Generalized Additive Model Multiple Imputation by Chained Equations With Package ImputeRobust

by Daniel Salfran, Martin Spiess

Abstract Data analysis, common to all empirical sciences, often requires complete data sets. Unfortunately, real world data collection will usually result in data values not being observed. We present a package for robust multiple imputation (the **ImputeRobust** package) that allows the use of generalized additive models for location, scale, and shape in the context of chained equations. The paper describes the basics of the imputation technique which builds on a semi-parametric regression model (GAMLSS) and the algorithms and functions provided with the corresponding package. Furthermore, some illustrative examples are provided.

Introduction

A common approach to allow valid inferences in the presence of missing data is "Multiple Imputation" (MI) introduced by Rubin (1987). Application of MI can be summarized in three steps. The first step is to create $m > 1$ sets of completed data sets by replacing each missing value with m values drawn from an appropriate posterior predictive distribution ("imputations"). In the second step, the required statistical analysis technique is applied to each of the completed data sets as if it were a completely observed data set. The third step is the pooling step, where the results from the m analyses are combined to form the final results, and allows statistical inferences in the usual way.

Basically, there are two ways of specifying imputation models: Joint modelling and fully conditional specification. The joint modelling approach requires specification of a multivariate distribution for the variables whose values are not observed and drawing imputations from their predictive posterior distribution using Markov Chain Monte Carlo (MCMC) techniques. Within this framework, the most common assumption is that the data is multivariate normally distributed (Schafer, 1997). Other alternatives for categorical data are based on the latent normal model (Albert and Chib, 1993) or the general location model (Little and Rubin, 2002).

This methodology is attractive if the assumed multivariate distribution is an appropriate model for the data but may lack the flexibility needed to deal with complex data sets encountered in applications. In such cases, the joint modelling approach may be too restrictive because the typical specification of a multivariate distribution is not sufficiently flexible to allow different continuous and discrete distributions (He and Raghunathan, 2009). For example, most of the existing model-based methods and software implementations assume that the data originate from a multivariate normal distribution (e.g. Honaker et al., 2011; Templ et al., 2011; van Buuren, 2007). However, the assumption of normality is inappropriate as soon as there are outliers in the data, or in the case of skewed, heavy-tailed or multimodal distributions, potentially leading to deficient results (van Buuren, 2012; He and Raghunathan, 2012).

With the fully conditional specification, also known as multivariate imputation by chained equations (van Buuren and Groothuis-Oudshoorn, 2011), a univariate imputation model is specified for each variable with missing values conditional on other variables of the data set. Starting from initially bootstrapped imputations, subsequent imputations are drawn by iterating over conditional densities (van Buuren and Groothuis-Oudshoorn, 2011; van Buuren, 2007). This framework splits high-dimensional imputation models into multiple one-dimensional problems and is appealing as an alternative to joint modelling in cases where a proper multivariate distribution can not be found. The choice of imputation models in this setting can vary depending on the type of variable to be imputed, for example, parametric models like the Bayesian linear regression or logistic regression. Liu et al. (2013) studied the asymptotic properties of this iterative imputation procedure and provided sufficient conditions under which the imputation distribution converges to the posterior distribution of a joint model when the conditional models are compatible.

De Jong (2012) and de Jong et al. (2014) proposed a new imputation technique based on generalized additive models for location, scale, and shape, GAMLSS, (Rigby and Stasinopoulos, 2005), which is a class of univariate regression models, where the assumption of an exponential family is relaxed and replaced by a general distribution family. This allows a more flexible modelling than standard parametric imputation models, not only based on the location (e.g. the mean), but also the scale (e.g. variance), and the shape (e.g., skewness and kurtosis) of the conditional distribution of the dependent variable to be imputed given all other variables. The R package **ImputeRobust**, described in the next section, provides the functions necessary to apply the GAMLSS imputation technique to missing data problems. It can be used as standalone tool or in combination with **mice** (van Buuren and Groothuis-Oudshoorn, 2011). Salfran (2018) provides an extensive comparison study of the new package and other Multiple Imputation techniques.

Robust imputation with gamlss and mice

The imputation method realized by the package **ImputeRobust** is based on a generalized additive model for location, scale and shape of the variable to be imputed. Specifically, we adopt the semi-parametric additive formulation of GAMLSS described in Stasinopoulos and Rigby (2007).

Let $Y = (Y_{ij})$, $i = 1, \ldots, n$ and $j = 1, \ldots, p$ be a matrix with n independent observations on p variables and let y_{ij} be the realization of variable Y_j with probability function $f(y_{ij}|\theta_{ij})$ conditional on parameters $\theta_{ij} = (\theta_{ij}^k)$, $k = 1, \ldots, 4$. Assume that g_k are known monotonic link functions relating the distribution parameters θ_{ij}^k to explanatory variables by the following equation:

$$g_k(\theta_{ij}^k) = \eta_k = X_k\beta_k + \sum_{l=1}^{L_k} h_{lk}(x_{lk}), \tag{1}$$

where θ_{ij}^k for $k = 1, 2, 3, 4$ are the location, scale and shape parameters of the distribution, X_k is a fixed known design matrix, β_k^T a vector of linear predictors, and $h_{lk}(x_{lk})$ are unknown smoothing functions of the explanatory variables.

Not all four parameters may be needed, depending on the conditional distribution $f(y_{ij}|\theta_{ij})$ which will be denoted as $\mathcal{D}$. The package **gamlss** (Rigby and Stasinopoulos, 2005) provides a wide range of possible continuous and discrete conditional distributions with varying number of parameters, although not all of these distributions have been adopted yet to create imputations (see Table 1 on section "Main functions").

