Efficient Methods for Handling Missing Data

Angelina Steffens

Born 9th July 1994 in Altenkirchen

17th March 2020

Master's Thesis Mathematics

Advisor: Prof. Dr. Michael Griebel

Second Advisor: Dr. Christian Rieger

Institute for Numerical Simulation

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Abstract

In this work, several approaches to handling missing data are analyzed. Discarding incomplete data during an analysis completely is a very popular approach, but it is usually not recommendable as there are many better ways to address missing values in a data set without loosing the information given by the incomplete data.

The statistical concept of imputation is applied to substitute missing values with plausible replacements. A wide variety of methods for handling missing data are described, ranging from a simple imputation using the mean of the data as the substitute value to more complex iterative completion methods utilizing dimensionality reduction approaches. The introduced methods are then applied to artificially constructed data sets as well as real-world applications, where the resulting completed data and the performance of the methods are compared.

Contents

1. Introduction

Today, information has become a significant type of currency. Data gets increasingly more important as more and more data is available.
Until the era of *Big Data* found its way into the digital world, analysis methods for data were normally based on small data sets which were assumed to be complete and quite noise-free as the data was usually collected for a specific problem. Nowadays however, data sets increase in size and the data acquisition process is usually no longer dependent on a specific task, such that the extraction of relevant information becomes more complicated.
An additional stumbling block in many practical applications is that some data points can be *incomplete*, where a data point x is said to be incomplete when some of its entries are missing or unspecified. Missing data is a common problem in most scientific research domains, from medicine (see for example [1], [2]) and psychology ([3]) to economics ([4]) and life sciences ([5],[6]), such that most scientists will most likely be confronted with the problem of missing data at some point. Since more often than not, further analysis with incomplete data is more complicated or even impossible, as most machine learning algorithms require that their inputs have no missing values, a major concern is the completion of the missing entries in a data set with as plausible values as feasible, possibly as a preprocessing step. In some cases, incomplete data can even lead to false interpretations of the data since the incomplete data sets can be misleading for statistical inference. Therefore, the missing entries in the data should be reasonably filled such that they do not falsely distort the quantities of interest.
Statistical analysis of data with missing values was revolutionized with [7] and [8] in the late 1980s, several still consistently employed missing data concepts are introduced therein. Today, many methods for dealing with missing data haven been developed and most statistical software includes packages for dealing with missing data. The field of missing data research is vast.

1.1 Missing Data Patterns and Mechanisms

Based on the descriptions and notations in [8], the rows of a data matrix usually represent units or observations and the columns represent characteristics or variables measured for each unit. It is useful to define patterns and mechanisms of missing data, where the patterns characterize the values which are missing and those which are observed, while the mechanisms concern the relationship between variables being missing and the values of those variables.

For a data matrix $Y = (y_{ij})$, $i = 1, \ldots, n$, $j = 1, \ldots, K$ encoding a data set of dimension $n \times K$, for n units and K variables, y_{ij} is the value of the variable Y_j for unit i.
A missingness indicator matrix $M = m_{ij}$ defines the *pattern of missing data* with the binary assignment

$$m_{ij} = \begin{cases} 1 & \text{if } y_{ij} \text{ is missing} \\ 0 & \text{if } y_{ij} \text{ is observed.} \end{cases}$$

A common alternative notation is the response indicator matrix R with reverse assignments. The following methods are restricted to the case where M is completely known, such that there are no *unknown unknowns*, where additional assumptions have to be made as the location of the missing data is not known, which may occur in survey data (see [9]). This still covers many practical applications.
There are several common patterns and it may be beneficial to sort the data matrix in order to detect if a pattern exists, as some methods are applicable to certain patterns of missingness. In addition, a pattern may provide practical information to facilitate any further analysis. For instance, missing values may be confined to one specific variable which might not be pertinent to the problem that is to be investigated and may therefore be deleted from the data set. The remaining data would then be complete such that any analysis would be more easily conducted without loosing any relevant information. However, in the subsequent chapters, the pattern of missing data is only secondary and will not be further discussed in depth.
Additionally, as introduced in [7], it is important to distinguish between unit and item non response. *Unit non response* occurs if a subset of units has no observable variables and *item non response* presents itself as missing values for certain variables of a specific unit. In the subsequent chapters only item non response and not unit non response will be treated as the methods for handling the two problems differ.

Assuming that the rows (y_i, m_i) are independent and identically distributed over i, the *missingness mechanism* is characterized by the conditional distribution of m_i given y_i, $f_{M|Y}(m_i|y_i, \theta)$, for the unknown parameter θ. It is possible to distinguish between three different categories of missing data mechanisms.
The data is called **missing completely at random (MCAR)** when the missingness depends neither on unobserved data nor on observed data. The probability of having a missing value does not depend on either the known values or the missing data, such that the cause of the missing data is unrelated to the data itself. More formally, for all i and distinct y_i, y_i^* in the sample space Y:

$$f_{M|Y}(m_i|y_i, \theta) = f_{M|Y}(m_i|y_i^*, \theta)$$

The missingness mechanism is called **missing at random (MAR)** if the missingness may depend on the observed data, but not on the unobserved data. The probability of a value being missing is the same only within groups defined by the observed data. For the value y_i of unit i, missingness depends on y_i only through the observed components $y_{(0)i}$ and not on the missing components $y_{(1)i}$, such that for all i and distinct $y_{(1)i}, y_{(1)i}^*$:

$$f_{M|Y}(m_i|y_{(0)i}, y_{(1)i}, \theta) = f_{M|Y}(m_i|y_{(0)i}, y_{(1)i}^*, \theta)$$

The data is **missing not at random (MNAR)**, whenever the missingness may depend also on the unobserved data. The probability of having a value that is missing varies for reasons that are unknown to us. The distribution of m_i depends on the missing components of y_i, such that the equation above does no longer hold for all i and $y_{(1)i}, y_{(1)i}^*$.

The three types of missing data mechanisms arise from different real-world problems. MAR is a much broader class than MCAR, which is often unrealistic, although it is nearly impossible to prove that data is MAR. MNAR is more complex and concepts are very different from those in case of MAR as they are more targeted on finding the causes of missingness. It is important to note that data which is MCAR will have unbiased estimates for the most primitive methods, such as the shortly after presented mean imputation, while data that is MAR leads to unbiased estimates for adequate methods, such as the later introduced regression imputation, such that for both missing data mechanisms general methods are available. For data that is MNAR, there is no quick fix. Most missing data methods presume that the data is MAR and that is the reason why the focus in the following chapters will be on methods using the MAR assumption.

1.2 Neural Networks and Recommender Systems

Neural networks and deep learning are considered state of the art in a vast range of application domains these days. In the following paragraphs, some basic concepts related to neural networks as well as a few neural network approaches to handling missing data are briefly introduced. A more extensive explanation of the topics covered in this section would go beyond the scope of this work. A popular reference for further detailed information on neural networks and deep learning is [10].

Artificial intelligence is a thriving field with many practical applications and active research topics. In early days, many intellectually difficult problems were easily solved, because they could be described by a list of formal rules. The challenge were tasks that are easy for people to perform, but hard to describe formally. The solution was to learn from experience.

Machine Learning uses the system's ability to acquire its own knowledge, by extracting patterns from raw data.

Deep Learning introduces a hierarchy of concepts, with each concept defined through its relation to simpler concepts, which has become increasingly popular particularly due to huge improvements in computer infrastructure.

Two neural network approaches to missing data based on distinct problems are autoencoders and generative adversarial networks.

An **autoencoder** is a neural network with the combination of an encoder function, which converts the input data into a different representation, and a decoder function, which converts the new representation back into the original format, such that as much information as possible is preserved, see chapter 14 in [10].

For an application of autoencoders to missing data, see for example [11] for an approach to incomplete electronic health data.

A **generative adversarial network (GAN)**, introduced in [12], consists of two neural networks, the discriminator and the generator, contesting against each other. The generator produces examples trying to imitate instances of the space of instances, while the discriminator tries to differentiate between fake and real examples.

Several adaptions of generative adversarial networks to incomplete data are available, see for instance [13]. There are many applications for missing data available as well, see [14] for an example of GANs for image imputation and [15] for time series imputation with GANs.

Thus, neural networks can be used to handle missing data. However, it is necessary to have a training set with complete data available and most neural network approaches today are constructed for specific large data sets. Hence, this might not be a useful technique for small incomplete problems.

Particularly since the Netflix Prize competition started in 2006[1], matrix completion and recommender systems have attracted more attention and both have become an even more popular field of research.
The Netflix Prize problem consisted of a movie-ratings matrix and the goal was to predict user ratings for movies, based on other ratings without any other information about the users or movies (see [18]). Thus, the Netflix Prize was an example of an application of matrix completion, which is the activity of filling in the missing entries of an incomplete matrix, very often done with a low rank matrix.
The goal was to find the best collaborative filtering algorithm, which is a strategy used by recommender systems to suggest items of interest to people based on their own preferences by collecting preference information from many users. Recommender systems are widely used in many commercial applications, for more details see for example [16] and [17].
Recommender systems are a special class of methods for dealing with a matrix completion problem. The analysis of possible approaches is an interesting current research topic related to missing information, but will not be further discussed since that is not within the framework of this work.

[1] see netflixprize.com

1.3 Outline

This work is subsequently structured as follows.
Chapter 2 will introduce the concept of imputation, a statistical approach to inferring missing data from statistical quantities or models. Starting with the simple option of using single imputation methods in section 2.1 and going forward to the newer and usually more plausible multiple imputation in section 2.2, which accounts for uncertainty from missing data by imputing the data several times. Another concept that will be shortly introduced in section 2.3 is that of maximum likelihood based techniques which does not impute the data directly but rather analyzes the incomplete data.
In chapter 3, linear and nonlinear dimensionality reduction methods will be introduced, first the most common linear dimensionality reduction methods principal component analysis, then a generalization with kernels, the so-called kernel principal component analysis and last, the nonlinear manifold method diffusion maps, see section 3.1. These three methods may then also be used for the completion of missing values, first with rather simple concepts in section 3.2 and then with more complicated completion approaches in section 3.3.
In the two chapters, algorithms for the handling of rather general data sets with missing data will be introduced, as they are applicable to a wide variety of applications. The list of methods cannot be exhaustive as the number of available methods is very large. The methods here are chosen based on their common use and their potential for application. More specific problem-based methods for handling missing data will not be examined as that would go beyond the scope of this work.
Chapter 4 will present several performance criteria for the evaluation of the given methods in section 4.1 and the methods' efficiency for a constructed simple linear example as well as an example with added noise will be analyzed in sections 4.2 and 4.3, respectively.
Chapter 5 will then study the results computed with the methods based on two different real-world applications, a set of images in section 5.1 and a bundle of time-series in section 5.2, where many more data sets with missing data are available since the problem of missing data is all-encompassing.
Since the focus of the completion methods included is not based on specific examples, as the given methods may also be applied to images and time series data, sometimes called longitudinal data, image completion and the completion of time series with possibly more suitable completion approaches for these specific uses will not be further addressed. For a short overview, the two types of data sets may be completed with special methods as follows.

A specific concept for image analysis is total variation (see [19]), the total variation norm is the proper norm for images (see [20]).
There are special TV norm based methods for images, which may be motivated by the fact that, since L^2 based methods return smooth results, such as the later describe principal component analysis, an L^1 based approach to image restoration and image denoising is preferred since images are non-smooth, especially in relation to edge detection. A disadvantage is that L^1 based estimation methods are nonlinear and computationally complex (see [21]).
Methods especially used for time series completion (see[22]) include some of the later described methods such as filling the last observation forward, interpolation and variants of the expectation-maximization algorithm, as well as methods specifically developed for time series, for example moving averages (see [23]) or Amelia (see [24]).
In chapter 6, a short conclusion follows.

2. Imputation - A Statistical Concept

In statistics, many concepts for dealing with missing data exist.
One possible way of treating missing data is to discard the affected data entirely. The so-called **complete case analysis** uses list-wise deletion to remove all incomplete data points. However, even though it is very popular and the standard setting in most statistical software, this is not recommended in the majority of cases, since it decreases the power of the analysis due to the unused partial data and can also lead to false interpretations.
Therefore, the missing entries in the data should be estimated in a meaningful way. The methods in the following sections are concerned with several possible solutions for inferring the missing values from the existing data from a statistical point of view.
Imputation is a statistical approach for handling missing data by replacing it with substituted values. A good introduction and a thorough resource for statistical analysis with missing data and imputation methods is [9] and [8], respectively. As such, this chapter is largely based on the information given therein.

2.1 Single Imputation Methods

In this section, a short introduction to several single imputation methods is given, where every missing value is replaced with an estimation. Single imputation can be seen in contrast to multiple imputation, which will be explained in the next section, where the value to be substituted is estimated multiple times.

Univariate Methods
A simple method of single imputation replaces missing data with a constant value. The constant can be chosen a priori for all entries of the data set or the imputed value can be based on a calculated central tendency of the data, for example the mean, median or mode of each feature.

Mean imputation for a feature $x \in \mathbb{R}^n$ with missing data replaces the missing values with the arithmetic mean $\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i$ of all m non-missing values in the feature.
Median imputation substitutes missing values in a feature with the median of the existing values, which is the middle value of the feature in the data set sorted by size.
Mode imputation finds the values that appear most often in the data set and replaces the missing data in each feature with the most frequent value for each feature.
These methods are **univariate**, they only use the non-missing values in the feature dimension of the missing data that is to be substituted.
Since for univariate single imputation methods, the imputed value is based on the feature, the axis on which the computation of the mean, median or mode respectively takes place has to be carefully identified for the data set. The default setting in most algorithms for two dimensional data sets encoded in a matrix is based on the feature being specified by the column. The imputation is therefore calculated with all non-missing values in the column of the feature with missing data that is being considered.
These imputation methods are an easily and quickly calculated possibility for imputation of data sets, but they are not very accurate in most cases and may lead to false conclusions. Additionally, their use might be ill-advised since they disregard correlations between features, systematically underestimate the variance since the values are constant and may introduce bias. In general, it can be concluded that imputation by a constant should be avoided.

Another category of imputation techniques is **hot-deck imputation**, where the imputation values are randomly chosen from a set of similar variables in the data set to fill the missing values. More accurately, hot-deck imputation replaces missing values of a feature with observed values from a different feature that are similar to the incomplete feature with respect to observed characteristics. In contrast, the so-called **cold-deck imputation** chooses the imputation value from a previously collected data set.
One simple form of hot-deck imputation called **last observation carried forward (LOCF)** fills a missing value with the last given value prior to it in the column of the ordered data set. The ordering can be based on several factors, for example the time of the observation. A similar method is **next observation carrier backward (NOCB)** which fills missing data by imputing the first value given after the missing entry.
Both methods have similar disadvantages as imputation by a central tendency, with the potential introduction of a bias and possible false interpretations and are therefore also not recommended in most applications.

Multivariate Methods

As opposed to the previously described univariate imputation methods, **multivariate** imputation algorithms use the entire set of available feature dimensions to estimate the missing values.

The first multivariate imputation method to be introduced is **regression imputation**, where the missing value is replaced by the value predicted by a regression analysis on the available variables. Hence, the relationship between features in the data set is preserved to a certain extent. In more detail, firstly a multivariate model is built from the observed data. Then in a second step, missing values are replaced with predictions calculated for the missing data with the multivariate model.

Regression models are applied in a large variety of fields using statistical analysis, with many theoretical sources available, see for example [25] for one of the earliest publications or [26] for one of the more modern books. To give a short overview, a regression analysis is used to find relationships between sets of variables that are statistically significant and to predict trends in the data. For example, assume that a regression model is to be fitted for a data set $\{x_i, y_i\}_{i=1}^m$, such that $x_i = \{x_{i1}, \ldots, x_{in}\}$. Selecting one $i \in \{1, \ldots, m\}$, multiple regression analysis predicts the dependent variable y_i by using the other independent variables $x_{i1}, \ldots, x_{in}$ which impact the dependent variable. A simple univariate regression uses only one independent variable x_{i1} for regressing y_i, such that $n = 1$.

There are many models in regression analysis, however the most common one is linear regression, where y_i is a linear combination of the parameters $a_0, \ldots a_n \in \mathbb{R}$ using a model of the type

$$y_i = a_0 + a_1 x_{i1} + a_2 x_{i2} + \ldots + a_n x_{in} + \epsilon_i = \mathbf{x_i}^\top \mathbf{a} + \epsilon_i,$$

with the vector of independent variables $\mathbf{x_i} = (1, x_{i1}, \ldots, x_{in})^\top$ and the parameter vector $\mathbf{a} = (a_0, \ldots, a_n)^\top$. An additive error term $\epsilon_i = y_i - \mathbf{x_i}^\top \mathbf{a}$ has to be introduced since under most real-world circumstances not all x_i lie on a hyperplane.

This model can be written for all i in matrix notation as

$$y = Xa + \epsilon,$$

with the vector of dependent variables $y = (y_1, \ldots, y_m)^\top$, the so-called design matrix $X = (\mathbf{x_1}^\top, \ldots, \mathbf{x_m}^\top)$ and the error matrix $\epsilon = (\epsilon_1, \ldots, \epsilon_m)$.

Technically, the linear regression model does not need to be linear in the independent variables, only in the parameters, such that a term of the form $\mathbf{x_i}^d, d > 1$ is also valid.
Multivariate regression models are a generalization with a vector-valued dependent variable which may be written as

$$Y = XA + E,$$

where Y is the matrix of dependent variables, X is the design matrix, A is the matrix of parameters to be estimated and E is the error matrix.
Ordinary least squares (OLS) is a very popular method for the parameter estimation. The objective is to minimize the sum of squared residuals $\hat{\epsilon_i}$,

$$\min_a \sum_{i=1}^{m} \hat{\epsilon}_i^2 = \min_a \sum_{i=1}^{m} (y_i - \mathbf{x_i}^\top a)^2.$$

Visually, these are the perpendicular distances between the data points and the regression line. The solution of the minimization problem is given by $\hat{a} = (X^\top X)^{-1} X^\top y$ and the predicted values are $\hat{y} = X\hat{a}$.
Since regression imputation is based on a parametric model, it is sensitive to the model itself and the imputation is only as good as the model. However, it takes a lot of effort to set up a reasonable model, such that typically for low computational effort, a standard model is used, for example a multivariate linear or multivariate normal model. The problem is that the imputed values fall directly on a regression line since they do not include an error term, which implies an overestimated correlation and an underestimated covariance.

For that reason, **stochastic regression** was introduced to supply some uncertainty about the value and reduce the bias. Now, the missing values are estimated by a regression approach plus a random residual value. This noise term improves the reproducibility of the relationships between variables in comparison with the deterministic approach.
However, stochastic regression imputation may lead to implausible values, such as negative values for a positive variable in the real data set. Additionally, it uses the same error term for all values whereas the actual error might differ by a lot between variables, for example the variability might be increasing in the ordered data set.

As a short excursus, even though not strictly a statistical technique, a similar approach is **interpolation** or rather extrapolation, an easy numerical method for handling missing values with an estimation based on other observations from the same feature. Extrapola-

tion is an estimation outside of the observation range and therefore more uncertain than interpolation which calculates estimations within the area of known observations.
Interpolation and extrapolation are usually explained in most introductory numerical analysis books, see for instance [27] or [28].
The simplest form of interpolation is linear interpolation, where two points are connected by a line. Outside of the interval given by the two points, this approach is known as linear extrapolation.
For more than two points which do not lie on a straight line, piecewise linear interpolation may be used. On a data set $\{(x_i, y_i)\}_{i=1}^{n}$, linear interpolants are given as straight lines between two points $\{x_j, y_j\}$ and $\{x_k, y_k\}$, $j \neq k, j, k \in \{1, \ldots, n\}$.
Linear interpolation may be easy to do, but it is not very precise. That is why one should work with a more general procedure. Polynomial interpolation tries to find the polynomial function

$$f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n,$$

with the lowest possible degree that passes through the data points such that $f(x_i) = y_i$. There exists a unique polynomial of degree at most $n - 1$ for n distinct data points. This is an infinitely continuously differentiable function and thus lies in the class C^∞.
Spline interpolation, where the interpolant is a piecewise polynomial, may be preferred to higher-dimensional polynomial interpolation, because it avoids the problem of oscillations at the edges of the interpolation interval, known as Runge's phenomenon. Polynomial pieces having degree $\leq n$ imply that the spline is of degree $\leq n$. The classical spline of degree n, is a function in C^{n-1} on the considered interval.
The degree of a polynomial or spline interpolation has to be properly chosen. The interpolation result may also depend on more than one variable. A simple multivariate interpolation method in more than one dimension is piecewise constant interpolation which locates the nearest neighbor, more precisely the value of the nearest point. It does not however consider any other values in the neighborhood.

This previously described method is the special case $k = 1$ of another widely used imputation method that is based on the **k-nearest neighbors algorithm (kNN)**.
KNN is a non-parametric machine learning algorithm that matches a data point with its k nearest neighbors. The number of neighbors to be considered, k, is a small integer that should be chosen based on the data. A small k increases the influence of noise while a large k tends to blur local effects. Since kNN makes no assumptions about the underlying distribution of the data, it may be used for a wide variety of data sets.

In a classification context, kNN allocates the training data into several classes based on specific characteristics and then assigns the test data on the basis of the classes of its k nearest neighbors in the training data. The prediction of the test data is typically based on the mode. Imputation with kNN usually replaces the missing data by the mean of the non-missing values of its k nearest neighbors in the training set. Two data points are close in this case if their non-missing features are close.
A distance measure needs to be introduced to quantify the similarity between data points. The distance metric should be chosen based on the properties of the data. For real-valued data, the Euclidean distance and Manhattan distance are usually very popular. While the Euclidean distance is beneficial for a data set with similar feature types, the Manhattan distance is useful for features which are not of a similar type.
The Euclidean distance is associated with the Euclidean norm, or L^2-norm, in an analytical context usually referred to as l^2-norm, and measures the length of the straight line between two points. More formally, for a point $x \in \mathbb{R}^n, n \in \mathbb{N}$ in the training set and a point $y \in \mathbb{R}^n$ in the test set, the Euclidean distance is calculated as the square root of the sum of the squared differences between x and y, such that

$$d(y,x) = \|x-y\|_2 = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}.$$

The Manhattan distance, associated with the L^1-norm, which may similarly to above also be referred to as l^1-norm, is calculated as the sum of the absolute differences between two points x and y,

$$d(y,x) = \|x-y\|_1 = \sum_{i=1}^{n}|x_i - y_i|.$$

Additionally, it is possible to introduce a weight for each neighbor, where standard kNN can be seen as using the weight $\frac{1}{k}$ for all k nearest neighbors. A major downside to kNN is the computational complexity. High-dimensional data incurs the curse of dimensionality. Moreover, kNN is susceptible to outlier.

All previously described single imputation strategies are easy to apply, but they have certain disadvantages. The main problem with most single imputation methods is that maximally likely values are computed and the standard errors are underestimated due to results having an overstated precision, since the statistical uncertainty in the imputations is neglected. Imputation results do not reflect any underlying distribution or real-world noise, but are rather treated with certainty as if all values were the actual true values.

2.2 Multiple Imputation Methods

To improve on the quality of the imputation results and account for the range of possible values, multiple imputation methods were developed.

Multiple imputation (MI) imputes missing values in a data set multiple times to account for uncertainty in the data and the imputations. It can be seen as a further development of simple imputation methods where missing values are only imputed once. Introduced by Donald Rubin in the 1970s and further developed in [7], it is now a common statistical approach to missing data, popular especially in social and economic experiments, and still a relevant research topic.

Multiple imputation is done in three phases: the imputation phase, the analysis phase and the pooling phase. In the first step, imputation is performed $m > 1$ times for incomplete data. The number of imputations m influences the result and is still a topic of discussion, see for example [29]. It may already be sufficient to use $3-5$ imputations for a moderate amount of missing information, but it is also theoretically beneficial to set m to a higher value, for a less biased outcome among other things, which will however need more computational time and a higher amount of storage. The value of m should hence in practice be adapted for the data set. A possible criterion for setting m is the average percentage of missing data. The imputed data sets are identical for the non-missing data, but they differ in the imputed values based on the method used for the imputation due to an added error term introduced for the uncertainty in the imputation.

The specification of the imputation model is arguably the most complex step in MI. Several possible approaches are available. Usually, imputed values are drawn from a simulated random distribution and as such are different for each missing entry. Stochastic regression may be used with its normally distributed error terms since they create different imputation results. Another possible single imputation method that also works for multiple imputation is hot-deck imputation where the random choice of the substituted value makes each MI result slightly different.

As a second step, the m completed data sets are all analyzed separately, as they would have been had the data been complete, for example to estimate parameters. The statistical analysis of the imputed data differs for all m data sets due to the differently imputed data that illustrates the uncertainty of the imputed value. In addition, it is important to analyze the spread of the imputed values. If all imputations are close to each other, then they have a low variance. Otherwise, the predicted value may not be very reliable.

In the last step, the analysis results are pooled into a final result. Pooling m sets of results into a single set of results is based on the so-called Rubin's rules as follows.

