Swiss Federal Institute of Technology Zurich

Seminar for Statistics

Department of Mathematics

Semester Paper

Fall 2015

David Pham

Missing Data: Empirical Comparison between Imputation and Nearest Neighbors Algorithms

Submission Date: February 15th 2016

Adviser: Dr. Martin Maechler

To the *R* community and *ESS* developers for their contribution.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Data has never been so cheap to create and to collect. Sensors for genetic studies, sensors on smartphone or cookies on most web browsers produce a vast amount of new data to analyze. Because of this quantity of data, new methodology had to be developed to cope with new problems: more features than observations in a data set, how to store and retrieve data efficiently and how to visualize it. One point that is not often emphasized is that most of the data are cleaned before they are analyze and one step is usually to handle missing data: that could be a observation or a feature that could not be measured

At some occasion, researchers just can ignore the incomplete observations. Nevertheless, in many modern problems, the probabilty of getting one complete observation is near 0 and these researchers should then discard almost the entire data set. *Multiple imputation* methods are one solution to this problem has been developped at the end of the twentieeth century and the beginning of the new millenium with the idea that under some assumption, possibles values for the missing variables could be retrieved by estimating the relationship of the missing features with the other observed variable and then sampling from this relationship. As an advantage, imputation methods also provide an estimate about the uncertainty that is inserted in the analysis by filling missing values.

A theoretical disadvantage of these methods is the computational costs as one has to fit many models. In contrast, *algorithmic methods* based on linear algebra and matrix completion, like nearest neighbors or the singular value decomposition are much faster, but they can not assess how much uncertainty is included through data completion.

Using the statistical software R, this semester paper studies some implementations of both methods in order to measure how they compare to each other. To that end, artifical missingness is created under several settings by deleting point from a complete data set and each of the implementation are ranked by how well they can retrieve the unboserved point. Theoretical definitions are prestened in the first chapter before digging into the results of the expermient.

Schafer and Graham (2002) and Little and Rubin (2002) offer a good technical overview of the multiple imputation, whereas Van Buuren (2012), Gelman and Hill (2006), and Matloff (2015) are more accessible. Troyanskaya, Cantor, Sherlock, Brown, Hastie, Tibshirani, Botstein, and Altman (2001) describes in detail how the algorithmic based methods are built with a comparision with data on genetics.

Chapter 2

Theoretical Background

This chapter provides an overview and an intuition on the field of missing data. It mainly follows Schafer and Graham (2002), Little and Rubin (2002), Van Buuren (2012), with some input from Wikipedia (2015), Matloff (2015), Gelman and Hill (2006), Troyanskaya et al. (2001). This chapter begins with a short description on the nature missingness, then describes several procedures in order to handle missing data.

2.1 Mechanism of missingness

Van Buuren (2012) describes two concepts helping us to understand how to solve the problem of missing data: intentional and unintentional missingness, as well as unit and item missingness. The experimenter can decide to not measures all possible variables in an experiment and encode his decisions as missing observations. This is a reasonable decision if the cost of measuring variables is material and unnecessary for some experimental case, such as in medical experimentation. However, it might also happen that the experimenter could not measure some variable, e.g. when a respondent to a survey refuse to answer to some questions. In this case, the missingness is named unintentional. The second concept of missingness is about unit and items: one says a unit is missing when none of the variables of interest could be measured, whereas item missinginess refers to some variable missing.

In order to complete missing data, assumptions need to be taken about the underlying mechanism creating missing observations: missing completely at random (MCAR), missing at random (MAR) and missing not at random (MNAR).

Notation Let $Y \in \mathbb{R}^{n \times p}$ be the data matrix containing missing data for n observations with p variables, $R = (R_{ij})_{i,j=1}^{n,p} \in \{0,1\}^{n \times p}$ denotes the response y_{ij} (i.e. $R_{ij} = 1$ is y_{ij} is observed, and is 0 otherwise). Y_{obs} and Y_{mis} denote observations which are observed, respectively, missing, such that $Y = (Y_{obs}, Y_{mis})$. Note that we always observe R and Y_{obs} whereas we usually do not have Y_{mis}.

MCAR The data are said to be *MCAR* if, for all $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, p\}$,

$$P(R_{ij} = 0 \mid Y_{obs}, Y_{mis}) = P(R_{ij} = 0),$$

or equivalently

$$P(Y = y \mid R_{ij} = r) = P(Y = y),\ y \in \mathbb{R}^{n \times p},\ r \in \{0, 1\}.$$

It means the probability of being missing depends does not depends on the actual value of Y.

MAR For multiple imputation, one requires only $R_{ij} \perp Y_{mis}$, for all $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, p\}$, that is

$$P(R_{ij} = 0 \mid Y_{obs}, Y_{mis}) = P(R_{ij} = 0 \mid Y_{obs}),$$

that is other observed variables impact of the probability of missingness but the missing mechanism only depends on the observed variables and not the actual missing value. In this case, we say the data Y are *MAR*.

MNAR The data are MNAR if for any valid pair of indices (i, j),

$$P(R_{ij} = 0 \mid Y_{obs}, Y_{mis})$$

can not be simplified. It essentially means that the rate of response depends on the actual value of the missing observations. The standard example is the survey about salary when people with high salary tend to hide their earnings.

Modern statistical technique can handle MNAR and MAR cases, whereas simple technique only MCAR, which is quite restrictive.

2.2 Statistical completion

Complete case analysis Unfortunately, One of the most used technique to cope with missing data: the researcher only keeps observation that are complete. This might lead to valid analysis, as the method does not introduce any bias if the missing values are uniformly distributed. Nevertheless, this methodology can not work in modern settings where the probability of one missing variable is quite high: Many data points would be discarded.

