# Clustering Approaches for Data with Missing Values: Comparison and Evaluation

Ludmila Himmelspach
Institute of Computer Science
Heinrich-Heine-Universität Düsseldorf
D – 40225 Düsseldorf, Germany
Email: himmelspach@cs.uni-duesseldorf.de

Stefan Conrad
Institute of Computer Science
Heinrich-Heine-Universität Düsseldorf
D – 40225 Düsseldorf, Germany
Email: conrad@cs.uni-duesseldorf.de

***Abstract*—Traditional clustering methods were developed to analyse complete data sets. Faults during the data collection, data transfer or data cleaning often lead to missing values in data so that common clustering methods can not be used for the data analysis. Therefore, in these cases clustering methods which can handle missing values in data are of great use. In this paper we discuss different approaches proposed in the literature for adapting partitioning clustering algorithms for dealing with missing values in data. We analyse them on two appropriate data sets and compare them with each other. We give particular attention to the analysis of the accuracy of these methods depending on the different missing-data mechanisms and the percentage of missing values in the data sets.**

## I. Introduction

With the rapid rise of possibilities to collect and to store large amounts of data electronically tools for the data analysis also have gained increasingly in importance. These magnitude of data can contain a lot of potentially important knowledge which, however, must be firstly extracted from data within the scope of a data mining process. Clustering represents an important technique for knowledge extraction from data. Its task is to identify groups of similar objects within a data set. Data clustering is used in many areas, including database marketing, web analysis, information retrieval, bioinformatics, and others.

Data objects to be clustered are represented by a definite number of feature values, which are available in form of a data matrix for the analysis. However, missing values which could be caused for example by faults during the data collection, data transfer or data cleaning, often occur in data. Missing values can be arranged randomly or occur according to certain patterns in the data matrices. They can be missing at random or depending on certain factors concerning the mechanisms that lead to missing values.

Traditional clustering methods were developed to analyse complete data sets. Therefore, clustering methods which can handle with missing values in data sets are of great importance. In the literature there already are some proposals for partitioning clustering algorithms handling incomplete data. However, the evaluation of these methods only confined to general cases or to comparison with very basic approaches, which based either on imputation of missing values or marginalisation of features or data items containing missing values before the cluster analysis. In this work we aim to give a deeper insight into clustering incomplete data. Therefore, we discuss different clustering methods for dealing with missing values in data. We analyse them on two appropriate data sets and compare them with each other. In order to demonstrate the performance of clustering methods on different types of incomplete data, we analyse the accuracy of these methods depending on the different missing-data mechanisms and the percentage of missing values in the data sets.

The remainder of the paper is organised as follows. In Section II we give an overview of partitioning clustering methods and missing-data mechanisms. In Section III we describe different approaches for clustering incomplete data sets. We present the evaluation results and compare the methods in Section IV. We close this paper with a short summary and discuss future works in Section V.

## II. Related Work

### A. Clustering Algorithms

Clustering is an important technique for the automatic partitioning of large amounts of data into groups or clusters. The objects within one cluster are to be as similar as possible while the objects from different clusters are to be as dissimilar as possible [5]. Traditional clustering methods can generally be subdivided into three categories: partitioning, hierarchical and density-based methods. While partitioning clustering methods aim for a simple flat decomposition of data objects into a given number of clusters, the hierarchical clustering methods produce a hierarchical representation of the data set. Density-based clustering methods identify clusters as regions of objects lying closely together which are separated by regions in which objects lie less closely together. In this paper we focus our consideration to the partitioning clustering algorithms.

*1) k-Means Algorithm:* The most known partitioning clustering method is the *k*-means algorithm [10]. The goal of the *k*-means algorithm is to find an optimal partitioning of $n$ data points $X = \{x_1, ..., x_n\}$ in $d$-dimensional metric data space into $k$ clusters. Each cluster $C$ is represented by its centroid $\mu_C$ which is calculated as an arithmetic mean of all data objects within the cluster $C$. As measure for the compactness of a clustering the squared error function is typically used which

978-1-4244-7571-1/10/$26.00 ©2010 IEEE

is defined as

$$TD^2 = \sum_{i=1}^{k} \sum_{x \in C_i} dist(x, \mu_{C_i})^2 \quad (1)$$

*k*-means determines an optimal partitioning of a data set by minimising the objective function given in Formula 1 in an iterative process. The algorithm begins with initialisation of cluster centroids, which are randomly chosen points in the data space. In the first iteration step each data object is assigned to cluster to which it is most similar. The distance between data object and cluster means is used as the similarity measure. In the second iteration step the new means are computed for each cluster. The last two steps must be iterated as long as cluster means change.

*2) Fuzzy c-Means Algorithm (FCM):* The *k*-means algorithm assigns each data object to exactly one cluster. Thus the information about the structure of a clustering gets lost although it is sometimes of note especially if a data item can be assigned to more than one cluster. To overcome this drawback the fuzzy c-means algorithm (FCM) assigns each data point $x_k$ to each cluster $C_i$ with a membership degree $u_{ik} \in [0, 1]$, which expresses the relative degree to which $x_k$ belongs to the cluster $C_i$. According to [1] the membership degree is calculated as follows:

$$u_{ik} = \left(D_{ik}^{1/(1-m)}\right) / \left(\sum_{j=1}^{c} D_{jk}^{1/(1-m)}\right) \quad (2)$$

for $1 \leq k \leq n$ and $1 \leq i \leq c$, where $m > 1$ is the fuzzification parameter, and $D_{ik} = \|x_k - \mu_{C_i}\|_A^2$ is the squared vector $A$-norm distance between data point $x_k$ and cluster prototype $\mu_{C_i}$ (in case $A = I_{d \times d}$, $\|x\|_A = \|x\|_2$ is the Euclidean norm).

