# DACSEIS

IST-2000-26057

# Workpackage 11

# Imputation and Non-Response

## Deliverable 11.2

2004

**List of contributors**:

Seppo Laaksonen, Statistics Finland;
Ueli Oetliker, Swiss Federal Statistical Office;
Susanne Rässler, University of Erlangen-Nürnberg;
Jean-Pierre Renfer, Swiss Federal Statistical Office;
Chris Skinner, University of Southampton.

**Main responsibility**:

Seppo Laaksonen, Statistics Finland;
Susanne Rässler, University of Erlangen;
Chris Skinner, University of Southampton.

**IST–2000–26057–DACSEIS**

The DACSEIS research project is financially supported within the IST programme of the European Commission. Research activities take place in close collaboration with Eurostat.

`http://europa.eu.int/comm/eurostat/research/`

`http://www.cordis.lu/ist/`

`http://www.dacseis.de/`

# Preface

The ultimate objective of this document is to provide pseudo code for the implementation of imputation methods in the DACSEIS simulation studies. This code will be given in Appendices B and C. Before defining the code, we shall outline the rationale for imputation in Chapter 1. This document distinguishes between two broad approaches: *single imputation* and *multiple imputation*, according to whether one ore more imputed values are constructed for each missing value. These two approaches are described in Chapter 2 and 3 respectively. We shall consider some relatively simple applications, focussing on univariate rather than multivariate missing data and not considering hierarchical data structures. The variable with missing values to be imputed is categorical in four applications (the labour force surveys of Austria, Finland and the Netherlands, and the micro census of Germany) and continuous in the other two applications (Swiss Household budget survey and German Survey of income and expenditure) (see deliverable D1.1). Some more detailed consideration of imputation in the Swiss Household Budget Survey is given in Appendix A.

Seppo Laaksonen — Statistics Finland
Susanne Rässler — University of Erlangen
Chris Skinner — University of Southampton

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Outline of Chapter

In this chapter we describe the general non-response setting in which imputation might be considered and outline why imputation is used. We also explain the distinction between single imputation and multiple imputation which underlies the structure of this report.

We make some remarks on the mechanisms for generating non-response in the DACSEIS universe. We do not, however, specify these techniques. The reports of these specifications are available for each DACSEIS file, see the deliverables for WP1.

## 1.2 Non-Response in Surveys

Sample surveys are invariably subject to non-response.

*Unit non-response* arises when no survey data are collected for a unit, for example because no contact is established with the unit or because of the unit's refusal to participate in the survey.

*Item non-response* arises when some data are collected for a unit but values of some items are missing, for example because the respondent refuses to provide the answer to a sensitive question or is unable to provide the answer to a question requiring complex information.

Some patterns of non-response might be viewed as either unit or item non-response, for example wave non-response or attrition in a longitudinal survey or non-response by one person in a household survey where other members of the household respond. See Groves *et al.* (2002) for further discussion of the sources and reasons for non-response in surveys.

All these cases of non-response may be considered as examples of *missing data* in surveys. Data may also be missing for other reasons. For example, no data will be available for all units which are not present in the sampling frame (the problem of *under-coverage*). In some surveys or censuses, a short questionnaire is completed by one group of respondents while a longer questionnaire is completed by the rest, leading to deliberately missing data in the first group on the variables only featuring in the longer questionnaire.

## 1.3 The Treatment of Non-Response: Weighting and Imputation

Ideally, non-response should be prevented from occurring, but in practice some amount of non-response inevitably arises. It is then necessary to consider how to treat the non-response at the estimation stage of the survey. Estimation typically involves the construction of a point estimator and an associated variance estimator.

Different approaches to point estimation may be adopted to account for non-response. The simplest kinds of methods just ignore the non-response. In the case of unit non-response this will typically involve treating the set of responding units as if it were the selected sample. In the case of item non-response this may involve deleting units which have missing values on any of the variables used in a particular analysis (*available cases analysis*) or may involve deleting units which have missing values on any of the survey variables (*complete cases analysis*). Such approaches may be subject to bias and, in general, do not make most efficient use of the data. We shall not consider them explicitly, although they will typically feature as special cases of the approaches we do consider.

The two principal methods used to correct for bias due to non-response and to make efficient use of data are *weighting* and *imputation*, besides other model-based techniques such as the EM-Algorithm, may also be used for some kinds of analyses.

Weighting is classically used to treat the problem of unit non-response, whereas imputation is classically used to treat problems of item non-response. Weighting is a 'unit-level' adjustment, providing a common form of adjustment for all analyses based on a common set of responding units and is thus natural for the treatment of unit non-response. It is less practical to use weighting to treat item non-response, since a different method of weighting would be required for estimates based upon different sets of variables. Weighting is particularly natural for complex survey data involving complex sampling designs and/or the presence of auxiliary population information. In such settings, weighted estimation is used even in the absence of non-response to adjust for unequal selection probabilities and to make efficient use of auxiliary population information. The additional use of weighting to treat non-response in such settings is sometimes called re-weighting, since it may involve the modification of an initial set of weights.

In contrast to weighting, imputation is a variable-specific adjustment and is thus natural to treat missing data in a given variable. Each missing value is replaced by an imputed (fabricated) value. Imputation tends to become more complicated and time consuming as the number of variables increases and so is more difficult to implement for the treatment of unit non-response in surveys with many variables.

## 1.4 Non-response Mechanisms

The process generating the non-response is called the *non-response mechanism*. This is a special case of a *missing data mechanism*, the process that leads to missing data. In the simulation studies, it will be necessary to specify non-response mechanisms and we discuss this in the next section. The nature of the non-response mechanism is important

for understanding the impact of non-response on estimation and therefore for determining the way that non-response will be treated in estimation. It is important to emphasize that the non-response mechanism is generally unknown to the user of the data. Weighting or imputation methods will typically be based upon some assumptions or model, explicit or implicit, about the non-response mechanism, but these assumptions or model will never be more than an approximation to the truth.

For the purpose of constructing weighting and imputation procedures and for evaluating bias and variance, it is useful to classify missing data mechanisms in a number of different ways. The following 'definitions' are heuristic. See Little and Rubin (2002) for more precise definitions.

*Missing Completely At Random (MCAR)*: The values of the set of variables used to construct a point estimator are missing completely at random if missingess is independent of all these variables.

Under the MCAR condition, non-response generally does not introduce bias into the point estimator.

*Missing at Random (MAR):* The values of the set of variables used to construct a point estimator are missing at random given an additional set of measured variables if missingess is independent of the values of the variables which are missing, conditional on the observed values of both sets of variables.

More simply, we may say that missingness is at random if it does not depend upon the underlying values which are missing, conditional on information used for estimation.

A mechanism which is not MAR is called *Not Missing At Random (NMAR)*. In this case, missingness of a variable will generally depend in some way on the value of the variable which is missing, even conditional on observable information, for example, non-response may be more likely to occur on income if income is high.

Under the MAR condition, it is generally feasible, in principle, to construct an adjusted point estimator which is approximately unbiased in large samples, providing the relation between the variables is correctly modelled.

*Ignorable missingness*: The missing data mechanism is ignorable if the properties of a given inference procedure (e.g. point estimation) are not affected by the nature of the mechanism.

In practice, the MAR condition often implies ignorable missingness for many kinds of inference procedures. The advantage of being able to assume ignorable missingness is that it is not necessary to specify a statistical model for the missing data mechanism when determining a weighting or imputation procedure.

*Non-ignorable missingness*: The missing data mechanism is non-ignorable if it is not ignorable.

In practice, the NMAR condition often implies non-ignorable missingness. It is generally more difficult to construct weighting and imputation procedures to give approximately unbiased estimates when non-response is non-ignorable.

## 1.5 Non-response Mechanisms for Simulation Study

In this section, we comment on the non-response mechanisms which could be employed in the simulation studies.

*Coverage:* It would be realistic to include some undercoverage and overcoverage in the simulation. Sources of undercoverage and overcoverage tend to be quite different. For example, in a typical household survey, overcoverage is common for emigrants, whereas undercoverage is common for infants and immigrants. The creation of overcoverage and undercoverage in the simulation could be straightforward if such characteristics are available in the universe file. However, the treatment of coverage problems is not the primary purpose of imputation and it therefore seems sensible not to create such problems, at least in the first simulation studies.

*Non-response*: It is easiest to start with the simple assumption that non-response is MCAR for each stratum but with a different level of non-response in each. This assumption may then be made progressively more complex by, for example:

- creating missing values with varying probabilities (response propensities) for each unit of the universe using a logistic regression model which may be estimated from a sample survey with similar structure to the universe. These are thus general response propensities, and are not dependent on the survey variables, corresponding to the *MAR* assumption. It is possible to add some random noise to modelled values.
- assigning an additional propensity coefficient to the item level (this is thus conditional on the fact that the unit responds): we suppose that for most variables this coefficient = 1, but for some more problematic variables less. This may be done purely using random selection (MCAR, MAR) or modelled exploiting logistic techniques also in this case.
- There may be different levels of missingness propensities (high, low) for testing that effect.

## 1.6 Reasons for Imputation

Imputation is used for a number of reasons in official statistics, including the following.

*Complete Datasets*: After missing values are replaced by imputed values, the original dataset with 'holes' becomes a complete dataset. This has various advantages. All analyses of this dataset will be mutually consistent. For example, if two tables are produced, cross classifying variable A by variable B and variable C by variable B, then the B margins of the two tables will be identical if both tables are based upon the same completed dataset but this might not be the case if the tables had been based upon data with different sets of missing values for the three variables. A completed dataset also avoids various problems of datasets with missing values, e.g. that different users might deal with these missing values in different ways (leading to inconsistent analyses) or may treat the missing data erroneously, e.g. treating a missing value code as real.

*Edit and Imputation*: A second reason for using imputation is to handle the result of editing, where e.g. invalid responses are identified, and it is desirable to replace them by valid values.

*Reduction of Item Nonresponse Bias*: A third reason for imputation is to reduce bias arising from item non-response. If the values of auxiliary variables $x$ are available for cases with missing values of $y$ and if $x$ and $y$ are correlated then it will often be possible to reduce the bias by imputing the missing values of $y$ using the observed $x$ values.

## 1.7 Single Imputation or Multiple Imputation

The traditional approach to imputation in official statistics is to produce just one imputed value for each missing item. This is called *single imputation*. This can achieve the various objectives described in the previous section. Imputation can, however, create a problem for variance estimation.

The standard approach to point estimation under imputation is to treat the imputed values in the completed dataset as if they were actual values. Imputation methods are generally designed so that this approach will lead to a less biased estimator than would arise if cases with missing values were simply deleted. There is a problem, however, if the same principle is applied to variance estimation, i.e. if standard errors are estimated from standard software using the completed dataset with the imputed values treated as real. In this case the estimated variances will generally be too small, since the variance estimation method will fail to allow for differences between the imputed values and the real values.

A number of alternative approaches to variance estimation in the presence of imputation are possible. Some approaches treat the single imputed values as given, on the assumption that imputation has been designed for the purposes above, such as minimising non-response bias. These approaches then construct valid variance estimators for the resulting point estimators. See e.g. Rao and Shao (1992), Shao (2002) and Lee *et al.* (2002). An alternative approach is to design the imputation method in such a way that a simple variance estimator can be constructed. One such approach is *multiple imputation* (Rubin, 1987).