Pairwise deletion This methods improve from the previous one by deleting observations only if the variable which is missing must be used in the model. This is typically relevant for computing correlation for example, although some care must be taken in this case, as the resulting correlation matrix might not be semi-positive definite anymore.

Single imputation The data matrix is sorted according to some order, *last observation carried forward* is the method of replacing the missing value with last valid value. The missing value can also be replaced with the mean of the other observations, however, correlations are attenuated. Regression imputation use the other variables as predictors to replace the missing value, although precision is misleadingly augmented, hence does not reflect the statistical errors of the missing data. This problem is partially solved by multiple imputation.

Multiple imputation Under the MAR assumption, the multiple imputation (MI) is similar to bootstrapping method: the distribution of each variable conditional and the others is fitted, then in case of missing value, a sample is drawn from this distribution. The desired statistics are averaged except for the standard error which is constructed by adding the variance of the imputed data and the within variance of each data set. The last step solves the problem of understating uncertainty. Standard errors reflect missing-data uncertainty and finite-sample variation.

More precisely, in the one-dimensional case, if the sample is large enough so that the estimator Q follows a Gaussian distribution, then the estimate $\hat{Q}$ and the standard error T can be computed from the estimates of $(Q^j, U^j)_{j=1}^m$, Q^j, respectively, U^j being the fitted value of Q, respectively the standard error, for data sets j:

$$\begin{aligned}
\hat{Q} &= m^{-1} \sum_{j=1}^{m} Q^j, \\
\hat{U} &= m^{-1} \sum_{j=1}^{m} U^j, \\
B &= (m-1)^{-1} \sum_{j=1}^{m} (Q^j - \hat{Q})^2, \\
T &= \hat{U} + (1 + m^{-1}) B.
\end{aligned}$$

For confidence interval, the Student's t approximation can be used with the degree of freedom given by

$$\nu = (m-1)\Big[1 + \frac{\hat{U}}{(1+m^{-1})B}\Big]^2.$$

The estimated rate of missing information for Q is approximately $\tau/(\tau+1)$ where $\tau = (1+m^{-1})B/\hat{U}$, the relative increase in variance due to non-response. See Schafer (1997) for more cases.

An advantage of MI is the number of need imputation: the efficiency based on m samples relative o an infinite number is $(1 + \lambda/m)^{-1}$, where λ is the rate of missing information, which measures the increase in the large-sample variance of a parameter estimate due to missing values. $m = 20$ is often good in practice.

Obviously, the missing values problem is dealt before the analysis with MI, in contrast with maximum likelihood estimation. The danger from MI is the ability to use different models for imputation and analysis, which might lead to inconsistency.

2.3 Algorithmic completion

The advantage of multiple imputation is the framework provides tools to account for the uncertainty about the estimated quantities, uncertainty introduced by the completion mechanism. In contrast, algorithmic methods do not offer such information, but they are often simpler, faster and more flexible. Singular value decomposition and nearest-neighbors are two common techniques.

Singular value decomposition Singular values of a matrix Y are the square root of the non-negative eigenvalues of Y^TY and the singular value decomposition (SVD) is provided by

$$\hat{Y}_J^c = U_J D_J V_J^T, \tag{2.1}$$

where $D_J \in \mathbb{R}^{N\times p}$ is a diagonal matrix containing the leading $J < p$ singular values of Y^c and $V_J \in \mathbb{R}^{p\times p}$ and $U_J \in \mathbb{R}^{N\times N}$ is the corresponding orthogonal matrix of J right and left singular vectors. It can be proved that $\hat{Y}^c$ is the nearest matrix of Y^c among matrices with rank J with respect to the sum of squares norm $||A||^2 = tr(AA^T)$.

If y_i is any row of Y^c, consider the regression of the p values in $y_i = (y_{i1}, \dots, y_{ip})^T$ on the eigen-vectors $v_1, \dots, v_J$, each p dimensional vectors. The regression solves

$$\min_\beta ||y_i - V_j\beta||^2 = \min_\beta \sum_{l=1}^{p} \big(y_{il} - \sum_{j=1}^{J} v_{lj}\beta_j\big)^2,$$

with solution $\hat{\beta} = (V_J^T V_J)^{-1} V_J^T Y = V_J^T Y$ (since V_J is orthogonal) and orthogonal values $\hat{y}_l = V_l\hat{\beta}, l \in \{1, \dots, J\}$. Thus, according to Equation (2.1), $Y^c V_J = U_J D_j$ gives all the (transposed) regression coefficients for all the rows and $\hat{Y}^c = U_J D_J V_J^T$ all the fitted values. Hence, once the matrix V_J is computed, SVD approximate each row of Y^c by its fitted vector obtained by regression (or projection) on V_J. This suggest for a row y_i of Y_{mis} with some missing components, they could possibly be imputed from

$$\min_\beta \sum_{l=1}^{p} 1(R_{il} = 1)\big(y_{il} - \sum_{j=1}^{J} v_{lj}\beta_j\big)^2,$$

where R_{il} is the response indicator of y_{il}.

The imputation procedure can thus be described as the following.

i.) Compute the SVD of Y^c and keep V_J.

ii.) For a row y^* with any missing element, compute

$$\hat{\beta}^* = (V_J^{*T} V_J^*)^{-1} V_J^{*T} y^*,$$

where V_j^* is the shortened version of V_J with the appropriate rows removed (corresponding the missing elements of y^*). Note V_J^* no longer has orthogonal columns.

iii.) The predictions of the missing elements are $V_J^{(*)}\hat{\beta}^*$ where $V_J^{(*)}$ is the complement in V_J of V_J^*.

Usually, the data matrix is centered before SVD, however, for missing data, an intercept has to be fitted and a method based simulation is provided afterwards. The previous methods usually discards a great number of data, particularly when $p >> N$. In contrast, the next iterative procedure circumvent the problem at the cost of more computation.