Unlike the *k*-means algorithm the cluster centroids are calculated based on all data points depending on their membership degree to the cluster:

$$\mu_{ij} = \left(\sum_{k=1}^{n} (u_{ik})^m x_{kj}\right) / \left(\sum_{k=1}^{n} (u_{ik})^m\right) \quad (3)$$

for $1 \leq i \leq c$ and $1 \leq j \leq d$.

FCM works in the same way as the *k*-means algorithm. Just instead of assigning data objects to the clusters in the first iteration step the membership degrees are calculated. The new cluster prototypes are computed according to Formula 3. The iterative process continues as long as the cluster centroids change only up to a value $\epsilon$. In this way the objective function given in Equation 4 is minimised in each iteration step. For more details see also [1].

$$J_m(U, \mu) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m \cdot D_{ik}. \quad (4)$$

### B. Data with Missing Values

As missing values can induce a random or conditional reduction of a data set, the performance of clustering algorithms for incomplete data depends on the mechanism that leads to missing data. The missing-data mechanisms refer to the relationship between missingness and the underlying feature values in the data matrix [8]. In general, there are three different types of missing-data mechanisms *MCAR*, *MAR* and *NMAR*. The missing values are called *missing completely at random (MCAR)*, if the missingness does not depend on the data values in the data matrix independent whether they are missing or observed. The missing values are denoted as *missing at random (MAR)*, if the missingness depends only on the values in the data matrix that are observed, and not on the components that are missing. If the missingness of data depends on the missing values themselves in the data matrix then the missing-data mechanism is called *not missing at random (NMAR)*. Although missing values MCAR represent the general case of missingness, missing values MAR and NMAR are also common in practise. For example, on questionnaires the question for income remain often unanswered by young people, thus missing values in feature income are MAR, because they depend on values in attribute age. Also high-income earners do not often disclose their income, in this case missing values in feature income are NMAR, because they depend on the values themselves.

In many cases the missing-data mechanisms are not known in advance. Nevertheless, they can be verified via suitable statistical test procedures such as one-sample tests for missing-data mechanism NMAR, two-sample tests for missing-data mechanism MAR or Little's MCAR test to verify the missing-data mechanism MCAR [9]. For more details see also [2], [8].

## III. Different Approaches for Clustering Incomplete Data

### A. k-Means Clustering with Soft Constraints

In [13] a *k*-means clustering method for data with missing values is presented which is based on so-called *Soft Constraints*. The idea of this approach is to define the soft constraints on features with missing values and to use these as additional information. The feature set $F_1, ..., F_d$ is divided into the set of completely observed features $F_{obs}$ and the set of features with missing values $F_{mis}$. A soft constraint between two complete data points $x_i$ and $x_j$ for $i, j \in \{1, ..., n\}$ is defined as a triple: $\langle x_i, x_j, s \rangle$, where

$$s = -\sqrt{\sum_{k \in F_{mis}} (x_{ik} - x_{jk})^2}. \quad (5)$$

The *strength* $s$ is proportional to distance in $F_{mis}$ and specifies the degree to which the data points $x_i$ and $x_j$ should be separated. Note that soft constraints are defined only between complete data items, for data items with missing values no constraints will be created.

The *k*-means clustering algorithm with soft constraints (KSC) classifies data set $X$ on the basis of feature values of set $F_{obs}$. The feature values of set $F_{mis}$ only provide a basis for defining soft constraints. The set of all defined soft constraints ($SC$) is passed to the algorithm as additional parameter at the beginning. KSC works in the same way as *k*-means algorithm (cf. Section II-A1). KSC differs from the

basic $k$-means algorithm in the first iteration step only in the assignment of data points to clusters. KSC assigns data points according to the Formula 6.

$$C := \underset{C_i}{\operatorname{argmin}} \left( (1-w)\frac{dist(x,\mu_{C_i})^2}{V_{max}} + w\frac{CV_x}{CV_{max}} \right) \quad (6)$$

The distance between a data point $x$ and a cluster centre $\mu_{C_i}$ is composed of a weighted sum of two values. The first one is the squared distance between data point and cluster centre normalised by the variance $V_{max}$ of all data points in data set $X$. The second value $CV_x$ corresponds to the sum of squared strengths $s$ of violated soft constraints which include the data point $x$. A soft constraint $\langle x_i, x_j, s\rangle$ is considered as violated if data points $x_i$ and $x_j$ are assigned to a one cluster although they belong to different clusters [13]. And the squared value of $s$ is the penalty for the false assignment. The value $CV_x$ is normalised by value $CV_{max}$, the sum of squared strengths $s$ of all soft constraints in $SC$ independently, whether they are violated or not. The relative importance of the constraints versus normalised distance to cluster prototypes is indicated by a weighting factor $w \in [0,1]$.

The distance between data items with missing values and cluster centres results only from the standardised squared distances between data points and cluster centres. The value $CV_x$ is zero because no soft constraints are defined for data items with missing values. For complete data items the distance to cluster centres additionally depends on the distance to data items in $F_{mis}$ which were assigned to the same cluster. The more such items exist and the farther they are from the data point, the bigger the distance is to the cluster centre.

### B. $k$-Means Clustering with Partial Distance Strategy

Another possibility to adapt the $k$-means algorithm for dealing with missing values is to use a partial distance function instead of Euclidean distance function during the calculation of similarity between two data points. This strategy is denoted as a partial distance strategy (PDS). It was already used in [3] in the context of the k-nearest neighbour search. Hathaway and Bezdek used it for fuzzy clustering of incomplete data [6]. We adopt this strategy to the $k$-means clustering algorithm for data with missing values. The corresponding algorithm is denoted here as *Partial Distance Strategy $k$-Means Algorithm* (PDSKMeans).

