Ludwig-Maximilians University

Master thesis

Imputation of missing data via penalization techniques

Author:
Felix Klug

Supervisor:
Prof. Dr. Christian
Heumann

27th of January 2015

Abstract

The aim of this master thesis is to give the user an estimate of uncertainty over missing data imputation. The full factorization approach is compared to the state-of-the-art approach of full conditional. The special feature in both algorithms is the penalization techniques. Both algorithms are used with different types of missing data like MAR, MCAR and NMAR.
Simulated datasets were conducted with copulas. Simulations were varied in rate of missing observations, refitting times and the use, or not, of LASSO regression for last fit.
Results are given in terms of accuracy of predicted values, pooled variance estimates and errors which occurred during programming and runtime. The full factorization approach showed advantages over full conditional especially if one looks on ratios of 10:1 observations to covariables. In cases were covariables were in higher numbers than observations, full conditional and full factorization nearly covered same results when ridge regression was used for all fits. Generally lasso regression did not improve accuracy of imputation results. This result can be generalized for all missing types used and simulations conducted in this master thesis. Imputed observations showed paths which are similar to MCMC bayesian statistics. The imputation steps alternated and converged to certain values.
Results were stable when different datasets and seeds for random numbers were processed. Run time for both approaches was high due to errors that occurred in different `R` packages and functions which are costly in terms of CPU usage.

Contents

List of Tables

List of Figures

Abbreviations and acronyms

MAR Missing at random

NMAR Non missing at random

MCAR Missing completely at random

LASSO Least absolute shrinkage and selection operator

CV Cross validation

CI Confidence interval

n Number of observations

p Number of covariables

n:p Ratio between number of observations and covariables

NA Non available meaning missing data

AIC Akaike information criterion

BIC Bayesian information criterion

CART Classification And Regression Tree

L1 Penalty term of LASSO regression; Also used as synonym for LASSO regression

L2 Penalty term of ridge regression; Also used as synonym for ridge regression

Chapter 1

Introduction: Missing data in statistics

Missing data is a huge statistical topic, because of the wide variety of situations one has to deal with it. I will give two examples of these situations in the following. Both do have NAs but the underlying types of missing data are different.
For example in political surveys missing data often occur when people want to vote for not politically desired parties. Voters of these parties will then tend to hide their opinions or simply vote for a more accepted party. The problem for the statistical analyst is now that the estimates for these parties are downwards biased. The type of missing data here is non missing at random (NMAR), because data is missing due to the observation itself. Another field in which missing data can cause problems is genomic analysis. Here one has to deal with a huge number of variables p and often a small number of observations n. If the variables p are genomes it is highly likely that missing data could be present because of measurement errors that happened at the experiments undertaken. Mathematically this gives two tasks: The first one is finding a solution for the missing data. This could only be done by using complete cases or imputing the data one time with an algorithm at hand. The second task is fitting estimates in a situation where variables p are in larger numbers than observations n. This problem can be solved with penalized regressions like ridge or LASSO regression or even CART algorithms. The type of missing data is more likely either Missing completely at random (MCAR) if the observations with missing occur independently from each other or Missing at random (MAR), if the missing data are linked with

especially one variable and are dependent on observations of other variables in the data.
Generally this master thesis will concentrate on algorithms which impute missing data. Most common algorithms today try doing this by giving the user solid numbers for the values missing. What most algorithm miss is estimating an uncertainty of these imputations. In the following I will introduce a known and a new approach, the Full conditional and Full factorization, for imputing missing data with uncertainty. The advantage of these procedures is that users are not given a single observation as solution in the algorithm but an entire distribution for each cell missing in the data.
Chapter 2 will lay the mathematical and statical foundations for both algorithms. Chapter 3 will then give an exact overview of the programming for simulations in `R` (R Core Team, 2014). Results and a comparison of both algorithms are presented in Chapter 4.1.

Chapter 2

Theory of new missing data approaches

2.1 General theory of missing data algorithms

This chapter will clarify the mathematical similarities between the full conditional and the full factorization approach. The independent differences will each be presented in separate sections (2.4,2.3) later on.
In general both algorithms are using penalized regressions and bootstrap methods to estimate missing data. The functions know five different regression families for the dependent variable, each of which different statistical models are being used. Following the main regressions are explained with their inner methods before going to the methods and models for the different distributions.
For further insights to the theory of imputing missing values one should have a look at Rubin's article "Inference and missing data" (Rubin, 1976) or (Schafer, 1997).

2.1.1 The LASSO

The least absolute shrinkage and selection operator (LASSO) is a linear model with an L1 penalization term (Tibshirani, 1996):

$$\hat{\beta}^{\text{lasso}} = \arg\min_{\beta} \left\{ \frac{1}{2} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \tag{2.1}$$

The absolute penalization term can shrink beta coefficients to zero so that they are selected from the model. In case of p >> n the LASSO can be used to thin out variables in models and make fitting in the presence of collinearity even possible. The LASSO is used for all families inside the function with a cross validated penalization term. The disadvantage of this method is that it has no analytical solution for the covariance matrix of the beta coefficients, because of the absolute penalization term that can not be differentiated.

2.1.2 Ridge regression

In contrast to lasso regression, the ridge regression shrinks the beta coefficients but is not generally used for variable selection (Tikhonov, 1943):

$$\hat{\beta}^{\text{ridge}} = \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{2.2}$$

The L2 penalization term allows a analytic solution for the covariance of betas (Fahrmeir et al., 2013):

$$\text{Cov}\left(\hat{\beta}_{\text{Ridge}}\right) = \sigma^2 (X^T X + \lambda I)^{-1} X^T X (X^T X + \lambda K)^{-1} \tag{2.3}$$

λ is given as the penalty coefficient calculated by cross-validation. K gives a penalization matrix with diag(0,1,...,1). Variance of ϵ is estimated with σ^2. Ridge regression yields coefficients in cases of collinearity or mathematically unstable generalized linear models.

2.1.3 Bootstrapping

In cases where analytic solutions for covariance of beta coefficients can not be calculated one can use bootstrapping (Hastie et al., 2011). Indices of the original data set are sampled with replacement and the new data matrix is fitted to the original models. This procedure is done R times and the resulting variances and covariances stem from the variance/covariance of the different bootstrap samples for each coefficient. The covariance matrix received should be unbiased.

$$Var(\hat{\beta}_j) = \frac{1}{R-1} \sum_{i=1}^{R} (\widehat{\beta_{ij}^{\text{Boot}}} - \bar{\beta_j^{\text{Boot}}}) \tag{2.4}$$

Bootstrapping can also be used to get unbiased estimates for beta coefficients. Therefore the mean of the Bootstrap samples of betas is used.

$$\hat{\beta}_j = \frac{1}{R} \sum_{i=1}^{R} \widehat{\beta_{ij}^{\text{Boot}}} \tag{2.5}$$

Stratified Bootstrapping

In case of binomial and categorical responses one should not use normal bootstrapping, because of unbalanced classes in the bootstrap sample. If one class is highly over represented regression coefficients are likely to be unstable. Prevention of this can be done by sampling inside of the classes. The advantage of this is that the proportions between classes stay the same (Davison et al., 2003).

2.1.4 Cross-Validation

To estimate the penalization factor λ cross-validation is used. The method splits the data into K different partitions and then uses K-1 partitions as training sets and one partition as a validation set. The validation usually computes a score like AIC, BIC or the cross-validation error.
The algorithm runs through all partitions and computes the statistical score. Finally the λ with the smallest statistical score is chosen. Criterions for statistical scores used in algorithms are discussed later on.

2.2 General form of algorithms

The algorithms start by computing the missing values in every variable one time. These values are then used as the first starting values for the algorithm. The variables are sampled throughout the whole algorithm to prevent finding local maxima or minima. The user can decide on whether to use LASSO regression for last imputation or not. Generally the algorithm runs in six different steps:

1. Cross-Validation to get best λ penalization factor for regression

2. Fit regression with p-1 independent variables
3. Sample variance of ϵ residuals
4. Compute covariance in case of ridge regression or use bootstrapping with lasso penalty
5. Sample beta coefficients with given covariances
6. Predict new observations for imputation; latest predictions are used as starting values for new predictions

In the following sections I will give detailed information about each of theses steps for the regression families at hand. These sections will also present programming in `R`.

2.2.1 First imputation using `mice` algorithm

Users can choose between giving their own starting values for NAs or letting the algorithm choose for themselves. In case of the second one first imputations for data are done by using Fully Conditional Specification (FCS) implemented by the MICE algorithm (van Buuren and Groothuis-Oudshoorn, 2011). This algorithm uses bayesian Gibbs sampling with Chained Equations. The method automatically recognizes regression families and fits imputed data by built-in imputation methods.
If the starting value is too far from the real data point the algorithm will probably not converge in time.

2.2.2 Step 1: Use Cross-validation for optimal λ

As described in 2.1.4 cross-validiation is used to get an optimal regression fit. Every regression family uses different `R` packages and therefore different cross-validation models. The following table will give an overview of functions and packages used for regression families in `R`.

