Imputation Methods in Dialysis Data

Mattias Albinsson, Erik Gillsbro

Master's thesis
2017:E26

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

CENTRUM SCIENTIARUM MATHEMATICARUM

Lund University

Master's Thesis in Mathematics

Imputation Methods in Dialysis Data

Mattias Albinsson and Erik Gillsbro

Lunds universitet
Lunds Tekniska Högskola

Supervised by
Prof. Kalle Åström[1], Ass. Prof. Niels Christian Overgaard[1] and Dr. Fredrik Ståhl[2]

[1]Department of Mathematics, Lund University, Sweden.
[2]Lytics Health AB, Malmö, Sweden.

LYTICS

June 6, 2017

Abstract

Imputation of data is the process of filling in missing values in an incomplete data set. Missing data is a common problem in many fields, not least in clinical research. This report aims to evaluate different methods for imputing missing data in health records of dialysis patients. The imputed data will, in a related project, be used to predict hospitalizations of dialysis patients. The hope is that an imputed data set will give a higher hit rate when predicting the hospitalizations of those patients. Seven different imputation methods, with varying complexity, were considered and compared to the presently used imputation method, which was to simply use the latest observed value as imputed value. The methods were evaluated according to their performance compared to a validation data set, as well as if improvement in prediction of hospitalizations were seen. We found that methods built on within-variable dependencies performed better than methods built on between-variable dependencies. Specifically, time series models using a Kalman filter gave the best results. Also, an improvement in the prediction algorithm could be seen when using more sophisticated imputation methods compared to using the presently used imputation method. When increasing the amount of missing data we still managed to obtain good results in contrast to the present method. All data analyzed in this project was from dialysis patients suffering from end stage renal disease.

Keywords: Imputation, Dialysis, Missing data

Preface

It is our hopes that the investigation on imputation methods can contribute to improve the prediction algorithm currently used at Lytics Health AB, and in the long run make the life of dialysis patients a little less cumbersome. This Master's Thesis project was carried out during the spring of 2017 at Lytics Health AB in Malmö, Sweden. Lytics is a company specialized in finding unique medical insights using artificial intelligence tools to develop the healthcare solutions of tomorrow. The data was provided from Centers for Dialysis Care in Cleveland, Ohio; whom Lytics are currently collaborating with.

We would like to give our gratitude to our supervisors Fredrik Ståhl at Lytics Health AB, who has helped us keeping the right course through out the project and to Kalle Åström and Niels Christian Overgaard, at the Department of Mathematics, whom have given us valuable insights in times of need. Niklas Hansson, whom we have had fruitful discussions with, and provided us with the prediction results. Finally, we would like to thank all the employees at Lytics for their hospitality during the course of our work.

Populärvetenskaplig sammanfattning - Imputationsmetoder för dialysdata

Vid de flesta processer som innefattar insamlande av data kan det ibland uppstå problem som gör att all data inte samlas in korrekt. När detta händer uppstår det tomma platser i datan där värden saknas. Det finns många sätt att hantera dessa tomma platser och i många fall kan man bortse från dem. Om ett komplett dataset behövs för analys kan så kallade imputationsmetoder användas för att fylla i de tomma platserna med rimliga värden. I den här rapporten har olika sådana imputationsmetoder implementerats, testats och utvärderats på data från dialyspatienter.

Med ökande välfärd och tillgång till medicin i världens mest avlägsna hörn har infektionssjukdomar fått se sig besegrade. Nu för tiden är det istället kroniska sjukdomar som är den största dödsorsaken världen över. Då kroniska sjukdomar inte går att bota med medicin läggs istället mycket forskning och resurser på att försöka förhindra att människor drabbas av dessa sjukdomar. Vi vet alla att man ska äta rätt, träna rätt och sova rätt för att leva så bra som möjligt. Det är dock lättare sagt än gjort och så småningom drabbas de flesta ändå av en kronisk sjukdom inför dödens bädd. Till exempel kan njurarna lägga av och du kan tvingas leva uppkopplad till en dialysmaskin resten av dina kvarvarande år. För att spä på eländet ännu mer händer det ofta att dialyspatienter insjuknar i akuta sjukdomar och tvingas bli inlagda på sjukhus. Så vore det inte skönt att i denna misär med tidsödande behandlingar flera gånger i veckan, strikta matscheman för att inte få i sig för mycket socker eller vätska och oförmågan att ens kunna kissa längre, i alla fall slippa att bli inlagd på sjukhus? Det är precis det som Lytics Health AB vill kunna undvika. De har utvecklat en algoritm som med hjälp av dialyspatientens data kan förutsäga om hen kommer bli inlagd på sjukhus inom de närmsta 30 dagarna och på så vis kunna sätta in åtgärder i tid för att förhindra det. Det finns dock ett problem och det är att algoritmen inte fungerar om datakvaliteten är för dålig. Till exempel om datan innehåller massa tomma platser där värden saknas. Det är där vårt examensarbete kommer in i leken. Målet med detta arbete är alltså att utveckla imputationsmetoder för att fylla i dessa saknade värden.

Det finns många sätt att angripa detta problem med saknad data. Det lättaste, vilket också används som imputationsmetod i dagsläget, kan vara att helt enkelt använda ett tidigare värde och fylla i där det fattas. Denna metod är dock ganska dålig och missvisande för variabler som ändrar sig mycket från behandling till behandling, som till exempel patientens puls. Vad vi har undersökt är om mer avancerade algoritmer, som använder sig av information från större del av datan, kan ge

ett bättre resultat. Tidsserie-modeller, maskininlärnings-modeller och linjära modeller har testats med varierande resultat. Utvärderingen skedde dels genom att jämföra hur nära algoritmerna kunde imputera saknade värden i ett givet dataset och dels genom att se om prediktionerna på dialyspatienter blev bättre.

Slutsatsen av arbetet är att man ganska lätt kan öka kvaliteten av datan genom att använda sig av imputationsmetoder. Vi ser en liten förbättring i resultatet för prediktionen när vi använder oss av imputationsmetoder. Med större mängd saknad data och fler imputerade variabler hade troligtvis skillnaden blivit större. I framtiden tror vi absolut att användandet av mer sofistikerade imputationsmetoder kommer att växa då värdet av högkvalitativ data ständigt ökar. Det bästa vore dock att angripa roten till problemet och minska mängden saknad data vid datainsamling.

Contents

Chapter 1

Introduction

Chronic diseases are the leading causes of death worldwide, with cancer and cardiovascular diseases (CVD) as the main reasons [1]. While chronic kidney disease (CKD) is not in itself one of the main causes of death, it is very common in people with CVD and it has been shown that CKD multiplies the risk for adverse outcomes in these conditions [1]. Chronic kidney disease is a family of diseases that are difficult and expensive to treat, and often no cure exists other than to regularly undergo dialysis or to get a kidney transplant. According to the National Kidney Foundation [2], 10% of the world's adult population is affected by chronic kidney disease and over 2 million people receive continuous treatments with dialysis. This is, however, only about a tenth of the number of people who are in need of dialysis to survive [2]. This lack of treatment for people is mainly due to the cost of dialysis and that it requires both a lot of staff and also a lot of space to be carried out properly. Another problem is that dialysis patients often tend to catch other diseases and sometimes become acutely ill. When this happens the patients have to be hospitalized for an unknown period of time which is both exhausting and possibly fatal for the patient as well as expensive and obstructive for the hospital. Consequently, to be able to predict whether or not a dialysis patient will be hospitalized in the near future would save lives, money, and give more people the opportunity to be treated. Hence this is an important step in modernizing today's treatments.

Missing data is a common problem in clinical research. For example, in clinical studies where patients are followed over time, patients may drop out of the study or not attend every scheduled clinic visit, giving rise to missing data on relevant measurements [3]. If a complete data set is needed for analysis, so called imputation methods can be used on the data containing missing values. Imputation of data is the process of filling in missing values in a data structure containing such. The aim of this project was to investigate and implement an imputation method which could fill in the gaps where data was missing in records of dialysis patients, preferably outperforming the method used today. Ultimately, this reconstructed data was supposed to be used, in another project, to predict the likelihood of a dialysis patient to be hospitalized in the near future. Thus the focus in this report will be concentrated on the construction and evaluation of imputation methods, not on constructing prediction models. However, we will evaluate our results both by how well the missing values are imputed and by how much the predictions improved with imputed data compared to non-imputed data.

The field of missing data is vast but nevertheless imputation is often neglected in scientific studies. There are a few standard methods to handle missing data which we name here. First, prevention of missing data is the most direct attack on reducing the amount of missing data. A lot of advice, examples, and preventive measures can be found in books on research methodology and data quality. Some examples are Shadish et al. [4], De Leeuw et al. [5], Dillman et al. [6], and Groves et al. [7]. Second, likelihood based approaches are widely used. A model for the data at hand has to be specified, then, the parameters of the model are estimated by maximum likelihood, the expectation maximization algorithm, the sweep operator or similar estimation techniques. Two good books are Little and Rubin [8] and Schafer [9]. Third, multiple imputation is an extension of the likelihood based approach were multiple data sets are imputed. It adds an extra step where the imputed data values are drawn and is considered the standard to which new methods are evaluated against. A good book explaining the concepts of multiple imputation is written by Van Buuren [10], a more in-depth book is written by Rubin [11].

There are many approaches of how to best impute values and the problem has been approached in numerous cases of medical data studies. To name a few, Shara et al. [12] made a study concerning imputation methods for missing renal function data and both Montez-Rath et al. [13] and Moniek et al. [14] used multiple imputation methods for imputing values in studies regarding kidney diseases and dialysis patients, respectively. The results are often quite good, hence the objective to be dealt with should be possible to overcome. However, the problem can be addressed in a variety of different ways and there exists no rule of thumb of which method to choose. What separates our work with previous ones is that we, apart from investigating classical imputation methods such as likelihood-based imputation and multiple imputation, also look more into both time series models and machine learning algorithms and compare their results.

Chapter 2

Background

We begin this chapter by introducing and defining Chronic kidney disease and End-stage renal disease. All patients from which data is collected, suffer from End-stage renal disease. Thereafter we discuss different types of missing data and how the different missing data mechanisms affect the imputation procedure. We continue by describing the data at hand in terms of distributions and missingness pattern as well as some important metrics to understand how the variables are connected. We end the chapter by describing the three data sets *original data*, *validation data*, and *synthetic data*.

2.1 Nephropaty

The kidney is a structurally complex organ with many important functions. It is responsible for the excretion of waste products from the metabolism, maintenance of appropriate acid balance, regulation of body water and salt, and secretion of a variety of hormones. Nephropaty is an umbrella term used to denote disease or damage to the kidney.

2.1.1 Chronic kidney disease and End-stage renal disease

Chronic kidney disease (CKD) is a gradual impairment in kidney function. It can be divided into five stages with increasing severity, described in Table 2.1 [15]. The two main causes of the disease are declining kidney function with age and lowered kidney function due to other diseases [16]. Thus, elderly patients with e.g. diabetes mellitus are particularly vulnerable.

To classify the functionality of the kidney, the glomerular filtration rate (GFR) is calculated. GFR is a measure of how much fluid the kidney can filter per minute, or more specifically, how much volume of fluid filtered from the renal glomerular capillaries into the Bowman's capsule per unit time [17], see Figure 2.1.

End-stage renal disease (ESRD) is defined as an irreversible decline in a person's kidney function [18]. It is the fifth and most severe stage of chronic kidney disease and patients that suffer from ESRD are in desperate need of dialysis or a kidney transplantation to survive. All data analyzed in this project was from patients suffering from ESRD.

Stage	Qualitative Description	Renal Function ($ml/min/1.73m^2$)
1	Normal GFR	≥ 90
2	Mild decay in GFR	60-89
3	Moderate decay in GFR	30-59
4	Severe decay in GFR	15-29
5	End-stage renal disease	15

Table 2.1: The five stages of Chronic Kidney Disease.

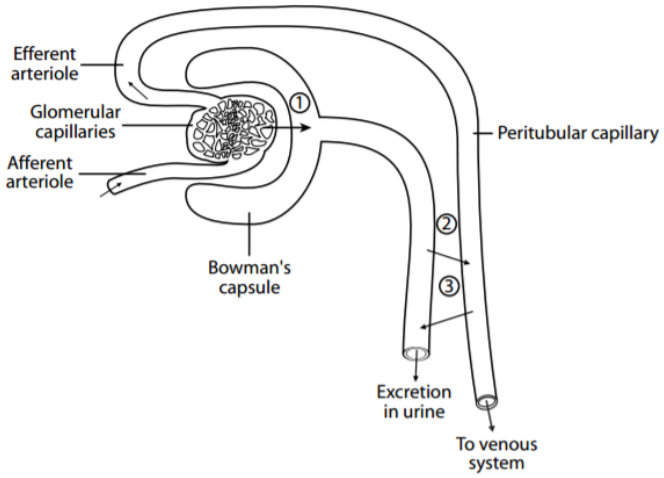


Figure 2.1: The basic renal process in a nephron. (1) Fluid and solutes are filtered through the Glomerular capillaries to Bowman's capsule. (2) Some of the filtered substances are reabsorbed by moving out of the renal tubule and into the peritubular capillaries. (3) Unfiltered substances from the peritubular capillaries are moved to the renal tubule where it is then excreted. Image from *Essentials of Human Physiology for Pharmacy* [17, Chapter 19].

2.1.2 Dialysis

When the renal function is below a certain filtration rate, the kidneys cannot remove waste products and excess water from the blood fast enough to keep a person healthy. To be able to survive this condition one can either get a kidney transplant or go on regular dialysis treatments. None of the alternatives are optimal and both, inevitably, bring complications which forces a change of lifestyle.

A dialysis treatment is typically carried out every second or third day for around four hours a time. This is due to the excess fluid, salt, and waste products a person with low kidney function accumulate each day. The dialysis process is time consuming due to the fact that the excess fluid stored in the cells diffuse from the cell to the blood vessels. This process is governed by the internal diffusion constant between different tissues and cannot be affected [19].

Since dialysis is carried out frequently and on a regular basis it is easy to keep track of the progress of the patient's situation. Many parameters are recorded and monitored during each

treatment and hence a vast amount of data can be collected for analysis. There is, however, often some inconsistency in the quality of the data and sometimes all measurements are not carried out, leaving the data with gaps of missing data, which will be discussed further on.

2.2 Missing data

There are several reasons to why data can be missing in a data set containing medical data. For example age is a variable that influences the rate of missing observations. The elderly may experience difficulty getting to the treatment site or may move to retirement homes or hospices, thus missing examinations [12]. The most likely reason for data from dialysis patients to be missing is arguably that the nurses sometimes forget to record all variables and/or that the machine fails to collect all data [14]. Also, various complications may arise making it impossible to carry out all measurements.

When a data analyst is presented with a data set containing missing values there are multiple ways to handle the abnormality. Many factors have to be considered before taking action and one has to ponder over the quality of the data; e.g. how much data is missing and connections between variables, to name a couple. To handle the missingness of the data, an easy, and sometimes quite good, action is to just ignore all measurements taken during the same time as the missing value occurs, called complete case analysis. This however, often comes at the price of loosing valuable information about the patient. Especially, if the amount of missing data is large, complete case analysis is a bad idea.

A much better method would be to try to estimate the values of the missing variables by using the information of the existing variables. This would yield a reconstructed data set with more plausible values, which in turn could be used for further analysis. In this thesis, different such imputation methods are discussed and evaluated on data from dialysis patients.

The underlying mechanism of the missing data points is of interest when performing imputation of missing data. We illustrate this by giving an example: assume that the blood pressure is measured less frequently when a patient has low blood pressure. We are interested in using the data to predict whether or not the patient will fall ill - with hypotonia as an indicator - and if the rows containing missing values simply are removed, we would loose important information which otherwise could have been used to tell if the patient will fall ill or not. This example strongly encourage the use of imputation methods to complete data sets on which further analysis will be of interest. The example also suggests that data may be missing in different ways, affecting which imputation method is best to use. We continue by discussing different types of missing data mechanisms.

2.2.1 Types of missing data

In this section we provide a more solid ground for the mechanisms that leads to missing data, as well as defining the three types of missingness: missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR) [11]. The best imputation method to use is strongly affected by what class the missing data belongs to.

We begin by defining the entire data set $Y = (y_{ij})$ and the indicator matrix $R = (r_{ij})$ with binary entries $(0, 1)$, where $r_{ij} = 0$ if y_{ij} is missing and $r_{ij} = 1$ if y_{ij} is observed. If $r_{ij} = 1$ then $y_{ij} \in Y_{obs}$ and if $r_{ij} = 0$ then $y_{ij} \in Y_{mis}$. The distribution of R may depend on $Y = (Y_{obs}, Y_{mis})$, called the missing data mechanism (MDM) which is described by the missing data model. Let ψ contain the generally unknown parameters of the missing data model, we can then form the

posterior distribution of R as $P(R \mid Y_{obs}, Y_{mis}, \psi)$. If the missingness does not depend on the values of Y, that is if

$$P(R = 0 \mid Y_{obs}, Y_{mis}, \psi) = P(R = 0 \mid \psi)$$

holds, then the data is said to be MCAR. To assume data to be MCAR is a restrictive assumption, and usually unrealistic. A less restrictive assumption is that the missingness only depends on the observed part of the data Y_{obs}, that is if

$$P(R = 0 \mid Y_{obs}, Y_{mis}, \psi) = P(R = 0 \mid Y_{obs}, \psi)$$

holds, then the data is said to be MAR. If the probability to be missing also depends on Y_{mis}, that is if

$$P(R = 0 \mid Y_{obs}, Y_{mis}, \psi)$$

does not simplify, the data is said to be NMAR.

2.2.2 Ignoring the missing data mechanism

An important question to answer when imputations are made is if the missing data and observed data come from the same distribution. If that is the case we can simplify our model of the missing data and build the model solely on observed data. The observed data consists of Y_{obs} and R with the joint density function $f(Y_{obs}, R \mid \theta, \psi)$ depending on parameters θ for the entire data Y, which we are interested in estimating; and parameters ψ for the indicator R, which is seldom of interest. Since we are not interested in ψ we would like to know when we can estimate θ without knowing ψ. To answer the question we start by defining the probability density function of the joint distribution of Y_{obs} and Y_{mis} as $f(Y \mid \theta) \equiv f(Y_{obs}, Y_{mis} \mid \theta)$. We can compute the marginal probability density of Y_{obs} by integrating out the missing data as

$$f(Y_{obs} \mid \theta) = \int f(Y_{obs}, Y_{mis} \mid \theta) dY_{mis}$$

and the likelihood of θ based on Y_{obs} ignoring the missing data mechanism as

$$L_{ign}(\theta \mid Y_{obs}) \propto f(Y_{obs} \mid \theta). \tag{2.1}$$

From which we can obtain maximum likelihood (ML) estimates of θ by maximizing over θ provided that the missing data mechanism can be ignored. To build a more general model we include R into the model and specify the joint density distribution of Y and R as

$$f(Y, R \mid \theta, \psi) = f(Y \mid \theta) f(R \mid Y, \psi),$$

with θ and ψ defined as before. Once again we can find the distribution of the observed data by integrating Y_{mis} out of the joint density as,

$$f(Y_{obs}, R \mid \theta, \psi) = \int f(Y_{obs}, Y_{mis} \mid \theta) f(R \mid Y_{obs}, Y_{mis}, \psi) dY_{mis} \tag{2.2}$$

and the likelihood of θ and ψ based on Y_{obs} and R including the missing data mechanism as

$$L_{full}(\theta, \psi \mid Y_{obs}, R) \propto f(Y_{obs}, R \mid \theta, \psi). \tag{2.3}$$

If we can determine when to base inference for θ on Equation 2.1 instead of on Equation 2.3 we have also answered the question when the missing data mechanism is ignorable. We observe that if the distribution of the missing data mechanism does not depend on the missing values Y_{mis}, we can write

$$f(R \mid Y_{obs}, Y_{mis}, \psi) = f(R \mid Y_{obs}, \psi), \qquad \forall\, Y_{mis}, \tag{2.4}$$

then we can rewrite Equation 2.2 as

$$f(Y_{obs}, R \mid \theta, \psi) = f(R \mid Y_{obs}, \psi) \int f(Y_{obs}, Y_{mis} \mid \theta) dY_{mis} = f(R \mid Y_{obs}, \psi) f(Y_{obs} \mid \theta). \tag{2.5}$$

As discussed in Section 2.2.1 the missing data are called MAR when Equation 2.4 holds (Rubin, [11]). From Little and Rubin [8] we get the following definition:

The missing data mechanism is ignorable for likelihood inference if:

1. MAR: the missing data are missing at random; and
2. Distinctness: the parameters θ and ψ are distinct, in the sense that the joint parameter space of (θ, ψ) is the product of the parameter space of θ and the parameter space of ψ.

The MAR requirement is generally considered to be the more important condition and for all practical purposes the missing data mechanism is said to be ignorable if MAR holds. We have not discussed the second condition, distinctness, but settle the matter by reffering to Schafer [9], who says "In many situations this is intuitively reasonable, as knowing θ will provide little information about ψ and vice-versa."

2.2.3 Applications of ignorability

It is important to understand that the observed data consists of both observed values, Y_{obs}, and the indicator matrix, R. If we can assume ignorability and exclude R from the posterior distribution $P(Y_{mis} \mid Y_{obs}, R)$ we can use much simpler models from which Y_{mis} is imputed. An important implication of the concept of ignorability is if

$$P(Y_{mis} \mid Y_{obs}, R) = P(Y_{mis} \mid Y_{obs})$$

holds; it implies that

$$P(Y \mid Y_{obs}, R = 1) = P(Y \mid Y_{obs}, R = 0),$$

meaning that the distribution of the data Y is identical for the observed data and missing data. Hence, we can build our model for the posterior distribution $P(Y \mid Y_{obs}, R = 1)$ given the observed data, i.e. an imputation model can be built using only the observed values since all the descriptive information exists in the observed data.

2.3 Data provided

In collaboration with the Centers for Dialysis Care (CDC), Lytics Health AB has access to a vast amount of dialysis data. The data provided for this study was in the form of a multivariate time series, which consisted of over 2 million measurements of 92 different variables, giving a total of over 185 million data points. Out of these 92 different variables, eleven were in most need of imputation, giving a total amount of about 22 million data points to work with. Below, the eleven variables of interest are listed. Note the abbreviations which are used later in tables and figures.

- Average blood flow rate [ml/min] (ABFR)
- Start sitting pulse [bpm] (SSP)
- Liters processed [l] (LP)
- Time dialyzed [min] (TD)
- Fluid removed [l] (FR)
- Patient temperature at start [°C] (PTS)
- Patient temperature at end [°C] (PTE)
- Start sitting systolic blood pressure [mmHg] (SSSBP)
- Start sitting diastolic blood pressure [mmHg] (SSDBP)
- End sitting systolic blood pressure [mmHg] (ESSBP)
- End sitting diastolic blood pressure [mmHg] (ESDBP)

2.3.1 Data acquisition

The process of acquiring dialysis data and an example of how the data collection for one patient might look like is shown in Figure 2.2. The image consists of three parts:

1. Many of the patients included in the data have undergone dialysis before collection of data at CDC began, meaning that all of the patients history is not present in our data. This is depicted in the image by the lists in the beginning of the time axis.
2. Typically the patients undergo three treatments per week which leaves the time series with unevenly sampled gaps. That is the data analyzed in this project. This is shown in the middle section of the image.

3. Sometimes patients have to be hospitalized for a longer period of time due to an acute illness. During these periods no data is collected, which leaves our data with gaps. The reason for this is that CDC does not have access to the patients, or their data, when they are hospitalized. This is shown by the red zone to the right in the image.

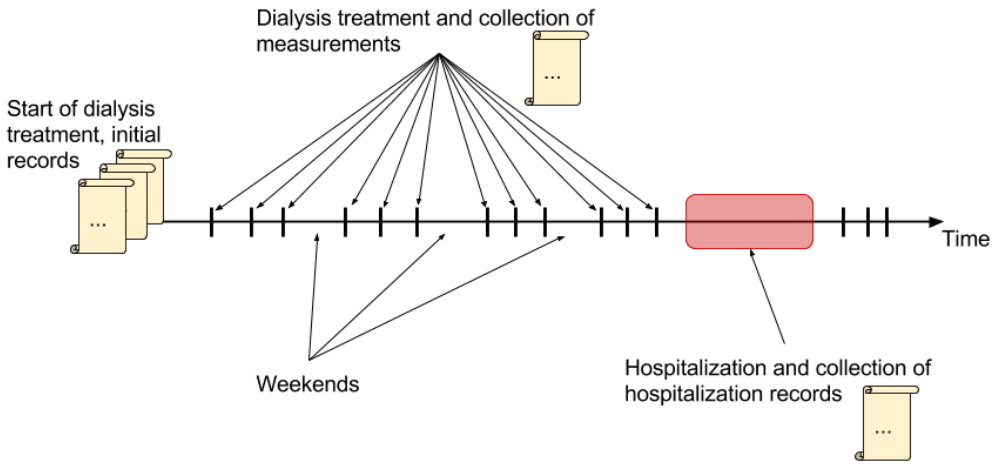


Figure 2.2: Process of acquiring data.

2.3.2 Distribution of data

It is also of interest to see how many different patients the data consists of, and how many treatments each patient has participated in. Table 2.2 shows some basic statistics about the data and Figure 2.3 shows the distribution of sequence lengths.

Number of unique patients	5786
Mean sequence length	297
Median sequence length	186

Table 2.2: Basic data statistics.

As can be seen, the data consisted of a lot of short sequences. This was unfortunate since it is often hard to say anything about a patient who has only just started his or her treatment. Different imputation methods are also more or less dependent on sequence lengths to perform well.

The variables would preferably be normally distributed which many statistical models rely on. This, however, is seldom the case when handling real data and sure enough that was not the case for some of our variables either. Below are two histogram plots of the variables *AverageBloodFlowRate* and *StartSittingPulse*, showing the distributions with a comparison to a normal distribution curve fitted to the data, see Figure 2.4. As can be seen, *AverageBloodFlowRate* is not normally distributed, but rather follows a multimodal Gaussian distribution. Whereas *StartSittingPulse* is more similar to a Gaussian distribution. The rest of the distributions of the variables can be found in Appendix A.