According to [12] the partial distance between two data points $x$ and $y$ in a $d$-dimensional Euclidean vector space is calculated as follows

$$D_{part}(x,y) = \frac{d}{d - \sum_{i=1}^{d} b_i} \sum_{\forall i: b_i = 0} (x_i - y_i)^2, \quad (7)$$

where

$$b_i = \begin{cases} 0, & if\ x_i\ and\ y_i\ are\ available \\ 1 & else \end{cases}$$

The partial distance function calculates the squared Euclidean distance between all available feature values of data points and standardises it by the reciprocal of the proportion of values used during the calculation. If for two data items all feature values are available, the partial distance function calculates the squared Euclidean distance between them.

Using the partial distance function the $k$-means algorithm can be simply adapted for incomplete data set. The basic $k$-means algorithm must be modified only in the first iteration step during the assignment of data items to clusters and during the calculation of cluster centres in the second step. PDSKMeans assigns data items to clusters as follows:

$$C := \underset{C_i}{\operatorname{argmin}} \left( D_{part}(x, \mu_{C_i}) \right) \quad (8)$$

Since data points within a cluster can have missing values, the coordinates of the cluster centroid are calculated as the arithmetic means of all available feature values of data points within the cluster according to Formula 9.

$$\mu_j(C_i) = \frac{1}{\sum_{x \in C_i} m_j} \sum_{\substack{\forall j: m_j = 1 \\ x \in C_i}} x_j \quad (9)$$

for $1 \le i \le k$ and $1 \le j \le d$ with

$$m_j = \begin{cases} 1, & if\ x_j\ available \\ 0 & else \end{cases}$$

In contrast to KSC from the previous section this method includes all available feature values during clustering so that the whole information about data points with missing values can be used in a meaningful way.

### C. Whole-Data Strategy (WDS)

A simple method to adapt the fuzzy c-means algorithm for handling data with missing values is the *whole-data strategy* (WDS) [6]. As the name already indicates only complete data items are regarded during clustering. Initially data items with missing values are deleted from the data set. Then complete data items are clustered via basic fuzzy c-means algorithm (cf. Section II-A2). As data items with missing values are not considered by the algorithm, they are not firstly assigned to any cluster. Finally incomplete data items are assigned to the nearest cluster centre by calculating the partial distances (cf. Formula 7).

### D. Partial Distance Strategy (PDS)

The second approach to adapt the fuzzy c-means algorithm for handling incomplete data is the *partial distance strategy* (PDS) [6]. This approach is based on the calculation of partial distances between data items with missing values and is referred to as *Partial Distance Strategy Fuzzy C-Means Algorithm* (PDSFCM).

The fuzzy c-means algorithm can be adapted to the partial distance strategy in a straightforward way. During the calculation of membership degrees of data items to clusters, the squared distance function must be replaced by the partial distance function. The cluster prototypes in the second iteration step of the algorithm are calculated then only on the basis of

available values of data items within a cluster (see Formula 10).

$$\mu_{ij} = (\sum_{k=1}^{n}(u_{ik})^m I_{kj} x_{kj})/(\sum_{k=1}^{n}(u_{ik})^m I_{kj}) \qquad (10)$$

for $1 \leq i \leq c$ and $1 \leq j \leq d$ with

$$I_{kj} = \begin{cases} 1, & if\ x_{kj}\ available \\ 0, & if\ x_{kj}\ missing \end{cases}$$

The advantage of this approach, in contrast to the whole-data strategy is that it can be used even if all data items have missing values.

### E. Optimal Completion Strategy (OCS)

The idea of another approach to adapt the fuzzy c-means algorithm for handling incomplete data is to estimate missing values depending on all cluster centres in every iteration step [6]. This method is referred to as *optimal completion strategy* (OCS) and algorithm changed in this way is referred to as *Optimal Completion Strategy Fuzzy C-Means Algorithm* (OCSFCM). The fuzzy c-means algorithm is changed by adding an additional third iteration step. At the beginning of the algorithm, the missing values in the data matrix are replaced by random values. The calculation of membership degrees in the first iteration step and the cluster centres in the second step works in the same way as in the FCM. The available and estimated values in the data matrix are not distinguished. In a third iteration step missing values are estimated depending on all cluster prototypes as follows:

$$x_{kj} = (\sum_{i=1}^{c}(u_{ik})^m \mu_{ij})/(\sum_{i=1}^{c}(u_{ik})^m) \qquad (11)$$

for $1 \leq k \leq n$ and $1 \leq j \leq d$.

As one can see in the formula, the cluster prototypes to which the data point shows higher membership have more influence during the calculation of the missing feature values of the data point.

### F. Nearest Prototype Strategy (NPS)

The *nearest prototype strategy* (NPS) is a modification of OCSFCM. The missing values of an incomplete data item are completely substituted by the corresponding values of cluster prototype to which the data item has the smallest partial distance [6]. The resulting algorithm is referred to as *Nearest Prototype Strategy Fuzzy C-Means Algorithm* (NPSFCM) and results from the OCSFCM by changing the third iteration step. Thus the missing values of an incomplete data item are calculated as follows:

$$x_{kj} = \mu_{ij} \quad \text{with } D_{ik} = min\{D_{part_{1k}}, ..., D_{part_{ck}}\} \qquad (12)$$

for $1 \leq k \leq n$ and $1 \leq j \leq d$.

If a data point with missing values has the minimal partial distance to more than one cluster, the choice of the nearest cluster prototype depends on the implementation of the $min$-function.

### G. Distance Estimation Strategy (DES)

The last approach discussed in this paper to adapt FCM for dealing with missing values is based on the estimation of distances between cluster prototypes and incomplete data items [11]. We refer to this method here as *Distance Estimation Strategy* (DES) and the corresponding algorithm as *Distance Estimation Strategy Fuzzy C-Means Algorithm* (DESFCM). This approach benefits from the fact, that not the data items themselves but the distances between them and cluster prototypes are important for the calculation of membership degrees. Therefore not the missing values of data items but the distances between them and cluster centres are estimated.