Regression family	`R` function	`R` package
Normal, Binomial & Poisson	`optL1()` & `optL2()`	`penalized` package (Goeman, 2010)
Categorical	`cv.glmnet()`	`glmnet` package (Friedman et al., 2010)
Gamma	`cv.lqa()`	`lqa` package (Ulbricht, 2012)

Table 2.1: Functions and packages for cross-validation in `R`

Each of these functions are given a user configured upper bound for the penalization factor. Per default this is 512. For normal, binomial, poisson and categorical family cross-validation is calculated via 10 data folds. In case of gamma responses a training and validation dataset is drawn from the original data.
Criteria for cross-validation are cross-validated likelihood (normal, binomial and poisson family), misclassification rate (categorical family) and Akaikes information criterion (gamma family).
There can be times when λ is set to zero. In that case different unregularized regression fits are used.

2.2.3 Step 2: Fitting regression model

Regression family	`R` function	`R` package
Normal, Binomial & Poisson ($\lambda \neq 0$)	`penalized()`	`penalized` package
Normal, Binomial & Poisson ($\lambda = 0$)	`glm()`	`stats` package
Categorical	`glmnet()`	`glmnet()` package
Gamma	`lqa()`	`lqa()` package

Table 2.2: Functions and packages for regression fitting in `R`

Table 2.2 shows the functions in `R` that are used for calculating regression fits. As explained before the user can choose between fitting entirely ridge regression and taking LASSO regression for the last fit. The benefit of lasso regression could be that in cases of p >> n unneeded covariables are selected from the regression model. As explained in Step 1 cross-validation can set the penalization parameter to zero leading to an unregularized model fit. This is problematic for the `penalized` function because a penalization factor λ is obligatory. To solve this problem normal generalized linear models are taken if λ is set to zero.

$$y = h(\eta) \tag{2.6}$$

$$\eta = \beta_0 + \sum_{i=1}^{p} x_i \beta_i \tag{2.7}$$

In 2.7 one can see the model formula for a generalized linear model. Link function and response functions are the same as in their regularized equivalents.

Regression family	Link function	Model formula
Normal	Identity	$y = x^T\beta$
Binomial	log Odds	$\log\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = x^T\beta$
Poisson	log	$\log(y) = x^T\beta$
Categorical	Multinomial log Odds	$\log\left(\frac{P(Y=r\|x)}{P(Y=k\|x)}\right) = x^T\beta$
Gamma	log	$\log(y) = x^T\beta$

Table 2.3: Link functions for regression models used

For gamma regression the canonical link function, the inverse function, is not used because this could lead to constraints on the beta coefficients. Also the interpretation of coefficients is much more complicated then usual, because small beta coefficients indicate greater values of dependent variable y. Log link for gamma regression gets rid of constraints on beta coefficients and makes interpretation for users straight forward.

2.2.4 Step 3: Sample variance of residuals ϵ

$$\widehat{\sigma}_{new} = \frac{\widehat{\sigma}}{\text{DF-1}} * \chi_{DF-1} \tag{2.8}$$

To sample the variance of residuals the model variance is divided through the degrees of freedom minus one for that model and multiplied by a random number of a chi-squared distribution with DF-1 degrees of freedom. The degrees of freedom here can be seen as the number of observations. Generally the variance of residuals can be compared to statistic for model fits. A huge variance would indicate a bad model fit and vice versa.

2.2.5 Step 4: Computing covariances for beta coefficients

To sample new beta coefficients the covarianc of estimated betas in the regression model is needed. Ridge regression allows for analytic solutions of these covariances as stated in section 2.1.2. To compute inverses for matrices the general inverse is used. It is implemented in the function `ginv` from the package `MASS`. This functions calculates the Moore-Penrose pseudoinversion of a matrix. The advantage of this method is that even in case of singularities

in matrices the algorithm still functions.
LASSO regression can not be differentiated because of the absolute penalization term. In every scenario bootstrapping is used. For categorical and binomial regression families stratified bootstrap is considered over normal bootstrapping. As stated above the binomial family is problematic considering analytic solutions of covariance matrices. So even in the ridge regression case one has to use bootstrapping for calculation of variances.

2.2.6 Step 5: Sample beta coefficients with given covariances

The resampling of beta coefficients relies on the function `rmvnorm` in the package `mvtnorm` (Genz et al., 2014). This function samples random numbers from a multivariate normal density.

$$f(x,y) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \tag{2.9}$$

$$\times \exp\left\{-\frac{1}{2(1-\rho^2)}\left[\left(\frac{x-\mu_1}{\sigma_1}\right)^2 - 2\rho\left(\frac{x-\mu_1}{\sigma_1}\right)\left(\frac{y-\mu_2}{\sigma_2}\right) + \left(\frac{y-\mu_2}{\sigma_2}\right)^2\right]\right\} \tag{2.10}$$

In 2.10 one can see a density of two-dimensional multivarite normal distribution. Cholesky decomposition is used to calculate inversions of sigma matrix.

2.2.7 Step 6: Predicting new observations for imputation

Predictions are different considering the regression families. In cases of normal, poisson and gamma distribution the algorithm is the same.

Algorithm 1 Predicition of imputed observations

procedure PREDICITION OF IMPUTED OBSERVATIONS
    Min_Var ← Minimum of variable to be imputed
    Max_Var ← Maximum of variable to be imputed
    Min_CI ← Min_Var - Proportion · StandardDev(Var)
    Max_CI ← Max_Var + Proportion · StandardDev(Var)
                                ▷ Proportion is taken to be 0.1 per default
    Minimal ← Are there any observations smaller then CI of Minima?
    Maximal ← Are there any observations greater then CI of Maxima?
▷ Predictions out of models are reviewed for outliers
     ▷ If any are found the observations are resampled with mean of not imputed observations and standard deviations
    if length(Minimal) > 0 or length(Maximal) >0 **then**
        mean_Var ← Mean of not imputed observations in variable
        if length(Minimal) > 0 **then**
            Eta[Minimal] ← Random numbers out of Variable with mean_Var and standard deviation
        end if
        if length(Maximal) > 0 **then**
            Eta[Maximal] ← Random numbers out of Variable with mean_Var and standard deviation
        end if
    end if
end procedure

Re-running regression models can cause problems in terms of extreme outliers accounting for singularity problems. Consider a single outlier in one run of the algorithms. Refitting this outlier could lead to an outlier even more extreme then the last imputed observation causing beta coefficients to become unstabile. To prevent this confidence intervals for minima and maxima of imputations are computed. A proportion of the standard deviation of the variable is added or taken from the minima or maxima. If an outlier is smaller or greater than one of CI boundaries it is resampled with the mean of the not imputed observations and standard deviation.
Outlier prevention in categorical regression is done by avoiding classes to be over represented. If a model predicts only one class for all observations and the number of observation exceeds a proportion of 40 % of all observations

then the prediction is discarded.
Binomial observations are sampled with p to be the mean of not imputed observations in the variable.

2.3 Full conditional approach

The full conditional approach as the name suggests starts with the full conditional of every variable in the dataset. This method does not work properly if the joint distribution does not exist (Drechsler and Rässler, 2008).

$$X_1|X_2,\ldots,X_p \tag{2.11}$$
$$X_2|X_1,\ldots,X_p \tag{2.12}$$
$$\vdots \tag{2.13}$$
$$X_p|X_1,\ldots,X_{p-1} \tag{2.14}$$

Every full conditional can be seen as a model with p-1 independent variables and one dependent variable. Variables are permuted in every refitting step to prevent the algorithm to be stuck on local optima. To give a deeper understanding of the algorithm one can see the algorithm in pseudo code at page 23.

2.4 Full factorization approach

The full factorization approach is based on a joint distribution over all variables p (Garcia et al., 2010). However in every programming step another joint distribution is selected. The program starts with a full model fit of all variables in the dataset and reduces one variable at a time until only one variable is left. Imputation for this variable is done by the marginal distribution.

$$X_1|X_2,X_3,X_4 \tag{2.15}$$
$$X_2|X_3,X_4 \tag{2.16}$$
$$X_3|X_4 \tag{2.17}$$
$$X_4 \tag{2.18}$$

The algorithm is similar to the full conditional approach and should work computationally faster because the regressions are not fitted on the whole

dataset. In every refitting step the joint distribution is permuted with the same reason as in the full conditional approach. The algorithm can be seen on 24.

