wwC~c.netI, data=aux.C)
> m.cnetI.cB-m.cnetI.cC

c.netI
      (-6,0]        (0,5]       (5,10]      (10,15]      (15,20]
1.000444e-11 6.366463e-12 9.094947e-12 1.091394e-11 1.455192e-11
     (20,25]      (25,30]      (30,40]      (40,50]     (50,200]
1.091394e-11 4.547474e-12 3.183231e-12 1.477929e-12 1.136868e-12

> # joint distribution of the matching variables
> tt.A <- xtabs(out.hz$weights.A~rb090+c.age, data=rec.A)
> tt.B <- xtabs(out.hz$weights.B~rb090+c.age, data=don.B)
> tt.C <- xtabs(new.wwC~rb090+c.age, data=aux.C)
> (prop.table(tt.A)-prop.table(tt.B))*100

        c.age
rb090          [16,24]       (24,49]       (49,64]      (64,100]
  male    0.000000e+00 -2.775558e-15 -1.387779e-15 -1.387779e-15
  female  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
```

```
> (prop.table(tt.A)-prop.table(tt.C))*100

        c.age
rb090       [16,24]    (24,49]    (49,64]   (64,100]
  male   -3.5960278  0.4416849  1.8174918 -0.4384661
  female  0.4739651  0.0017933 -0.4987798  1.7983386

> #distance tt.A-tt.C
> 1/2*sum(abs(prop.table(tt.A)-prop.table(tt.C)))

[1] 0.04533274
```

6 Exploring uncertainty due to the statistical matching framework

When the objective of SM consists in estimating a parameter (macro approach) it is possible to tackle the problem in an alternative way consisting in the "exploration" of the uncertainty on the model chosen for (X_M, Y, Z), due to the lack of knowledge typical of the basic SM framework (no auxiliary information is available). This approach does not end with a unique estimate of the unknown parameter characterizing the joint p.d.f. for (X_M, Y, Z); on the contrary it identifies an interval of plausible values for it. When dealing with categorical variables, the estimation of the intervals of plausible values for the probabilities in the table $Y \times Z$ are provided by the Fréchet bounds:

$$\max\{0; P(Y=j)+P(Z=k)-1\} \leq P(Y=j, Z=k) \leq \min\{P(Y=j); P(Z=k)\}$$

for $j = 1, \ldots, J$ and $k = 1, \ldots, K$, being J and K the categories of Y and Z respectively.

If the X_M variables are introduced, by conditioning on them, it is possible to derive the following result (D'Orazio *et al.*, 2006a):

$$P_{j,k}^{(low)} \leq P(Y=j, Z=k) \leq P_{j,k}^{(up)}$$

with

$$\begin{aligned} P_{j,k}^{(low)} &= \sum_i P(X_M=i) \max\{0; P(Y=j|X_M=i)+P(Z=k|X_M=i)-1\} \\ P_{j,k}^{(up)} &= \sum_i P(X_M=i) \min\{P(Y=j|X_M=i); P(Z=k|X_M=i)\} \end{aligned}$$

for $j = 1, \ldots, J$ and $k = 1, \ldots, K$.

In the SM basic framework, the probabilities $P(Y=j|X_M=i)$ are estimated from A, the $P(Z=k|X_M=i)$ are estimated from B while the marginal distribution $P(X_M=i)$ can be estimated indifferently on A or on B, assuming that both the samples, being representative samples of the same population, provide not significantly different estimates of $P(X_M = i)$. If this is not the case, before computing the bounds it would be preferable to harmonize the distribution of the X_M variables in A and in B by using the function `harmonize.x`.

In **StatMatch** the Fréchet bounds for $P(Y = j, Z = k)$ ($j = 1, \ldots, J$ and $k = 1, \ldots, K$), conditioning or not on the X_M variables, are provided by `Frechet.bounds.cat`.