The proposed imputation method selects the conditional distribution $\mathcal{D}$ for each of the variables to be imputed with the MICE algorithm (van Buuren and Groothuis-Oudshoorn, 2011). This distribution defaults to normal for continuous data, but other alternatives may be chosen. Users can take advantage of this option to restrict imputations to a certain range, e.g., by specifying a truncated normal distribution. Alternatives already included are Logit and Poisson models to handle binary and count data.

The chosen distribution $\mathcal{D}$ defines the type and number of parameters to be modelled, e.g., for the default normal distribution the mean and variance are estimated (individually for each point), but other distributions may require the estimation of the skewness and kurtosis in addition. Adopting models with more parameters increases their flexibility and thus may increase the chance that the imputation procedure is proper in the sense of Rubin (1987). On the other hand, larger sample sizes may be needed to identify the larger number of parameters.

Implementation

The implementation of the imputation method for our simulations uses the **gamlss** package (see Rigby and Stasinopoulos, 2005; Stasinopoulos and Rigby, 2007) in R to fit model (1) based on (penalized) maximum likelihood estimation and adopting the default link functions. Rigby and Stasinopoulos (2005) and Stasinopoulos and Rigby (2007) provide a description of the fitting algorithms used by this package. As smoothing functions h_{jk}, we use P-splines with 20 knots, a piecewise polynomial of degree three, a second order penalty and automatic selection of the smoothing parameter using the Local Maximum Likelihood criterion (see Eilers and Marx, 1996). To prevent abnormal termination of the algorithm, for example if samples are too small, the degree of the polynomial, the order of the penalty, or the stopping time of the fitting algorithm are reduced as a fallback strategies.

Let Y_j be one incompletely observed column of Y. The observed and missing parts of Y_j are denoted by Y_j^{obs} and Y_j^{mis}, respectively. Let $Y_{-j} = (Y_1, \ldots, Y_{j-1}, Y_{j+1}, \ldots, Y_p)$ denote the collection of variables in Y except Y_j. The package **gamlss** does not support Bayesian inference, hence it is not possible to obtain multiple imputations by drawing from the posterior predictive distribution. However, draws from the predictive posterior distribution are approximated by the bootstrap predictive distribution (Harris, 1989):

$$f^\star(Y_j^{mis}|Y_j^{obs}, Y_{-j}) = \int f(Y_j^{mis}|\tilde{\eta}, Y_{-j}^{mis}) f(\tilde{\eta}|\hat{\eta}(Y_j^{obs}, Y_{-j}^{obs})) d\tilde{\eta}\,, \tag{2}$$

where $\tilde{\eta}$ denotes the possible values of the imputation model parameters, $\hat{\eta}(Y_j^{obs}, Y_{-j}^{obs})$ is an estimator of such parameters, and $f(\tilde{\eta}|\hat{\eta}(Y_j^{obs}, Y_{-j}^{obs}))$ is the sampling distribution of the imputation parameters evaluated at the estimated values. The sampling distribution is simulated with a parametric bootstrap acting as a replacement for the posterior distribution of the imputation parameters. Algorithm 1 describes the imputation process. For a clear presentation, we drop the index i.

Algorithm 1 GAMLSS imputation

1. Fit model $Y_j \sim \mathcal{D}(\hat{\mu}_j, \hat{\sigma}_j, \hat{\eta}_j, \hat{\tau}_j)$ using the observed data $\{Y_j^{obs}, Y_{-j}^{obs}\}$

2. Resample Y_j^{obs} as follows:

$$Y_{j\star}^{obs} \sim \mathcal{D}(\hat{\mu}_j, \hat{\sigma}_j, \hat{\eta}_j, \hat{\tau}_j)\,.$$

   Define a bootstrap sample $B = \{Y_{j\star}^{obs}, Y_{-j}^{obs}\}$

3. Refit the above model using B. This leads to adapted estimators $\dot{\mu}_j, \dot{\sigma}_j, \dot{\eta}_j$ and $\dot{\tau}_j$. Draw n_{mis} imputations for Y_j^{mis} as follows:

$$\tilde{Y}_j^{mis} \sim \mathcal{D}(\dot{\mu}_j, \dot{\sigma}_j, \dot{\eta}_j, \dot{\tau}_j)\,.$$

4. Repeat m times steps 2 and 3 to generate m imputed data sets.

Main functions

The two main functions included in the **ImputeRobust** package are the imputing and fitting functions `mice.impute.gamlss()` and `ImpGamlssFit()`, respectively.

The function `ImpGamlssFit()` is internal and its job is to read in the data and model parameters and create a bootstrap predictive sampling function, i.e., it will work through steps 1 to 3 of Algorithm 1. The fitting step depends on the **gamlss** package, and the same control options described in Stasinopoulos and Rigby (2007) can be passed to `ImpGamlssFit()` when calling the `mice()` function. By default, `ImpGamlssFit()` uses P-splines as the smoothers in model (1), which are considered to be stable in the **gamlss** package, but sometimes the fitting algorithm may diverge. The implemented imputation method will catch this event and will try to correct it by automatically cutting down the complexity of the model.

The necessary formula objects for the model are automatically created by the function during execution time. The type of imputation model and its parameters, like the degree of the P-splines, can be controlled with the argument `gam.mod`. Another way of controlling the complexity of the fitting and imputation steps is the `lin.terms` argument. This argument can be used to specify variables by their name, that should enter model (1) linearly.