Algorithm 2 Full conditional approach

procedure FULL CONDITIONAL APPROACH
    Data_Comp ← Compute starting values for missing
textitdata
        ▷ *Starting values are calculated by MICE or directly by the user*
    p ← Length of **covariables** in *data*
    Covariable_List ← **List** with length p
    Times ← Number of times algorithm should repeat
                ▷ *Old imputations X_i are used as new starting values*
    for i in 1 **to** Times **do**
        for j in sample(1 **to** p) **do**
            Covariable_List[j] ← *Regression*$(X_j|X_1,\ldots,X_{j-1},X_{j+1},\ldots,X_p)$
▷ *Ridge/LASSO for metrical, poisson and binomial approaches;* `glmnet` *for categorical approach*
            if L1 **then**
                                ▷ Use LASSO regression in last imputation?
                if i!=Times **then**
                                    ▷ Use ridge regression in first imputations
                    if j is nominal, poisson or gamma **then**
                        $\lambda_2 \leftarrow$ Cross-valdition(data,model)
                        $\widehat{Cov(\widehat{\beta_R})} \leftarrow (X^TX+\lambda K)^{-1}X^TX(X^TX+\lambda K)^{-1}$
                    else ▷ If X_j is binary or categorical use bootstrap instead
                        $\widehat{Cov(\widehat{beta_R})} \leftarrow$ Bootstrap
                    end if
                    $\widehat{\beta}_{New_i} \sim MVN(\widehat{\beta}_i,\text{Cov}(\beta_R))$
                    $X_i \sim N(\widehat{\eta}_{New_i},\widehat{\sigma}^2_{New_i})$
                end if
            else ▷ Use LASSO regression
                $\lambda_1 \leftarrow$ Crossvaldition(data,model)
                $\widehat{Cov(\widehat{beta_R})} \leftarrow$ Bootstrap
                $X_i \sim N(\widehat{\eta}_{New_i},\widehat{\sigma}^2_{New_i})$
            end if
        end for
    end for
end procedure

Algorithm 3 Full factorization approach

procedure FULL CONDITIONAL APPROACH
    Data_Comp ← Compute starting values for missing*data*
        ▷ *Starting values are calculated by MICE or directly by the user*
    p ← Length of **covariables** in *data*
    Covariable_List ← **List** with length p
    Times ← Number of times algorithm should repeat
                ▷ *Old imputations X_i are used as new starting values*
    for i in 1 **to** Times **do**
        for j in sample(1 **to** p) **do**
            Covariable_List[j] ← $Regression(X_j|X_1,\ldots,X_{j-1},X_{j+1},\ldots,X_p)$

            $Regression(X_1|X_2,\ldots,X_{j-1},X_{j+1},\ldots,X_p)$ ▷ *Ridge/LASSO for metrical, poisson and binomial approaches;* `glmmet` *for categorical approach*
            if L1 **then**
                        ▷ Use LASSO regression in last imputation?
                if i!=Times **then**
                            ▷ Use ridge regression in first imputations
                    if j is nominal, poisson or gamma **then**
                        $\lambda_2 \leftarrow$ Cross-valdition(data,model)
                        $\widehat{Cov(\widehat{\beta_R})} \leftarrow (X^TX+\lambda K)^{-1}X^TX(X^TX+\lambda K)^{-1}$
                    else ▷ If X_j is binary or categorical use bootstrap instead
                        $\widehat{Cov(\widehat{beta_R})} \leftarrow$ Bootstrap
                    end if
                    $\widehat{\beta}_{New_i} \sim MVN(\widehat{\beta}_i, \text{Cov}(\beta_R))$
                    $X_i \sim N(\widehat{\eta}_{New_i}, \widehat{\sigma}^2_{New_i})$
                end if
            else ▷ Use LASSO regression
                $\lambda_1 \leftarrow$ Cross-valdition(data,model)
                $\widehat{Cov(\widehat{beta_R})} \leftarrow$ Bootstrap
                $X_i \sim N(\widehat{\eta}_{New_i}, \widehat{\sigma}^2_{New_i})$
            end if
        end for
    end for
end procedure

Chapter 3

Simulation

3.1 Introduction to simulation

To compare approaches a simulation was conducted. Datasets are simulated with different missing values, variable numbers and times the algorithms reruns the regression fits. This chapter will give an overview of the simulating topics beginning with the algorithm used for simulations of the data sets. The functions for missing values and variety of simulating the approaches are explained later on.

3.1.1 Simulating datasets

To simulate datasets copulas are used to retrieve datasets with correlation structures. Datasets simulated here have an exchangeable structure. This means the correlation between all variables stays the same.

$$Corr(X) = \begin{pmatrix} 1 & \alpha & \alpha & \dots & \alpha \\ \alpha & 1 & \alpha & \dots & \alpha \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ \alpha & \alpha & \alpha & \dots & 1 \end{pmatrix} \tag{3.1}$$

Output of copulas are uniform distributions, which give the marginal distribution of each variable. One has to transform these variables by quantile functions to get the actuall distributions. The `copula` (Hofert et al., 2014) (Jun Yan, 2007) (Ivan Kojadinovic and Jun Yan, 2010) (Marius Hofert and Martin Mächler, 2011) package supports various modes which can be used for

simulating marginal distributions. In this case the `normalCopula` function is used which is based on a normal distribution.

Algorithm 4 Simulating datasets

```
1: procedure SIMULATING DATASETS(n,distrib,means,sd,strength=0.2,seed=1234)
2:     set.seed(seed)
3:                                   ▷ A seed is set to ensure reproducibility
4:     dim ← Length of distrib
5:       ▷ Dimensions of new dataset are evaluated of the input parameters
6:     Copula ← Copula(n=n,copula=normalCopula(strength,dim=dim))
7:   ▷ Copula is evaluated with a normalCopula random number generator
   with exchangeable structure for correlations
8:     for j in 1:dim do
9:  ▷ The uniform distributions are transformed with quantile functions in
   R, means and standard deviations as input parameters
10:         if Distribution == "normal" then
11:             Variable ← qnorm(p=Copula,means=means,sd=sd)
12:         else
13:             if Distribution == "binomial" then
14:                 Variable ← qbinom(p=Copula,size=1,prob=means)
15:                     ▷ A coin is flipped one time with probability of means
16:             else
17:                 if Distribution == "poisson" then
18:                     Variable ← qpois(p=Copula,lambda=means)
19:                 else
20:                     if Distribution == "gamma" then
21:                         Scale ← (Standard deviation)^2/Means
22:                         Shape ← (Means)^2/Standard deviation
23:                         Variable ← qgamma(p=Copula,scale=Scale,shape=Shape
24:                     else
```

The probabilities for a categorical variable are data driven by proportion of observations between certain boundaries. The strength of the exchangeable structure is set to 0.2 by default.

```
25:                     if Distribution == "categorical" then        ▷ The
   categorical distribution chooses on probability of different classes on its on
26:         ▷ Boundaries are set by number of different classes for categorical
   variables
27:                         Dsort ← sort(Copula)
28:                                                  ▷ Copula is sorted by values
29:                         Bound ← seq(0,1,by=1/means
30:             ▷ Boundaries are set uniformly over range from zero to one
31:                         for i in 1:length(Bound)-1 do
32:                             PropClass ← Proportion of observations be-
   tween Boundary i and Boundary i-1
33:                         end for
34:                         Variable ← sam-
   ple(1:mean,size=n,replace=TRUE,prob=PropClass)
35:                     end if
36:                 end if
37:             end if
38:         end if
39:     end if
40: end for
41: end procedure
```

3.1.2 Simulating missing values with MCAR

Simulating missing values with underlying missing completely at random is done by theoretically flipping a coin for every observation and set this observation to missing in case of success.

Algorithm 5 Simulating missing values MCAR

```
1: procedure MISSING VALUES(data,missing=0.1,seed=1234)
2:     set.seed(seed)
3:                                          ▷ Set seed for reproducibility
4:     row ← Number of rows in dataset
5:     col ← Number of columns in dataset
6:     for j in 1:row do
7:         for i in 1:col do
8:             na ← Coin flip with probability of success set by missing
9:             if Coin flip is succesfull then
10:                data[i,j] ← NA
11:            end if
12:        end for
13:    end for
14: end procedure
```

3.1.3 Simulating missing values with NMAR

NMAR (non missing at random) is a missing mechanism where missing values are dependent on observations of the variable itself. Consider scaling people. If a person is overweighted then it could happen that scaling fails due to limits of measurement. The observation is missing because of the observation itself. To simulate this structure quantiles of every metrical variable are taken. Per default these are the 5 and 95 % quantiles. So very small and very large observations are taken to be missing. In case of binomial or categorical variables MCAR is used with a default missing rate of 0.1 . The algorithm can be seen in pseudo code on page 29.

Algorithm 6 Simulating missing values NMAR

```
1: procedure MISSING VALUES(data,quantile=(0.05,0.95),missing=0.1,seed=1234)
2:     set.seed(seed)
3:                                          ▷ Set seed for reproducibility
4:     col ← Number of columns in dataset
5:     for i in 1:col do
6:         if Length of unique observations of variable are larger then categories of categorical variable then
7:   ▷ The if clause is necessary to differ between metrical and binomial or categorical variable
8:             Upper_quantile ← 0.95 % quantile of variable i
9:             Lower_quantile ← 0.05 % quantile of variable i
10:                ▷ Here default quantiles are taken; User can change these
11:            Data which are greater then Upper_quantile are set to NA
12:            Data which are smaller then Lower_quantile are set to NA
13:        else
14:                ▷ MCAR is used for binomial and categorical variables
15:            row ← Number of rows in dataset
16:            for j in 1:row do
17:                na ← Coin flip with probability of success set by missing
18:                if Coin flip is successfull then
19:                    data[i,j] ← NA
20:                end if
21:            end for
22:        end if
23:    end for
24: end procedure
```

3.1.4 Simulating missing values with MAR

The third missing mechanism is one where values do not depend on the observations themselves but are dependent on observations of other variables. Take the example of scaling people of the NMAR section. Now consider that before scaling was done anyone in the population was undergoing several medical tests. Within these tests blood pressure, cholesterol level and other medical properties are measured. If a value at scaling is now missing because measurement fails, this values depends on the other values of observations.

