IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 5, 2021, accepted December 11, 2021, date of publication December 20, 2021,
date of current version January 4, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3136435

Toolbox for Distance Estimation and Cluster Validation on Data With Missing Values

MARKO NIEMELÄ, SAMI ÄYRÄMÖ, AND TOMMI KÄRKKÄINEN, (Senior Member, IEEE)
Faculty of Information Technology, University of Jyvaskyla, 40640 Jyvaskyla, Finland

Corresponding author: Marko Niemelä (marko.p.niemela@jyu.fi)

The work of Tommi Kärkkäinen was supported by a grant from the Academy of Finland under Grant 311877 (Demo) and Grant 315550 (HNP-AI).

ABSTRACT Missing data are unavoidable in the real-world application of unsupervised machine learning, and their nonoptimal processing may decrease the quality of data-driven models. Imputation is a common remedy for missing values, but directly estimating expected distances have also emerged. Because treatment of missing values is rarely considered in clustering related tasks and distance metrics have a central role both in clustering and cluster validation, we developed a new toolbox that provides a wide range of algorithms for data preprocessing, distance estimation, clustering, and cluster validation in the presence of missing values. All these are core elements in any comprehensive cluster analysis methodology. We describe the methodological background of the implemented algorithms and present multiple illustrations of their use. The experiments include validating distance estimation methods against selected reference methods and demonstrating the performance of internal cluster validation indices. The experimental results demonstrate the general usability of the toolbox for the straightforward realization of alternate data processing pipelines. Source code, data sets, results, and example macros are available on GitHub. https://github.com/markoniem/nanclustering_toolbox

INDEX TERMS Missing values, distance estimation, clustering, cluster validation.

I. INTRODUCTION

In many machine learning tasks, the volume of data is limited, necessitating that all the available data values be utilized as extensively as possible. The assumption that the data is complete is often invalid in real-world applications [1]. A simple strategy for avoiding the problem of missing data is to omit incomplete observations. However, this is not an efficient use of data because the important information may be lost. A more sophisticated strategy is to impute missing values as part of a data preprocessing step. Different imputation mechanisms have been developed for various data types, e.g., binary, ordinal, categorical, and string attributes [2]. The nearest neighbors method is a common imputation approach for numerical values, which uses an average (with or without weights) of the k-nearest neighbors [2].

Estimating distances is an alternative way to address problems with missing values. A well-known distance estimation method is the partial distance strategy (PDS) [3], which is also known as a general similarity measure [4]. This approach involves similar limitations as the nearest neighbors method so that its accuracy is highly correlated to the number of missing values in data. In [5] and in [6], the expected distance estimations were reported to be more accurate than the data imputation or the PDS for selected real-world data sets. However, the performance of these methods has not been tested in unsupervised machine learning tasks such as data clustering. Clustering can benefit from accurate distance estimation with missing values because both currently popular initialization methods like K-means++ [7] and the computation of cluster centroids are based on distances and not on observations themselves. In [5] and in [6], data values were assumed to be missing at random (MAR), where missingness may depend on the available data. MAR is a less restrictive mechanism than missing completely at random (MCAR) in which the values are missing independently of any other data values.

Many unsupervised and supervised techniques, and their combinations, have been used for data imputation. Imputation of missing sensor spatially and/or temporally dependent data using autoencoder and alternation projection onto convex sets

The associate editor coordinating the review of this manuscript and approving it for publication was Xi Peng.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

based training was proposed in [8]. Shallow neural networks (both multi-layered perceptron and radial basis function) with genetic algorithms were put forward in [9]. Fuzzy clustering and support vector regression, also with a genetic algorithm-based parameter estimation, were hybridized in [10]. Decision trees and their ensembles were applied in [11]. More recently, deep learning methods, especially deep autoencoders, have been proposed and tested for mainly spatio-temporal data, e.g., in [12]–[18]. We do not address these more complex techniques here because of laborious tuning of an extensive number of metalevel parameters (e.g., what network architecture, how many layers, what kind of layers, which loss function, what training method, how much and what kind of data needed, etc., see [13], [19]).

Cluster analysis is often considered as one of the core techniques in descriptive data mining and knowledge discovery [20], statistics [21], and pattern recognition [22]. It is a stepwise process with at least nine elements to be chosen/carried out before achieving the results [23]–[25]. The elements are related to data selection, data preprocessing, selection of distance measure, choice of clustering criterion, selection missing data strategy, validation of the created algorithms, selection of the number of clusters, and finally, interpretation of results.