The default response distribution family used by the fitting and imputation methods is the normal distribution, but it can be modified with the argument `family`. The selected family determines how many parameters are to be modelled. To improve the stability of the imputation method, distributional parameters can be restricted to be the same for all units. The maximum number of parameters to be fitted is controlled by `n.ind.par`, that is, the value of k in equation (1). For example, if the Johnson' SU family (a four parameter continuous distribution) is selected and `n.ind.par = 2`, then the mean and the variance are modelled individually with p-splines, but the shape parameters are restricted to be the same for all units and are modelled as contant values.

The function `mice.impute.gamlss()` has the same structure as the imputation methods included in the **mice** package, meaning that the named method `"gamlss"` can be directly passed as an argument to the `mice()` function. This function also passes all modified default arguments to the fitting function.

Additional functions are included in the package to set the `family` and `n.ind.par` arguments to non-default values. This allows users to mix different gamlss imputation methods within one call to the function `mice()`. The functions are variantes of `mice.impute.gamlss()` where `"gamlss"` is replaced with a method from Table 1 with the syntax `mice.impute.method()`. The name of the function is a reference to the corresponding family from `gamlss.family` (see Stasinopoulos and Rigby, 2007)

Method	Model distribution
gamlssNO	Normal
gamlssBI	Binomial
gamlssGA	Gamma
gamlssJSU	Johnson’s SU
gamlssPO	Poisson
gamlssTF	Student’s t
gamlssZIBI	Zero inflated Binomial
gamlssZIP	Zero inflated Poisson

Table 1: Included univariate gamlss imputation models.

Usage

The imputation methods provided by **ImputeRobust** add a new semiparametric method to the already included methods in **mice**. This means that it can be used directly with the function `mice()`.

Simple example

As an illustration let us consider an example using the proposed method to estimate the parameters in a linear regression model with multiple imputation. To do this we created a data set with $n = 500$ units composed of four independent variables and one dependent variable with the following code:

```
> set.seed(19394)
> n <- 500
> mu <- rep(0, 4)
> Sigma <- diag(4)
> Sigma[1,2] <- 0.15; Sigma[1,3] <- 0.1; Sigma[1,4] <- -0.1
> Sigma[2,3] <- 0.25; Sigma[2,4] <- 0.05
> Sigma[lower.tri(Sigma)] = t(Sigma)[lower.tri(Sigma)]
> require(”MASS”)
> rawvars <- mvrnorm(n, mu = mu, Sigma = Sigma)
> pvars <- pnorm(rawvars)
> X.1 <- rawvars[,1]
> X.2 <- qchisq(pvars, 3)[,3]
> X.3 <- qpois(pvars, 2.5)[,2]
> X.4 <- qbinom(pvars, 1, .4)[,4]
> data <- cbind(X.1, X.2, X.3, X.4)
> beta <- c(1.8, 1.3, 1, -1)
> sigma <- 4.2
> y <- data %*% beta + rnorm(n, 0, sigma)
> data <- data.frame(y, data)
```

Thus, we obtain correlated covariates $X.1, \ldots, X.4$ which are random draws from specific distributions, that is, values for $X.1$ are drawn from a standard normal random distribution, values for $X.2$ are drawn from a χ^2 distribution with three degrees of freedom, values for $X.3$ are drawn from a Poisson distribution with parameter $\lambda = 2.5$, and values for $X.4$ come from a Bernoulli distribution with parameter $\pi = 0.4$. The dependent variable, y, is created according to the linear regression model:

$$y_i = \beta_0 + X.1\beta_1 + X.2\beta_2 + X.3\beta_3 + X.4\beta_4 + \epsilon, \quad \epsilon \sim N(0, \sigma^2). \tag{3}$$

The vector of linear predictors, β, and the error variance, σ^2, are chosen so that the coefficient of determination, R^2, is 0.5.

In this first example, we create missing values in $X.2$ to $X.4$ with a monotone MAR mechanism dependent on y and $X.1$ as shown:

```
> r.s <- cbind(y, X.1) %*% c(2,1)
> r.s <- scale(r.s)
> pos <- cut(r.s, quantile(r.s, c(0, .5, 1)), include.lowest=TRUE)
> p.r <- as.numeric(c(.9, .2))
> p.r <- as.vector(p.r[pos])
> R2 <- as.logical(rbinom(length(p.r),1,p.r))
> r.s <- cbind(y[!R2], X.1[!R2]) %*% c(2,1)
> r.s <- scale(r.s)
> pos <- cut(r.s, quantile(r.s, c(0, .4, 1)), include.lowest=TRUE)
> p.r <- as.numeric(c(.32, .27))
> p.r <- as.vector(p.r[pos])
> R3 <- as.logical(rbinom(length(p.r),1,p.r))
> R4 <- runif(nrow(data[!R2,][!R3,]), 0, 1) >= .25
> data$X.2[!R2] <- NA
> data$X.3[!R2][!R3] <- NA
> data$X.4[!R2][!R3][!R4] <- NA
```