To get a proper algorithm working quantiles of variables like in the NMAR algorithm were taken as boundaries for missing values. The difference of NMAR and MAR algorithm is now that missing values are not implemented into variables itself but into other variables of dataset. Quantiles of categorical and binomial variables do not make sense so in these case simple MCAR was used with a missing success rate of 0.2 . The algorithm takes only one variable at a time to create missing values.

Algorithm 7 Simulating missing values MAR

procedure MISSING VALUES(data,quantile=(0.05,0.95),missing=0.1,seed=1234)
    set.seed(seed)
                                ▷ *Set seed for reproducibility*
    col ← Number of columns in dataset
    Var ← 1:ncol
    ▷ *Var vector is set as the independent variable of the MAR algorithm*
    for i in 1:col **do**
        if Length of unique observations of variable are larger then categories of categorical variable **then**
    ▷ *The if clause is necessary to differ between metrical and binomial or categorical variable*
            Independent ← Sampled from numbers of Var
            while Independent == i **do**
                Independent ← Sampled from numbers of Var
            end while
                        ▷ *Independent variable can not be variable itself*
            Upper_quantile ← 0.95 % quantile of variable i
            Lower_quantile ← 0.05 % quantile of variable i
                ▷ *Here default quantiles are taken; User can change these*
            Observations in independent variable which are greater then Upper_quantile are set to NA

3.1.5 Variations over simulating steps

To evaluate and compare the different approaches a small variety of simulations are undertaken. These simulations can be generalized for larger populations due to proportions of observations and covariables in the dataset.

```
        Observations in independent variable which are smaller then
  Lower_quantile are set to NA
            Var ← Var without used independent variable
        else
                    ▷ MCAR is used for binomial and categorical variables
            row ← Number of rows in dataset
            for j in 1:row do
                na ← Coin flip with probabiilty of success set by missing
                if Coin flip is succesfull then
                    data[i,j] ← NA
                end if
            end for
            Var ← Var without used independent variable
        end if
    end for
end procedure
```

For all missing structures, datasets are simulated with a ratio of 10:1 and 2:3 between Observations and Variables. These proportions were chosen to set boundaries in cases were penalization techniques are not required and cases where they are crucial to have. For MAR mechanism a proportion of 3:2 is also examined. The missing value probability is set to 0.2 / 0.4 .
Simulated algorithms are for example compared by paths of their simulating steps and also the density of their simulating results in total. The distribution families are equally split over the number of covariables p. Mean and standard deviation are different for each variable in a distribution family and depend on the number of covariable p the user has selected. For distributions where standard deviations are not needed for simulating process none are reported. In case of binomial distribution, mean gives the probability of success. For categorical distributions, mean is always 3 and gives the number of classes for the variable. 3 was picked because run time grew relevantly in case of greater class numbers.

	Mean		**Standard deviation**	
Distribution	Upper boundary	Lower boundary	Upper boundary	Lower boundary
Gamma	20	2	10	2
Normal	20	1	10	1
Binomial	0.7	0.3	-	-
Poisson	20	3	-	-
Categorical	3	3	-	-

Table 3.1: Mean and standard deviations for each distribution in simulating process

To give an example of simulating distributions consider a dataset of dimension 50 for p and 100 for n. To split the distributions equally over the dataset each distribution is simulated 10 times. The gamma distribution starts with a mean of 2 and a standard deviation of 2 and so on.

Mean	2	4	6	...	20
Standard deviation	2	2.88	3.77	...	10

Table 3.2: Example of 10 gamma distributions for simulating process

Chapter 4

Results of simulation

This chapter is going to focus on giving statistical results for the simulations undertaken. The three different mechanisms for missing values are being compared and also plots for the simulating steps are going to be depictured. Plots presented in this chapter were all done with `ggplot2` (Wickham, 2009).

4.1 Comparison of statistical accuracy of predictions

The main goal of the featured algorithm is to predict uncertainty. But one should not forget the goodness of prediction in missing values. To compare this prediction the mean and standard deviation for every variable is taken and a 95 % confidence interval is calculated with the 95 % quantile of the normal distribution. The number of true values of observations is counted to give a proportion of goodness of prediction for metrical variables.

$$\widehat{\theta} \pm z_{1-\alpha} \cdot \text{ std} \tag{4.1}$$

In case of binomial and categorical variables the number of right predictions is counted and divided by the number of all predictions to give a ratio of true predicted observations. If that number exceeds 0.5 the prediction is taken to be correct for the algorithm at hand. Number of observations for all scenarios is always 100. This section will only give a small overview of the results for all three missing value states. The complete tables with all results can be seen in the appendix following page ii. If observations are missing at the

tables the algorithm was aborted due to errors which are explained later on. The predicted numbers are based on certain simulated datasets. This means numbers will vary if one redoes the simulations.

4.1.1 Tables for accuracy of predicted data

The following table will give first impressions on the differences between missing values structures. Models here were fitted with 10 covariables and 100 observations. Quantiles for MAR and NMAR are 0.05 and 0.95. Proportion for missing values at MCAR was 0.2 . LASSO fitting for last fit was not used for these models.

Algorithm	Missing value structure	Refitting Times	Percentage of accurate predictions
Full factorization	MCAR	20	38.46 %
Full conditional	MCAR	20	31.25 %
Full factorization	MCAR	50	37.50 %
Full conditional	MCAR	50	32.69 %
Full factorization	MAR	20	38.23 %
Full conditional	MAR	20	35.29 %
Full factorization	MAR	50	41.17 %
Full conditional	MAR	50	35.29 %
Full factorization	NMAR	20	10.48 %
Full conditional	NMAR	20	09.79 %
Full factorization	NMAR	50	09.79 %
Full conditional	NMAR	50	09.79 %

Table 4.1: Comparison of accurate predictions over missing value structures with no LASSO fitting and n >> p

In nearly all models fitted the percentage of accurate predictions in full factorization is greater than in the full conditional approach. Smallest values are observed with NMAR, which is not surprising due to the fact that missing values are only dependent on themselves and can not be easily predicted with regression models. MCAR and MAR do have comparable values with MAR accurate predictions slightly smaller then MCAR. If one looks at the refitting times values decrease in MCAR structure while they stay the same or increase in MAR structure.
The advantages of penalized regressions like LASSO or ridge regression clearly are the p >> n case. The next table will give an overview of these cases with ratios 2:3 observations to covariables and LASSO fitting which in these cases was used.

Algorithm	Missing value structure	Refitting Times	Percentage of accurate predictions
Full factorization	MCAR	20	21.12 %
Full conditional	MCAR	20	49.71 %
Full factorization	MCAR	50	31.70 %
Full conditional	MCAR	50	49.75 %
Full factorization	MAR	20	27.40 %
Full conditional	MAR	20	50.22 %
Full factorization	MAR	50	36.83 %
Full conditional	MAR	50	51.57 %
Full factorization	NMAR	20	14.98 %
Full conditional	NMAR	20	17.38 %
Full factorization	NMAR	50	16.59 %
Full conditional	NMAR	50	17.85 %

Table 4.2: Comparison of accurate predictions over missing value structures with LASSO fitting and n << p

The general results of the n >> p are contradicted here. Generally the full conditional approach fits better then the full factorization. A reason for this behavior in case of MAR is that the dependent variable was used before the missing values for the independent variable were fitted. So the dependency structure can not be refitted. In case of full conditional with penalized approaches the important variables are selected from the dataset. Comparing lasso step in the last fitted model one will give an overview in tabular 4.3 of some exemplary models fitted in MCAR. The results can be generalized for all missing mechanism (see also appendix following ii).

Algorithm	Missing value structure	Refitting Times	p	Use L1 penalty?	Percentage of accurate predictions
Full factorization	MCAR	20	10	No	38.46 %
Full conditional	MCAR	20	10	No	31.25 %
Full factorization	MCAR	20	10	Yes	33.17 %
Full conditional	MCAR	20	10	Yes	30.28 %
Full factorization	MCAR	20	150	No	50.04 %
Full conditional	MCAR	20	150	No	49.38 %
Full factorization	MCAR	20	150	Yes	21.12 %
Full conditional	MCAR	20	150	Yes	49.71 %

Table 4.3: Comparison of accurate predictions using L1 penalty and different numbers of covariables

When covariable and observation numbers are in "good" proportion of 1:10, lasso regression should not be used. The accuracy of predictions is

decreasing because of bias in the linear predictor. Also these cases would normally not need penalized regression fits. Looking at huge covariable numbers and small observations numbers like in the 150:100 lasso regression is obsolete as well. In the full factorization approaches, numbers even decrease when using lasso models for the last refit. Ridge regression scenarios were in these cases more stable.

4.1.2 Accuracy of predictions within distributions

To show wich distributions are responsible for the accurate prediction rates a table is presented. It holds numbers of every distributions and the percentage to which these predictions are accurate in the sense of chapter 4.1. The numbers presented are as always dependent on the simulated datasets and model fits. Dimension of data sets was 100 observations with 10 covariables. Fitting was done by a MAR mechanism with 0.95 and 0.05 upper and lower quantile and no use of lasso at the 50ths refit.