Using a notation similar to [8], the pooled estimate of m parameters $\hat{\theta}_i, i = 1, \ldots, m$ is simply the average of all m parameter estimates

$$\bar{\theta}_m = \frac{1}{m} \sum_{i=1}^{m} \hat{\theta}_i.$$

The variability associated with the pooled estimate $\bar{\theta}_m$ is composed of the average within-imputation variance, which captures the typical sampling variability,

$$\bar{W}_m = \frac{1}{m} \sum_{i=1}^{m} W_i,$$

for the associated sampling variances $W_i, i = 1, \ldots m$, and the between-imputation component,

$$B_m = \frac{1}{m-1} \sum_{i=1}^{m} (\hat{\theta}_i - \bar{\theta}_m)^2,$$

which captures the estimation variability due to missing data.
Thus, the total variability is

$$T_m = \bar{W}_m + \frac{m+1}{m} B_m,$$

where $(1 + 1/m)$ is an adjustment for finite m.
If done well, MI thus creates an unbiased estimate with the desired statistical properties that also preserves the uncertainty about the structure in the data.
An exemplary illustration of the three main steps in multiple imputation (imputation, analysis and pooling) for $m = 4$ different imputed data sets can be found in figure 2.1.

One of the preferred multiple imputation methods today is **multiple imputation by chained equations (MICE)**, also known as "fully conditional specification" or "sequential regression multiple imputation". A good introduction is given in [30] and a more thorough explanation can be found in [9].
Many multiple imputation procedures assume a large joint model for all of the variables, however these are rarely appropriate in large data sets with variables of varying types. MICE is a more flexible, iterative approach to these joint models, where a series of regression models is run for each variable modeled according to its distribution conditional upon the other variables in the data.
To elaborate on the process, MICE can be divided into several steps.

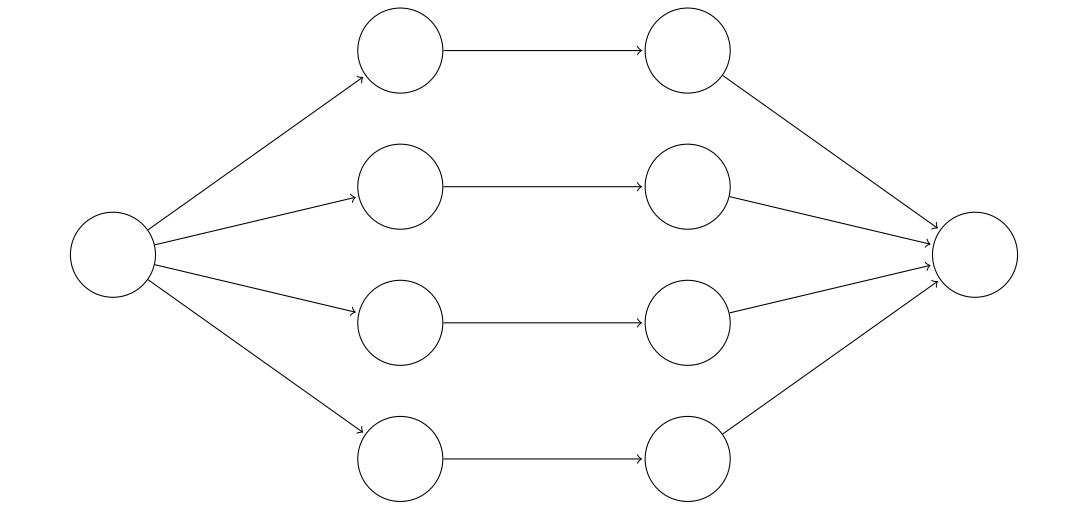


Figure 2.1: Illustration of the main MI steps based on a similar figure in [9].

In the first step, MICE performs a simple imputation for every missing value in the data set, for example a mean imputation. The imputed values work as place holders for the following steps.
Then, for every variable with missing data a regression model is applied to impute the missing values. In this case, the regression model can be chosen from a wide variety of possibilities. For example, given continuous variables, a modified linear regression model as well as special cases of hot-deck imputation may be used.
Bayesian and bootstrap multiple imputation modify normal linear regression models by adding noise and additionally including parameter uncertainty. Since the regression parameters are typically unknown for incomplete data sets, they must be estimated from the data and hence induce uncertainty which is thereby represented in the model. Bayesian methods draw the parameters directly from their posterior distributions, whereas bootstrap methods re-sample the observed data and re-estimate the parameters from the re-sampled data, for more information see for example chapter 3 in [9]. Predictive mean matching may be seen as a variation of hot-deck imputation which imputes values based on randomly chosen nearest observed values in the data set.
The iterative imputation step starts with one variable, for which all originally missing entries are regressed on the other variables in the imputation model, such that this variable is the dependent variable and all the other variables are the independent variables in the step.

The missing values in the dependent variable are then replaced with the regression predictions from the regression model. When this variable is now subsequently used as an independent variable in the regressions of all other variables, the observed and imputed values are used.
The previously described steps are repeated for each variable with missing data, which concludes one iteration with imputed values based on the regression predictions.
This iteration is repeated several times to update the imputation values each time and the last imputed values are kept as the final imputation results.
Optimally, the parameters of the regression have converged until the final iteration in the sense that the distribution of the regression parameters becomes stable to avoid dependence on the order of the imputation. However, convergence can only be guaranteed in several special case, such as under the multivariate normal model.
Thus, generally, the optimal number of iterations needs to be identified for the data set. Depending upon the amount of missing information in the data, many research papers use about 10 iterations, while others suggest that as many as 40 imputed data sets can improve the results (see [29]).
Advantages of MICE include that it is very flexible with respect to its data as it can handle variables of varying data types as well as complexities such as bounds. A major disadvantage however is that it does not have the same theoretical justification as other imputation methods, since the main justification rests on simulation studies.

The other major MI approach not discussed much further uses **Markov Chain Monte Carlo (MCMC)**. MCMC is a collection of methods for generating pseudo-random draws from probability distributions via Markov chains whose stationary distribution is a distribution of interest.
A Markov chain is a memoryless discrete-time stochastic process, more explicitly, it is a sequence of random variables in which a future event depends only upon the present state, not on any events that happened in the past. A stationary distribution of a Markov chain is a probability distribution that is invariant in time.
The goal of MCMC is to generate values of a multidimensional random variable Z with density $f(Z)$. Instead of drawing from f directly, which might be difficult to do, a sequence $\{Z_1, \ldots, Z_t, \ldots\}$ is generated with stationary distribution f such that each element depends on the preceding one. For a time t sufficiently large, $Z(t)$ is approximately a random draw from f. Multiple imputation with MCMC simulates approximately independent random draws of the missing values.

A data augmentation (DA) approach is the primarily used Markov Chain Monte Carlo method for handling missing data (see [31]).
Given some initial parameter value θ^0 and a data matrix $Y = \{Y_{obs}, Y_{mis}\}$ with missing data Y_{mis} and observed data Y_{obs}, it is an iterative method that creates a Markov chain by iterating a two-step approach to generate a stochastic sequence $\{(Y_{mis}, \theta^1), (Y_{mis}, \theta^2), \dots\}$ which converges in distribution to the joint posterior distribution $p(Y_{mis}, \theta|Y_{obs})$.
In the imputation step, DA simulates the missing values Y_{mis}^{t+1} based on the current parameter estimates θ^t by drawing from the conditional distribution $p(Y_{mis}|Y_{obs}, \theta^t)$.
The posterior step of DA involves the simulation of the parameters θ^{t+1} given the current completed data by drawing from the posterior distribution $p(\theta|Y_{obs}, Y_{mis}^{t+1})$.
For a sufficiently large t, θ^t can be regarded as an approximate draw from $p(\theta|Y_{obs})$ and Y_{mis}^t as an approximate draw from $p(Y_{mis}|Y_{obs})$.
Since of the information from one step of DA is contained in the previous step, the parameter estimates and imputed values from two subsequent steps are very similar and not useful as random draws. Therefore, several steps of DA are performed until a random draw is selected. The number of iterations between two draws has to be determined such that they are similar to two random draws.
If certain necessary assumptions are satisfied, DA converges, however the rate of convergence depends on the fractions of missing information for one or more components of θ and may be very slow for high fractions, see section 3.5 in [32].

As a general final remark about this paragraph, multiple imputation methods are not to be used as "black box algorithms" to achieve good imputation results. During the imputation process, several judgments have to be made, especially by setting up an appropriate imputation model, and the defaults should be appropriately adjusted to prevent misleading results. Sometimes, simpler methods may actually be sufficient and the more complex methods should not be used to simplify the imputation process.

2.3 Maximum Likelihood Procedures

This section will shortly introduce another popular approach to handling missing data based on **maximum likelihood (ML)**, see for example part II in [8] for an in-depth discussion of likelihood-based approaches to the analysis of data with missing values. Maximum likelihood estimation is a fundamental statistical method for estimating parameters by maximizing a likelihood function.

In a more simple statistical context, for a collection of independent and identically distributed (iid) random variables $X_1, \ldots, X_n$ with density function $f(x;\theta)$, where θ is in the parameter space $\Theta \subset \mathbb{R}^d$, the **likelihood** function of θ on the basis of $X_1, \ldots, X_n$ is the joint density of the random variables, defined as

$$L_n(\theta) = \prod_{i=1}^{n} f(X_i;\theta).$$

$\hat{\theta}$ is a **maximum likelihood estimate (MLE)** of θ, if

$$L_n(\theta) \leq L_n(\hat{\theta}), \qquad \forall\ \theta \in \Theta.$$

If the maximum of the likelihood function is unique, the MLE is given as

$$\hat{\theta} = \underset{\theta\in\Theta}{\operatorname{argmax}}\, L_n(\theta).$$

Determining $\hat{\theta}$ would involve calculating the derivative of a product of n functions, so that typically the **log-likelihood** function

$$l_n(\theta) = \log L_n(\theta) = \sum_{i=1}^{n} \log(f(X_i;\theta))$$

is maximized instead, since likelihood and log-likelihood have the same extremal points.

Missing data analysis using maximum likelihood, also sometimes called "full information maximum likelihood", is a method that does not impute any data, but rather analyzes the full, incomplete data set to compute maximum likelihood estimates. ML does the parameter estimation by determining the value that maximizes the likelihood function based on the available data, which is the value of the parameter that is most likely to

have resulted in the observed data. This means that ML methods handle missing data and parameter estimation in one step. Hence, the complete-case algorithms typically used for imputed data sets cannot be applied. Therefore, implemented ML procedures are only available in a very limited capacity.
Monotone incomplete data is a pattern of missing data where the variables can be arranged such that for a variable with missing data all variables afterwards have missing data at least for the same observations.
In a data set with a general monotone pattern of missing data, the likelihood function may be factorized, such that it is computed for variables with complete data and those with missing data separately.
The appropriate factorization for this pattern, with Y_i being more observed than Y_j for observations $i < j$, is given as

$$\prod_{i=1}^{n} f(y_{i1}, \ldots, y_{i,J}|\theta) = \prod_{i=1}^{n} f(y_{i1}|\theta_1) \prod_{i=1}^{r_2} f(y_{i2}|y_{i1}, \theta_2) \prod_{i=1}^{r_J} f(y_{iJ}|y_{i1}, \ldots, y_{i,J-1}, \theta_J),$$

where for $j = 1, ..., J$, $f(y_{ij}|y_{i1}, \ldots, y_{i,j-1}, \theta_j)$ is the conditional distribution of y_{ij} given $y_{i1}, \ldots, y_{i,j-1}$, indexed by the parameter θ_j (see section 7.4.2 in [8]).
However, incomplete data in practice usually does not have a pattern that allows ML estimates to be calculated by factorizing the likelihood. Hence, iterative methods are required to compute ML estimates.

A popular likelihood based iterative technique is an **expectation-maximization algorithm (EM)** which was introduced in [33]. Nowadays, there are many types of extensions of EM available.
EM produces maximum likelihood estimates by exploiting the interdependence of the missing data and the parameters (see [32]). The missing data contains information relevant to estimating the parameters which are helpful for finding likely values of the missing data. This suggests that parameters may be estimated in the presence of observed values based on a two-step approach.
EM is a parametric method that solves the incomplete data problem by alternating between such two steps, the expectation and the maximization step.
After replacing the missing data with a simple imputation by plausible estimated values and estimating parameters, the expectation and maximization steps are iterated.
During the expectation step, the missing values are estimated, given the observed data and the current parameter estimates.

More specifically, for the current estimate θ^t of the parameter θ, the expectation step finds the expected complete-data loglikelihood if θ were θ^t defined as:

$$Q(\theta|\theta^t) = \mathbb{E}(l_n(\theta|y)f(Y_{mis}|Y_{obs}, \theta = \theta^t)),$$

for the missing data encoded in Y_{mis} and the observed data in Y_{obs}.
Thus, the conditional expectation of the missing data given the observed data and current estimated parameters is computed and these expectations are substituted for the missing data. In the maximization step, parameters are estimated again, where a maximum likelihood estimation of the parameters is performed for the observed and completed data to check whether the estimate is the most likely one.
More formally, θ^{t+1} is determined by maximizing the expected complete-data loglikelihood:

$$\theta^{t+1} = \underset{\theta}{\operatorname{argmax}}\, Q(\theta|\theta^t).$$

If the estimate is not the most likely value, the two steps are re-iterated until the estimates converge, such that the estimates do not change any longer between two consecutive iteration steps.
Important advantages of EM include that the method preserves the relationship with other variables and that convergence may be observed (see sections 8.3 - 8.4 in [8]).
A major drawback is the slow convergence of EM. The rate of convergence is linear with rate proportional to the fraction of information that is observed. Additionally, the maximization step may not have a closed form solution which complicates the computation of the parameter estimations.
EM bears a strong resemblance to DA in the previous section. Data augmentation can be seen as a stochastic version of EM with the imputation step corresponding to the expectation step and the posterior step corresponding to the maximization step respectively.

This concludes the present chapter on statistical methods for incomplete data. In the next chapter, several numerical methods for handling missing values are introduced.

3. Matrix Completion by Dimensionality Reduction

Nowadays, many practical applications use high-dimensional data. However, the curse of dimensionality makes working with those high-dimensional data sets quite difficult. In this context, the term "curse of dimensionality" describes that with an increasing dimension of the data, executing an algorithm has an exponentially increasing complexity. For that reason, a class of unsupervised machine learning algorithms, named dimensionality reduction methods, take advantage of the fact that data sets may have an underlying structure. Hence, the goal is to find this structure in the high-dimensional data set, to extract the relevant features and disregard redundant information to be able to work in fewer dimensions. In this chapter several methods for handling missing data in relation to dimensionality reduction are presented. The list of dimensionality reduction algorithms presented is not comprehensive and the selection is based on the applications to incomplete data explained in the subsequent sections.

3.1 Dimensionality Reduction Methods

Many practical problems concern high-dimensional data that typically lies close to a much lower dimensional manifold due to its underlying structure. Only a small number of degrees of freedom is expected to be required to describe the data to a reasonable accuracy. The aim is thus to find computationally efficient dimensionality reduction methods that project data points onto a lower dimensional space.
In many applications, the goal is to infer a quantitative model M of a model class $\mathcal{M}$ for a given set of sample points $\mathcal{X} = \{x_1, x_2, ..., x_N\}, x_j \subset \mathbb{R}^D$. The purpose of these models very often includes the simplification of the representation of the data set to help to reveal underlying structures and predict future samples.

3.1.1 Principal Component Analysis

The standard well-known linear method for dimensionality reduction is **Principal Component Analysis (PCA)**. By identifying patterns in data, the number of dimensions can be reduced with little loss of information. This is based on the premise that if the given data lies close to a linear subspace, then it can be fitted to approximate each data point mapped to the subspace, such that the distance between the data points and their projection onto the subspace is minimal.

The following exposition on PCA is largely based on [34]. Geometrically, classical PCA can be seen as the problem of fitting a low-dimensional (affine) subspace S of dimension $d \ll D$ to a set of N data points $\mathcal{X} = \{x_j \in \mathbb{R}^D\}_{j=1}^N$ in a high-dimensional space $\mathbb{R}^D$. Each data point x_j can be represented as

$$x_j = \mu + Uy_j, \ j = 1, 2, ..., N,$$

where $\mu \in S$ is a point in the subspace, $U \in \mathbb{R}^{D \times d}$ is a matrix that forms a basis for S and $y_j \in \mathbb{R}^d$ is the vector of new coordinates of x_j in the subspace.
To get a unique solution for the problem, common constraints are that the average of the y_j be zero, $\frac{1}{N}\sum_{j=1}^N y_j = 0$, and that the columns of U be orthonormal, $U^\top U = I_d$. In practice, the given data points are very likely imperfect and can therefore for example be represented as

$$x_j = \mu + Uy_j + \epsilon_j, \ j = 1, 2, ..., N$$

with some additive noise ϵ_j. Hence, the goal is to minimize the sum of the squared errors

$$\min_{\mu, U, \{y_j\}} \sum_{j=1}^N \|x_j - \mu - Uy_j\|^2, \text{ s.t. } U^\top U = I_d \text{ and } \sum_{j=1}^N y_j = 0.$$

Calculating a solution for this optimization problem yields the results

$$\hat{\mu} = \hat{\mu}_N = \frac{1}{N}\sum_{j=1}^N x_j \ \text{ and } \ \hat{y}_j = U^\top (x_j - \hat{\mu}).$$

Rewriting the optimization problem with the optimal values as

$$\min_U \sum_{j=1}^N \|(x_j - \hat{\mu}_N) - UU^\top (x_j - \hat{\mu}_N)\|^2 \text{ s.t. } U^\top U = I_d.$$

shows that it can be assumed that all data points have zero mean. Otherwise, one can simply subtract the mean from each point before computing U. For the matrix of data points $X = [x_1, x_2, ..., x_N] \in \mathbb{R}^{D \times N}$, the singular value decomposition (SVD) can be written as $X = U_X \Sigma_X V_X^\top$ and $XX^\top = U_X \Sigma_X^2 V_X^\top$ is the eigenvalue decomposition of $XX^\top$.
A solution of the optimization problem for U is given by the first d columns of U_X which are are the top d eigenvectors of $XX^\top$ for an ordered Σ_X. The columns of $\Sigma V^\top = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_N] \in \mathbb{R}^{d \times N}$ are the **principal components** $\mathcal{Y} = \{y_j \in \mathbb{R}^d\}_{j=1}^N$.

From a statistical point of view, PCA estimates the principal components of a multivariate random variable.
For a random variable $x \in \mathbb{R}^D$ and an integer $d < D$, the first d principal components $y \in \mathbb{R}^d$ of x are defined as the d uncorrelated affine components,

$$y_i = u_i^\top x + a_i \in \mathbb{R},\ u_i \in \mathbb{R}^D, i = 1, 2, ..., d,$$

where $a_i = -u_i^\top \mu$, such that the variance of y_i is maximized subject to $u_i^\top u_i = 1$ and $\mathrm{Var}(y_1) \geq \mathrm{Var}(y_2) \geq ... \geq Var(y_d) > 0$. The covariance matrix is given by $\Sigma_x = \mathbb{E}[(x-\mu)(x-\mu)^\top] \in \mathbb{R}^{D \times D}$, where $\mu = \mathbb{E}(x) \in \mathbb{R}^D$ is the mean. For $\mathrm{rank}(\Sigma_x) \geq d$, the unit norm vectors $\{u_i\}_{i=1}^d$ are the d eigenvectors of Σ_x associated with its d largest eigenvalues $\{\lambda_i\}_{i=1}^d$, where $\lambda_i = \mathrm{Var}(y_i)$.
Since Σ_x and μ may not be known, the sample mean $\hat{\mu}_N = \frac{1}{N} \sum_{j=1}^N x_j$ and the maximum likelihood estimate $\hat{\Sigma}_N = \frac{1}{N} \sum_{j=1}^N (x_j - \hat{\mu}_N)(x_j - \hat{\mu}_N)^\top$ can be used instead as they are a reasonable approximations.

So far, the number d of principal components was assumed as known. In general, however, d has to be estimated. Noise-free data points would lie exactly in the subspace of dimension d, so that d can be estimated as $d = \mathrm{rank}(X)$. But, the data matrix X is usually of full rank when there is noise. Therefore, d has to be estimated in some other fashion, since $d = \mathrm{rank}(X)$ would not achieve dimensionality reduction.
It is possible to find a solution to PCA for all $d = 1, 2, ..., D$, so that the best estimate $\hat{d}$ can be chosen by looking at the spectrum of solutions for different values of d, while keeping the model as simple as possible. For a good balance between the complexity of the model and the fidelity of the data to the model, a criterion has to be chosen. A simple model selection criterion is choosing a threshold such that the fraction of information to be discarded is less than the given threshold.

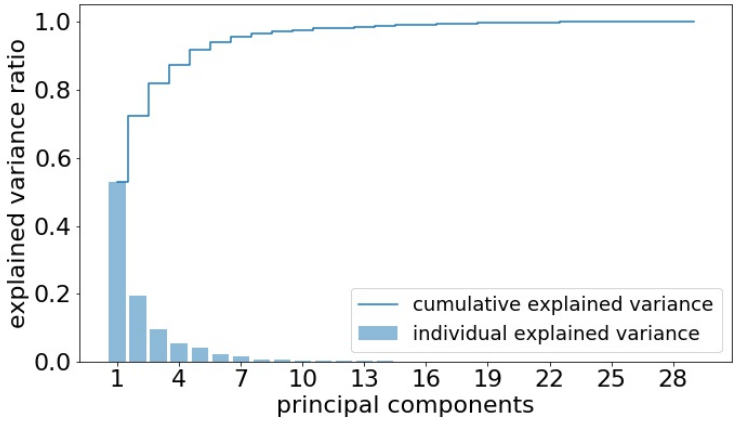


Figure 3.1: The explained variance of the 30 shares included in the DAX over the course of the year 2018 decreases with the eigenvalues, with the first eigenvalue already explaining more than 50% of the variance and 95% of the variance being explained by only 7 eigenvalues.

In many applications, the decrease in the absolute value of the ordered eigenvalues is significant, so that with a small number of eigenvalues a large amount of variance can already be explained. One example for this phenomenon that gives the motivation for dimensionality reduction is stock data, which can be seen in figure 3.1.

3.1.2 Kernel Principal Component Analysis

In the previous section, Principal Component Analysis was used to fit a low-dimensional linear or affine subspace to data. However, this approach may fail to capture nonlinear structures. In that case, Cover's theorem may be useful, which can be paraphrased as "A complex pattern-classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated" (page 231 in [35]).
That is to say with a high probability the data can be projected via a nonlinear transformation into a higher dimensional space where it is then linearly separable. The following explanation is based on [36] and Chapter 4 in [34].