DESFCM uses another variant of the FCM as basis. This variant initialises not the cluster prototypes at the beginning but the membership degrees of data items to clusters. In the first iteration step the cluster prototypes are calculated on the basis of completely available data items and their membership degrees to these prototypes. The data set $X_1, ..., X_n$ is divided into the set of completely observed data items $X_{obs}$ and the set of data items with missing values $X_{mis}$. Then the cluster prototypes are calculated as follows:

$$\mu_{ij} = (\sum_{k=1}^{|X_{obs}|}(u_{ik})^m x_{kj})/(\sum_{k=1}^{|X_{obs}|}(u_{ik})^m) \qquad (13)$$

for $1 \leq i \leq c$ and $1 \leq j \leq d$.

In the second iteration step, the membership degrees are calculated from the distances between data items and cluster prototypes (cf. Formula 2). DESFCM algorithm uses the squared Euclidean distance function as a basis, which is calculated as a sum of the squares of the differences between corresponding values (cf. Formula 14).

$$D_{ik} = dist_2^2(x_k, \mu_i) = \sum_{j=1}^{d}(x_{kj} - \mu_{ij})^2 \qquad (14)$$

As the differences $(x_{kj} - \mu_{ij})$ for all missing values $x_{kj}$ cannot be calculated, according to [11] they are estimated depending on the values for the feature $j$ of all completely available data items $x_k \in X_{obs}$ as follows:

$$(x_{kj} - \mu_{ij})^2 = \frac{\sum_{k=1}^{|X_{obs}|} u_{ik}(x_{kj} - \mu_{ij})^2}{\sum_{k=1}^{|X_{obs}|} u_{ik}} \qquad (15)$$

The estimation formula includes the membership degrees of complete data items to the cluster prototype. In this way the squared differences including values of data items close to cluster prototype have more influence during the estimation.

## IV. Data Experiments

To analyse and compare the clustering methods for incomplete data described above, we have conducted several experiments on two test data sets. For our evaluation we can not use real incomplete data sets, because we compare the clustering results of the complete data set with clustering results of the same data set with missing values. In this way

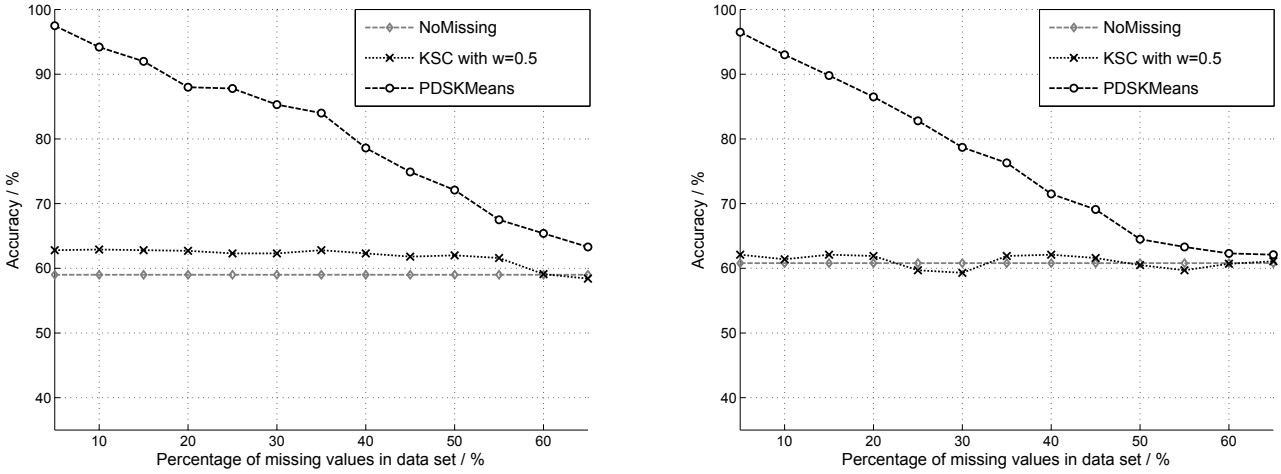



Fig. 1. Accuracy for $k$-means methods on (a) data set 1 and (b) data set 2 in dependence on percentage of missing values MCAR.

we assume the partitioning of complete data as the "real" one and use the accuracy as the performance measurement for clustering algorithms.

Both data sets were generated by a composition of three 3-dimensional Gaussian distributions. Data set 1 consists of 100 data items which are uniformly distributed on three clusters. As in real world applications clusters are generally differently sized, and there is often no clear separation between clusters, we generated another test data set. Data set 2 contains 300 data items which are differently distributed on three clusters with 50, 100 and 150 data items. The data points were generated within the groups via Gaussian distributions with different standard deviations. Furthermore, the centres were chosen near to each other. Both data sets were clustered with different cluster numbers using the $k$-means resp. fuzzy c-means algorithms. As expected the best results were achieved for three clusters. We got a silhouette coefficient [7] of 0.61 for data set 1 and 0.4 for data set 2, that indicates a considerably weaker structure than for data set 1. As dependent features do not provide additional information for the clustering, we ensured that the values of different attributes are uncorrelated in both data sets. Besides, in this way we want to avoid some biases in our experiments, e.g. that missing values are compensated by values of correlated features.