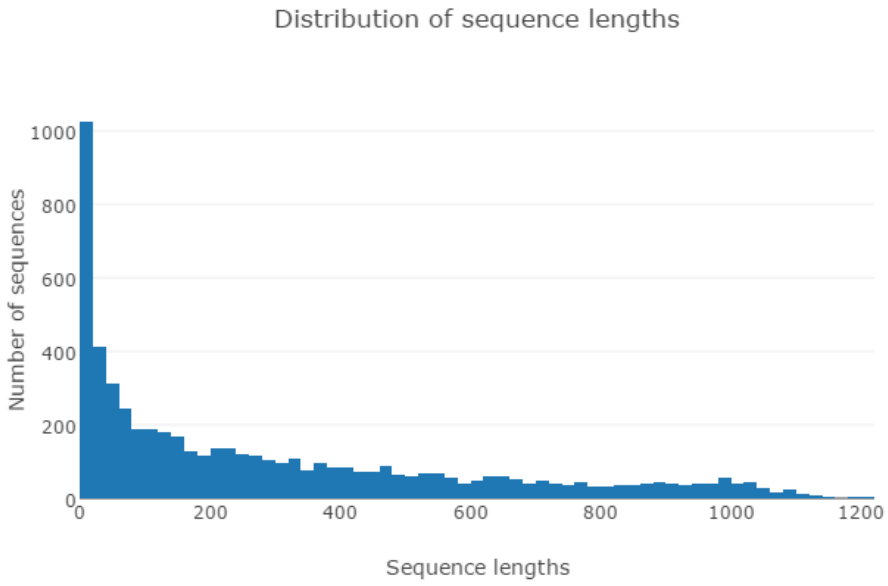
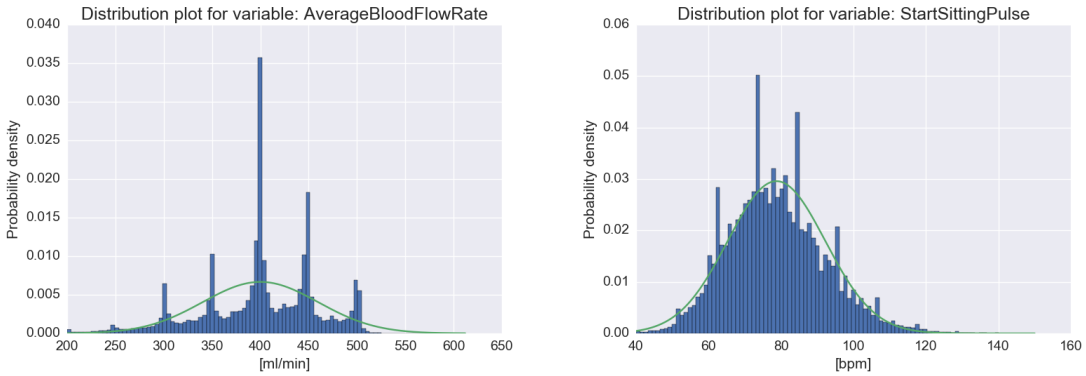


Figure 2.3: Histogram of sequence lengths (number of treatments).



(a) (b)

Figure 2.4: Distribution of two variables, with a normal curve (green) for reference.

Here, one can clearly see that the average blood flow rate is a variable which is preset on the dialysis machine in gaps of 50 ml/min, giving the characteristic spikes of the histogram in Figure 2.4a.

To better understand how the values are distributed in the different variables, a box plot was created, see Figure 2.5. The bottom and top of the boxes are the first and third quartiles and the

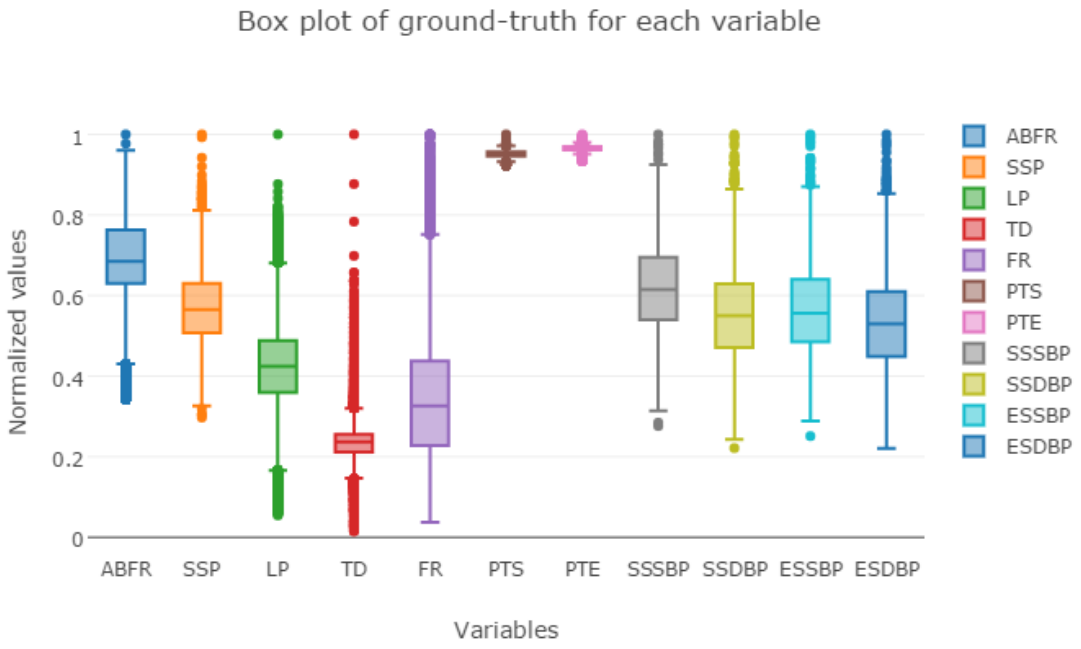


Figure 2.5: Box plot of variables in validation data. Normalized by dividing each variable with its maximum value. See text for explanation.

line inside the box is the second quartile (i.e. the median). The ends of the vertical lines (whiskers) extending from the boxes are calculated by taking 1.5 times the interquartile range (i.e. the height of the box). The points above and below the whiskers represents outliers.

2.3.3 Missing data patterns

Missing data patterns can be classified into different types. A missing data pattern is said to be univariate if there is only one variable containing missing data, and multivariate if there are several variables containing missing data. The data analyzed in this project had a multivariate missing data pattern.

In the raw data provided, about 18% of the rows had at least one value missing. We discovered that in many of these rows all variables were missing which, to us, seemed peculiar. This specific missingness pattern was investigated further and we found that all the occasions where the entire row was missing was due to missed treatments. There could be many different reasons for missing a treatment but the most frequent ones were due to hospitalizations, vacations, and personal reasons. Unfortunately, we have no access to the measurements at the hospitals and therefore there is no interest in imputing them. We decided to remove all the rows with missed treatments since there was no way to validate imputations on occasions that did not occur. This yielded a data set where about 15% of the rows had at least one missing value. However, some wanted characteristics of the

data were distorted, i.e. by removing 3% of the rows, the time series characteristics of the variables are altered. On the other hand we have no access to the measurements and imputing the missing values due to hospitalization would not be preferable. The missingness pattern of the data set with hospitalizations removed is shown in Figure 2.6.

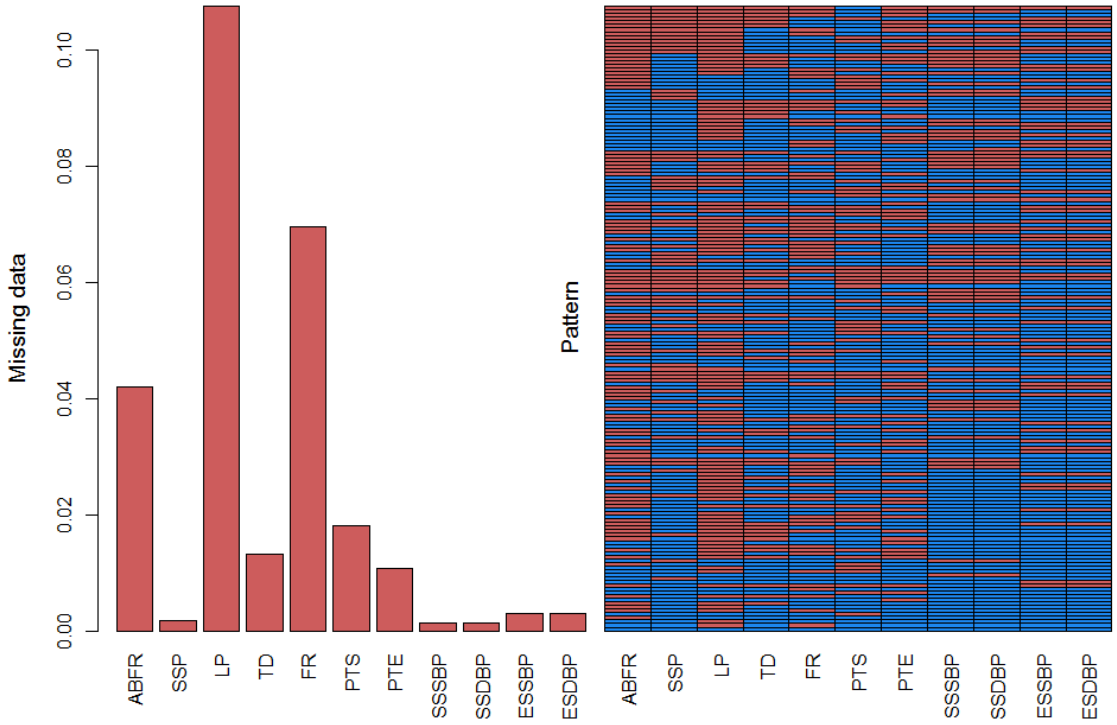


Figure 2.6: Left: frequency of missingness in each variable. Right: Observed missingness patterns in the data set. At the top of the plot the least frequent occurring patterns are located, with gradually increasing frequency towards the bottom. The last row contains only observed values and constitutes 86% of the data. Blue: observed, Red: missing.

The most frequent missing variable is *LitersProcessed* and the most frequent occurring pattern is when *LitersProcessed* and *FluidRemoved* is missing together. Also, we see that the pairs of blood pressure variables are always missing at the same time.

2.3.4 Analyzing connections between variables

From Figure 2.6 one can tell that there are many different types of patterns in which data can be missing. However, just knowing the missingness pattern is not enough. It is of great interest to know when variables are missing or occurring at the same time or how the variables are correlated. The different metrics below have all been used in some way to acquire more information about the data and about the connections between the variables.

Proportion of usable cases

Before deciding an appropriate approach for handling a missing value problem, it is good to get some feeling for how each variable connects to the others. One way of doing this is to calculate the proportion of usable cases [10]. The proportion of usable cases is a measurement of how often variables are present (or missing) at the same time in a multivariate data set. It can be used to get an initial implication of what kind of imputation method (if any) is suitable for the specific data. If a variable A always is present when another variable B is missing, it seems reasonable to think that a model using characteristics from A could be used to impute missing values in B. On the other hand if A and B are missing simultaneously, they would probably not be good at imputing one another. Van Buuren [10], defined the measurement as

$$I_{jk} = \frac{\sum_i^n (1 - r_{ij}) r_{ik}}{\sum_i^n (1 - r_{ij})}, \tag{2.6}$$

where r_{ij} is indicating if the value at row i and column j is present or missing and r_{ik} indicates if the value at row i and column k is present or missing. Equation 2.6 can be interpreted as the number of variable pairs (Y_j, Y_k) with Y_j missing and Y_k observed, divided by the total number of missing cases in Y_j; i.e. if Y_k is always present when Y_j is missing, then $I_{jk} = 1$.

VAR	ABFR	SSP	LP	TD	FR	PTS	PTE	SSSBP	SSDBP	ESSBP	ESDBP
ABFR	0.00	0.99	0.38	0.71	0.65	0.93	0.93	0.99	0.99	0.95	0.95
SSP	0.70	0.00	0.66	0.74	0.82	0.88	0.96	0.65	0.65	0.97	0.97
LP	0.73	0.99	0.00	0.87	0.34	0.96	0.97	1.00	1.00	0.98	0.98
TD	0.02	0.97	0.00	0.00	0.45	0.81	0.80	0.99	0.99	0.86	0.86
FR	0.78	1.00	0.02	0.90	0.00	0.96	0.96	1.00	1.00	0.97	0.97
PTS	0.82	0.99	0.78	0.86	0.83	0.00	0.75	0.99	0.99	0.91	0.91
PTE	0.71	0.99	0.67	0.75	0.75	0.57	0.00	1.00	1.00	0.82	0.82
SSSBP	0.82	0.56	0.77	0.88	0.87	0.89	0.98	0.00	0.00	0.98	0.98
SSDBP	0.82	0.56	0.77	0.88	0.87	0.89	0.98	0.00	0.00	0.98	0.98
ESSBP	0.33	0.99	0.32	0.38	0.38	0.47	0.37	0.99	0.99	0.00	0.00
ESDBP	0.33	0.99	0.32	0.38	0.38	0.47	0.37	0.99	0.99	0.00	0.00

Figure 2.7: Proportion of usable cases between variables. The numbers represent how often a variable is present while another variable is missing. A large number (blue) means that when the variable on the Y-axis is missing, often the variable on the X-axis is present. Conversely, a small number (red) means that when the variable on the Y-axis is missing, so is often the variable on the X-axis. Variables in between are colored violet. The abbreviations of the variables are explained in the beginning of this section.

From Figure 2.7 one can, for example, deduct that when the variable *TimeDialyzed* is missing, so are also always the variables *AverageBloodFlowRate* and *LitersProcessed*. Therefore a model using these two variables might be unsuitable to impute missing values for *TimeDialyzed*.

Correlation between variables

Correlation is a measure of the statistical relationship between two variables. It can give a hint on what methods to use for imputation, i.e. if variables have high correlation between each other it might be suitable to use models that exploit information between variables. Here the Pearson product-moment correlation coefficient has been used as a measure. It is derived by dividing the covariance of two variables by the product of their standard deviation,

$$\rho_{Y_1,Y_2} = \frac{\mathrm{C}[Y_1, Y_2]}{\sigma_{Y_1}\sigma_{Y_2}}, \tag{2.7}$$

where C is the covariance between the variables and σ is the standard deviation of each variable. The correlation coefficient approaches 1 or -1 in the case of a perfect linear relationship. Values closer to zero indicate that the relationship between the two variables is low, i.e. no linear relationship. Figure 2.8 shows the correlation between all eleven variables. As can be seen there is mostly low correlation, but for the blood pressure variables.

VAR	ABFR	SSP	LP	TD	FR	PTS	PTE	SSSBP	SSDBP	ESSBP	ESDBP
ABFR	1.00	0.01	0.65	0.05	0.17	-0.03	-0.02	0.06	0.08	-0.01	0.01
SSP	0.01	1.00	0.04	0.04	0.06	0.10	0.08	0.00	0.22	-0.08	0.11
LP	0.65	0.04	1.00	0.56	0.37	0.01	0.04	0.02	0.07	-0.02	0.04
TD	0.05	0.04	0.56	1.00	0.34	0.02	0.05	-0.02	0.03	-0.03	0.03
FR	0.17	0.06	0.37	0.34	1.00	0.00	0.05	0.10	0.15	-0.06	0.05
PTS	-0.03	0.10	0.01	0.02	0.00	1.00	0.50	0.02	0.05	0.03	0.06
PTE	-0.02	0.08	0.04	0.05	0.05	0.50	1.00	0.04	0.09	0.05	0.09
SSSBP	0.06	0.00	0.02	-0.02	0.10	0.02	0.04	1.00	0.64	0.53	0.38
SSDBP	0.08	0.22	0.07	0.03	0.15	0.05	0.09	0.64	1.00	0.36	0.53
ESSBP	-0.01	-0.08	0.02	-0.03	-0.06	0.03	0.05	0.53	0.36	1.00	0.64
ESDBP	0.01	0.11	0.04	0.03	0.05	0.06	0.09	0.38	0.53	0.64	1.00

Figure 2.8: Correlation between variables. The numbers represent correlation between two variables. A large number (blue, $\rho \geq 0.75$) means that the correlation is high. Conversely a small number (red, $\rho \leq 0.25$) means that the correlation is low. Variables in between are colored violet.

Autocorrelation

For a time series it is also relevant to look for correlation within each variable, namely autocorrelation. If values in a variable have strong connections to previous values, it might be more suitable to build models which extract that information rather than use information in other variables. Figure 2.9 shows the autocorrelation for one variable, and as can be seen the correlation looks higher than for most variables in Figure 2.8. The autocorrelation plots for all other variables can be found in Appendix B.

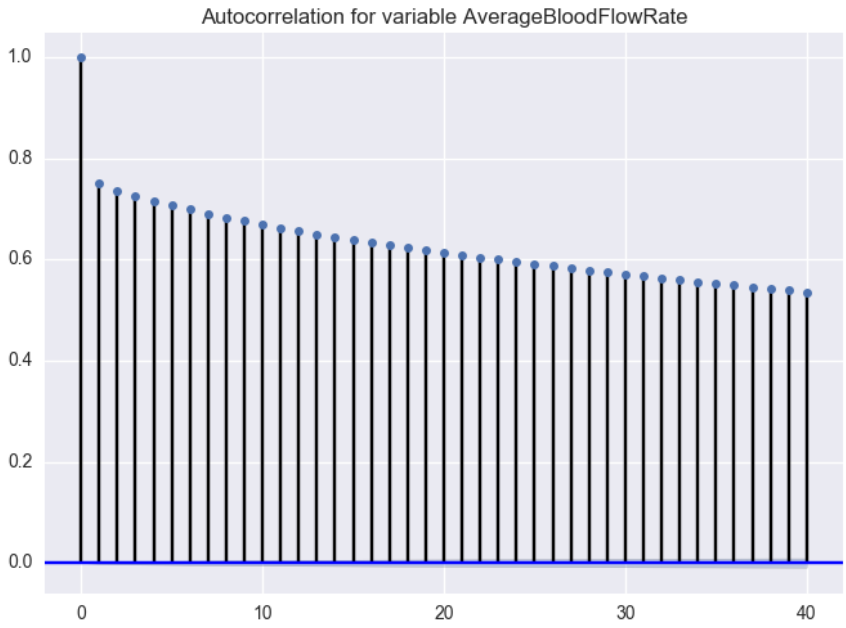


Figure 2.9: Autocorrelation of the variable *AverageBloodFlowRate* with 40 lags.

2.3.5 Different data sets

Before beginning to analyze and implement different imputation methods it was important to have a few different data sets which could be used in the analysis. It was decided that three data sets would be needed. One preprocessed version of the raw data (see Section 4.1), from now on called *original data*, one data set for validation, only containing complete rows, from now on called *validation data*, and one artificially destroyed data set which, after being imputed, could be compared to the validation data, called *synthetic data*.

Original data

The *original data* was constructed by preprocessing the raw data in several different ways, see Section 4.1. The outcome was a data set where approximately 15% of the rows of the raw data had been removed, i.e. the data set now contained around 1.72 million rows. A full explanation of all preprocessing steps is given in Section 4.1.

Validation data

The *validation data* was created by removing all rows containing missing values in the *original data*. Doing so, again reduced the number of rows in the data with approximately 15%. This final data set now contained around 1.45 million rows. This complete data set is referred to as *validation data*,

even though only some of its values (i.e. the values destroyed when creating the synthetic data) will be used for evaluation.

Synthetic data

The method of creating synthetic data to evaluate a model is widely used in different areas and studies. The *synthetic data* was created by destroying the *validation data* according to certain patterns of missingness. In this case it was important to obtain the same characteristics (missingness pattern) as that of the *original data* since the imputation methods were to be used on that specific data set. If an imputation method performs well on the *synthetic data* set which closely resembles the *original data*, it is reasonable to believe that it will also perform well on the *original data*. The *synthetic data* can be thought of as training data, since all of its non-missing values could be used to train an imputation model. For the sake of the performance of the imputation method, however, it could also be of interest to evaluate it on different rates of missing data. One method might be better if 30% of the data points are missing whilst another outperforms the rest if 10% is missing. Thus, evaluating on different data sets can be used as a robustness measure of the imputation methods. For a more thorough explanation of the process of how synthetic data was created, see Section 4.1.3.

Chapter 3

Imputation Methods

In the context of statistical literature, imputation means "filling in the data". The filling in or imputation can be done in a variety of different ways, with some of the methods discussed and described here. However, the common way to handle missing values in a data set has been to delete the rows where missing values have occurred, also know as complete case analysis. By doing so we distort the underlying distribution of the data, and information crucial to further analysis might be lost. To mitigate the problem, we rely on statistical analysis to infer reliable imputations.

We continue this section by establishing some mathematical notation, which will be consistent in the rest of the report. Much of the notation has been borrowed from Schafer [9] and Van Buuren [10]. Let the entire data set (including missing values), on which imputation will be conducted, be represented by the $n \times p$ matrix $\mathbf{Y}$; where n denotes the number of measurements (rows) and p the number of variables (columns). We can partition $\mathbf{Y}$ as $\mathbf{Y} = (\mathbf{Y}_j, \mathbf{Y}_{-j})$ where $\mathbf{Y}_j$, the response, is a $n \times 1$ vector of the j:th column in $\mathbf{Y}$, and $\mathbf{Y}_{-j}$ is a $n \times p-1$ matrix consisting of the remaining columns in $\mathbf{Y}$. Furthermore let $Y_{i,j}$, $i = 1, 2, \ldots, n$ denote an individual element of $\mathbf{Y}_j$. We will partition $\mathbf{Y}_j$ into $\mathbf{Y}_j = (\mathbf{Y}_j^{obs}, \mathbf{Y}_j^{mis})$ where all the n_{obs} observed values in $\mathbf{Y}_j$ is in $\mathbf{Y}_j^{obs}$ and all the n_{mis} missing values in $\mathbf{Y}_j$ is in $\mathbf{Y}_j^{mis}$. With $\mathbf{Y}_j$ defined we can partition $\mathbf{Y}_{-j}$ in the same way, as $\mathbf{Y}_{-j} = (\mathbf{Y}_{-j}^{obs}, \mathbf{Y}_{-j}^{mis})$, where the $1 \times p-1$ vector $\mathbf{Y}_{i,-j}$ is in $\mathbf{Y}_{-j}^{obs}$ if $Y_{i,j}$ is observed and in $\mathbf{Y}_{-j}^{mis}$ if $Y_{i,j}$ is missing. Note that the values in $\mathbf{Y}_{i,-j}$ could be observed or missing regardless of if $\mathbf{Y}_{i,-j} \in \mathbf{Y}_{-j}^{obs}$ or $\mathbf{Y}_{i,-j} \in \mathbf{Y}_{-j}^{mis}$, which subgroup $\mathbf{Y}_{i,-j}$ belongs to depends only on $Y_{i,j}$. We have thus expressed $\mathbf{Y}_i$, the i:th row of $\mathbf{Y}$ as a row vector and $\mathbf{Y}_j$, the j:th column of $\mathbf{Y}$ as a column vector, where each column is assumed to be an independent realization of a random vector $(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_p)$. $\mathbf{Y}$ is assumed to belong to a multivariate normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, as

$$\mathbf{Y}_i \mid \boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \qquad i = 1, 2, \ldots, n,$$

where $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is an unknown parameter.

3.1 Naïve imputation methods

The naïve imputation methods are all easy and quick to implement and in many programming modules, e.g. `pandas` in Python, these methods already exists as ways to fill in or delete missing

values. They can be used conveniently, and if the amount of missing data is not to large, the result can be surprisingly good.

3.1.1 Complete case analysis

In complete case analysis, also know as list-wise deletion, all measurements with missing data are removed. This way of handling missing data is easy to understand and to execute. However, a simple example will highlight the main problem of dealing with missing data in this way. Assume that 10% of the data is missing, the data set consists of $p = 10$ variables and $n = 1000$ rows, and the missing data mechanism is assumed to be MCAR. Then, in worst case, one value at each row will be missing and we will have to delete all rows; thus leaving no data to analyze. From this example we conclude that the method only works if a small amount of data is missing.

Another problem arising from list-wise deletion is if the data is captured in sequences and treated as a time series. By deleting an entire row the sequential structure is ruptured making it much harder to use time series analysis tools, e.g. identifying seasonal patterns or fit auto regressive models. Also, complete case analysis requires the missing data to be MCAR, otherwise all statistical inference will be biased [10].

3.1.2 Forward fill

One of the easiest ways to handle missing data is to simply use the latest recorded value of a variable as replacement. The perks with this method is that it is very easy to implement and that each imputed value is an actually recorded measurement. This makes the method robust and it will never impute unrealistic values. The obvious downside with this method is that should a variable vary a lot between measurements, the pattern would get distorted, i.e. data are kept constant over time when it should be varying, resulting in biased results. If the previous value is an outlier the method would also increase the distortion of the data instead of creating a more realistic data set. Forward fill is used as a baseline method in this report and thus the mission is to beat its performance.

3.1.3 Mean substitution

Another simple and straight forward way to impute the missing values of $\mathbf{Y}_j$ is to just replace them with the mean of the observed values as

$$\hat{\boldsymbol{Y}}_j^{mis} = \mathrm{E}[\mathbf{Y}_j^{obs}] = \frac{1}{n_{obs}} \sum_{i=1}^{n_{obs}} Y_{i,j}^{obs},$$

which is a convenient and easy fix. However, by conducting mean imputation we distort the distribution in several ways. Mean imputation will underestimate the standard deviation, disturb the relations between variables (correlations), introduce biased estimates of almost any estimate other than the mean and bias if the data is not MCAR [10].

3.2 Linear models

3.2.1 Regression imputation

In regression imputation we try to estimate the missing values, $\mathbf{Y}_j^{mis}$, given all the other variables $\mathbf{Y}_{-j}^{mis}$ by fitting a linear model to the observed data. If we, in a multivariate time series, have univariate missing data, i.e. only one variable contains missing data, it implicates that all values in $\mathbf{Y}_{-j}^{mis}$ is observed and we can form the well-known model

$$\hat{Y}_{i,j}^{mis} = \mathbf{Y}_{i,-j}^{mis}\hat{\boldsymbol{\beta}} = \hat{\beta}_0 + \hat{\beta}_1 Y_{i,1}^{mis} + \cdots + \hat{\beta}_{j-1} Y_{i,j-1}^{mis} + \hat{\beta}_{j+1} Y_{i,j+1}^{mis} + \cdots + \hat{\beta}_p Y_{i,p}^{mis}$$

and in matrix notation

$$\hat{\mathbf{Y}}_j^{mis} = \mathbf{Y}_{-j}^{mis}\hat{\boldsymbol{\beta}}.$$

The column vectors in $\boldsymbol{Y}_{-j}^{mis}$ are called the regressors and the elements in $\hat{\boldsymbol{\beta}}$ are called the parameters, which are least squares estimates computed from the observed data. A column of ones of length n has been added to $\mathbf{Y}_{-j}$ to account for the mean, yielding p parameters. To find the parameters we minimize the squared error given by the difference between the observed data and the p-dimensional line fitted to the data as

$$\hat{\boldsymbol{\beta}} = arg\min_{\beta} \|\mathbf{Y}_j^{obs} - \mathbf{Y}_{-j}^{obs}\boldsymbol{\beta}\|^2,$$

with the solution

$$\hat{\boldsymbol{\beta}} = (\mathbf{Y}_{-j}^{obs\top}\mathbf{Y}_{-j}^{obs})^{-1}\mathbf{Y}_{-j}^{obs\top}\mathbf{Y}_j^{obs}.$$

For real data, it is unlikely that all the missing values will be on a regression line. To account for the variability in the data more sophisticated methods are required. More theory about linear regression and least squares in the context of imputation can be found in e.g. Little and Rubin [8], Schafer [9], and Van Buuren [10].