A nonlinear extension of PCA, called **Nonlinear Principal Component Analysis (NLPCA)**, is based on this principle. NLPCA embeds the data into a high-dimensional space via a nonlinear mapping and then applies PCA to the embedded data.
More precisely, there exists a nonlinear embedding $\phi : \mathbb{R}^D \to \mathbb{R}^M$, where $\mathbb{R}^M$ is a higher-dimensional space such that the structure of the embedded data is approximately linear in it.
For a data point $x \in \mathbb{R}^D$, $\phi(x) \in \mathbb{R}^M$ is called the *feature* and the space $\mathbb{R}^M$ is called the *feature space*. For the sample mean of the feature space $\hat{\phi} = \frac{1}{N}\sum_{j=1}^{N}\phi(x_j)$, the centered embedded data matrix is given by

$$\Phi = [\phi(x_1) - \hat{\phi}, \ldots, \phi(x_N) - \hat{\phi}] \in \mathbb{R}^{M\times N}.$$

The principal components are then given by the eigenvectors of the embedded sample covariance matrix

$$\Sigma_{\phi(x)} = \frac{1}{N}\sum_{j=1}^{N}(\phi(x_j) - \hat{\phi})(\phi(x_j) - \hat{\phi})^\top = \frac{1}{N}\Phi\Phi^\top \in \mathbb{R}^{M\times M}.$$

Let $u_i \in \mathbb{R}^M,\ i = 1, \ldots, M$ be the M eigenvectors of $\Sigma_{\phi(x)}$,

$$\Sigma_{\phi(x)}u_i = \lambda_i u_i,\ i = 1, \ldots, M,$$

then the d nonlinear principal components of each x are given by

$$y_i = u_i^\top(\phi(x) - \hat{\phi}) \in \mathbb{R},\ i = 1, ..., d.$$

By a similar argumentation as in the case of PCA, the eigenvalues of $\Phi^\top\Phi \in \mathbb{R}^{N\times N}$ and $\Phi\Phi^\top \in \mathbb{R}^{M\times M}$ coincide.
If the dimension of $\mathbb{R}^M$ is too high for a computation of the nonlinear principal components from the eigenvectors of the embedded covariance matrix, but $N \ll M$, it may hence be possible to use the lower-dimensional eigenvalue decomposition of $\Phi^\top\Phi \in \mathbb{R}^{N\times N}$.
For every eigenvector $u \in \mathbb{R}^M$ of $\Phi\Phi^\top$ it holds that $\Phi\Phi^\top u = \lambda u$. This can be rewritten as $u = \Phi(\lambda^{-1}\Phi^\top u)$. Defining $w = \lambda^{-1}\Phi^\top u \in \mathbb{R}^N$, it holds that $\|w\|^2 = \lambda^{-1}$ and $\Phi^\top\Phi w = \lambda w$, so w is an eigenvector of $\Phi\Phi^\top$ with eigenvalue $\lambda \neq 0$.
For the eigenvector u of $\Phi^\top\Phi$ and the eigenvector w of $\Phi\Phi^\top$, it holds that $u = \Phi w$ and the d nonlinear principal components can also be calculated as

$$y_i = w_i^\top\Phi^\top(\phi(x) - \hat{\phi}) \in \mathbb{R},\ i = 1, ..., d.$$

A special case of NLPCA called **Kernel Principal Component Analysis (KPCA)** is the extension of PCA with techniques of kernel methods. It is used to compute the nonlinear principal components from the eigenvectors of the kernel matrix directly from the given data by applying the so-called kernel trick to compute the low-dimensional embedding without explicitly computing the embedded data.
To elaborate on this, given the embedding function $\phi : \mathbb{R}^D \to \mathbb{R}^M$, the **kernel function** $\kappa : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ of two vectors $x, y \in \mathbb{R}^D$ is defined as the inner product of their features

$$\kappa(x, y) = \phi(x)^\top\phi(y) \in \mathbb{R}.$$

The kernel function κ is symmetric,

$$\kappa(x, y) = \kappa(y, x)\ \forall x, y \in \mathbb{R}^D,$$

and positive semi-definite,

$$\forall f \in L^2(\mathbb{R}^D) : \int_{\mathbb{R}^D}\int_{\mathbb{R}^D} \kappa(x, y)f(x)f(y)dxdy \geq 0,$$

where $L^2(\mathbb{R}^D) = \{f : \mathbb{R}^D \to \mathbb{R} \text{ s.t. } \int f(x)^2dx < \infty\}$ is the space of all square integrable functions.
The *centered kernel* can be defined as

$$\tilde{\kappa}(x, y) = (\phi(x) - \hat{\phi})^\top(\phi(y) - \hat{\phi}) \in \mathbb{R}$$

and computed from κ via

$$\tilde{\kappa}(x,y) = \kappa(x,y) - \frac{1}{N}\sum_{j=1}^{N}\kappa(x,y_j) - \frac{1}{N}\sum_{i=1}^{N}\kappa(x_i,y) + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\kappa(x_i,y_j).$$

Define a *kernel matrix* $\mathcal{K} = [\kappa_{ij}] \in \mathbb{R}^{N\times N}$ as $\kappa_{ij} = \kappa(x_i, y_j)$ and the *centered kernel matrix* $\tilde{\mathcal{K}} = \Phi^\top\Phi$ as

$$\tilde{\mathcal{K}} = \mathcal{K} - \frac{1}{N}\mathcal{K}11^\top - \frac{1}{N}11^\top\mathcal{K} + \frac{1^\top\mathcal{K}1}{N^2}11^\top \quad = (I - \frac{1}{N}11^\top)\mathcal{K}(I - \frac{1}{N}11^\top) = \tilde{\mathcal{K}} = J\mathcal{K}J$$

with the centering matrix $J = I - \frac{1}{N}11^\top$.
For $\tilde{\kappa}_x = [\tilde{\kappa}(x_1,x),\ldots,\tilde{\kappa}(x_N,x)]^\top = \Phi^\top(\phi(x)-\hat{\phi}) \in \mathbb{R}^N$ and $\kappa_x = [\kappa(x_1,x),\ldots,\kappa(x_N,x)]^\top \in \mathbb{R}^N$ it holds that

$$\tilde{\kappa}_x = \kappa_x - \frac{1}{N}\mathcal{K}1 - \frac{1}{N}11^\top\kappa_x + \frac{1^\top\mathcal{K}1}{N^2}1.$$

Therefore, the nonlinear principal components can be computed as

$$y_i = w_i^\top\Phi^\top(\phi(x) - \hat{\phi}) = w_i^\top\tilde{\kappa}_x,\ i = 1,...,d,$$

where w_i is the normalized eigenvector of $\tilde{\mathcal{K}}$ associated with its i-th largest eigenvalue λ_i, such that $\|w_i\| = \lambda_i^{-1}$. Since the eigenvalue decomposition of the matrix $\tilde{\mathcal{K}} = [\tilde{\kappa}_{x_1},...,\tilde{\kappa}_{x_N}]$ is given as $\tilde{\mathcal{K}} = V_{\tilde{\mathcal{K}}}\Lambda_{\tilde{\mathcal{K}}}V_{\tilde{\mathcal{K}}}^\top$, the top d eigenvectors V_d and eigenvalues Λ_d yield $V_d\Lambda_d^{-1/2} = [w_1,...,w_N]$. As a consequence, the low-dimensional coordinates can be obtained from the top d eigenvectors and eigenvalues of the centered kernel matrix as

$$Y = \Lambda_d^{-1/2}V_d^\top\tilde{\mathcal{K}} = \Lambda_d^{-1/2}V_d^\top V_{\mathcal{K}}\Lambda_{\mathcal{K}}V_{\mathcal{K}}^\top = \Lambda_d^{1/2}V_d^\top.$$

Therefore, the nonlinear components are computed directly from the kernel function $\kappa(x,y) = \phi(x)^\top\phi(y)$ without computing the embedded data $\phi(x)$.

Several different possible kernels can be used, popular choices, that where first derived from support vector machines, include

- polynomial kernel:
$$\kappa(x,y) = (\gamma x^\top y + c)^n,$$
- rbf kernel (radial basis function):
$$\kappa(x,y) = \exp(-\gamma\|x-y\|^2),$$

with the special case of a Gaussian kernel

$$\kappa(x, y) = \exp(-\frac{\|x - y\|^2}{\sigma^2}).$$

- cosine kernel:

$$\kappa(x, y) = \frac{x^\top y}{\|x\|\|y\|},$$

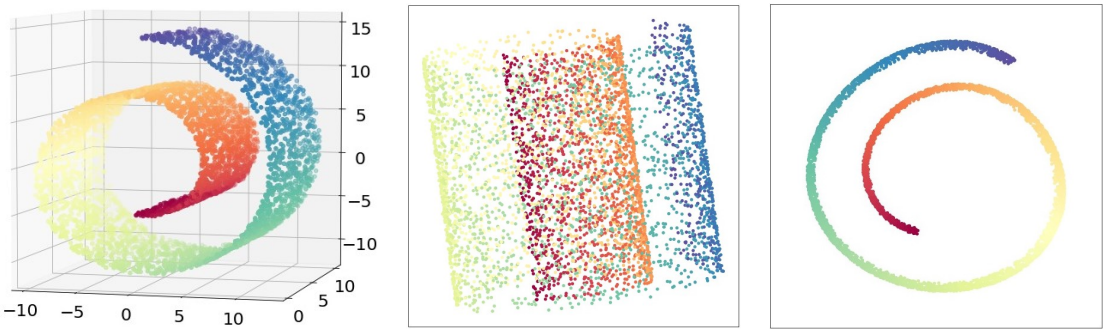
- sigmoid kernel:

$$\kappa(x, y) = \tanh(\gamma x^\top y + c),$$

which is important for support vector machines.

KPCA with the linear kernel $\kappa(x, y) = x^\top y$ yields standard PCA.
The question of choosing the best kernel for a given problem is sadly yet unsolved.

A major advantage of KPCA over other methods of NLPCA is that no nonlinear optimization is involved, so that all minima are global minima.
Unfortunately, KCPA also has its problems. A major downside is that standard PCA is a basis transformation and allows the reconstruction of the original pattern, whereas in KPCA this may no longer be possible. A specific vector might not have a pre-image so that it has to be approximated.



(a) Original (b) PCA (c) DM

Figure 3.2: The original swiss roll data set in 3D (a) with an unsuitable embedding in 2D given by PCA (b) and a 2D embedding with DM (c).

3.1.3 Diffusion Maps

As addressed in the previous section, PCA is a linear model which might not yield appropriate results for nonlinear examples. In that case, a nonlinear method that is able to embed data with an underlying nonlinear structure is preferable.
One useful manifold learning method for dimensionality reduction is **Diffusion Maps (DM)**, where high dimensional data is embedded into a lower dimensional manifold. For a canonical nonlinear example where DM is applicable see figure 3.2, which depicts the so-called swiss roll data set and its two-dimensional embeddings using PCA and DM.
The construction and notation in the following paragraph leading to a dimensionality reduction method with the diffusion map algorithm is mainly based on [37].

Let $(X, \mathcal{A}, \mu)$ be a measure space with data set X and probability measure μ.
The **diffusion kernel** $k : X \times X \longrightarrow \mathbb{R}$ satisfies for all $x, y \in X$:

- k is symmetric: $k(x, y) = k(y, x)$,
- k is positivity preserving: $k(x, y) \geq 0$.

A kernel that is commonly used is the **Gaussian kernel**

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{\sigma}\right)$$

with $\sigma > 0$ a scaling parameter of the kernel.

For the construction of the **diffusion process**, the kernel is normalized as follows:
For all $x \in X$, let

$$v(x) = \int_X k(x,y)d\mu(y)$$

and set

$$a(x,y) = \frac{k(x,y)}{v(x)}.$$

The new kernel $a(x,y)$ is well-defined and non-negative, but not symmetric anymore. However, the conservation property

$$\int_X a(x,y)d\mu(y) = 1$$

holds.
The integral operator $\tilde{A}$ defined on $L^2(X)$ with the kernel

$$\tilde{a}(x,y) = a(x,y)\sqrt{\frac{v(x)}{v(y)}}$$

is symmetric and yields the spectral decomposition

$$\tilde{a}(x,y) = \sum_{i\geq 0} \lambda_i^2 \phi_i(x)\phi_i(y)$$

with eigenvalues $\lambda_0 = 1 \geq \lambda_1 \geq \lambda_2 \geq ... \geq 0$ and eigenvectors ϕ_i for $i = 1, 2, ...$.
Let $m \in \mathbb{N}$ and $\tilde{a}^{(m)}(x,y)$ be the kernel of $\tilde{A}^m$, then

$$\tilde{a}^{(m)}(x,y) = \sum_{i\geq 0} \lambda_i^{2m} \phi_i(x)\phi_i(y).$$

The family of **diffusion distances** $\{D_m\}$ for $m \in \mathbb{N}$ is defined by

$$\begin{aligned} D_m^2(x,y) &= \tilde{a}^{(m)}(x,x) + \tilde{a}^{(m)}(y,y) - 2\tilde{a}^{(m)}(x,y) \\ &= \sum_{i\geq 0} \lambda_i^{2m}(\phi_i(x) - \phi_i(y))^2. \end{aligned}$$

To achieve dimensionality reduction, the following key observation is paramount:
Since $0 \leq \lambda_i \leq \lambda_0 = 1,\ i = 1, 2, \ldots$ holds and the number of significant eigenvalues decreases with increasing m, the kernel $\tilde{a}^{(m)}(x,y)$ and also $D_m(x,y)$ can be calculated with only a few terms up to an accuracy $\delta > 0$.

For $s(\delta, m) = \max\{l \in \mathbb{N} : |\lambda_l|^m > \delta|\lambda_1|^m\}$, the approximate diffusion distance with accuracy δ can be calculated with a finite number of terms:

$$D_m^2(x, y) = \sum_{i=0}^{s(\delta,m)} \lambda_i^{2m}(\phi_i(x) - \phi_i(y))^2.$$

Finally, the family of **diffusion maps** $\{\Phi_m\}$ can be introduced, such that dimensionality reduction leads to $\Phi_m : X \longrightarrow \mathbb{R}^{s(\delta,m)}$ being given by

$$\Phi_m(x) = \begin{pmatrix} \lambda_0^m \phi_0(x) \\ \lambda_1^m \phi_1(x) \\ \vdots \\ \lambda_{s(\delta,m)}^m \phi_{s(\delta,m)}(x) \end{pmatrix}$$

for $m \in \mathbb{N}$ and $\lambda_0, ..., \lambda_{s(\delta,m)}$ the $s(\delta, m)$ largest eigenvalues and $\phi_0, ..., \phi_{s(\delta,m)}$ the corresponding eigenvectors.
Φ_m embeds the data in a Euclidean space of dimension $s(\delta, m)$, such that the Euclidean distance equals the diffusion distance up to δ.

3.2 Incomplete Dimensionality Reduction - A Missing Data Approach with Dimensionality Reduction

After looking at imputation methods in the previous chapter, which is the statistical approach to missing data, the following section addresses a numerical view on handling incomplete data based on the remarks concerning dimensionality reduction in the previous section. Starting with the linear dimensionality reduction method PCA, the methods are adapted to handle missing data.
To recap, PCA calculates the eigenvectors and eigenvalues of the covariance matrix of the centered data, orders the eigenvectors by the value of the corresponding eigenvalue in decreasing order and may ignore eigenvectors of lesser significance to achieve dimensionality reduction by projecting the data onto its principal components via multiplication of the data with the feature vector of the eigenvectors.
To reconstruct the original data, the transformed data is multiplied by the inverse of the feature vector which is the transposed of the feature vector due to the orthogonality requirement of the eigenvectors and added to the mean. If the data was reduced, then obviously the reconstructed data is not entirely accurate.

Now, unlike in the previous case, the data set is not complete, therefore a full PCA can no longer be performed. In that case, consider a complete subset of columns or rows of the data set which can be used for a standard PCA.
To be more precise, given a data set represented as a matrix of dimension $M \times L$, then assume that there either exists a subset of dimension $m \times L$ or $M \times l$ which is complete, where m or l respectively are not too small. Now, given the complete data matrix of dimension $D \times N$, where $D \in \{m, M\}$ and $N \in \{l, L\}$, the eigenvalue decomposition of the centered covariance matrix of the data set is calculated for the PCA.
More explicitly, for the centered matrix $A = X - \mu \in \mathbb{R}^{D \times N}$, with data matrix $X \in \mathbb{R}^{D \times N}$ and mean $\mu \in \mathbb{R}^D$, the covariance is given by $C = A^\top A \in \mathbb{R}^{N \times N}$. The non-zero eigenvalues of the matrices $C = A^\top A$ and $\tilde{C} = AA^\top$ coincide, which can be easily seen by the following explanation.
Let v_i be the eigenvectors of $A^\top A$ with corresponding eigenvalues μ_i, $A^\top A v_i = \mu_i v_i$, and let u_i be the eigenvectors of $AA^\top$ with corresponding eigenvalues λ_i, $AA^\top u_i = \lambda_i u_i$.
Then

$$A^\top A v_i = \mu_i v_i.$$

Multiplying with A yields

$$AA^\top Av_i = \mu_i Av_i.$$

Setting $\tilde{C} = AA^\top$ results in

$$\tilde{C}u_i = \mu_i u_i,$$

where $u_i = Av_i$ is the relation of the eigenvectors of C and $\tilde{C}$.
Therefore, it can be computationally efficient to choose between using C if $N < D$ and $\tilde{C}$, if $D < N$, for PCA. Although, one has to bear in mind that if the second case holds, the eigenvectors that are computed by the PCA, have to be multiplied with A as seen above to obtain the correct solution. Assume for now, that the data set is not too large and the first case holds, hence PCA is done for C.
The eigenvalues computed with the PCA can be sorted by their size in decreasing order, whose corresponding eigenvectors are then normalized and saved as columns in a matrix. However, only a small ratio K of eigenvalues is needed to explain most of the variance in the data, so that eigenvectors whose corresponding eigenvalues explain only very small amounts of data are discarded. This threshold for the dimensionality reduction step is chosen in advance. Each normalized vector can then be represented as a linear combination of the K eigenvectors up to some small error.
Given the data set without having missing data, one of the vectors to be reconstructed $v \in \mathbb{R}^D$ is chosen. This vector now includes artificially produced missing values.
The locations of missing entries can be encoded in a vector w.
The weights for the linear combination of the eigenvectors that the incomplete vector v can be represented with, see for example [38], are given by

$$\omega_i = u_i^\top (v - \mu),\ i = 1, .., K.$$

The vector can then be reconstructed with

$$\hat{v} = \mu + \sum_{i=1}^{K} \omega_i u_i.$$

Only the unknown values are replaced by the calculated values, so that the reconstructed vector is determined by

$$v = w \odot v + (1 - w) \odot \hat{v}.$$

This is repeated until the weights have converged, that is until they do not change substantially any longer from one iteration step to the next.

Thus, this method is linked to the multiple imputation approach as well as expectation maximization methods described in the previous chapter.
There is an iterative two step approach to finding the missing values. The weights are re-calculated at each iteration for the completed vector and then the vector is completed with the linear combination of eigenvectors using those weights.

Similar to above, KPCA or Diffusion Maps can be utilized instead of PCA with the data matrix for analogue results, such that the reconstruction of incomplete data with dimensionality reduction methods can be illustrated in the following steps:

- Use the chosen dimensionality reduction method for the complete sub-matrix X.
- Find the K largest eigenvalues with corresponding eigenvectors given by the dimensionality reduction method.
- Iteratively alternate between the following two steps until convergence:
  - Calculate the weights needed to represent the given entries of the incomplete vector v as a linear combination of the computed eigenvectors.
  - Complete the missing entries of the vector with the calculated weights while leaving the existing entries untouched.

3.3 Matrix Completion with Incomplete PCA

In this last part of the chapter, several different approaches to matrix completion with principal component analysis are introduced, based on section 3.1 in [34]. They are more complicated but also more widely applicable than the simple approach in the previous section.
For the completion method in the previous section, it had to be assumed that a complete sub-matrix of the incomplete data matrix exists. This may however not always hold true, especially if the values are missing at random. In that case, the method in the previous section cannot be applied and a different approach needs to be found. This is where the motivation of the following methods for PCA with missing data comes from.
First, a very simple method based on mean and covariance completion is described, then an approach based on iterative PCA similar to the one in the previous section is presented and in the subsequent paragraphs, methods for solving the matrix completion problem based on maximum likelihood estimation, convex optimization, and alternating minimization are featured.

Recall that the goal of PCA is to minimize the sum of the squared errors

$$\min_{\mu,U,\{y_j\}} \sum_{j=1}^{N} \|x_j - \mu - Uy_j\|^2, \text{ s.t. } U^\top U = I_d \text{ and } \sum_{j=1}^{N} y_j = 0,$$

where the point μ is given by the sample mean

$$\hat{\mu}_N = \frac{1}{N} \sum_{i=1}^{N} x_j,$$

the orthonormal basis U is given by the top d eigenvectors of the covariance matrix

$$\hat{\Sigma}_N = \frac{1}{N} \sum_{i=1}^{N} (x_j - \hat{\mu}_N)(x_j - \hat{\mu}_N)^\top,$$

and the matrix of low-dimensional coordinates $Y = [y_1, y_2, ..., y_N] \in \mathbb{R}^{d \times N}$ is given by

$$y_j = U^\top (x_j - \hat{\mu}_N), j = 1, ..., N.$$

Now, a data point x exists which is incomplete.

If the point x has M missing entries, it can be partitioned, without loss of generality, as $\begin{bmatrix} x_U \\ x_O \end{bmatrix}$, where $x_U \in \mathbb{R}^M$ denotes the unobserved entries and $x_O \in \mathbb{R}^{D-M}$ denotes the observed entries.
Hence, x is known only up to an M-dimensional affine subspace:

$$x \in L = \left\{ \begin{bmatrix} 0 \\ x_O \end{bmatrix} + \begin{bmatrix} I_M \\ 0 \end{bmatrix} x_U,\ x_U \in \mathbb{R}^M \right\}.$$

The incomplete samples can be arranged as the columns of the incomplete data matrix $X = [x_1, x_2, ..., x_N] \in \mathbb{R}^{D \times N}$ and a matrix $W \in \mathbb{R}^{D \times N}$ encodes the locations of missing entries,

$$w_{ij} = \begin{cases} 1 & \text{if } x_{ij} \text{ is known} \\ 0 & \text{if } x_{ij} \text{ is missing.} \end{cases}$$

Let $W \odot X$ be the Hadamard product of the two matrices, which is defined as the entry-wise product $(W \odot X)_{ij} = w_{ij} x_{ij}$.
Incomplete PCA now has to find the missing entries $(11^\top - W) \odot X$ additionally to μ, U and Y from the known entries $W \odot X$. Additionally, $\hat{\mu}_N$ and $\hat{\Sigma}_N$ can no longer be calculated directly when some entries of x_j are missing.

Mean and Covariance Completion

There are several possible approaches to solve the incomplete PCA problem.
The simplest approach computes a mean and covariance from the known entries only and uses those in the optimization problem associated with PCA, so that

$$\hat{\mu}_i = \frac{\sum_{j=1}^N w_{ij} x_{ij}}{\sum_{j=1}^N w_{ij}} \quad \text{and} \quad \hat{\sigma}_{ik} = \frac{\sum_{j=1}^N w_{ij} w_{kj} (x_{ij} - \hat{\mu}_i)(x_{kj} - \hat{\mu}_k)}{\sum_{j=1}^N w_{ij} w_{kj}}$$

with $i, k = 1, ..., D$. This approach is not recommended for various reasons, although it is a good initialization for subsequent methods.

Iterative PCA

Another simple method uses iterative PCA (IPCA), which is closely related to multiple imputation described in 2.2. IPCA can be briefly illustrated analogously to the method described in the previous section based on the iterative usage of a standard PCA.
After initializing the missing values in the data set, usually with the mean from the known entries as described above, the following iterative procedure is used.

In the first step, a principal component analysis is done on the data set which has been completed by a simple imputation method like mean imputation.
Then, the previously missing entries may be updated with new values, which are computed similar to the previous section as a linear combination of the first several eigenvectors from the calculated decomposition. Then, the PCA may once again be done on the completed data set, such that the two steps are iterated until convergence (see [39]). Iterative PCA is easily generalized to other dimensionality reduction methods such as KPCA as described previously.

PPCA and Expectation Maximization

Another approach that will be briefly discussed in this section is an EM algorithm for a probabilistic PCA (PPCA), see [34] for more information.
PPCA is motivated as follows for the complete case. Since PCA is not a proper generative model, it cannot be used to generate new samples of the random variable x, as the low-dimensional representation $\{y_j \in \mathbb{R}^d\}$ of the data points $\{x_j \in \mathbb{R}^D\}, d < D$ and the error $\{\epsilon_j\}$ are not treated as random variables.
For that reason, y and ϵ are assumed to be independent random variables with probability density functions $p(y)$ and $p(\epsilon)$, respectively and new samples of x are generated with

$$x = \mu + By + \epsilon,$$

where $\mu \in \mathbb{R}^D$ is a point on the affine subspace L and $B \in \mathbb{R}^{D\times d}$ is a basis for L. While B is a matrix of rank d, it is no longer assumed to be orthonormal.
Denote the mean and covariance of y by μ_y and Σ_y and assume that ϵ has zero mean and covariance Σ_ϵ, then the mean and covariance of the observations are given by

$$\mu_x = \mu + B\mu_y \quad \text{and} \quad \Sigma_x = B\Sigma_y B^\top + \Sigma_\epsilon.$$

The parameters of this model, μ, B, μ_y, Σ_y and Σ_ϵ, may now be estimated from μ_x and Σ_x by the maximum likelihood principle explained in 2.3.
If μ_x and Σ_x are known, assume for uniqueness of the parameters that $\mu_y = 0$, μ may be estimated in the same way as for PCA as $\hat{\mu} = \mu_x$. Additionally, assume that $\Sigma_y = I_d \in \mathbb{R}^{d\times d}$ and $\Sigma_\epsilon = \sigma^2 I_D$, for $\sigma > 0$, such that $\Sigma_x = BB^\top + \sigma^2 I_D$.
However, μ_x and Σ_x are usually unknown and cannot be used to estimate the parameters directly. In that case, the iid samples $\{x_j\}_{j=1}^N$ are used instead to estimate the model parameters μ, B and σ with the maximum likelihood principle.

Assuming that $y \sim \mathcal{N}(0, I)$ and $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, the data points are normally distributed, $x \sim \mathcal{N}(\mu_x, \Sigma_x)$ with mean μ_x and covariance Σ_x, as previously described.
The maximum likelihood estimates for μ_x and Σ_x are obtained from the derivatives of the log-likelihood of x as

$$\hat{\mu}_N = \frac{1}{N} \sum_{j=1}^{N} x_j$$

and

$$\hat{\Sigma}_N = \frac{1}{N} \sum_{j=1}^{N} (x_j - \hat{\mu}_N)(x_j - \hat{\mu}_N)^\top.$$

Using the ML estimates, the model parameters can be estimated just as above.
Now, assume that every incomplete data point x is drawn from a normal distribution $\mathcal{N}(\mu_x, \Sigma_x)$, where $\mu_x = \mu$ and $\Sigma_x = BB^\top + \sigma^2 I_D$ as before. The EM algorithm may then iteratively estimate the model parameters $\theta = (\mu, B, \sigma)$ from the incomplete samples.
While the EM approach is simple, it does not necessarily converge to a global optimum. Thus, the correct solution may not be found.