The basic idea of *multiple imputation* is to create $m$ imputed values for each missing item. For any parameter $\theta$, analysis of the $m$ completed datasets (and treatment of the imputed values as genuine) will lead to $m$ point estimates $\widehat{\theta}_1, \ldots, \widehat{\theta}_m$ of $\theta$ as well as $m$ variance estimates $\widehat{v}_1, \ldots, \widehat{v}_m$. Rubin proposes to combine the point estimates by taking their mean and proposes a variance estimator for this point estimator which is a simple function of the $m$ point estimates and variance estimates. It is clear that this approach will only work for certain kinds of imputation methods, in particular the imputation method must be stochastic since otherwise identical datasets and estimates will be produced. However, it is quite easy to modify even non-stochastic single imputation methods such that suitable multiple imputations can be created, e.g. by using the approximate Bayesian bootstrap according to Rubin and Schenker (1986). The basic kind of imputation method required for multiple imputation is what Rubin calls a *proper* method, which basically means that inference from the multiply imputed data sets is randomisation-valid.

In this document, we shall present single imputation methods first in Chapter 2. These imputation methods cover ones traditionally used in official statistics, with the main objective being to reduce bias from non-response in the resulting point estimates. Multiple imputation methods are then considered in Chapter 3. These methods may be seen to be extensions of some of the basic regression imputation methods considered in Chapter 2.

The relative advantages of single and multiple imputation are the subject of some debate. See, for example, RUBIN (1996), MENG (1994) and NIELSEN (2003).

# Chapter 2

# Single Imputation

## 2.1 Introduction

This chapter describes some single imputation methods, that is methods that produce a single imputed value for each missing item. We shall focus on the univariate case, where missing values for a single variable, $y$, are to be imputed. We begin by outlining the structure of the imputation process and then proceed to describe some imputation methods. This field is so wide that it is possible to outline only a number of standard techniques. We take the broad objective of the imputation process to be the reduction of non-response bias in the point estimator which results from treating the imputed values as real values. As noted in Chapter 1, we do not consider here the issue of variance estimation.

## 2.2 Outline of the Imputation Process

In this section we outline the series of steps involved in the process of imputation. We assume that editing has already taken place and that the missing values have been identified.

**Step (i) Selection of training data set and specification of auxiliary variables**

A *training data set* needs to be selected upon which the imputation method can be based. This will often be a subset of respondents for the same data set with missing values requiring imputation, but it may alternatively be an external analogous data set from the previous period, for example. This training data set should include not only values of $y$ but also values of a vector of *auxiliary variables*, $x$. The $x$ variables should be observed for cases where $y$ is to be imputed (Laaksonen, 2002b). For most imputation methods, it is desirable that the auxiliary variables be chosen so that they predict $y$ as well as possible and so that the MAR assumption is plausible, that is that missingness on $y$ is approximately independent of $y$ conditional on the values of $x$.

**Step (ii) Construction of imputation model**

Most imputation methods are motivated explicitly or implicitly by a model, referred to here as the *imputation model*. Two alternative target variables may be used for building an imputation model, either $y$, the variable being imputed, or the missingness indicator of this variable, denoted $R$. The model for each particular case may be of any type, i.e. parametric or non-parametric. Moreover, the model may be estimated from the training data, or 'logically deduced' The purpose of the modelling is to achieve high predictability. Just one model can be estimated for the data set that requires imputation, or alternatively, several models can be estimated, one for each sub-set. These sub-sets are often referred to as *imputation cells* or *classes*. They should be as homogeneous as possible with respect to missingness, in other words, the missingness mechanism should be presumed to be ignorable within such a cell.

**Step (iii) Choice of two features of imputation method**

There are two particular features of the imputation method which need to be chosen, depending on the final imputation method being applied. First, there is a need to decide on the prediction role of imputation, whether it is appropriate to use a deterministic imputation method which provides 'best predictions' of the missing true values or whether it is appropriate to use a stochastic method, where the distribution of the possible imputed values corresponds to the uncertainty about the missing values. Second, it is often desirable to decide on a metric for measuring nearness if the imputation method requires this kind of option. Typically, such nearness metrics are based on a Euclidean distance measure or other model-external solutions, often using auxiliary variables that are not used in building a model. Alternatively, the metrics can be taken from model outputs. An example of this approach is the so-called *'regression based nearest neighbour' (RBNN)* technique (LAAKSONEN, 2000, and LAAKSONEN, 2002b) in which nearness is measured using predicted values of the imputation model.

**Step (iv) Choice of imputation method itself.**

Finally the imputation method needs to be selected, based upon the outcomes of steps (i)-(iii). We distinguish two broad kinds of imputation method. If the imputed values are derived from a model, either as predicted values or based upon an estimated distribution, we refer to the method as *model-based*. Alternatively, if the imputed value consists of the actual value of $y$ for a responding unit, possibly selected using an imputation model and a distance metric, we refer to the method as a *'donor'* method (NB. some authors use the term *'hot deck'* to denote this general case; we use the term 'hot deck' as a special case of a donor method). Note that this technique may be used for finding a good observed residual (noise term), too (see LAAKSONEN, 2002a).

It should be noted that it is desirable in the imputation process that the imputed values are *flagged*. When flagged so that several alternative imputed values can be available for one missing value even in the case of a single imputation, the user has the flexibility to apply in an analysis such imputed values as are considered the best for the purpose concerned.

The general structure of imputations is given in Figure 2.1, broken down into stages 0, A, B and C. We shall refer to this figure when giving rules for implementing a computer program for imputations in the case of the DACSEIS simulation task.

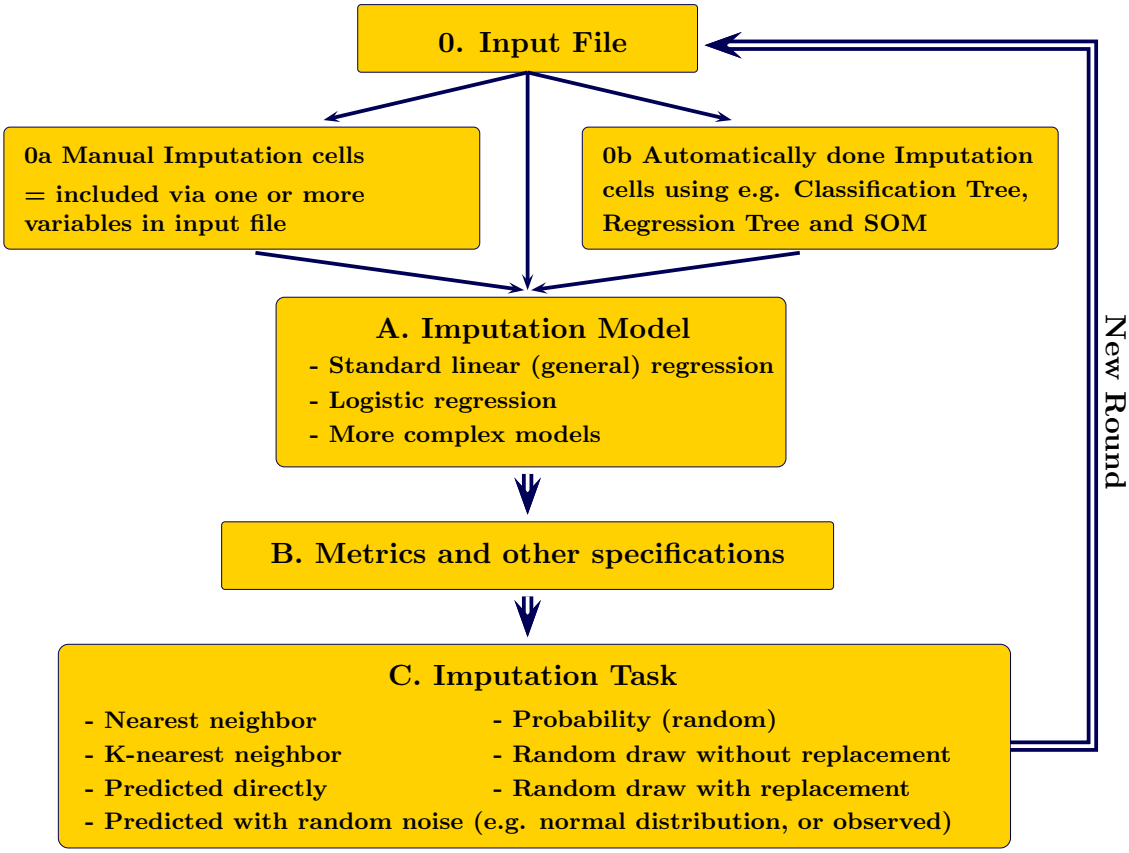


Figure 2.1: Structure of imputations

The *input file* in stage 0 in the figure is a standard file with missing item values which are to be imputed. There should be a standardised symbol for a missing value such as a point '.'. Alternatively, a user should be able to determine in the beginning of the process what values should be considered as missing values (e.g '-9'), and be imputed, consequently.

The specification of imputation cells in stages 0a or 0b of the figure and the specification of imputation model in stage A correspond to step (ii) of the imputation process above. The specification of imputation metrics in stage B corresponds to step (iii). Finally, the imputation tasks in C correspond to step (iv).

## 2.3 Univariate or Multivariate Missingness Patterns

The input file will in general include some variables with missing values and some variables without missing values. With $k$ variables with missing values, there are $2^k - 1$ possible missingness patterns. These may be studied as follows.

1. create a binary variable for each of the $k$ variables so that 1 = non-missing, and 0 = missing;
2. construct a multidimensional frequency table of these binary variables from the input file;
3. the output file of the previous table is the missingness pattern;
4. each pattern may then be given a code.

As noted earlier, we shall focus on the simplest case when there is just one variable, $y$, that may be missing ($k = 1$) so that there is just one missingness pattern, i.e. item non-repondents for which $y$ is missing. With multivariate missingness ($k > 1$), there will in general be more than one missingness pattern. Imputation then becomes more complex. One option is to impute separately for each missingness pattern drawing on auxiliary information from complete records. Another option is to chain the imputation tasks, imputing for one variable at a time.

*DACSEIS specification:* Provide the missingness patterns and code these. It seems that in most cases there will be just one pattern but there may be two for the Swiss data.

## 2.4 Imputation Model

As noted in Section 2.2, the target variable in the imputation model may either be $y$, the variable with missing values (assumed again to be univariate), or be $R$, the missing value indicator for the $y$ variable, i.e. $R = 1$ if $y$ is missing and $R = 0$ if it is observed.

Let $x_1, x_2, \ldots$ denote additional continuous variables in the input file which are completely observed for all sample units and $z_1, z_2, \ldots$ additional categorical variables which are also complete. Standard imputation models, as in cases 1 and 2 below, consist of a regression model, representing the conditional distribution of $y$ given $x_1, x_2, \ldots, z_1, z_2, \ldots$.

Many imputation methods depend upon a set of *imputation cells*, formed using the values of $x_1, x_2, \ldots, z_1, z_2, \ldots$. The simplest way to form imputation cells is by cross-classifying some of the $z_1, z_2, \ldots$ variables. The imputation model may be specified separately within imputation cells or may incorporate imputation cells via covariates.

**Case 1: Linear regression model for continuous $y$ variable**

In this case, the variable $y$ is continuous or handled as a continuous variable. The $x$ and $z$ variables may have been imputed at an earlier step. A standard model for $y$ given $x_1, x_2, \ldots$ is the linear regression model

$$y = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + \epsilon$$

More generally the variables $y$ and $x_1, x_2$ may be transformed first, for example the logarithm transformation might be applied to variables such as income, wages and turnover. Dummy explanatory variables may also be introduced to represent the effect of the $z_1, z_2, \ldots$ variables. The estimation of such a linear model may follow standard techniques

available in standard software. See the SAS specifications in Appendix B. Estimation may take place using sampling weights. Thus, this should be included as an option in the computer codes.

*Example of DACSEIS specifications:*

**The Swiss expenditure data (analogous to German data):**

1. For expenditure:
   - two options for $y$: (a) $y$ = expenditure or (b) $y$ = log(expenditure+1[1])
   - two options for $x, z$: (a) one key variable chosen by the Swiss group, (b) all available auxiliary variables.
2. For income:
   - two options for $y$: (a) y = income or (b) $y$ = log(income+1[1])
   - two options for $x, z$: (a) one key variable chosen by the Swiss group (e.g. completed income), (b) all available auxiliary variables plus completed income.