More precisely, to generate missing values in $X.2$ and $X.3$, we first generate variables $r^* = 2y + X.1$. Let Ri be a response indicator, where $Ri = 1$ if $X.i$ is observed and $Ri = 0$ if $X.i$ is not observed. Then missing values in $X.2$ and $X.3$ are generated according to

$$\Pr(R2 = 1|r^*) = \begin{cases} 0.9 & \text{if } r^* \leq r^*_{0.5} \\ 0.2 & \text{else,} \end{cases}, \; \Pr(R3 = 1|r^*, R2 = 0) = \begin{cases} 0.32 & \text{if } r^* \leq r^*_{0.4} \\ 0.27 & \text{else,} \end{cases}$$

and $\Pr(R3 = 1|R2 = 1) = \Pr(R3 = 1|r^*, R2 = 1) = 1$. Finally, under both conditions, missing values in $X.4$ are generated independently from y and $X.1$ with probability 0.25 for those units for which $X.3$ is not observed. The frequency of missing values and the pattern can be visualized with the **mice** function `md.pattern()`:

```
> library(ImputeRobust)
> md.pattern(data)
    y X.1 X.4 X.3 X.2
276 1   1   1   1   1   0
 66 1   1   1   1   0   1
127 1   1   1   0   0   2
 31 1   1   0   0   0   3
    0   0  31 158 224 413
```

The output is generated by the **mice** package, for details see van Buuren and Groothuis-Oudshoorn (2011). It can be seen that only 276 out of the 500 units are fully observed and that the missing pattern is monotone with the smallest amount of missing values in $X.4$ followed by $X.2$. $X.3$ has the highest amount of missing values.

The imputation task can be performed with a simple call of the `mice()` function:

```
> predictorMatrix <- matrix(c(rep(c(0,0,1,1,1),2), rep(0,5), c(0,0,1,0,0),
+                             c(0,0,1,1,0)), nrow = 5)
> imps <- mice(data, method = "gamlss", predictorMatrix = predictorMatrix,
+               visitSequence = "monotone", maxit = 1, seed = 8913)
 iter imp variable
  1   1  X.4  X.3  X.2
  1   2  X.4  X.3  X.2
  1   3  X.4  X.3  X.2
  1   4  X.4  X.3  X.2
  1   5  X.4  X.3  X.2
```

In this example with a monotone missing pattern, the missing values are imputed in accordance with Rubin (1987, p. 171). The variables are ranked according to the amount of missing values and imputations are drawn starting with the most frequently observed incomplete variable. In the next step, the most frequently observed variable of the remaining incompletely observed variables is imputed. This procedure continues until all variables with missing values are completed, using completely observed but also completed variables to impute the next. In `mice()` the arguments `visitSequence` and `predictorMatrix` control the column order in which incomplete variables are imputed and which variables serve as predictors in each imputation model, respectively. By setting these arguments to non-default values the required order of imputing values is achieved.

The result is an object of class Multiply Imputed Data Set (`mids`) with contents:

```
> print(imps)
Multiply imputed data set
Call:
mice(data = data, method = "gamlss", predictorMatrix = predictorMatrix,
    visitSequence = "monotone", maxit = 1, seed = 8913)
```

```
Number of multiple imputations:  5
Missing cells per column:
  y X.1 X.2 X.3 X.4
  0   0 224 158  31
Imputation methods:
       y      X.1      X.2      X.3      X.4
"gamlss" "gamlss" "gamlss" "gamlss" "gamlss"
VisitSequence:
X.4 X.3 X.2
  5   4   3
PredictorMatrix:
    y X.1 X.2 X.3 X.4
y   0   0   0   0   0
X.1 0   0   0   0   0
X.2 1   1   0   1   1
X.3 1   1   0   0   1
X.4 1   1   0   0   0
Random generator seed value:  8913
```

The argument method = "gamlss" in the mice function call implies by default, that imputations are drawn assuming a normal distribution for the response variable in the imputation model given all covariables. This has been shown to lead to acceptable results in situations in which even non-plausible values are imputed (de Jong et al., 2014). A useful way of inspecting the distribution of the original and an imputed data is the stripplot() function as shown in figure 1.

```
> library(lattice)
> stripplot(imps)
```