3.3 Multivariate methods

In general, missing data is not restricted to only one variable and it is more natural to assume that missing values can appear in any of the variables. Here we will introduce a few methods on how to handle multivariate missing data.

3.3.1 k-Nearest Neighbors

The k-Nearest Neighbors (kNN) algorithm is used in many fields for classification or regression problems, but it can also be of use in the world of missing data. It can be used both as a univariate method by looking at the nearest neighbors in one variable, or as a multivariate method by checking

for nearest neighbors in other variables. The latter was chosen for this task since already mean substitution and similar univariate techniques had been tested. The methodology, as proposed by Troyanskaya et al. [20], is as follows:

- First choose how many (k) neighbors to consider.
- Normalize all variables so that they have zero mean and a variance of 1.
- Choose one variable and one missing value to impute.
- The Euclidian distances between the rest of the values in the variable to be imputed and all other variables are computed.
- The k variables with smallest Euclidian distances to the variable to be imputed are chosen as neighbors.
- A weighted mean of the values from those k variables, corresponding to the same row as the missing value, are used as imputation value.

For instance, if a study contains four variables A, B, C and D, and variable A has a missing value at row 1. Then, first all observed values in A are compared to the corresponding observed values in B, C and D, and the variables with smallest distances to A are regarded as nearest neighbors. A weighted average of the values corresponding to the same row as that of the missing value in A in the nearest neighbors, say B and C if $k = 2$, is then used for imputation.

The distances for one variable, in this case $\mathbf{Y}_1$ are calculated by using the mean squared error:

$$\mathbf{d}_1 = \frac{1}{n}\sum_{i=1}^{n}(Y_{i,1} - Y_{i,j})^2, \qquad j = 2 \dots p,$$

where $\mathbf{d}_1$ is a vector containing distances for variable $\mathbf{Y_1}$ to all other variables. Finally the k variables with shortest distances are chosen as neighbors, called $\mathbf{d}_1^*$. The weights are then calculated using

$$\mathbf{w}_1 = \frac{1}{\mathbf{d}_1^*},$$

giving a vector with k weights. This method, however, would work poorly if introduced to missing data patterns where many variables are missing at the same time or if it was introduced to mixed-type data. Another downside is that if the Euclidian distances between the variables are too big, the imputations will become bad.

3.3.2 Joint modeling and expectation maximization

In this section we explain the background to Algorithm 1 in Section 4.2.2. Joint modeling (JM) starts from the assumption that the data can be described by a multivariate distribution. Assuming ignorability (Section 2.2.2), imputations are created as draws from the distribution fitted to the observed data. The idea is as follows. For a general missing data pattern, missing data can occur anywhere in the data. This means, in practice, that the conditional distribution from which imputations are drawn varies from row to row. As an example, say that the missing

pattern of row i is $r_i = (0, 0, 1, 1)$, then we need to draw the imputations from the distribution $P_i(Y_1^{mis} Y_2^{mis} | Y_3, Y_4, \phi_{1,2})$, where $\phi_{1,2}$ represents the unknown parameters of the imputation model. Assuming data to be multivariate normal $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the ϕ-parameters are functions of $\theta = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ [9]. The sweep operator (described below) transforms θ into ϕ and allow rapid computations for the ϕ-parameters. The θ-parameters are usually unknown and need to be estimated from the observed data. The solution is to iterate imputation and parameter estimation using a general algorithm known as data augmentation [21]. This algorithm is closely related to the expectation maximization (EM) algorithm, developed by Dempster, Laird, and Rubin in 1977 [22]. At step t, the algorithm draws $\boldsymbol{Y}_{mis}$ and θ by altering the following steps:

$$\hat{\boldsymbol{Y}}_{mis}^{(t)} \sim P(\boldsymbol{Y}_{mis} \mid \boldsymbol{Y}_{obs}, \hat{\theta}^{(t-1)}),$$
$$\hat{\theta}^{(t)} \sim P(\theta \mid \boldsymbol{Y}_{obs}, \hat{\boldsymbol{Y}}_{mis}^{(t)}).$$

The first step, the imputation step, is analogous to the E-step in the EM algorithm and the second step, the posterior step, is analogous to the M-step in the EM algorithm.

In practice, the missing values are imputed using linear regression of observed values, as discussed in section 3.2.1. The regression parameters are obtained by using the sweep operator (sweeping) on the augmented covariance matrix,

$$\boldsymbol{\Sigma}^{*(t)} = \begin{pmatrix} -1 & \boldsymbol{\mu}^{\top(t)} \\ \boldsymbol{\mu}^{(t)} & \boldsymbol{\Sigma}^{(t)} \end{pmatrix}, \tag{3.1}$$

on the rows and columns corresponding to the variables where there are observed values.

The update of $\theta^{(t)}$ can be carried out in two steps [9]:

$$(\boldsymbol{\Sigma}^{(t)} \mid Y^{(t)}) \sim \mathcal{W}^{-1}(\boldsymbol{S}^{(t)}, n-1),$$
$$(\boldsymbol{\mu}^{(t)} \mid \boldsymbol{\Sigma}^{(t)}, Y^{(t)}) \sim \mathcal{N}(\bar{\boldsymbol{Y}}^{(t)}, \frac{\boldsymbol{\Sigma}^{(t)}}{n}),$$

where $(\bar{\boldsymbol{Y}}^{(t)}, \boldsymbol{S}^{(t)})$ is the sample mean and sample covariance matrix, respectively, of $\boldsymbol{Y}$ from the filled in data $\boldsymbol{Y}^{(t)}$; and $\mathcal{W}^{-1}$ is the inverted Wishart distribution [23].

The sweep operator

The sweep operator (swp) is a computationally efficient way to find the regression coefficients in linear regression. It is defined for symmetric matrices as described by Rubin and Little [8]. A $p \times p$ symmetric matrix $\mathbf{G}$ is said to be swept on row and column k if it is replaced by another symmetric matrix $\mathbf{H}$ with elements defined as

$$\mathbf{H} = \text{swp}[k]\mathbf{G},$$

where

$$\begin{aligned} h_{kk} &= \frac{-1}{g_{kk}}, \\ h_{jk} = h_{kj} &= \frac{g_{jk}}{g_{kk}}, \qquad j \neq k, \\ h_{jl} &= g_{jl} - \frac{g_{jk}g_{kl}}{g_{kk}}, \qquad j \neq k, l \neq k. \end{aligned}$$

There is also an operator inverse to sweep that turns predictor variables into outcome variables, called reverse sweep (rsw) and is defined by

$$\mathbf{H} = \text{rsw}[k]\mathbf{G},$$

where

$$\begin{aligned} h_{kk} &= \frac{-1}{g_{kk}}, \\ h_{jk} = h_{kj} &= -\frac{g_{jk}}{g_{kk}}, \qquad j \neq k, \\ h_{jl} &= \frac{g_{jk}g_{kl}}{g_{kk}}, \qquad j \neq k, l \neq k. \end{aligned}$$

Both operators are commutative and the inverse sweep is the inverse operator to sweep; that is

$$\text{rsw}[k]\text{swp}[k]\mathbf{G} = \text{swp}[k]\text{rsw}[k]\mathbf{G} = \mathbf{G}.$$

As an example, suppose we have a sample of n observations on 2 variables y_1 and y_2. Let $\mathbf{G}$ denote the $(2+1) \times (2+1)$ matrix

$$\mathbf{G} = \begin{bmatrix} 1 & \bar{y}_1 & \bar{y}_2 \\ \bar{y}_1 & n^{-1}\sum y_1^2 & n^{-1}\sum y_2 y_1 \\ \bar{y}_2 & n^{-1}\sum y_1 y_2 & n^{-1}\sum y_2^2 \end{bmatrix},$$

where $\bar{y}_1$ and $\bar{y}_2$ are the sample means. We index the rows and columns 0 to 2 to let variable y_1 correspond to row 1 and variable y_2 correspond to row 2. Sweeping $\mathbf{G}$ on row and column 0 yields

$$\mathbf{\Sigma}^* = \text{swp}[0]\mathbf{G} = \begin{bmatrix} -1 & \bar{y}_1 & \bar{y}_2 \\ \bar{y}_1 & s_{11} & s_{21} \\ \bar{y}_2 & s_{12} & s_{22} \end{bmatrix},$$

where s_{jk} is the sample covariance. Note that this matrix is identical to the augmented covariance matrix in Equation 3.1. Sweeping $\mathbf{\Sigma}^*$ on row and column 1 yields the symmetric matrix

$$\text{swp}[0,1]\mathbf{G} = \text{swp}[1]\mathbf{\Sigma}^* = \begin{bmatrix} -(1+\bar{y}_1^2/s_{11}) & \bar{y}_1/s_{11} & \bar{y}_2 - s_{12}\bar{y}_1/s_{11} \\ \bar{y}_1/s_{11} & -1/s_{11} & s_{12}/s_{11} \\ \bar{y}_2 - s_{12}\bar{y}_1/s_{11} & s_{12}/s_{11} & s_{22} - s_{12}^2/s_{11} \end{bmatrix}.$$

From column 2 of the matrix we are provided with the intercept ($\bar{y}_2 - s_{12}\bar{y}_1/s_{11}$) and the slope ($\beta$) of the regression of Y_2 on Y_1 (s_{12}/s_{11}). Hence, by sweeping $\mathbf{\Sigma}^*$ on the row and column corresponding to y_1 we have obtained the slope parameter connecting Y_2 to Y_1 [8]. These parameters are then used in the imputation step of the joint modeling algorithm.

3.3.3 Fully conditional specification

In Section 3.3.2 we discussed how imputations can be made by a multivariate, often normal, model. The central problem was to find the multivariate distribution of θ. Fully conditional specification (FCS) is closely related to this procedure with the main difference that each variable is modeled with a specific distribution. With FCS we want to obtain a posterior distribution of θ by sampling iteratively from conditional distributions of the form

$$\begin{aligned} &P(Y_1 \mid Y_{-1}, \theta_1), \\ &\vdots \\ &P(Y_p \mid Y_{-p}, \theta_p). \end{aligned}$$

The parameters $\theta_1, \ldots, \theta_p$ are treated as specific to the respective conditional distribution and does not have to be the product of a factorization of the true joint distribution $P(Y \mid \theta)$. The t:th iteration of the method consists of the following successive draws

$$\begin{aligned} \hat{\theta}_1^{(t)} &\sim P(\theta_1 \mid Y_1^{obs}, \hat{Y}_2^{(t-1)}, \ldots, \hat{Y}_p^{(t-1)}), \\ \hat{\mathbf{Y}}_1^{(t)} &\sim P(Y_1 \mid Y_1^{obs}, \hat{Y}_2^{(t-1)}, \ldots, \hat{Y}_p^{(t-1)}, \hat{\theta}_1^{(t)}), \\ &\vdots \\ \hat{\theta}_p^{(t)} &\sim P(\theta_p \mid Y_p^{obs}, \hat{Y}_1^{(t)}, \ldots, \hat{Y}_{p-1}^{(t)}), \\ \hat{\mathbf{Y}}_p^{(t)} &\sim P(Y_p \mid Y_p^{obs}, \hat{Y}_1^{(t)}, \ldots, \hat{Y}_{p-1}^{(t)}, \hat{\theta}_p^{(t)}). \end{aligned}$$

FCS has several important practical advantages over joint modeling. The most important being that FCS allows for flexible multivariate models because it splits a p-dimensional problem into p one dimensional problems. However, there are some drawbacks. (1) One has to specify a conditional density for each variable, separately. Thus, if the number of variables is large, substantial effort might be needed to find the appropriate model for each variable. (2) FSC is often computationally exhaustive compared to joint modeling, e.g. one cannot apply the sweep operator to increase the speed of computation. (3) Little is known about the quality of the imputations because the joint distribution may not exist theoretically [24]. However, simulation studies have confirmed that, in practice, FCS generates plausible imputations; even under complex missing data patterns [24]. The implementation of FCS will be described in Section 4.2.5 where it is used in the MICE imputation, see Algorithm 2.

3.4 Machine learning algorithms

Machine learning algorithms are applicable in most areas within statistics and data analysis. They have been shown to achieve great results in many areas including pattern recognition, classification problems and computer vision. It is therefore reasonable to suspect that they might perform well on imputation tasks as well.

The main idea with machine learning is to construct algorithms that can learn from a subset of the data and thereafter make predictions on the rest of the data. Once a model has been trained it can be introduced to similar data and classify it [25]. Applying this to imputation problems, it should be possible to train a model on observed data and then use that model to predict the values of the missing data.

3.4.1 Decision trees

The goal of a regression tree model is to predict a target value, based on multiple input variables. This is easiest illustrated using only two input variables as seen in Figure 3.1. The data is partitioned into different branches depending on if conditions at the nodes are satisfied or not. Observations satisfying the condition at each junction are assigned to the right branch, and the others to the left branch [26]. Each junction is called a node and the terminal nodes are called the leaves of the tree, which corresponds to the different outcome possibilities (regions). It is straight forward to follow the threshold at each node, finally reaching a leaf that hopefully corresponds to a constant response value close to that of the validation data.

Two important features when using decision trees for imputations are that they can handle both continuous and categorical variables and that they are non-parametric, which make them easy to implement without having to account for probability distributions of the variables (which can be a daunting task if the variables does not belong to a well-known distribution). For multivariate imputation, a univariate approach can be used fitting one model to each variable. The methodology is as follows:

- First extract the rows of the observed and missing values of the variable to be imputed.
- Train a regressor using all values corresponding to the observed rows in all variables except the variable to be imputed, using the observed values in the variable to be imputed as target values.
- Predict the missing values in the variable to be imputed by using the trained model with all values in the rest of the variables corresponding to the missing rows in the variable to be imputed.

Since the interpretation of the trees are not vital when used for imputation, there is no need to prune them. Also a bigger tree generally creates lower bias, which is of interest in imputations.

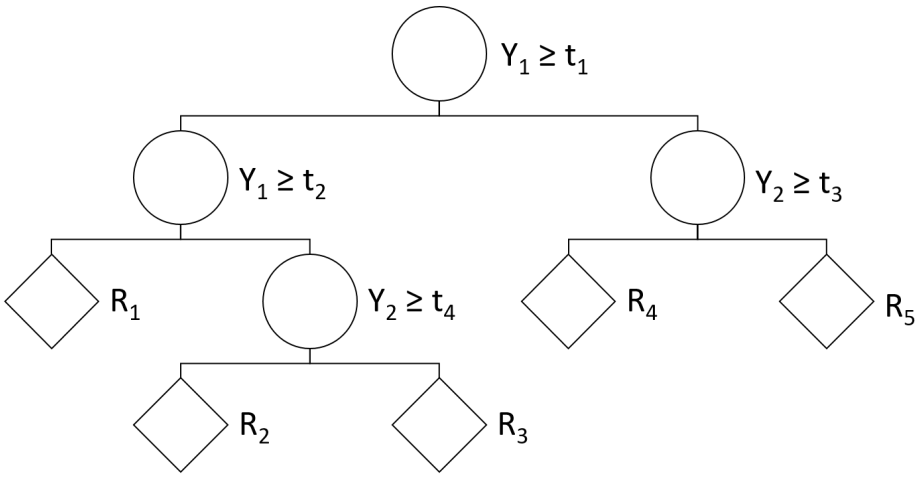


Figure 3.1: Example of regression tree with two variables Y_1 and Y_2. The thresholds t_i decide where to split the nodes and the leaves R_i corresponds to final outcome.

3.4.2 Random forest

Often, it is found that the performance of an algorithm can be improved by combining multiple models instead of sticking to one single model. Applying this reasoning to decision trees brings us to the next topic. When increasing the number of decision trees to create an ensemble of trees, a so called Random forest is created [27]. The idea behind random forests is to train multiple decision trees on different parts of the training set and to let them vote for the best fitting value. The goal is to construct a large collection of de-correlated trees and then compute the average of them, thus preventing overfitting and reducing variance [26]. Applying random forests to imputation tasks is done in the same way as for the decision trees, and since now multiple trees are used it is a form of multiple imputation, discussed in Section 3.6. An algorithm using random forests for imputation, called *missForest*, was proposed by Stekhoven and Bühlmann [28], see Algorithm 3. The implementation of that algorithm can be seen in Section 4.2.6.

3.5 Time series modeling

Due to the nature of the data, a patient is followed over time, an attractive approach is to model the data as a time series. A reasonable assumption is that a value in one variable is dependent on previous values in that same variable. The idea of time series modeling is to exploit this dependency. Conceptually, this is done by treating the signals, y_t[1], as the output of some linear system h_t, driven by random noise, e_t. The hard part is to find a good model of h_t. To find a good model an identification procedure is involved where the parameters describing the system is estimated, forming the estimated system $\hat{h}_t$. The measured signal is then filtered through the inverse system, $\hat{h}_t^{-1}$, in order to find the estimated input signal $\hat{e}_t$. The reason for doing so is to examine if the input signal is completely random or if there is any structure left that could be

[1]Standard notation in time series literature is to use t as the index for time. To be consistent with such notation we use $t = (1, \ldots, n)$ for the time series models instead of i.

exploited to further improve the model. In the sense of imputation this approach can be used to first build a model for the data at hand and then use the estimated model to impute the missing values. More information about time series modeling can be found in e.g. "An Introduction to Time Series Modeling" by Jakobsson [29].

3.5.1 ARIMA-models

There are two basic linear processes, namely the moving average (MA) process and autoregressive (AR) process which, if combined, form the autoregressive moving average (ARMA) process defined as

$$y_t + a_1 y_{t-1} + a_2 y_{t-2} + \cdots + a_p y_{t-p} = e_t + c_1 e_{t-1} + c_2 e_{t-2} + \cdots + c_q e_{t-q},$$

or equivalent

$$A(z)y_t = C(z)e_t,$$

where $A(z)$ and $C(z)$ are monic polynomials of order p and q, respectively,

$$\begin{aligned} A(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_p z^{-p}, \\ C(z) &= 1 + c_1 z^{-1} + c_2 z^{-2} + \cdots + c_q z^{-q}. \end{aligned}$$

To handle trends in the data the ARMA model can be extended to an autoregressive integrated moving average (ARIMA) of order (p, d, q). The model then looks like

$$A(z)(1 - z^{-1})^d y_t = C(z)e_t,$$

with $A(z)$ and $C(z)$ defined as before.

To find the appropriate model order we filter our signal through $\hat{h}_t^{-1}$ to obtain the residuals $\hat{e}_t$. If our model describes the observations accurately, the residuals should be white. A natural way to test if the residuals are white is to compute the autocorrelation function (ACF) of the residuals. If the estimated residuals are white, then, asymptotically, the ACF of the residuals, $\rho_{\hat{e}}(k)$ will be normally distributed, as $\rho_{\hat{e}}(k) \sim \mathcal{N}(0, \frac{1}{n})$, for $k \neq 0$, where n denotes the sample size and $\rho_{\hat{e}}(k)$ is defined as

$$\begin{aligned} \rho_{\hat{e}}(k) &= \frac{r_{\hat{e}}(k)}{r_{\hat{e}}(0)}, \\ r_{\hat{e}}(k) &= C[\hat{e}_t, \hat{e}_{t-k}]. \end{aligned}$$

3.5.2 The Kalman filter

To handle non-stationary processes we turn to the Kalman filter which expresses the optimal linear reconstruction, interpolation and prediction of the state vector given observations of the input

and output of the system [29]. We continue by defining the state space representation of a linear time-invariant system as

$$\begin{aligned}\mathbf{x}_{t+1} &= \mathbf{A}\mathbf{x}_t + \mathbf{e}_t,\\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t + \mathbf{w}_t,\end{aligned}$$

where $\mathbf{y}_t$ is the measurement vector, being the observed signal at time t, and $\mathbf{x}_t$ is the state vector. The matrices $\mathbf{A}$ and $\mathbf{C}$ are known matrices of appropriate dimension. Here, we assume univariate measurements yielding a scalar y_t. Any external input is also omitted since it will not be included in our case. The noise parameters $\mathbf{e}_t$ and $\mathbf{w}_t$ are called the process noise and measurement noise, respectively. The former captures the model uncertainty and the latter describes the measurement noise (previously denoted e_t). With the state space model defined we omit the derivation of the Kalman filter and continue by establishing the update equations. The optimal linear reconstruction, $\hat{\mathbf{x}}_{t|t}$ and the one-step prediction, $\hat{\mathbf{x}}_{t+1|t}$ can be computed recursively using

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(y_t - \mathbf{C}\hat{\mathbf{x}}_{t|t-1}), \tag{3.2}$$

$$\hat{\mathbf{x}}_{t+1|t} = \mathbf{A}\hat{\mathbf{x}}_{t|t}, \tag{3.3}$$

with $\mathbf{K}_t$ denoting the Kalman gain, formed as

$$\mathbf{K}_t = \mathbf{R}^{x,x}_{t|t-1}\mathbf{C}^\top[\mathbf{R}^{y,y}_{t|t-1}]^{-1},$$

where the variances $\mathbf{R}^{x,x}_{t|t-1}$ and $\mathbf{R}^{y,y}_{t|t-1}$ are defined as

$$\begin{aligned}\mathbf{R}^{x,x}_{t|t-1} &= \mathrm{V}[\mathbf{x}_t - \hat{\mathbf{x}}_{t|t-1}],\\ \mathbf{R}^{y,y}_{t|t-1} &= \mathrm{V}[y_t - \hat{y}_{t|t-1}].\end{aligned}$$

Then, the updated state vector can be used to form predictions of $\hat{y}_t$ as

$$\hat{y}_{t|t} = \mathbf{C}\hat{\mathbf{x}}_{t|t}. \tag{3.4}$$

In the sense of imputation the update equations can be used in the following way. Assume that a missing value occurs at y_t. The best estimate of the state $\hat{\mathbf{x}}_{t|t}$ is simply to keep the former state $\hat{\mathbf{x}}_{t|t-1}$, i.e. neglecting the second term in Equation 3.2. We can then compute the next state $\hat{\mathbf{x}}_{t+1|t}$ using the transition equation, Equation 3.3. The missing value is imputed using Equation 3.4 yielding an estimate, $\hat{y}_t$, of the missing value which in turn is inserted into Equation 3.2 to update the states $\hat{\mathbf{x}}_{t+1|t+1}$ [30].

3.6 Multiple imputation

Multiple imputation (MI) can be applied to a variety of imputation methods. The idea behind the method is to draw multiple possible imputations for each missing value and thereafter impute the

statistically most likely value. This introduces more variation than to only consider one possible imputation value and reflects the uncertainty of the missing data.

Multiple imputation is now accepted as the best general method to deal with incomplete data in many fields [10]. It was developed by Donald B. Rubin in the 1970's as a solution to a practical problem with missing data. Rubin observed that imputing one value (single imputation) for the missing value could not be correct in general. He needed a model to relate the missing data to the observed data and his solution was to create multiple imputations that reflect the uncertainty of the missing data. The method has its roots in Bayesian framework. In Rubin [11] the methodological and statistical groundwork are provided as well as the formulas required to combine the multiple completed data estimates (know as Rubin's rules) and outlines the conditions under which statistical inference under multiple imputation will be valid. Figure 3.2 shows the intended methodology when using multiple imputation. First several imputations are done on the same data set, then the results from each imputed data set is analyzed and ultimately the data is pooled together to give a final imputed data set. As long as an imputation method has some kind of variability, i.e. does not always impute the same values on one specific data set, multiple imputation can (and should) be used.

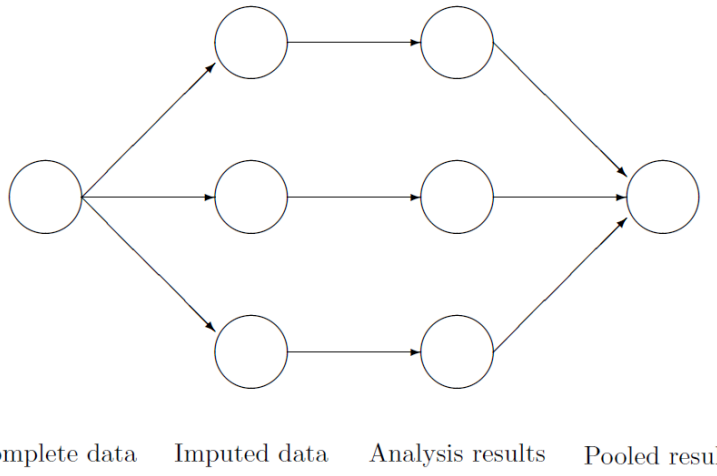


Figure 3.2: Main steps in multiple imputation. Image from *Flexible Imputation of Missing Data* [10].

Rubin's rules

MI refers to the procedure of replacing each missing value with $M \geq 2$ imputed values, forming M complete data sets in total. Ideally, the M imputations of $\mathbf{Y}_{mis}$ should be M independent draws of the parameters and the missing values. Let θ be a scientific estimand, which is a quantity of scientific interest that we can compute if we observe the entire population $\mathbf{Y}$, for example the mean age of a population. Unfortunately, θ is almost never known and we have to estimate θ from the observed data. The goal of MI is to find an estimate $\hat{\theta}$ that is unbiased and confidence valid [10]. Unbiased means that the average of $\hat{\theta}$ is equal to θ as

$$\mathrm{E}[\hat{\theta} \mid Y] = \theta.$$

For the estimand to be confidence valid it is required that the average of the estimated covariance matrix of $\hat{\theta}$ is equal or larger than the variance of $\hat{\theta}$ as

$$\mathrm{E}[W \mid Y] \geq \mathrm{V}[\hat{\theta} \mid Y],$$

with W as the estimated covariance matrix of $\hat{\theta}$. In summary this means that the estimate $\hat{\theta}$ should on average be equal to the true value θ, and the associated covariances should not be smaller than the variance caused by the sampling process.