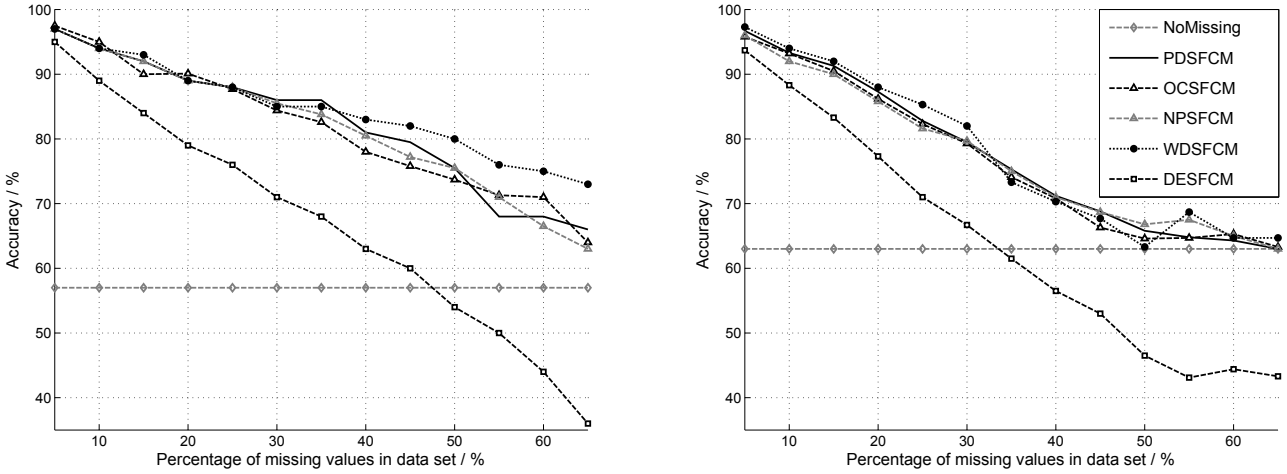
To generate an incomplete data set, the complete data set is modified by successively removing values in two of three features with different probabilities. The percentage of missing values was calculated in relation to all values in a data set. Depending on the cause of missingness, missing values can induce a random or conditional reduction of a data set. Therefore, to test whether the performance of clustering algorithms for incomplete data depends on different missing-data mechanisms, we deleted the values from test data according to the common missing-data mechanisms MCAR, MAR and NMAR described in Section II-B. Afterwards, these missing-data mechanisms were proved with appropriate statistical hypothesis tests using software package *SPSS 16.0 for Windows*.

In our experiments we proceeded as follows: first we clustered the complete data sets with basic $k$-means resp. fuzzy c-means algorithms to get the actual distribution of the data items into clusters as baseline. Then we clustered the incomplete data sets with various clustering algorithms for data with missing values. To create the test conditions as real as possible, we initialised the cluster prototypes with random values at the beginning. For the stopping criterion of FCM $\| \mu - \mu' \| < \epsilon$ we used the Frobenius norm distance. In all our experiments we set the value $\epsilon$ to 0.0001. We compare the introduced methods with another standard method for handling missing data, which discards all features with missing values, relying only on the completely available features. This approach is referred to here as *NoMissing*. As the basic algorithms $k$-means and fuzzy c-means find different partitions of the complete data sets into clusters, we make a separate comparison of clustering methods for data with missing values. Below we present the results of our experiments organised according to missing-data mechanisms.

### A. Test Results on Data with Missing Values MCAR

Figure 1 shows the performance results for three $k$-means methods for incomplete data, KSC, PDSKMeans and NoMissing, on two test data sets with missing values "missing completely at random". To create comparable conditions for KSC and PDSKMeans, in this experiment the values were deleted in two features according to a multivariate pattern. That is if values were deleted from a data item, then they were deleted in two features for this item [8]. In this way two kinds of data items occur in the data set: completely available data items and data items with missing values in the second and third features.

To evaluate the performance, we compare the averaged accuracy obtained over 20 trials in relation to the percentage of




(a) (b)

Fig. 2. Accuracy for fuzzy c-means methods on (a) data set 1 and (b) data set 2 in dependence on percentage of missing values MCAR.

missing values in the data sets. As Figure 1 shows, KSC is only marginally better on the average than the NoMissing approach. Although KSC and PDSKMeans use the same number of feature values for clustering data, the experimental results show that, nevertheless, PDSKMeans delivers considerably better results than KSC. The difference in the performance of both algorithms decreases with increasing number of missing values in the data sets.

Figure 2 represents the averaged accuracy obtained over 20 trials for various fuzzy c-means methods for incomplete data in relation to the percentage of missing values in data sets. The missing values are "missing completely at random" and they occur in two features according to a multivariate pattern. The experimental results show that the algorithms WDSFCM, PDSFCM, OCSFCM and NPSFCM deliver nearly equally good results and the accuracy for WDSFCM is actually slightly better on average. DESFCM shows the worst results. Its accuracy decreases faster with increasing number of missing values in the data sets than for other algorithms. In the case of a high percentage of missing values in the data sets, the accuracy for DESFCM lies even below the accuracy for the NoMissing method.

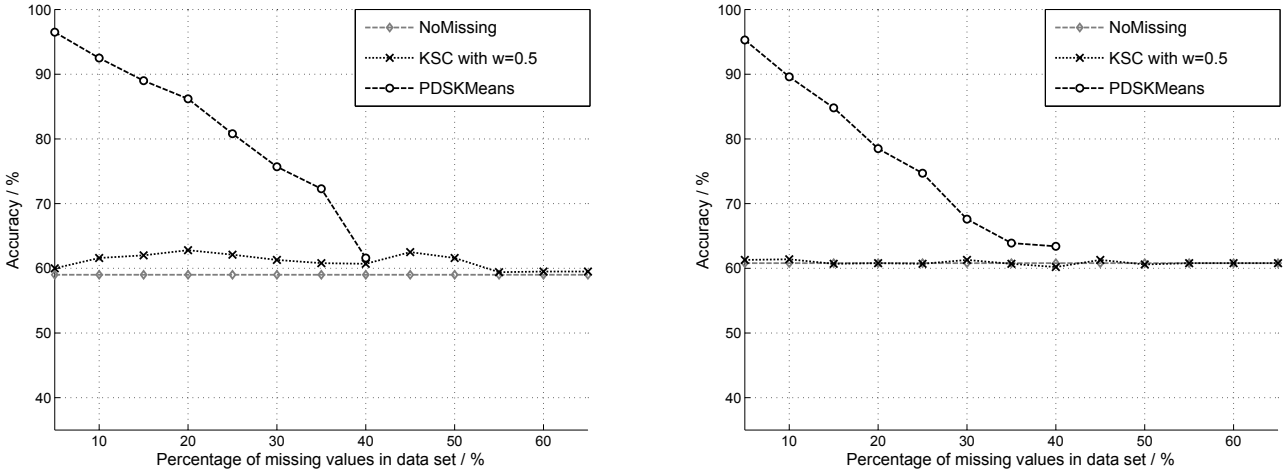
The performance results for both $k$-means and fuzzy c-means methods for incomplete data shows that the accuracy for all methods is considerably better on data set 1 with uniformly distributed data items than on data set 2 with differently sized clusters. This is due to the fact that all methods assign data items to clusters only depending on distances between data items and cluster prototypes. These methods disregard the information about the cluster shapes. In this way, it is highly possible, that especially the marginal objects of large clusters are assigned falsely to the nearest small clusters.