(1) Set y^* as Y with all missing values filled by the mean of their row.

(2) Solve the problem

$$\min_{V_J, D_J, U_J} ||Y^* - m1^T - U_J D_J V_J^T||_F^2 \tag{2.2}$$

where $||\cdot||_F^2$ is the sum of squares of all non-missing elements and $m \in \mathbb{R}^N$ is the row means of Y^*.

(3) Predict the missing values of Y with the fitted values.

(4) Reset Y^* as Y with the missing values replaced by the result of previous step.

(5) Repeat steps 2-5, until the size of the relative update of the missing values become negligible.

According to Hastie, Tibshirani, Sherlock, Eisen, Brown, and Botstein (1999), only 6 iterations are necessary. Interestingly, the solution of Equation (2.2) is a fixed point, i.e. if missing values are filled, and the SVD algorithm is executed on the complete matrix, the solution remains identical.

Soft-impute completion The representation of the data Y described in Equation (2.1) can be softened a little bit to get faster and more stable completion. In order to do so, define $\mathcal{P}_\Omega(Y)$ as the operator that *projects* entries in Y with indices not in Ω to 0 and keep the other elements unchanged. If Ω is the set of indices (i, j) where Y have non-missing values (i.e. $R_{ij} = 1$), then $\Omega^\perp$ is the set of indices where Y has missing value and $\mathcal{P}_\Omega(Y)$ is a version of Y where missing value have been replaced with 0. The data completion problem can be interpreted as finding a matrix M minimizing

$$\frac{1}{2}||\mathcal{P}_\Omega(Y) - \mathcal{P}(M)||_F^2 + \lambda||M||_*,$$

where $||M||_F^2$ is the sum of the squared elements of M and $||M||_*$ is the sum of singular value of M (also called the nuclear norm). If Y^* solves this problem then it satisfies the following condition

$$Y^* = S_\lambda(Z),$$

where

$$Z = \mathcal{P}_\Omega(Y) + \mathcal{P}_{\Omega^\perp}(Y^*).$$

and the operator $S_\lambda(Z)$ is defined as following.

i.) Find the SVD decomposition of $Z = UDV^T$ and let d_i be the singular value of D.

ii.) Put a soft threshold on the singular values, that is, define

$$d_i^* = (d_i - \lambda)_+$$

iii.) Reconstruct $S_\lambda(Z) = UD^*V^T$. This is called the *soft-thresholded SVD*. A sufficiently large λ reduce the ranks of D^* and consequently of $S_\lambda(Z)$ as well.

Z is thus a completed version of Y, with missing value filled in. For small matrices, this is computationally feasible and for large matrices, consult Hastie and Mazumder (2015) for the methodology using sparse representation of matrices.

K-nearest neighbors completion Troyanskaya et al. (2001) presents the other end of the spectrum in term of data usage: *K-nearest neighbor averaging*. The algorithm is described as following.

i.) Computed the Euclidean distance between y^* and all the rows in Y^c, using only those co-ordinates not missing in y^*. Identify the K closest observations.

ii.) Impute the missing coordinates of y^* by averaging the corresponding coordinates of the K closest with weights proportional to their distances to y^*.

Empirically, the number of neighbors K between 5 to 10 is often a good choice for most data set.

Chapter 3

Empirical Comparison of Imputation Methods

3.1 Data set and R packages

The FLAS data set is studied in Schafer (1997) and is a great candidate for the simulation study. The data were collected in 1987 to investigate the impact of the Foreign Language Attitude Scale (FLAS), a new measure, for predicting success in learning new foreign languages. Tables 3.1 and 3.2 provide a short summary of data. The `mice` and `mi` packages have been applied to the original data set to create an artificial complete one. The latter is used as baseline to compare the imputations by different methods. Five *R* packages were selected in the study.

Amelia implemented by Honaker, King, and Blackwell (2011) provides bootstrapping methods and EM algorithm for multiple impute analysis.

impute authored by Hastie, Tibshirani, Narasimhan, and Chu (1999) uses the `impute.knn` function to provide nearest neighbors imputation.

mi created by Gelman and Hill (2011) implements the multiple imputations in a Bayesian framework.

mice written by van Buuren and Groothuis-Oudshoorn (2011) allows the user to impute values with chained equations.

softImpute distributed by Hastie and Mazumder (2015) uses singular value decomposition (or a version thereof) to complete data sets.

Each of these packages offers a function for completing data set. The simslapar packages from Hofert and Maechler (2015) provides a stable framework to conduct the simulation and gather the output from parallel simulations.

3.2 Methodology

Let $\mathcal{M}$ denote the set of imputation methods and let $Y = (y_{ij})_{i,j=1}^{n,p} \in \mathbb{R}^{n\times p}$ be a complete data matrix.

Table 3.1: FLAS data set, summary of numerical variables

Statistic	N	Mean	St. Dev.	Min	Max
FLAS	279	82.487	14.026	28	110
MLAT	230	24.257	6.256	9	40
vSAT	245	501.514	91.162	210	790
mSAT	245	564.249	88.707	320	800
eng	242	53.950	15.402	19	113
HGPA	278	2.750	0.617	0.500	3.990
CGPA	245	3.294	0.477	2.000	4.000

Table 3.2: FLAS data set, summary of factor variables

Statistic	N	Factors				
Age	268	-19	20+			
		124	144			
Sex	278	M	F			
		152	126			
Number of prior		none	1–2	3+		
foreign language	268	71	73	124		
Prior Language		french	spanish	german	russian	
	279	67	78	114	20	
Grades		F	D	C	B	A
	232	1	5	22	79	125

The experience requires to chose a complete data matrix, then to replace some of its entries with missing values and then to rank the imputation methods. In order to cope with the randomness, N_{sim} simulations are performed and then aggregated.