### Case 2. Logistic regression model for binary $y$

In this case, $y$ is binary and the model is the usual logistic regression (or probit regression) model with $x_1, x_2, \ldots, z_1, z_2, \ldots$ defining the covariates.

*Example of DACSEIS specifications:*

**LFS and similar data (German, Austria, Netherlands, Finland):**

- one option for binary $y$ so that $y = 1$ if unemployed and $y = 0$ otherwise
- two options for $x, z$: (a) one key variable chosen by each national group (e.g. region), (b) all available auxiliary variables without interactions.

### Case 3. Logistic Regression Model for Response Indicator R

Here the variable $R$ is the missing value indicator for the $y$ variable.

*Example of DACSEIS specifications:*

- one option for $R$ (= 0 if non-missing, = 1 if missing)
- two options for $x, z$ : (a) one key variable chosen by each national group (e.g. region), (b) all available auxiliary variables without interactions.

[1]Such a number which is correct for the data, if 1 is not good, then use 10, 100 or 1000.

## 2.5 Distance Metrics

Many imputation methods require the specification of a distance function, measuring how close two units are with respect to the auxiliary variables $x_1, x_2, \ldots, z_1, z_2, \ldots$. Distance metrics include:

1. **Distance metrics not based on models:**
   - Euclidean distance based on a continuous variable;
   - Euclidean distance based on several variables, each subject to user-specified scaling;
   - a metric for a categorical variable, e.g. geographical area, defined as 0 if two units share the same value of the variable and 1 if not.
   - a metric for categorical variables, defined as the number of variables taking different values for the two units.
2. **Distance metrics based on models:**
   - Euclidean distance between the predicted values of the $y$ variable based upon a regression model for $y$ given $x_1, x_2, \ldots, z_1, z_2, \ldots$.

*DACSEIS specifications:*

Two alternatives which are easy to implement, both using LFS types of data (not for expenditures or incomes):

- based on the predicted values of the model
- all units within a certain area are as close to each other.

## 2.6 Imputation Methods

An imputation method replaces a missing or deficient value with a fabricated one, the *imputed value*. We consider two broad types:

**A. Donor methods** identify, for each missing value, a *donor* which is another unit in the same database for which the value of this variable is present, and use this value as the imputed value. The unit for which the value is imputed is called the *recipient*.

**B. Model-based methods** fit the imputation model in some way using the training data and determine the imputed value using this fitted model. Note that some donor methods also involve fitting a model.

## A. Donor Methods

**A1. Random draw methods** (often called random hot deck methods)

Two options for drawing:

**A1a:** Random draw with replacement: A donor is randomly chosen from a given set with non-missing values. Thus the same donor may be chosen many times.

**A1b:** Random draw without replacement: as above but when a real donor is chosen, it cannot be chosen again. Note: if the missingness rate is higher than 50%, all missing values cannot be imputed.

Two options for the given set:

- 1u: Overall imputation, thus donors are drawn from all cases with non-missing values.
- 1s: Cell-based imputation, donors are drawn from the same imputation cell as the recipient. Selection from each cell is independent of selection from other cells. This may be interpreted as using a distance metric defined by the imputation cell.

Thus, we have the four possible methods:

| | |
|---|---|
| 1a + 1u = 1au | 1a + 1s = 1as |
| 1b + 1u = 1bu | 1b + 1s = 1bs |

Method 1*au* could be recommended for use for benchmarking purposes. Thus, the results from different methods could be compared with these results. Note that the construction of cells in 1*s* may involve model-fitting. For example, consider a binary variable $y$, which may be predicted directly from a logistic regression model or the model may be used to create imputation cells. In this case, the model is estimated using the respondent data and predicted values of the probability that $y = 1$ are determined for both nonrespondents and respondents. There are several ways that these scores could be grouped to create imputation cells. The two main approaches are:

1. division into equal intervals, e.g. (0%, 10%), [10%, 20%), [20%, 30%), ..., [90%, 100%)
2. division into intervals with equal frequencies, respectively (like deciles).

If the data set is small, very many intervals cannot be used. Problems may arise if some intervals or imputation cells include too many missing values for y. Both approaches are relatively easy to program. See the SAS specifications in Appendix B.

### A2. Nearest Neighbor methods

Possible options for choosing a donor for a given distance metric are:

**A2a**: The donor is selected to be the nearest to the recipient with respect to the distance metric. This requires calculating all the distances for all potential donors for each recipient. If the metric is based on a single variable, such as the predicted value, then the values of this variable should be sorted first, and then the nearest donors from above and then from below the value of the recipient unit are compared (the metric distances calculated), and the nearest is chosen.

For SAS, this can be done e.g. so that values for a reasonable number of lags have been constructed for both directions (below and above). This solution works both for continuous and categorical variables, thus for linear models and logistic models. See the SAS specification in Appendix B.

If there are several units with the same distance, two main options are available: (i) a random selection of all these, or (ii) the average value of all (this is possible only for continuous variables).

**A2b**: To choose randomly one donor of the $m$ nearest ones, where the user specifies the value of $m$. This reduces to option **A2a** for the case $m$=1.

## B. Model-based Methods

### B1. Simple Methods based upon Imputation Cells

**a. Mean imputation:** the mean of observed values of $y$ within each cell is calculated and this value is assigned to all missing units within that cell. It is possible to use robust methods to estimate the mean, e.g. by trimming outliers first.

**b. Median imputation:** as in **a.**, but using the median rather than the mean.

**c. Mode imputation:** for categorical variables, the most common value may be used as an imputed value.

**d. Average of the specified percentile** as in **a.**, but calculated for cases falling between two percentiles, e.g. p25-p75. Note that all of these may be weighted or unweighted (i.e. weights = 1).

### B2. Probability-based methods for Categorical Variables

The following method is for categorical or categorised variables, and may be done as earlier within cells.

This method calculates the proportions of observed values falling into each category. These are sorted cumulatively within the interval [0, 1] so that each category has its own interval indicating its probability. For example, categories $a$, $b$ and $c$ with relative frequencies 0.2, 0.7 and 0.1 may be assigned to intervals (0, 0.2], (0.2, 0.9] and (0.9, 1.0) respectively. Each imputed value is determined by taking a random number with a uniform distribution from

the same interval (0, 1). If the random number lies within the category (0, 0.2], then the imputed value = $a$, and respectively for the others.

### B3. Linear Regression Imputation

The method of regression imputation generates imputed values by fitting a linear regression model

$$y = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + \epsilon$$

to the responding units and then setting imputing values as $\widehat{y} = \widehat{\beta}_1 x_1 + \widehat{\beta}_2 x_2 + \ldots + \widehat{\beta}_p x_p$, where $(\widehat{\beta}_1, \widehat{\beta}_2, \ldots, \widehat{\beta}_p)$ is the vector of least squares estimates of the regression coefficients.

A special case is *ratio imputation* in which there is a single variable $x$ and no intercept is included in the regression model so that $\widehat{y} = \widehat{\beta} x$ and where $\widehat{\beta}$ is typically determined by the ratio of the mean of $y$ by the mean of $x$ amongst responding cases, perhaps within imputation cells.

### B4. Linear Regression Imputation with added noise

This method is the same as **B3** except that a noise term is added, so the imputed value is given by $\widehat{y} = \widehat{\beta}_1 x_1 + \widehat{\beta}_2 x_2 + \ldots + \widehat{\beta}_p x_p + \widehat{\epsilon}$, where $\widehat{\epsilon}$ may be determined in different ways. It is however good to include an option in order to avoid very high and, respectively, very low noise terms because these are not often realistic in practice (see the discussion on bounds before). It is not clear how to do that; one standard way is to use a truncated distribution (e.g. normal) so that a user could choose that truncation level, e.g. using the number of standard errors (e.g. one standard error, or 95% confidence interval).

It is also possible to determine $\widehat{\epsilon}$ from the data, i.e., from *observed residuals* for non-missing units. It is simplest to randomly choose each noise term from these residuals (similarly to methods A1). This methodology is not a full model-based method, since the last operation has been done using donor methodology. This thus shows that an imputation method may be also a mixture of donor-based and model-based methods. There may be many other situations where this may be competitive, too. We give some examples next.

### B5. Regression Imputation for Categorical Variables

Linear regression is inappropriate if $y$ is categorical. In this case, a categorical outcome regression model may be fitted to the respondent data, for example logistic regression when $y$ is binary. The model will imply a certain set of probabilities, $P_k$, that $y$ will fall into the different categories $k$ and $\widehat{P}_k$, the estimated values of the $P_k$, may then be determined for each case with a missing value of $y$ and these values used to determine imputed values, as in method **B2** above.

### B6. Two-step methods for Mixed Binary-Continuous Variables

An imputation method may involve several consecutive steps. A common case is the following two-step one which may be implemented fairly easily. The variable $y$ is mixed binary-continuous, for example earnings where a proportion of the population have no earnings and the remainder of the population have positive earnings. The binary variable is denoted $z$, i.e. has a job or not in our example, so that $y = 0$ if $z = 0$ and $y > 0$ if $z = 1$. Now we go to the two steps:

**Step 1** Construct a logistic regression model for $z$ with the same explanatory variables as before. Then for each unit with $y$ missing, use the B5 method to impute $z$.

**Step 2** Impute $y$ using one of the methods B3 or B4 for cases with $z = 1$. Impute $y = 0$ if $z = 0$.

*DACSEIS specifications:*

**Swiss data (and possibly the German EVS):**

We recommend linear regression imputation (B3) for this case. Since there are two model specifications and two different variable selections, we will have the following imputation specifications if possible; however, for practical reasons we cannot perform the first three specifications for the Swiss data:

(i). linear regression imputation for $y$ = expenditure with one auxiliary variable chosen by a country group,

(ii). linear regression imputation for $y$ = expenditure with all available auxiliary variables,

(iii). linear regression imputation for $y$ = log-transformed expenditure with all available auxiliary variables. Note that after imputation, the exponent transformation must be performed,

(iv). linear regression imputation for $y$ = income with all available auxiliary variables including (imputed) expenditure (log-transformed) based on specification (ii) (other specifications may be used). No intercept term. The model should be constructed both directly and including the robustness specifications as proposed by the Swiss group (see Appendix A).

(v). If a good imputation cell structure is available, we may also test ratio imputation within cells for *income* using variable *expenditure* as auxiliary variable. This specification could be as follows (within each cell):

- calculate ratio: sum(income)/sum(expenditure) = $q_c$ using respondents' data
- for each missing income calculate income(imputed) = $q_c$* expenditure(known)
- Later, if time permits, we may specify a respective method with noise term (bounds are needed if theoretical residuals are used; for empirical residuals we propose to exploit nearest neighbour methods).

**LFS type of data**

We propose the following 4 methods:

(i). Use the variable *region* (or another subgroup with a reasonable number of categories) as the imputation cell, and draw a random donor without replacement method (A1b).

(ii). Build a logistic imputation model for the target variable so that this is $y = 1$ if unemployed, and $y = 0$ otherwise (see model section above) and transfer all available auxiliary variables to this model as explanatory variables. And choose a donor using predicted values of this model based on a nearest neighbour technique, as in method A2 above (incl. SAS codes).

(iii). Build a logistic imputation model for the missingness indicator,$R$, for this binary variable $y$ and choose a donor using predicted values of this model based on a nearest neighbour technique, as in method A2 above (incl. SAS codes).

(iv). Build a logistic imputation model for $R$, as in (iii)., and choose a donor at random without replacement within 10 imputation cells so that these will have been constructed as explained above in Case 2 of the imputation model section.

Note that the option of using weights (sampling weights) should be allowed in the modelling task.