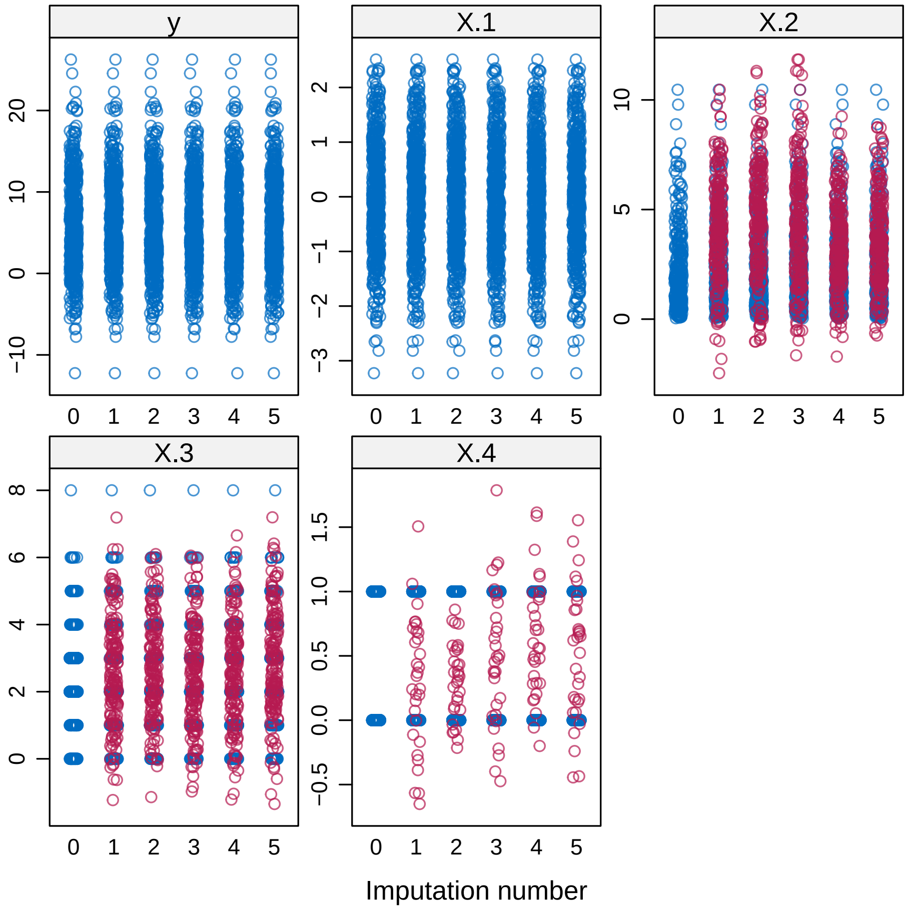

Figure 1: Stripplot of the five variables in the original and the five imputed data sets. Observed data values are blue and imputed data values are red.

The model of interest, as per equation (3), is the linear regression of y on $X.1$, $X.2$, $X.3$, and $X.4$ that created the original data set. The true value of the regression coefficient is `c(1.8,1.3,1,-1)`. The imputed data sets can be analysed as follows:

```
> fit <- with(imps, lm(y ~ X.1 + X.2 + X.3 + X.4))
> print(pool(fit))
Call: pool(object = fit)

Pooled coefficients:
(Intercept)         X.1         X.2         X.3         X.4
  1.0729769   1.7565698   1.1600009   0.6262502  -0.4023920

Fraction of information about the coefficients missing due to nonresponse:
(Intercept)         X.1         X.2         X.3         X.4
  0.4477033   0.5111319   0.9276184   0.7934569   0.2925600

> round(summary(pool(fit)), 2)
              est   se     t    df Pr(>|t|) lo 95 hi 95 nmis  fmi lambda
(Intercept)  1.07 0.57  1.87 22.89     0.07 -0.11  2.26   NA 0.45   0.40
X.1          1.76 0.26  6.79 17.73     0.00  1.21  2.30    0 0.51   0.46
X.2          1.16 0.29  4.00  4.47     0.01  0.39  1.93  224 0.93   0.90
X.3          0.63 0.27  2.33  6.89     0.05 -0.01  1.26  158 0.79   0.74
X.4         -0.40 0.46 -0.87 49.39     0.39 -1.33  0.52   31 0.29   0.26
```

Modifying the imputation model

The default behaviour of the `"gamlss"` method of using a normal distribution can be overridden by setting the argument `family` to any distribution contained in the **gamlss.dist** package (Stasinopoulos et al., 2017). This will define globally the new response distribution for all imputation methods that call the `"gamlss"` method. If the user wants to set different distribution families, to suit the particularities of the variables to be imputed, several functions fully compatible with `mice()` are already included.

In the previous example, variables $X.3$ and $X.4$ were treated as continuous variables in the imputation step, and it was possible for $X.2$ to take on negative values, even though is a strictly positive variable. We will show now how the GAMLSS imputation model can be adjusted to accommodate different types of distributions simultaneously. For the new imputation task, we use a Gamma distribution for $X.2$, a Poisson distribution for the imputation of $X.3$, and a Binomial distribution for $X.4$. The incomplete data set is the same as before, where $X.1$ and y need not to be imputed.

```
> imps <- mice(data, method = c("", "", "gamlssGA", "gamlssPO", "gamlssBI"),
+              seed = 8913)
 iter imp variable
  1   1  X.2  X.3  X.4
  1   2  X.2  X.3  X.4
  1   3  X.2  X.3  X.4
  1   4  X.2  X.3  X.4
  1   5  X.2  X.3  X.4
  2   1  X.2  X.3  X.4
  2   2  X.2  X.3  X.4
  ...
```

Figure 2 shows the effect of the changes in the response distribution for the imputation model. Choosing the Gamma distribution for $X.2$, the Poisson distribution for $X.3$ and the Binomial distribution for $X.4$ imputes realistic values as compared to, e.g., choosing the normal distribution. Whether to favor the imputation of realistic values or not, is a different problem. Since the final goal is statistical validity of MI based estimation (Rubin, 1996), in some applications it may be better to allow for the imputation of "unrealistic" while using a flexible model, for example, impute continuous values when the variable to be imputed is a count variable (see de Jong, 2012; Salfran, 2018).

```
> stripplot(imps)
```

The model of interest in this example is the same as in section 2.4.1. The results of running the analysis are slightly different due to choosing different imputation models.

```
> fit <- with(imps, lm(y ~ X.1 + X.2 + X.3 + X.4))
> round(summary(pool(fit)), 2)
```