3.2.1 Simulation of missingness

Two types of missing were implemented: MCAR and MAR. The former is quit straightforward to implement: Given a data matrix and a missingness rate, defined as the ratio of missing value over the number of entries in the data matrix, the response[1] R_{ij} for the element y_{ij} follows a binomial distribution with probability equal to the missingness rate.

Implementation of MAR usually make assumptions on the underlying multivariate distribution is a little more involved. Nevertheless, a mechanism based of the empirical distribution of the missingness pattern can also been used. Missingness patterns are defined as $R_i = (R_{i1}, \dots, R_{ip})$, where R_{ij} is the response of y_{ij}. The raw data set contained pattern of missingness and for a missing rate, one could sample the patterns from it with the multinomial distribution until the desired missingness rate is reached. The patterns

[1] Recall that R_{ij} is 0 if y_{ij} is missing and 1 otherwise.

are then randomly assigned to our completed data set. This method has the disadvantage that it can not attain any missingness rate between 0 and 1, as one can only assign one pattern per observation.

Table 3.3 displays the distribution of missingness pattern in the FLAS data set[2]. Using the third and last column of the same table, under the condition there exist at least one missing value, one can get the conditional expected missing rate ($\approx$ 20.8%) for one observation. Hence, there exists some threshold for the missing rate which our version of the MAR mechanism can not overcome. In the FLAS data set, 30% is approximately this threshold.

Missingness pattern	Frequency	Probability	♯ of missing values	Missing rate
111111111111	174		0	0.00
111110111111	26	0.25	1	0.08
111111000101	20	0.19	4	0.33
111111111110	18	0.17	1	0.08
111110111110	15	0.14	2	0.17
111111000100	7	0.07	5	0.42
100111111111	3	0.03	2	0.17
100110111111	3	0.03	3	0.25
111111110110	2	0.02	2	0.17
111110000101	2	0.02	5	0.42
100111000101	2	0.02	6	0.50
111111110111	1	0.01	1	0.08
111111000001	1	0.01	5	0.42
111110000100	1	0.01	6	0.50
111011111110	1	0.01	2	0.17
100111111110	1	0.01	3	0.25
100110111110	1	0.01	4	0.33
100110000100	1	0.01	8	0.67

Table 3.3: Distribution of the vector of response (or missingness pattern) over the observation of the FLAS data set. Each observation of the data set is a p dimensional vector with a corresponding vector of response $R_i = (R_{i1}, \ldots, R_{ip})$, where $R_{ij} = 0$ if the j-th entries of the i-th observation is missing. The third column is the second column normalized after leaving the first row (as there are no missing data). The last column is the proportion of 0 in each missing pattern (the first column).

3.2.2 Ranking methods

For a simulated data matrix $Y^l = (y^l_{ij})^{n,p}_{i,j=1}$, $l \in \{1, \ldots, N_{sim}\}$ with missing values, its associated response matrix $R^l = (R^l_{ij})^{n,p}_{i,j=1}$ and an imputation method $m \in \mathcal{M}$ with predicted values $\hat{y}^l_{ij}$ if y^l_{ij} is missing. For *numerical* variables, the scaled mean squared

[2] `na.pattern` function from the `Hmisc` package compute the missingness pattern , although the 0 and 1 are swapped.

error (SMSE),

$$\mathrm{SMSE}_{l,j}^{m} = \Big\{ \sum_{i=1}^{n} 1(R_{ij}^{l} = 0) \Big\}^{-1} \sum_{i=1}^{n} \Big(\frac{\hat{y}_{ij}^{l} - y_{ij}^{l}}{\mu_j} \Big)^2 \cdot 1(R_{ij}^{l} = 0),\ j \in \{1, \ldots, p\},\ m \in \mathcal{M}, \tag{3.1}$$

where $\mu_j = n^{-1} \sum_{i=1}^{n} y_{ij}$, is used to assess the quality of imputation methods. For *factors* variables, the conservative $0 - 1$ loss is employed. To aggregate the measure across the columns of the data matrix, for each column, the imputation methods are ranked according to their SMSE, then these ranks are summed by imputation method. More precisely, the score s_l^m of the imputation methods $m \in \mathcal{M}$ can be expressed as

$$s_l^m = \sum_{j=1}^{p} \sum_{\nu \in \mathcal{M}} 1(\mathrm{SMSE}_{l,j}^{m} \leq \mathrm{SMSE}_{l,j}^{\nu}). \tag{3.2}$$

The scores s_l^m are then used to assess and compare[3] the performance of the imputation methods.

3.3 Implementation constraints

Data type for soft impute and nearest neighbors The implementation of two methods did not allow for factors variable. Although it is not a paramount task to transform the factor into numerical data type, some care should be taken to make conversion correctly.

Nearest neighbors imputation It appears that the `impute.knn` method from the `impute` package from the *bioconductor* repository causes *segmentation fault* (with the underlying FORTRAN code) when the number of neighbors is either to high with respect to the available data.

Collinear dimensions If the data matrix Y has collinear variables, then some challenges might occur when variables are collinear. More precisely, although multiple imputation techniques try to estimate

$$Y_j \mid Y_{k_1}, \ldots, Y_{k_j},$$

using regression models, their results might be unstable if Y_{k_i}, $i \in \{1, \ldots, k_j\}$ are linearly dependent. Said differently, as regression parameters depends on the quality of the inversion the data matrix, but a matrix with collinear columns has an unstable inverse[4]. The `Amelia` package is the most prone to this issue, although `mice` and `mi` algorithms fail to converge at some point when Y_{k_i} exhibit high collinearity.