# Chapter 3

# Multiple Imputation

## 3.1 Introduction

The theory and principles of multiple imputation (MI) are extensively described by Rubin (1987), although this book is very difficult to read. An excellent and comprehensive treatment of data augmentation and multiple imputation is provided by Schafer (1997). Introductions to MI are given by Schafer (1999a), Little and Rubin (2002), and, for the DACSEIS project, by Rässler (2004).

The basic approach is to undertake imputation $m$ times to create $m$ imputed datasets. Standard complete-case analysis performed for each of the $m$ imputed datasets leads to $m$ estimates $\widehat{\theta}_1, \ldots, \widehat{\theta}_m$ of any given parameter $\theta$. The resulting point estimate of $\theta$ is taken to be the mean of these estimates. Suitable R-Code for pooling estimates is given in Appendix C. For the purpose of variance estimation it is necessary that the imputation method fulfils certain conditions, referred to as *proper* imputation by Rubin (1987), and that the complete-data estimates are asymptotically normal (like maximum likelihood estimates are) or $t$ distributed. The basic idea of a proper imputation method is that it should reflect the uncertainty about the missing value correctly. From a frequentist perspective the requirement is that randomization valid inference will be drawn from the multiply imputed data sets. For a discussion of the proper property see Brand (1999) or Rässler (2004). Provided imputation fulfils these criteria, a simple variance estimator can be constructed following the "MI paradigm" (Rubin, 1987). A common choice of $m$ is to take $m = 5$. Empirical evidence from the DACSEIS simulations suggests that is is better for variance estimation of Horvitz-Thompson type estimates to use $m = 15$ or even better $m = 30$. In principle, a higher number of imputations leads to better results of the MI estimate concerning efficiency and coverage, but usually smaller numbers of $m$ are sufficient, see Rubin (1987), pp. 114-115.

This chapter describes some multiple imputation procedures proposed for DACSEIS which, in principle, are proper MI routines. The problem of multivariate missing data is considered first, including the use of iterative methods, including regression switching, which permit multivariate methods to be constructed from univariate methods. Some of these univariate methods are then described with particular reference to the DACSEIS applications. These methods may be viewed as extensions of the regression imputation methods

considered in the previous chapter, which allow for additional uncertainty arising from the estimation of model parameters.

## 3.2 MI Methods for Multivariate Missing Data

### 3.2.1 Iterative Univariate Methods

One approach to handling general multivariate missing data patterns is based on the assumption that the variables follow a multivariate normal model, for example SCHAFER (1999b) implements procedures under this assumption in the software NORM. This approach has become quite popular for multiple imputation in multivariate settings. However, assuming a multivariate normal distribution for categorical variables with missing values is often not regarded as a good choice. Recently, RUBIN (2003) suggests iterative univariate multiple imputation procedures for large-scale data sets. They are successfully used for multiple imputation in the U.S. National Medical Expenditure Survey (NMES) where the data set to be imputed consists of up to 240 variables of different scales and 22,000 observations. Such routines have been used quite efficiently in the context of "mass imputation", i.e., imputing a high amount of data that are typically missing by design. This is the situation in the so-called data fusion case and the split questionnaire survey designs, see, e.g., RÄSSLER (2002). For the pseudouniverses and the simulation study to be performed within the DACSEIS project, we therefore suggest such multiple imputation routines as state-of-the-art. The advantages and disadvantages of this approach are described herein, also the necessary pseudocode is provided in S-PLUS/R.

It is said that iterative univariate imputations were first implemented by KENNICKELL (1991) and KENNICKELL (1994); see SCHAFER and OLSEN (1999).[1] The intuitively appealing idea behind the iterative univariate imputation procedure is to overcome the problem of suitably proposing and fitting a multivariate model for mixtures of categorical and continuous data by reducing the multivariate imputation task to conventional regression models iteratively completed. In many surveys it may be difficult to propose a sensible joint distribution for all variables of interest. On the other hand there is a variety of procedures available for regression modeling of continuous and categorical univariate response variables such as ordered or unordered logit/probit models (see GREENE, 2000). Thus any plausible regression model $Y|X = x, \Theta = \theta$ may be specified for predicting each univariate variable $Y_{mis}$ that has to be imputed given all the other variables. This approach is also known as regression switching, chained equations, or variable-by-variable Gibbs sampling; see VAN BUUREN and OUDSHOORN (1999). In the variable-by-variable Gibbs sampling approach it is also possible to include only relevant predictor variables, thus reducing the number of parameters.

[1] Ready to use and available for free via the Internet is software called MICE which is a recent implementation of some iterative univariate imputation methods in S-PLUS as well as R; see VAN BUUREN and OUDSHOORN (2000). Moreover, there is the free SAS-callable application IVEware, which also provides iterative univariate imputation methods.

### 3.2.2 Gibbs Sampling

Gibbs sampling is a Monte Carlo technique to simulate draws from a multivariate density function by repeatedly drawing from its conditional density functions, which is especially interesting when the joint distribution is not easily simulated but the conditional distributions are. Thus, even in a high dimensional problem, all of the simulations may be univariate, which usually is an advantage. For an introduction to understand the way the Gibbs sampler works the interested reader is referred to Casella and George (1992). We will shortly describe the main principle here.

Suppose that a $p$-dimensional random variable $U$ (here $U$ may denote the data as well as the parameters) is partitioned into non overlapping subvectors $(U_1, U_2, \ldots, U_k)$, $(k \leq p)$ containing all components of $U$. Let $f_U$ denote the joint distribution of $U$ and also the distribution of interest. Starting with an initial value $U^{(0)}$ of $U$, the Gibbs sampling algorithm generates a sequence of values $U^{(0)}, U^{(1)}, U^{(2)}, \ldots$ where in iteration $t$, $t \geq 1$, $U^{(t)}$ is generated from $U^{(t-1)}$ by iteratively drawing from the conditional distribution of each subvector given all the others. The value of $U^{(t)} = (U_1^{(t)}, U_2^{(t)}, \ldots, U_k^{(t)})$ is obtained by successively drawing from the distributions

$$\begin{aligned}
U_1^{(t)}|u_2^{(t-1)}, u_3^{(t-1)}, \ldots, u_k^{(t-1)} &\sim f_{U_1|U_2,U_3,\ldots,U_k}(u_1|u_2^{(t-1)}, u_3^{(t-1)}, \ldots, u_k^{(t-1)}) \\
U_2^{(t)}|u_1^{(t)}, u_3^{(t-1)}, \ldots, u_k^{(t-1)} &\sim f_{U_2|U_1,U_3,\ldots,U_k}(u_2|u_1^{(t)}, u_3^{(t-1)}, \ldots, u_k^{(t-1)}) \\
&\ldots \\
U_k^{(t)}|u_1^{(t)}, u_2^{(t)}, \ldots, u_{k-1}^{(t)} &\sim f_{U_k|U_1,U_2,\ldots,U_{k-1}}(u_k|u_1^{(t)}, u_2^{(t)}, \ldots, u_{k-1}^{(t)})
\end{aligned} \tag{3.1}$$

For each $U_j$ the value of $U_j^{(t)}$ is generated conditionally on the most recently drawn values of all other variables. According to Markov chain theory[2] the distribution of $U$ converges to the desired distribution $f_U$ under mild regularity conditions, i.e. the sequence $\{U^{(t)} : t = 0, 1, 2, \ldots\}$ has a stationary distribution equal to $f_U$.

### 3.2.3 Regression-switching

To illustrate the principle of the regression-switching let us assume the simple case with 3 variables $A$, $B$ and $C$ each with missing data. Then Rubin (2003) proposes:

- Begin by arbitrarily filling in all missing $B$ and $C$ values!
- Then, fit a model of $A|B, C$ using those units where $A$ is observed, and impute the missing $A$ values!
- Next, toss the imputed $B$ values, and fit a model of $B|A, C$ using those units where $B$ is observed, and impute the missing $B$ values!
- Next, toss the imputed $C$ values, and fit a model of $C|A, B$ using units where $C$ is observed, and impute the missing $C$ values!
- Iterate!

[2]For an excellent and very extensive description of Monte Carlo methods in general and Markov chain methods in particular see Robert and Casella (1999).

This procedure allows great flexibility due to the possible conditional specifications. Each specification simply is a univariate regression. It has to be mentioned that there are some theoretical shortcomings, because it is possible to generate incompatible distributions via implicit contradictions in the specified conditional specifications. The practical implications of this phenomenon in iterative univariate imputation are still quite unknown, see SCHAFER and OLSEN (1999). A "real" Gibbs sampler starts with an existing but intractable joint distribution for the variables of interest, iteratively generating random variables from easier to operate full conditional distributions derived from its joint distribution. In the context of iterative univariate imputations the conditional distributions are specified in the hope that these conditional distributions will define a suitable joint model. However, even if there is no such joint distribution for the data, the Markov chain Monte Carlo Method (MCMC) can be implemented, and each conditional specification may be a good empirical fit to the data, see RUBIN (2003), VAN BUUREN and OUDSHOORN (2000), and BRAND (1999).

## 3.3 MI Methods for Univariate Missing Data

In this section we consider MI methods for univariate missing data. These might be combined for multivariate missing data following the methods in the previous section.

### 3.3.1 Continuous Variables: HBS Type of Data

To impute missing data for a continuous variable $y$, such as income or expenditure in the Household Budget Surveys (HBS), we propose to extend the linear regression imputation approach of Section 2.6. In contrast to Chapter 2, we shall adopt a Bayesian framework, using prior distributions, which will be the usual uninformative or flat priors. The continuous variable (income or expenditure data, whichever has data missing) may be transformed by its logarithm before performing imputation. Notice that after imputation the values have to be transformed back. To assure that only values are imputed that lie within a certain range, also upper and lower bounds can be given. After performing the final imputation step for the missing $y$ values, each row of the imputed data set is examined to see whether any of the imputed values is out of range. In such cases these values are redrawn until the constraints are satisfied. According to SCHAFER (1997), p. 204, this procedure leads to approximate proper multiple imputations under a truncated normal model.

The basic algorithm is as follows.

- as in Section 2.4, assume the underlying linear regression model

$$y = \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + \epsilon = X\beta + \epsilon, \quad \epsilon \sim N(0, \sigma^2).$$

- Assume that $y$ has $n_{mis}$ missing data, variables $X$ are fully observed or already imputed. $y_{obs}$ and $X_{obs}$ refer to the jointly observed part, $X_{mis}$ to the missing part $y_{mis}$.

- Let $\widehat{\beta}$ and $\widehat{\sigma}^2 = (y_{obs} - X_{obs}\widehat{\beta})'(y_{obs} - X_{obs}\widehat{\beta})/(n_{obs} - p)$ be the least squared estimates from the observed data.
- Multiple imputation procedure for $j = 1, 2, \ldots m$:
  1. Draw $(\sigma^2|X) \sim (y_{obs} - X_{obs}\widehat{\beta})'(y_{obs} - X_{obs}\widehat{\beta})\chi^{-2}_{n_{obs}-p}$
  2. Draw a vector of $p$ variables from $(\beta|\sigma^2, X) \sim N(\widehat{\beta}, \sigma^2(X'_{obs}X_{obs})^{-1})$
  3. Draw $(Y_{mis}|\beta, \sigma^2, X) \sim N(X_{mis}\beta, \sigma^2)$ independently for every missing value $i = 1, 2, \ldots, n_{mis}$.

For the independent variables $X$ all available auxiliary variables from the universes may be taken. If both, income and expenditure have missing values, then the regression switching can be applied.

The resulting approach is very similar to the linear regression imputation plus added noise in the previous chapter, except that the additional uncertainty about the parameters $\beta$ and $\sigma^2$ is also allowed for.