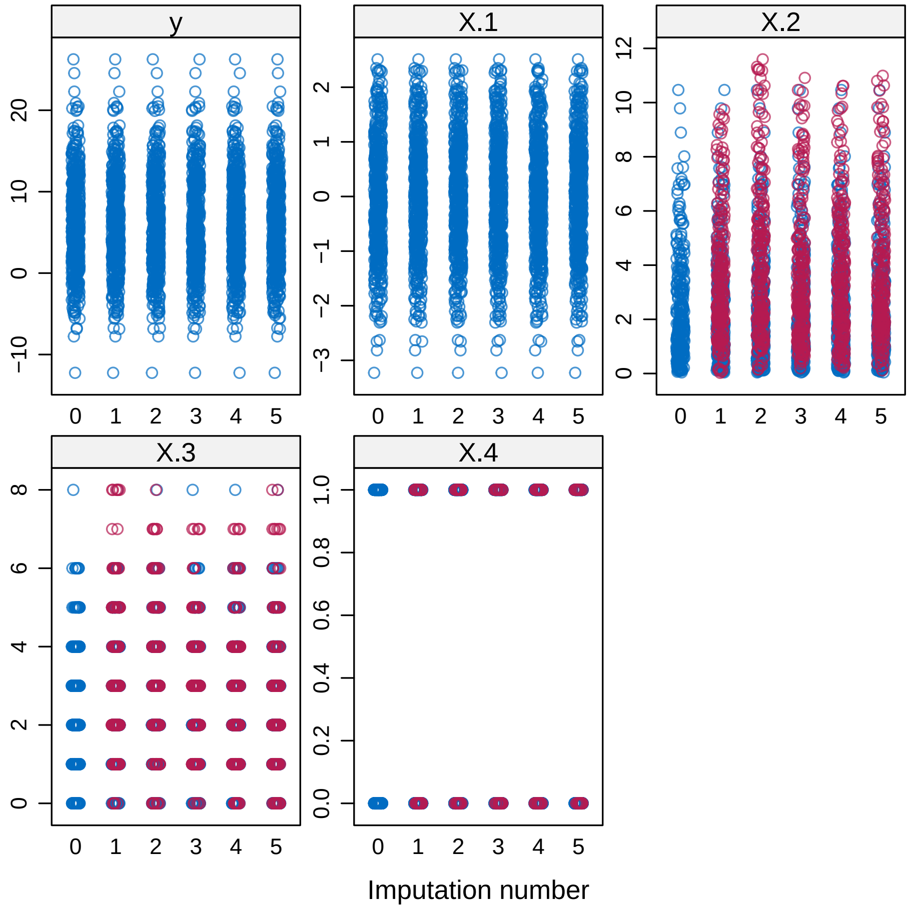

Figure 2: Stripplot of the five variables in the original data and the five imputed data sets. Observed data values are blue and imputed data values are red.

```
              est   se     t     df Pr(>|t|) lo 95 hi 95 nmis  fmi lambda
(Intercept)  0.78 0.48  1.61  40.46     0.11 -0.20  1.76   NA 0.33   0.30
X.1          1.78 0.25  7.27  18.00     0.00  1.27  2.30    0 0.51   0.46
X.2          1.19 0.13  8.89  10.43     0.00  0.90  1.49  224 0.66   0.60
X.3          0.70 0.21  3.37   8.90     0.01  0.23  1.18  158 0.71   0.65
X.4         -0.32 0.40 -0.80 127.39     0.43 -1.12  0.48   31 0.16   0.15
```

Chronic kidney disease data

A real world example data set was retrieved from the Machine Learning Database Repository at the University of California, Irvine (Lichman, 2013). The dataset contains 400 observations from which some variables were selected. The data already contained 149 rows where some values were missing. This example illustrates some of the specific extra arguments that can be passed to the imputation method, mainly with the objective of controlling the speed of the gamlss() function.

```
> library(RWeka)
> data <- read.arff("data/chronic_kidney_disease_full.arff")
> data <- data[,c("age", "bp", "bgr", "bu", "sc", "sod", "pot", "hemo",
+              "class")]
> data$class <- ifelse(data$class == "ckd", 1, 0)
```

The missing data pattern is non-monotone. The fully conditional specification approach behind **mice** will handle this case. Running mice with the default fitting arguments of gamlss in complex data sets like this would take a long time. There are several ways in which the speed of the imputation can be adjusted. One alternative is by changing the values controlling the GAMLSS fitting algorithm. The arguments n.cyc, bf.cyc, cyc control respectively the number of cycles of the outer, backfitting, and inner algorithms of gamlss (Rigby and Stasinopoulos, 2005; Stasinopoulos and Rigby, 2007). Other arguments that can be changed are the convergence criterions of the outer and inner algorithms with the `c.crit` or `cc` arguments, respectively.

The choice of imputation model can also be modified by the user. The relevant argument is `gam.mod`. The default imputation model is a P-spline with two degrees of freedom and second order differences, this amounts to `gam.mod = list(type = "pb",par = list(degree = 2,order = 2)`. The degrees and order can be changed. Also, the model can be set to be linear using `type = "linear"`.

The following code shows the result of imputing missing values of the chronic kidney disease data set. With the current parameters and without any other optimization this took 10 minutes computing time with a Core i7-3520M CPU.

```
> meth <- c("gamlssJSU", "gamlssJSU", "gamlssJSU", "gamlssJSU", "gamlssJSU",
+           "gamlssJSU", "gamlssJSU", "gamlssJSU", "gamlss")
> imps <- mice(data, method = meth, n.cyc = 3, bf.cyc = 2, cyc = 2,
+              maxit = 5, m = 5, seed = 8901,
+              gam.mod = list(type = "linear"))
 iter imp variable
  1   1  age  bp  bgr  bu  sc  sod  pot  hemo
  1   2  age  bp  bgr  bu  sc  sod  pot  hemo
  1   3  age  bp  bgr  bu  sc  sod  pot  hemo
  1   4  age  bp  bgr  bu  sc  sod  pot  hemo
  1   5  age  bp  bgr  bu  sc  sod  pot  hemo
  2   1  age  bp  bgr  bu  sc  sod  pot  hemo
  2   2  age  bp  bgr  bu  sc  sod  pot  hemo
  ...
```