Timing Usually, CPU time, i.e. time spent by the processor on the *R* process, is measured to evaluate the speed performance. Nonetheless, a trend has emerged for packages implement pretty good parallelism, i.e. packages implement the parallelism procedures themselves. It leads to underestimated human elapsed time as most of the computational burden is performed by sub-process.

[3]It yields a ranking where the lower number is better than a higher one.
[4]The matrix is so-called *ill conditioned*.

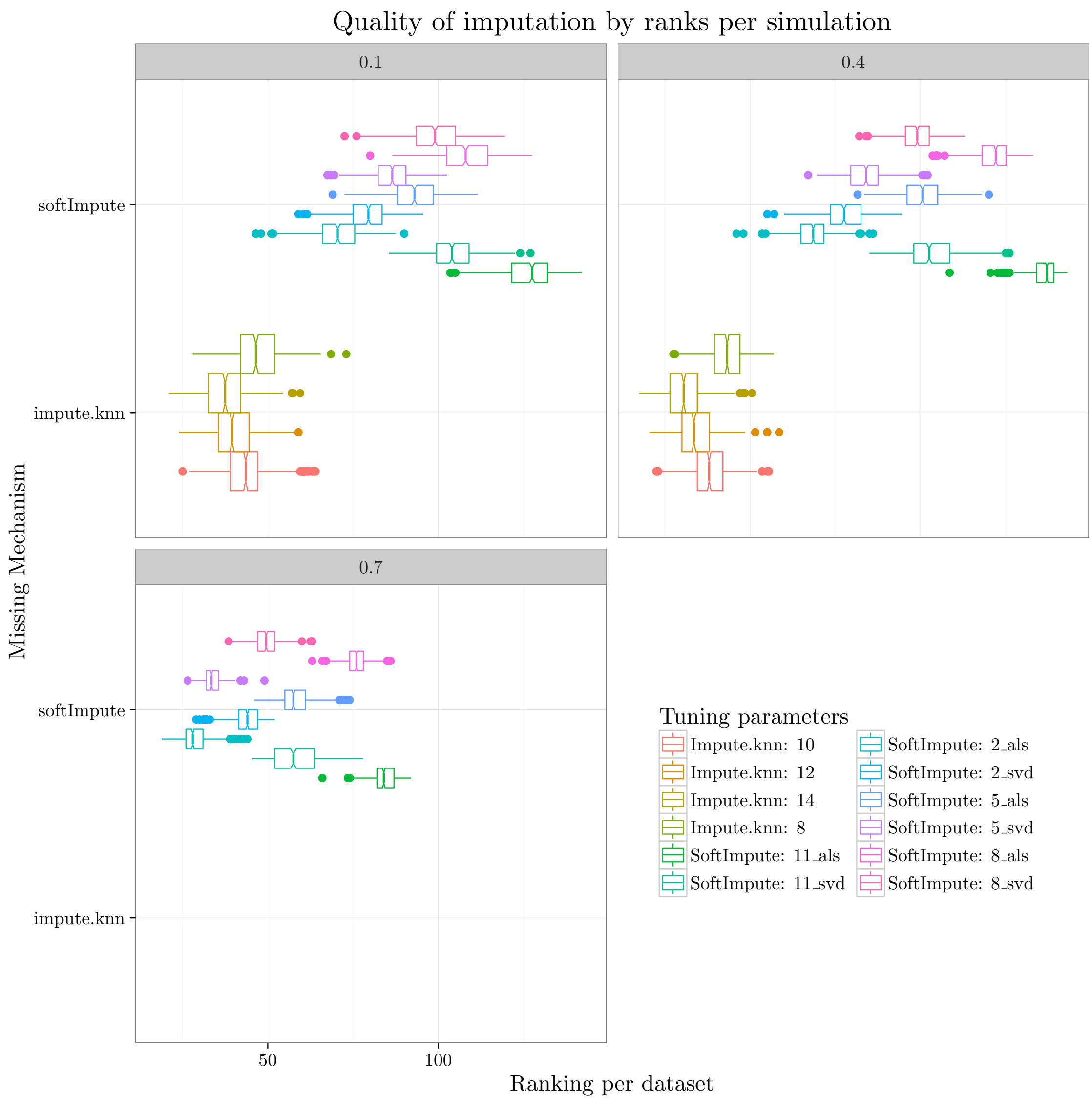

Figure 3.1: Relative ranking of imputation quality of the tuning parameters of softImpute and impute.knn. For impute.knn, the number of neighbors is the tuning parameters, whereas for softimpute, it is the maximum rank and estimation method of the output matrix.

Tuning parameters For the packages `softImpute` and `impute`, some defaults parameters for the imputation methods are provided. Figure 3.1 displays how the quality of the imputation evolves with the parameters. The `impute.knn` function, the quality of the inference grows with the number of neighbors. The `softImpute` function unexpectedly offers good default parameters, even if some restrain should be kept when the missing rate is high.

3.4 Results

Figure 3.2 summarize the results under the MCAR missing mechanism. `softImpute` and `mice` were the only package able to cope with a quite high missing rate ($p \geq 0.7$). `impute.knn` from the `impute` package, when it works, is almost always the method with the best score, as defined in Equation (3.2). `mice` offers a good balance between speed, robustness and quality of imputations, but the methods depends on the linear dependency of the columns of the data matrix. Although `softImpute` can cope with almost any type of data matrix, its inferences are sub-par with the other methods. Some further analysis might be needed to confirm this results. `Amelia` is the package which is the least able to cope with a random impute matrix: routines failed without exception with any missing rate above $p > 0.3$. The `Amelia` and `mi` packages use by default parallel back-ends to perform their computations. However, they are the slowest methods in terms of elapsed time. This might be overcome by setting the number of iteration to a lower threshold. Nonetheless, this is not recommended as convergence is not guaranteed when data input might have dimensions with strong linear dependency.