How certain we are of our estimate $\hat{\theta}$ depends on what we know about $\mathbf{Y}_{mis}$. Therefore we have to summarize the distribution of θ under varying $\mathbf{Y}_{mis}$. The possible values of θ given the observed data $\mathbf{Y}_{obs}$ can be found in the posterior distribution $P(\theta \mid Y_{obs})$, which is often intractable but can be decomposed into two parts which is easier to compute as

$$P(\theta \mid Y_{obs}) = \int P(\theta \mid Y_{obs}, Y_{mis}) P(Y_{mis} \mid Y_{obs}) dY_{mis}. \tag{3.5}$$

The interpretation of Equation 3.5 is best understood from right to left. If we use $P(Y_{mis} \mid Y_{obs})$ to draw imputations for Y_{mis}, denoted $\hat{\mathbf{Y}}_{mis}$, we can then use $P(\theta \mid Y_{obs}, \hat{Y}_{mis})$ to calculate θ from the hypothetically complete data set $(\mathbf{Y}_{obs}, \hat{\mathbf{Y}}_{mis})$. It can be shown that the posterior mean of $P(\theta \mid Y_{obs})$ is

$$\mathrm{E}[\theta \mid Y_{obs}] = \mathrm{E}\big[\mathrm{E}[\theta \mid Y_{obs}, Y_{mis}] \mid Y_{obs}\big].$$

Which implies that that the combined estimate $\bar{\theta}_M$ can be computed as: Let $\hat{\theta}_m$, $m = 1, \ldots, M$ be the M complete-data estimates calculated from M repeated imputations under one model. The combined estimate is then equal to

$$\bar{\theta}_M = \frac{1}{M} \sum_{m=1}^{M} \hat{\theta}_m. \tag{3.6}$$

For simplicity, θ is assumed to be a scalar. The extension to vector θ is straight forward.

The posterior variance of $P(\theta \mid Y_{obs})$ is the sum of two components,

$$\mathrm{V}[\theta \mid Y_{obs}] = \mathrm{E}\big[\mathrm{V}[\theta \mid Y_{obs}, Y_{mis}] \mid Y_{obs}\big] + \mathrm{V}\big[\mathrm{E}[\theta \mid Y_{obs}, Y_{mis}] \mid Y_{obs}\big].$$

This is known as the law of total variance. The first component is known as the average within-imputation variance, denoted $\bar{W}_M$. The second component is known as the between-imputation variance, denoted B_M. If we let the number of imputations $M \to \infty$, then the total posterior variance of θ is $T_M = B_M + \bar{W}_M$. To estimate T_M for finite M, we compute the average within-imputation variance as

$$\bar{W}_M = \frac{1}{M} \sum_{m=1}^{M} \bar{W}_m, \tag{3.7}$$

where $\bar{W}_m$ is the covariance matrix associated with the m:th imputation of $\hat{\theta}_m$; and the between-imputation component as

$$B_M = \frac{1}{M-1} \sum_{m=1}^{M} (\hat{\theta}_m - \bar{\theta}_M)^2. \tag{3.8}$$

Combining the two Equations 3.7 and 3.8 is tempting but incorrect. We have to take into account that a finite number of M imputations have been computed and thus only approximates θ. Rubin suggested that the adjustment for finite M should be B_M/M and we arrive at the total variability for $\bar{\theta}_M$ as

$$T_M = \bar{W}_M + B_M + \frac{B_M}{M} = \bar{W}_M + (1 + \frac{1}{M})B_M. \tag{3.9}$$

To highlight the three different sources of the total variance T_M we summarise them as:

1. $\bar{W}_M$, the variance caused by the fact that we are taking a sample instead of observing the entire population.
2. B_M, the variance introduced by the missing values in the sample.
3. B_M/M, the variance caused by the fact that $\bar{\theta}_M$ is estimated for finite M.

Equations 3.6 and 3.9 are know as Rubin's rules, where the addition of the extra term B_M/M is crucial to make multiple imputation work at low values of M [10]. Excluding it would result in failed standard statistical methods such as too low p-values and too narrow confidence intervals.

Chapter 4

Methodology

In the preceding chapter we discussed the theoretical framework of how to infer proper imputations. The key findings were

- The missingness of the data strongly affects which imputation method to use.
- If the missing data is missing at random (MAR) or missing completely at random (MCAR), we can build a model of the observed data which in turn is used to impute the missing data.
- To account for variability in the missing data one should use multiple imputation.
- To be able to use standard statistical tools as p-values and confidence intervals for multiple imputed values, one should use Rubin's rules.
- Compute outflux, influx, proportion of usable cases, and correlation to investigate the connection between variables.

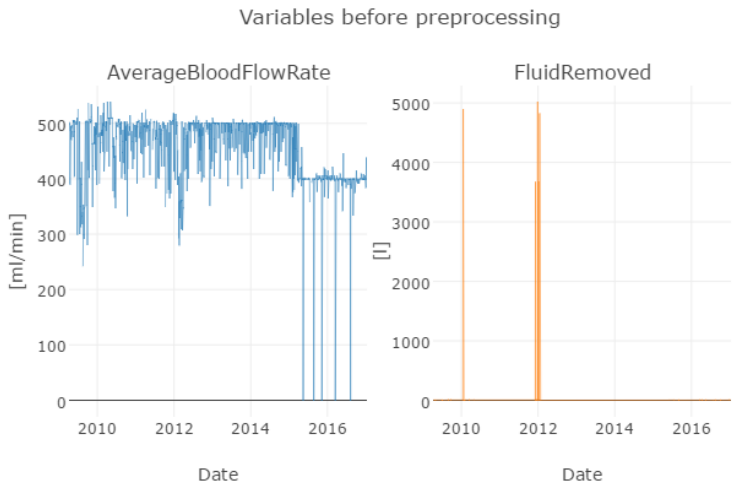
The imputation methods described in Section 3 were all, to some extent, implemented in this study. We continue this chapter by explaining how the methods were implemented, which assumptions were made about the data when implementing each method and, which simplifications or "engineering assumptions" were inflicted on the models.

4.1 Preprocessing of data

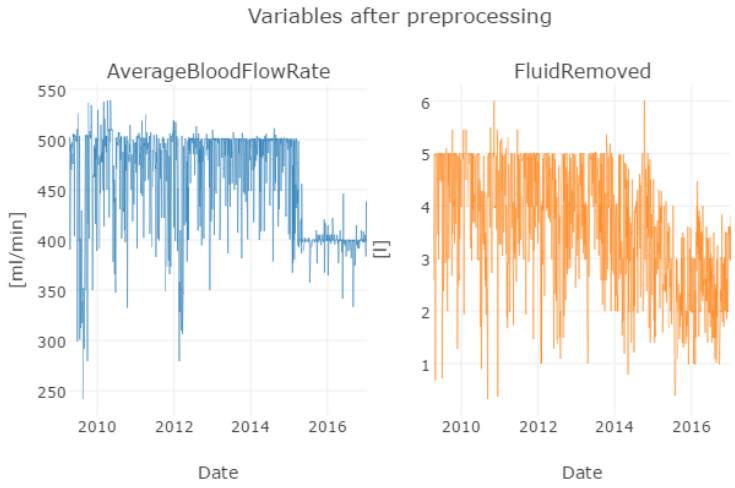
Even data that is not missing can sometimes be in need of some preprocessing to acquire the preferred form. Since there often is no foolproof way to perform and collect the measurements, some outliers are expected due to human and data capturing errors.

To verify if and what kind of preprocessing was needed, first, visual inspection of the data from a few different patients was performed. Realizing that a lot of variables contained outliers (see for example Figure 4.1a) which would significantly alter the outcomes of an imputation model, it was concluded that the data indeed was in need of preprocessing. Since a lot of the data appeared to have been documented in the wrong scale, e.g. milliliters instead of liters, a simple threshold technique was sufficient to take care of most outliers. However, the rows of the variables that clearly was wrong, see Appendix C.1, had to be discarded. Finally the preprocessed data seen in Figure

C.1, in Appendix C.3, could be used for further analysis. After preprocessing, approximately 15% of the raw data was removed. All preprocessing steps and explanations to why a specific threshold was set are given in Appendix C.



(a) Variables *AverageBloodFlowRate* and *FluidRemoved* from one patient before preprocessing.



(b) Variables *AverageBloodFlowRate* and *FluidRemoved* from one patient after preprocessing.

Figure 4.1

4.1.1 Preparing data for analysis

A framework of how to prepare the data for analysis was constructed using the following steps: First the raw data was considered.

1. The raw data was sorted by patient ID and date-time.
2. All elements with a value of zero was replaced with NaNs, since no variable could have a value of zero.
3. All duplicates were removed.
4. The variables containing temperature in Fahrenheit were converted to Celsius.

This gave a crude improvement of the raw data set which variables then could be visually inspected (see Figure 4.1a) to decide further preprocessing steps. When that was done the following steps were made to create the data set referred to as *original data*.

1. Outliers that were recorded in the wrong scale were converted to the right scale, see Appendix C.2.
2. Non-convertible outliers that clearly were unrealistic were set to NaN, see Appendix C.1.
3. All rows where all variables contained NaNs were removed. Those were treatment occasions which the patient had not attended, thus they should not be imputed.

These steps provided a data set with a more realistic missingness pattern which was used as a baseline for all proceeding analysis. The outcomes of the preprocessing steps can be seen in Appendix C.3, an example is given in Figure 4.1.

The next step was to construct validation data which could be used to evaluate the performance of the imputation methods. This was simply done by removing all rows in *original data* which contained any missing values. The final step was to construct the synthetic data set on which the performance of the imputation methods could be tested. This was done by developing an algorithm which extracted the missing data pattern from the original data and then destroyed the validation data in that manner. This algorithm could also be used to destroy the validation data in any wanted amount of missingness, making it possible to evaluate the imputation methods on a greater variety of data sets.

4.1.2 Transformation and normalization of data

When constructing models that analyzes multivariate data it is sometimes important that all variables are scaled to a similar size, so that the relation between the variables become accurate. Many imputation methods also require the data to belong to a normal distribution to work properly. Therefore both normalization/standardization and transformation of the data might be appropriate before analyzing it.

To decide upon appropriate transformations of the variables, a Box-Cox power transformation was used [35]. The Box-Cox power transformation searches for an appropriate exponent (λ) to transform the data to a normal shape. Each λ, except $\lambda = 0$, indicates the power to which the data should be raised, and usually values from $\lambda = -5$ to $\lambda = 5$ are considered. The Box-Cox transformation is given below:

$$\mathbf{Y}_j^{trans} = \begin{cases} \frac{\mathbf{Y}_j^{\lambda}-1}{\lambda}, & \text{if } \lambda \neq 0, \\ \log(\mathbf{Y}_j), & \text{if } \lambda = 0, \end{cases} \tag{4.1}$$

where $\mathbf{Y}_j^{trans}$ is the transformed variable and λ is the power parameter.

Using the Box-Cox transformation on the validation data gave varying outcomes depending on which variable was considered. In Figure 4.2 the recommended power parameter (red line) and a probability plot before and after transformation of the variable *AverageBloodFlowRate* is shown. The rest of the variables are found in Appendix D.1.

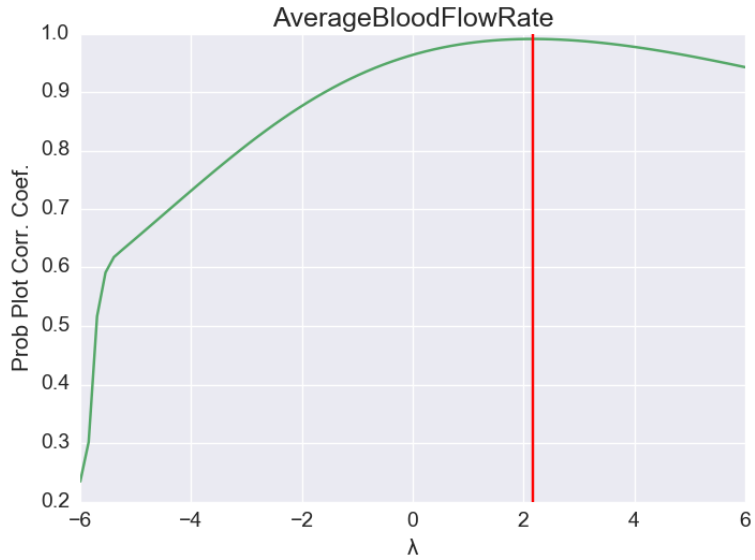
Normalization or standardization of the data was done depending on which imputation method was used. Univariate methods naturally was not in need of any normalization since they only use values from the same variable. Multivariate methods on the other hand uses values from all other variables to impute the missing values, and thus to give similar weight to all variables, it is a good idea to normalize or standardize the data. Standardization transforms the data to have zero mean and unit variance using the following equation

$$\mathbf{Y}_j^{stand} = \frac{\mathbf{Y}_j - \mu_j}{\sigma_j},$$

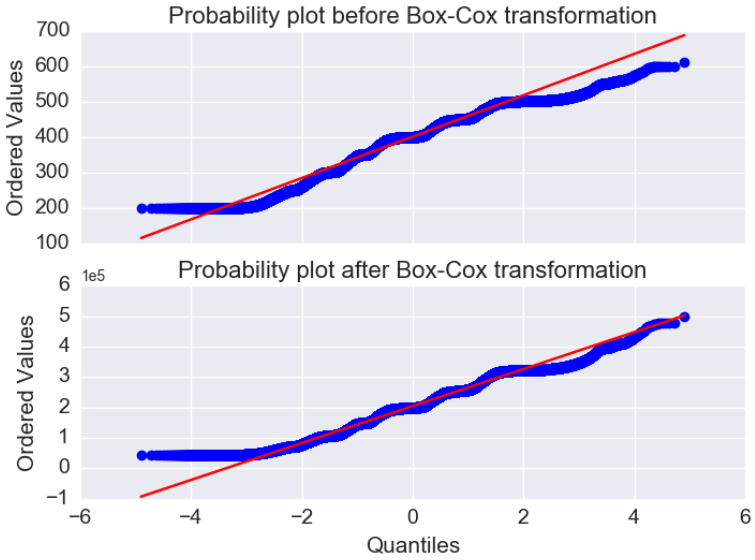
where $\mathbf{Y}_j^{stand}$ is the standardized variable, μ_j is the mean of each variable and σ_j is the standard deviation of each variable. Normalization scales the numeric variables in a range from 0 to 1 and is given by the equation

$$\mathbf{Y}_j^{norm} = \frac{\mathbf{Y}_j - \mu_j}{Y_j^{max} - Y_j^{min}},$$

where $\mathbf{Y}_j^{norm}$ is the normalized variable, μ_j is the mean of each variable, Y_j^{max} is the maximum value of each variable and Y_j^{min} is the minimum value of each variable.



(a) Box-Cox plot with recommended power coefficient highlighted by the vertical red line.



(b) Probability plot before and after transformation using Equation 4.1 with λ set as recommended in Figure 4.2a. The blue line shows the probability plot of the variable *AverageBloodFlowRate* against the quantiles of a normal distribution. The red line shows the normal distribution for reference.

Figure 4.2

As can be seen in Figure 4.2b, the result after transforming the data does not differ that much from the original probability plot, and the distribution of the data still does not resemble that of

a normal distribution. The same was true for all other variables (see Appendix D.1) and since the difference before and after transformation was so small it was concluded that transformation of the data was redundant.

4.1.3 Creating synthetic data

To be able to create data sets with the same characteristics as *original data*, a function which had *original data* and *validation data* as input was created. It extracted the missingness pattern of *original data* and used that pattern to remove elements in *validation data*. The function could also create data sets with a higher ratio of missing data, making it possible to evaluate the imputation methods further. It was important to keep the original missingness pattern of the data even though the amount of missing data increased, thus the function was constructed so that the relation of missingness between variables was kept constant, see Figure 2.6. Using this function, data sets with up to 100% of the rows containing missing values, could be constructed (compared to 15% of the rows for *original data*). An important thing to remember is that when constructing a data set with 100% of the rows containing missing values, only about 17% of the total amount of data points are missing, which makes it possible to still use imputation methods.

4.2 Implementation of algorithms

In this section explanations of how the different imputation methods were implemented will be provided. For the more complex methods there will also be descriptive algorithms showing each step of the process.

The easiest imputation methods, i.e. forward filling and mean substitution, were directly implemented using the `fillna` method in the `pandas` module in Python 3.5, and the `Imputer` class in the `sklearn` module in Python 3.5, respectively. Before running these algorithms however, the data set was grouped patient-by-patient. This was done since it seemed reasonable that the mean of each patient would yield better imputations than just imputing the mean of the whole population.

An alternative mean imputation, which used the mean of k nearest neighbors in one variable as imputation value, was also considered, but since the result was similar to that of the other mean substitution, it was discarded.

4.2.1 k-Nearest Neighbors imputation

The implementation of the kNN imputation algorithm was inspired by that used by Troyanskaya et al. in the article *Missing value estimation methods for DNA microarrays* [20]. First, all variables were grouped patient-by-patient and standardized to get zero mean and a variance of one. Then for each missing value in each variable the algorithm tried to find $k = 10$ other variables which had observed values present at that time. It then choose the k variables which contained values with the smallest (Euclidian) distance to the observed values in the variable to be imputed. Then a weighted average of the values from the k closest variables was used as an imputation. The weights of the variables were decided by examining the similarity of each variable to the variable to be imputed. The kNN algorithm was implemented using the `fancyimpute` package in Python version 3.5.

4.2.2 Joint modelling imputation

As discussed in Section 3.3.2 the assumption of the joint model for multivariate data is that the data comes from a multivariate normal distribution. This assumption is not entirely correct for the data at hand but by normalizing the data we strive to make it follow a multivariate normal distribution.

The implementation is as described in Algorithm 1, with three modifications. First and foremost we add two additional variables to the original eleven. The additional variables are the variable to be imputed shifted one step forward in time and one step backward in time, respectively. The reason for doing so is to capture the time dependency in the data. As one can see in the description of the algorithm, one missing value is imputed at a time using linear regression with noise; hence, it is easy to add extra variables. The second modification is at step 1 and rises as an effect of the shifted variables. We want to keep the order of the measurements to make use of the time dependencies in the data. Therefore we choose not to sort the data according to the missing patterns, ending up with a slower execution time of the algorithm. In addition, the number of missing patterns, S, is large (see Figure 2.6), making it impossible to keep the order of time in the data. The third modification is at step 11. Instead of drawing the mean vector and covariance matrix we choose to estimate the mean vector and covariance matrix from the imputed data set.

The algorithm is updated in an iteratively fashion, which means that we have to make sure that the algorithm has converged. To evaluate when the algorithm has converged we plot the mean of each variable and consider when the difference in the mean between two iterations is smaller than some stopping criterion ϵ.

The starting value for the mean vector is initialized as the estimated mean from the observed data. As for the covariance matrix, the user has two alternatives: (1) initializing with ones on the diagonal and zero elsewhere or (2) initialize with the estimated covariance of the observed data. This algorithm was implemented in Python 3.5 from scratch.

Algorithm 1: Imputation of missing data by a joint model for multivariate data according to Van Buuren [10].

1. Sort rows of $\mathbf{Y}$ into S missing data patterns $\mathbf{Y}_s$, $s = 1, \ldots, S$.
2. Initialize $\theta^{(0)} = (\boldsymbol{\mu}^{(0)}, \boldsymbol{\Sigma}^{(0)})$ by a reasonable starting value.
3. **for** t *in* T **do**
    4. **for** s *in* S **do**
        5. Calculate parameters $\boldsymbol{\phi}_s = \text{swp}(\theta^{(t-1)}, s)$ by sweeping the predictors of pattern s out of $\theta^{(t-1)}$.
        6. Calculate p_s as the number of missing data points in pattern s. Calculate $o_s = p - p_s$ as the number of observed data points in pattern s.
        7. Calculate the Choleski decomposition $\mathbf{C}_s$ of the $p_s \times p_s$ submatrix of $\boldsymbol{\phi}_s$ corresponding to the missing data in pattern s.
        8. Draw a random vector $\boldsymbol{z} \sim \mathcal{N}(0, 1)$ of length p_s.
        9. Take $\boldsymbol{\beta}_s$ as the $o_s \times p_s$ submatrix of $\boldsymbol{\phi}_s$ of regression weights.
        10. Calculate imputations $\overset{(t)}{s} = \mathbf{Y}_s^{obs}\boldsymbol{\beta}_s + \mathbf{C}_s^{\top}\boldsymbol{z}$, where $\mathbf{Y}_s^{obs}$ is the observed data in pattern s.
    end
    11. Draw $\theta^{(t)} = (\boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})$ from the normal inverted-Wishart distribution according to Schafer [9].
end

4.2.3 Imputation using time series models and expectation maximization

A similar methodology as that of the joint model imputation method, was used with the distinct difference that instead of a linear regression model at the imputation step an ARIMA(p,d,q) model was used. Again, the data was assumed to belong to a multivariate normal distribution and appropriate transformations were applied to the data. An ARIMA(2,1,1) model was selected as the most appropriate model order to use for each variable. The model order was decided by using various time series techniques for model evaluation, e.g. the autocorrelation function of the residuals, which is presented in Appendix B.

The algorithm comprises of the following steps: (1) replace the missing values by initial guesses, e.g. forward fill; (2) estimate the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$; (3) estimate the coefficients for the ARIMA model, for each of the univariate time series; (4) re-estimate the missing values using updated estimates of the parameters and the coefficients of the time series; (5) iterate until the convergence criterion is reached, set to 10^{-6} [36]. The method was implemented using the `mtsdi` package version 0.3.3 in R version 3.2.4.

4.2.4 Imputation via the Kalman filter

Each variable was imputed as a univariate time series using a Kalman filter on each variable. In other words, correlation between the variables was not included. The model order was selected as an ARIMA(2,1,1) model which was obtained in the same way as for the time series method. The selected model was in turn fed to the Kalman filter where the imputations were made. The method was implemented using the `imputeTS` package version 2.1 in R version 3.2.4. The package is built on the theory given by Durbin and Koopman [30] and briefly described in Section 3.5.2.

4.2.5 Multivariate imputation by chained equations

When researching imputation methods one is bound to come across the multivariate imputation by chained equations (MICE) algorithm. It seems to be ubiquitous in the world of missing data and builds on fully conditional specification (FCS) (see Section 3.3.3). The MICE algorithm models each variable conditional upon the other variables in the data, using a series of regression models to impute the missing values [37]. This makes it possible to model each variable according to its distribution instead of trying to fit a general model for all of the variables. One of the biggest perks with the chained equations approach is that it can handle both continuous and categorical values (though in this report only continuous values are considered). The use of multiple imputations, which was described in Section 3.6, makes the algorithm take into account the uncertainty in the imputations and creates statistically accurate standard errors.

The regression model defined for each variable was a Bayesian ridge regression model, where the output is assumed to be Gaussian distributed around $\mathbf{Y}_{-j}\boldsymbol{\beta}$ as

$$P(Y_j \mid \boldsymbol{Y}_{-j}, \boldsymbol{\beta}, \sigma) \sim \mathcal{N}(Y_j \mid \boldsymbol{Y}_{-j}\boldsymbol{\beta}, \sigma).$$

The prior for the the parameter $\boldsymbol{\beta}$ is given by a spherical Gaussian

$$P(\boldsymbol{\beta} \mid \lambda) \sim \mathcal{N}(\beta \mid 0, \lambda^{-1}\boldsymbol{I}_p),$$

and the priors over σ and λ are chosen to be gamma distributions, the conjugate prior for the precision of the Gaussian. In our case, $\lambda = 0.001$ and σ is estimated from the residual sum of squares. The MICE algorithm was implemnted using the `fancyimpute` package in Python version 3.5. The different steps of the MICE algorithm can be found in Algorithm 2. To account for variability in the algorithm the average of ten runs was used as final imputation.

Algorithm 2: Imputation of missing values using MICE as proposed by Van Buuren [10].

Data: $\mathbf{Y}_j$ is the variable to be imputed, $\mathbf{Y}_{-j}$ are the other variables, $\mathbf{R}$ is the missingness matrix, and T is the number of iterations.

1. Specify an imputation model $P(Y_j^{mis}|Y_j^{obs}, Y_{-j}, \mathbf{R})$ for variable $\mathbf{Y}_j$ with $j = 1, \ldots, p$.
2. **for** j *in* $\boldsymbol{Y}$ **do**
    3. Fill in starting imputations $\hat{\mathbf{Y}}_j^{(0)}$ by random draws from $\mathbf{Y}_j^{obs}$.
end
4. **for** t *in* T **do**
    5. **for** j *in* $\boldsymbol{Y}$ **do**
        6. Define $\hat{\mathbf{Y}}_{-j}^{(t)} = (\hat{\mathbf{Y}}_1^{(t)}, \ldots, \hat{\mathbf{Y}}_{j-1}^{(t)}, \hat{\mathbf{Y}}_{j+1}^{(t-1)}, \ldots, \hat{\mathbf{Y}}_p^{(t-1)})$ as the currently complete data except Y_j.
        7. Draw $\hat{\phi}_j^{(t)} \sim P(\phi_j^{(t)}|Y_j^{obs}, \hat{Y}_{-j}^{(t)}, \mathbf{R})$.
        8. Draw imputations $\hat{\mathbf{Y}}_j^{(t)} \sim P(Y_j^{mis}|Y_j^{obs}, \hat{Y}_{-j}^{(t)}, \mathbf{R}, \hat{\phi}_j^{(t)})$.
    end
end

4.2.6 Random forest imputation

When implementing the random forest imputation algorithm the steps of Algorithm 3 were followed. For each variable a model was trained and fitted using all other variables, plus shifted versions (lag ± 1) of the variable to be imputed. The `sklearn` package `RandomForestRegressor` version 0.18 in Python version 3.5 was used as random forest regressor, and the number of trees was set to 10. Since the regressor only works with real values, initial guesses for the missing values had to be estimated during training. To keep things simple forward filled values were used as initial guesses. When all of the missing values for each variable had been imputed, that imputed data set was used as an initial guess in the next iteration. The algorithm was then iterated until the stopping criterion was met, which was as soon as the difference Δ, defined as

$$\Delta = \sum_{\forall j} \left(\frac{\sum_{j \in p} (\mathbf{Y}_{new}^{imp} - \mathbf{Y}_{old}^{imp})^2}{\sum_{j \in p} (\mathbf{Y}_{new}^{imp})^2} \right), \tag{4.2}$$

between the newly imputed data set and the previous one increased. In other words until the algorithm had converged and the results did not improve any longer. The average of ten regressors, summing up to 100 trees, was then used for the final imputations.

Algorithm 3: Imputation of missing values using Random forest as proposed by Stekhoven and Bühlmann [28].

Data: $\mathbf{Y}$ is an $n \times p$ matrix, Δ as defined in Equation 4.2.
1. Make initial guess for missing values (e.g. forward fill).
2. Sort indices of columns in $\mathbf{Y}$ w.r.t. increasing amount of missing values, store in vector $\mathbf{k}$.
3. **while** $\Delta_{new} \leq \Delta_{old}$ **do**
   4. Store previously imputed matrix, $\mathbf{Y}_{old}^{imp}$.
   5. **for** *s in* $\boldsymbol{k}$ **do**
      6. Fit a random forest: $\mathbf{Y}_{s}^{obs} \mid \mathbf{Y}_{-s}^{obs}$.
      7. Predict $\mathbf{Y}_{s}^{mis}$ using $\mathbf{Y}_{-s}^{mis}$.
      8. Update imputed matrix, using predicted $\mathbf{Y}_{s}^{mis}$.
   end
   9. Update Δ.
end
10. Return imputed matrix $\mathbf{Y}^{imp}$.