Convex Optimization

A further alternative for solving the incomplete PCA problem is using a convex relaxation based on the principle of compressed sensing, see for instance [40] for more details. The matrix completion is done by minimizing a convex objective function with a guaranteed globally optimal minimizer. More explicitly, for a matrix $X \in \mathbb{R}^{D \times N}$, assume that only a subset of entries is observed, defined as

$$\Omega = \{(i, j) \ : \ x_{ij} \text{ is observed } \}.$$

Let $\mathcal{P}_\Omega : \mathbb{R}^{D \times N} \to \mathbb{R}^{D \times N}$ be the orthogonal projector onto the span of all matrices vanishing outside of Ω so that

$$(\mathcal{P}_\Omega(X))_{ij} = \begin{cases} x_{ij}, & \text{if } (i, j) \in \Omega \\ 0, & \text{otherwise.} \end{cases}$$

The missing entries in X may now be completed by finding a complete low rank matrix $A \in \mathbb{R}^{D \times N}$ coinciding with X on Ω.
This results in the optimization problem

$$\min_A \operatorname{rank}(A) \quad \text{s.t.} \quad \mathcal{P}_\Omega(A) = \mathcal{P}_\Omega(X).$$

Since this rank-minimization problem is NP-hard, a convex relaxation is considered

$$\min_A \|A\|_* \quad \text{s.t.} \quad \mathcal{P}_\Omega(A) = \mathcal{P}_\Omega(X),$$

where $\|A\|_* = \sum \sigma_i(A)$ is the nuclear norm of the matrix A which is defined as the sum of all singular values of A.
Under certain conditions on the measurement operator $\mathcal{P}$, the solution to the convex optimisation problem coincides with that of the rank minimization problem, which is a well defined problem with a unique solution (see [40]).
For a simple solution, a penalty term is introduced on A, such that the solution to the given optimization problem is found by solving

$$\min_A \tau\|A\|_* + \frac{1}{2}\|A\|_F^2 \quad \text{s.t.} \quad \mathcal{P}_\Omega(A) = \mathcal{P}_\Omega(X),$$

where $\|A\|_F = \sqrt{\sum_i \sum_j |a_{ij}|^2}$ is the Frobenius norm of a matrix A.
The method of Lagrange multipliers is used to find the optimal solution of this penalized convex optimization problem. For the Lagrangian function

$$\mathcal{L}(A, Z) = \tau\|A\|_* + \frac{1}{2}\|A\|_F^2 + \langle Z, \mathcal{P}_\Omega(A) - \mathcal{P}_\Omega(X)\rangle,$$

where $Z \in \mathbb{R}^{D\times N}$ is a matrix of Lagrange multipliers.
The optimal solution is given by the saddle point of the Lagrangian, $\max_Z \min_A \mathcal{L}(A, Z)$, which can be iteratively found. Although this approach yields a correct solution, a major problem of this method is scalability, since a convex problem has to be solved with the two matrices A and Z of the same size as X.

Alternating Minimization

The last method in this section involves a reasonable approach to solving the matrix completion problem for a large matrix X. Unlike the convex optimisation approach, the alternating minimization algorithm on hand is scalable. However, while these methods have been successfully used for many years, theoretical guarantees for the convergence and optimality do not exist. Although under certain conditions, alternating minimization methods do converge to the globally optimal solution, see for example [41].
The idea behind alternating minimization is to find μ, U and Y that minimize the error $\|X - \mu 1^\top - UY\|_F^2$ for the known entries of X.

As in the previous approach, let $\Omega = \{(i,j) : w_{ij} = 1\}$ be the set of known entries of X, then the cost function to be minimized is given by

$$\|\mathcal{P}_\Omega(X - \mu 1^\top - UY)\|_F^2 = \|W \odot (X - \mu 1^\top - UY)\|_F^2 = \sum_{i=1}^{D}\sum_{j=1}^{N} w_{ij}(x_{ij} - \mu_i - u_i^\top y_j)^2,$$

where $\mathcal{P}_\Omega : \mathbb{R}^{D\times N} \longrightarrow \mathbb{R}^{D\times N}$ is the orthogonal projector onto the span of all matrices vanishing outside of Ω, that is those with missing entries.
In the case of complete, zero-mean data, this problem reduces to the low-rank matrix approximation problem based on explicit factorization $\min_{U,Y} \|X - UY\|_F^2$ which can be solved with the singular value decomposition of X. The alternating minimization algorithm provides an alternative to the SVD solution.
The top d eigenvector of a matrix can be computed by the *orthogonal power iteration method*: Suppose that $A \in \mathbb{R}^{N\times N}$ is a symmetric positive semi-definite matrix with eigenvectors $\{u_i\}_{i=1}^N$ and eigenvalues $\{\lambda_i\}_{i=1}^N$ sorted in decreasing order, such that $\lambda_d > \lambda_d + 1$. Let $U^0 \in \mathbb{R}^{N\times d}$ be an arbitrary matrix whose column space is not orthogonal to the subspace $\{u_i\}_{i=1}^d$ spanned by the top d eigenvectors. Then, the sequence of matrices

$$U^{k+1} = AU^k(R^k)^{-1},$$

where $Q^kR^k = AU^k$ is the QR decomposition of AU^k, converges to a matrix $U \in \mathbb{R}^{D\times d}$ whose columns are the top d eigenvectors of A with rate of convergence $\frac{\lambda_{d+1}}{\lambda_d}$.
Power Factorization is a generalization of this approach for computing the top d singular vectors of a (possibly) non-square matrix X. It iterates between the following two steps until it converges to the rank-d approximation of X.
Given $Y \in \mathbb{R}^{d\times N}$, an optimal orthogonal solution for U of the minimization problem is the Q factor of the QR decomposition of $XY^\top(YY^\top)^{-1}$. Given U, the optimal Y is $U^\top X$.
For matrix completion by power factorization with incomplete, zero-mean data, the two steps can be adapted. Given Y, the optimal U can be computed from

$$\left(\sum_{j=1}^{N} w_{ij}y_j y_j^\top\right) u_i = \sum_{j=1}^{N} w_{ij}x_{ij}y_j,\ i = 1, ..., D,$$

which can be derived from taking the derivative of the cost function with respect to u_i and setting it to zero. U can once again be replaced by the Q factor of the QR decomposition of $U = QR$ to fulfill the orthogonality constraint $U^\top U = I$.

Similarly, given U, the optimal Y is computed from

$$\left(\sum_{i=1}^{D} w_{ij} u_i u_i^\top\right) y_j = \sum_{i=1}^{D} w_{ij} x_{ij} u_i,\ j = 1, ..., N,$$

which is obtained by calculating the derivative of the cost function with respect to y_j and setting it to zero.

The PCA problem with incomplete data can now be solved similarly with the alternating minimization approach. Since the mean μ is non-zero and has to be recovered, the derivative of the cost function with respect to μ_i,

$$\left(\sum_{j=1}^{N} w_{ij}\right) \mu_i = \sum_{j=1}^{N} w_{ij}(x_{ij} - u_i^\top y_j),\ i = 1, ..., D,$$

is used to compute the optimal μ. To additionally enforce the uniqueness constraint $Y1 = 0$, μ can be replaced by $\mu + \frac{1}{N}UY1$ and Y by $Y(I - \frac{1}{N}11^\top)$. Then, given μ and Y, the optimal U is computed similarly to before including the non-zero mean and vice versa.

The complete procedure can be summarized as in algorithm 1.

Algorithm 1 Matrix Completion by Alternating Minimization

Input: Matrices $W \in \mathbb{R}^{D\times N}$, $X \in \mathbb{R}^{D\times N}$, dimension $d \in \mathbb{N}$

Initialize $\begin{bmatrix} u_1^\top \\ \vdots \\ u_D^\top \end{bmatrix} \leftarrow U^0 \in \mathbb{R}^{D\times d}$ and $[y_1, \cdots, y_N] \leftarrow Y^0 \in \mathbb{R}^{d\times N}$

while $\mu 1^\top + UY$ has not converged **do**

$\mu_i \leftarrow \frac{\sum_{j=1}^{N} w_{ij}(x_{ij} - u_i^\top y_j)}{\sum_{j=1}^{N} w_{ij}}$

$u_i \leftarrow \left(\sum_{j=1}^{N} w_{ij} y_j y_j^\top\right)^{-1} \sum_{j=1}^{N} w_{ij}(x_{ij} - \mu_i) y_j$

$U = \begin{bmatrix} u_1^\top \\ \vdots \\ u_D^\top \end{bmatrix} \leftarrow UR^{-1}$, where $QR = \begin{bmatrix} u_1^\top \\ \vdots \\ u_D^\top \end{bmatrix}$

$Y = [y_1, \cdots, y_N]$, where $y_j \leftarrow \left(\sum_{i=1}^{D} w_{ij} u_i u_i^\top\right)^{-1} \sum_{i=1}^{D} w_{ij}(x_{ij} - \mu_i) u_i$

$\mu \leftarrow \mu + \frac{1}{N}UY1$ and $Y \leftarrow Y(I - \frac{1}{N}11^\top)$

Output: μ, U and Y

Algorithm 1 coincides with algorithm 3.5 in [34]

4. Examples with Simulated Data

In the next two chapters, various applications for the completion methods in the previous chapters are presented to assess the general capabilities of each method for different real-world problems. The algorithms are implemented in Python 3.7.
In this chapter, the completion methods are used on simulated data sets. In the subsequent chapters, the methods are applied to real data sets, first for images of human faces in chapter 5.1 and then for share indices in chapter 5.2.
To be able to evaluate the performance of each described method, the next section briefly introduces the performance criteria used, see [42] for an exemplary paper using several different performance criteria.

4.1 Performance Criteria

L^p-norm and error

While there are many possible criteria to evaluate the performance of algorithms, one of the main criteria very often used is related to L^p-norms, also in an analytical context known as l^p-norms, defined as follows.
Let $1 \leq p < \infty$ be a real number, then the L^p-norm of a finite vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is defined as

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}.$$

The Minkowski distance of order p between two points $x, y \in \mathbb{R}^n$ is similarly defined as

$$d_p(x, y) = \|x - y\|_p = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}.$$

This is closely related to the generalized mean defined as

$$M_p(x_1, \ldots, x_n) = \left(\frac{1}{n} \sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

and a generalized mean error given as

$$ME_p(x_1, \ldots, x_n) = \left(\frac{1}{n} \sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}.$$

The limit case $p = \infty$ for all these quantities is given as a variant of the first limit case, which is defined as $\max_i |x_i|$.

The calculations for the completion methods are usually done with the Euclidean distance, which is the special case $p = 2$ of the Minkowski distance. For example, OLS and PCA are dependent on the L^2-norm.

This is the reason why the Root-Mean-Square Error (RMSE) is an evaluation criterion used in the subsequent analysis of the algorithms applied to various data sets.

The RMSE for the observed values x^{obs} and the completed values $\hat{x}$ is defined as the square root of the mean square error (MSE), the case $p = 2$ for the generalized mean error, such that

$$x_{RMSE} = \sqrt{x_{MSE}} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i^{obs} - \hat{x}_i)^2}.$$

Another criterion used is the Mean Absolute Error (MAE), the average of the difference between the actual values and the completed values of the data, given as the case $p = 1$, defined as

$$x_{MAE} = \frac{1}{n} \sum_{i=1}^{n} |x_i^{obs} - \hat{x}_i|.$$

For some applications it might be relevant to not only look at a mean error, but to look at the error for each missing value. Especially in the later chapters on real-world applications, the absolute error for each completed value is computed and then displayed for further inspection.

Another point of view is to be taken for noisy data, especially in the following sections, where the error should be calculated from the underlying model and not from the actual event, such that in the subsequent sections with noisy data the error is calculated as a deviation from the expected value.

For example, in the case of absolute values, the deviation may be calculated as the average absolute deviations from the mean,

$$x_{dev} = \frac{1}{n}\sum_{i=1}^{n} |x_i^{obs} - \bar{x}|,$$

for $\bar{x}$ being the mean of x for the noisy data.
The standard deviation, typically used in a statistical analysis, is defined as the square root of the average of the squared deviations from the mean,

$$x_{std} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i^{obs} - \bar{x})^2}.$$

Execution time
The second criterion used for the subsequent examples is another relevant factor for comparing the completion methods. The time needed for the execution of the algorithm is important, especially for large data sets. Algorithms especially developed for very large data sets exist and other algorithms do not scale well. Hence, the execution time of each method applied to the data sets is evaluated as well.

While the previously described criteria are applicable very well to all data sets, particular examples require special criteria to be better able to compare the results. Several real world applications have introduced their own terms for the comparison of results since an L^2-distance might not make sense in the context of the task.
However, the algorithms for completion are usually computed with the L^2-norm, if applicable, and the results are then judged by an additional criterion to compare it in a task specific way. Specialized methods for those criteria and applications do exist, but they will not be further investigated as that would go beyond the scope of this work.

Total Variation (TV)
As one of the more specific concepts for image analysis is total variation, it will be used in section 5.1 to compare the results calculated with standard methods.
For a smooth u defined on the space $\Omega \in \mathbb{R}^2$, the total variation norm is essentially an L^1-norm of the gradient of u, defined as

$$\|u\|_{TV(\Omega)} = \int_{\Omega} |\nabla u| dx.$$

Images usually have discrete values and grayscale images are typically described with 256 possible pixel values which encode levels of grey ranging from black, given by the value 0, to white, encoded as 1, and all levels of grey with values in $(0, 1)$, since they are normalized, such that an image u is defined on $\Omega = [0, 1]^2$.
Total variation may be discretized (see [43]), such that for the matrix $(u_{i,j})$ encoding the discrete image for $i, j = 1, \ldots, n$ pixels, it holds that

$$\|u\|_{TV(\Omega)} \approx \sum_{i,j} \sqrt{(\nabla_x u)_{i,j}^2 + (\nabla_y u)_{i,j}^2},$$

where ∇_x, ∇_y are discretizations of the derivatives.
Possible discretizations of the derivatives include central differences,

$$(\nabla_x u)_{i,j} = \frac{u_{i+1,j} - u_{i-1,j}}{2} \quad \text{and} \quad (\nabla_y u)_{i,j} = \frac{u_{i,j+1} - u_{i,j-1}}{2}$$

or, more commonly used, one-sided finite differences,

$$(\nabla_x u)_{i,j} = u_{i+1,j} - u_{i,j} \quad \text{and} \quad (\nabla_y u)_{i,j} = u_{i,j+1} - u_{i,j}.$$

Other more complex nonlinear discretisations exist as well to achieve symmetry and possibly even consistency (see [44]).

Directional accuracy

After discussing a relevant error measure for images in section 5.1, two useful measures to check the precision of the share data completion in section 5.2 are introduced.
The completion approaches specifically targeted to time series data were not the focus of the introduced methods, as more general concepts were addressed. However, the results calculated with those algorithms may still be analyzed with a more fitting measure for time series data. While an error norm yields a comparable result, it is not necessarily as meaningful in a stock exchange environment, whereas the directional accuracy is more significant. If it is high enough, then shares can be bought and sold more efficiently.
The directional accuracy of the completion methods is computed, such that the direction of the observed share price is compared with the direction of the completed share price for each trading day. The trend of the share price from one trading day to the next, given by the closing price of a share, may be encoded with the values $\{-1, 0, 1\}$, where -1 stands for a decrease in the share price, 1 is equivalent to an increased share price and 0 encodes the unlikely event of no change in share price.

More formally, for a share price vector $x \in \mathbb{R}^n$ on n trading days, the direction of the share $d(x_i)$ on day i is given as

$$d(x_i) = \text{sgn}(x_i - x_{i-1})$$

with $\text{sgn}(x)$ being the sign-function of x taking the values as described above.
Comparing the direction of the observed share price $d(x_i^{obs})$ and the completed share price $d(x_i^{compl})$ may also be encoded by the values $\{-1, 0, 1\}$ as

$$d^{err}(x_i) = \text{sgn}(d(x_i^{obs}) - d(x_i^{compl})).$$

The directional accuracy of a method may then be calculated with the mean of the absolute values of $d^{err}(x_i)$ over all completed trading days $m < n$ as

$$d_m^{acc}(x) = 1 - \frac{1}{m} \sum_{i=1}^{m} |d^{err}(x_i)| \ \in [0, 1],$$

similar to the MAE. This results may be multiplied with 100 to receive percentage values in the range $[0, 100]$. The higher the directional accuracy in this context, the better.

Another closely related possibility to measure the performance of the algorithm is calculating the difference of the rate of change for two consecutive trading days between the actual share values and the completed share values, which is equivalent to a discretized version of a derivative for a time step 1 and as such similar to the TV norm explained above.

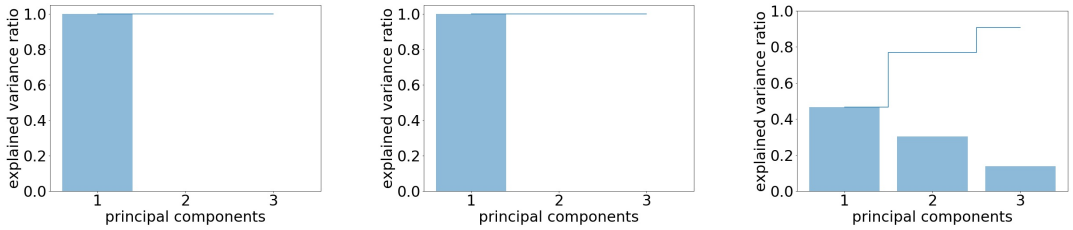
Thus, based on the fact that the missing data is artificially introduced in the following examples and a completion error can be calculated exactly, the performance criteria for comparing the completion methods in the following applications include mean errors, execution time, total variation and the directional accuracy.
If the incomplete data is not artificial, only the quality of the method may be evaluated, see section 2.5.2 in [9] for possible evaluation criteria which also include the root mean square deviation.

4.2 Linear Example

Since several methods mentioned in the previous chapters use linear models, the first simple example is based on a linear function.
An inclined two-dimensional plane in the three-dimensional real space is given as follows: For two real-valued vectors x and y, whose entries are evenly spaced samples over a specified interval that form a grid, z is given as the linear function

$$z = f(x, y) = ax + by + c$$

for fixed constants $a, b, c \in \mathbb{R}$.
In the following, the variables are exemplary set as $a = 5, b = -2$ and $c = 10$ and the two vectors x, y are given as 100 evenly spaced samples over the interval $[-1, 1]$ which induces a grid of dimension 100×100 with 10000 grid points. The data for z is hence given on those grid points and stored in a matrix of dimension 100×100. This example is also illustrated later in figure 4.3a.
The results are similar for a grid with more data points and less data points as well. Since the underlying structure is very simple, very few data points suffice to be able to accurately find the data structure with most methods. A denser grid increases the computation time, but does not improve the following results. Hence, the used number of data points is quite arbitrarily chosen in this example.
If the data were preprocessed, such that it were centered to the mean and maybe even additionally component-wise scaled to unit variance, the axis intercept given by c would be irrelevant for the completion due to centering the data. The slope of the plane however would not be changed by this type of preprocessing.
The preprocessed data would result in perfect completion results in the linear example described above, however this would be a very constructed example since the mean and the variance of the complete data are used and the missing data is only introduced afterwards such that the complete data is known and used during the preprocessing step, which is not realistic. Hence, this is not relevant for any type of real-world example.
Using a standardization with incomplete mean and variance, which only includes the mean and variance of the known data and ignores missing values, would not change the results significantly in the following examples since the relevant algorithms include calculations with nonzero mean.
Thus, the data is utilized in the following completion methods without scaling or transforming the data.



(a) PCA

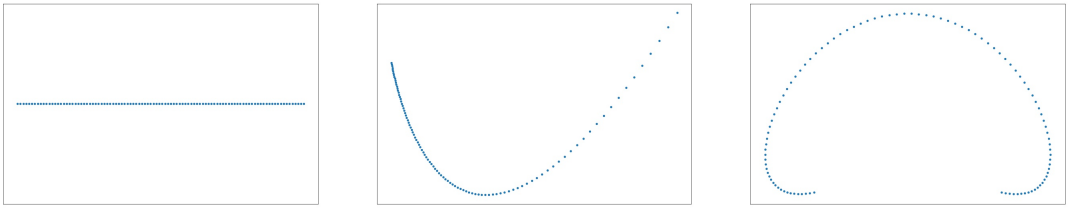
(b) KPCA with a polynomial kernel

(c) KPCA with radial basis functions

Figure 4.1: The first three principal components and their explained variance ratios of the linear example computed by PCA and Kernel PCA with a polynomial kernel and radial basis functions respectively.

Using implementations given by the machine learning library scikit-learn [45] for the subsequent methods, PCA and Kernel PCA are computed for the complete data.
In figure 4.1, the principal components and the explained variance ratios are depicted. The bar plot illustrates the individual explained variance ratios calculated by PCA, which is the same as KPCA with a linear kernel, as well as KPCA with an exemplary chosen polynomial kernel of degree 3 and Gaussian radial basis functions, respectively, where the line illustrates the cumulative explained variance ratio.

Due to the underlying simple linear structure of the data, only one principal component is needed for linear PCA and KPCA with a polynomial kernel to explain the variance. Since a polynomial kernel is a generalization of a linear kernel, as a linear kernel is a polynomial kernel of degree 1, it is possible to achieve the same results for both methods in this example.
Interestingly, for more complex nonlinear kernels in KPCA, more components are introduced and the explained variance ratio of the main component decreases. Since KPCA can be described as embedding the data nonlinearly into a high dimensional space to be able to use linear PCA there, the dimension of the data is artificially increased. Thus, the possible number of components increases.
For example, KPCA with radial basis functions needs more principal components to describe the variance in the data with the first principal component explaining slightly less than 50% and the first three principal components explaining approximately 90%, whereas five principal components yield a cumulative explained variance ratio of 99%.
Overall, it might be suggested that a dimensionality reduction with standard PCA is better suited for this linear data set than a dimensionality reduction with KPCA and more complex kernels.



(a) PCA

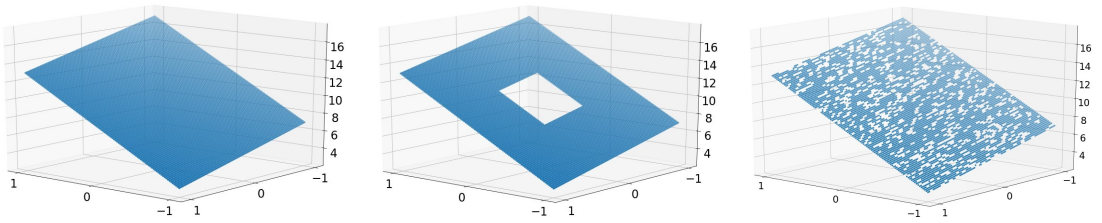
(b) KPCA with a polynomial kernel

(c) KPCA with radial basis functions

Figure 4.2: Two-dimensional embedding of the linear example data given by PCA and Kernel PCA with a polynomial kernel and radial basis functions respectively.

The corresponding two-dimensional embeddings of the data computed by PCA and KPCA with a polynomial kernel and radial basis functions can be seen in figure 4.2. A clear structure can be found in all three embeddings. PCA yields an embedding which is linear, while the embeddings given by KPCA with a polynomial kernel of degree three and the one calculated with Gaussian radial basis functions are both nonlinear.
The first principal component for KPCA with a polynomial kernel technically does not include 100% of the explained variance ratio, but only about 99%, which explains the possibility of a nonlinear two-dimensional embedding. The first principal component of linear PCA however includes 100% of the explained variance ratio up to machine precision. Thus, the first two embeddings include almost all of the variance in the data, whereas the two-dimensional embedding given by KPCA with radial basis functions is based on the first two principal components, which only include approximately 80% of the explained variance ratio.

So far in the analysis, the data set has been complete as depicted in figure 4.3a. Now, by introducing missing values and setting the chosen points in the data that are supposed to be missing to "NaN" (not a number), the data is no longer complete and the missing values need to be reconstructed. To achieve that goal, the methods which were introduced in chapters 2 and 3 may be applied to complete the missing data.
In the following paragraphs, two cases of missing data in the given linear example are analyzed. In the first example, an area of missing data is artificially introduced, as pictured in figure 4.3b, with a square in the interior of the plane being missing. The second example includes a percentage of random missing points as illustrated in figure 4.3c, where the percentage of missing data is exemplary chosen as 10%.