Figure 3.3 shows the measure from Equation (3.1) for missingness rates $p \in \{0.2, 0.5, 0.7\}$. For numerical data, all methods but `softImpute` have the same order of errors with K-nearest neighbors being slightly better than the others. However, the latter method does not allow for inference of the imputation error. As Figure 3.4 shows, the previous statements are not impacted by changing the missingness mechanism to (our implementation of) MAR.

3.5 Open questions

Heuristically, the quality of models output normally depends on the amount of available data. In the missing data framework, precision are needed for this notion: Is it the number of complete observations, the number of non-missing values, or a mixture of both? Interactions between the number of observation n, the number of dimensions p and imputation methods with their optimal parameters are left unanswered with this work. In order to answer this question, a multivariate sampling mechanism should be devised and tested.

Moreover, are there any reasonable solutions which can be applied to overcome collinearity? One could cluster the similar dimension and then pick one randomly to create an imputed value. Nevertheless, such solutions were not yet implemented.

Additionally, this small simulation study has been applied to certain data set and it should be interesting to repeat the experience with other real data.

Finally, the concern of this work has been to apply imputation methods to retrieve potential candidate values for inference, it would have been interesting to verify the quality of inferences performed with the imputed data set, for example multiple imputation against nearest neighbors.

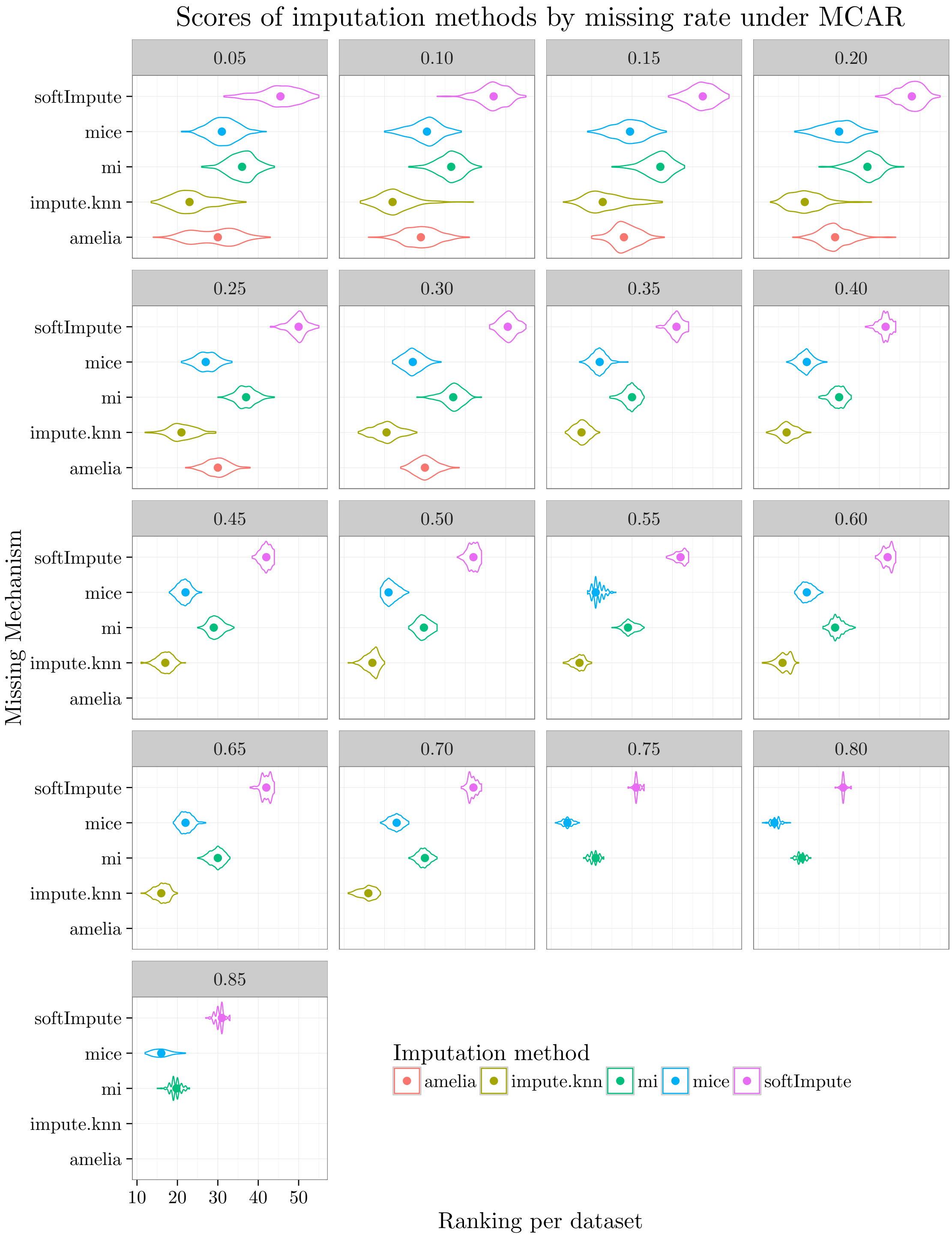

Figure 3.2: Rankings of imputation methods on the FLAS data set grouped by missing rate, under the MCAR mechanism with missing rate. Labels in the boxes provide the missing rate.