Clustering divides data into disjoint groups (clusters) where an ideal cluster is compact and isolated [24]. Partitional clustering methods use prototype points to represent clusters and, therefore, are also referred to as prototype-based clustering methods [26]. The methods are aimed to minimize the variance around the prototype points based on an error (score) function, and they are also called variance minimization techniques [27]. The iterative relocation procedure decreases the values of the error function until final convergence is reached [28], [29].

Cluster validation is a crucial part of cluster analysis, in which a clustering solution's quality (ideality) is being assessed. Cluster validation indices (CVIs) provide quality measures that indicate the number of clusters. The three main types of indices are relative, external, and internal [30]. The relative index compares multiple clustering results obtained with different initial settings of the clustering algorithm, whereas an external index utilizes additional information or metadata that can explain the number or form of the cluster structures. The external indices can be used, e.g., for comparing different clustering methods using the metadata of the actual cluster labels. However, internal cluster validation indices are probably the most commonly used estimates because they utilize only the information obtainable from data and clustering results. Numerous different clustering methods, including internal cluster validation indices, have been developed because of the high diversity of data [24]; for example, challenging data sets may include noise, overlapped clusters, multiple dimensions, and/or different densities [31].

This paper introduces a toolbox that encapsulates many methods and algorithms to perform cluster analysis in the presence of missing data. The versatile functionality allows a toolbox user to generate many forms of experimental settings and to realize various forms of new experiments to better understand and improve unsupervised learning with missing values.

The methodological bases in Sections II–V explain background theory related to distance computation with missing values, data preprocessing, clustering, and cluster validation. Section VI gives an overview of the toolbox, including descriptions of the sample data sets and essential toolbox functions. Section VII describes experiments that are divided into three parts. In the first part, the performance of distance estimation algorithms is measured in the direct estimation of pairwise distances in data sets with missing values. The second part compares clustering methods and cluster validation indices on two-dimensional (2D) data sets with missing values. In addition, the validation results, which are based on a key point selection function [32], are given. The results are validated against the reference results given in previously published research papers. In the third part, experiments are conducted on multidimensional data sets that were created by a recently published data generator [33]. Finally, the content and the toolbox performance are discussed and summarized in Sections VIII–IX.

II. COMPUTATION OF DISTANCES WITH MISSING VALUES

Let $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^n$ for all i, denote the observational data set with N observations of size n.

A. AVAILABLE DATA STRATEGY

The available data strategy (ADS) (see [34]) restricts distance computations to available values via binary projection vectors, $\{\mathbf{p}_i\}_{i=1}^N$, $\mathbf{p}_i \in \{0, 1\}^n$, which represent the sparsity pattern of each observation:

$$(\mathbf{p}_k)_i = \begin{cases} 1, & \text{if } (\mathbf{x}_k)_i \text{ exists,} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

The ADS is used in K-spatialmedians clustering (see, e.g. [35]), and it generalizes easily for various distance measures. For instance, the Euclidean distance between two incomplete n-dimensional column vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ is defined as

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\sum_{i=1}^{n} ((\mathbf{p}_1)_i(\mathbf{x}_1)_i - (\mathbf{p}_2)_i(\mathbf{x}_2)_i)^2}. \tag{2}$$

B. PARTIAL DISTANCE STRATEGY

The PDS computes the sum of pairwise available vector values and scales the sum by the ratio of the original dimension of the vectors and the number of available pairwise values [4]. The Euclidean distance reads then as

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{\frac{n}{n^*} \sum_{i=1}^{n} ((\mathbf{p}_1)_i(\mathbf{x}_1)_i - (\mathbf{p}_2)_i(\mathbf{x}_2)_i)^2}, \tag{3}$$

where n^* is the number of pairwise known values. Similarly to the ADS, the PDS can be generalized to other distance measures, such as the City block distance (see [36]).

C. EXPECTED SQUARED EUCLIDEAN DISTANCE

The framework for estimating the expected distance between two data vectors is presented in [5]. The proposed framework was designed for estimating squared Euclidean distances in the presence of missing data values and under the assumption of multivariate normal distribution. The assumption of multivariate normally distributed data is used for estimating expected values to replace the missing values in the data. The central limit theorem states that normal distribution can be used to approximate nearly any continuous distribution with a sufficiently large sample (see, e.g., [37]). The basic elements of the framework are given in Appendix A, and a more detailed description is given in [5].