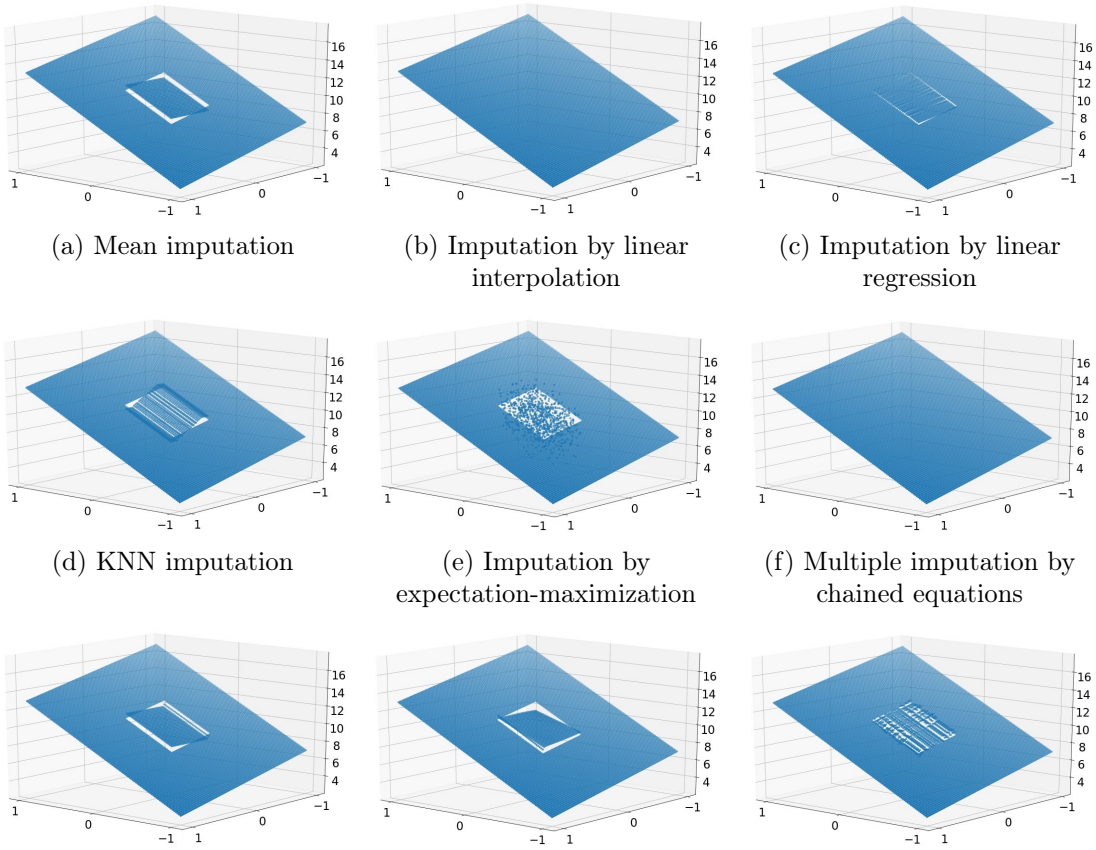
(a) Complete example (b) Missing area (c) Missing random points

Figure 4.3: Linear example

Starting with the first case, the linear example includes an illustrated missing square of data which is imputed with the described methods.
Completing the data with a simple imputation method like using a central tendency (mean, median or mode) as well as filling the missing values backwards or forwards with the last observation carried forward or the next observation carried backward, respectively, does not yield reasonable results, with the example of mean imputation shown in figure 4.4a.
Due to the underlying structure, interpolation is a good alternative. The calculated interpolation results are based on all available grid points, where the number of necessary neighboring grid points is dependent on the degree of the polynomial. It utilizes the values on and close to the boundary of the missing area. Choosing more or less grid points for a finer or coarser grid respectively does not change the interpolation results. Since the data is derived from a linear object, it is possible to fill the missing square without error with a linear interpolation. Quadratic interpolation and higher polynomial interpolations yield the same result, but they are not necessary, since the coefficients of the terms with higher exponents than the linear term will be zero. Spline interpolation is also a viable method. As an example, figure 4.4b shows the result of a linear interpolation of the missing data, which is an exact reconstruction of the original structure in the data. Similarly, an imputation based on a linear regression produces a good completion result, as can be seen in figure 4.4c. Although, the illustration shows that the imputation is not perfect.
Looking at more complicated imputation methods, it can be seen in figures 4.4d and 4.4e that imputation with kNN and especially using the likelihood based expectation-maximization algorithm are not reasonable choices for completing the data in this example. Utilizing multiple imputation by chained equations, the resulting completed data set does not differ from the original data, as seen in figure 4.4f. Thus, MICE is a very good choice for the imputation method in this example.

Going further, the methods using dimensionality reduction can be considered as well. Linear PCA yields an acceptable result, with a slight tilting due to the complete PCA only being run on two dimensions and the third dimension being the one with incomplete data. Another complicating factor is that PCA is being done globally on the data, so that missing data might result in PCA trying to fit a different structure which might be nonlinear, even though the underlying structure is linear.
Using Kernel PCA as well as Diffusion Maps for completing the data does not improve on the results of PCA in this case, with figure 4.4g showing the completion results with the missing entries being calculated via a single PCA of the complete subset of the incomplete data.
Iterative PCA completes each column of the incomplete data iteratively, similar to MICE. After a simple imputation with a constant value for all missing values, in this case setting all unknown values to zero, all previously missing entries in a specified column are imputed. This is done by calculating the weights needed for the principal components to represent the given entries of the incomplete column. The principal components are computed by PCA on the completed data without the chosen column. This is then iterated for all other columns with missing data, where the imputed data is updated in each step. The difference to incomplete PCA is that here, the data to be used for PCA is first completed by a simple imputation method of choice and then updated in each step, such that only one column is not used for PCA in each iteration, whereas for incomplete PCA, the analysis is only done on the complete subset and thus not applicable to data with missing values in most of the columns. Figure 4.4h illustrates that iterative PCA improves on the results of incomplete PCA.
Using matrix completion by alternating minimization improves the completion results even further compared to the simpler incomplete PCA, as can be seen in figure 4.4i. However, it does not reach the exact result as interpolation and MICE do.

Considering some of the evaluation criteria introduced in the previous section, it can be observed that none of the errors yield surprising results based on the illustrated completed data. The L^p-error for $p = 1, 2$ as well as the total variation varies between zero for the best result calculated with various interpolation methods and the highest error given by mode imputation. The root mean square error for the interpolation results as well as the results of MICE is zero, while mode imputation produces the highest RMSE with a value of 0.6. The mean absolute error ranges similarly between zero for interpolation as well as MICE and the highest MAE being assigned to mode imputation. All methods are adequately fast for this example, even though it already becomes apparent that MICE is the slowest method in comparison with all other tested methods.



(a) Mean imputation

(b) Imputation by linear interpolation

(c) Imputation by linear regression

(d) KNN imputation

(e) Imputation by expectation-maximization

(f) Multiple imputation by chained equations

(g) Incomplete PCA

(h) Iterative PCA

(i) Alternating minimization

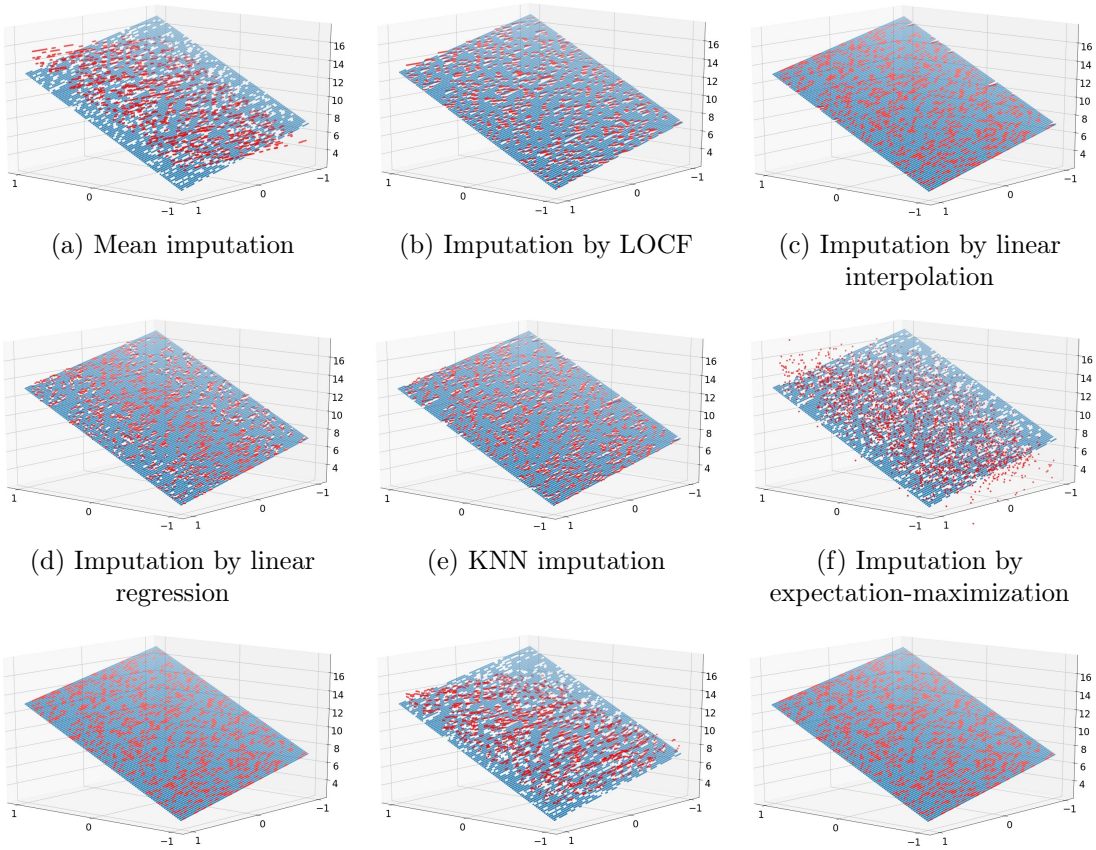
Figure 4.4: Completion results for the linear example with an area of missing data.

Thus, it may be concluded that for the first example, interpolation as well as MICE are the preferred choices for reconstructing the missing data, while imputation by linear regression and matrix completion by alternating minimization also yield acceptable results. However, this example does not include random missing data points, but only one large area of missing data and is as such not necessarily a good indicator for the performance of the methods with practical applications that include more scattered missing data.

Thus, more randomly distributed missing values are achieved by looking at the second example, where a certain percentage of values is artificially removed, which implies that the data is missing at random.
In the following illustrations of the reconstructions computed with the described methods, the imputed values are colored in red to better distinguish the completed results from the incomplete data.

The simple imputation results behave in a similar way as they did in the first example, this time with imputation by LOCF in figure 4.5b in addition to mean imputation in figure 4.5a as the illustrated examples. It can be discerned that simple imputation by a central tendency does not result in reasonable results. While filling the missing values by using the LOCF or the NOCB improves on these results, the imputation computed by filling the values forward or backward is still not very accurate.
As in the previous example, interpolation yields good results due to the underlying structure of the data, see figure 4.5c for the results of linear interpolation, even though there are some aberrations on the boundary for polynomial interpolation results since it is problematic to interpolate with the missing boundary terms. However, the imputation results for linear regression worsen, with several gaps in the illustration of the completed data, see figure 4.5d.
Using the expectation-maximization algorithm yields bad results, as it did in the first example, with many outlier, see figure 4.5f. Imputation using kNN does not fill the gaps correctly either, as can be seen in figure 4.5e.
Incomplete PCA is not applicable in this example, since there is no complete sub-matrix that may be used for the computation. In contrast, iterative PCA may be applied, but the results are not satisfactory, as shown in figure 4.5h.
MICE as well as matrix completion by alternating minimization yield accurate completions, with the results of MICE shown in figure 4.5g, while the results given by alternating minimization are illustrated in figure 4.5i.

Once again, the performance criteria may be examined. They do not include any surprising results based on the illustrations of the complete data. Matrix completion by alternating minimization, MICE and the various interpolation methods achieve negligible L^p-errors for $p = 1, 2$ as well as total variation.
While the RMSE for matrix completion by alternating minimization, multiple imputation by chained equations and spline interpolation is zero and negligible for the other polynomial interpolation results, albeit not being zero due to the problems on the boundary, the results computed with the imputation by a central tendency as well as the expectation-maximization algorithm and iterative PCA yield a high RMSE, with mode imputation once again having the highest RMSE. Similar results can be found for the MAE of the completion methods.
Table 4.1 summarizes the results of the performance criteria for the completion methods applied to the second example with the relevant errors rounded to three decimal places.



(a) Mean imputation

(b) Imputation by LOCF

(c) Imputation by linear interpolation

(d) Imputation by linear regression

(e) KNN imputation

(f) Imputation by expectation-maximization

(g) Multiple imputation by chained equations

(h) Iterative PCA

(i) Alternating minimization

Figure 4.5: Completion results for the linear example with random missing data.

Hence, it may be concluded that MICE and alternating minimization are two very useful methods in this context, which shall be further underlined in the next chapters with real-world applications. However, linear interpolation results in (almost) perfect reconstruction due to the data structure and is hence the preferred method for this example. This is, however, not really the case anywhere but in a strategically modeled example and the conclusion may already be discarded with the following slightly more complicated constructed example.

Method	L2	L1	TV	RMSE	MAE	Time
Mean	29.847	50.837	48.747	0.571	0.246	<0.1s
Median	30.267	53.818	51.354	0.585	0.255	<0.1s
Mode	60.765	104.242	93.859	1.131	0.496	<0.1s
LOCF	2.922	7.556	5.414	0.060	0.026	<0.1s
NOCB	2.905	7.556	5.414	0.061	0.026	<0.1s
Linear Regression	1.679	2.912	3.131	0.048	0.015	<1s
Linear Interpolation	0.718	1.131	0.566	0.008	0.000	<0.1s
Quadratic Interpolation	0.718	1.131	0.566	0.008	0.000	<0.1s
Spline Interpolation	0.000	0.000	0.000	0.000	0.000	<0.1s
KNN	1.467	3.911	3.046	0.037	0.013	<0.1s
EM	32.902	66.161	81.684	0.800	0.324	<1s
MICE	0.000	0.000	0.000	0.000	0.000	∼1s
Iterative PCA	23.022	85.269	74.969	0.488	0.195	<1s
Alternating Minimization	0.000	0.000	0.000	0.000	0.000	<0.1s

Table 4.1: Overview of the performance criteria for the completion methods applied to the linear example with 10% random missing data.

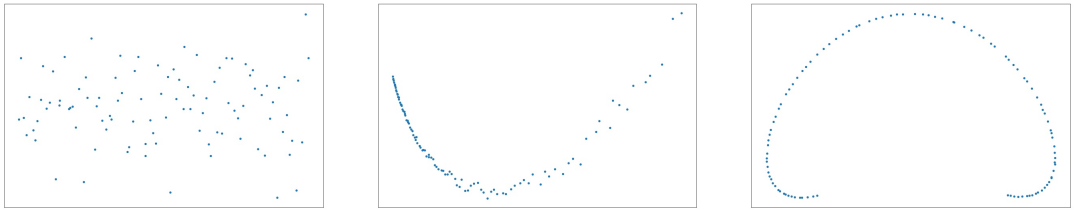
4.3 Noisy Linear Example

To make the previous example slightly more challenging and more similar to a real-life application, noise can be added to the data by randomly drawing samples from a normal distribution for all matrix entries and adding a sample to each one of them. The probability density function of a normal distribution is given by

$$\phi(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

with the mean and standard deviation in the present case chosen as $\mu = 0$ and $\sigma = 0.1$, respectively. With the added noise, the computations from the previous section may be repeated for the new data set.

The explained variance ratios of the principal components for the noisy data computed via PCA and KPCA with a polynomial kernel as well as radial basis functions achieve values similar to the linear model without noise, where the first principal component of standard PCA and KPCA with a polynomial kernel achieves almost 100% of the explained variance ratio, whereas KPCA with an rbf kernel needs more than one component, such that the first component includes around 50% of the explained variance ratio and the first three principal components explain approximately 90% of the variance.



(a) PCA

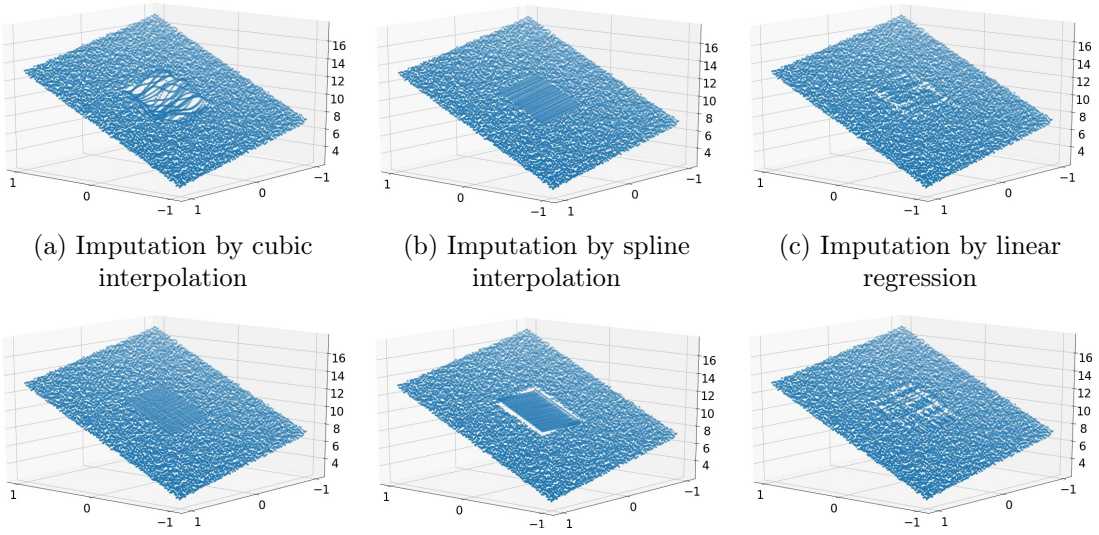
(b) KPCA with a polynomial kernel

(c) KPCA with radial basis functions

Figure 4.6: Two-dimensional embedding of the linear example data given by PCA and Kernel PCA with a polynomial kernel and radial basis functions respectively.

The two-dimensional embeddings of the noisy data shown in figure 4.6 however differ from their counterpart without noise. While the embedding of the noisy data given by KPCA with radial basis functions is akin to the embedding without noise and as such apparently rather robust, the two-dimensional embedding given by the first two principal components computed by KPCA with a polynomial kernel is slightly more noisy than the embedding for the data without noise, but an underlying structure is still clearly visible. However, the embedding given by the components calculated with PCA is quite chaotic and no clear structure is visible. Even though the data still has an underlying linear structure, standard PCA is no longer able to recognize this structure after the addition of some noise, which is a displeasing result.

Now, introducing missing information in the data again, the previously discussed example with an artificially introduced area of missing values resembling a square may be analyzed as a first example with added noise, where a few selected results are depicted in figure 4.7.
Imputation by a central tendency as well as filling the missing values by using the LOCF or the NOCB achieves similar results as before, where none of the imputation results are satisfactory, whereupon mode imputation once again brings up the rear with respect to the error of the imputed values. Interestingly, while linear interpolation doesn't decline much in quality, polynomial interpolation of degree ≥ 2 worsens noticeably compared with the noise-free data. This can be easily explained by the interpolation trying to find higher dimensional structures in the noisy data for higher exponents, whereas linear interpolation yields acceptable results. Despite that, spline interpolation of higher degree is able to achieve acceptable results as before, apparently with a piecewise linear interpolation.



(a) Imputation by cubic interpolation

(b) Imputation by spline interpolation

(c) Imputation by linear regression

(d) Multiple imputation by chained equations

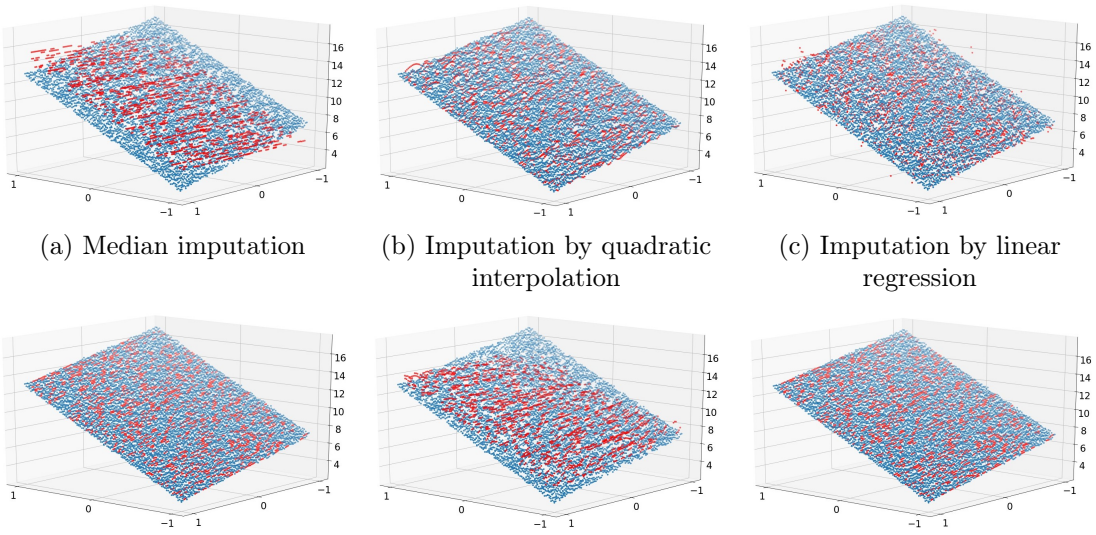
(e) Incomplete PCA

(f) Alternating minimization

Figure 4.7: A selection of completion results for the noisy linear example with a large area of missing data.

While imputation with linear regression involved worse results than polynomial interpolation of higher degree in the linear data without noise, this is no longer the case for the given noisy data. Now, linear regression is able to achieve better results than interpolation with higher dimensional polynomials, even though linear interpolation an spline interpolation still produces better results. Imputation by kNN and the expectation-maximization algorithm yield unsatisfactory results as before, while MICE still produces a very accurate imputation. Examining the matrix completion methods using dimensionality reduction, incomplete PCA as well as iterative PCA produce inadequate results, while the completed values calculated with alternating minimization are more acceptable.

The performance criteria once again include no surprises after looking at the illustrations of the completed data. First, the L^p-errors and total variation as well as the RMSE and the MAE are examined, which underline the described results.
None of the simple imputation methods have small errors, with mode imputation involving the highest errors, followed by the errors for the expectation-maximization algorithm, while imputation with kNN causes slightly lower errors. Polynomial interpolation of a higher degree also yields rather high errors, while linear and spline interpolation lead to negligible errors. In this example, MICE causes the smallest errors, while the matrix completion methods all give rise to mediocre error results.



(a) Median imputation

(b) Imputation by quadratic interpolation

(c) Imputation by linear regression

(d) Multiple imputation by chained equations

(e) Iterative PCA

(f) Alternating minimization

Figure 4.8: A selection of completion results for the noisy linear example with a percentage of randomly missing data.

Additionally, the evaluation criteria for this example may also include the comparison of the deviations, since the error terms are calculated with noisy data and should therefore be taken with a grain of salt as the distance of the completed value to the actual value may be comparatively high, while the deviation of the completed value would be actually very small and as such rather accurate. To incorporate this phenomenon in the analysis, the mean average deviation as well as the standard deviation of the completed data is computed alongside the errors and studied in relation with the MAD and STD of the given complete data, respectively, with comparable results. Comparing the MAD of the completion methods, it can be seen that especially mode imputation as well as EM to a slightly lesser degree increase the MAD of the data significantly, whereas MICE and spline interpolation are able to complete the data such that the MAD remains the same. All other methods do not change the MAD very much, it is only slightly increased, but none of the methods decrease the MAD.

Turning towards the second example introduced in the previous section, where a percentage of data points is randomly missing, normally distributed noise is once again added to the data. A selection of the previously addressed completion methods applied to the noisy data is illustrated in figure 4.8.

Unsurprisingly, none of the simple imputation methods yield very accurate results. Similar to the first noisy example, the interpolation results deteriorate with higher dimensional polynomial functions, while spline interpolation is still able to achieve a good completion of the missing values. In contrast to before, linear regression is not a good choice for this example, as there are outlier in the completed data.
Using the maximum-likelihood based EM algorithm gives poor results, while imputation by kNN again produces a mediocre outcome. Once more, MICE is a good choice for the imputation of the missing information. As in the noise-free example, iterative PCA does not compute acceptable results, while matrix completion by alternating minimization produces very good results.

The analysis of the performance criteria leads to expected outcomes. Imputation by a central tendency, once again especially mode imputation, and the EM algorithm as well as iterative PCA entail high errors, followed by linear regression. While no method is obviously without error due to the noise addition, linear and spline interpolation, MICE as well as matrix completion by alternating minimization lead to small errors. Filling the values forward or backward as well as imputing the values by using kNN entails mediocre error results. Looking at the deviation of the completed data, it can be seen that mode imputation and EM increase the MAD significantly, while mean and median imputation as well as linear regression and iterative PCA increase it to a lesser degree, whereas the other methods do not change it substantially. Once again, no method decreases the MAD.

Effect of the percentage of randomly missing data
The percentage of randomly missing data up to this point has been manually set to 10%. However, this is an arbitrary choice and therefore, the impact of the missing percentage on the completion results is examined in the following paragraph.
For that matter, the previously used methods are applied to a range of percentages of randomly missing data in the last example of a linear noisy data set, starting with 1% and ranging up to 80% of missing data in 1% steps. Increasing the percentage of randomly missing data even further may lead to problems in the completion methods, starting with completely missing columns or rows, which most methods do not have the ability to handle, as well as columns or rows which are very sparsely filled. As an example in that case, polynomial interpolation of a higher degree may run into problems, since several entries are necessary for the interpolation to be feasible. Other completion methods do encounter similar problems for very sparse matrices, for instance if the matrix is singular, which means that the determinant of the matrix is zero, it is no longer possible to

compute the eigenvectors of the matrix as they do not exist in this regard, such that a PCA is no longer possible. Thus, the percentage of missing data has been capped at 80% for the analysis. Additionally, an amount of missing data higher than that is usually unfit for most further treatment as the amount of missing information grows too large. To prevent any outlier in the randomly chosen missing data from strongly impacting the results, the completion is performed several times. The performance criteria applied to the results are then averaged over all executions. Hereinafter, the computations are performed 100 times. To avoid complications in the error calculations, boundary values that may not be filled by using the LOCF are imputed with the NOCB instead to avoid not being able to impute those entries.