4.3 Evaluation of imputation methods

In the end we want to be able to say whether our imputation method has improved the complete data set or not. Especially, we want to compare if any improvement has occurred compared to the Forward fill method used in the present situation.

To be able to analyze the implemented imputation methods, the validation data (called Y) was artificially destroyed to create a new data set (Y^*), see Section 2.3.5. The different imputation methods could then be used to reconstruct the destroyed data set ($\hat{Y}$) and then the result could be compared to the original data set, see Figure 4.3. The error (e) between each imputed point was then calculated (using normalized root mean square error, see Section 4.3.1) and the results for the different methods were compared, see Table 5.1 in Section 5.

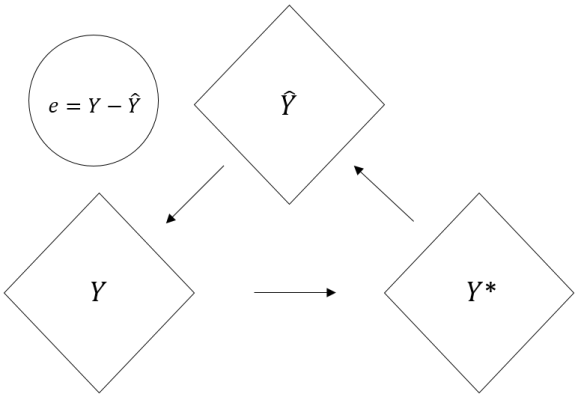


Figure 4.3: The imputation process; the original dataset Y is partially destroyed giving a dataset with missing values, Y^*. An imputation method is then used to reconstruct Y from Y^*. The reconstructed dataset $\hat{Y}$ is then compared to the original dataset Y giving the imputation error e.

4.3.1 Minimizing the error

The obvious way to compare two methods, when imputed data and validation data is available, is to measure how well the two methods are at recreating the lost data. A standard criterion for accuracy is to measure how far the imputed value is from the true value. Here, we use the normalized root mean squared error (NRMSE) as a distance measure [31],

$$NRMSE = \sqrt{\frac{\mathrm{E}[(\boldsymbol{Y}_j^{mis} - \hat{\boldsymbol{Y}}_j^{mis})^2]}{\mathrm{V}[\boldsymbol{Y}_j^{mis}]}}, \tag{4.3}$$

where the mean and variance are computed over the missing entries in the matrix. We know the true value of $Y_{i,j}^{mis}$ since the data is artificially destroyed. When the imputations are accurate we have a NRMSE close to zero and when they are bad we will have a value around one. Note that, if the mean of the observed data is used to impute the missing values, assuming that ignorability holds, then, the NRMSE will be 1, since the nominator then is the definition of the variance.

Comparison to validation data

As a baseline for comparison, forward fill was used. Since Lytics use forward fill as their current imputation method it serves as a good baseline method to compare with other methods. The normalized root mean squared error (Equation 4.3) was then used to evaluate the performance of each imputation method.

Bias and similarity measures

In addition to the NRMSE we were interested in evaluating the accuracy and agreement of the imputed values. The accuracy was computed as the mean difference between the observed and imputed values, denoted

$$bias_j = \frac{1}{n_j^{mis}} \sum_{i=1}^{n_j^{mis}} (Y_{i,j}^{mis} - \hat{Y}_{i,j}^{mis}).$$

The most common indicator of agreement when evaluating imputation methods is Pearson's correlation coefficient, ρ, defined in Equation 2.7. The correlation is computed between the true values and the imputed values. However, the values of ρ might not be related to the magnitude of the errors. To avoid the issue, we use a different measure of agreement proposed by Willmott [32]; known as "index of agreement", and defined as

$$d_j = 1 - \frac{\sum_{i=1}^{n_j^{mis}} (Y_{i,j}^{mis} - \hat{Y}_{i,j}^{mis})^2}{\sum_{i=1}^{n_j^{mis}} (|Y_{i,j}^{mis} - \bar{\mathbf{Y}}_j^{mis}| + |\hat{Y}_{i,j}^{mis} - \bar{\mathbf{Y}}_j^{mis}|)^2},$$

where $\bar{\mathbf{Y}}_j^{mis}$ denotes the mean of variable j. The value of d ranges between 0 and 1 with 0 being lack of agreement and 1 being perfect agreement.

4.3.2 Statistical confirmation

It is also of interest to investigate if the imputed values are not statistically different to the validation data. To do this the Welch's t-test [33] and Kullback-Leibler divergence [34] were used as metrics. The imputed values were compared to the validation values using these two different approaches. For the methods containing stochasticity (`RF` and `MICE`) the mean of ten imputed data sets were used as final values.

Welch's t-test

Welch's t-test is used to reject that two populations have the same mean and is defined as

$$t_j = \frac{\bar{\mathbf{Y}}_j^{imp} - \bar{\mathbf{Y}}_j^{val}}{\sqrt{\frac{(s_j^{imp})^2}{n_j^{imp}} + \frac{(s_j^{val})^2}{n_j^{val}}}}, \tag{4.4}$$

where $\bar{\mathbf{Y}}_j^{imp}$ is the mean of imputed variable j, $\bar{\mathbf{Y}}_j^{val}$ is the mean of validation variable j; s_j^2 is the sample variance and n_j is the sample size (equal in this case). The p-values from the t-test can then be used to reject the null hypothesis of equal averages.

Welch's t-test was calculated for each method and variable. In our case the null hypothesis was that the imputed population had the same average as the validation population and thus we did not want to reject it. The Welch's t-test was calculated using the function `ttest_ind` in the `scipy.stats` module in Python 3.5.

Kullback-Leibler divergence

The Kullback-Leibler (KL) divergence can be used as a measure of how much one probability distribution diverges from another. We use it to see how similar behavior we can expect from the imputed variables compared to the validation variables. It is defined as

$$S_j = \sum_{i=1}^{n_j^{mis}} Y_{i,j}^{imp} \log(\frac{Y_{i,j}^{imp}}{Y_{i,j}^{val}}), \tag{4.5}$$

where S_j is the KL divergence and $Y_{i,j}$ is a normalized variable. A small value of S (approaching zero) indicates that a similar behaviour can be expected of the two distributions, whilst a large number (approaching one) indicates the opposite.

The KL divergence was also calculated for each method and variable. In our case a small number is preferred since the imputed variables should behave similar as the validation variables. The KL divergence was calculated using the `entropy` function in the `scipy.stats` module in Python 3.5.

4.3.3 Prediction improvement

In the end, it is of interest to see if the prediction algorithm for hospitalizations of patients can be improved by using a more advanced imputation method. A logical, second evaluation criterion would then be to compare how much better the predictions become when using the imputed data set compared to when the forward filled data set is used. To do this, we used four data sets with different amount of missing values, but with the same missingness pattern as *original data*. As a metric we used the area under the receiver operating curve (AUROC) or in short, the area under curve (AUC). An AUC score of 1 means that we have a perfect classifier and a score of 0.5 is equivalent to guessing. Information about the receiver operating curve can be found in e.g. *The Elements of Statistical Learning* [26].

4.3.4 Sequence length

The more sophisticated methods require some data in order to perform well. Therefore, in addition to the three evaluation methods, the number of measurements for each patient were also investigated. Some of the imputation methods, e.g. the Kalman filter and the expectation maximization built on ARIMA(p,d,q) models, require a number of measurements to compute good estimates of the a and c coefficients. Our approach was to split the data set according to patients and then pick patients with more than L measurements, $L = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400, 500, 1000)$. If a patient had more than L measurements the number of measurements were reduced to only include the first L values. The reduced data set was imputed patient-by-patient using all methods and the NRMSE was computed for each value of L. The results are shown in Section 5.4.

Chapter 5

Results

Three different criteria were used to evaluate the performance of the imputation algorithms. Those were

1. Comparison to validation data.
2. Statistical performance.
3. Result from prediciton algorithm.

In the sections below, the results from the three criteria can be found.

5.1 Comparison to validation data

Below are the errors between the imputed data and validation data, computed for each variable and imputation method. The metric used for comparison is the normalized root mean squared error, see Equation 4.3, where a lower number means an imputation closer to the validation data. The Kalman filter method outperforms the other methods in all variables except for one, *LitersProcessed*, and reduces the average NRMSE from 0.90 to 0.67, a reduction of $\sim 25\%$.

Variable	FFill	RF	Kalman	TS-EM	MS	MICE	kNN	JM
ABFR	0.70	0.59	**0.53**	0.58	0.73	0.89	0.65	0.65
SSP	0.88	0.72	**0.66**	0.70	0.76	0.96	0.78	0.77
LP	0.77	**0.46**	0.58	0.50	0.69	0.65	0.50	0.58
TD	0.95	0.76	**0.69**	0.72	0.74	0.97	0.77	0.86
FR	0.80	0.66	**0.60**	0.66	0.72	0.92	0.69	0.77
PTS	1.07	0.83	**0.78**	0.79	0.84	0.91	0.85	0.88
PTE	1.10	0.85	**0.80**	0.81	0.85	0.93	0.87	0.91
SSSBP	0.83	0.75	**0.65**	0.69	0.78	0.83	0.78	0.77
SSDBP	0.89	0.76	**0.69**	0.74	0.79	0.84	0.81	0.80
ESSBP	0.89	0.79	**0.68**	0.71	0.81	0.84	0.78	0.82
ESDBP	0.94	0.80	**0.70**	0.73	0.79	0.84	0.78	0.84
Mean error	0.89	0.72	**0.67**	0.69	0.77	0.87	0.75	0.79

Table 5.1: NRMSE for each imputation method and variable. The lowest errors are in bold font. The mean of 10 imputed data sets for algorithms `RF` and `MICE` (containing stochasticity) was used when calculating the error.

5.1.1 Bias and similarity measures

The bias and similarity measures (Pearson's correlation (ρ) and agreement index (d)) are summarized in table 5.2. The values presented are the mean value over all variables for each method. For more detailed information see Appendix E.

	FFill	RF	Kalman	TS-EM	MS	MICE	kNN	JM
bias	-0.152	-0.180	-0.061	0.410	**−0.055**	-0.132	-0.211	-0.263
ρ	0.593	0.684	**0.735**	0.722	0.630	0.466	0.590	0.632
d	0.767	0.807	0.831	**0.836**	0.747	0.557	0.741	0.780

Table 5.2: Aggregated values of the bias, correlation, and agreement index for different methods. The best value for each metric is in bold font.

5.2 Statistical measures of imputations

The p-values from Welch's t-test for each variable and imputation method are presented below. For imputed variables with a large p-value ($p > 0.05$) the null hypothesis of equal means cannot be rejected.

Variable	FFill	RF	Kalman	TS-EM	MS	MICE	kNN	JM
ABFR	0.677	**0.033**	0.781	0.077	0.941	0.922	**0.000**	0.124
SSP	0.863	0.500	0.904	0.312	0.597	0.670	0.374	0.935
LP	0.872	0.151	0.866	**0.000**	0.694	0.793	**0.000**	**0.000**
TD	0.677	**0.028**	0.871	**0.000**	0.971	0.447	0.775	0.120
FR	0.617	0.275	0.656	**0.000**	0.543	0.165	**0.001**	**0.000**
PTS	0.943	0.423	0.592	0.801	0.387	0.976	0.302	0.873
PTE	0.451	0.901	0.884	0.928	0.907	0.498	0.668	0.303
SSSBP	0.605	0.488	0.773	0.769	0.815	0.658	0.932	0.582
SSDBP	0.845	0.563	0.798	0.270	0.945	0.753	0.774	0.568
ESSBP	0.381	0.342	0.429	0.724	0.328	0.220	0.556	0.672
ESDBP	0.658	0.185	0.493	0.445	0.469	0.093	0.674	0.102
Mean	0.690	0.354	0.732	0.393	0.691	0.563	0.460	0.389

Table 5.3: p-values from Welch's t-test for each method and imputed variable compared to the validation data. All values with a p-value below the significance level of 0.05 are highlighted with bold font. Those imputed variables do not have the same average as the validation data with statistical significance.

Below the Kullback-Leibler divergences between the imputed data and validation data can be seen. All imputed variables show more similarity than dissimilarity compared to the validation variables.

Variable	FFill	RF	Kalman	TS-EM	MS	MICE	kNN	JM
ABFR	0.006	0.004	**0.003**	0.004	0.007	0.009	0.005	0.005
SSP	0.010	0.007	**0.006**	0.007	0.008	0.013	0.008	0.008
LP	0.020	**0.008**	0.012	**0.008**	0.017	0.015	0.010	0.012
TD	0.017	0.011	**0.009**	0.010	0.011	0.018	0.011	0.14
FR	0.080	0.057	**0.048**	0.053	0.069	0.106	0.062	0.069
PTS	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
PTE	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
SSSBP	0.012	0.010	**0.007**	0.008	0.011	0.012	0.010	0.010
SSDBP	0.018	0.013	**0.011**	0.012	0.014	0.016	0.015	0.015
ESSBP	0.014	0.011	**0.008**	0.009	0.012	0.013	0.011	0.012
ESDBP	0.021	0.015	**0.012**	0.013	0.015	0.017	0.015	0.016
Mean	0.018	0.013	**0.011**	**0.011**	0.015	0.020	0.013	0.015

Table 5.4: KL-divergence for each method and imputed variable compared to the validation data. The values closest to 0 for each variable are highlighted with bold font. Those variables have a behaviour most similar to that of the validation data.

5.3 Prediction results

In Table 5.5 the results from the predictions are presented. A random forest classifier was used to classify the patients on (1) no imputed data, (2) imputed data using `FFill` and (3) imputed data using `Kalman`. A larger AUC-score indicates better performance. A thorough explanation of the

classifier is given in the Master's Thesis by our colleague Niklas Hansson [38]. The percentages refers to the number of rows in the data set with at least one missing value, and the percentages within parenthesis refers to the total amount of missing data. The first column containing 15% of the rows with at least one missing value shows the results from the *original data*. Due to lack of time we choose to only evaluate `Kalman`. It was selected because it performed best on the previous evaluation methods.

Missing	15% (2.5%)	25% (4.3%)	50% (8.6%)	75% (12.8%)	100% (17.1%)
w/o imputation	0.615	0.613	0.608	0.604	0.604
FFill	0.632	0.631	0.629	0.628	0.627
Kalman	**0.635**	**0.635**	**0.633**	**0.635**	**0.636**

Table 5.5: Average AUC score on 10 runs with a random forest classifier for different amount of missing data. The best score for each rate of missingness is highlighted with bold font.

We note that, regardless of the amount of missing data, the `Kalman` imputed data sets produces similar results whereas for `FFill` and the not imputed data sets the AUC score decreases with increasing missing data.

5.4 Sequence length

In this section the errors for different methods and sequence lengths are presented. The number of patients within each group is shown in Figure 5.1. In Figure 5.2 the NRMSE of all methods are presented.

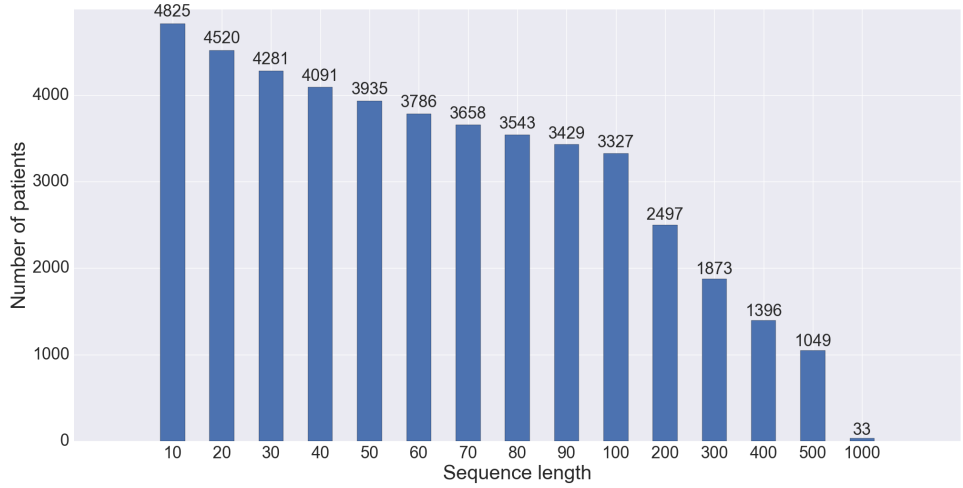


Figure 5.1: The number of patients with sequences at least L long. Note the change of scale on the x-axis, both after $L = 100$ and $L = 500$.
$L = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400, 500, 1000)$

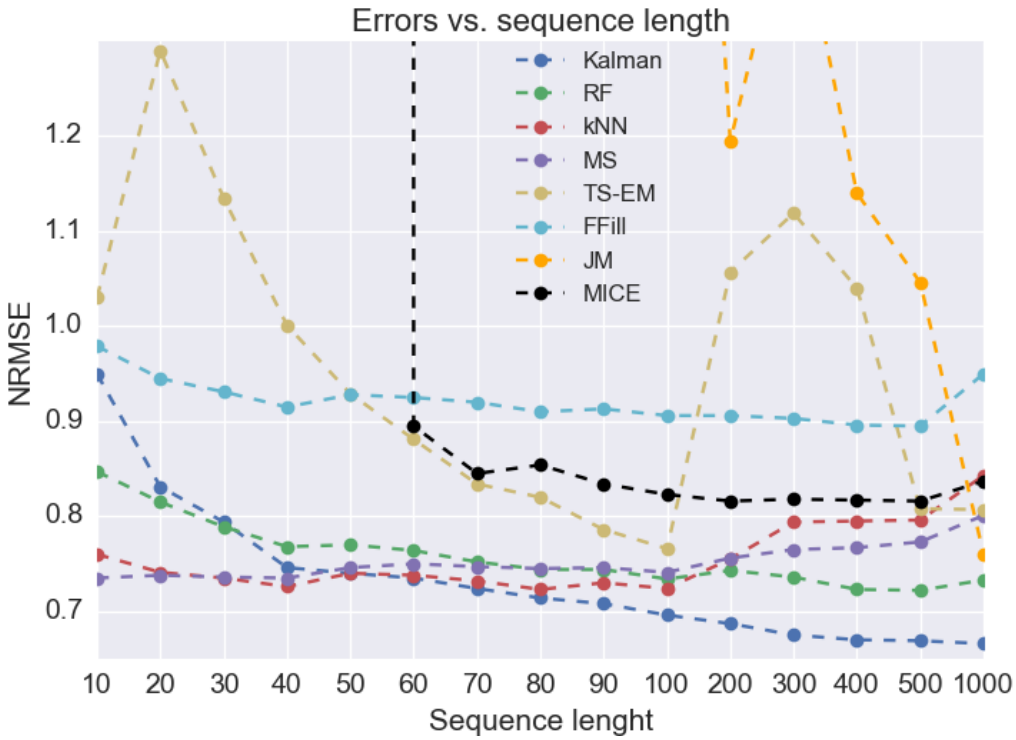


Figure 5.2: Average NRMSE of all variables, computed for all imputation methods using different sequence lengths L. Imputations are done in a patient-by-patient manner. Note the change of scale on the x-axis, both after $L = 100$ and $L = 500$.

The NRMSE is decreasing with increasing L, at least for the more sophisticated methods, except for `TS-EM` where we see an increase in NRMSE after $L = 100$ which we have no explanation for. However, it appears to be a peak at $L = 300$ for both `TS-EM` and `JM` that could be explained by some instability in the estimation of the mean vector and covariance matrix of the multivariate Gaussian distribution. For the more naïve approaches, the error is stable regardless of sequence length.

The `JM` algorithm performs bad in the beginning but have a rapid decrease in error when the sequence length is increased. The `MICE` algorithm also performs bad at the beginning and we see no large reduction in error after $L = 60$. Both `Kalman` and `RF` are well-behaved even for small values of L and continue to decrease with increasing L.

5.5 Merging of methods

To impute the best possible data set, considering NRMSE, we decided to use the imputed values of `RF` for variable *LitersProcessed* and use `Kalman` for all other variables. The resulting average NRMSE was 0.66, an improvement of 0.01. When predictions were computed on this data set we

achieved an AUC score of 0.634, which is worse than the AUC score of `Kalman`. The reduction of the AUC score was 0.001 compared to `Kalman`.

Furthermore, to reduce the NRMSE and have a generic solution for each patient, we decided to merge the methods `MS`, `RF`, and `Kalman` in the following way. For patients with less than 40 rows we used `MS`; for all other patients we used `Kalman`, except for variable *LitersProcessed* where we imputed using `RF`. This resulted in an average NRMSE of 0.66 and an AUC score of 0.631. The NRMSE was reduced by 0.01 compared to `Kalman`, but the prediction algorithm performed worse and the AUC score was reduced by 0.004 compared to `Kalman` and 0.001 compared to `FFill`.

Chapter 6

Discussion

With an increasingly older population, where many will suffer from chronic diseases, we need to redefine how to treat patients. Arguably, every country in the western world is battling increasing healthcare costs and by putting more effort into preventive measures there could be much to gain. As an example, Centers for Dialysis Care (CDC), the collaborating partner of Lytics Health AB in this project, says that a hospitalized dialysis patient costs about $2,700 per night. If an improved imputation method can increase the performance of the prediction algorithm and help to reduce the number of hospitalized patients that would benefit the hospitals, patients, and the society.

6.1 In general

There is a lot of work to be done in the field of missing data. Just to make it knowledgeable to people that efficient and relatively precise imputation algorithms exists, and that there is much to gain by using sophisticated imputation methods, would increase the usage of them. As the use of big data is steadily increasing, so should the value and interest in imputations, which would result in more imputation methods emerging.

A problem with imputing values is how to handle the data set after it has been imputed. Should imputed values be regarded as actual recordings or should they be used only with great care? In most research fields, and especially in medical research, the latter choice is the way to go since it would be misguiding and potentially fatal to treat imputed values as actual measurements. It could give a doctor the wrong impression about a patient or make an algorithm give false results. Instead one could mark all imputed values in some way which gives them lesser weight when used in e.g. a prediction algorithm.

6.2 Assumptions of the data

Throughout the report we assume the missing data mechanism to be MAR. The assumption might not be entirely valid but is essential for building models of the observed data to be used when imputing missing data. We have reason to believe that the missing data mechanism might be NMAR, for example, some measurements of the blood pressure may not be taken when a patient has low blood pressure, meaning that the value of the variable that is missing is related to the

reason it is missing. However, we have no way of finding out the true missing data mechanism since that would require a thorough investigation of how the data capturing procedure is carried out. The only true way to distinguish between NMAR and MAR is to measure the missing data in some way, e.g. interviewing the healthcare personnel and find out when and why measurements are not taken.

Many of the implemented methods assume Gaussian distributed data. This assumption is clearly violated for some of the variables, which can be seen in Appendix A. Nevertheless, the NRMSE is reduced compared to `FFill`, but we might be able to further reduce the error by putting more effort into finding more suitable distributions for each variable. This would probably improve the result of the `MICE` imputation where we can define a model for each variable. On the other hand `RF`, the only non-parametric method, does not perform any better than the other methods indicating that it is not crucial that the assumptions of the distributions are correct.

For the methods `RF` and `JM` we introduce shifted variables where we shift the variable to be imputed ± 1 step in time and add as additional variables. To allow for this shift we assume that the imputations are done offline, meaning that all data is available. However, if online imputation is required we have a problem. Assume that we want to compute predictions when new values are measured and added to the data, meaning that a daily update of imputations and predictions of the data is required. This, in turn will introduce a problem of causality, namely that we have no access to the next value making the forward shift impossible. Depending on situation one has to decide if it is more important to have an online procedure or to be able to use future values.

6.3 Sequence lengths

We had reason to believe that imputations in a patient-by-patient manner would be preferable compared to imputations of the entire data set with one model. This was mainly due to the fact that no human is identical to another. In addition, the prediction method developed at Lytics is generic and computes predictions from the first day a patient is entering the system. Hence, missing values can occur as early as the first day, which requires a generic imputation method. Therefore we had to investigate how well the methods perform on patients with few treatments. As can be seen in Table 2.2, half of the patients have had 186 treatments or less which implies that it is important that the methods are well-behaved with few data points. In Figure 5.2 we see decreasing error for the more advanced methods. For `Kalman`, `RF`, `kNN`, `TS-EM`, `MICE`, and `JM` the error is reducing but they do not become better than the average error for the full data set. We believe that these results are strong indications that an ensemble method, combining the strengths of a sophisticated method when imputing patients with long sequences and a more naïve method when imputing patients with short sequences, where there is not enough data for the sophisticated method to function properly, should be used to further reduce the NRMSE and build a more generic method.

Moreover, when looking at the sequence length plot (see Figure 5.2) it is evident that some methods perform very poorly when introduced to short sequences of data. One has to decide whether it is more important to have a robust method for all sequence lengths, or to have a mixture of models that might outperform the more robust method. In medical applications the more robust methods is preferable since there is little to no room for error.

6.4 Error and prediction

Turning to Table 5.1, containing the NRMSE for all variables, it is evident that `Kalman` outperforms the other methods in all variables except for one. The second best method, considering the average NRMSE, is `TS-EM`. `Kalman` is a univariate method, considering only one variable at a time, whereas `TS-EM` is a multivariate method using information both from the other variables and from previous values within the variable to be imputed. These two methods rely heavily on previous values of the time series in contrast to `RF`, `MICE`, and `JM` which impute using regression over the other variables. Two interesting issues arise from these results: (1) there is much to gain by considering time series approaches when imputing this type of health care data and (2) it appears to be of less importance to consider dependencies between variables when imputing this particular data set.

The connection between the variables is of great importance. As one can see in Figures 2.6 and 2.7 the variables *StartSittingSystolicBloodPressure* and *StartSittingDiastolicBloodPressure* are always missing at the same time as well as *EndSittingSystolicBloodPressure* and *EndSittingDiastolicBloodPressure*. The two pairs are the variables with highest correlation meaning that they should be of use when imputing, but with the variables missing at the same time they are useless for imputing one another. Furthermore, the correlation between variables are low or almost zero for many of the combinations. However, this is not the case for the autocorrelation of a variable; the strong dependency of previous values can be seen in the autocorrelation plots in Appendix B. This is the main reason for adding the additional features, where the variable to be imputed is shifted in time.

When investigating the bias it is clear that `Kalman` is unbiased for all variables whereas `TS-EM` is unbiased for half of the variables. Nevertheless, `TS-EM` is outperforming all other methods at agreement index (d) and has the second highest value of correlation (ρ); where `Kalman` has the highest value. Again, verifying that these two methods are best suited for imputation.