Let us define the index sets of missing M_i and available A_i values of observation $\mathbf{x}_i$ as specified by $\mathbf{p}_i$, i.e., $M_i = \{1 \leq j \leq n | (\mathbf{p}_i)_j = 0\}$ and $A_i = \{1 \leq j \leq n | (\mathbf{p}_i)_j = 1\}$. Following the assumption that missing values are generated from conditional multivariate normal distribution, in which data values are independent, and missing values depend on the available values under the MAR assumption on the sparsity pattern, the expectation of the squared distance between two data vectors reads as:

$$E\Big[||\mathbf{x}_1 - \mathbf{x}_2||^2\Big] = \sum_{i=1}^{n} \Big(((\mathbf{x}'_1)_i - (\mathbf{x}'_2)_i)^2 + (\sigma'_1)_i^2 + (\sigma'_2)_i^2\Big),$$

$$(\mathbf{x}'_k)_i = \begin{cases} (\mathbf{x}_k)_i, & \text{if } i \in A_k; \\ E[(\mathbf{x}_k)_i | (\mathbf{x}_k)_{A_k}], & \text{if } i \in M_k; \end{cases}$$

$$(\sigma'_k)_i^2 = \begin{cases} 0, & \text{if } i \in A_k; \\ Var[(\mathbf{x}_k)_i \,|\, (\mathbf{x}_k)_{A_k}], & \text{if } i \in M_k. \end{cases} \quad (4)$$

With the complete derivation given in Appendix B, the ith observation concerning the missing values is normally distributed with the mean vector

$$(\boldsymbol{\mu}'_i)_{M_i} = (\boldsymbol{\mu})_{M_i} + \Sigma_{M_i A_i} \Sigma_{A_i A_i}^{-1} ((\mathbf{x}_i)_{A_i} - (\boldsymbol{\mu})_{A_i}), \quad (5)$$

and covariance matrix

$$\Sigma'_{M_i M_i} = \Sigma_{M_i M_i} - \Sigma_{M_i A_i} \Sigma_{A_i A_i}^{-1} \Sigma_{A_i M_i}. \quad (6)$$

Estimating $\boldsymbol{\mu}$ and Σ for incomplete data is not a simple task, especially if the number of missing values is large compared to the number of available ones. A method based on available data is a fast alternative for estimating the covariance matrix [38]. However, the iterative expectation maximization (EM) algorithm with the maximum negative log-likelihood convergence criterion is more commonly used, e.g., in [5], [39], and [6].

1) EXPECTATION MAXIMIZATION

The EM is an iterative method to find the best estimates for the parameters in a statistical model [40], [41]. It consists of two alternating steps: expectation and maximization. The expectation step estimates the missing values in the data set. The maximization step optimizes the model parameters to fit the data best. The steps are repeated until the final convergence is reached.

The EM algorithm for estimating the mean vector $\boldsymbol{\mu}$ and the covariance matrix Σ of a data set with missing values under the assumption of the conditional multivariate normal distribution is given in Algorithm 1. The algorithm includes a bias matrix $\mathbf{B}$ with the same size as the covariance matrix Σ.

Algorithm 1 Expectation Maximization

Input: An incomplete data set $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i \in \mathbb{R}^n$.
1. Compute mean vector $\boldsymbol{\mu}$ of available values of the data set.
2. Impute missing values by $\boldsymbol{\mu}$ to obtain the imputed matrix $\mathbf{X}_{imp}$.
3. Recompute $\boldsymbol{\mu}$ and compute covariance matrix Σ by using imputed data.
4. Create a zero matrix $\mathbf{B}$ which size is equal to Σ.
until *final convergence* **do**
  for each $\mathbf{x}_i$ **for which** M_i **is nonempty do**
    5. Impute missing values by using the formula (5).
    6. Use formula (6) and compute $\mathbf{B}_{M_i M_i} = \mathbf{B}_{M_i M_i} + \Sigma'_{M_i M_i}$.
  7. Recompute $\boldsymbol{\mu}$ and update covariance as $\Sigma = \Sigma + \mathbf{B}/N$.
  8. Remove the imputed values from the $\mathbf{X}$.
  9. Restore zeros to the matrix $\mathbf{B}$.
Output: Mean vector $\boldsymbol{\mu}$ and covariance matrix Σ.