Several of the evaluation criteria, namely the L^1-error and the L^2-error as well as the root mean square error and the mean absolute deviation, are plotted for a diverse selection of the completion methods for the range of randomly missing data percentages spanning from 1% to 80%. They are illustrated in figure 4.10.
On the right-hand side, the legend for the performance criteria of the illustrated methods is shown, as it is the same for all the plots.

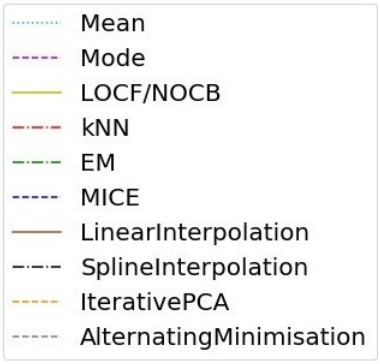
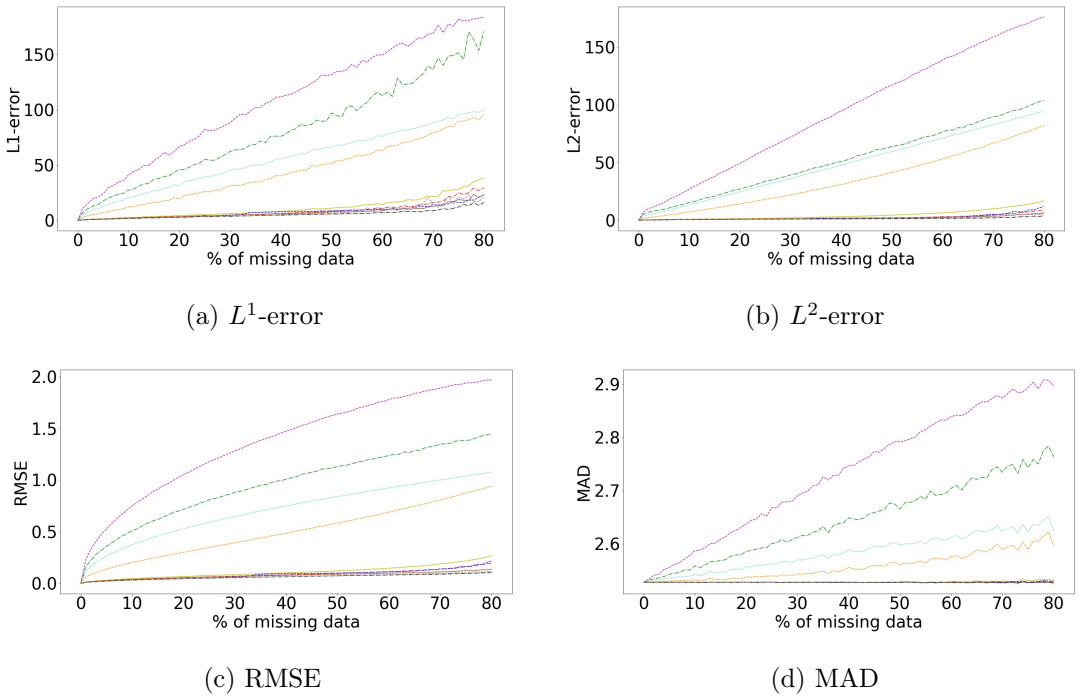


Figure 4.9: Legend of methods

In the first subfigure, the L^1-error is depicted for the selected methods. As expected, the error reaches the highest values for mode imputation, followed by the expectation-maximization algorithm, mean imputation and iterative PCA, in that order. Not depicted are the errors for median imputation, which are very close to the ones calculated for mean imputation. The lowest error is reached for spline interpolation closely followed by the rest of the selected methods, including the described methods that are not depicted. Interestingly, all methods behave in an approximately linear fashion, with a few peaks, such that it can be said that no matter which percentage of missing data is chosen, the magnitude of the error for each method may be ordered in the same way.
L^p-errors are closely related to each other, such that the correspondence between the L^1-error and the L^2-error is given as

$$\|x - y\|_2 \leq \|x - y\|_1 \leq \sqrt{n}\|x - y\|_2,$$

which follows from the Cauchy-Schwarz inequality. The norms are thus equivalent.



(a) L^1-error (b) L^2-error

(c) RMSE (d) MAD

Figure 4.10: A selection of evaluation criteria for the possible percentages of randomly missing data.

Thus, it is not surprising that the same associations between the different methods and for the different values of the percentage of randomly missing data can be found for the L^2-error as they were portrayed for the L^1-error, though the L^2-error develops in a slightly smoother mode.

Interestingly, similar performances may be detected for the RMSE and the MAD, which are illustrated in the second row of the figure. If one would plot the mean absolute error, the standard deviation or the total variation error of the methods, the results would be comparable to RMSE, MAD and L^p-errors, respectively.

The same succession of the error size computed from the completion methods can be discovered for RMSE and MAD, where mode imputation leads to the highest RMSE and MAD, followed by EM, mean imputation and iterative PCA in that order. Smaller errors may be found with the other methods. In the case of RMSE, spline interpolation causes the smallest error closely followed by alternating minimization and the other depicted methods. However, the RMSE performs asymptotically more like a logarithmic function instead of having a linear behavior as the error does not increase linearly with the percentage of missing data.

Hence, the behavior of the completion methods may be viewed as being independent of the chosen percentage of randomly missing data, such that using 10% in the previously described examples does not have any negative consequences for the analysis. If one were to illustrate the time needed to compute the imputation results for various percentages of missing data, it would be noticeable that the execution time does not change much for the range of the percentage of randomly missing data considered and is thus not dependent on the percentage itself. However, it is already observable that MICE has the longest execution time by far, followed by EM and iterative PCA. All other methods need very little time for one particular imputation of the incomplete data. More details about the execution time may be found in the following paragraph.

Effect of the size of the incomplete data set

The size of the data set had been set at beginning of this section, such that a grid of size 100×100 was used. However, it is possible to increase or decrease that size and analyze the impact on the computations. Hereinafter, whenever the size of the data set is referred to as a number n, it means that the grid is of size $n \times n$. Thus, until now, the size of the data set was fixed as $n = 100$. Looking at several different grid sizes, ranging from a very small grid of size 10×10 to a data set of size 1000, it is possible to study the impact on the evaluation criteria. With increasing size of the data set, the L^p-errors understandably increase while keeping 10% as the percentage of missing values fixed in each incomplete data set, which implies more missing values in nominal terms for larger data sets. RMSE and MAE as well as MSD and STD do not change with the data size. As previously noted, the defining effect of the size of the incomplete data set is realized on the execution time of the completion methods. Looking at several different grid sizes, it is possible to plot the execution time of each method for a selection of grid sizes.
In figure **??**, the execution times are depicted for grids of size $k^2 \times k^2$, where $k = 3, \ldots, 32$, such that the possible grid sizes range from 9×9 to 1024×1024. It is easy to see that MICE is very slow, with nonlinear time complexity, while the other methods are much faster, even though EM and iterative PCA take longer than the other methods as well and the time needed increases for larger grid sizes. For MICE, only the execution time up to a data size of 400 is depicted in the figure. The time needed for much larger data sets is no longer reasonable compared with the other completion methods. Thus, the effect of the size of the incomplete data set on the execution time is especially pronounced for MICE. It may be said that, while MICE may be an accurate imputation method and particularly useful for small data sets, it is not wise to apply it to larger data sets.

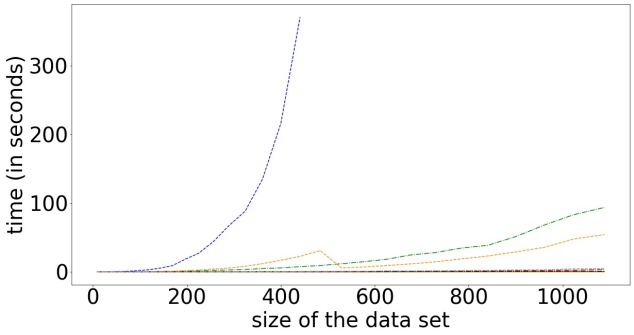


Figure 4.11: The execution time for some of the completion methods dependent on the size of the incomplete data set.

As a short side note, one may also look at nonlinear constructed data via some function $f(x, y) = ax^m + by^n + c$, for example with the same variables $a = 5, b = 2$ and $c = 10$ as well as $m = 1$ and $n = 3$ on the given grid constructed with x, y as evenly spaced samples on $[0, 1]$. The results however do not include any surprises. They are very similar to the observations made with the previously described linear example. Thus, they will not be included here.

As a conclusion, it can be noted that since most real world application are noisy to a certain extent, the previously described examples are already a good indication why several of the imputation methods are not a reasonable choice and others are a sensible option for the imputation of incomplete data. In the linear examples under consideration interpolation was a sensible option, owing to the clear underlying structure of the data, which is usually not the case for real-world applications. MICE is normally a very good choice for small data sets as well, which leads to accurate results. Matrix completion by alternating minimization is another reasonable choice for the completion of data sets with missing data, also for larger data sets.
In the next chapter, real-world applications are analyzed.

5. Two Real-World Applications

In this chapter, several different real-world data sets are used to evaluate the performance of the completion methods.
In the first section, a data set of human face images is used to analyze the possibility of completing parts of a picture based on the other face images and the complete part of the image.
The second section treats an application in the financial world. Considering stock indices, the defined completion methods are used to complete parts of a share based on the other share prices and the existing portion of the share price in question.

5.1 Faces

The first real-world example of an application for data completion are face images with missing data. Two different scenarios for missing values in face images will be presented. The data set of face images used is given as a complete data set and missing values are artificially introduced to be better able to compare the completions and evaluate the performance of each method.
In the first problem setting, a set of face images without missing values is given in a first step. Then, the problem is to find the missing values of a new face image by using the already given data. This is equivalent to having a complete subset of the data as in the first example in the previous chapter with a missing square of information.
The other problem setting will only include a set of incomplete face images, so that for all face images the missing values have to be found. This is to be understood analogously to the second example in the previous chapter with a percentage of randomly missing data.

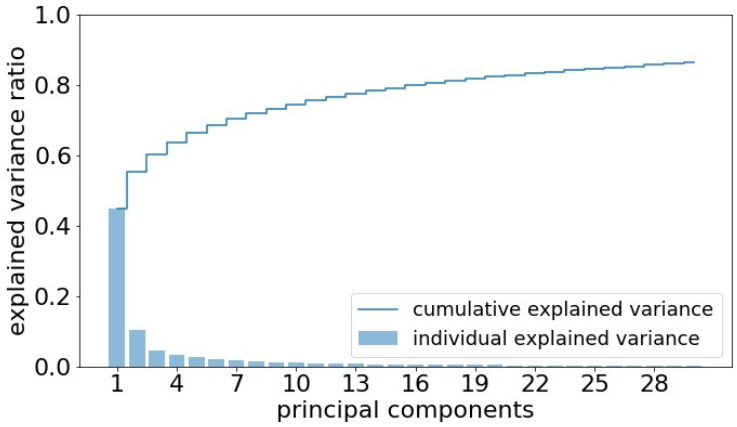


Figure 5.1: The principal components and their explained variance ratio of the Olivetti faces data set computed with PCA

Suppose that there is a set of N face images, where each image is of size $d \times d$. Every image is flattened to a vector of size $D = d^2$ which is saved as a column in the matrix of dimension $D \times N$. In this example, the Olivetti faces data set from AT&T Laboratories Cambridge ([46]) is utilized. Each face image is a grayscaled image with every pixel having an integer value between 0 and 256, which has been converted to a floating point value in $[0, 1]$. The data set includes 400 images of size 64×64 which are saved in a matrix of dimension 4096×400.

For the given complete data set, it is more efficient to compute the eigenvectors and eigenvalues of the smaller matrix $C = A^\top A \in \mathbb{R}^{400\times400}$ than those of $AA^\top \in \mathbb{R}^{4096\times4096}$. Similar to the previously used constructed linear example, the principal components and their explained variance ratio can be exemplary calculated with PCA. The first 30 principal components are depicted in figure 5.1. While the decay of the individual explained variance ratio is not as noticeable as in the previous examples, it is possible to observe that a small ratio of all 400 possible eigenvalues is already sufficient to encode a large amount of the explained variance ratio, such that 2 eigenvalues include 50% of the explained variance ratio and 17 eigenvalues are needed for 80% of the variance. For a cumulative explained variance ratio of 90% and 95%, 47 and 98 principal components are needed, respectively. This is a sizeable reduction from all 400 eigenvectors.

Reshaping the normalized eigenvectors $u_i,\ i = 1, ..., K$ as images of the original size 64×64, the so-called eigenfaces can be pictured. The first few eigenfaces of the Olivetti faces are depicted in figure 5.2, where it becomes clear why they are also sometimes called ghost faces.

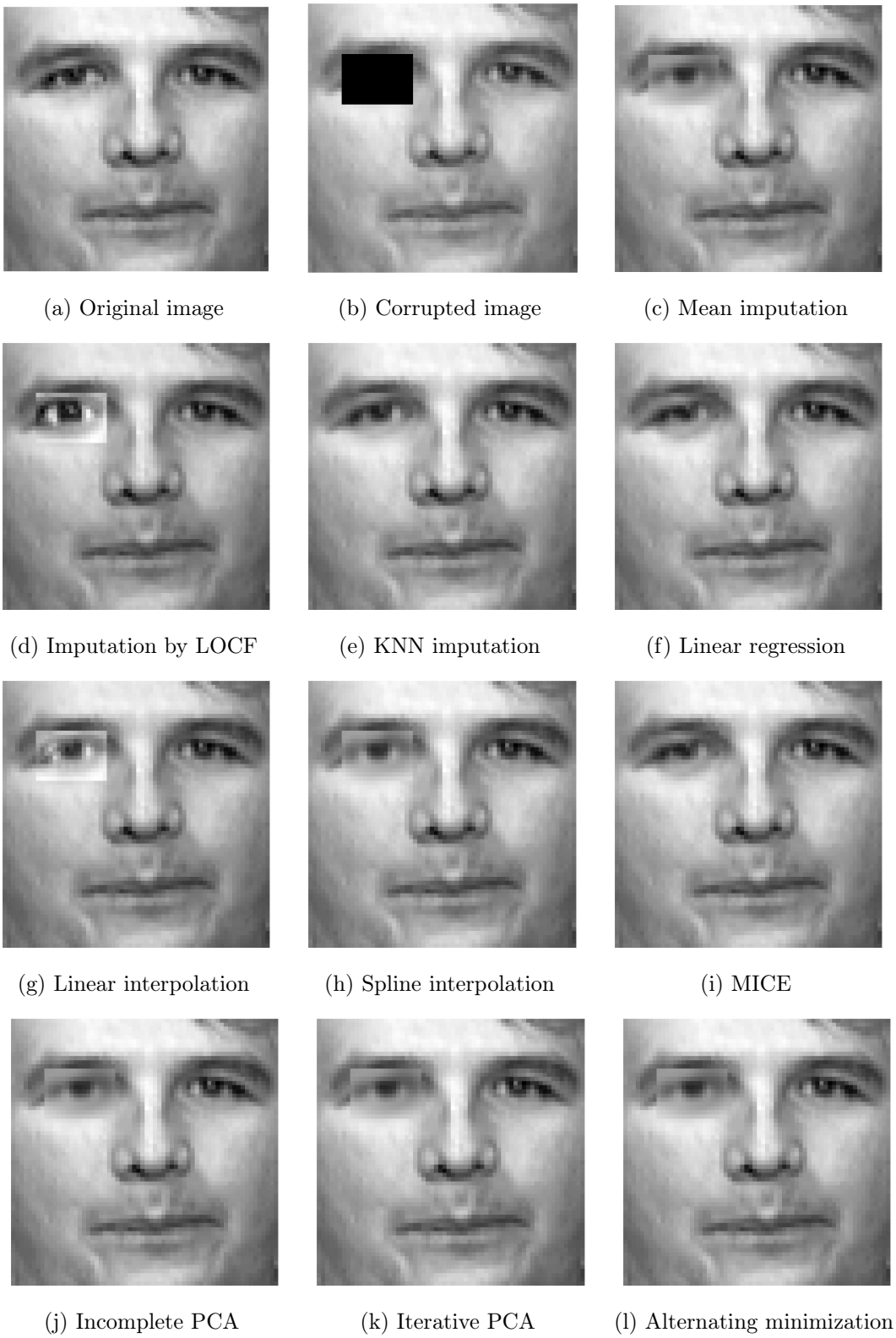
Figure 5.2: The first 12 eigenfaces of the Olivetti faces data set from AT&T.

For the first scenario of missing data, assume that there is a set of $n < N$ face images of size $d \times d$ that are complete, where N is the number of all face images. The complete face images are flattened and saved in the matrix of dimension $D \times n$, where $D = d^2$. The new face image $\tilde{X}_{new} \in \mathbb{R}^{d \times d}$ is flattened such that $X_{new} \in \mathbb{R}^D$ which includes artificially produced missing values. In the given example with the Olivetti data set, there is one selected image of missing data, such that $n = N - 1 = 399$. Assume further that the missing data is located in one region, such that the face image is known only up to the area of one eye. The first and second subfigure of figure 5.3 depict the original image and the incomplete image, respectively.

Starting with the statistical approach to missing data, different imputation methods introduced in section 2 can be applied to the entire data set of dimension $D \times N$ with missing data in the column of the previously chosen incomplete face image. Several illustrations of the results of the completion methods for one selected image can be found in figure 5.3. The results are heavily dependent on the axis chosen for the reconstruction. Since the images are all cropped and centered, all given methods impute values

that can be depicted as an eye in the face image for the correctly chosen axis. Here, the imputation is done along the row axis. A reconstruction along the column axis results in a reconstruction that would only depend on the given face image, which would lead to a grey reconstruction without structure. Simple imputation methods like mean and median imputation yield good results, since they can impute the mean or median of all eyes respectively, where the mean imputation can be seen in the third subfigure. Mode imputation does not impute the data such that an eye can be illustrated. The mode of each pixel is apparently distinct and as such not useful for the imputation of a missing eye in the image. Imputation by forward filling, depicted in the fourth subfigure, uses the area of the image of the previous person in the data set to fill the missing data points, hence the entries for the area around the eye of the previous person are imputed, which yields an eye which does not really fit the original. The same holds for NOCB. EM has similar problems as mode imputation, with the completed data being rather chaotic without a discernible structure. Using kNN to impute the data yields accurate results, where apparently neighboring values may be found which impute the data properly. Linear regression is also a good choice, while the interpolation results, with linear interpolation and spline interpolation depicted in the seventh and eighth subfigure, are not performing as precisely, despite the results being graphically acceptable, since they derive from the interpolation over values corresponding to the entries responsible for the eyes. The imputation given by spline interpolation does accomplish better results compared with the polynomial interpolations, that achieve similar imputations for varying degrees, where the linear interpolation results are depicted. MICE once again leads to an accurate and visually pleasing result. The problem in this case is that the data set is not very small and thus, the computation of the imputation takes much longer than for any of the other cases.

Moving on to the completion methods introduced in section 3.2, the dimensionality reduction methods are initially used on the complete subset of the data of dimension $D \times n$, which does not include the image with missing data. Given the eigenvectors of the data set with no missing data, computed by PCA, KPCA or DM, the weights for the linear combination of the eigenvectors, that the new face image can be represented with, can now be calculated and the incomplete face vector can then be reconstructed with them. As an example, the results given by incomplete PCA are depicted in figure 5.3. Similarly, iterative PCA may also be used with the corresponding results depicted in the second to last subfigure. The alternating minimization approach introduced in 3.3 is depicted in the last subfigure with results comparable to the other matrix completion methods using dimensionality reduction.



(a) Original image (b) Corrupted image (c) Mean imputation

(d) Imputation by LOCF (e) KNN imputation (f) Linear regression

(g) Linear interpolation (h) Spline interpolation (i) MICE

(j) Incomplete PCA (k) Iterative PCA (l) Alternating minimization

Figure 5.3: A selection of completion methods for a face image with missing data from AT&T's Olivetti faces data set.

Since many values in face images depend on the neighboring values, it is recommendable to smoothen the reconstructed values to obtain a more visually appealing image for the viewer instead of having a large discrepancy between the reconstructed entries and the correct entries. If an area of data is missing in the face image, then only the reconstructed boundary terms should be influenced. The interior of the reconstructed area should not be affected by the smoothing procedure. To this effect, a possible transition for the missing area is given by compensating with a weighted average that is taken with the correct neighboring values. If the missing entries are not constricted to one area, but scattered around the image, then all the missing values should be smoothened by a weighted average with the given neighboring values. However, a smoothing of the boundary entries has not been applied to the reconstructions in figure 5.3 to be better able to compare the completion methods.
Examining the performance criteria introduced at the beginning of the last chapter, the error of the completion methods restricted to the area of the missing data can be considered. It may be noticed that mode imputation, LOCF/NOCB, EM as well as polynomial interpolation yield relatively high L^1 and L^2-errors, while kNN, MICE and linear regression involve small errors and the completion methods via dimensionality reduction induce mediocre errors. The same holds for the total variation which was previously indicated as the important error measure for images, and RMSE as well as MAE. It is important to note that while most methods are able to impute the missing area in less than a second, linear regression as well as MICE take much longer, with MICE having a computation time of almost two minutes for completing the small area, which is more than 200 times the time needed for alternating minimization.

Another interesting perspective can be found by looking at the error restricted to the area of the missing data for each pixel. In figure 5.4, the errors of three selected completion methods for each pixel of the missing area are displayed. It holds that the darker the color the higher the absolute error of the reconstruction. All the completion methods of the previous figure may be characterized by one of the three error heat maps shown.
The first subfigure shows the absolute error for linear interpolation, which is similar to the pixel-wise error for polynomial interpolations of higher degree as well as LOCF and NOCB, while the second subfigure depicts the absolute error of MICE, similar to kNN and linear regression. In the last subfigure, the absolute error in the area of missing data between the original data and the reconstructed part for alternating minimization is depicted, with incomplete PCA and iterative PCA, mean and median imputation as well as spline interpolation having a similar error plot.

(a) Linear interpolation (b) MICE (c) Alternating minimization

Figure 5.4: Absolute error for each pixel between the original image and the reconstruction for three of the completion methods shown in the previous figure.

This can also be linked to the errors discussed above, with the first subfigure relating to the highest errors. The error for every pixel illustrates that the pixels in the upper left area of the missing area, as well as in the left part of the eye, differ more strongly than in the rest of the image. The second subfigure is relating to the lowest errors, while the methods with an error plot similar to the third subfigure lead to moderate errors, where the error is once again higher on the upper boundary of the missing area as well as in the area of the left part of the eye. The completion methods are apparently only able to reconstruct the larger underlying structure of the data and not every last detail. Especially the interpolation methods have an additional large error in the upper region of the image, which can also be deduced from the reconstruction.

Until now, the area of missing data was suitably chosen corresponding to the eye region, since the reconstruction is not trivial, but yields good result with the chosen methods. However, there are other possible ways of introducing missing data. Two different regions of missing data are illustrated in figure 5.5 with the corresponding completion methods, which were also chosen for the previous figure.
The first subfigure shows that removing the entries corresponding to one cheek in the face image also results in a good reconstruction for several methods. This is largely due to the fact that the image of a cheek does not have large variations in the pixel values. This result is hence not very meaningful, and a simple mean imputation would already be sufficient.
Another example, which usually does not obtain very good results for most methods, is the reconstruction of the nose area in a face image. Since the images in this data set are of low quality, a nose cannot be effectively reconstructed up to the last detail, as for example shown with a dimensionality reduction method in the last subfigure.