If one examines the distributions of the errors for `FFill` and `Kalman`, see the figures in Appendix F, one can see that the variance of the residuals are reduced with `Kalman`; a good indication of improvement. On the other hand, for some of the variables `FFill` has a more pronounced peak at zero, indicating that this method is better at finding the exact value. The effect is most evident for the variables *TimeDialyzed* and *AverageBloodFlowRate*, which are patient specific, fixed settings and seldom changed. Choosing the previous value would thus be a reasonable thing to do. However, `Kalman` is reducing the variance of the errors for both variables and we cannot tell if it is more important to hit zero error often or to reduce the overall error when using the imputed data for predictions. Further investigation of the variable *AverageBloodFlowRate* shows that it has five pronounced peaks, see Figure 2.4. In Figure F.1 in Appendix F, two of these peaks are still visible at ± 50 for `FFill` but not for `Kalman`, indicating that `Kalman` is successful at capturing the multimodal behaviour in the variable.

The Welch's t-test and Kullback-Leibler (KL) divergence were used as statistical measurements to evaluate if the imputed data was statistically close enough to the validation data, see Tables 5.3 and 5.4. The p-values of the Welch's t-test showed that most methods and variables imputed values such that the null hypothesis of equal means could not be rejected, i.e. a p-value above 0.05 indicates that the null hypothesis at the 5% level cannot be rejected. For example, the p-value of the variable *AverageBloodFlowRate* for imputation method `FFill` is 0.677, which indicates that the null hypothesis of equal means most certainly could not be rejected. Looking at the next column, the p-value of the same variable for the imputation method `RF` is shown. This value is only 0.033 which indicates that we can reject the null hypothesis of equal means with a 0.05 significance and

thus we have shown that the means of the imputed values and validation values are significantly different. Continuing to the KL divergence it is fair to say that all imputed data sets show more similarity to the validation data than dissimilarity, since all values in Table 5.4 are closer to 0 than 1. This is reassuring results and can be interpreted as only a small loss of information when using imputed data to represent the validation data. However, the difference between the different methods might be more interesting to look at. Once again the `Kalman` imputation achieves the best result on average, but this time tied with `TS-EM`. In general it looks like the difference in results between the more sophisticated imputation methods and the easier ones are greatest for the variables with most amount of missing data (*LitersProcessed* and *FluidRemoved*). This could be an indicator that with increasing amount of missingness an increasingly complex model should be used to achieve the best results.

The results of the prediction algorithm was evaluated. Looking at the first row of Table 5.5 in Section 5.3 there seems to be a rather small dependence between AUC score and amount of missing data. The difference between 15% and 100% of the rows containing missing values is just 0.009 units when not imputing any values. This could be because the random forest classifier is very robust and that it efficiently can use the observed values for classification. What is interesting is to look at the result for the `Kalman` imputation. The AUC score is consistently about 0.020 units higher, independent of the amount of missingness, compared to leaving the missing values blank. This might be an indicator that imputing values at least improves the prediction algorithm slightly. Another interesting point is that the AUC score for `Kalman` imputation is rather constant. Both 15% missing and 100% missing gives an AUC score of 0.635. This could be a confirmation that the `Kalman` imputation is very robust to the amount of missing data, further strengthening its position as best imputation method. The imputation method used today, `FFill`, is slightly worse than `Kalman` for the original data but interestingly we see that the AUC score is decreasing with increasing missing data, again verifying that a more sophisticated method is preferable when the amount of missing data is increasing. Also, we see a big difference in the AUC score if we impute using either `FFill` or `Kalman`, suggesting that imputations are important to achieve better prediction results.

When we merge different methods we see a small reduction in NRMSE but not an increase in AUC score compared to `Kalman`. Considering the merged method between `Kalman` and `RF`, it is definitely questionable if it is better than using `Kalman` since the AUC score for the merged method was reduced. Also, considering the merged method of `Kalman`, `RF`, and `MS`, it seems like there is not much to gain by imputing the missing values in a patient-by-patient fashion, even when naïve imputation methods are used to mitigate the problem of short sequence lengths.

6.5 Performance of algorithms

Below is a small summary of the performance of all algorithms. The main strengths and weaknesses for each method are listed.

- `FFill`
  - Strengths: Easy to implement, readily available in many libraries; fast.
  - Weaknesses: Bad imputations.
- `RF`

  - Strengths: Can handle mixed-type data.
  - Weaknesses: No within-variable dependencies (this can be added as extra variables); slow.

- `Kalman`
  - Strengths: Within-variable dependencies.
  - Weaknesses: No between-variable dependencies (this can be added as input to the system); requires enough data to fit an ARIMA model; assumes Gaussian distributed data.

- `TS-EM`
  - Strengths: Both within- and between-variable dependencies.
  - Weaknesses: Unstable at some sequence lengths; requires enough data to fit an ARIMA model; assumes Gaussian distributed data.

- `MS`
  - Strengths: Easy to implement; performs better than `FFill`
  - Weaknesses: Imputing the same value, always.

- `MICE`
  - Strengths: Can handle continuous and categorical variables.
  - Weaknesses: Requires a lot of data to work.

- `kNN`
  - Strengths: Both within- and between-variable dependencies.
  - Weaknesses: No good if distance to neighbors are too far.

- `JM`
  - Strengths: Performs well with a lot of data.
  - Weaknesses: No within variable dependencies (this can be added as extra variables); requires a lot of data; assumes Gaussian distributed data.

6.5.1 Optimization of models

During this project the main objective has been to implement a variety of models rather than optimizing them to perform as well as possible on this specific data set. Therefore a lot of work can be done to fine-tune the algorithms. Most of the models have parameters to tweak, and to achieve the best possible imputations each variable and each patient would probably need its own specific settings. For example, using the `Kalman` imputation method, one could build a specific ARIMA filter, optimizing the parameters for each patient and variable. Increasing the number of trees in the `RF` model could also result in slightly better imputations. Another way to possibly enhance the performance of the algorithm is to have one generic main model for the entire data set as prior and then use a patient-specific model for each patient in the data set.

To further investigate the distribution of each variable and to find more suitable transformations could also help to improve the performance of the methods or make it easier to fit appropriate models to the data. As we have seen, some of the variables are far from Gaussian distributed. It should also be considered to cluster the data into different subgroups, e.g. according to sex, age, or ethnicity, to possibly build better suited models.

6.6 Ethical considerations

As always when handling medical data the well-being and privacy of the patient is central. There are strict rules and regulations to be followed and it is of great importance to keep the data secure and encrypted. It could be devastating for an individual if his or her records were compromised and most work involving big data analysis is dependent on not violating that.

Imputing values for medical data is mathematically not any different from imputing any other kind of data. However, a poorly imputed value could change the outcome of the prediction algorithm in the consecutive step, which might classify a patient as stable when he or she actually is in risk of hospitalization. Or, it could, the other way around, classify a stable patient as in risk of hospitalization, making the physicians consider preventive actions which possibly could worsen the condition of the patient. Therefore the algorithms developed quite literally can be the difference between life and death. This brings up the question of how much one can and should trust the algorithm as well as who is to blame if something goes wrong. If the algorithm says that there is a great risk for a patient to be hospitalized and preventive measures should be considered, but a physician decides to ignore it, is the physician to blame if the patient dies? And the other way around, if the physician follows the advice presented by the algorithm and the patient dies, are the developers of the algorithm responsible? There are no easy answers to these kind of questions and ultimately it is the responsibility of the state to come up with laws deciding what is "right".

Also, the lack of human contact has to be considered when using algorithms that are supposed to say something about a patient. All recorded variables have been handled by humans, and most likely the nurses taking care of each patient have great knowledge about the health of each individual. But all of the information one can gather by meeting a person face-to-face is, unfortunately, impossible to record. Hence the algorithm developers has to rely solely on a few recorded variables to make an opinion. This lack of human contact is the main reason to not only use algorithms to diagnose a patient. There is simply too much and too subtle information in the meeting between two persons to be able to record and use it all in an algorithm.

6.7 Future work

The next step in this project would be to impute values of more variables. Since the prediction algorithm uses around 200 variables for its predictions it is likely that more of them are in need of imputation. One would also have to further investigate if the imputed values really make a difference for the prediction algorithm, and if so specify a threshold to know at which amount of missingness, imputation methods are worth to use. It would also be interesting to investigate other patterns of missing data and validate the performance of the different algorithms.

To better evaluate the performance and properness of the imputations, a simulation technique developed by Brand et al. [39] could be used. The main idea with this technique is to construct a large amount of copies of the incomplete data set and then use a stochastic imputation method to

impute all missing values. The average of each imputed point is then used as final imputation. A set of equations are then used to confirm if the imputations are proper or not, and the large amount of imputed data makes it possible to calculate more accurate statistics. This would increase the credibility of the imputations and make them even more trustworthy.

Chapter 7

Conclusions

In conclusion, all of the implemented algorithms performed better than the `FFill` method when measuring the average normalized root mean square error. This means that on average the newly imputed values had smaller errors than that of the imputed values using `FFill`. Of the methods investigated in this report the `Kalman` imputation method performed best on 10 out of the 11 variables. From this result it can be concluded that time series models might be preferable when handling this kind of data. However, since the `RF` imputation performed best on one variable, a merging of the two methods yielded an even better result considering NRMSE. Therefore, to lower the NRMSE even more, a mixture of models is to be preferred.

When investigating the performance of the algorithms on different sequence lengths, it was evident that the more sophisticated algorithms had trouble getting good results for too short sequences. Therefore, an additional mixture of models with `MS` imputing sequences with less than 40 measurements combined with a mix of `Kalman` and `RF` for all other sequences was investigated. The resulting NRMSE for this model was however not better than that of the merged `Kalman-RF` model.

Moreover, when investigating the performance of the prediction algorithm, comparing `Kalman` to `FFill`, we saw a slight improvement in the area under the curve score. For `Kalman` we managed to keep the AUC score at a constant level while increasing the amount of missing data, but for `FFill` the score decreased with higher rate of missingness. Using the merged `Kalman-RF` model, the AUC score was reduced compared to `Kalman` but increased compared to `FFill`. The other merged method, imputed in a patient-by-patient manner with `MS` for imputation of short sequences, gave a worse AUC score compared to `Kalman`, `Kalman-RF`, and `FFill`. It is thus not recommended to impute in a patient-by-patient manner, but rather use the entire data set at once. Also, we recommend to use sophisticated imputation methods to impute the missing data since that improves the predictions of hospitalizations.

Appendices

Appendix A

Distribution of *original data*

In this appendix we present the distribution plots of *original data*. Many of our methods assume the data to be normally distributed, which does not seem to be the case. However, some variables behave worse than others. For example *AverageBloodFlowRate* looks like some sort of multimodal Gaussian and *TimeDialyzed* has two pronounced peaks at 240 minutes and 210 minutes corresponding to 4 hours and 3.5 hours, respectively. These variables are machine settings and can therefore be expected to have distinct values. The other nine variables are of a more natural origin and are closer to a normal distribution. Nevertheless, they are not truly normally distributed, for example a Gamma distribution might be more valid for e.g. *FluidRemoved* and *EndSittingSystolicBP*.

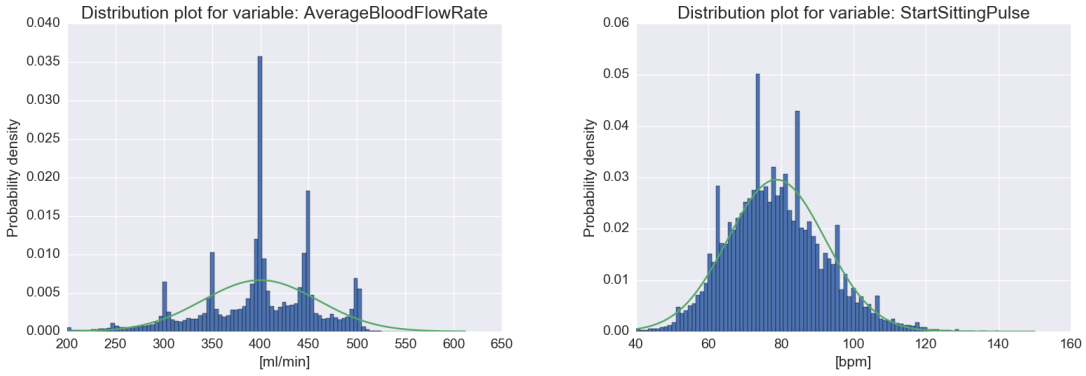


Figure A.1: Distribution plots of variables with normal curve for reference.

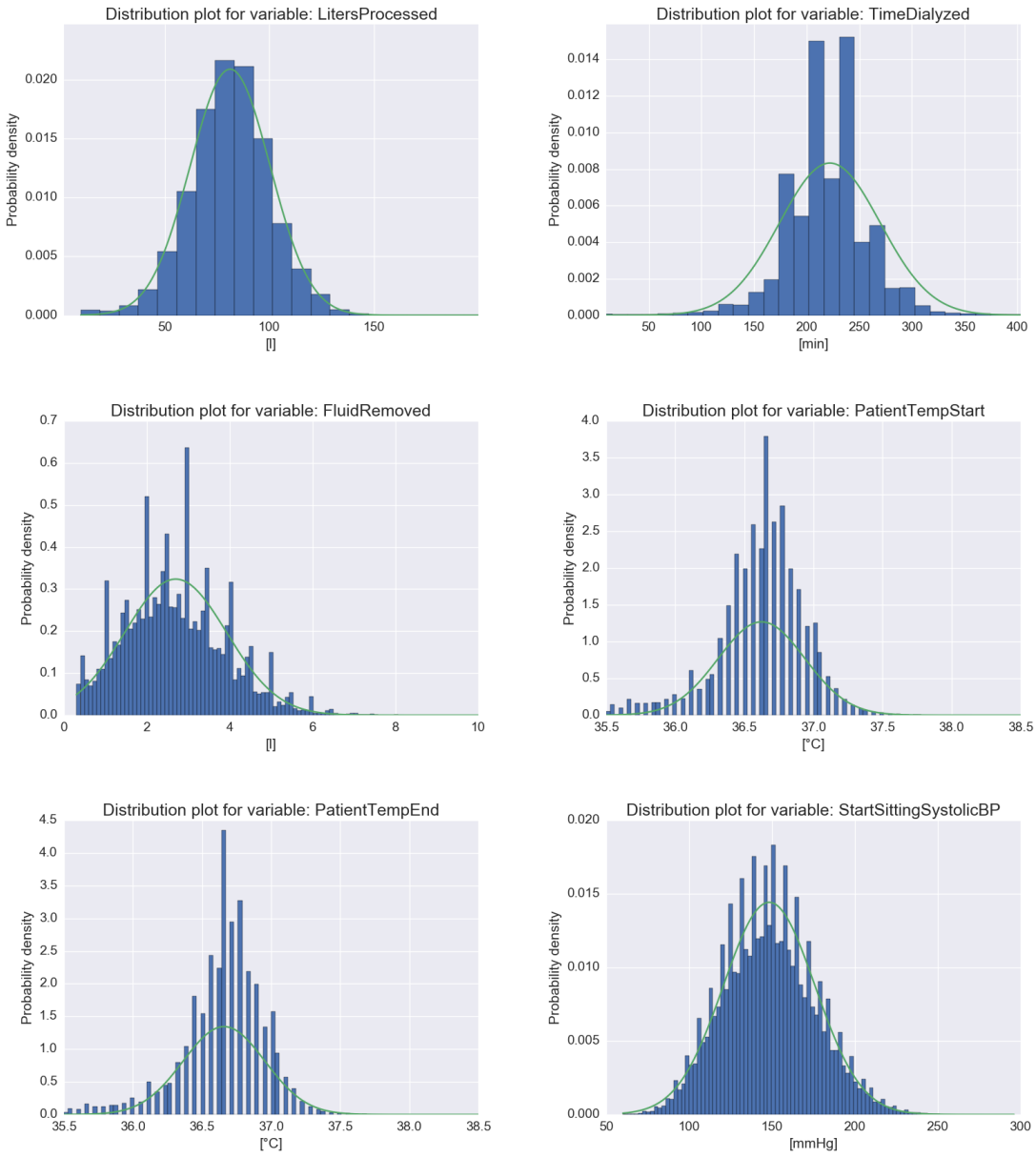


Figure A.2: Distribution plots of variables with normal curve for reference.

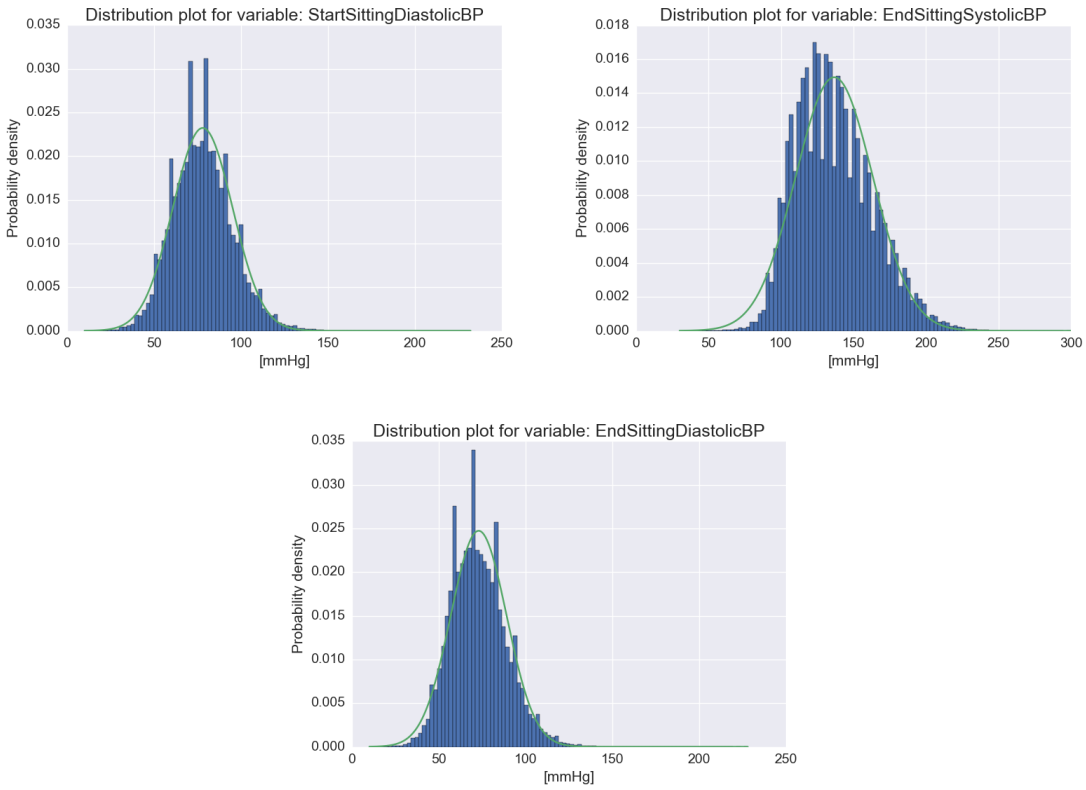


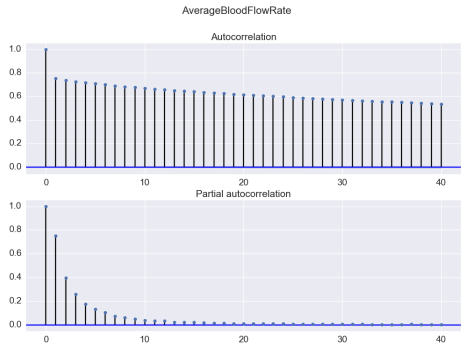
Figure A.3: Distribution plots of variables with normal curve for reference.

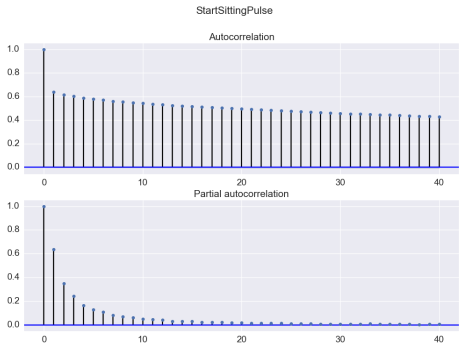
Appendix B

Model order selection for time series models

It was unfeasible to use the full data set when estimating the ARIMA(p,d,q) model. The decision of not useing the full data set was taken based on two reasons; (1) the iterative estimation techniques of the EM algorithm were time consuming and (2) by using the full data set the time order is distorted when we change between patients. Therefore a subset of patients where chosen and they were required to have more than 1000 measurements to be selected. The best model order were estimated to each patient and then the model order which coincide with most patients were selected. To select model order for the time series models we used the autocorrelation function (ACF) and partial autocorrelation function (PACF). We decided that the correct model order was found when the ACF and PACF av the residuals were statistically white.

In Figures B.2 and B.3 we present the ACF and PACF of all variables of one patient with the number of recorded measurements, $L > 1000$.





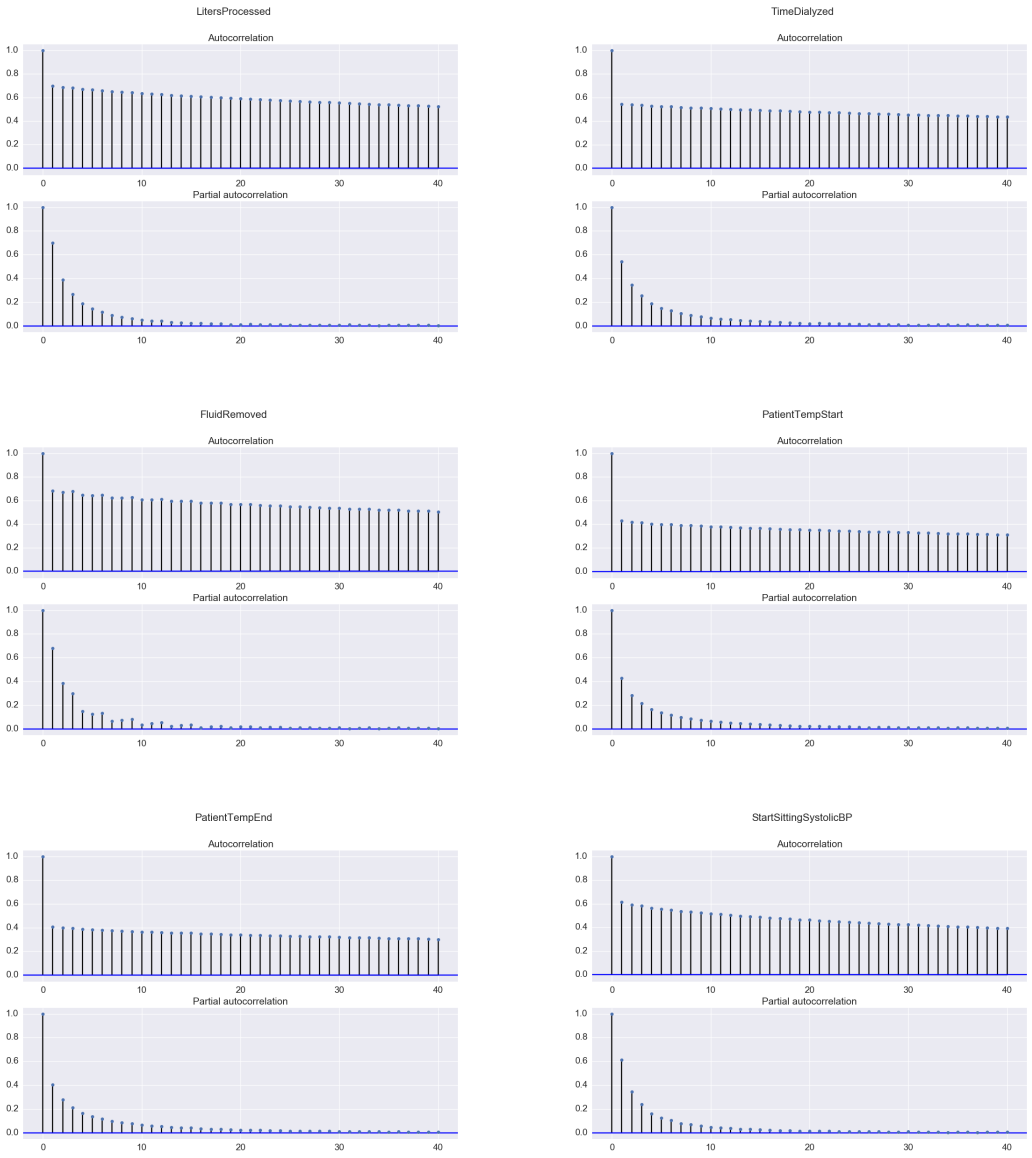


Figure B.2: Top: ACF of the original signal.
Bottom: PACF of the original signal.

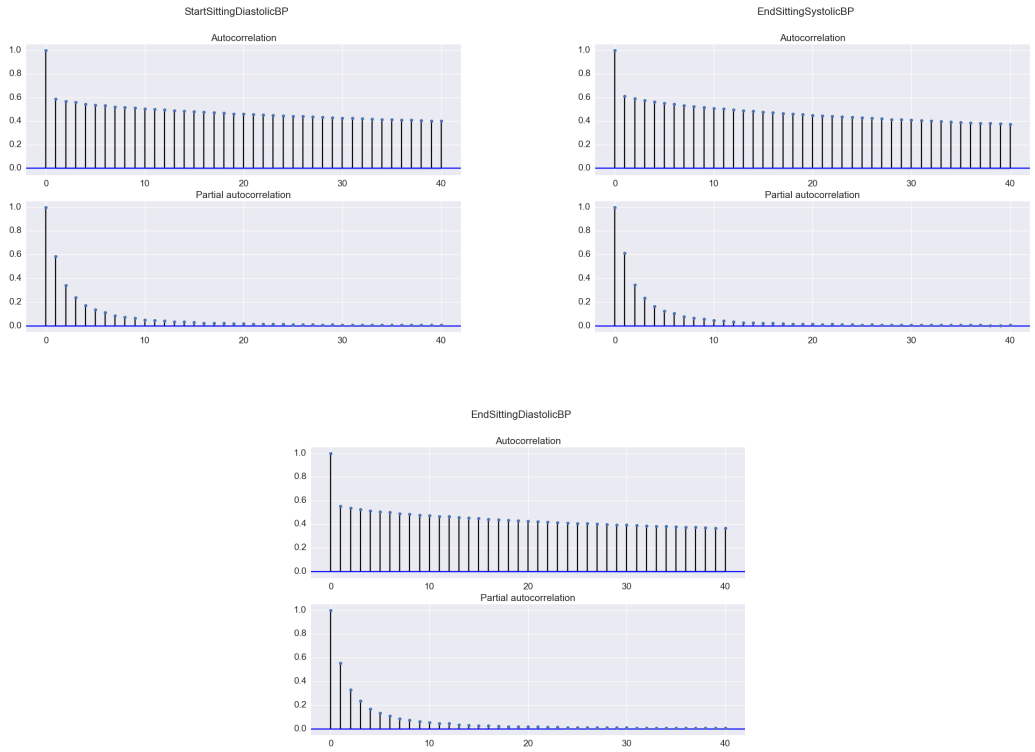


Figure B.3: Top: ACF of the original signal.
Bottom: PACF of the original signal.