Figure 3 shows that even with an extreme simplification of the fitting algorithm of `gamlss`. The imputation method still manages to produce acceptable values.

```
> stripplot(imps)
```

We used the imputed data set to fit a logistic regression in which we modelled the probability of having chronic kidney disease as a function of age and some other blood related information.

```
> fit <- with(imps, glm(class ~ age + bp + bgr + sc + sod + pot + hemo,
+                       family=binomial(link='logit')))
> round(summary(pool(fit)), 2)
              est    se     t     df Pr(>|t|)  lo 95 hi 95 nmis  fmi lambda
(Intercept) 33.04 12.85  2.57  79.40     0.01   7.47 58.61   NA 0.21   0.19
age         -0.03  0.02 -1.16  33.22     0.25  -0.07  0.02    9 0.36   0.32
bp           0.09  0.04  2.45 134.55     0.02   0.02  0.16   12 0.15   0.13
bgr          0.03  0.01  2.29  12.24     0.04   0.00  0.07   44 0.61   0.55
sc           4.27  1.59  2.69  14.12     0.02   0.87  7.67   17 0.57   0.51
sod         -0.15  0.09 -1.62  45.86     0.11  -0.34  0.04   87 0.30   0.27
pot         -0.68  0.38 -1.82  45.99     0.08  -1.44  0.07   88 0.30   0.27
hemo        -1.72  0.37 -4.65  30.81     0.00  -2.48 -0.97   52 0.38   0.34
```

Conclusion

The imputation method based on gamlss is a fairly new imputation technique which is provided to properly handle situations in which fully parametric assumptions with respect to the conditional distribution of the variable to be imputed is questionable. The technique is based on the idea of attaining imputations avoiding possibly misspecified imputation models. Salfran (2018), de Jong et al. (2014) and de Jong (2012) showed that this technique outperforms standard imputation techniques in several scenarios. The **ImputeRobust** package is a step forward in the development of an imputation method that requires weak distributional assumptions, but still allows valid inference. It extends the set of applications of the existing GAMLSS package by allowing it to interact with the **mice** package.

By building on the popular **mice** package, the advantages of standard and robust imputation procedures are available in a user friendly way. Further research will focus on improving the efficiency of this semi-parametric imputation technique, requiring only weak assumptions for generating multiple imputations but still allowing valid inferences.

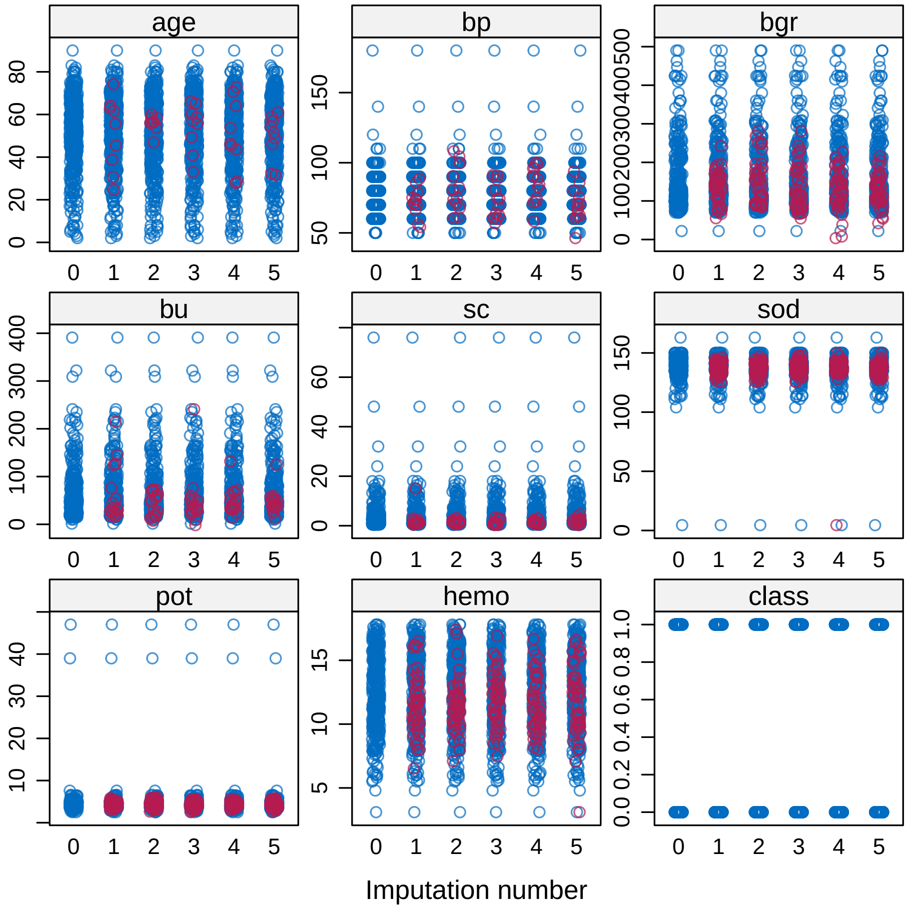

Figure 3: Stripplot of the chronic kidney data set variables with missings and five imputed data sets. Observed data is in blue and imputed data in red.

Acknowledgements

The authors gratefully acknowledge financial support from the German Science Foundation via grant SP 930/8-1.