(a) Linear interpolation (b) MICE (c) Alternating minimization

ii) Missing cheek

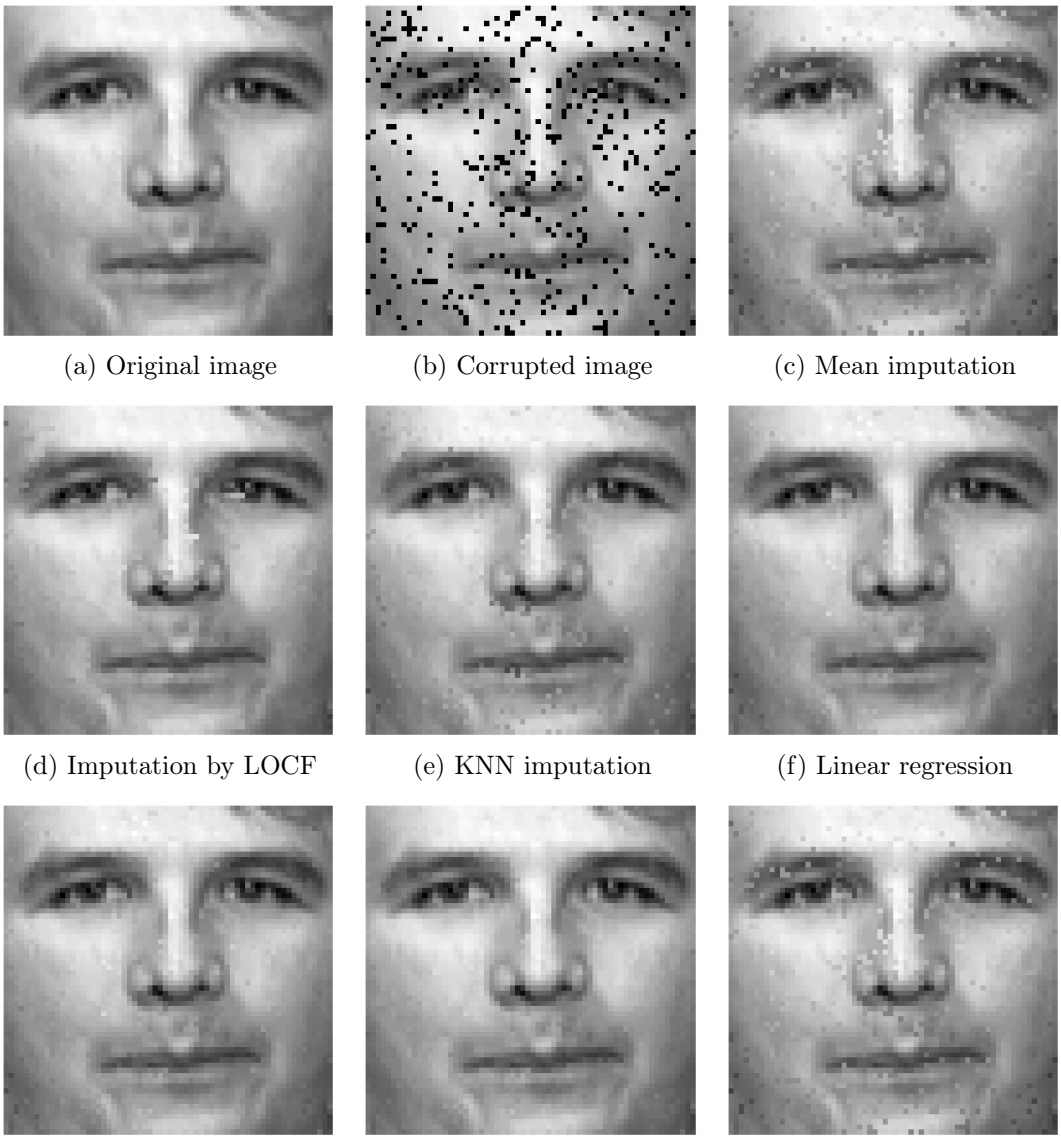
(d) Linear interpolation (e) MICE (f) Alternating minimization

i) Missing nose

Figure 5.5: Example for the reconstruction of a face image with different areas of missing data from AT&T's Olivetti faces data set.

For the last example of image completion, consider the second scenario where the entire data set is incomplete. As in the first example, the Olivetti data set is used. Since removing a specific feature for all face images cannot yield an accurate reconstruction of that feature, it is not a well-posed problem and should be avoided. Generating a percentage of random missing entries is however still a reasonable task for the entire data set. This is also a possibility for one image of missing data only as in the previous example, which yields similar results as the subsequent example and will therefore not be further discussed to avoid repetition. Since incomplete PCA, incomplete KPCA and incomplete DM use a previously given complete data set, it is not possible in this example to apply those methods. All other methods can be used for the completion of the random missing entries.

As fixed in the analogous example in the previous chapter, 10% of randomly missing data is artificially introduced. Once again, the first and second subfigure of figure 5.6 depict the original image and the incomplete image, respectively. Beginning with simple imputation methods, it is observable that results of an imputation by a central tendency are dependent on the axis on which the reconstruction takes place. The reconstruction based on the axis corresponding to the chosen image yields the same colour value for all missing entries, the mean or median of the image values in the specific image, and is hence not a good reconstruction choice. On the other hand, imputing missing values with the central tendency chosen, based on the axis corresponding to a specific pixel of all face images, leads to a slightly more accurate completion, which is illustrated for mean imputation in the third subfigure.
LOCF and NOCB yield results that do not have a color value close to the actual value due to being constrained to the images listed before and after the image shown. If the images are completed with the LOCF along the other axis, that is filling the values based on the neighboring value in the same image, the error reduces itself, but the results consist of color stripes as can be seen for imputation with the LOCF in the fourth subfigure.
Interpolation and regression results achieve similarly acceptable results. Higher polynomial interpolations and spline interpolation yield the same results and are therefore omitted from the figure.
This time, imputation using kNN is not one of the better methods, as the completed face image includes several outlier which can be seen in the corresponding subfigure.
In the given example, MICE produces the best results with a relatively low error, while matrix completion via alternating minimization is not quite as accurate or visually pleasing. Iterative PCA yields the same results in this example and are thus not portrayed.
An advantage of alternating minimization is the transposition invariance of the algorithm, that is to say in contrast to all other methods applied in this example, the alternating minimization approach yields the same reconstruction independent of the axis. Hence, the matrix of images can be transposed without changing the result. Furthermore, if one where to use the previously defined smoothing operation, the results would be much more visually pleasing.
With regard to the performance criteria, MICE causes the smallest errors, while taking the longest with an computation time of almost ten minutes compared to a few seconds for the other methods. Interestingly, in contrast to the constructed example in the previous chapter, all methods decrease the deviation of the data set or keep it at the same value, while the completion methods increased the MAD and STD for the example data



(a) Original image (b) Corrupted image (c) Mean imputation

(d) Imputation by LOCF (e) KNN imputation (f) Linear regression

(g) Linear interpolation (h) MICE (i) Alternating minimization

Figure 5.6: A selection of completion methods for a chosen image with 10% of randomly missing data from AT&T's Olivetti faces data set.

in the previous chapter. The variance of the data is hence decreased by the completion. To conclude, the first real-world application yields promising results for linear regression and MICE as well as in a limited capacity for kNN and especially matrix completion by alternating minimization, especially since it is much faster than the regression approach for the entire data set.

5.2 Shares

After looking at image completion in the previous section, the next real-world application is based on a bundle of time series in a financial setting. More specifically, given a share index, the daily closing prices of the individual shares are collected. [1] The price history of each share is then saved as a column in the matrix that includes all shares of the specific share index, such that for a specific point in time, the share prices of all shares in the index can be read off the row corresponding to the date.
The previously described completion methods will be subsequently used for missing entries in the share price indices which have been artificially introduced.

Two share indices are further discussed in the following paragraphs. The German share index **DAX (Deutscher Aktienindex)** is a well known share index in Germany, it includes 30 major German companies which are listed at the Frankfurt Stock Exchange. The **Dow Jones Industrial Average** similarly includes 30 large companies listed at stock exchanges in the United States, the New York Stock Exchange and the Nasdaq Stock Market. The companies that were included in the DAX and the Dow at the end of the year 2018 are listed in table 5.1.
Several possible reasons for a change in the company composition in an index exist, which depend on the market capitalization and stock market turnover of the already included companies and possible replacements. The composition has been modified since the end of 2018 and also during the year 2018. However, in the following paragraphs, the computations are done with the company composition given at the end of the year 2018 to be able to compare the results better. One company in each share index has only been given with incomplete data for 2018 and will therefore be excluded from further analysis for better comparison and study of the results.

In the following part, all individual share prices of the DAX and Dow companies during the year 2018 are each saved alphabetically in a matrix of dimension 251×29, where 251 is the number of banking days for which share price data is available and 29 is the number of shares in the index with complete data.
The share prices for the DAX companies in 2018 are depicted in figure 5.7. It can be seen that the share prices of almost all shares have not fluctuated much and that there were apparently no major events that impacted the entire market.

[1] share data downloaded with permission from ariva.de.

Table 5.1: The companies included in the DAX and Dow at the end of 2018

DAX	**Dow**
Adidas	3M Company
Allianz	American Express
BASF	Apple
Bayer	Boeing
Beiersdorf	Caterpillar
BMW	Chevron
Continental	Cisco
Covestro	Coca-Cola
Daimler	Dow Inc.*
Deutsche Bank	Exxon Mobil
Deutsche Börse	Goldman Sachs
Deutsche Post	The Home Depot
Deutsche Telekom	IBM
E.ON	Intel
Fresenius	Johnson& Johnson
Fresenius Medical Care	JPMorgan Chase
HeidelbergCement	McDonald's
Henkel	Merck&Co.
Infineon	Microsoft
Linde*	Nike
Lufthansa	Pfizer
Merck	Procter&Gamble
Munich Re	The Travelers Companies
RWE	UnitedHealth Group
SAP	United Technologies
Siemens	Verizon
ThyssenKrupp	Visa
Volkswagen	Walmart
Vonovia	Walgreens Boots Alliance
Wirecard	The Walt Disney Company

* shares are omitted in the following analysis

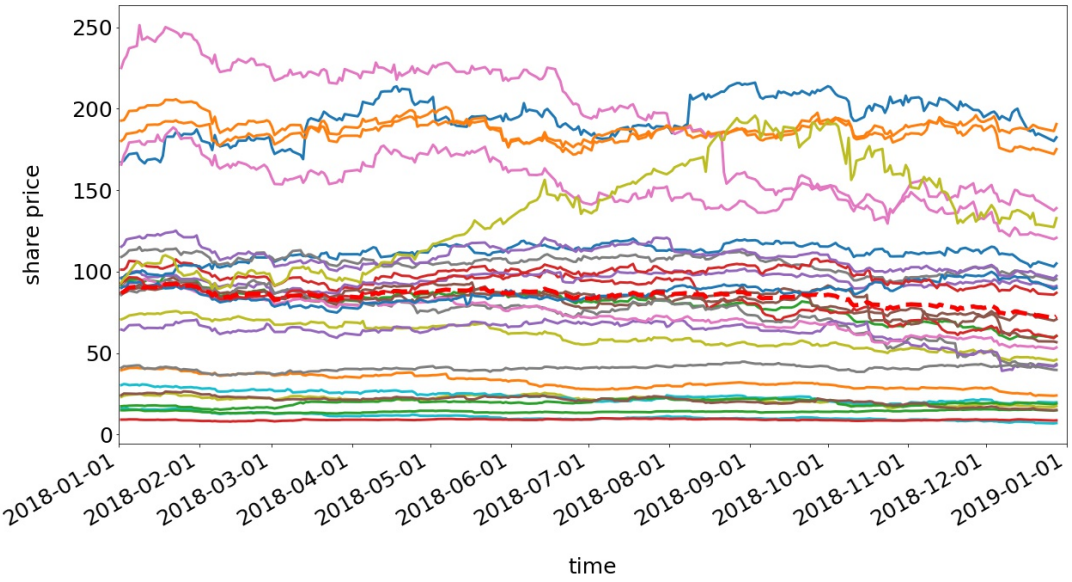


Figure 5.7: Individual prices for the shares included in the DAX for the year 2018 with the unweighted mean of all individual shares illustrated as the red dotted line.

However, the year 2018 was not really successful for many stocks in the DAX. On average, the DAX share prices declined during the year, such that the individual share prices decreased by approximately 16%, where the unweighted mean of the share prices is additionally illustrated for each banking day as a dotted red line in figure 5.7, which shows no peaks over time with a slightly decreasing trend.
If one were to look at the weighted share prices as they are used in the DAX, the decrease would be by approximately 18%. The DAX developed from a value of $12871,39$ at the beginning of the year to a value of $10558,96$ at the end of December 2018.
The share with the largest loss in the DAX in nominal values was the automotive company Continental with a decrease of more than 100 €. The share price started in January 2018 at 225 € and the end price had decreased to 120.75 € in December, which is a decline of more than 45%. The largest real loss was recorded for the Deutsche Bank share with a loss of 56% from a starting price of 15.96 € to an end price of 6.97 €. Only five DAX companies could observe an increase in their share price, with the top stock gainer Wirecard being the only DAX component with a two digit increase in 2018. Wirecard's share price increased from 93.28 € to 132.80 € by about 42%. The financial service provider had a high volatility during the year with a maximum price of 195.75 € on September, 3rd.

The situation in the US market was slightly better than in Germany, even though it was still not a lucrative year for more than half of the companies included in the Dow. On average, the share prices decreased by 5%, while the Dow lost about 6% with a value of 24824.01 in January and a closing level at the end of the year of 23327.46.
The largest decrease in nominal and real values was observed at the investment bank Goldman Sachs with a 34% decrease from a share price of 255$ to 167$. The highest percentage increase could be recorded for the pharmaceutical company Merck with an increase of more than 35% from 56$ to 76$, while the health care company UnitedHealth Group registered the highest nominal increase from 221$ to 249$.

Looking at the situation approximately ten years earlier, the individual share prices of companies in the Dow during the financial crisis in $2007 - 2009$ are an example where shares underwent large deviations. An interesting analysis can be conducted by looking at the correlation between the individual share prices. For that, the available individual share prices of the companies included in the Dow 2018 that are given on 454 working days between June 2007 and April 2009 are saved alphabetically in a matrix and the pairwise correlation of the values is computed. However, only 20 of the 30 companies now in the Dow were included at that time, such that the companies analyzed in the subsequent paragraphs were not all included in the Dow at that time.
The linear correlation coefficient for two company's share price vectors x, y with values on $n = 454$ working days is calculated as

$$\mathrm{Corr}(x,y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}},$$

where $\bar{x}, \bar{y}$ are the respective mean share prices and x_i, y_i are the share prices at time i. Linear correlation is a possible measure of a linear relationship between the two vectors with values between -1 and 1, where $\mathrm{Corr}(x,y) = 0$ implies that no linear correlation between x and y is present.
In figure 5.8, the correlation heat map for the companies during the financial crisis depicts that apart from two firms, all other company's share prices seem to have behaved similarly with a positive linear correlation. The two companies whose shares are negatively correlated to most of the other company's shares are McDonald's and Walmart. It may be concluded that during the financial crisis in $2007 - 2009$ most share prices fell, while the share prices for those two companies did not decrease.

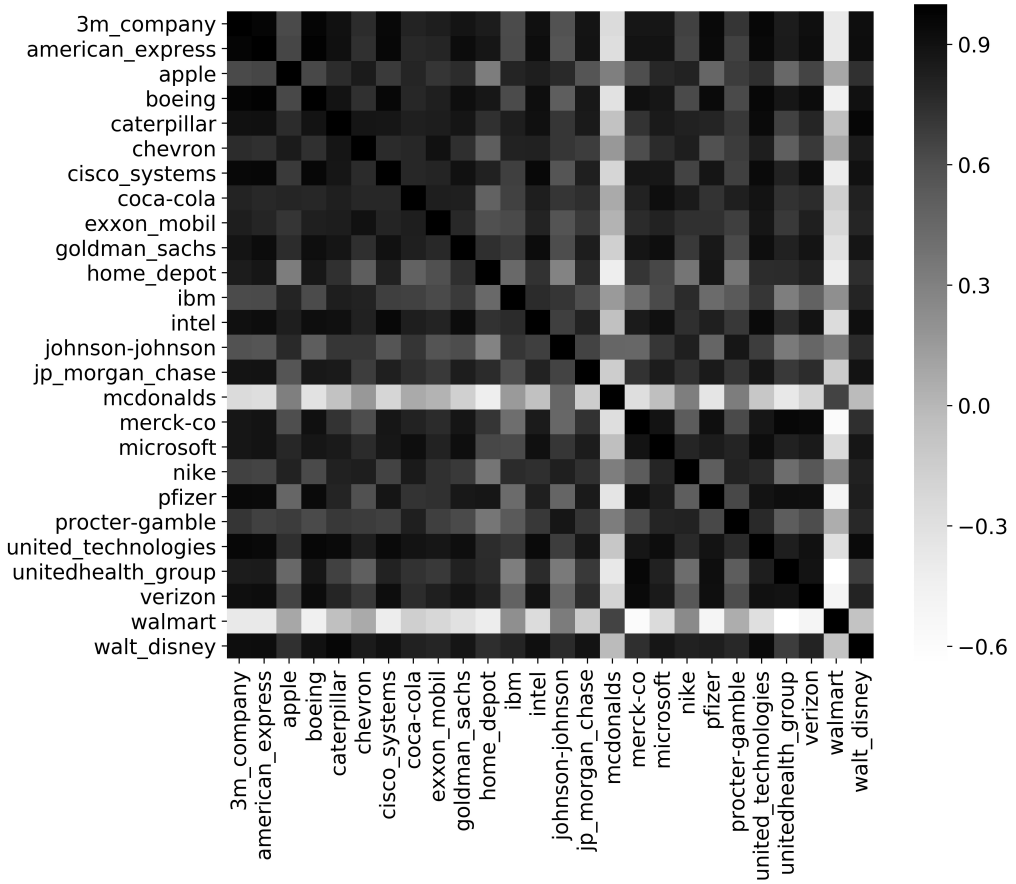


Figure 5.8: Correlation heat map of the individual share prices that are included in the Dow during the financial crisis 2007-2009.

Looking at the share prices during that time, as shown in figure 5.9, this assessment can be confirmed. McDonald's and Walmart did not loose their market value in the winter of 2008-2009 while the other company's share prices decreased. On average, the share prices did decrease during that time and the Dow index, which was heavily volatile during that winter, fell to a new low on March 09th, 2009. More precisely, between June 2007 and April 2009, on average the share prices decreased by about 47%, with American Express being the top loser with a decrease of 78%. Goldman Sachs had the highest nominal decrease of almost 120 points, from 230$ to 110$, which was a decrease of 52%. The only two companies with a share price increase were Walmart (+6.9%) and McDonald's (+8.5%). Therefore, it can be said, that between mid-2007 and mid-2009, during the height of the financial crisis, the Dow shares decreased significantly on average, while McDonald's and Walmart shares increased slightly.

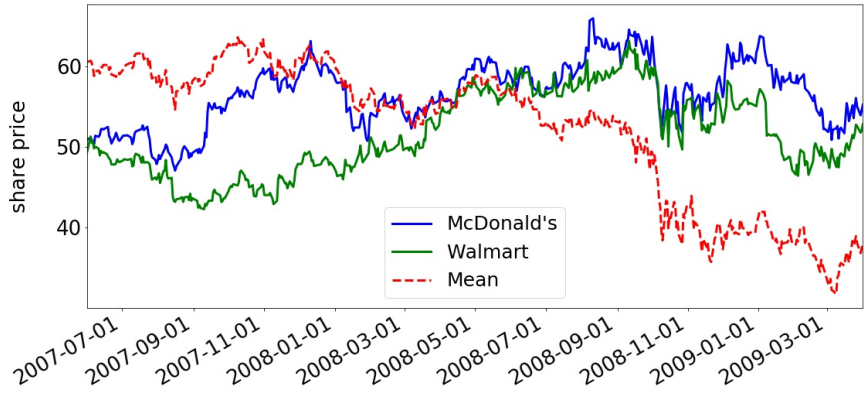
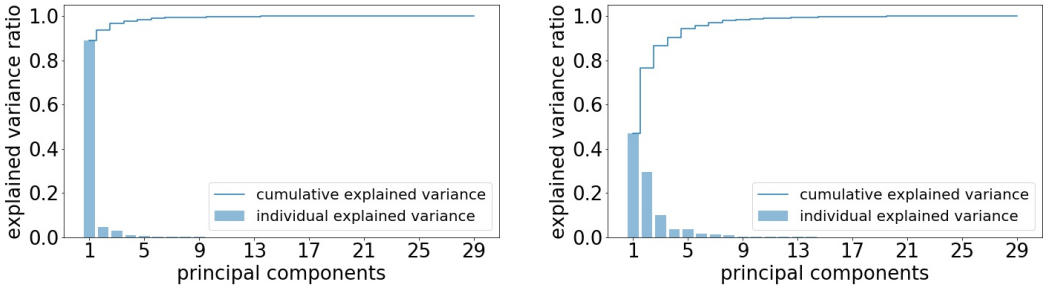


Figure 5.9: Share prices of the companies McDonald's and Walmart in comparison with an unweighted mean of all Dow share prices during the financial crisis 2007-2009.

Under more common circumstances, the behavior of share prices may be exceedingly diverse. During a less strenuous market situation, not all industry sectors are always impacted by a specific event. For instance considering the US market in 2018, pharmaceutical companies like Pfizer and Merck were highly positively correlated in 2018 with a linear correlation coefficient of 0.946, companies like Walmart and IBM (0.008) or Home Depot and Merck (-0.009) were pretty much uncorrelated and the two companies Goldman Sachs and Merck were highly negatively correlated with a linear correlation coefficient of -0.795.

The share price behavior impacts the ensuing analysis as well. Once again, the principal components and their explained variance ratio calculated with PCA may be evaluated. Looking at figure 5.10, it can be observed that due to the similar trend of most of the company's share prices, there are less principal components needed during the time of the financial crisis to explain the variance in the data than during the year 2018, where the share price development was not heavily influenced by a crisis and the behavior of the shares was more diverse. More detailed, looking at the share prices during the financial crisis in 2007-2009, the first principal component already includes almost 90% of the explained variance ratio, while 3 principal components are sufficient for more than 95% of the variance in the data. In contrast, for the data of the individual share prices of the Dow in 2018, the first principal component only explains about 47% of the data and 6 principal components are needed to explain 95% of the variance. The principal components and their explained variance ratio for the DAX in 2018 were depicted as the leading example for the introduction of PCA in section 3.1.1 with similar results.



(a) Dow 2007-2009 (b) Dow 2018

Figure 5.10: The principal components and their explained variance ratio of the individual share prices included in the Dow computed with PCA for two different time frames.

In the subsequent paragraphs, the data set is no longer complete as missing data is artificially produced.
In a first example, one share in the share index is selected and for a randomly chosen time frame, here chosen as one month, the data of the share price is removed. The second example once again includes a percentage of randomly missing data.
As previously explained in section 4.1, it may be noted that while an error norm is a possible criterion, it may not be the most fitting for share data. As the directional accuracy is refutably more significant, it is calculated for each method to check for the correctness of the reconstruction in this paragraph. Additionally, the nominal change in the data between the completed data and the actual data is compared for each time step with missing data.
Below, various share prices in different time frames are looked at for a diverse analysis. As before, the different described approaches for the completion of the missing values are examined. The completed part will be subsequently graphically represented in red.
For the first example, the individual share prices of the DAX companies during the year 2018 are used. The matrix of the share prices is of size 29×251, where 29 is the number of shares and 251 is the number of banking days. By removing information in one of the 29 shares randomly and using the data of the other 28 shares as the complete sub-matrix, data completion with the previously described methods can be achieved.
The results for one share with missing data pertaining to one specific month is depicted in figure 5.11, which shows a selection of completion methods for the Fresenius Medical Care share with missing information for April 2018, where the imputation is done along the axis corresponding to the individual share.

As a short side note, if one were to use the completion along the axis corresponding to the time instead of along the axis linked to the individual share, most methods would have an improved directional accuracy, especially the imputation methods using mean, median or mode as well as filling the values with the LOCF or the NOCB. However, the error would be much larger due to the reconstruction not being close to the original. This is explained by the share prices not being similar on average. For example, a simple forward filling uses the share price data of another company which does not necessarily have matching values.

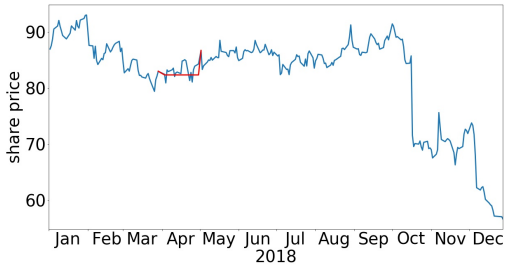
Now, for the completion along the axis corresponding to the share, the simple imputation methods, namely using the mean, median, mode, LOCF or NOCB, do not lead to acceptable results in relation to the directional accuracy, with directional accuracies around 10%, as the imputation is constant with time, which is exemplary illustrated for the mean imputation in 5.11a with the imputed results depicted in red. The outcome for kNN imputation is much better with a directional accuracy of almost 70%, see figure 5.11b, while EM entails very erratic values with a directional accuracy of around 45%.

Looking at interpolation with polynomial functions and splines, the results are inaccurate and the directional accuracy lies only between 50% and 60%, see subfigures 5.11c and 5.11d for linear interpolation and spline interpolation, respectively. Linear interpolation finds a linear interpolant between the last known share price and the next known share price and is as such also not close to the actual behavior of a share price. Higher polynomial interpolants achieve even worse results with imputed values not lying close to the actual values. Spline interpolation causes similar results as linear interpolation, even for splines of higher degree, such that it may be said, that the results for splines in this context are once again better than those for polynomials.

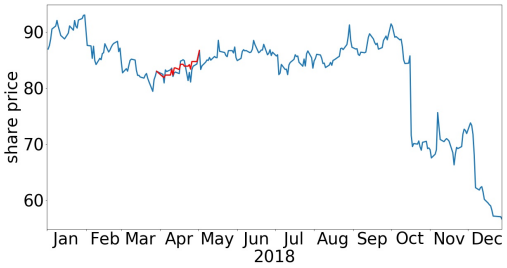
Linear Regression and MICE, shown in subfigures 5.11e and 5.11f respectively, involve a good completion with a directional accuracy of over 80%. The same may be said for the matrix completion approaches using dimensionality reduction. Incomplete PCA and iterative PCA as well as alternating minimization achieve a directional accuracy of over 85% in this case with the completion results for incomplete PCA and alternating minimization shown in 5.11g and 5.11h, respectively.