	Accuracy of predictions in percent	
Distributions	Full conditional	Full factorization
Normal	33.33 %	44.44 %
Poisson	95.00 %	95.00 %
Gamma	28.00 %	44.44 %
Binomial	80.00 %	80.00 %
Categorical	30.76 %	33.33 %

Table 4.4: Accuracy of predictions within distributions for p << n

The numbers for full factorization exceed the numbers for full conditional. The point of interest clearly lies on the numbers for the poisson distributions. Nearly all of these observations are predicted accurate, means within the CI. On the contrary the numbers for the categorical and gamma distributions are the lowest. Conclusion is that variance within these predictions has to be very high. Poisson and binomial distribution show similar results for both approaches.

4.1.3 Comparison of fitting on different datasets

To verify that results were not only random artifacts, the whole process of model fitting was redone with different datasets and seed for random number generation being used. Models compared in table 4.5 were fitted with 20 refits, no usage of lasso regression, 0.2 missing rate in case of MCAR and 0.05 and 0.95 quantile boundaries for NMAR/MAR. Accuracy numbers are given in percent.

		seed=1000		seed=1234	
		Full factorization	Full condtional	Full factorization	Full conditional
MAR	p=10	35.91	30.28	38.23	35.29
	p=150	48.07	49.02	51.12	50.35
NMAR	p=10	12.5	11.18	10.48	09.79
	p=150	18.73	18.87	16.77	17.49
MCAR	p=10	36.89	31.01	38.46	31.25
	p=150	50.69	50.79	50.04	49.38

Table 4.5: Comparison of accuracy results in different seeds used for data simulation

Results for accuracy seem to be stable over all missing structures and different covariables modelled.

4.1.4 Analysis of variances within distributions

Dimension of 10:1

To quantify how variances differ between full factorization and full conditional approaches pooled variances are are used. Pooled variances are often used in statistical testing (Toutenburg et al., 2008).

$$S^2 = \frac{\sum_{i=1}^{n}(n_i - 1)s_i^2}{\sum_{i=1}^{n}(n_i - 1)} \tag{4.2}$$

In the following table different, but comparable, models of every missing structure are shown with their pooled variances in all distributions except categorical. For this distributions it is uncertain how variances should be calculated. All distributions shown in table 4.7 were fitted without using lasso and 50 refits. For MAR and NMAR upper quantiles and lower quantiles 0.95 and 0.05 were taken and 0.2 missing success probability for MCAR. Results

for simulations using lasso for last refit can be seen following page vii in the appendix.

	MAR		MCAR		NMAR	
	Full conditional	Full factorization	Full conditional	Full factorization	Full conditional	Full factorization
Normal	4.13	10.42	1.69	4.82	1.36	7.17
Gamma	33.45	74.39	25.01	79.33	14.83	30.79
Poisson	21.15	21.99	18.29	18.57	7.96	9.51
Binomial	1.80	5.41	1.478	3.51	0.88	5.02

Table 4.6: Table of pooled variances for ratios 10:1

Pooled variances are for all missing mechanism greater at full factorization. This is not surprising because accuracy in predictions was better for ratios 10:1 in full factorization than in full conditional. Figure 4.1 illustrates the confidence intervals for the three metrical distribution at an example of a MAR model fit. Model was fitted with 50 refits no lasso using and upper and lower quantile of 0.95 and 0.05 .

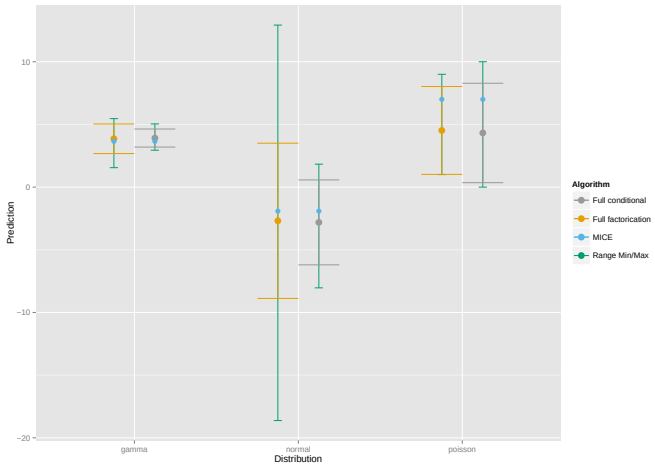


Figure 4.1: CI plot for 10:1 ratio at MAR fit

Two out of three confidence intervals are greater for full factorization then full conditional. The plot also shows starting values for both algorithms and the minimum/maximum range of imputed values. All confidence intervals are smaller then the underlying imputations. Also one can see that algorithms are dependent on starting values in this case MICE imputations.

Dimension of 2:3

The following table will show pooled variances and CI interval plots for same simulations as before but dimension proportions of 2:3 .

	MAR		MCAR		NMAR	
	Full conditional	Full factorization	Full conditional	Full factorization	Full conditional	Full factorization
Normal	9.82	10.45	9.44	10.09	6.44	6.38
Gamma	118.14	119.15	129.99	134.12	72.17	76.42
Poisson	18.06	17.92	16.93	16.66	10.41	10.33
Binomial	8.97	9.85	10.51	10.59	8.87	9.18

Table 4.7: Table of pooled variances for proportion 2:3

Variance in MAR and MCAR are greater then in NMAR. This is not surprising because imputed values in a NMAR structure will be much more uniformly to the data at hand. Variances in full factorization exceed full conditional as in 10:1 proportions.

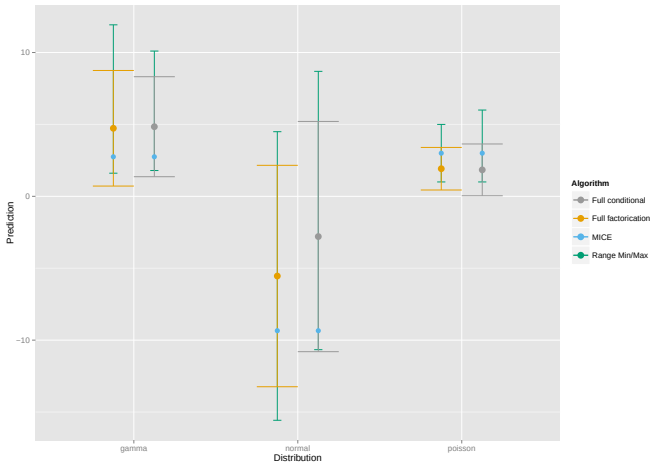


Figure 4.2: CI plot for 2:3 proportion at MAR fit

The CI plot shows quite different results for a 2:3 dimension than in 10:1 . For instance the confidence intervals are in most of the cases greater then range between minimum and maximum. An interesting observation for normal distribution is that mean prediction for full factorization is closer to MICE imputation then the full conditional approach. Gamma imputations confidence intervals have grown relevantly to 10:1 ratios.

Comparison of pooled variances between different simulations

Pooled variances can be compared to each other if underlying datasets are the same, which in this case is true. Pooled variances for a MAR simulation with 50 refits and quantiles of 0.95 and 0.05 were taken for comparison.

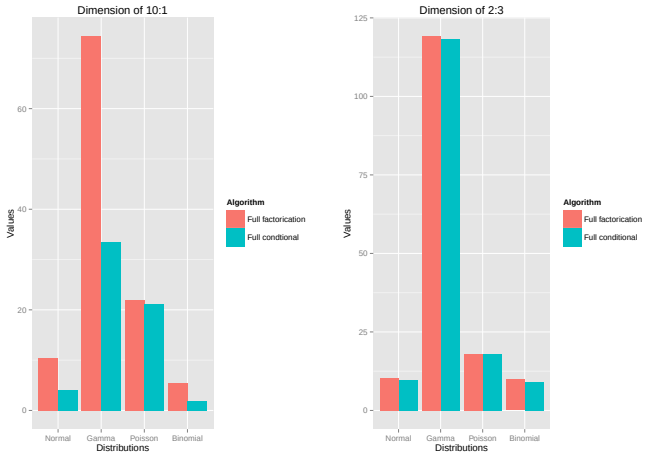


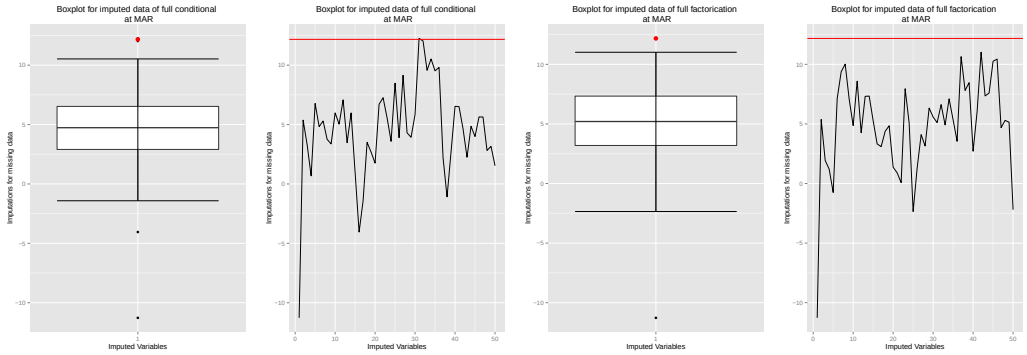
Figure 4.3: Pooled variances for 50 refits and no usuage of lasso fitting

Big differences between both approaches can be seen for normal and gamma distribution. The 10:1 dimension plots seems to be interessting because variances for 2:3 are nearly the same. Big differences in 10:1 arise if looking at normal and gamma distribution. Here the full factorization approach covers much more variance then the full conditional approach. The comparison of lasso models can be found following page vii.