### 3.3.2 Binary Variables: LFS Type of Data

For the target binary variable with missing values we base the imputations on a logistic regression model. For the Labour Force Surveys (LFS) and its data, the employment variable has to be either recoded to zero and one; e.g., to 1 if $y$ = unemployed and to 0 otherwise, or split into dummy variables; e.g., to employment (yes/no) and unemployment (yes/no) leaving the third category for non labor force or simply the rest. Also region should be recoded to dummy variables. Then the basic algorithm we propose for DACSEIS is as follows:

- Assume the underlying data model of a logistic regression

$$\ln\left(\frac{\theta}{1-\theta}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p = X\beta, \quad \theta = P(Y = 1|X).$$

- Assume that $y$ has $n_{mis}$ missing data, variables $X$ are fully observed or already imputed. $y_{obs}$ and $X_{obs}$ refer to the jointly observed part, $X_{mis}$ to the missing part $y_{mis}$.
- Let $\widehat{\beta}$ be the iterative least squares estimates from the observed data (or any approximate ML estimate) and $\widehat{V}(\widehat{\beta})$ its estimated covariance matrix (e.g., from the inverse Fisher information matrix $I(\widehat{\beta})^{-1}$).
- Apply the large sample normal approximation for $j = 1, 2, \ldots, m$:
  1. Draw a vector of $p$ variables from $(\beta|X) \sim N(\widehat{\beta}, \widehat{V}(\widehat{\beta}))$
  2. For every $i \in mis$ calculate $\theta_i = 1/(1 + \exp(-X'_i\beta)$.
  3. Draw $n_{mis}$ independent uniform (0, 1) random numbers $u_i$ for $i = 1, 2, \ldots, n_{mis}$ and if $u_i > \theta_i$ impute $Y_i = 0$, otherwise impute $Y_i = 1$.

For the independent variables $X$ again all available information may be taken.

If a variable has more than one category it is typically recoded into dummy variables. Then, if more than one (dummy) variable (after recoding) has missing data, then we may impute the most populous category first versus the rest. If zero is imputed, then we impute the next category versus the rest, and so on. If one is imputed all remaining categories are set to zero.

This approach is again similar to the regression imputation methods for binary variables described in the previous chapter, except that allowance is made for uncertainty about $\beta$.

### 3.3.3 Semicontinuous Variables

Semicontinuous variables were called mixed binary-continuous variables in the previous chapter. They take the value zero with positive probability and otherwise take a positive continuously distributed value. For example, such variables may occur in an LFS when respondents are asked the number of months they have been unemployed. These variables will have a large amount of zeros (the employed) and a continuous part (unemployment time).

For imputation we first impute the '0' vs. the '+' using the logistic regression, then, if a '+' is imputed the linear regression is used for imputing these missing values. This follows an approach of SCHAFER (1997), p. 381, published in detail by SCHAFER and OLSEN (1999), one may encode each semicontinuous variable $U$ to a binary indicator $W$ (with $W = 1$ if $U \neq 0$ and $W = 0$ if $U = 0$) and a continuous variable $V$ which is treated as missing whenever $U = 0$; for an illustration, see Figure 3.1.

| Unit no. | U |
|---|---|
| 1 | 12 |
| 2 | NA |
| 3 | 0 |
| 4 | 0 |
| 5 | NA |
| ... | ... |
| n-1 | 3 |
| n | 0 |

| Unit no. | W | V |
|---|---|---|
| 1 | 1 | 12 |
| 2 | NA | NA |
| 3 | 0 | NA |
| 4 | 0 | NA |
| 5 | NA | NA |
| ... | | ... |
| n-1 | 1 | 3 |
| n | 0 | NA |

Figure 3.1: Encoding semicontinuous variables

Notice that a relationship between $W$ and $V$ would have little meaning and could not be estimated by the observed data. However, we aim at generating plausible imputations for the original semicontinuous variable $U$ and, thus, are only interested in the marginal distribution for $W$ and the conditional distribution for $V$ given $W = 1$. MCMC procedures have been shown to behave well in this context with respect to the parameters of interest, see SCHAFER and OLSEN (1999).

## 3.4 Typical Problems that May Occur with the Chained Equation MI

### 3.4.1 Incorporating the Sampling Design

In the multiple imputation model, stratification can be incorporated by including strata indicators as covariates. Clustering may be incorporated by multilevel models that include random cluster effects, see Little and Rubin (2002), p. 90, or Schafer and Yucel (2002). On the other hand, these effects can be controlled by a design-based complete-data inference.

### 3.4.2 Collinearity

It should be noted that the problem of collinearity may occur when the coefficients of the linear regression model are to be estimated from the observed data. If some covariates of $X$ show only very little variability it may happen that the remaining part in $X_{obs}$, which belongs to the observed part of $Y$, has one or more variables with constant values. In these cases the algorithm given herein should be extended and a plausibility check performed that excludes such variables from the regression model.

### 3.4.3 Rounding Off

For discrete valued data, e.g. annual income in multiples of $1,000$ Euros, we propose to proceed with the linear regression model and then

1. either round off the imputed variable to equal one of the observable values in the data set or
2. use a "predictive mean matching" approach as discussed by Rubin (1986) and Little (1988).

The latter has the great advantage that only values which are really observed can be imputed and the imputation is even more robust against misspecification of the linear model. It has been used quite successfully for single imputation, see Rässler *et al.* (2002).

### 3.4.4 Monotone Missingness Versus Incompatibility

The regression-switching approach has the theoretical limitation of possibly generating incompatible distributions via implicit contradictions on their conditional specification. This may be of importance if the rate of missingness is high and there are a lot of different conditional specifications. If the missingness has a monotone pattern, for illustration see Figure 3.2, then we may impute the variable from left to right always regressing the

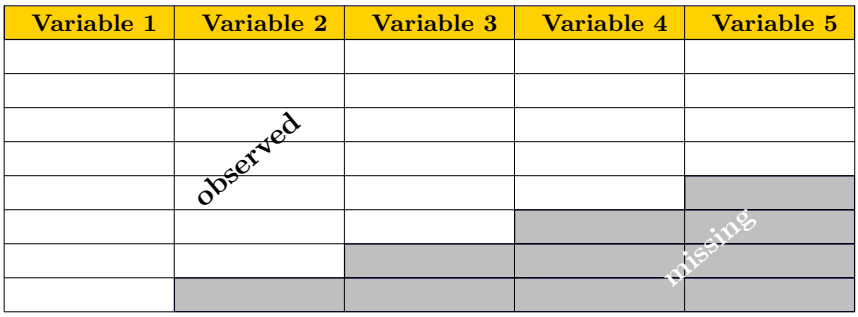


Figure 3.2: Monotone pattern of missingness

variable to be imputed on all other variables on the left. This procedure is continued until all missing values have been imputed.

Monotone patterns of missingness have the advantage that they can be handled noniteratively because each imputation model being fit is conditioning only on the variables to the left. The resulting univariate models are automatically distributionally compatible. When the missing data are not monontone but "approximately" monotone, RUBIN (2003) proposes to first impute those values which disturb the monotone missingness pattern regressing on all other variables (thus, possibly specifying an incompatible Gibbs sampler) and then impute the monotone missing values from left to right.

## 3.5 Conclusions

In general, when a model is used as a device for imputation, the meaning or interpretation of its parameters is not essential. The utility of the model lies in its ability to predict and simulate missing observations. A sensible imputation method for complex surveys should preserve basic relationships among variables. We believe that our proposed models are capable of preserving these effects. With multivariate longitudinal or clustered data also correlations among observations from the same subject or cluster should be preserved, too. Then more complex imputation models may be used, see SCHAFER and YUCEL (2002). In practice, inference by multiple imputation is quite robust to departures from the imputation model because that model applies not to the entire data set but only to its missing parts. For example, simulations by SCHAFER (1997) have shown that rounding off imputed values to the nearest category may only lead to minor biases. This also holds for departures from the missing at random (MAR) assumption. Empirical evidence suggests that multiple imputation under MAR often is quite robust against violations of this assumption. Even an erroneous assumption of MAR may have only minor impact on estimates and standard errors computed using multiple imputation strategies. Only when missing not at random (NMAR) is a serious concern and the fraction of missing information is substantial, does it seem necessary to model jointly the data and the missingness. Moreover, because the missing values cannot be observed, there is no direct evidence in the data to address the NMAR-assumption. Therefore, it can be more helpful to consider several alternative models and to explore the sensitivity of resulting inferences.

Hence, we conclude that the regression-switching approach seems to be quite promising in large data sets and also for high amounts of missing values. Even in the context of "mass imputation", i.e., split questionnaire survey designs and data fusion we find good frequentist properties. In the U.S. the regression-switching multiple imputation approach is basically applied in the NHANES (a split project) and NMES. The basic routines are already implemented in MICE (SPLUS and R version) and IVEware, Raghunathan's SAS callable application. We have provided some pseudocode especially for the DACSEIS universes.

Let us finish with a quote from the recent Little and Rubin (2002) book, p. 90: "The right way to assess the relative merits of the methods, from a frequentist perspective, is through comparisons of their repeated-sampling operating characteristics in real settings, not their theoretical etiologies."

# Appendix A

# Imputation of Missing Income Data in the Swiss Household Budget Survey

The Swiss household budget survey 1998 (HBS98), was conducted in 12 monthly waves of a stratified simple random sample of private households in Switzerland with $9,295$ fully participating households. During a month, each participating household reported all its expenditures and the income of its members. Missing data were detected based on supplementary information like the economic activity status given by the participating persons themselves (implying a certain type of income) or simply based on a 'don't want to give' -declaration of the persons. This led to a higher percentage of detected item non-response in the INCOME variable than the EXPENDITURE variable. In general, item non-response is not extremely frequent in the HBS98 data since households that did not deliver a complete report were considered as non-participating and thus were joined to the unit non-response.

It was found that the **INCOME and EXPENDITURE** variables are quite correlated. Therefore an imputation model for the total of household income explained by the total of household expenditures was used to impute missing INCOME data. Because of the observed distribution with a strong wing towards higher income, a **LOG10 transformation** was applied before the regression. Additionally, a more robust **L1** (or LAD) regression was used as a first step in order to identify outliers. After discarding those outliers, a classic **L2** (or LS) regression was applied to the remaining points in order to obtain the coefficients for imputation. These regressions were done separately for each socio-economic group (STASOCIO). The INCOME value was only imputed if the calculated value was higher than the original value (concerns households with partially missing income data). After imputation the discarded outliers were joined back to the data for the estimation procedure.

A flowchart description for the implemented imputation model in the HBS98 data looks as follows:

1. **Sum up** all income and all expenditures per household to the total per household. (These totals correspond to the INCOME and EXPENDITURE variables of the HBS98 Pseudo Universes.)

2. Transform the totals of income and expenditures with the **LOG10** transformation.
3. Identify all households with **missing** (also partially missing) data in income. Separate them from the others. (All households: 9,295; households with missing and partially missing income: 505; households without missing income: 8,790)
4. Calculate **residuals** of **L1 regression** (LOG_INC = beta1_L1_k * LOG_EXP; without intercept) for each socio-economic group (k stands for the 6 classes of the variable STASOCIO).
5. Divide the residuals by the STD_MAD (=1.48***MAD**); outliers are defined as observations with absolute residual strictly greater than 2.5.
6. Separate the **outliers** from the others.
7. Calculate the **L2 regression coefficients** (LOG_INC = beta1_L2_k * LOG_EXP; without intercept) on the remaining points for each socio-economic group.
8. Calculate the imputed total income from the total expenditures for each household with missing (or partially missing) income data using the obtained L2 regression coefficients: **LOG_INC = beta1_L2_k * LOG_EXP**
9. **Impute** total income only if the imputed value is greater than the original value (concerns households with partially missing income data). (Of the 505 households with missing income data, only 304 were imputed because of this condition).
10. Recalculate the imputed INCOME on the **linear** scale (INCOME = 10**LOG_INC).