Overall, it can be seen that many of the results yield a very good outcome. The reconstructed part lies close to the original, where the completed part is colored in red.

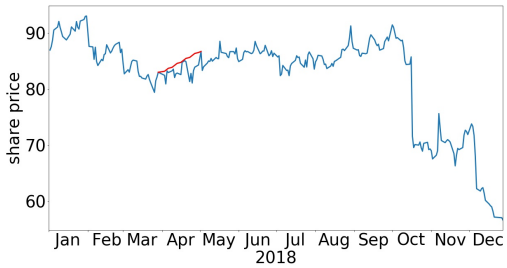
Conspicuously, almost all of the completion approaches yield results that achieve a directional accuracy that is better than 50%, which would equate to guessing the direction randomly. Only the simpler imputation methods as well as EM do no cause a directional accuracy higher than by guessing randomly.



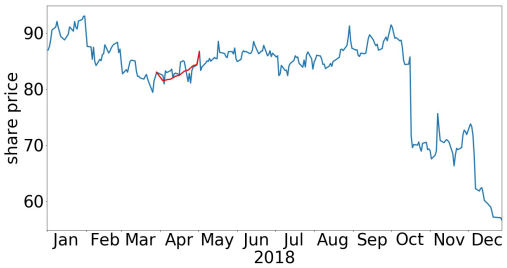
(a) Mean imputation



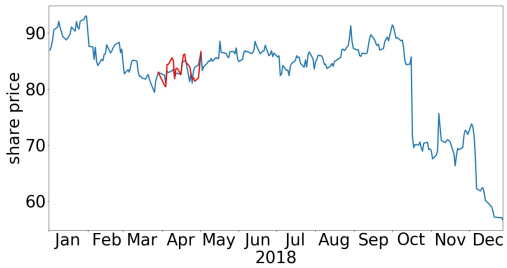
(b) KNN imputation



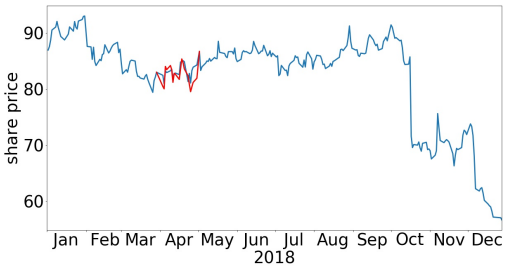
(c) Linear interpolation



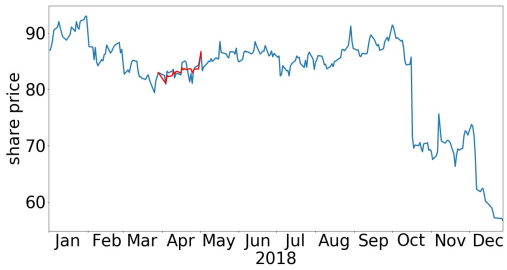
(d) Spline interpolation



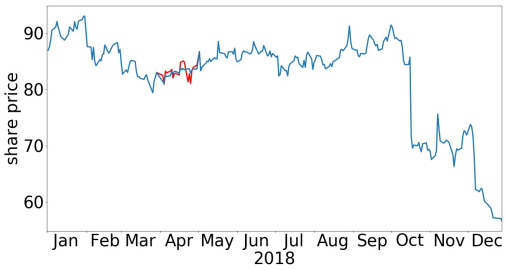
(e) Linear regression



(f) MICE



(g) Incomplete PCA

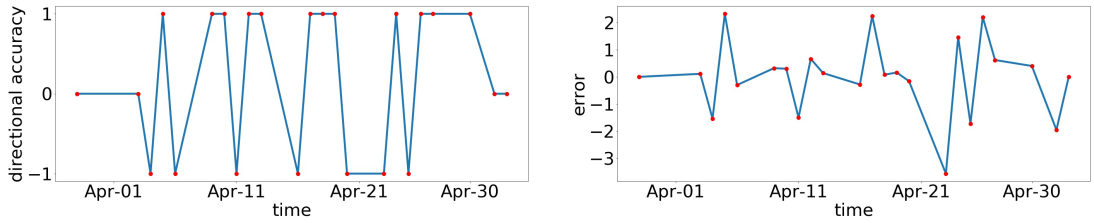


(h) Alternating minimization

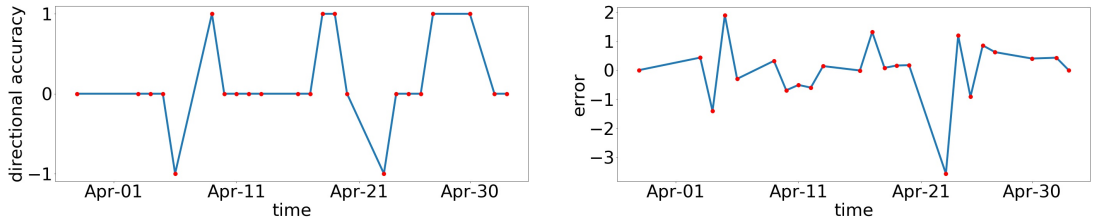
Figure 5.11: A selection of completion methods (in red) for a share price development with missing data corresponding to one month using the Fresenius Medical Care share price included in the DAX with missing information for April 2018 as an example.

The directional accuracy as well as the error for each value may be considered as well, for a diverse selection of the element-wise results of the missing area see figure 5.12.
For simple imputation methods like mean imputation, the directional accuracy is not very high with many peaks in both directions, while kNN imputation has much less peaks in both directions. Interestingly, while interpolation is never positive for negative trends, the opposite may be said for the regression approaches such as MICE. Matrix completion by alternating minimization has very few inaccuracies in this instance, in both directions.
As a final note on this first example, there are 348 combinations of 29 shares with 12 possible missing months ($29 * 12 = 348$). It is possible to average over the respective percentages representing the mean directional accuracies for the completion methods applied to all possible cases of having one individual share with a month of missing information as well as looking at the ratio of cases where the completion methods have achieved the highest directional accuracy compared with all other completion methods.
As explained above, it is easy to understand that all the simpler imputation methods, like using the mean, median, mode, LOCF or NOCB, have a mean directional accuracy of around $10 - 15\%$, which means that they are not useful for this specific application. The various interpolation approaches achieve a mean directional accuracy of $51\% - 56\%$, with the polynomials of higher degree having a lower directional accuracy and spline interpolation achieving the highest directional accuracy out of these methods. Imputation using kNN has a mean directional accuracy of 46% and the EM algorithm achieves 54%. Using matrix completion methods achieves a directional accuracy of 67% on average and the best directional accuracy in 36% of the cases, while the regression approaches with MICE achieve a slightly higher 71% directional accuracy with the best directional accuracy in 44% of the cases. Thus, MICE as well as the dimensionality reduction approaches achieve a mean directional accuracy of more than 50%. All other methods are hence not much more useful, or even less so, than randomly choosing a direction.

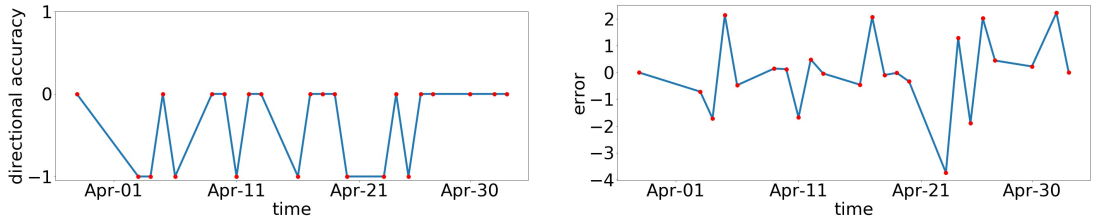
Now consider the individual share prices of the Dow companies during the same time period as the DAX, the matrix of the share prices has dimensions 29×251. Once again, one specific example of a share price with a time frame of one month of missing data is exemplary chosen and the completion results as well as the directional accuracies of the two previously well-performing methods MICE and alternating minimization may be analyzed. This time, the Exxon Mobile share with missing information in November 2018 is selected. The directional accuracy for MICE is 71%, while matrix completion by alternating minimization achieves an even higher percentage of 83%, which is the best directional accuracy compared with all other approaches.



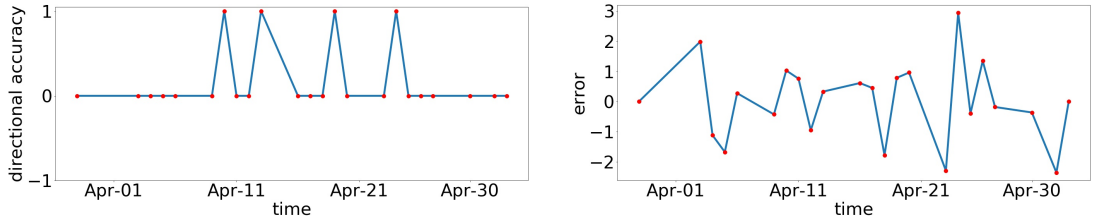
(a) Directional accuracy (left) and element-wise error (right) for mean imputation



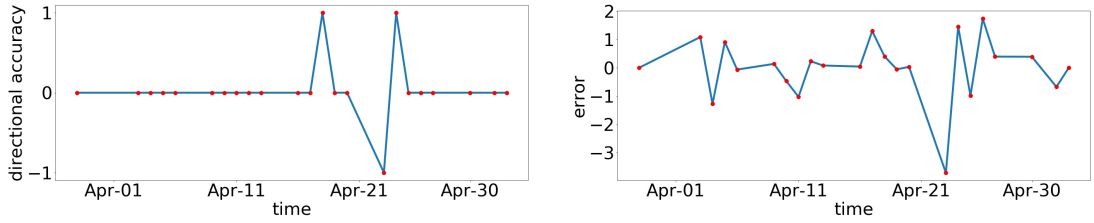
(b) Directional accuracy (left) and element-wise error (right) for kNN imputation



(c) Directional accuracy (left) and element-wise error (right) for linear interpolation

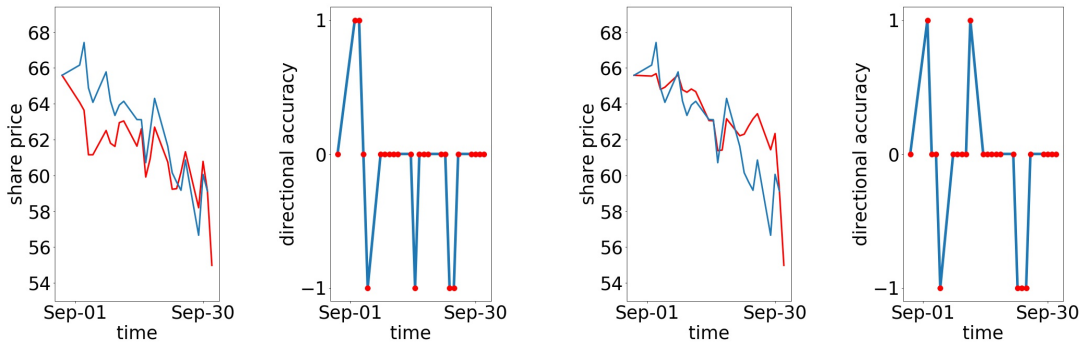


(d) Directional accuracy (left) and element-wise error (right) for MICE



(e) Directional accuracy (left) and element-wise error (right) for alternating minimization

Figure 5.12: The directional accuracy and the error of a selection of completion methods for the Fresenius Medical Care share price with missing information for April 2018.



(a) MICE (b) Alternating minimization

Figure 5.13: Completion results (in red) and the directional accuracy of two completion methods for the United Technologies share price with missing information for September 2008.

As the last example of a share index for a certain time frame in this section, the individual share prices of the Dow companies during the financial crisis 2007-2009 are examined. The time series in this case is longer than before.
The completion results as well as the directional accuracies for one share with a month of missing data are depicted in figure 5.13. The figure illustrates the completion results computed with the two approaches MICE and alternating minimization applied to the United Technologies share with missing information corresponding to September 2008 limited to the area of missing data with the completed share prices shown in red as well as the directional accuracies for the month of missing data on each banking day.
Both MICE and alternating minimization have a directional accuracy of 75%, such that both approaches have the best directional accuracy compared with all other completion methods, although the inaccuracies do not take place at the same time steps and do not necessarily occur in the same direction. In addition, the figures of the results illustrate that MICE underestimates the values for the first few days, while alternating minimization overestimates the last few values.

As in the DAX example, there are 348 possible combinations of missing data for the Dow in 2018. The time of the financial crisis analyzed here includes 22 months, such that the number of possible combinations increases to 572. The averages over all mean directional accuracies for the completion methods as well as the ratios of having the best directional accuracy compared with all other methods are displayed in table 5.2.

Method	**Mean directional accuracy**		**Best case ratio**	
	2018	2007-2009	2018	2007-2009
Mean	12.78%	13.02%	0%	0%
LOCF	15.19%	13.93%	2%	1%
KNN	52.12%	51.56%	2%	4%
EM	54.02%	54.41%	6%	4%
MICE	71.33%	73.51%	47%	54%
Linear Regression	70.47%	73.03%	38%	49%
Linear Interpolation	54.46%	56.98%	6%	6%
Spline Interpolation	56.14%	57.47%	9%	6%
Incomplete PCA	65.45%	70.52%	31%	39%
Iterative PCA	65.45%	70.52%	31%	39%
Alternating Minimization	65.45%	70.52%	31%	39%

Table 5.2: Averaged mean directional accuracy and best case ratio for several completion methods applied to all combinations of missing information for each month and each individual share listed in the Dow in 2018 and 2007-2009.

It is important to clarify that if the directional accuracy of two or more methods coincides, then all methods with the highest directional accuracy are counted as having the best directional accuracy. For that reason, the sum of the so-called best case ratio over all methods is not equal to 100%. The regression approaches achieve the best results, with 71% and 73.5% directional accuracy for MICE in 2018 and 2007-2009 respectively, closely followed by the dimensionality reduction approaches, where all methods achieve the same directional accuracies of 65% and 70.5% for the two time periods.
The percentages do not vary too much between the two time frames, it is however noticeable that the completion methods are on average more successful in 2007-2009. This is especially significant for the dimensionality reduction approach, due to the lower amount of dimensions needed for a high amount of explained variance. In approximately half of the possible cases, MICE is able to achieve the best directional accuracy, while the dimensionality reduction approaches are the best method in that respect for 31%/39% of the cases. All other methods can be disregarded for this application as the best directional accuracy is almost exclusively achieved by those two types of approaches.

Similarly to all the previous example applications, a percentage of missing data may also be artificially introduced. In the given case, 10% of the share prices are unknown as well as the share price on the day before and on the day after, such that for 10% of the banking days, the data for three subsequent days is missing.

Iterating 1000 times over 10% of randomly chosen missing data to improve on the quality of the analysis, different areas for missing data in the shares are introduced. For all three previously considered data sets, namely the individual share prices included in the DAX in 2018, in the Dow in 2018 and in the Dow during the financial crisis, the completion methods may be applied to the constructed incomplete data sets. Regarding the directional accuracies of the methods, the percentages do not really vary much ($\pm 1\%$) between the share indices and the choice of the data set is thus not relevant for the subsequent analysis. Since the missing areas in the time series are rather small, not many outlier are possible.
For example, the mean of the directional accuracy over all artificially produced 1000 cases for the Dow in 2007-2009 ranges between 71% and 87% for the completion methods, with the simple imputation methods using a central tendency and a forward/backward filling approach having the lowest at around 74% and 71%, respectively, the EM algorithm having a directional accuracy of 83% on average and the other methods using a nearest neighbors, regression, interpolation or dimensionality reduction approach ranging around $86\% - 87\%$.

To conclude, the two approaches MICE and alternating minimization once again achieve the best results out of the described methods. The directional accuracy is higher and it is also relevant that the reconstructed part usually lies close to the original data.

6. Conclusion

In this work, several different methods for completing missing data were analyzed.
As a first strategy, the statistical concept of imputation was utilized for dealing with incomplete data by replacing the missing values with a substitute. The single imputation methods were the simplest options, which include univariate methods like using the mean, median or mode of the incomplete data as substitutes. Imputation with the last observation carried forward or the next observation carried backward were also simple to apply. One type of multivariate simple imputation used regression, while the similar interpolation approach was another candidate. The last simple imputation method introduced was using the k-nearest neighbors algorithm. More advanced methods used multiple imputation, where missing values are imputed several times to account for the uncertainty in the data and the imputations. One of the state of the art methods in statistics is multiple imputation by chained equations, where a stochastic regression model is applied several times. Another course of action was a maximum likelihood based technique, the expectation-maximization algorithm.
After considering the statistical approach by imputing the values with a substitute, another approach with dimensionality reduction methods like PCA, the more general KPCA and the nonlinear manifold method diffusion maps was introduced, with the most advanced technique using alternating minimization for the completion of the data.

All the described methods were first applied to a simulated example, where a linear three-dimensional object was constructed and later noise was added to increase the similarity to a real-world application. All applications used two different types of missing data. In the first case, one large area of missing data was generated, such that a subset of the data columns or rows was still complete. For the second instance, a percentage of randomly missing data was artificially introduced. The completion strategies were applied to both cases and the results already indicated that the statistical approach with MICE and

the dimensionality reduction approach with alternating minimization were the two most reasonable choices out of the considered methods for completing the missing data. The choice of dimensionality reduction method made virtually no difference for the results, such that PCA lead to the same completion as KPCA or DM.
This assertion was substantiated by the two chosen real-world applications. The first example were face images taken from the Olivetti faces data set and the second example were the individual share prices included in the share indices DAX and Dow for a period of time. Thus, the two approaches MICE and alternating minimization were the most effective for the given data sets.
Nevertheless, it became clear that while MICE was usually the best choice for the completion based on the performance criteria applied, MICE is very slow for large data sets as the execution time increases very fast due to having to work with a regression model several times.

However it is important to note that there is no perfect way to compensate for missing values in a data set. Every method for completing missing values may perform better for certain data sets, but may not work well on other types of data sets. As addressed during the introduction of the performance criteria, there do exist special methods for images or time series based on specific error norms or concepts. However, the completion algorithms discussed here are applicable to a wide variety of problems. Even though these algorithms may not provide the best results for specific problems, they are a useful general tool for missing data.

The handling of missing data with statistical methods as well as matrix completion are an active field of research with many algorithms available and only a small collection of widely applicable and generally used techniques were evaluated.
To conclude, the method used for handling missing data should always be dependent on the data set itself as well as on the type of missing information. There have not been any universal rules developed for completing missing information and the task of choosing the most appropriate method lies with the user. Nonetheless, MICE for small data sets and alternating minimization for larger data sets are a good starting point, especially compared with removing incomplete data points or using the usually applied simple imputation methods.

Bibliography

[1] Jim Dziura, Lori Post, Qing (Amanda) Zhao, Zhixuan Fu, and Peter Peduzzi. Strategies for dealing with missing data in clinical trials: From design to analysis. *The Yale journal of biology and medicine*, 86:343–358, 2013.

[2] Akbar Waljee, Ashin Mukherjee, Amit Singal, Yiwei Zhang, Jeffrey Warren, Ulysses Balis, Jorge Marrero, Ji Zhu, and Peter Higgins. Comparison of imputation methods for missing laboratory data in medicine. *BMJ open*, 3, 2013.

[3] Gabriel Schlomer, Sheri Bauman, and Noel Card. Best practices for missing data management in counseling psychology. *Journal of counseling psychology*, 57:1–10, 2010.

[4] Chisimkwuo John, Emmanuel Ekpenyong, and Charles Nworu. Imputation of missing values in economic and financial time series data using five principal component analysis (pca) approaches. *Central Bank of Nigeria Journal of Applied Statistics*, pages 51–73, 2019.

[5] Caterina Penone, Ana Davidson, Kevin Shoemaker, Moreno Di Marco, Carlo Rondinini, Thomas Brooks, Bruce Young, Catherine Graham, and Gabriel Costa. Imputation of missing data in life-history trait datasets: Which approach performs the best? *Methods in Ecology and Evolution*, 5, 2014.

[6] Gláucia Ferrari and Vitor Ozaki. Missing data imputation of climate datasets: Implications to modeling extreme drought events. *Revista Brasileira de Meteorologia*, 29:21–28, 2014.

[7] Donald B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley, 1987.

[8] Roderick J A Little and Donald B Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, Inc., New York, NY, USA, 1986.

[9] Stef Van Buuren. *Flexible Imputation of Missing Data, Second Edition.* CRC Press, Taylor & Francis Group, 2018.

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[11] BRETT BEAULIEU-JONES and Jason Moore. Missing data imputation in the electronic health record using deeply learned autoencoders. volume 22, pages 207–218, 2017.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Y. Bengio. Generative adversarial nets. 2014.

[13] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. Gain: Missing data imputation using generative adversarial nets. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5689–5698, 2018.

[14] Dongwook Lee, Junyoung Kim, Won-Jin Moon, and Jong Chul Ye. Collagan : Collaborative gan for missing image data imputation. *CoRR*, 2019.

[15] Yonghong Luo, Xiangrui Cai, Ying ZHANG, Jun Xu, and Yuan xiaojie. Multivariate time series imputation with generative adversarial networks. In *Advances in Neural Information Processing Systems 31*, pages 1596–1607. Curran Associates, Inc., 2018.

[16] Francesco Ricci, Lior Rokach, and Bracha Shapira. *Recommender Systems Handbook*, volume 1-35, pages 1–35. 2010.

[17] Prem Melville and Vikas Sindhwani. *Recommender Systems*, pages 1056–1066. Springer US, 2017.

[18] Robert M. Bell, Yehuda Koren, and Chris Volinsky. The bellkor solution to the netflix prize, 2008.

[19] Antonin Chambolle, Vicent Caselles, Matteo Novaga, Daniel Cremers, and Thomas Pock. An introduction to total variation for image analysis. 2010.

[20] L I Rudin. *Images, numerical analysis of singularities and shock filters.* PhD thesis, 1987.

[21] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259 – 268, 1992.

[22] Samuel Arcadinho and Paulo Mateus. Time series imputation. *ArXiv*, abs/1903.09732, 2019.

[23] James Honaker and Gary King. What to do about missing values in time series cross-section data. *American Journal of Political Science*, 54(3):561–581, 2010.

[24] James Honaker, Gary King, and Matthew Blackwell. Amelia ii: A program for missing data. *Journal of Statistical Software, Articles*, 45(7):1–47, 2011.

[25] C.F. Gauss. *Theoria combinationis observationum erroribus minimis obnoxiae*. Commentationes Societatis Regiae Scientiarum Gottingensis recentiores. H. Dieterich, 1823.

[26] Ashish Sen and Muni Srivastava. *Regression Analysis: Theory, Methods and Applications*. Springer Science+Business Media New York, 1990.

[27] Raimer Kress. *Numerical Analysis*. Graduate Texts in Mathematics. Springer Science+Business Media New York, 1998.

[28] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical Mathematics*. Texts in Applied Mathematics. Springer, Berlin, Heidelberg, 2007.

[29] John Graham, Allison Olchowski, and Tamika Gilreath. How many imputations are really needed? Some practical clarifications of multiple imputation theory. *Prevention science: the official journal of the Society for Prevention Research*, 8:206–213, 2007.

[30] M. J. Azur, E. Stuart, C. Frangakis, and P. Leaf. Multiple imputation by chained equations: What is it and how does it work? *International Journal of Methods in Psychiatric Research*, 20:40–49, 2011.

[31] Martin A. Tanner and Wing Hung Wong. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82(398):528–540, 1987.

[32] J.L. Schafer. *Analysis of Incomplete Multivariate Data*. CRC Press, Chapman and Hall, 1997.

[33] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[34] René Vidal, Yi Ma, and S. Shankar Sastry. *Generalized Principal Component Analysis*. Springer Science+Business Media New York, 2016.

[35] Simon Haykin. *Neural Networks and Learning Machines*. Pearson Education Inc., 2009.

[36] Bernhard Schölkopf, Alexander J. Smola, and Klaus-Robert Müller. In *Advances in Kernel Methods - Support Vector Learning*, chapter Kernel Principal Component Analysis, pages 327–352. MIT Press, Cambridge, MA, USA, 1999.

[37] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences of the United States of America*, 102:7426–7431, 2005.

[38] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–86, 1991.

[39] Julie Josse, J. Pagès, and Francois Husson. Multiple imputation in principal component analysis. *Adv. Data Analysis and Classification*, 5:231–246, 10 2011.

[40] Emmanuel Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Communications of the ACM*, 9:717–772, 11 2008.

[41] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. *Proceedings of the Annual ACM Symposium on Theory of Computing*, 12 2012.

[42] Schmitt J. A comparison of six methods for missing data imputation. *Journal of Biometrics & Biostatistics*, 06, 2015.

[43] Pascal Getreuer. Rudin-osher-fatemi total variation denoising using split bregman. *Image Processing On Line*, 2:74–95, 2012.

[44] Jingyue Wang and Bradley Lucier. Error bounds for finite-difference methods for rudin-osher-fatemi image smoothing. *SIAM J. Numerical Analysis*, 49:845–868, 2011.

[45] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[46] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *2nd IEEE Workshop on Applications of Computer Vision*, 1994.