### *B. Test Results on Data with Missing Values MAR*

The performance results for KSC and PDSKMeans on data sets with missing values "missing at random" are depicted in Figure 3. To compare these results with other experimental results, the feature values were removed in two features according to a multivariate pattern. Again, we compare the averaged accuracy obtained over 20 trials in relation to the percentage of missing values in data sets. The accuracy for PDSKMeans is not computed from $40\%$ missing values in data sets. This is due to the fact that some clusters consist only of incomplete data items, therefore not all coordinates of cluster prototypes could be calculated. As the algorithm terminated only after a few iterations (1-4), it does not provide reliable clustering results (in some cases data items are assigned to the initial cluster prototypes, e.g. they are assigned to random points in the data space).

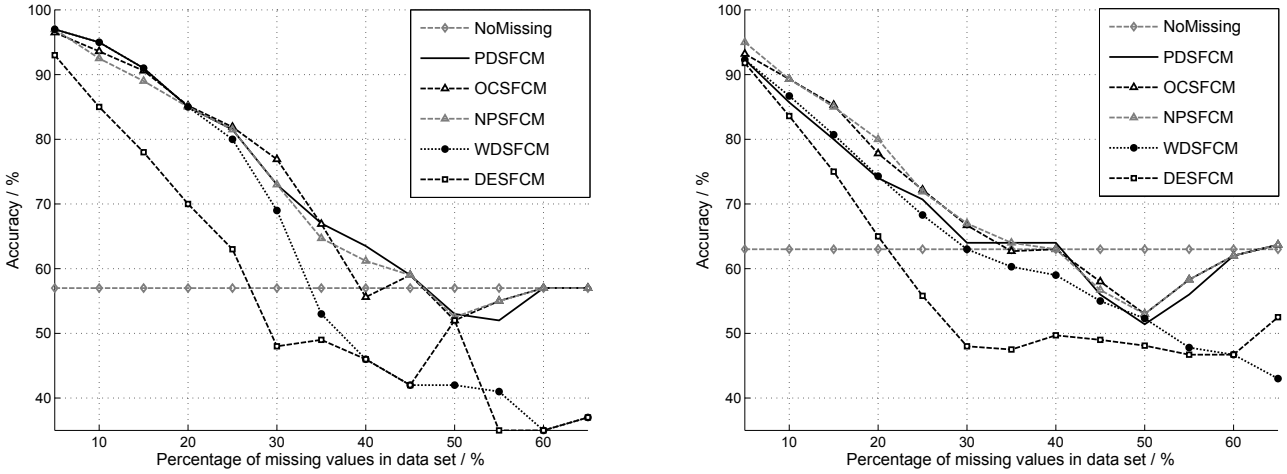
For less than $40\%$ missing values in data sets, PDSKMeans produces considerably lower number of misclassification errors than KSC. But in comparison to missing values MCAR, PDSKMeans performs poorer on data with missing values MAR.

Figure 4 shows the averaged accuracy obtained over 20 trials for fuzzy c-means methods for incomplete data on data sets with missing values MAR. As $k$-means methods, fuzzy c-means methods for incomplete data perform considerably worse on data with missing values MAR than on data sets with missing values MCAR. In the case of a high percentage of missing values in the data sets the accuracy for fuzzy c-means algorithms lies even below the accuracy for the NoMissing approach. WDSFCM and DESFCM produce noticeably more misclassification errors than other approaches. This is due to the fact that missing values MAR, in contrast to missing values MCAR, occur in data items depending on values of available features and thus, they occur in data items with particular properties. In this way missing values MAR induce a conditional reduction of data set so that complete data items do not represent the whole data set anymore. As WDSFCM and DESFCM calculate cluster prototypes only on the basis of complete data items, the computed cluster prototypes differ




(a) (b)

Fig. 3. Accuracy for $k$-means methods on (a) data set 1 and (b) data set 2 in dependence on percentage of missing values MAR.




(a) (b)

Fig. 4. Accuracy for fuzzy c-means methods on (a) data set 1 and (b) data set 2 in dependence on percentage of missing values MAR.

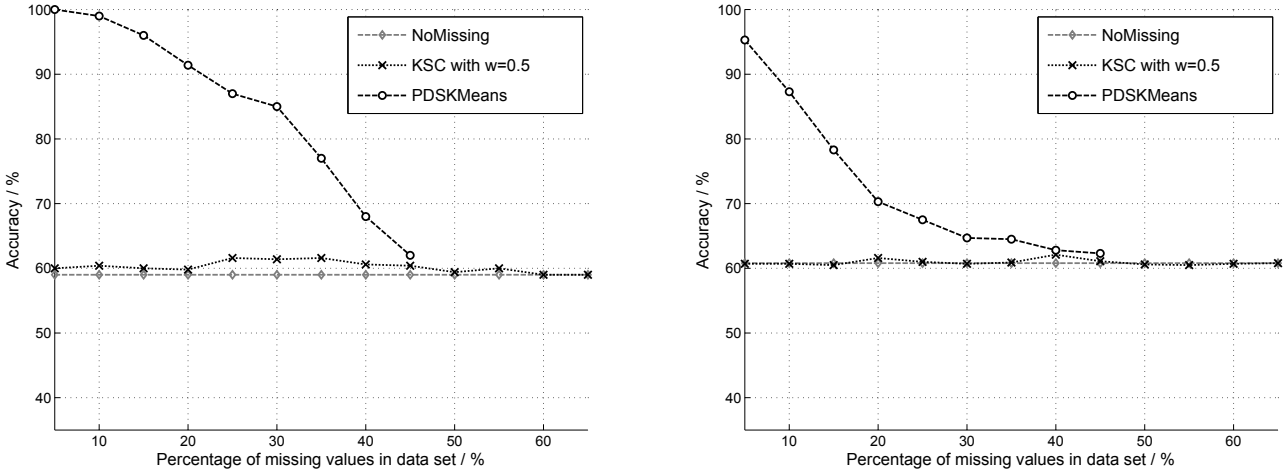
from the actual cluster centres. Consequently the partitioning of data items found by these two approaches differs from the actual partitioning.