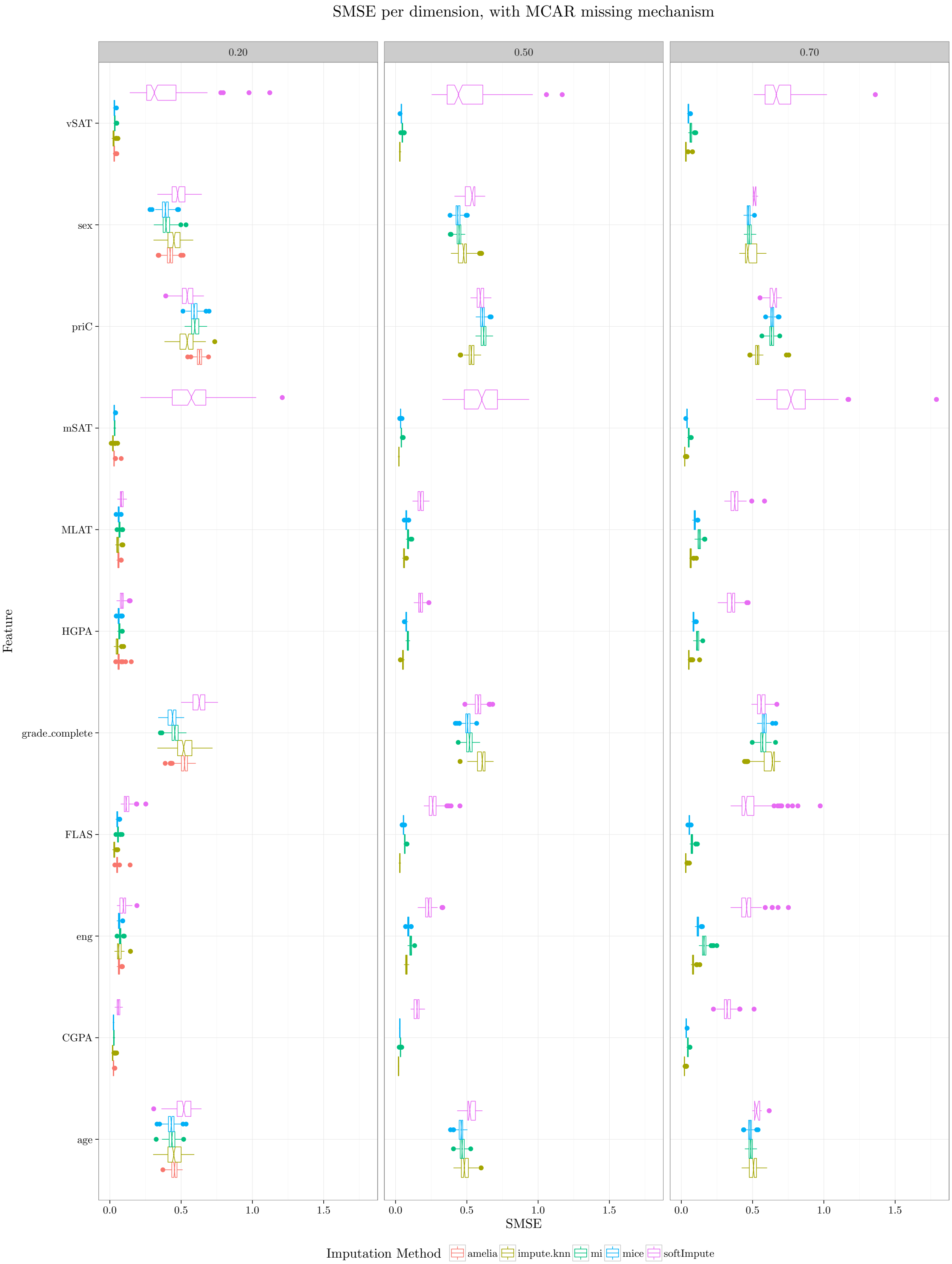

Figure 3.3: SMSE for selected missingness rate with MCAR against imputation methods.

Figure 3.4: SMSE of imputation methods on the FLAS data set grouped by missing rate, under the MAR mechanism. Labels in the boxes provide the missing rate.

Chapter 4

Conclusion

In this semester paper, part of theory of data completion is reviewed. The main work is devoted to build a framework to test several R packages for imputation methods on the FLAS data set. With this data set, it appears that K-nearest neighbor imputation works fairly well, although its implementation might throw frustrating low level errors (segmentation faults). Except for `softImpute`, all methods have the same order of error (distance between the imputed and true value). In practice, they could unfortunately not be used as black-box as most data matrix have collinear dimensions which constitutes an issue for all the algorithms based on regression.

Finally, as departing words, one should not forget why these technique exists. From Schafer and Graham (2002),

> With or without missing data, the goal of a statistical procedure should be to make valid and efficient inferences about a population of interest – not to estimate, predict, or recover missing observations nor to obtain the same results that we would have seen with complete data.

Bibliography

Gelman, A. and J. Hill (2006). *Data analysis using regression and multilevel/hierarchical models.* Cambridge University Press.

Gelman, A. and J. Hill (2011). Opening windows to the black box. *Journal of Statistical Software 40.*

Hastie, T. and R. Mazumder (2015). *softImpute: Matrix Completion via Iterative Soft-Thresholded SVD.* R package version 1.4.

Hastie, T., R. Tibshirani, B. Narasimhan, and G. Chu (1999). *impute: impute: Imputation for microarray data.* R package version 1.42.0.

Hastie, T., R. Tibshirani, G. Sherlock, M. Eisen, P. Brown, and D. Botstein (1999). Imputing missing data for gene expression arrays.

Hofert, M. and M. Maechler (2015). *simsalapar: Tools for Simulation Studies in Parallel with R.* R package version 1.0-5.

Honaker, J., G. King, and M. Blackwell (2011). Amelia II: A program for missing data. *Journal of Statistical Software 45*(7), 1–47.

Little, R. and D. Rubin (2002). Statistical analysis with missing data.