```
> #comparing joint distribution of the X_M variables in A and in B
> t.xA <- xtabs(wwA~c.age+rb090, data=rec.A)
> t.xB <- xtabs(wwB~c.age+rb090, data=don.B)
> prop.table(t.xA)-prop.table(t.xB)

          rb090
c.age              male       female
  [16,24]   0.003661141  0.000891681
  (24,49]   0.010995148 -0.004970682
  (49,64]  -0.009456418  0.010772175
  (64,100] -0.006383618 -0.005509426

> #
> #computing tables needed by Frechet.bounds.cat
> t.xy <- xtabs(wwA~c.age+rb090+work, data=rec.A)
> t.xz <- xtabs(wwB~c.age+rb090+c.netI, data=don.B)
> out.fb <- Frechet.bounds.cat(tab.x=t.xA, tab.xy=t.xy, tab.xz=t.xz,
+                               print.f="data.frame")
> out.fb

          work   c.netI low.u       low.cx         CIA
1      working   (-6,0]     0 0.0000000000 0.062451939
2  not working   (-6,0]     0 0.0130833912 0.058088772
3      working    (0,5]     0 0.0000000000 0.047854165
4  not working    (0,5]     0 0.0100349443 0.043450884
5      working   (5,10]     0 0.0000000000 0.065841680
6  not working   (5,10]     0 0.0325145796 0.074582323
7      working  (10,15]     0 0.0044505872 0.083877816
8  not working  (10,15]     0 0.0409858756 0.090285053
9      working  (15,20]     0 0.0315476801 0.106111106
10 not working  (15,20]     0 0.0330428837 0.083072522
11     working  (20,25]     0 0.0271769197 0.080524320
12 not working  (20,25]     0 0.0221981534 0.055298451
13     working  (25,30]     0 0.0035480015 0.042818593
14 not working  (25,30]     0 0.0058632580 0.024631748
15     working  (30,40]     0 0.0000000000 0.033456492
16 not working  (30,40]     0 0.0032094037 0.018254479
17     working  (40,50]     0 0.0000000000 0.008213067
18 not working  (40,50]     0 0.0001182705 0.005004024
19     working (50,200]     0 0.0000000000 0.010221237
20 not working (50,200]     0 0.0002598896 0.005961328
        up.cx       up.u
1  0.10745732 0.11953297
2  0.12054071 0.11953297
3  0.08127010 0.09037855
4  0.09130505 0.09037855
5  0.10790942 0.14101622
6  0.14042400 0.14101622
7  0.13317699 0.17515499
8  0.16971228 0.17515499
9  0.15614074 0.18965562
10 0.15763595 0.18965562
11 0.11362462 0.13543995
12 0.10864585 0.13543995
13 0.06158708 0.06746229
14 0.06390234 0.06746229
15 0.04850157 0.05171910
16 0.05171097 0.05171910
17 0.01309882 0.01334022
18 0.01321709 0.01334022
19 0.01592268 0.01630008
20 0.01618257 0.01630008
```

References

Alfons A., Kraft S. (2011) "simPopulation: Simulation of synthetic populations for surveys based on sample data". R package version 0.3. http://CRAN.R-project.org/package=simPopulation

Andridge R.R., Little R.J.A. (2009) "The Use of Sample Weights in Hot Deck Imputation". *Journal of Official Statistics*, **25**(1), 21–36.

Andridge R.R., Little R.J.A. (2010) "A Review of Hot Deck Imputation for Survey Nonresponse". *International Statistical Review*, **78**, 40–64.

Berkelaar M. and others (2011) "lpSolve: Interface to Lpsolve v. 5.5 to solve linear–integer programs". R package version 5.6.6. http://CRAN.R-project.org/package=lpSolve

Bertsekas D.P., Tseng P. (1994). "RELAX-IV: A Faster Version of the RELAX Code for Solving Minimum Cost Flow Problems". *Technical Report*, LIDS-P-2276, Massachusetts Institute of Technology, Cambridge. http://web.mit.edu/dimitrib/www/RELAX4_doc.pdf

Breiman, L. (2001) "Random Forests", *Machine Learning*, **45**(1), 5–32.

Breiman L., Friedman J. H., Olshen R. A., and Stone, C. J. (1984) *Classification and Regression Trees*. Wadsworth.

Cohen M. L. (1991) “Statistical matching and microsimulation models”, in Citro and Hanushek (eds) *Improving Information for Social Policy Decisions: The Uses of Microsimulation Modeling. Vol II Technical papers.* Washington D.C.

D’Orazio M. (2010) “Statistical matching when dealing with data from complex survey sampling”, in *Report of WP1. State of the art on statistical methodologies for data integration*, ESSnet project on Data Integration, 33–37, http://www.essnet-portal.eu/sites/default/files/131/ESSnetDI_WP1_v1.32.pdf