The beta1_L2_k coefficients that were actually used are the following:

- STASOCIO = -1 0.9913
- STASOCIO = 1 1.0185
- STASOCIO = 2 0.9986
- STASOCIO = 3 0.9941
- STASOCIO = 4 0.9949
- STASOCIO = 5 1.0119

# Appendix B

# Core SAS Codes for Single Imputation Methods

The code in this Appendix is organised according to the headings in Sections 2.4 and 2.6.

**Case 1. Estimated linear regression model for continuous target variable**

```
proc glm data=pattern1a;
/*this implies the file or the missingness pattern*/
class z1 z2;
/*these are categorical variables being used in the model if
not mentioned under class statement, the variable will
continuous*/
model y = z1 z2 x1 x2 x3/solution; /*model specification*/
output out = pattern1b p=ypred;
/*name for output file plus a new variable with the predicted
values in this file; the residuals may be included
automatically, too*/
run; /*running the programme*/
```

or if the intercept has not used:

```
proc glm data=pattern1a;
class z1 z2;
model y = z1 z2 x1 x2 x3 /solution noint;
output out = pattern1b p=ypred; run;
/*this specification does not give the intercept for the
model*/
```

**Note**: if there are robust methods available for this model, please give this opportunity for a user.

The output of this program gives, for example, a value for 'ROOT MSE' (if you cannot automatically exploit ROOT MSE then you have to calculate it using residuals or (y-ypred), thus working with the output file = pattern1b).

Thus: we will have the predicted values for **y = ypred** and we may calculate the respective values taking into account the model uncertainty. This depends on the assumption of the

error term. In a standard case, this may be simply based on a normal distribution, that is, we obtain:

**yprednoise = ypred + rannor($u$)*ROOT MSE**

in which rannor($u$) is random term with mean = 0 and standard deviation = 1 with $u$ = seed number.

All robustness tests possible could be included in this process. Also, if a certain re-scaling ( e.g. = log) has been used in modelling, the values should have been further transformed to the initial dimension (in case of log, this means exp-transformation).

Moreover, a user should be able to specify the certain bounds for this solution, so not to include the full range of rannor($u$) but for example as follows;

```
data pattern1b; set pattern1b;
if rannor(u) > k then rannor(u) =k;
if rannor(u) < -m then rannor(u)=-m
/*usually k = m, i.e. symmetric.*/
```

A more demanding option that requires a new data step is to draw the noise term from the set of observed residuals. This drawing may be done either with or without replacement.

**Case 2. Estimated logistic model for categorical (or categorised) target variable**

We here are looking only at the binary case, that is, $y = 1$ or $y = 0$.

```
proc logistic data=pattern2a;
class z1 z2;
model y = z1 z2 x1 x2 x3 ;
output out = pattern2b p=ypred;
run;
```

**Case 3. Estimated logistic model for missingness indicator**

This case is analogous to case 2 as far as the model specification is concerned but the target variable differs from it essentially. Now it is a categorical variable of the response/missingness mechanism so that $R = 1$ if the value is non-missing, and $R = 0$ if it is missing. Thus we obtain the analogous program to the previous one:

```
proc logistic data=pattern3a descending;
class z1 z2;
model R = z1 z2 x1 x2 x3 ;
output out = pattern3b p=Rpred;
run;
```

In this case, the predicted values are 'non-missingness probabilities', response probabilities, response propensities, propensity scores (all of these terms are used), and lie within the interval (0, 1).

### Donor Methods Specifications

The data statement in SAS, e.g.:

```
Data pattern10; set pattern1b;
if ypred < .1 then impcell=1;
else if ypred < .2 then impcell=2;
...
else if ypred < .9 then impcell=9;
else impcell=10;
```

The latter case needs to calculate cumulative frequencies and then divide these into cells, respectively. There is an option is SAS for this purpose.

**A2b. Possible options for choosing a donor when the metrics has been determined:**

1. Value from a nearest donor to a unit with the missing value. This thus requires calculating all the distances for this unit.

For example, if the metrics are based on the predicted values, then these values are good to sort first, and then the nearest ones from above and then from below this unit are compared (the metric distances calculated), and the nearest is chosen.

For SAS, this can be done e.g. so that a reasonable number of lagged variables have been constructed for both directions (below and above). This solution works both for continuous and categorical variables, thus for linear models and logistic models. In the latter case, the predicted values (*ypred*) are probabilities, the response propensities (*Rpred*).

```
proc sort data= pattern1b; by ypred; /*sorting by predicted y*/
data pattern11; set pattern1b;
lagy1=lag(y);
lagy3=lag2(y);
lagy5=lag3(y);
lagy7=lag4(y);
lagy9=lag5(y);

/*lagged variables for observed values*/

lagp1=lag(ypred);
lagp3=lag2(ypred);
lagp5=lag3(ypred);
lagp7=lag4(ypred);
lagp9=lag5(ypred);

/*lagged variables for predicted values*/

run;

proc sort data=pattern11; by descending ypred;

/*sorting to the opposite order and constructing the lagged
variables as above*/
```

```
data pattern11;
set pattern11;
lagy2=lag(y);
lagy4=lag2(y);
lagy6=lag3(y);
lagy8=lag4(y);

lagp2=lag(ypred);
lagp4=lag2(ypred);
lagp6=lag3(ypred);
lagp8=lag4(ypred);
lagp10=lag5(ypred);

diff10=abs(lagp1-ypred);

/*absolute distances to the unit with missing value in the
neighborhood*/

diff20=abs(lagp2-ypred);
diff30=abs(lagp3-ypred);
diff40=abs(lagp4-ypred);
diff50=abs(lagp5-ypred);
diff60=abs(lagp6-ypred);
diff70=abs(lagp7-ypred);
diff80=abs(lagp8-ypred);
diff90=abs(lagp9-ypred);
diff100=abs(lagp10-ypred);

r=ranuni(8);

/*random number with uniform distribution*/

if y ne. then yimpnn=y;
/*imputation starts, yimpnn = observed or
imputed value*/ else if yimpnn=. and diff10<diff20  then
yimpnn=lagy1; else if yimpnn=. and diff10>diff20  then
yimpnn=lagy2; else if yimpnn=. and diff10=diff20  and r<.5 then
yimpnn=lagy2;

/*random choice for equal distance*/

else if yimpnn=. and diff10=diff20 and r>.5 then yimpnn=lagy1;
else if yimpnn=. and diff10<diff40  then yimpnn=lagy1;
else if yimpnn=. and diff10>diff40  then yimpnn=lagy4;
else if yimpnn=. and diff10=diff40  and r<.5 then yimpnn=lagy4;
else if yimpnn=. and diff10=diff40  and r>.5 then yimpnn=lagy1;
else if yimpnn=. and diff10<diff60  then yimpnn=lagy1;
else if yimpnn=. and diff10>diff60  then yimpnn=lagy6;
else if yimpnn=. and diff10=diff60  and r<.5 then yimpnn=lagy6;
else if yimpnn=. and diff10=diff60  and r>.5 then yimpnn=lagy1;
else if yimpnn=. and diff10<diff80  then yimpnn=lagy1;
else if yimpnn=. and diff10>diff80  then yimpnn=lagy8;
else if yimpnn=. and diff10=diff80  and r<.5 then yimpnn=lagy8;
else if yimpnn=. and diff10=diff80  and r>.5 then yimpnn=lagy1;
else if yimpnn=. and diff30<diff20  then yimpnn=lagy3;
else if yimpnn=. and diff30>diff20  then yimpnn=lagy2;
else if yimpnn=. and diff30=diff20  and r<.5 then yimpnn=lagy3;
```

```
else if yimpnn=. and diff30=diff20  and r>.5 then yimpnn=lagy2;
else if yimpnn=. and diff30<diff40  then yimpnn=lagy3;
else if yimpnn=. and diff30>diff40  then yimpnn=lagy4;
else if yimpnn=. and diff30=diff40  and r<.5 then yimpnn=lagy4;
else if yimpnn=. and diff30=diff40  and r>.5 then yimpnn=lagy3;
else if yimpnn=. and diff30<diff60  then yimpnn=lagy3;
else if yimpnn=. and diff30>diff60  then yimpnn=lagy6;
else if yimpnn=. and diff30=diff60  and r<.5 then yimpnn=lagy3;
else if yimpnn=. and diff30=diff60  and r>.5 then yimpnn=lagy6;
else if yimpnn=. and diff30<diff80  then yimpnn=lagy3;
else if yimpnn=. and diff30>diff80  then yimpnn=lagy8;
else if yimpnn=. and diff20=diff80  and r<.5 then yimpnn=lagy2;
else if yimpnn=. and diff20=diff80  and r>.5 then yimpnn=lagy8;

/*only the nearest neighbours have been chosen completely
correctly, next ones so that the first one from one side,
next from the opposite, etc. some type of stopping criterion,
that is how far the nearest can be lying, should be applied;
here the maximum is = 10; if all these are still missing
(empty), the imputed value will remain missing, too*/

else if yimpnn=. then yimpnn=lagy5;
else if yimpnn=. then yimpnn=lagy6;
else if yimpnn=. then yimpnn=lagy7;
else if yimpnn=. then yimpnn=lagy8;
else if yimpnn=. then yimpnn=lagy9;
else if yimpnn=. then yimpnn=lagy10;
```

# Appendix C

# S-Plus / R codes for the Proposed Imputation Methods

## C.1 Single Imputation for Continuous Data

See Section 2.6 for discussion of the methods in this section and Section C.2.

### C.1.1 Regression Imputation

```
function(Xorg, Yorg, mis = -9, lower = - Inf, upper = Inf, loglin = FALSE,
intercept = FALSE)
{
# name: SIlinreg()
# Performs a single regression imputations for a continuous variable with
# missing data
# Y is the univariate variable which has missing values described by mis
# Xorg can be a vector or a matrix containing the covariates
# lower and upper bounds of the values to be imputed can be considered
# if necessary, the data can be transformed before imputing them
#
######################## data preparation ###################################
#
    n <- length(Yorg)
    if(min(Yorg, na.rm = T) < lower) {
        print(c("Lower bound is too high, set to minimum observed value"))
        lower <- min(Yorg, na.rm = T)
    }
    if(max(Yorg, na.rm = T) > upper) {
        print(c("Upper bound is too low, set to maxmimum observed value"))
        upper <- max(Yorg, na.rm = T)
    }
    Y <- ifelse(Yorg == mis, NA, Yorg)
    if(loglin) {
        Y <- log(Y)
        if(lower != - Inf) {
            lower <- log(lower)
```

```
        }
        upper <- log(upper)
    }
    if(is.matrix(Xorg) == F) {
        Xorg <- as.matrix(Xorg)
    }
    q <- length(Xorg[1, ])
    XYobs <- na.omit(cbind(Xorg, Y))
    Xobs <- XYobs[, 1:q]
    Yobs <- XYobs[, (q + 1)]
#
############################ SI as specification ############################
####### model-donor without noise term, pure regression imputation #######
#
    if(intercept) {
        regress <- lm(Yobs ~ Xobs)
        X <- cbind(c(rep(1, n)), Xorg)
    }
    else {
        regress <- lm(Yobs ~ Xobs - 1)
        X <- Xorg
    }
    fittedY <- X %*% regress$coef
    Yimp <- ifelse(Y == "NA", fittedY, Y)
#
############################ preparing the output ############################
#
    if(loglin) {
        Yimp <- cbind(Xorg, exp(Yimp))
    }
    else {
        Yimp <- cbind(Xorg, Yimp)
    }
    Yimp
}
```