As in the case of missing values MCAR, all methods perform better on data set 1 with uniformly distributed data items than on data set 2 with differently sized clusters.

### *C. Test Results on Data with Missing Values NMAR*

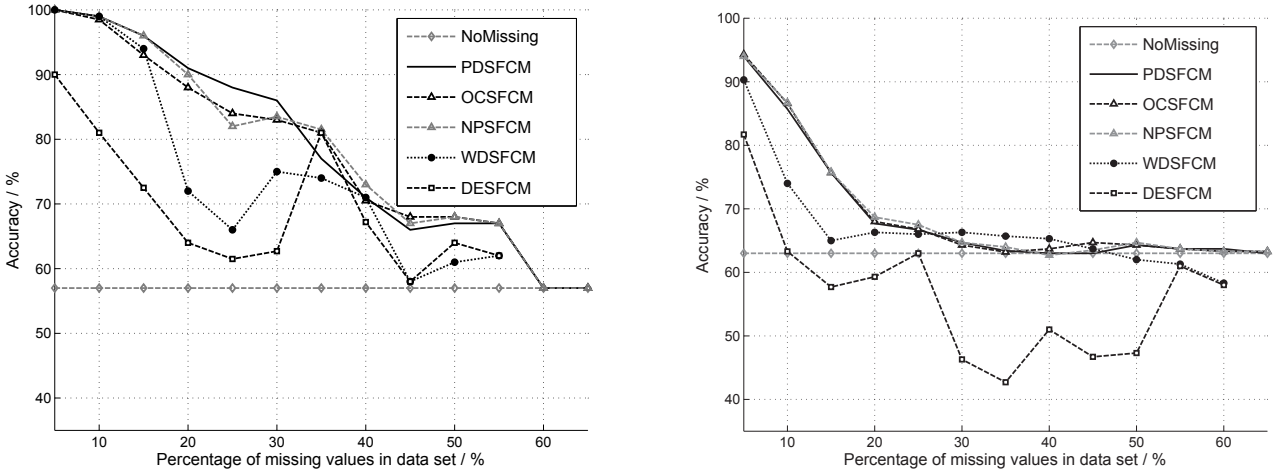
Finally, we evaluated presented methods on test data with missing values "not missing at random", i.e. the missingness of values in data items depends on the actual realisation of these values. In the case of missing values NMAR in more than one feature only the general pattern could be generated, i.e. missing values in a data item can appear in one or more features [8]. To generate the multivariate missing-data pattern for data with missing values NMAR in multiple features, the values of these features must strongly correlate with each other, but this contradicts to our assumption.

Figure 5 shows the experimental results for KSC and PDSKMeans on data sets with missing values NMAR in two features. The accuracy for PDSKMeans is not computed for $50\%$ and more missing values in data sets, because not all cluster prototypes could be completely calculated and the algorithm terminated after only a few iterations. So PDSKMeans did not provide reliable clustering results. As in the case of missing values MAR, the algorithms perform worse than on data with missing values MCAR. The reasons behind this are the same – missing values NMAR induce a conditional reduction of data sets so that the complete data items do not represent the whole data sets. For a low number of missing values in data sets (less than $35\%$), PDSKMeans performs better than KSC, but in comparison to our previous experiments, PDSKMeans uses also more available feature values than KSC.




(a) (b)

Fig. 5. Accuracy for $k$-means methods on (a) data set 1 and (b) data set 2 in dependence on percentage of missing values NMAR.




(a) (b)

Fig. 6. Accuracy for fuzzy c-means methods on (a) data set 1 and (b) data set 2 in dependence on percentage of missing values NMAR.

The performance results for fuzzy c-means methods for incomplete data are shown in Figure 6. The accuracy for PDSFCM, OCSFCM and NPSFCM differs again insignificantly and lies by a small percentage of missing values in data sets considerably above the accuracy for the NoMissing approach. With increasing number of missing values in the data sets the accuracy for these algorithms converges to the accuracy for the NoMissing approach. For more than $30\%$ missing values in data set 2, there are no significant differences between these four approaches. The unsteady development of accuracy for WDSFCM and DESFCM on data set 1 shows the negative effects of a conditional reduction of a data set. The accuracy for these algorithms plummets for $25\%$ missing values, it improves with increasing number of missing values and declines again for $45\%$ missing values in the data set. The accuracy of clustering produced by WDSFCM and DESFCM strongly depends on the distribution of complete data items, because they provide a basis for the calculation of cluster centres for these two algorithms. In the cases of about $25\%$ and $45\%$ of missing values NMAR in data set 1, complete data items represent a biased mapping of the whole data set. Therefore WDSFCM and DESFCM produce a high number of misclassification errors. As the other three methods use all data items for calculating cluster prototypes, their accuracy graphs do not have such inhomogeneities. Therefore, clustering results produced by these three methods are more reliable in practise than clustering results produced by WDSFCM and DESFCM.