4.1.5 Paths, histograms and boxplots for imputed data

This section will focus on the paths and distributions of the imputed data. Because of the wide variety of models fitted every picture will only be exemplary. The pictures shown here will focus on a MAR mechanism fitted with 50 refits and upper and lower quantiles of 0.05 and 0.95 for missing structure. Because graphs are looking similar for metrical variables the normal distribution is used. Plots for gamma and poisson distribution can be found in the appendix on page vi. In case of binomial and categorical variables only histograms are shown between the real data and the last refitted imputations.

Normal distribution



(a) Full conditional approach

(b) Full factorization approach

Figure 4.4: Comparison of Paths and Boxplots for a normal variable

The paths for both algorithm are similar to MCMC paths for bayesian statistics. The values alternate round a mean value after a specific amount of time. In this case both approaches seem to have missed the real data point but are converging round the same spot. Overall the distributions for the imputed variables are in both approaches overestimated as can be seen in figure 4.5.

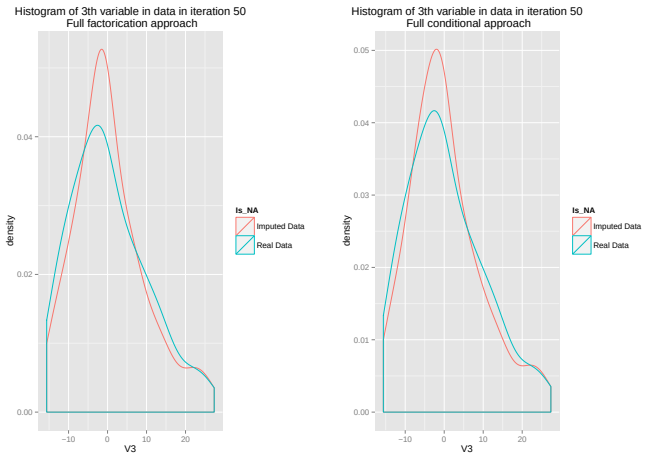


Figure 4.5: Histogram for normal variable

Binomial distribution

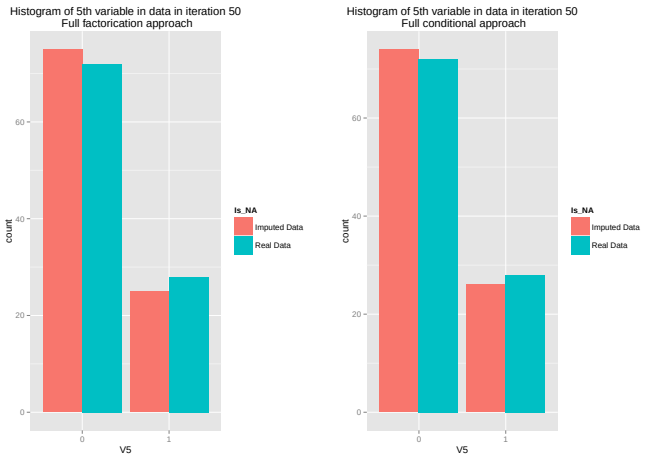


Figure 4.6: Histogram for binomial variable

This examples shows that generally both algorithms are very close to the real data. There are only minor differences between approaches regarding the goodness of prediction here.

Categorical distribution

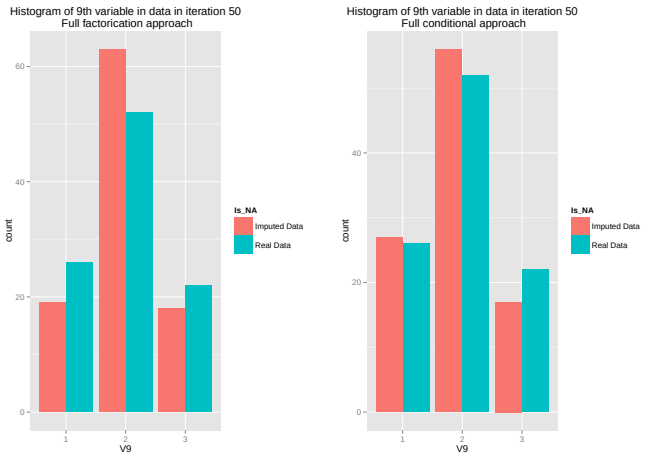


Figure 4.7: Histogram for categorical variable

Results for the categorical variable are similar to the binomial variable. Both variables are close to the real data. However this time the full factorization approach overestimates the second class. The full conditional approach seems to be closer to real data.

4.1.6 Problems with implementation of algorithms

In the implementation phase of `R` programming several problems did arise. These problems were based on different `R` packages and on the algorithms themselves. This chapter will give a short overlook over some of the problems and the solutions which were implemented with the function `tryCatch` in `R`.

Cross validation failes

Before fitting regression models, the best penalization penalty was determined using cross-validation. In some scenarios the algorithm failed or gave a penalization factor of infinity. To prevent from this behavior the user can give boundaries in which the algorithm searches for the best penalty factor. Also when cross validation failed completely observations were resampled in case of metrical distributions or a observation was flipped from zero to one in case of binomial distribution.

Predictions of classes result in abortion of algorithms

Regression fits for categorical variables do rely on uniformly distributed classes in the variable. If one variable is totally over represented beta coefficients tend to get unstable. This was a huge problem for bootstrapping with binomial and categorical variables. Because of the sampling with replacement at bootstrapping dependent variables could include only one class or one class was left out. To prevent this behavior indices for data were sampled in class groups so that proportion of classes were not changed. Also if the algorithm tried to predict classes with one class over 0.4 of the whole prediction, meaning one class highly overrepresented the prediction was discarded and the algorithm continued.

Mysterious error in full factorization approach results in huge time loss

Bootstrapping at full factorization lead to a mysterious error. Covariables at bootstrapping alternated even if the base model fit was correct. Until now the origin of this error is unknown. A first try was to simple redo the procedure and in mostly all times the error was solved. In some times even that resulted in an abortion of the algorithm. The main reason for this error could be the unknown source code of several `R` packages. If this project should continue all of the programming code should be rewritten to prevent unknown errors. Also a different program language should be chosen like `C` or `C++` to boost the algorithm in runtime. The described error lead to a time miss management at full factorization. In theory the full conditional approach should be much slower then the full factorization approach. Because of this error the results were vice-versa, which can be seen in figure 4.8. Here an example of 100 observations was fitted with covariable numbers p 10,20 and 30.

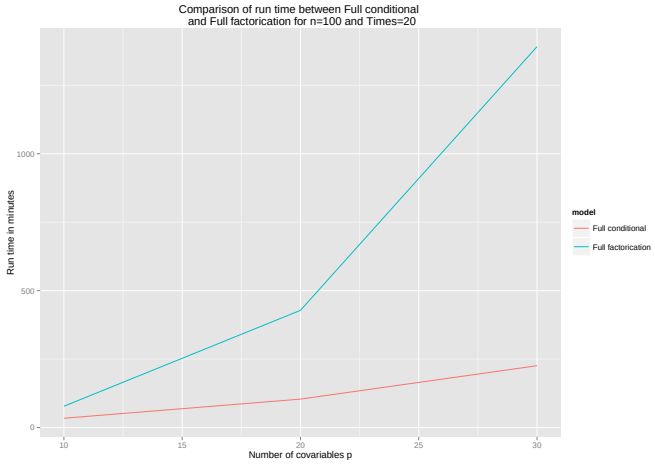


Figure 4.8: User Time in minutes needed to fit algorithms

Chapter 5

Summary and perspective

In this thesis two algorithms were implemented and compared with each other. Both algorithm rely on refitting penalized regression models. Datasets and algorithms for the three main missing data structures were simulated. The result section showed that the full factorization approach has advantages to the full conditional approach. Missing observations are better predicted in case of n >> p and nearly all mechanisms of missing data. In these model cases using lasso regression for last refit did not improve the accuracy of predictions. This maybe due to the bias a penalized regression model requires. If datasets have "good" proportions like 10:1 (Observations/Covariables) penalized regressions are not needed and therefore do not improve model fits. Simulation results also covered the opposite proportions in which covariables are in greater numbers then observations. The results here showed that full factorization gave nearly the same accuracies than full conditional if lasso regressions was not used for last refits. Evidence gives reason that biases in lasso regression lead to non accurate imputed missing values.
Pooled variances for single distributions displayed for full factorization approaches larger confidence intervals in cases of 10:1 proportions, which is a reasonable assumption for the better accuracy in full factorization. If one looks at the missing mechanisms implemented in this thesis MCAR and MAR did have comparable results. Only in the case of NMAR predictions could not approximate missing observations. This behavior is not surprising because missing data values at NMAR are missing because of the observations themselves, meaning that either the observations were very large or very small.
Both algorithms did have problems with errors in `R` packages. Both programs should be implemented in languages that run faster like `C` or `C++`. The refit-

ting process showed that both algorithms were in need of special predicting functions because of extreme outliers.
The full factorization approach seems to have potential to take a further look in it.