The termination criterion for Algorithm 1 is based on the negative log-likelihood function that for the multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ reads as:

$$\begin{aligned} &\ln(\mathcal{L}(\boldsymbol{\mu}, \Sigma)) \\ &= \sum_{i=1}^{N} \frac{1}{2}[\ln(\det(\Sigma)) + (\mathbf{x}_i - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) + n \ln(2\pi)] \\ &= \frac{1}{2} N [\ln(\det(\mathbf{L})^2) + n \ln(2\pi)] \\ &\quad + \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \\ &= \frac{1}{2} N [2 \sum_{j=1}^{n} \ln(\mathbf{L}_{jj}) + n \ln(2\pi)] \\ &\quad + \frac{1}{2} \sum_{i=1}^{N} (\mathbf{x}_i - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x}_i - \boldsymbol{\mu}), \end{aligned} \quad (7)$$

where $\mathbf{L}$ is obtained from the Cholesky decomposition of the covariance matrix Σ, i.e., $\Sigma = \mathbf{L}\mathbf{L}^T$. Convergence is reached when there is no significant change in the values of the log-likelihood function between successive iterations.

2) FINAL ALGORITHM

The steps for computing ESDs for incomplete data are given in Algorithm 2.

D. EXPECTED EUCLIDEAN DISTANCE

In [6], the work in [5] and [39] was continued by extending the ESD distance for the expected Euclidean distance (EED). It was shown that the EED distance could be modeled with a

Algorithm 2 Expected Squared Euclidean Distances

Input: An incomplete data set $\{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i \in \mathbb{R}^n$.

1. Compute the mean vector $\boldsymbol{\mu}$ and covariance matrix Σ of incomplete data set using Algorithm 1.

for each $\mathbf{x}_i$ **for which** M_i **is nonempty do**

2. Compute the conditional mean $(\boldsymbol{\mu_i}')_{M_i}$ using formula (5) and the conditional covariance matrix $\boldsymbol{\Sigma}'_{M_iM_i}$ using formula (6), respectively.
3. Impute missing values of $\mathbf{x}_i$ by values from $(\boldsymbol{\mu_i}')_{M_i}$ to obtain $\mathbf{x}'_i$.
4. Impute conditional variance terms of $\boldsymbol{\sigma}'^{2_i}$ from the diagonal of $\Sigma'_{M_iM_i}$.

for each pair of $\mathbf{x}_i$ and $\mathbf{x}_j$ **in** $\{\mathbf{x}_i\}_{i=1}^N$ and **in** $\{\mathbf{x}_j\}_{j=i+1}^N$ **do**

5. Compute the expected distance by utilizing the formula (4).

Output: Pairwise squared Euclidean distances $\boldsymbol{d}_{ij}$ of data vectors.

Nakagami distribution if the distances are assumed to follow the Gamma distribution. The expected Nakagami distributed values can then be obtained as follows:

$$E\left[(\sum_{i=1}^{n}((\mathbf{x}_1)_i-(\mathbf{x}_2)_i)^2)^{\frac{1}{2}}\right]=E[z^{\frac{1}{2}}]=\frac{\Gamma(m+\frac{1}{2})}{\Gamma(m)}\left(\frac{\Omega}{m}\right)^{\frac{1}{2}},$$
$$m=\frac{E[z]^2}{Var[z]},\quad \Omega=E[z], \tag{8}$$

where m and Ω are the shape and spread parameters of the Nakagami distribution, respectively, and Γ is the Gamma function.

Under the independence assumption (as in [5], [39]), the variance can be expressed as

$$\begin{aligned}Var[z] &= Var\left[\sum_{i=1}^{n}((\mathbf{x}_1)_i-(\mathbf{x}_2)_i)^2\right]\\ &=\sum_{i=1}^{n}Var[((\mathbf{x}_1)_i-(\mathbf{x}_2)_i)^2]\\ &=\sum_{i=1}^{n}E[((\mathbf{x}_1)_i-(\mathbf{x}_2)_i)^4]-E[((\mathbf{x}_1)_i-(\mathbf{x}_2)_i)^2]^2\\ &=\Big(\sum_{i=1}^{n}E[(\mathbf{x}_1)_i^4+(\mathbf{x}_2)_i^4-4(\mathbf{x}_1)_i^3(\mathbf{x}_2)_i\\ &\quad -4(\mathbf{x}_1)_i(\mathbf{x}_2)_i^3+6(\mathbf{x}_1)_i^2(\mathbf{x}_2)_i^2]\Big)\\ &\quad -\sum_{i=1}^{n}E[((\mathbf{x}_1)_i-(\mathbf{x}_2)_i)^2]^2,\end{aligned} \tag{9}$$

where the expected values are obtainable using non-central moments. Table 1 presents moments of the normal distribution that can be used directly in the case of multivariate Gaussian distribution. However, weighted moments are needed if the data are assumed to follow Gaussian mixture distribution (see [6] for more details).