### C.1.2 Ratio Imputation for Swiss Data

```
function(cell, Xorg, Yorg, mis = -9, lower = -Inf, upper = Inf,
loglin = FALSE)
{
# name: SIratio()
# Performs a single ration imputations for a continuous variable with
# missing data
# Yorg is the univariate variable which has missing values described by
# mis (income)
# Xorg is the univariate covariate (expenditure)
# cell it the vector of the cell variable
# lower and upper bounds of the values to be imputed can be considered
# if necessary, the data can be transformed before imputing them
#
########################## data preparation ##################################
#
```

```
    n <- length(Yorg)
    label <- seq(1, n, 1)
    if(min(Yorg, na.rm = T) < lower) {
        print(c("Lower bound is too high, set to minimum observed value"))
        lower <- min(Yorg, na.rm = T)
    }
    if(max(Yorg, na.rm = T) > upper) {
        print(c("Upper bound is too low, set to maxmimum observed value"))
        upper <- max(Yorg, na.rm = T)
    }
    Y <- ifelse(Yorg == mis, NA, Yorg)
    if(loglin) {
        Y <- log(Y)
        if(lower != -Inf) {
            lower <- log(lower)
        }
        upper <- log(upper)
    }
    cXY <- cbind(cell, Xorg, Y, label)
#
############################ SI as specification ############################
#################### ratio imputation within cells ######################
#
    tab <- table(cell)
    l <- length(tab)
    names <- as.numeric(dimnames(tab)[[1]])
    XYimp <- c(NA, NA, NA, NA)
    for(i in 1:l) {
        XYcell <- cXY[cXY[, 1] == names[i], ]
        XYobs <- na.omit(XYcell)
        q <- sum(XYobs[, 3])/sum(XYobs[, 2])
        Yimp <- ifelse(XYcell[, 3] == "NA", q * XYcell[, 2], XYcell[, 3])
        XYcell.imp <- cbind(XYcell[, 1:2], Yimp, XYcell[, 4])
        XYimp <- rbind(XYimp, XYcell.imp)
    }
#
########################### preparing the output ############################
#
    XYimp <- na.omit(XYimp)
    if(loglin) {
        XYimp[, 3] <- exp(XYimp[, 3])
    }
    XYimp <- XYimp[order(XYimp[, 4]), ]
    XYimp[, 1:3]
}
```

## C.2 Single Imputation for LFS Type of Data

### C.2.1 Specification (i)

```
function(cell, Yorg, wor = FALSE, mis = -9)
{
# name: SILFS1()
# Performs a single imputation for a categorical variable with
# missing data
# Yorg is the univariate variable which has missing values
# described by mis
# cell is the vector of the cell variable
# imputation can be done with and without replacement
#
########################## data preparation ##############################
#
    Y <- ifelse(Yorg == mis, NA, Yorg)
    n <- length(Yorg)
    label <- seq(1, n, 1)
    cY <- cbind(cell, Y, label)
#
########################## SI as specification ###########################
################### model (1): Hot deck within cells #####################
#
    tab <- table(cell)
    l <- length(tab)
    names <- as.numeric(dimnames(tab)[[1]])
    cY.imp <- c(NA, NA, NA)
    for(i in 1:l) {
        Ycell <- cY[cY[, 1] == names[i], ]
        n.i <- length(Ycell[, 1])
        if(wor) {
            Yobs <- na.omit(Ycell)
            Yimp <- Ycell[, 2]
            for(j in 1:n.i) {
                if(Yimp[j] == "NA") {
                Yobs <- na.omit(Yobs)
                nobs.i <- length(Yobs[, 1])
                number <- round(runif(1, 0.51, (nobs.i + 0.5)), 0)
                Yimp[j] <- Yobs[number[j], 2]
                Yobs[number[j], 2] <- NA
                }
            }
        }
        else {
            Yobs <- na.omit(Ycell)
            nobs.i <- length(Yobs[, 1])
            nmis.i <- n.i - nobs.i
            number <- round(runif(n.i, 0.51, (nobs.i + 0.5)), 0)
            Ydonor <- rep(NA, n.i)
            for(j in 1:n.i) {
                Ydonor[j] <- Yobs[number[j], 2]
            }
            Yimp <- ifelse(Ycell[, 2] == "NA", Ydonor, Ycell[, 2])
        }
```

```
        Ycell[, 2] <- Yimp
        cY.imp <- rbind(cY.imp, Ycell)
    }
#
############################ preparing the output ############################
#
    cY.imp <- na.omit(cY.imp)
    cY.imp <- cY.imp[order(cY.imp[, 3]), ]
    cY.imp[, 1:2]
}
```

### C.2.2 Specification (ii)

```
function(Xorg, Yorg, mis = -9)
{
# name: SILFS2()
# model (ii) as proposed by Laaksonen, a predictive mean matching on
# estimated logit model
# Performs logistic imputation for a binary variable with missing data
# Y is the univariate dichotomous variable which has missing values
# described by mis
# Xorg can be a vector or a matrix containing the covariates
#
###############################################################################
#
    n <- length(Yorg)
    Y <- ifelse(Yorg == mis, NA, Yorg)
    X <- cbind(c(rep(1, n)), Xorg)
    q <- length(X[1, ])
    XYobs <- na.omit(cbind(X, Y))
    Xobs <- XYobs[, 1:q]
    Yobs <- XYobs[, (q + 1)]
    logobs <- summary(glm(Yobs ~ Xobs - 1, family = binomial(link = logit)
        ))
    beta <- logobs$coefficients[, 1]

    prob <- 1/(1 + exp( - X %*% beta))
    probobs <- 1/(1 + exp( - Xobs %*% beta))
    Yimp <- Y
    for(i in 1:n) {
        if(Y[i] == "NA") {

            Ydonor <- Yobs[min(abs(prob[i] - probobs))
                == abs(prob[i] - probobs)]
            l <- length(Ydonor)
#there can be more than one donor if there is too less variability in X!
#Thus, choose one at random (hot deck!) or take the median or mode
            if(l > 1) {
                number <- round(runif(1, 0.51, (l + 0.5)), 0)
                Yimp[i] <- Ydonor[number]
            }
            else {
                Yimp[i] <- Ydonor
```

```
            }
        }
    }
    Yimp <- cbind(Xorg, Yimp)
    Yimp
}
```

### C.2.3 Specification (iii)

```
function(Xorg, Yorg, mis = -9)
{
# name: SILFS3()
# model (iii) as proposed by Laaksonen, a propensity score matching
# Performs logistic imputation for a binary response variable
# Y is the univariate dichotomous original variable which has
# missing values described by mis
# Xorg can be a vector or a matrix containing the covariates
#
#############################################################################
#
    n <- length(Yorg)
    Yorg <- ifelse(Yorg == mis, NA, Yorg)
    Y <- ifelse(Yorg == "NA", 0, 1)
    X <- cbind(c(rep(1, n)), Xorg)
    q <- length(X[1, ])
    XYobs <- na.omit(cbind(X, Yorg))
    Xobs <- XYobs[, 1:q]
    Yobs <- XYobs[, (q + 1)]
    logit <- summary(glm(Y ~ X - 1, family = binomial(link = logit)))
    beta <- logit$coefficients[, 1]

    prob <- 1/(1 + exp( - X %*% beta))
    probobs <- 1/(1 + exp( - Xobs %*% beta))
    Yimp <- Yorg
    for(i in 1:n) {
        if(Yorg[i] == "NA") {
            Ydonor <- Yobs[min(abs(prob[i] - probobs))
                == abs(prob[i] - probobs)]
            l <- length(Ydonor)
#there can be more than one donor if there is too less variability in X!
#Thus, choose one at random (hot deck!) or take the median or mode
            if(l > 1) {
                number <- round(runif(1, 0.51, (l + 0.5)), 0)
                Yimp[i] <- Ydonor[number]
            }
            else {
                Yimp[i] <- Ydonor
            }
        }
    }
    Yimp <- cbind(Xorg, Yimp)
    Yimp
}
```

### C.2.4 Specification (iv)

```
function(Xorg, Yorg, wor = FALSE, mis = -9)
{
# name: SILFS4()
# model (iv) as proposed by Laaksonen, a hot deck propensity score
# matching
# Performs logistic imputation for a binary response variable
# Y is the univariate dichotomous original variable which has missing
# values described by mis
# Xorg can be a vector or a matrix containing the covariates
# wor: imputation with and without replacement
#
#############################################################################
#
n <- length(Yorg)
label <- seq(1, n, 1)
Yorg <- ifelse(Yorg == mis, NA, Yorg)
Y <- ifelse(Yorg == "NA", 0, 1)
X <- cbind(c(rep(1, n)), Xorg)
q <- length(X[1, ])
logit <- summary(glm(Y ~ X - 1, family = binomial(link = logit)))
beta <- logit$coefficients[, 1]
prob <- 1/(1 + exp( - X %*% beta))

########################## Hot deck within cells ############################
#
prob.q <- quantile(prob, seq(0, 1, 0.1))
print(prob.q)
cell <- rep(1, n)
for(i in 2:10)
    cell <- ifelse(((prob > prob.q[i - 1]) & (prob <= prob.q[i])), i, cell
        )
print(cell)
cY <- cbind(cell, Yorg, label)
tab <- table(cell)
l <- length(tab)
names <- as.numeric(dimnames(tab)[[1]])
cY.imp <- c(NA, NA, NA)
for(i in 1:l) {
    Ycell <- cY[cY[, 1] == names[i], ]
    n.i <- length(Ycell[, 1])
    if(wor) {
        Yobs <- na.omit(Ycell)
        Yimp <- Ycell[, 2]
        for(j in 1:n.i) {
            if(Yimp[j] == "NA") {
                Yobs <- na.omit(Yobs)
                nobs.i <- length(Yobs[, 1])
                number <- round(runif(1, 0.51, (nobs.i + 0.5)), 0)
                Yimp[j] <- Yobs[number[j], 2]
                Yobs[number[j], 2] <- NA
            }
        }
    }
    else {
```

```
        Yobs <- na.omit(Ycell)
        nobs.i <- length(Yobs[, 1])
        nmis.i <- n.i - nobs.i
        number <- round(runif(n.i, 0.51, (nobs.i + 0.5)), 0)
        Ydonor <- rep(NA, n.i)
        for(j in 1:n.i) {
            Ydonor[j] <- Yobs[number[j], 2]
        }
        Yimp <- ifelse(Ycell[, 2] == "NA", Ydonor, Ycell[, 2])
    }
    Ycell[, 2] <- Yimp
    cY.imp <- rbind(cY.imp, Ycell)
}
Yimp <- na.omit(cY.imp)
Yimp <- Yimp[order(Yimp[, 3]), 2]
Yimp <- cbind(Xorg, Yimp)
Yimp
}
```

## C.3 Multiple Imputation Codes

### C.3.1 Pooling the Estimates

Description: function MIinference()

Input variables:

- thetahat: a vector of $m$ standard complete-case estimates,
- varhat.thetahat: a vector of $m$ estimated variances of the standard complete-case estimates,
- alpha: the significance level, default value 5%,
- theta: the true value of the population parameter to be estimated, if available, default is set to not available.