It is evident that the variables behave similar. For simplicity, we decided that we wanted one model which could describe all variables reasonable well. One could argue that we should select different models for each variable but we chose to keep it as simple as possible. Using this model gave residuals that we considered to be almost white and thus a suitable model. The ACF and PACF of the residuals are shown in Figures B.4 and B.5.

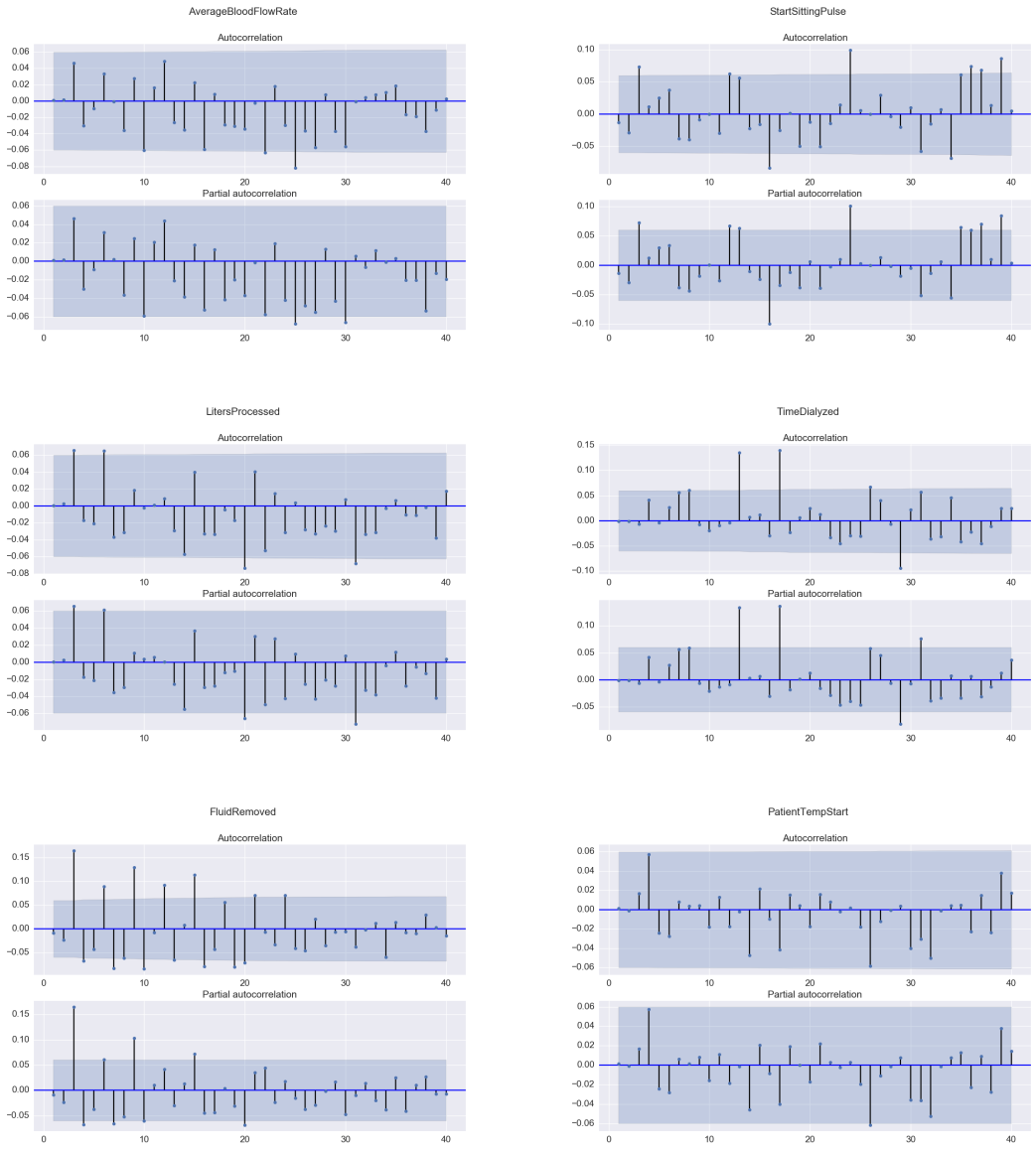


Figure B.4: Top: ACF of the residuals between the true signal and estimated signal.
Bottom: PACF of the residuals between the true signal and estimated signal.
The dark blue field represents the 95% confidence interval of the residuals being zero; the value at lag 0 is removed.

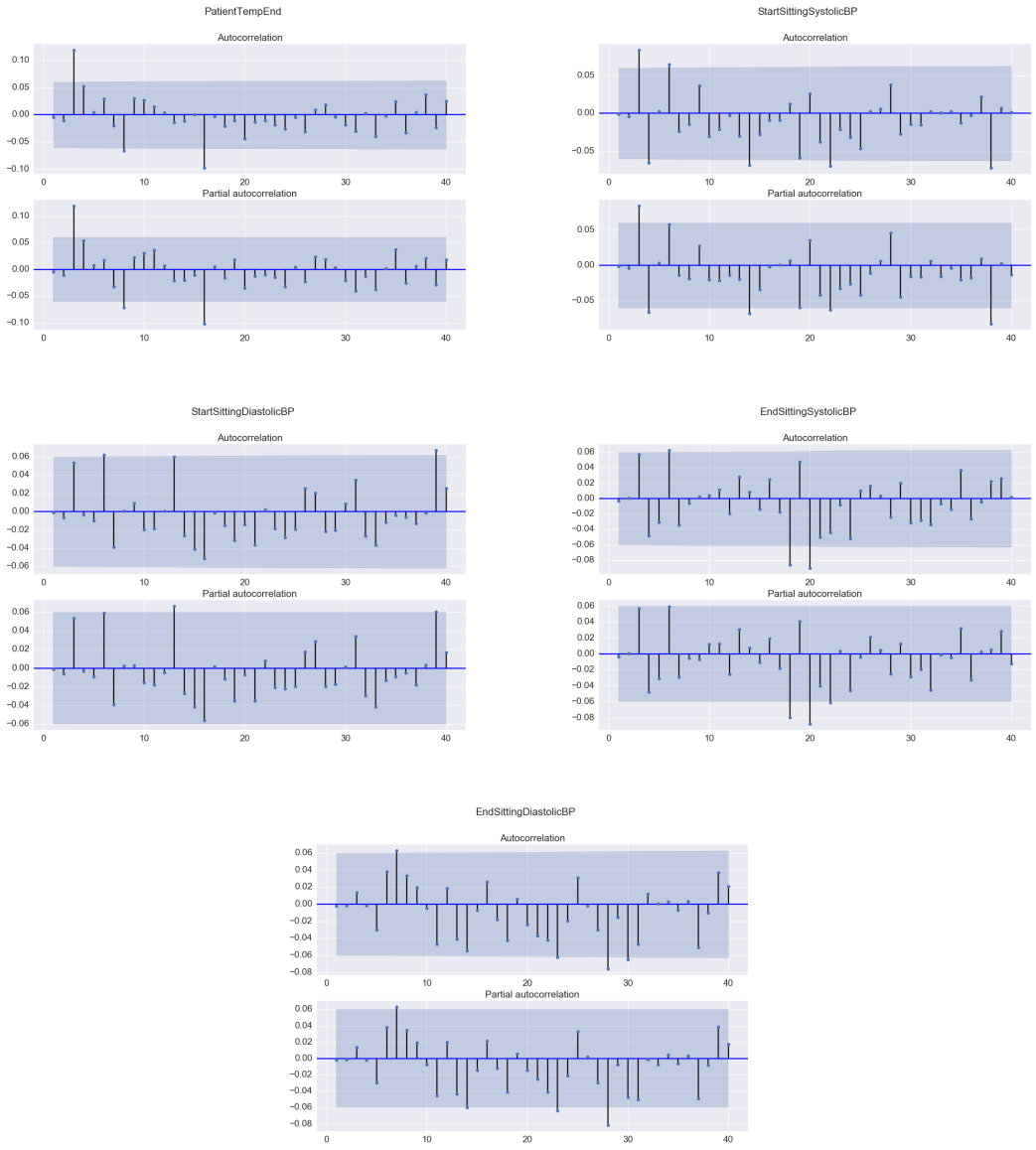


Figure B.5: Top: ACF of the residuals between the true signal and estimated signal.
Bottom: PACF of the residuals between the true signal and estimated signal.
The dark blue field represents the 95% confidence interval of the residuals being zero; the value at lag 0 is removed.

Appendix C

Preprocessing steps

Below the proceedings of preprocessing are listed, for each variable deemed to be in need of such. First the *Outlier Removal* algorithm is explained and thereafter the *Unit Conversion* algorithm.

C.1 Outlier removal

Elements containing any of the following outliers were set to NaN:

- Negative values, since all variables measured were done so in positive units.
- Average blood flow above 1000 ml/min and below 200 ml/min.

  The average blood flow through healthy kidneys is approximately 25% of the cardiac output, which amounts to 1100 ml/min in a 70-kg adult male [40]. The blood flow through diseased kidneys and through the dialysis machine, however, is much lower. In our data the average blood flow rate is around 300-500 ml/min and a high maximum boundary was therefore set to 1000 ml/min. Unfortunately not much data exists on what an optimal blood flow rate would be, but a blood flow rate above 700 ml/min in the blood pump is unlikely. The minimum blood flow rate was set to 200 ml/min after close examination of the data and relying on Sam et al. [41], who writes that the blood flow rate usually is between 500-800ml/min in conventional, thrice-weekly hemodialysis sessions.
- Pulse above 150 and below 40 bpm.

  A normal resting heart rate for healthy adults ranges from 60 to 100 beats per minute [42]. Dialysis patients often have a significantly higher pulse due to their illness and therefore the upper limit of the span was set to 150 bpm. A resting pulse of above 150 bpm would indicate serious distress and most likely a normal dialysis treatment would not be possible to carry out.
- Body temperatures above 40 and below 35.5 Celsius.

  Normal body temperature ranges between 36.5-37.5 °C [43]. It was argued that values outside of the range 35.5-40 °C would be unrealistic since patients with such extreme temperatures would already be hospitalized.

- Liters processed below 5 liters.

  If a patient processed less than 5 liters of blood it would imply that the treatment only went on for less than 20 minutes. This is clearly a to short time for dialysis and therefore the boundary was set to 5 liters.

- Systolic blood pressure above 290 or below 60 mmHg.

- Diastolic blood pressure above 150 or below 30 mmHg.

  Normal blood pressure in an adult is approximately 120/80 mmHg [44]. For patients with renal diseases however, the blood pressure can vary a lot and typically dialysis patients have higher blood pressure than normal. Getting guidance from the American Heart Association [44] and accounting for the varying blood pressures in dialysis patients, the systolic blood pressure boundaries were set to 60-290 mmHg and diastolic blood pressure boundaries were set to 30-140 mmHg. The blood pressure will typically decrease a lot after a dialysis treatment since a lot of excess fluid is removed. Therefore the threshold to count as an outlier was set very low for those particular variables.

C.2 Unit Conversion

The units of the following outliers were converted to the right scale. The process was highly empirical and done in the order presented below:

- Fluid removed
  - >1000 l were divided by 1000.
  - >100 l were divided by 100.
  - >8 l were divided by 10.
  - <0.005 l were multiplied by 1000.
  - <0.05 l were multiplied by 100.
  - <0.3 l were multiplied by 10.

    The fluid removed in our data typically ranged between 2-4 liters during a 3-4 hour dialysis session. There exists no rule of thumb of how much fluid a patient should remove per session and typically each patient after a while finds out how much to remove to achieve best results. To remove more than 1 liter per hour though, is rare and rates above that has been proven to increase cardiovascular mortality [45].

- Liters processed
  - >1000 l were divided by 100.
  - >200 l were divided by 10.
  - <12 l were multiplied by 10.
  - <2 l were multiplied by 100.

    The average amount of blood processed in our data was around 100 liters. Since the amount of blood processed is calculated by multiplying the blood flow rate with the total time dialyzed the conversions stated above makes sense.

C.3 Outcomes of preprocessing

Below are some examples of variables before and after preprocessing. The data in each plot is from the same patient.

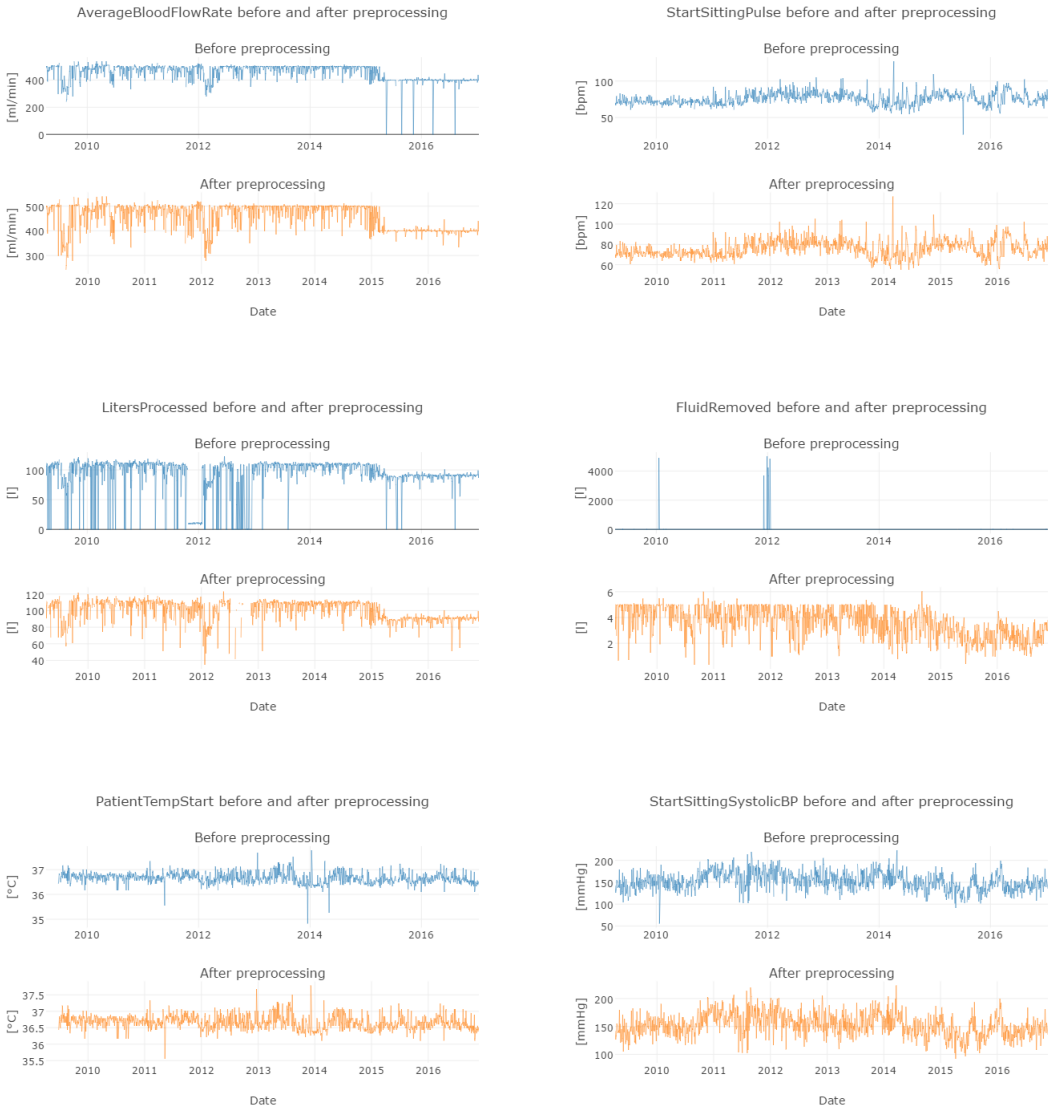


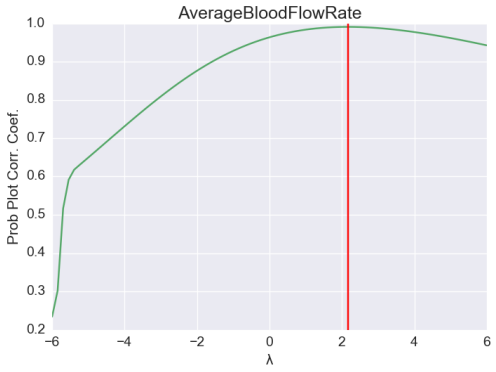
Figure C.1: Some variables before and after preprocessing

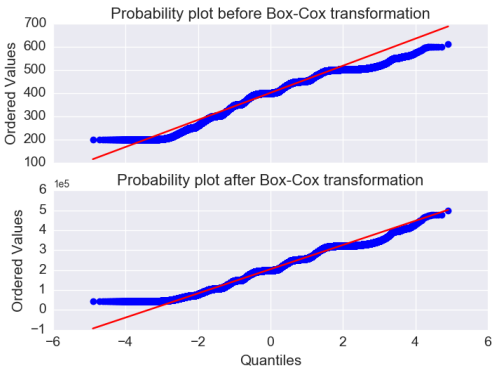
Appendix D

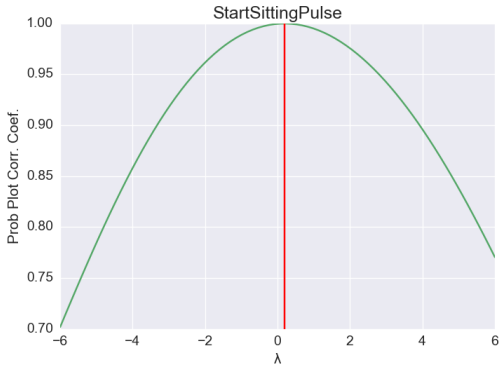
Transformation of data

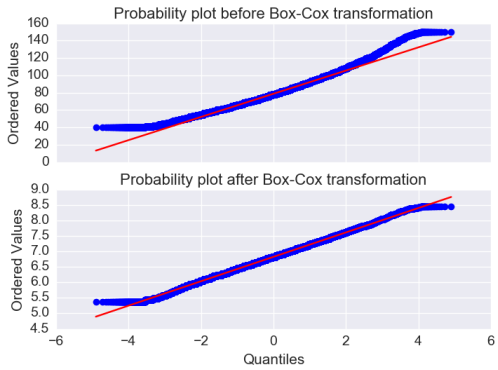
D.1 Box-Cox plots and transformations

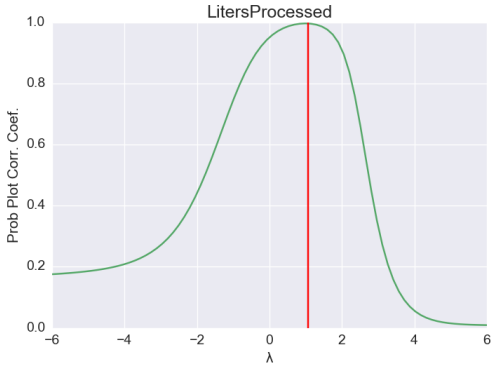
Box-Cox plots for all variables and probability plots for all variables, before and after transformation with λ as recommended by matching Box-Cox plot.

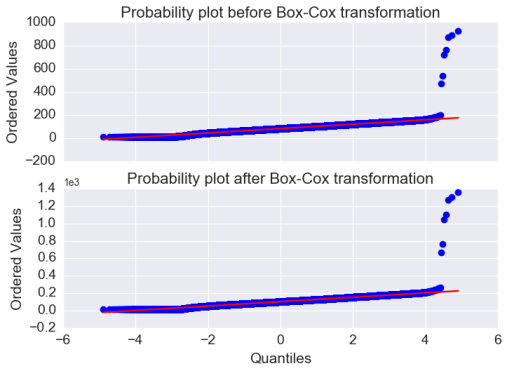


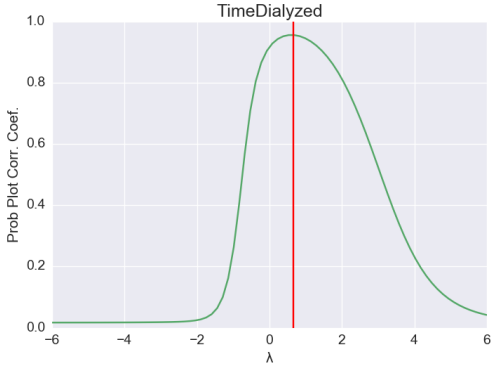


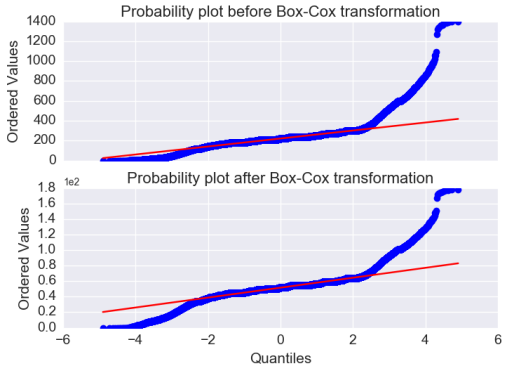

StartSittingPulse
Prob Plot Corr. Coef.
λ

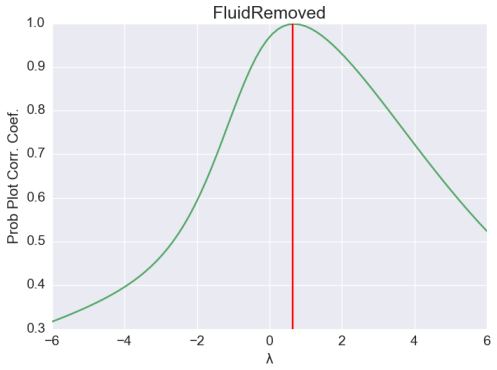

Probability plot before Box-Cox transformation
Ordered Values
Probability plot after Box-Cox transformation
Ordered Values
Quantiles

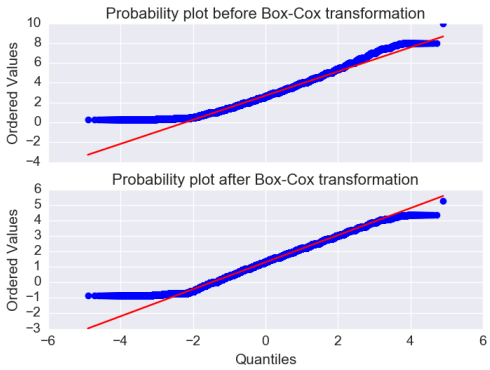

LitersProcessed
Prob Plot Corr. Coef.
λ

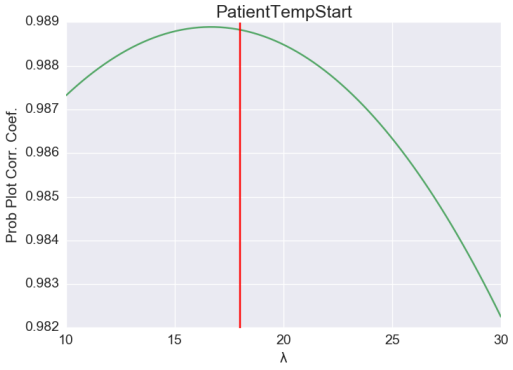

Probability plot before Box-Cox transformation
Ordered Values
Probability plot after Box-Cox transformation
1e3
Ordered Values
Quantiles

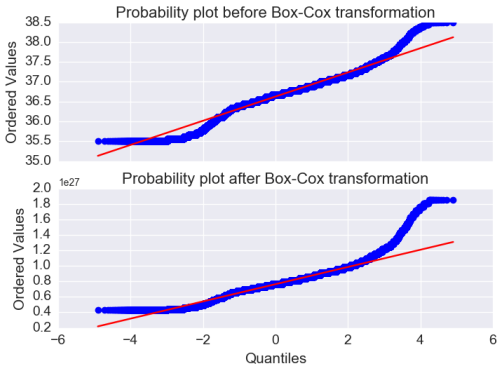

TimeDialyzed
Prob Plot Corr. Coef.
λ

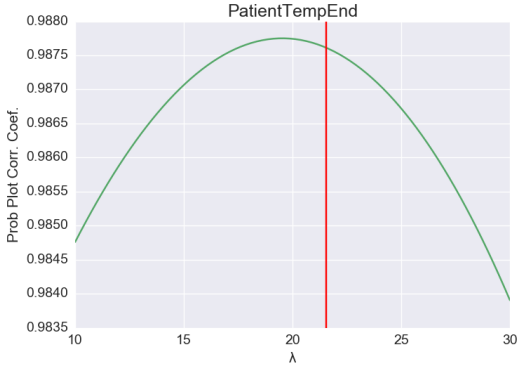

Probability plot before Box-Cox transformation
Ordered Values
Probability plot after Box-Cox transformation
1e2
Ordered Values
Quantiles

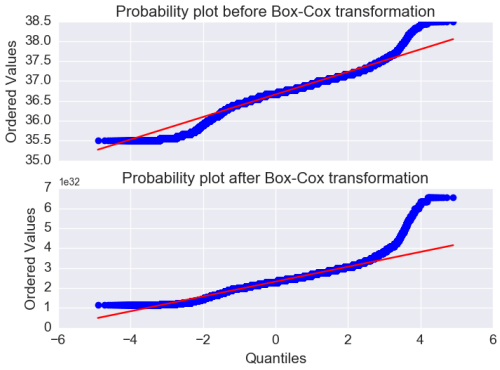

FluidRemoved
Prob Plot Corr. Coef.
λ

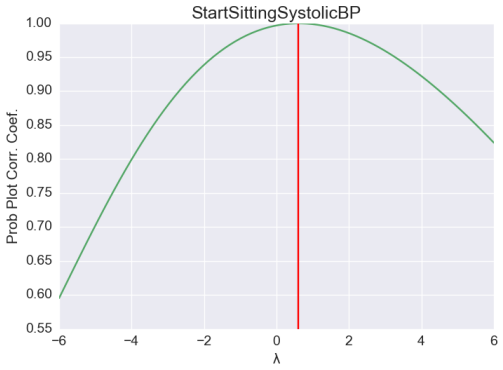

Probability plot before Box-Cox transformation
Probability plot after Box-Cox transformation
Ordered Values
Quantiles