Matloff, N. (2015). New r software/methodology for handling missing data.

Schafer, J. L. (1997). *Analysis of incomplete multivariate data.* CRC press.

Schafer, J. L. and J. W. Graham (2002). Missing data: our view of the state of the art. *Psychological methods 7*(2), 147.

Troyanskaya, O., M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman (2001). Missing value estimation methods for dna microarrays. *Bioinformatics 17*(6), 520–525.

Van Buuren, S. (2012). *Flexible imputation of missing data.* CRC press.

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

Wikipedia (2015). Imputation (statistics).

Appendix A

R Implementation Details

The complete R code to generate the result is stored on https://github.com/davidpham87/ethz_missing_data where the procedure are well documented.

A.1 Completion of the original FLAS data set

One of the critical element in the comparison of completion methods is the complete data set from which the artificial incomplete data set is derived. As explained, the original FLAS data set contains missing value and the following steps to complete it are described below.

For each of the `mice` and `mi` packages, 20 imputed data set are created. The selected final observation is then either the arithmetic average for numerical variables or the mode for factor variables among the 40 imputed data set and it becomes the baseline for all the comparison.

```r
library(mice)
library(mi)
library(Hmisc)  #package for na.pattern() and impute()
library(data.table)
library(parallel)

source('completion_fns.R')

set.seed(1)

options(mc.cores=max(detectCores()/2, 1)) # multithreading machine

FLAS ← readRDS('../data/FLAS.rds')

## Complete the FLAS data set with the complete grades
## 0 = F, 1 = D, 2 = C, 3 = B, A = 4
lower.grades ← read.csv('../data/flas_lower_grades.csv')
grades ← as.numeric(FLAS[, 'grade']) + 2

for (i in seq(1, nrow(lower.grades))){
  grades[lower.grades[i, 1]] ←  lower.grades[i, 2]
}
```

```r

grades ← factor(grades, labels=c('F', 'D', 'C', 'B', 'A'))
FLAS[, 'grade_complete'] ← grades

saveRDS(FLAS, '../data/FLAS_clean.rds')

names(FLAS)  #show variable names
md.pattern(FLAS) #show patterns for missing data
na.pattern(FLAS) #show patterns for missing data

dataset ← FLAS
save.path ← '../data/FLAS_complete_average.rds'
column.type.mi ← list(grade_complete="ordered-categorical")
n ← 20

data.imputed ← imputeData(FLAS, n, column.type.mi) # Impute data from mice
    and mi
data.imputed ← do.call(c, data.imputed) # Flatten the variable as data
    imputed is a list of list of df.
data.average ← averagingImputations(data.imputed) # Mean for numerical value
    and mode otherwise

saveRDS(data.average, save.path)
```

A.2 Generation of low-level errors

If the reader is interested in looking at the output of these script, they are available on github.com/davidpham87/ethz_missing_data/tree/master/R/bug_replication.

Amelia `Amelia`'s implementation seems to have difficulties to handle some structures of data matrix with missing values. The following code snippet shows how to reproduce this errors.

```r
### Functions for runing the simulation
setwd('../')
source('simulation_fns.R')

toLatex(sessionInfo(), locale=FALSE)

### Bug: In Amelia, when the provided matrix is not invertible,
### the process has a core dump segmentation fault.

####################################################################
### FLAS Dataset

flas.li ← loadFLASData()

####################################################################
### Test case

## Exception Throw (core dumped)
dataset ← MCAR(flas.li$data, p=0.35, random.seed=59)
```

```

## Throw exception
a.out ← amelia(dataset, 20, noms=c("lang", "age",  "priC", "sex"),
               ords=c("grade_complete"), p2s=0)

## Same core dumped
dataset ← MCAR(flas.li$data, p=0.7, random.seed=1)
## Throw exception (not invertible)
a.out ← amelia(dataset, 20, noms=c("lang", "age",  "priC", "sex"),
               ords=c("grade_complete"), p2s=0)
```

Impute The `impute` package seems to call an underlying `fortran` procedure. The call can throw some segmentation fault errors depending on the nature of some arguments.

```
### Functions for runing the simulation

## Bug appears when p=0.05, n.imptuation = 100, n.random.seed=3
## The call:
## do.call(impute::impute.knn, c(list(as.matrix(dataset)), list(NULL))
## Creates a memory dump.

setwd('../')
source('simulation_fns.R')


#####################################################################
### Simulation Args for FLAS
imputation.methods ← c("impute.knn")
sfile.path ← paste0("simulation_rds/imputation_", "20151230_1430_",
                    paste0(imputation.methods, collapse='_'), ".rds")

#####################################################################
### FLAS

flas.li ← loadFLASData()
flas.imputation.args ← imputationArgsFLAS()

set.seed(1)
dataset ← MCAR(flas.li$data, 0.10, random.seed=1)

## Transfrom factors to integers
## boolean vectors stating factors columns
fctrs ← sapply(1:ncol(dataset), function(jdx)
    any(c("factor", "string") %in% class(dataset[1, jdx])))
lvls ← lapply(dataset[, fctrs], levels)

dataset[, fctrs] ← lapply(dataset[, fctrs], as.numeric)
x ← as.matrix(dataset)

### Core dump
replicate(10, {
  impute::impute.knn(x, NULL)
})
```

Declaration of Originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor .

Title of work (in block letters):

Missing Data: Empirical Comparison between Imputation and Nearest Neighbors Algorithm

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
PHAM	David-Olivier Hoang-Nam

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the Citation etiquette information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work .
- I am aware that the work may be screened electronically for plagiarism.
- I have understood and followed the guidelines in the document *Scientific Works in Mathematics.*

Place, date:

Zurich, 15.02.2015

Signature(s):

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.