Bibliography

Davison, A. C., Hinkley, D. V., and Young, G. A. (2003). Recent developments in bootstrap methodology. *Statistical Science*, 18(2):141–157.

Drechsler, J. and Rässler, S. (2008). *Does convergence really matter?* Number 15. Physica-Verlag, Heidelberg. Shalabh and C. Heumann (Hrsg.), Recent advances in linear models and related areas. Essays in honour of Helge Toutenburg, (Statistical theory and methods, 15).

Fahrmeir, L., Kneib, T., Lang, S., and Marx, B. (2013). *Regression: Models, Methods and Applications*. Springer Series in Statistics. Springer.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Garcia, R., Ibrahim, J., and Zhu, H. (2010). Variable selection for regression models with missing data. *Stat Sin*, 20(1):149–65.

Genz, A., Bretz, F., Miwa, T., Mi, X., Leisch, F., Scheipl, F., and Hothorn, T. (2014). *mvtnorm: Multivariate Normal and t Distributions*. R package version 1.0-0.

Goeman, J. J. (2010). *Penalized R package*. Version 0.9-42.

Hastie, T., Tibshirani, R., and Friedman, J. (2011). *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Hofert, M., Kojadinovic, I., Maechler, M., and Yan, J. (2014). *copula: Multivariate Dependence with Copulas*. R package version 0.999-12.

Ivan Kojadinovic and Jun Yan (2010). Modeling multivariate distributions with continuous margins using the copula R package. *Journal of Statistical Software*, 34(9):1–20.

Jun Yan (2007). Enjoy the joy of copulas: With a package copula. *Journal of Statistical Software*, 21(4):1–21.

Little, R. J. A. and Rubin, D. B. (1986). *Statistical Analysis with Missing Data*. John Wiley & Sons, Inc., New York, NY, USA.

Marius Hofert and Martin Mächler (2011). Nested archimedean copulas meet R: The nacopula package. *Journal of Statistical Software*, 39(9):1–20.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63:581–590.

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

Schafer, J. L. (1999). Multiple imputation: a primer. *Stat Methods Med Res*, 8:3–15.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288.

Tikhonov, A. N. (1943). On the stability of inverse problems [english]. *Comptes Rendus (Doklady) de l'Academie des Sciences de l'URSS*.

Toutenburg, H., Heumann, C., Schomaker, M., and Wissmann, M. (2008). *Induktive Statistik*. Springer-Lehrbuch. Springer.

Ulbricht, J. (2012). *lqa: Penalized Likelihood Inference for GLMs*. R package version 1.0-3.

van Buuren, S. and Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(3):1–67.

Wickham, H. (2009). *ggplot2: elegant graphics for data analysis*. Springer New York.

Chapter 6

Appendix

6.1 MCAR

Algorithm	p	Refitting times	Proportion of missing values	Use L1 penalty?	Percentage of accurate predictions
Full factorization	10	20	0.2	No	38.46 %
Full conditional	10	20	0.2	No	31.25 %
Full factorization	10	50	0.2	No	37.5 %
Full conditional	10	50	0.2	No	32.69 %
Full factorization	10	20	0.2	Yes	33.17 &
Full conditional	10	20	0.2	Yes	30.28 %
Full factorization	10	50	0.2	Yes	33.65 &
Full conditional	10	50	0.2	Yes	32.21 %
Full factorization	10	20	0.4	No	35.96 %
Full conditional	10	20	0.4	No	32.51 %
Full factorization	10	50	0.4	No	- %
Full conditional	10	50	0.4	No	- %
Full factorization	10	20	0.4	Yes	50.73 %
Full conditional	10	20	0.4	Yes	53.44 %
Full factorization	150	20	0.2	Yes	21.12 %
Full conditional	150	20	0.2	Yes	49.71 %
Full factorization	150	50	0.2	Yes	31.70 %
Full conditional	150	50	0.2	Yes	49.75 %
Full factorization	150	20	0.2	No	50.04 %
Full conditional	150	20	0.2	No	49.38 %
Full factorization	150	50	0.2	No	50.34 %
Full conditional	150	50	0.2	No	49.98 %
Full factorization	150	50	0.4	No	49.90 %
Full conditional	150	50	0.4	No	49.58 %
Full factorization	150	50	0.4	Yes	23.21 %
Full conditional	150	50	0.4	Yes	49.60 %

Table 6.1: Goodness of prediction for MCAR

6.2 MAR

Algorithm	p	Refitting times	Quantiles used for missing values: Lower quantile	Upper quantile	Use L1 penalty?	Percentage of accurate predictions
Full factorization	10	20	0.05	0.95	No	38.23 %
Full conditional	10	20	0.05	0.95	No	35.29 %
Full factorization	10	50	0.05	0.95	No	41.17 %
Full conditional	10	50	0.05	0.95	No	35.29 %
Full factorization	10	20	0.05	0.95	Yes	34.55 &
Full conditional	10	20	0.05	0.95	Yes	35.29 %
Full factorization	10	50	0.05	0.95	Yes	33.08&
Full conditional	10	50	0.05	0.95	Yes	36.76 %
Full factorization	10	20	0.1	0.9	No	44.44 %
Full conditional	10	20	0.1	0.9	No	40.90 %
Full factorization	10	50	0.1	0.9	No	44.94 %
Full conditional	10	50	0.1	0.9	No	39.39 %
Full factorization	10	20	0.1	0.9	Yes	44.94 %
Full conditional	10	20	0.1	0.9	Yes	38.38 %
Full factorization	10	50	0.1	0.9	Yes	41.41 %
Full conditional	10	50	0.1	0.9	Yes	39.89 %
Full factorization	150	20	0.05	0.95	Yes	27.40 %
Full conditional	150	20	0.05	0.95	Yes	50.22 %
Full factorization	150	50	0.05	0.95	Yes	36.83%
Full conditional	150	50	0.05	0.95	Yes	51.57%
Full factorization	150	20	0.05	0.95	No	51.12 %
Full conditional	150	20	0.05	0.95	No	50.35 %
Full factorization	150	50	0.05	0.95	No	52.29 %
Full conditional	150	50	0.05	0.95	No	51.52 %
Full factorization	150	50	0.1	0.9	No	51.19 %
Full conditional	150	50	0.1	0.9	No	51.58 %
Full factorization	150	50	0.1	0.9	Yes	30.60 %
Full conditional	150	50	0.1	0.9	Yes	51.58 %

Table 6.2: Goodness of prediction for MAR

6.3 MAR ratio of 3:2

Algorithm	p	Refitting times	Quantiles used for missing values Lower quantile	Upper quantile	Use L1 penalty?	Percentage of accurate predictions
Full factorization	60	20	0.05	0.95	No	47.88 %
Full conditional	60	20	0.05	0.95	No	47.44 %
Full factorization	60	20	0.05	0.95	Yes	30.55 %
Full conditional	60	20	0.05	0.95	Yes	47.33%
Full factorization	60	50	0.05	0.95	No	48.00 %
Full conditional	60	50	0.05	0.95	No	48.00 %
Full factorization	60	50	0.05	0.95	Yes	41.11 %
Full conditional	60	50	0.05	0.95	Yes	47.88 %
Full factorization	60	20	0.1	0.95	No	47.98 %
Full conditional	60	20	0.05	0.9	No	47.58 %
Full factorization	60	20	0.1	0.95	Yes	30.27 %
Full conditional	60	20	0.05	0.9	Yes	46.94 %

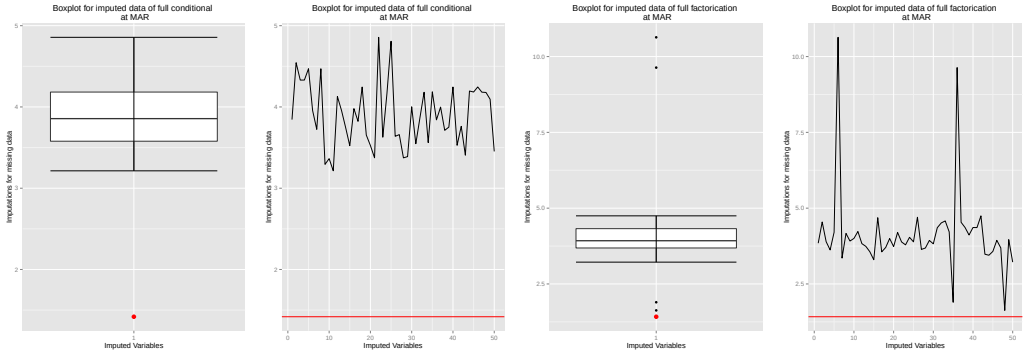
Table 6.3: Goodness of prediction for MAR ratio 3:2