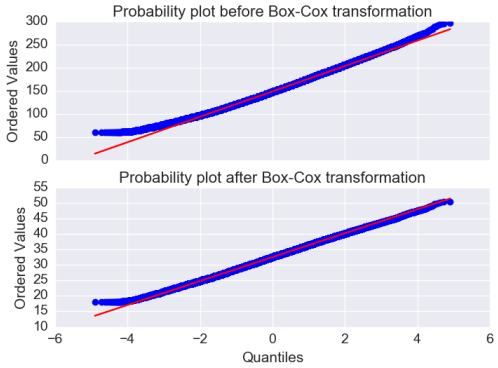

PatientTempStart
Prob Plot Corr. Coef.
λ

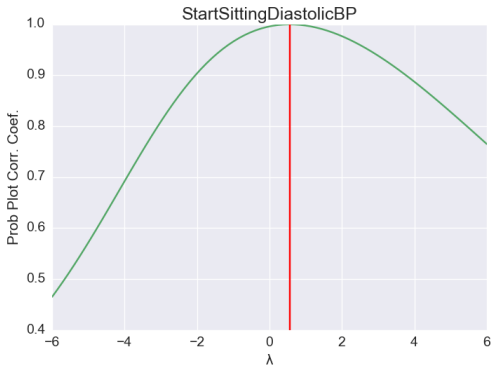

Probability plot before Box-Cox transformation
Probability plot after Box-Cox transformation
Ordered Values
Quantiles

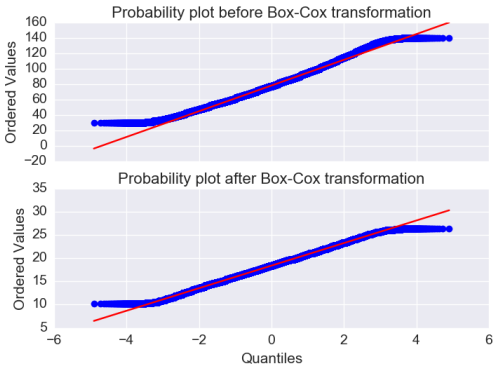

PatientTempEnd
Prob Plot Corr. Coef.
λ

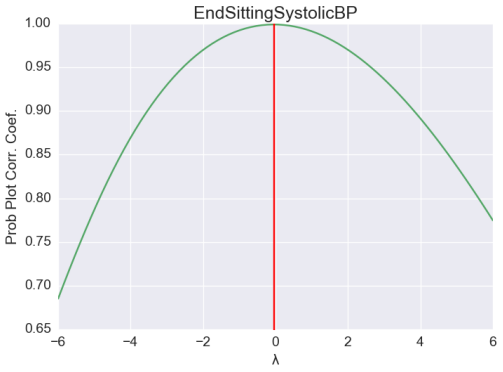

Probability plot before Box-Cox transformation
Probability plot after Box-Cox transformation
Ordered Values
Quantiles

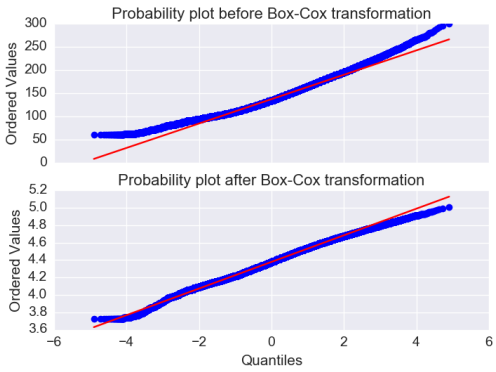

StartSittingSystolicBP
Prob Plot Corr. Coef.
λ

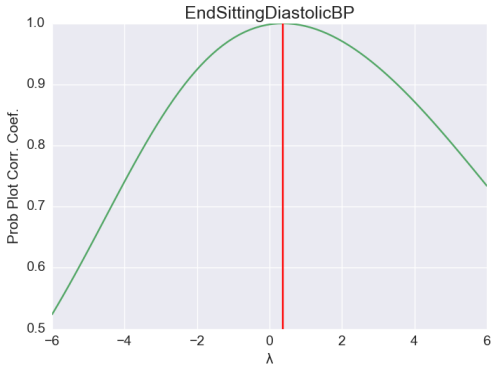

Probability plot before Box-Cox transformation
Probability plot after Box-Cox transformation
Ordered Values
Quantiles

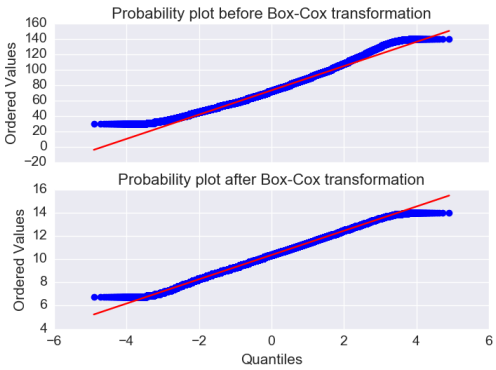

StartSittingDiastolicBP
Prob Plot Corr. Coef.
λ


Probability plot before Box-Cox transformation
Probability plot after Box-Cox transformation
Ordered Values
Quantiles


EndSittingSystolicBP
Prob Plot Corr. Coef.
λ


Probability plot before Box-Cox transformation
Probability plot after Box-Cox transformation
Ordered Values
Quantiles


EndSittingDiastolicBP
Prob Plot Corr. Coef.
λ
1.0
0.9
0.8
0.7
0.6
0.5
−6
−4
−2
0
2
4
6


Probability plot before Box-Cox transformation
Ordered Values
160
140
120
100
80
60
40
20
0
−20
Probability plot after Box-Cox transformation
Ordered Values
16
14
12
10
8
6
4
−6
−4
−2
0
2
4
6
Quantiles

Appendix E

Similarity measures

In this appendix we have collected the similarity measures for all methods and present them variable by variable. The confidence interval (CI) is built for the bias.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	-0.242	(-0.572,0.088)	0.751	0.863
SSP	0.016	(-0.436,0.468)	0.613	0.784
LP	-0.003	(-0.076,0.069)	0.699	0.073
TD	-0.143	(-0.709,0.422)	0.553	0.734
FR	0.003	(-0.003,0.009)	0.678	0.823
PTS	-0.000	(-0.004,0.004)	0.427	0.659
PTE	-0.002	(-0.007,0.003)	0.400	0.639
SSSBP	-0.497	(-1.507,0.513)	0.634	0.796
SSDBP	-0.156	(-0.802,0.490)	0.596	0.773
ESSBP	-0.508	(-1.193,0.177)	0.611	0.783
ESDBP	-0.143	(-0.579,0.292)	0.561	0.752

Table E.1: **Forward fill**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	0.695	(0.418,0.973)	0.815	0.899
SSP	-0.217	(-0.585,0.152)	0.692	0.808
LP	0.093	(0.050,0.136)	0.888	0.940
TD	-0.849	(-1.301,-0.397)	0.662	0.797
FR	0.005	(0.000,0.010)	0.749	0.849
PTS	0.002	(-0.001,0.005)	0.566	0.704
PTE	0.000	(-0.003,0.004)	0.533	0.679
SSSBP	-0.546	(-1.439,0.346)	0.680	0.813
SSDBP	-0.274	(-0.824,0.276)	0.665	0.803
ESSBP	-0.488	(-1.097,0.122)	0.643	0.794
ESDBP	-0.402	(-0.769,-0.036)	0.633	0.785

Table E.2: **Random forest**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	-0.087	(-0.335,0.161)	0.848	0.912
SSP	0.039	(-0.298,0.377)	0.749	0.848
LP	0.010	(-0.043,0.064)	0.818	0.893
TD	-0.061	(-0.349,0.470)	0.723	0.821
FR	0.002	(-0.002,0.007)	0.800	0.882
PTS	0.001	(-0.002,0.004)	0.620	0.737
PTE	0.000	(-0.004,0.003)	0.597	0.715
SSSBP	-0.221	(-0.997,0.556)	0.761	0.852
SSDBP	0.117	(-0.385,0.618)	0.720	0.822
ESSBP	-0.391	(-0.913,0.130)	0.738	0.840
ESDBP	-0.199	(-0.522,0.124)	0.714	0.818

Table E.3: **Kalman**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	-0.087	(-0.335,0.161)	0.848	0.912
SSP	0.039	(-0.298,0.377)	0.749	0.848
LP	0.010	(-0.043,0.064)	0.818	0.893
TD	-0.061	(-0.349,0.470)	0.723	0.821
FR	0.002	(-0.002,0.007)	0.800	0.882
PTS	0.001	(-0.002,0.004)	0.620	0.737
PTE	0.000	(-0.004,0.003)	0.597	0.715
SSSBP	-0.221	(-0.997,0.556)	0.761	0.852
SSDBP	0.117	(-0.385,0.618)	0.720	0.822
ESSBP	-0.391	(-0.913,0.130)	0.738	0.840
ESDBP	-0.199	(-0.522,0.124)	0.714	0.818

Table E.4: **TS-EM**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	-0.021	(-0.362,0.320)	0.686	0.792
SSP	0.165	(-0.224,0.553)	0.648	0.765
LP	0.023	(-0.042,0.088)	0.721	0.823
TD	-0.013	(-0.430,0.454)	0.671	0.779
FR	0.003	(-0.003,0.008)	0.694	0.803
PTS	0.002	(-0.001,0.005)	0.549	0.672
PTE	0.000	(-0.004,0.003)	0.531	0.652
SSSBP	-0.169	(-1.104,0.766)	0.624	0.744
SSDBP	0.030	(-0.542,0.602)	0.613	0.739
ESSBP	-0.449	(-1.074,0.176)	0.587	0.712
ESDBP	-0.201	(-0.566,0.165)	0.610	0.733

Table E.5: **Mean substitution**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	0.001	(-0.406,0.426)	0.459	0.552
SSP	0.141	(-0.348,0.631)	0.280	0.371
LP	-0.003	(-0.064,0.057)	0.759	0.852
TD	-0.201	(-0.777,0.376)	0.239	0.285
FR	0.001	(-0.006,0.007)	0.400	0.501
PTS	-0.000	(-0.004,0.003)	0.433	0.475
PTE	-0.002	(-0.006,0.003)	0.379	0.424
SSSBP	-0.297	(-1.291,0.696)	0.557	0.670
SSDBP	-0.113	(-0.722,0.496)	0.541	0.627
ESSBP	-0.542	(-1.191,0.107)	0.541	0.662
ESDBP	-0.451	(-0.840,-0.061)	0.536	0.662

Table E.6: **MICE**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	-1.48	(-1.801,-1.166)	0.740	0.848
SSP	0.325	(-0.095,0.745)	0.602	0.763
LP	-0.711	(-0.760-0.662)	0.854	0.920
TD	-0.046	(-0.527,0.434)	0.612	0.763
FR	-0.014	(-0.019,-0.008)	0.708	0.831
PTS	0.005	(0.000,0.011)	0.285	0.485
PTE	0.001	(-0.005,0.007)	0.270	0.468
SSSBP	-0.010	(-0.979,0.958)	0.615	0.774
SSDBP	0.084	(-0.527,0.694)	0.581	0.752
ESSBP	-0.335	(-0.961,0.292)	0.614	0.774
ESDBP	-0.132	(-0.510,0.245)	0.606	0.767

Table E.7: **kNN**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

	bias	**CI**	$\boldsymbol{\rho}$	**d**
ABFR	-0.494	(-0.797,-0.191)	0.772	0.873
SSP	-0.027	(-0.420,0.367)	0.647	0.789
LP	-0.283	(-0.337,-0.229)	0.823	0.905
TD	-0.600	(-1.112,-0.088)	0.556	0.729
FR	-0.052	(-0.057,-0.046)	0.676	0.816
PTS	0.000	(-0.003,0.004)	0.503	0.678
PTE	-0.003	(-0.007,0.001)	0.464	0.651
SSSBP	-0.439	(-1.364,0.485)	0.660	0.805
SSDBP	-0.274	(-0.856,0.307)	0.629	0.784
ESSBP	-0.220	(-0.849,0.410)	0.625	0.785
ESDBP	-0.504	(-0.889,-0.118)	0.602	0.769

Table E.8: **Joint model**; CI: 95% confidence interval for the bias. ρ: Pearson's correlation between true values and imputed values. d: Agreement index between true values and imputed values.

Appendix F

Distribution of errors

In this appendix the distribution of errors for the imputation methods `FFill` and `Kalman` are compared. Note that `FFill` often has a wider distribution, but also a more prominent peak at zero.

Figure F.1: Distribution plots of errors.

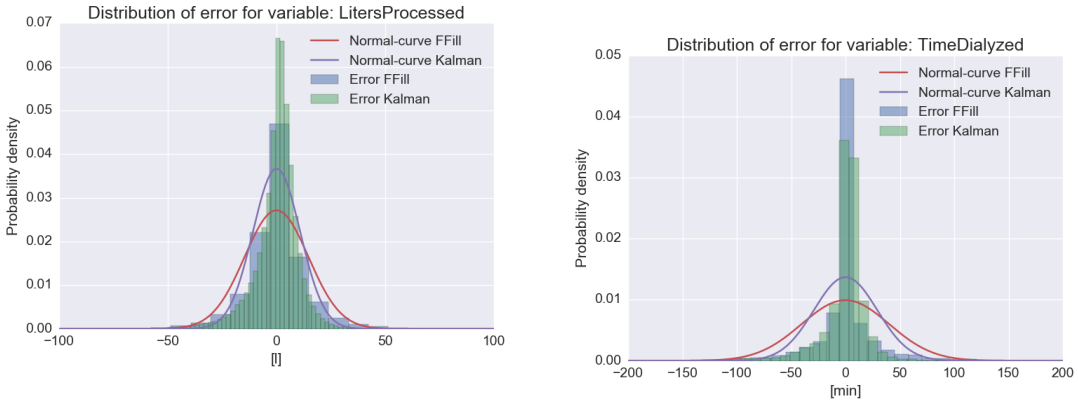


Figure F.2: Distribution plots of errors.

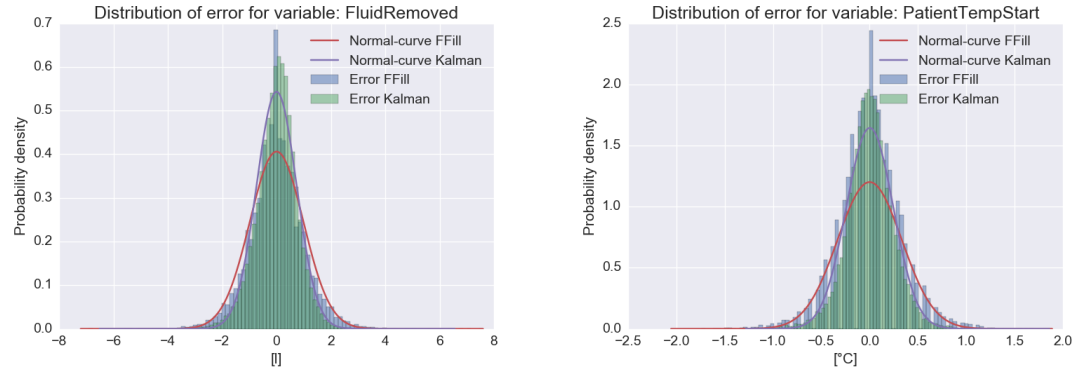


Figure F.3: Distribution plots of errors.

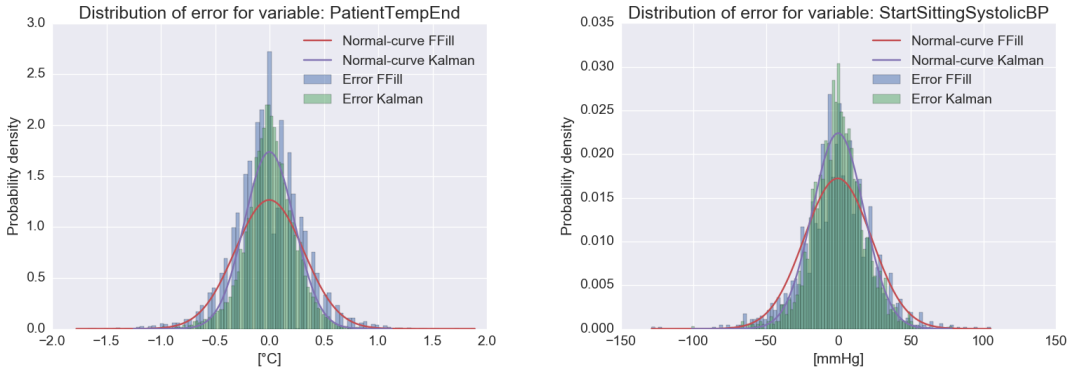


Figure F.4: Distribution plots of errors.

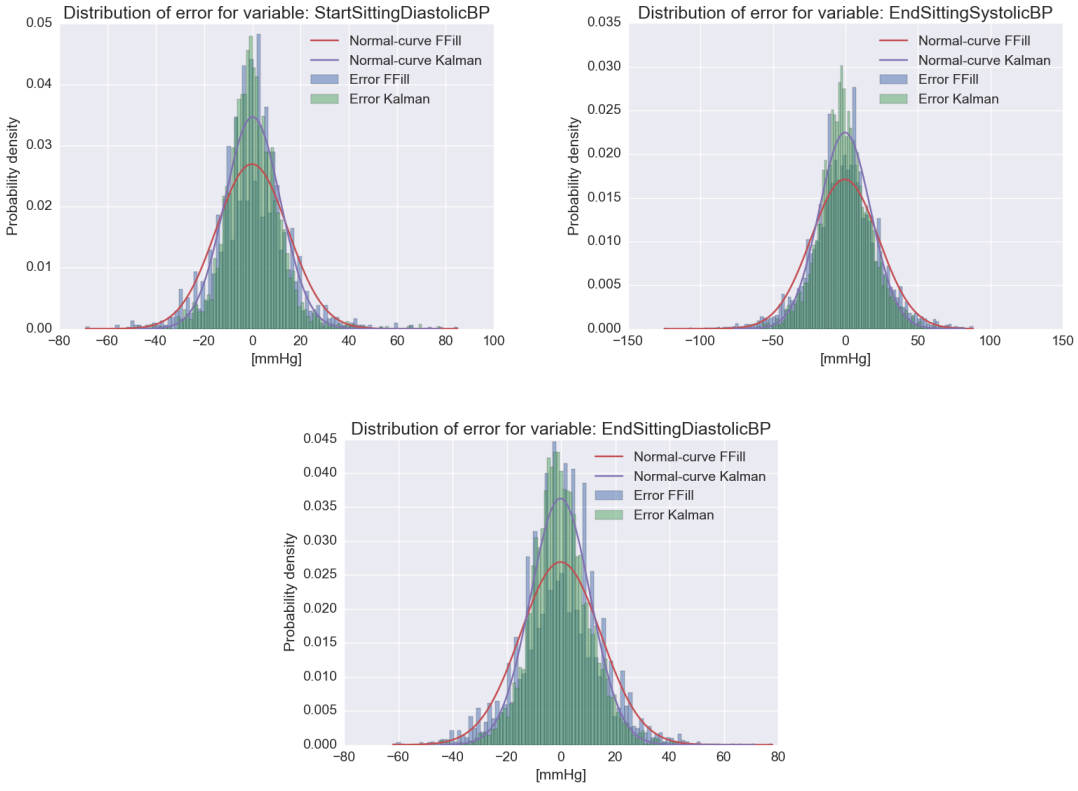


Figure F.5: Distribution plots of errors.

Bibliography

[1] A. S. Levely, et al, *Chronic kidney disease as a global public health problem: Approaches and initiatives - a position statement from Kidney Disease Improving Global Outcomes*, Kidney International, Vol. 72, nr 3, pp.247-259, 1 August, 2007.

[2] National Kidney Foundation, https://www.kidney.org/kidneydisease/global-facts-about-kidney-disease, accessed 2017-02-22, updated March 2015.

[3] T. R. Fleming *Addressing missing data in clinical trials*, Ann Intern Med, Vol. 154, pp. 113-117, 2011.

[4] W. R. Shadish, T. D. Cook, and D. T. Campbell, *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*, Wadsworth Publishing, Florence, KY, 2nd edition, 2001.

[5] E. D. De Leeuw, J. J. Hox, and D. A. Dillman, *International Handbook of Survey Methodology*, Lawrence Erlbaum Associates, New York, 2008.

[6] D. A. Dillman, J. D. Smyth, and L. Melani Christian, *Internet, Mail, and Mixed-Mode Surveys: The Tailored Design Method* John Wiley Sons, New York, 3rd edition, 2008.

[7] R. M. Groves, F. J. Fowler Jr., M. P. Couper, J. M. Lepkowski, E. Singer, and R. Tourangeau, *Survey Methodology* John Wiley Sons, New York, 2nd edition, 2009.

[8] R. J. A. Little and D. B. Rubin, *Analysis With Missing Data second edition*, Wiley series in Probability and Statistics, 2002.

[9] J. L. Schafer, *Analysis of Incomplete Multivariate Data*, Chapman & Hall/CRC, 1997

[10] S. van Buuren, *Flexible Imputation of Missing Data*, Chapman & Hall/CRC, Interdisciplinary Statistics Series, 2012.

[11] D. B. Rubin, *Multiple Imputation for Nonresponse in Surveys*, New York:John Wiley & Sons, Inc., 1987.

[12] N. Shara, S. A. Yassin, E. Valaitis, H. Wang, B. V. Howard, et al., *Randomly and Non-Randomly Missing Renal Function Data in the Strong Heart Study: A Comparison of Imputation Methods*, PLoS ONE 10(9): e0138923.. doi:10.1371/journal.pone.0138923, 2015.

[13] M. E. Montez-Rath, W. C. Winkelmayer and M. Desai, *Adressing Missing Data in Clinical Studies of Kidney Diseases*, Clin J Am Soc Nephrol, Vol. 9, pp. 1328-1335, 2014.

[14] M. C. M. Moniek et al. *Multiple imputation: dealing with missing data*, Nephrology Dialysis Transplantation, vol 28, pp. 2415-2420, 2013.

[15] National Kidney Foundation, *K/DOQI clinical practice guidelines for chronic kidney disease*, 2002, Retrieved 10 March 2017.

[16] K. Jameson, S. Jick, K. W. Hagberg, B. Ambegaonkar, A. Giles and D. O'Donoghue, *Prevalence and management of chronic kidney disease in primary care patients in the UK*, Int J Clin Pract, Vol. 68, nr 9, pp 1110-1121, September, 2014.

[17] L. Kelly, *Essentials of Human Physiology for Pharmacy*, CRC PRESS Pharmacy Education series, 2005.

[18] M. Abbasi, G. M. Chertow and Y. N. Hall, *End-stage renal disease*, BMJ Clin Evid, 2010.

[19] J. T. Daugirdas, *Dialysis Time, Survival, and Dose-targeting Bias*, Kidney International, Vol. 83, pp. 9-13, 2013.

[20] O. Troyanskaya, et al, *Missing value estimation methods for DNA microarrays*, Bioinformatics, Vol. 17, no. 6, pp.520-525, 2001.

[21] M. A. Tanner, W. H. Wong, *The Calculation of Posterior Distributions by Data Augmentation*, Journal of the American Statistical Association, Vol. 82, nr. 398, pp. 528-540, 1987.

[22] A. P Dempster, N. M. Laird, D. B. Rubin, *Maximum Likelihood from Incomplete Data via the EM Algorithm*, Journal of the Royal Statistical Society, Vol. 39, pp. 1-38, 1977.

[23] J. Wishart, *The generalised product moment ditribution in samples from a normal multivariate population*, Biometrika, Vol. 20A, nr. 1-2, pp.32-52, 1928.

[24] S. van Buuren, J. P. L. Brand, C. G. M. Groothuis-Oudshoorn and D. B. Rubin, *Fully conditional specification in multivariate imputation*, Journal of Statistical Computation and Simulation, Vol. 76, nr. 10, pp. 1049-1064, 2006.

[25] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[26] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning*, Springer, New York, 2nd edition, 2009.

[27] L. Breiman, *Random Forests*,Machine Learning, Vol.45:5-32, 2001.

[28] D. J. Stekhoven and P. Bühlmann, *MissForest–non-parametric missing value imputation for mixed-type data*, Bioinformatics, Vol. 28:112-118, 2012.

[29] A. Jakobsson, *An Introduction to Time Series Modeling*, Studentlitteratur AB, 2015.

[30] J. Durbin, S. J. Koopman, *Time Series Analysis by State Space Methods*, Oxford Statistical Science Series, 2nd edition, 2012.

[31] S. Oba et al., *A Bayesian missing value estimation method for gene expression profile data*, Bioinformatics, Vol. 19, nr. 16, pp. 2088-2096, 2003.

[32] C. J. Willmott, *Some comments on the evaluation of model performance*, Bulletin American Meteorological Society, Vol. 63, nr. 11, pp. 1309-1369.

[33] B. L. Welch, *The generalization of "Student's" problem when several diffferent population variances are involved*, Biometrika, Vol 34, nr 1-2, pp 28-35, 1947.

[34] S. Kullback and R. A. Leibler, *On information and sufficiency*, Annals of Mathematical Statistics, Vol 22, nr 1, pp 79-86, 1951.

[35] G. E. P. Box and D. R. Cox, *An Analysis of Transformations*. Journal of the Royal Statistical Society B, Vol 26, pp. 211-252, 1964.

[36] W. L. Junger, A. Ponce de Leon, *Imputation of missing data in time series for air pollutants*, Atmospheric Environment, Vol. 102, pp. 96-104, 2015.

[37] M. J. Azur, E. A. Stuart, C. Frangakis and P. J. Leaf, *Multiple Imputation by Chained Equationes: What is it and how does it work?*, Int J Methods Psychiatr Res., March 1; 20(1): 40-49, 2011.

[38] N. Hansson, *Deep learning for prediction of hospitalisations of dialysis patients*, http://www.bibliotek.dtu.dk/english/servicemenu/publish/student, 2017.

[39] J. P. L. Brand et al., *A toolkit in SAS for the evaluation of multiple imputation methods*, Statistica Neerlandica, Vol. 57, nr. 1, pp. 36–45, 2003.

[40] D. C. Eaton and J. P. Pooler, *Vander's Renal Physiology*, Lange Medical Books/McGraw-Hill, 8th edition, 2004.

[41] R. Sam, *Hemodialysis: Diffusion and Ultrafiltration*, Austin. J. Nephrol. Hypertens. Vol.1, nr.2, 2014.

[42] *"Target Heart Rates"*, American Heart Association, 4 April 2014, Retrieved 27 April 2017.

[43] D. Karakitsos and A. Karabinis, *Hypothermia therapy after traumatic brain injury in children*, N.Engl.J.Med, Vol.359, nr.11, pp. 1179-1180.

[44] *"Understanding blood pressure readings"*, American Heart Association, 11 January 2011, Retrieved 27 April 2017.

[45] J. E. Flythe, S. E. Kimmel and S. M. Brunelli, *Rapid Fluid Removal During Dialysis is Associated With Cardiovascular Morbidity and Mortality*, Kidney Int., Vol 79, nr.2, pp.250-257, 2011.

Master's Theses in Mathematical Sciences 2017:E26
ISSN 1404-6342

LUTFMA-3318-2017

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/