Output variables:

- theta: the true value of the population parameter to be estimated, if available,
- MIestimate: the MI estimate of theta,
- CI.low: lower bound of the confidence interval,
- CI.up: upper bound of the confidence interval,
- width: width of the confidence interval,
- cvg: the coverage which is 1, if the true parameter is element of the confidence interval, 0, if not, and NA, if theta is not available,

- B: the between-imputation variance,
- W: the within-imputation variance,
- total: the total variance of the MI estimate.

```
function(thetahat, varhat.thetahat, alpha = 0.05, theta = NA)
{
# name: MIinference()
# Calculates the MI estimate, the between-imputation variance,
# the within-imputaton variance, the total variance,
# the confidence interval, and, if possible, the coverage
#
#########################################################################
#
   m <- length(thetahat)
   lambda <- 1 - (alpha/2)
   MIestimate <- mean(thetahat)
   B <- var(thetahat, unbiased = T)
   W <- mean(varhat.thetahat)
   total <- W + (1 + 1/m) * B
   DF <- (m - 1) * (1 + W/((1 + 1/m) * B))^2
   CI.low <- MIestimate - qt(lambda, DF) * sqrt(total)
   CI.up <- MIestimate + qt(lambda, DF) * sqrt(total)
   if(theta != "NA") {
       if((CI.low <= theta) & (CI.up >= theta)) {
           cvg <- 1
       }
       else {
           cvg <- 0
       }
   }
   else {
       cvg <- NA
   }
   width <- CI.up - CI.low
   c(theta, MIestimate, CI.low, CI.up, width, cvg, B, W, total)
}
```

### C.3.2 MI Algorithm for Continuous Variables: HBS Type of Data

Description: function MIlinreg()

Input variables:

- Xorg: a vector or a matrix containing the independent variables of the regression without constant,
- Yorg: a vector of the continuous variable with missing values,
- mis: the code of the missing values in Yorg, default ist set to -9,

- lower: the lower bound, if the Yorg values have such, default is - Inf,
- upper: the upper bound, if the Yorg values have such, default is Inf,
- loglin: specifies whether a loglinear model should be used or not, default ist set to FALSE,
- m: number of imputed data sets, default is 5.

Output variables:

- Yimpit: a list consisting of $m$ imputed data sets ready for complete-data analyses.

```
function(Xorg, Yorg, mis = -9, lower = - Inf, upper = Inf,
loglin = FALSE, m = 5)
{
# name: MIlinreg()
# Performs proper multiple imputations for a continuous variable
# with missing data according to RUBIN (1987), p. 167
# Y is the univariate variable which has missing values coded
# with mis
# Xorg can be a vector or a matrix containing the covariates
# lower and upper bounds of the values to be imputed can be
# considered
# if necessary, the data can be transformed before imputing them
#
#####################################################################
#
    n <- length(Yorg)
    if(min(Yorg, na.rm = T) < lower) {
        print(c("Lower bound is too high, set to
                minimum observed value"))
        lower <- min(Yorg, na.rm = T)
    }
    if(max(Yorg, na.rm = T) > upper) {
        print(c("Upper bound is too low, set to
                maxmimum observed value"))
        upper <- max(Yorg, na.rm = T)
    }
    Y <- ifelse(Yorg == mis, NA, Yorg)
    if(loglin) {
        Y <- log(Y)
        if(lower != - Inf) {
            lower <- log(lower)
        }
        upper <- log(upper)
    }
    n1 <- length(na.omit(Y))
    X <- cbind(c(rep(1, n)), Xorg)
    q <- length(X[1, ])
    XYobs <- na.omit(cbind(X, Y))
    Xobs <- XYobs[, 1:q]
    Yobs <- XYobs[, (q + 1)]
    Yimpit <- list(Y)
```

```
    Vobs <- solve(t(Xobs) %*% Xobs)
    betadachobs <- Vobs %*% (t(Xobs) %*% Yobs)
    sigma2obssum <- sum((Yobs - Xobs %*% betadachobs)^2)
    #
    ######################## MI ####################################
    #
    for(i in 1:m) {
        sigma2 <- sigma2obssum/rchisq(1, (n1 - q))
        beta <- rmvnorm(1, mean = as.vector(betadachobs),
                        cov = (sigma2 * Vobs))
        redo <- TRUE    /*control for bounds*/
        while(redo) {
            Ymis <- rnorm(n, (X %*% t(beta)), sqrt(sigma2))
            Yimp <- ifelse(Y == "NA", Ymis, Y)
            Yimp <- ifelse(Yimp < lower, NA, Yimp)
            Yimp <- ifelse(Yimp > upper, NA, Yimp)
            if(length(na.omit(Yimp)) == n) {
                redo <- FALSE
            }
        }
        if(loglin) {
            Yimpit[[i]] <- cbind(Xorg, exp(Yimp))
        }
        else {
            Yimpit[[i]] <- cbind(Xorg, Yimp)
        }
    }
    Yimpit
}
```

### C.3.3 MI Algorithm for Binary Variables: LFS Type of Data

Description: function MIlogit()

Input variables:

- Xorg: a vector or a matrix containing the independent variables of the regression without constant,
- Yorg: a vector of the binary variable with missing values,
- mis: the code of the missing values in Yorg, default ist set to -9,
- m: number of imputed data sets, default is 5.

Output variables:

- Yimpit: a list consisting of $m$ imputed data sets ready for complete-data analyses.

```
function(Xorg, Yorg, mis = -9, m = 5)
{
# name: MIlogit()
# Performs proper multiple imputations for a binary variable with
# missing data according to RUBIN (1987), pp. 169-170.
# Y is the univariate dichotomous variable which has missing values
# coded with mis
# Xorg can be a vector or a matrix containing the covariates
#
#####################################################################
#
   n <- length(Yorg)
   Y <- ifelse(Yorg == mis, NA, Yorg)
   n1 <- length(na.omit(Y))
   X <- cbind(c(rep(1, n)), Xorg)
   q <- length(X[1, ])
   XYobs <- na.omit(cbind(X, Y))
   Xobs <- XYobs[, 1:q]
   Yobs <- XYobs[, (q + 1)]
   Yimpit <- list(Y)
   logobs <- summary(glm(Yobs ~ Xobs - 1,
             family = binomial(link = logit)))
   betadachobs <- logobs$coefficients[, 1]

   se.betadachobs <- logobs$coefficients[, 2]

   cor.betadachobs <- logobs$correlation

   cov.betadachobs <- cor.betadachobs * (se.betadachobs
                                         %*% t(se.betadachobs))
#
############################ MI ######################################
#
   for(i in 1:m) {
       beta <- rmvnorm(1, mean = as.vector(betadachobs),
                       cov = cov.betadachobs)
       beta <- as.vector(beta)
       prob <- 1/(1 + exp( - X %*% beta))
       randomuniform <- runif(n)
       Ymis <- ifelse(randomuniform > prob, 0, 1)
       Yimp <- ifelse(Y == "NA", Ymis, Y)
       Yimpit[[i]] <- cbind(Xorg, Yimp)
   }
   Yimpit
}
```

# References

**Brand, J. P. L. (1999)**: *Development, Implementation and Evaluation of Multiple Imputation Strategies for the Statistical Analysis of Incomplete Data Sets*. Thesis Erasmus University Rotterdam, Print Partners Ispkamp, Enschede, The Netherlands.

**Casella, G. and George, E. I. (1992)**: Explaining the Gibbs sampler. *The American Statistician* **46**, 167–174.

**Greene, W. (2000)**: *Econometric Analysis*. Fourth edition. Upper Saddle River NJ.

**Groves, R. M., Dillman, D. A., Eltinge, J. L. and Little, R. (2002)**: *Survey Nonresponse*. New York: John Wiley & Sons.

**Kennickell, A. B. (1991)**: *Imputation of the 1989 Survey of Consumer Finances: Stochastic Relaxation and Multiple Imputation*. Proceedings of the Survey Research Methods Section, American Statistical Association, 1-10.

**Kennickell, A. B. (1994)**: *Multiple Imputation of the 1983 and 1989 Waves of the SCF*. Proceedings of the Survey Research Methods Section, American Statistical Association, 523-528.

**Laaksonen, S. (2000)**: Regression-based nearest neighbor hot decking. *Computational Statistics* **15**, 65–71.

**Laaksonen, S. (2002a)**: Need for high level auxiliary data service for improving the quality of editing and imputation. In *Paper for the UNECE Work Session on Data Editing in Helsinki, 27-29 May*. available on the UNECE website: `http://www.unece.org`.

**Laaksonen, S. (2002b)**: Traditional and new techniques for imputation. *The Journal Statistics in Transition* **9**, 1013–1036.

**Lee, H., Rancourt, E. and Särndal, C. E. (2002)**: Variance estimation from survey data under single imputation. In *Survey Nonresponse*, eds R. M. Groves, D. A. Dillman, J. L. Eltinge and R. Little. New York: John Wiley & Sons.

**Little, R. J. A. (1988)**: Missing data adjustments in large surveys. *Journal of Business & Economic Statistics* **6**, 287–297.

**Little, R. J. A. and Rubin, D. B. (2002)**: *Statistical Analysis with Missing Data*. Second edition. New York: John Wiley & Sons.

**Meng, X. L. (1994)**: Multiple-imputation inferences for uncongenial sources of input. *Statistical Science* **9**, 538–558.

**Nielsen, S. F. (2003)**: Proper and improper multiple imputation (with discussion). *International Statistical Review* **71**, 593–627.

**Rao, J. N. K. and Shao, J. (1992)**: Jackknife variance estimation with survey data under hot deck imputation. *Biometrika* **79**, 811–822.

**Rässler, S. (2002)**: *Statistical Matching: A Frequentist Theory, Practical Applications and Alternative Bayesian Approaches*. Lecture Notes in Statistics, 168. New York: Springer-Verlag.

**Rässler, S. (2004)**: *The Impact of Multiple Imputation for DACSEIS*. DACSEIS Research Paper number 5.

**Rässler, S., Koller, F. and Mäenpää, C. (2002)**: A split questionnaire survey design applied to German media and consumer surveys. In *Proceedings of the International Conference on Improving Surveys, ICIS 2002, Copenhagen*.

**Robert, C. P. and Casella, G. (1999)**: *Monte-Carlo Statistical Methods*. New York: Springer-Verlag.

**Rubin, D. B. (1986)**: Statistical matching using file concatenation with adjusted weights and multiple imputations. *Journal of Business and Economic Statistics* **4**, 87–95.

**Rubin, D. B. (1987)**: *Multiple Imputation for Nonresponse in Surveys*. New York: Wiley.

**Rubin, D. B. (1996)**: Multiple imputation after 18+ years. *Journal of the American Statistical Association* **91**, 473–489. with discussion, 507-515, and rejoinder, 515-517.

**Rubin, D. B. (2003)**: Nested multiple imputation of NMES via partially incompatible MCMC. *Statistica Neerlandica* **57**, 3–18.

**Rubin, D. B. and Schenker, N. (1986)**: Multiple imputation for interval estimation from simple random samples with ignorable nonresponse. *Journal of the American Statistical Association* **81**, 366–374.

**Schafer, J. L. (1997)**: *Analysis of Incomplete Multivariate Data*. Chapman & Hall, London.

**Schafer, J. L. (1999a)**: Multiple imputation: A primer. *Statistical Methods in Medical Research* **8**, 3–15.

**Schafer, J. L. (1999b)**: *Multiple Imputation under a Normal Model*. Version 2, Software for Windows 95/98/NT. `http://www.stat.psu.edu/~jls/misoftwa.html`.

**Schafer, J. L. and Olsen, M. K. (1999)**: *Modeling and Imputation of Semicontinuous Survey Variables*. The Pennsylvania State University. Technical Report No. 00-39.

**Schafer, J. L. and Yucel, R. M. (2002)**: Computational strategies for multivariate linear mixed effects models with missing values. *Journal of Computational and Graphical Statistics* **11**, 437–457.

**Shao, J. (2002)**: Replication methods for variance estimation in comlex surveys with imputed data. In *Survey Nonresponse*, eds R. M. Groves, D. A. Dillman, J. L. Eltinge and R. J. A. Little, pp. 303–314. New York: John Wiley & Sons.

**Van Buuren, S. and Oudshoorn, C. G. M. (1999)**: *Flexible Multivariate Imputation by MICE*. TNO Report PG/VGZ/99.054, Leiden.

**Van Buuren, S. and Oudshoorn, C. G. M. (2000)**: *Multivariate Imputation by Chained Equations*. TNO Report PG/VGZ/00.038, Leiden.