D’Orazio M. (2011) “StatMatch: Statistical Matching”. R package version 1.0.3. http://CRAN.R-project.org/package=StatMatch

D’Orazio M., Di Zio M., Scanu, M. (2005) “A comparison among different estimators of regression parameters on statistically matched files trough an extensive simulation study”. *Contributi Istat*, **2005/10**

D’Orazio M., Di Zio M., Scanu M. (2006a) “Statistical matching for categorical data: Displaying uncertainty and using logical constraints”. *Journal of Official Statistics* **22**, 137–157.

D’Orazio M., Di Zio M., Scanu M. (2006b) *Statistical matching: Theory and practice.* Wiley, Chichester.

D’Orazio M., Di Zio M., Scanu M. (2008) “The statistical matching workflow”, in: *Report of WP1: State of the art on statistical methodologies for integration of surveys and administrative data*, “ESSnet Statistical Methodology Project on Integration of Survey and Administrative Data”, 25–26. http://cenex-isad.istat.it/

D’Orazio M., Di Zio M., Scanu M. (2010) “Old and new approaches in statistical matching when samples are drawn with complex survey designs”. *Proceedings of the 45th “Riunione Scientifica della Societa’ Italiana di Statistica”*, Padova 16–18 June 2010.

Gower J. C. (1971) “A general coefficient of similarity and some of its properties”. *Biometrics*, **27**, 623–637.

Hothorn T., Hornik K., Zeileis A. (2006) “Unbiased Recursive Partitioning: A Conditional Inference Framework”. *Journal of Computational and Graphical Statistics*, **15**(3), 651–674.

Korn E.L., Graubard B.I. (1999) *Analysis of Health Surveys.* Wiley, New York

Kovar J.G., MacMillan J., Whitridge P. (1988) “Overview and strategy for the Generalized Edit and Imputation System”. Statistics Canada, Methodology Working Paper, No. BSMD 88-007 E/F.

Liaw A., Wiener M. (2002) “Classification and Regression by randomForest”. *R News*, **2**(3), 18–22.

Lumley T. (2011) “survey: analysis of complex survey samples”. R package version 3.24-1. http://CRAN.R-project.org/package=survey

Meyer D., Buchta C. (2011) “proxy: Distance and Similarity Measures”. R package version 0.4-7. http://CRAN.R-project.org/package=proxy

Moriarity C., Scheuren F. (2001) “Statistical matching: a paradigm for assessing the uncertainty in the procedure”. *Journal of Official Statistics*, **17**, 407–422.

Moriarity C., Scheuren F. (2003). “A note on Rubin’s statistical matching using file concatenation with adjusted weights and multiple imputation”, *Jour. of Business and Economic Statistics*, **21**, 65–73.

R Development Core Team (2011) *R: A language and environment for statistical computing. R Foundation for Statistical Computing*, Vienna, Austria. ISBN 3-900051-07-0, http://www.R-project.org/.

Rässler S. (2002) *Statistical matching: a frequentist theory, practical applications and alternative Bayesian approaches.* Springer Verlag, New York.

Renssen R.H.(1998) “Use of statistical matching techniques in calibration estimation”. *Survey Methodology* **2**4, 171–183.

Rubin D.B. (1986) “Statistical matching using file concatenation with adjusted weights and multiple imputations”. *Journal of Business and Economic Statistics*, **4**, 87–94.

Särndal C.E., Swensson B., Wretman J. (1992) *Model Assisted Survey Sampling.* Springer-Verlag, New York.

Särndal C.E., Lundström S. (2005) *Estimation in Surveys with Nonresponse.* Wiley, New York.

Scanu M. (2008) “The practical aspects to be considered for statistical matching”. in: *Report of WP2: Recommendations on the use of methodologies for the integration of surveys and administrative data*, “ESSnet Statistical Methodology Project on Integration of Survey and Administrative Data”, 34–35. http://cenex-isad.istat.it/

Singh A.C., Mantel H., Kinack M., Rowe G. (1993) “Statistical matching: use of auxiliary information as an alternative to the conditional independence assumption”. *Survey Methodology*, **19**, 59–79.

Wu C. (2004) “Combining information from multiple surveys through the empirical likelihood method”. *The Canadian Journal of Statistics*, **32**, 15–26.