### *D. Discussion*

In this section we discuss the results of evaluated clustering approaches for incomplete data. For the $k$-means clustering algorithm we analysed and compared two methods KSC and PDSKMeans. Experiments showed that the performance of KSC is not considerably better than the performance of the

NoMissing approach, although KSC uses more information than NoMissing. PDSKMeans generally produces more accurate results than KSC. For small percentage of missing values in data set the accuracy of PDSKMeans is above 90%. With increasing number of missing values in data set the accuracy of PDSKMeans converge to the accuracy of the NoMissing approach. A drawback of KSC is that it cannot be used when missing values occur in all features, because it carries out the analysis on the basis of completely available features. The applicability of PDSKMeans is restricted only if no data item within a cluster has a value in an attribute, so that not all coordinates of the cluster prototype can be calculated. As our experiments showed, this often occurs in the case of high percentage of missing values MAR and NMAR.

We have also evaluated five approaches for adapting fuzzy c-means clustering algorithm for data with missing values. Our experiments showed that PDSFCM, OCSFCM and NPSFCM produce the most accurate results. Although the averaged accuracy is very similar for these three approaches, OCSFCM and NPSFCM produce more instable results than PDSFCM. The accuracy of clustering obtained by OCSFCM and NPSFCM varies from trial to trial up to $10\%$. Performance of WDSFCM depends strongly on the missing-data mechanism. When missing values are MCAR, WDSFCM achieves comparatively good and even partially more accurate assignment of data items as three approaches called above. In the case of missing values MAR or NMAR WDSFCM performs noticeably more poorly. DESFCM produced the worst results of all approaches. Performance of DESFCM decreases faster with increasing percentage of missing values in data sets than for other approaches. Partially it even performs worse than the NoMissing approach.

Our experiments showed that the performance of algorithms depends quite considerably on whether missing values induce a random or conditional reduction of data set. On data sets with missing values MAR or NMAR, all algorithms produced more misclassifications than on data sets with missing values MCAR. Particularly the performance difference is quite significant for WDSFCM and DESFCM, which perform the iterative process or parts of it only on the basis of complete data items. But this means also that the refinement algorithm for cluster centres initialisation [4] would fail to work on large data sets with missing values MAR or NMAR, if subsamples would consist only of completely available data items. Therefore, in order to achieve optimal clustering results, both incomplete and complete data items are to be included in the whole iterative process.

The experiments also showed that all algorithms for incomplete data perform better on uniformly distributed data (data set 1) than on data set with differently sized clusters (data set 2). This is due to the fact that these methods estimate missing values of incomplete data items respectively distances between incomplete data items and cluster centres completely disregarding information about cluster sizes. This way the marginal objects of a large cluster, which have large distances to their cluster centre, are falsely assigned to the nearest small cluster.

## V. Conclusion and Future Work

In this paper we discussed different approaches proposed in the literature for adapting partitioning clustering algorithms to incomplete data. We analysed them on two appropriate data sets and compared them with each other. In order to demonstrate the performance of clustering methods on different types of incomplete data, we analysed the accuracy of these methods depending on the different missing-data mechanisms and the percentage of missing values in the data sets. The experiments showed that all algorithms for incomplete data produced more misclassifications on data sets with missing values MAR or NMAR than on data sets with missing values MCAR. Furthermore, our experimental results showed that the algorithms perform better on uniformly distributed data than on data set with differently sized clusters. But in real world applications data objects are generally distributed on differently sized clusters. Therefore, in order to achieve more accurate clustering results on unequally distributed data, in the future work we plan to develop an extension for some of presented methods, which takes the cluster dispersion into account.

There is another interesting open question regarding the applicability of clustering methods for incomplete data treated in this paper. In all our experiments we assumed the real cluster number because we have calculated it before on complete data. However, in practise the number of clusters often is not known a priori. The silhouette coefficient, which we used for calculation of cluster number, can only be applied if all values of data items are known or estimated as for OCSFCM and NPSFCM. As other clustering algorithms for incomplete data do not fill in the missing values, we can not use the silhouette coefficient to analyse these methods to what extend they can calculate the real cluster number. To our knowledge there is no measure similar to silhouette coefficient, which works on incomplete data. Therefore, in our future work we intend to develop a such measure and to analyse the algorithms with regard to the correct calculation of the cluster number.

## References


[1] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer Academic Publishers, 1981.

[2] R. P. David Freedman and R. Purves, *Statistics*, Norton, New York, 1998.

[3] J. K. Dixon, "Pattern Recognition with Partly Missing Data", *IEEE Transactions on System, Man and Cybernetics*, vol. 9, pp. 617–621, 1979.

[4] U. M. Fayyad, C. Reina, and P. S. Bradley, "Initialization of Iterative Refinement Clustering Algorithms", In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, pp. 194–198, 1998.

[5] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, San Francisco, CA, USA, 2000.

[6] R. J. Hathaway and J. C. Bezdek, "Fuzzy c-means Clustering of Incomplete Data", *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 31, no. 5, pp. 735–744, 2001.

[7] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & Sons, 1990.

[8] R. J. Little and D. B. Rubin, *Statistical Analysis with Missing Data*, John Wiley & Sons, 2002.

[9] R. J. A. Little, "A Test of Missing Completely at Random for Multivariate Data with Missing Values", *Journal of the American Statistical Association*, vol. 83, no. 404, pp. 1198–1202, 1998.
[10] S. P. Lloyd, "Least Squares Quantization in PCM", *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–136, 1982.
[11] M. Sarkar and T.-Y. Leong, "Fuzzy k-means Clustering with Missing Values", In *Proceedings of American Medical Informatics Association Annual Symposium (AMIA)*, pp. 588–592, 2001.
[12] S. Theodoridis, *Pattern Recognition*, Elsevier Books, Oxford, 2003.
[13] K. Wagstaff, "Clustering with Missing Values: No Imputation Required", In *Classification, Clustering, and Data Mining Applications (Proceedings of the Meeting of the International Federation of Classification Societies)*, pp. 649–658, 2004.