Bibliography

J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993. ISSN 1537-274X. URL https://doi.org/10.1080/01621459.1993.10476321. [p61]

R. de Jong. *Robust Multiple Imputation*. PhD thesis, Universität Hamburg, 2012. URL http://ediss.sub.uni-hamburg.de/volltexte/2012/5971/. [p61, 67, 69]

R. de Jong, S. van Buuren, and M. Spiess. Multiple imputation of predictor variables using generalized additive models. *Communications in Statistics - Simulation and Computation*, 45(3):968–985, 2014. ISSN 1532-4141. URL https://doi.org/10.1080/03610918.2014.911894. [p61, 66, 69]

P. H. C. Eilers and B. D. Marx. Flexible smoothing with b -splines and penalties. *Statistical Science*, 11 (2):89–121, 1996. ISSN 0883-4237. URL https://doi.org/10.1214/ss/1038425655. [p62]

I. R. Harris. Predictive fit for natural exponential families. *Biometrika*, 76(4):675–684, 1989. ISSN 1464-3510. URL https://doi.org/10.1093/biomet/76.4.675. [p62]

Y. He and T. E. Raghunathan. On the Performance of Sequential Regression Multiple Imputation Methods with Non Normal Error Distributions. *Communications in Statistics - Simulation and Computation*, 38(4):856–883, 2009. ISSN 0361-0918. URL https://doi.org/10.1080/03610910802677191. [p61]

Y. He and T. E. Raghunathan. Multiple imputation using multivariate gh transformations. *Journal of Applied Statistics*, 39(10):2177–2198, 2012. ISSN 0266-4763. URL https://doi.org/10.1080/02664763.2012.702268. [p61]

J. Honaker, G. King, and M. Blackwell. Amelia II: A Program for Missing Data. *Journal of Statistical Software*, 45(7):1–47, 2011. ISSN 1548-7660. URL https://doi.org/10.18637/jss.v045.i07. [p61]

M. Lichman. UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml. [p68]

R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, 2002. ISBN 9780471183860. URL https://doi.org/10.1002/9781119013563. [p61]

J. Liu, A. Gelman, J. Hill, Y.-S. Su, and J. Kropko. On the stationary distribution of iterative imputations. *Biometrika*, 101(1):155–173, 2013. ISSN 1464-3510. URL https://doi.org/10.1093/biomet/ast044. [p61]

R. A. Rigby and D. M. Stasinopoulos. Generalized additive models for location, scale and shape (with discussion). *Journal of the Royal Statistical Society C*, 54(3):507–554, 2005. ISSN 1467-9876. URL https://doi.org/10.1111/j.1467-9876.2005.00510.x. [p61, 62, 68]

D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. John Wiley & Sons, 1987. ISBN 978-0-471-65574-9. URL https://doi.org/10.1002/9780470316696. [p61, 62, 65]

D. B. Rubin. Multiple imputation after 18+ years. *Journal of the American Statistical Association*, 91(434): 473–489, 1996. ISSN 1537-274X. URL https://doi.org/10.1080/01621459.1996.10476908. [p67]

D. Salfran. *Multiple Imputation for Complex Data Sets*. PhD thesis, Universität Hamburg, 2018. URL http://ediss.sub.uni-hamburg.de/volltexte/2018/9058/. [p62, 67, 69]

J. Schafer. *Analysis of Incomplete Multivariate Data*. Chapman & Hall, 1997. ISBN 9781439821862. URL https://doi.org/10.1201/9781439821862. [p61]

D. M. Stasinopoulos and R. A. Rigby. Generalized additive models for location scale and shape (gamlss). *Journal of Statistical Software*, 23(7), 2007. ISSN 1548-7660. URL https://doi.org/10.18637/jss.v023.i07. R package version 5.0-2. [p62, 63, 68]

M. Stasinopoulos, B. R. with contributions from Calliope Akantziliotou, G. Heller, R. Ospina, N. Motpan, F. McElduff, V. Voudouris, M. Djennad, M. Enea, A. Ghalanos, and C. Argyropoulos. *Gamlss.dist: Distributions for Generalized Additive Models for Location Scale and Shape*, 2017. URL https://CRAN.R-project.org/package=gamlss.dist. R package version 5.0-2. [p67]

M. Templ, A. Kowarik, and P. Filzmoser. Iterative stepwise regression imputation using standard and robust methods. *Computational Statistics & Data Analysis*, 55(10):2793–2806, 2011. ISSN 0167-9473. URL https://doi.org/10.1016/j.csda.2011.04.012. [p61]

S. van Buuren. Multiple imputation of discrete and continuous data by fully conditional specification. *Statistical Methods in Medical Research*, 16(3):219–242, 2007. ISSN 1477-0334. URL https://doi.org/10.1177/0962280206074463. [p61]

S. van Buuren. *Flexible Imputation of Missing Data*. Chapman and Hall/CRC, 2012. ISBN 9781439868256. URL https://doi.org/10.1201/b11826. [p61]

S. van Buuren and K. Groothuis-Oudshoorn. Mice: Multivariate imputation by chained equations. *Journal of Statistical Software*, 45(3), 2011. ISSN 1548-7660. URL https://doi.org/10.18637/jss.v045.i03. R package version 2.30. [p61, 62, 65]

Daniel Salfran
Psychological Methods and Statistics
Faculty of Psychology and Movement Science
Universität Hamburg
Von-Melle-Park 5, 20146 Hamburg
Germany
danielsalfran@gmail.com

Martin Spiess
Psychological Methods and Statistics
Faculty of Psychology and Movement Science
Universität Hamburg

Von-Melle-Park 5, 20146 Hamburg
Germany
martin.spiess@uni-hamburg.de