6.4 NMAR

Algorithm	p	Refitting times	Quantiles used for missing values: Lower quantile	Upper quantile	Use L1 penalty?	Percentage of accurate predictions
Full factorization	10	20	0.05	0.95	No	10.48 %
Full conditional	10	20	0.05	0.95	No	09.79 %
Full factorization	10	50	0.05	0.95	No	09.79 %
Full conditional	10	50	0.05	0.95	No	09.79 %
Full factorization	10	20	0.05	0.95	Yes	08.39 &
Full conditional	10	20	0.05	0.95	Yes	09.79 %
Full factorization	10	50	0.05	0.95	Yes	08.39 &
Full conditional	10	50	0.05	0.95	Yes	10.48 %
Full factorization	10	20	0.1	0.9	No	10.13 %
Full conditional	10	20	0.1	0.9	No	08.75 %
Full factorization	10	50	0.1	0.9	No	09.21 %
Full conditional	10	50	0.1	0.9	No	10.59 %
Full factorization	10	20	0.1	0.9	Yes	9.67 %
Full conditional	10	20	0.1	0.9	Yes	8.29 %
Full factorization	10	50	0.1	0.9	Yes	11.05 %
Full conditional	10	50	0.1	0.9	Yes	10.59 %
Full factorization	150	20	0.05	0.95	Yes	14.98 %
Full conditional	150	20	0.05	0.95	Yes	17.31 %
Full factorization	150	50	0.05	0.95	Yes	16.59 %
Full conditional	150	50	0.05	0.95	Yes	17.85 %
Full factorization	150	20	0.05	0.95	No	16.77 %
Full conditional	150	20	0.05	0.95	No	17.49 %
Full factorization	150	50	0.05	0.95	No	16.64 %
Full conditional	150	50	0.05	0.95	No	17.80 %
Full factorization	150	50	0.1	0.9	No	23.08 %
Full conditional	150	50	0.1	0.9	No	22.24 %
Full factorization	150	50	0.1	0.9	Yes	12.19 %
Full conditional	150	50	0.1	0.9	Yes	22.24 %

Table 6.4: Goodness of prediction for NMAR

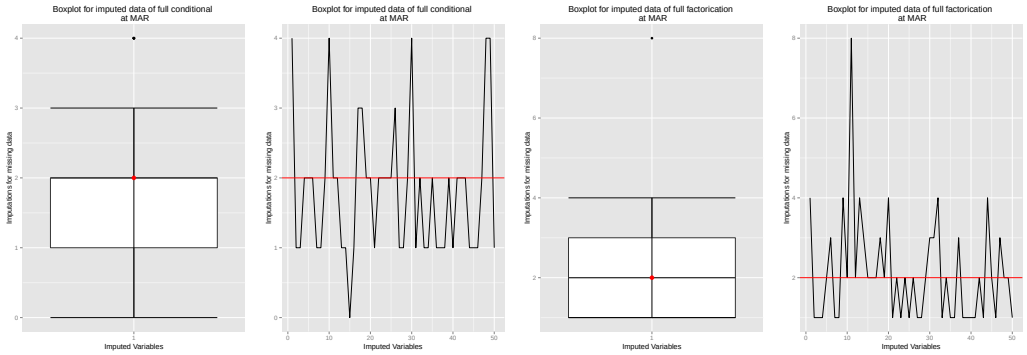
6.5 Extra Plots



(a) Full conditional approach

(b) Full factorization approach

Figure 6.1: Comparison of Paths and Boxplots for a gamma distribution



(a) Full conditional approach

(b) Full factorization approach

Figure 6.2: Comparison of Paths and Boxplots for a poisson distribution

6.6 Pooled variances and CI plots for lasso usage

Models compared in table 6.5 were fitted with 20 refits, usage of lasso regression, 0.2 missing rate in case of MCAR and 0.05 and 0.95 quantile boundaries for NMAR/MAR. Accuracy numbers are given in percent.

10:1 ratio

	MAR		MCAR		NMAR	
	Full conditional	Full factorization	Full conditional	Full factorization	Full conditional	Full factorization
Normal	5.45	8.02	2.59	2.55	1.89	3.59
Gamma	61.29	138.58	43.68	49.35	24.23	32.71
Poisson	20.42	18.85	17.74	12.84	6.66	6.31
Binomial	2.72	3.99	2.65	2.64	1.84	7.44

Table 6.5: Table of pooled variances for proportion 10:1 using lasso

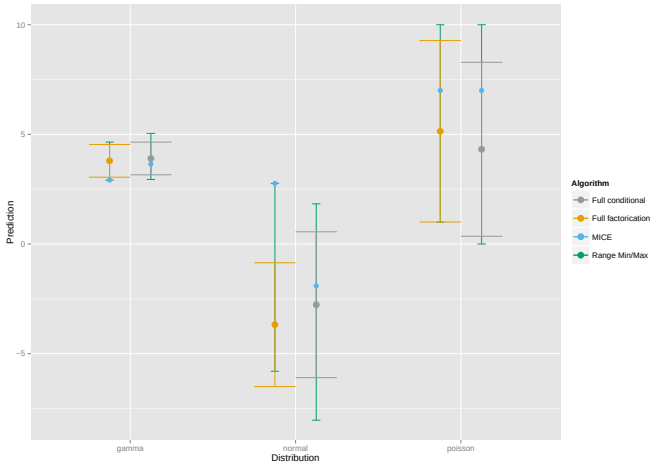


Figure 6.3: CI plot for 10:1 proportion at MAR fit using LASSO and 50 refits

Imputations seem to be not dependent on first starting values, especially if one looks at full factorization and normal and gamma distribution. Full conditional means of imputation are still very close to the `mice` imputation. For normal and poisson distribution the means of imputation are not centered around the range of maximum/minimum imputations.

2:3 ratio

	MAR		MCAR		NMAR	
	Full conditional	Full factorization	Full conditional	Full factorization	Full conditional	Full factorization
Normal	9.90	5.79	9.47	5.80	6.43	3.49
Gamma	116.15	151.43	126.95	134.72	71.08	91.67
Poisson	18.09	8.80	16.92	5.86	10.41	4.38
Binomial	8.98	5.12	10.55	4.75	8.97	5.45

Table 6.6: Table of pooled variances for proportion 2:3 using LASSO

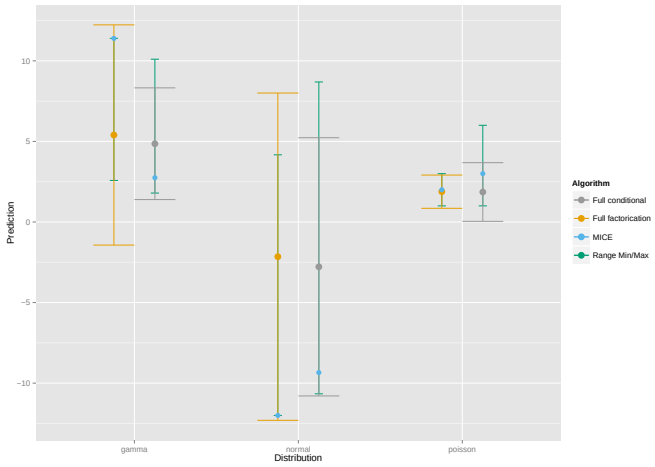


Figure 6.4: CI plot for 2:3 proportion at MAR fit using LASSO and 50 refits

Comparison of pooled variances between different simulations

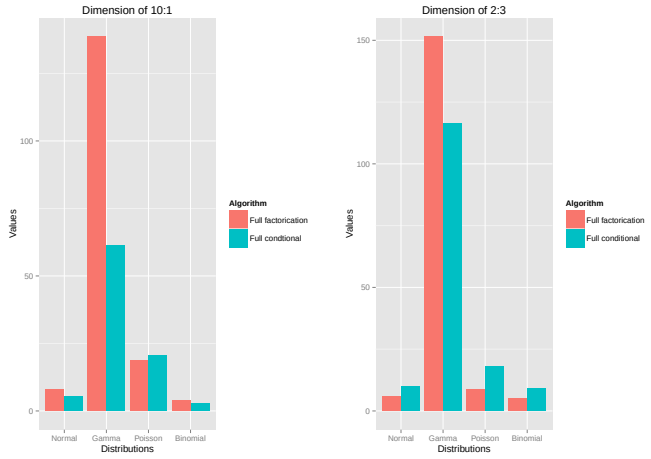


Figure 6.5: Pooled variances for 50 refits and usage of LASSO fitting

6.7 Content of CD-ROM

The appended CD-ROM holds the following contents

- `R-Scripts and Simulations` All scripts and simulations that were undertaken for the master thesis. Simulations were saved as `.RData` files, which can be loaded into the `R` console with the `load` command.
  - `Datasets simulated via copula`
    - `Seed 1234`
    - `Seed 1000`
  - `Dependencie structures for MAR mechanism`
    - `Seed 1234`
    - `Seed 1000`
  - `MAR Simulations`
    - `Seed 1234`
    - `Seed 1000`
  - `MCAR Simulations`
    - `Seed 1234`
    - `Seed 1000`
  - `NMAR Simulations`
    - `Seed 1234`
    - `Seed 1000`
  - `R-Scripts`
- `Pictures` Contains all pictures that are depictured in the thesis and extra plots for further research
- `Thesis` The thesis in pdf format

Declaration of Authorship

I hereby confirm that I have authored this master's thesis independently and without use of others then the indicated resources.

Munich, 27th of January 2015

Felix Klug