The computation of the EED distances is based on the same framework as in Algorithm 2. However, additional steps are required which are given in Algorithm 3.

TABLE 1. Non-central moments of normal distribution.

Expected value	Non-central moment
$E[\mathbf{x}_k]$	$\hat{\mathbf{x}}_k$
$E[\mathbf{x}_k^2]$	$\hat{\mathbf{x}}_k^2+\boldsymbol{\sigma}_k^2$
$E[\mathbf{x}_k^3]$	$\hat{\mathbf{x}}_k^3+3\hat{\mathbf{x}}_k\boldsymbol{\sigma}_k^2$
$E[\mathbf{x}_k^4]$	$\hat{\mathbf{x}}_k^4+6\hat{\mathbf{x}}_k^2\boldsymbol{\sigma}_k^2+3\boldsymbol{\sigma}_k^4$

Algorithm 3 Expected Euclidean Distances

Input: An incomplete data set $\{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i \in \mathbb{R}^n$.

1. Utilize **Algorithm 2** to obtain the spread parameter Ω for each pair of data vectors.

for each pair of $\mathbf{x}_i$ and $\mathbf{x}_j$ **in** $\{\mathbf{x}_i\}_{i=1}^N$ and **in** $\{\mathbf{x}_j\}_{j=i+1}^N$ **do**

2. Compute $Var(z)$ by using formula (9) and non-central moments $(E[\mathbf{x}_k], E[\mathbf{x}_k^2], E[\mathbf{x}_k^3], E[\mathbf{x}_k^4])$ given in Table 1.
3. Use formula (8) to obtain the shape parameter m and the final distance d_{ij}.

Output: Pairwise Euclidean distances $\boldsymbol{d}_{ij}$ of data vectors.

III. DATA PREPROCESSING

A. FEATURE SCALING

Feature scaling is a typical preprocessing step in data analysis. Various data types are often measured in different units, which may lead to data types with large scales dominating the other data types in data-driven models. Various feature scaling approaches have been proposed, but the most commonly used approaches are the z-score and min-max normalization.

The z-score method equalizes the data type weights by transforming each one to a zero mean and unit variance. It is obtained by a linear transformation, subtracting the mean and by dividing the standard deviation:

$$x'=\frac{x-\mu}{\sigma}=\frac{1}{\sigma}x-\frac{\mu}{\sigma}=\alpha x-\beta, \tag{10}$$

where μ and σ are the sample mean and standard deviation of the available values in the data set, respectively, and x' is the scaled value.

Min-max normalization scales data to the selected range. The range may depend on the performed task, but $[-1, 1]$ and $[0, 1]$ are probably the most common choices. The min-max formula for an arbitrary range $[a, b]$ can be written as follows:

$$x'=a+\frac{(x-\min(x))(b-a)}{\max(x)-\min(x)}, \tag{11}$$

where min and max are computed for the available values.

B. K-NEAREST NEIGHBORS IMPUTATION

The k-nearest neighbors (kNN) method is a well-known and popular approach for imputing numerical values [2]. This method can be implemented by finding the closest complete observation for an incomplete observation and imputing the missing values or taking an average of k closest observations, of which some can be partially incomplete. If the missing values of the data vector are not available in the k closest observations, the k should be increased until imputation succeeds.

In the literature, there exist many variants of kNN imputation, e.g., complete-case kNNI (CCkNNI), where a data vector with missing values is imputed by using the average value of a set of k nearest complete observations, or incomplete-case kNNI (ICkNNI), where data vectors are selected from the case library in which the eligible nearest neighbors share the same complete values as $\mathbf{x}_i$ and a missing value is available. In [42], it was suggested that up to $k = 5$ neighbors should be considered. If there are not enough neighbors, the missing value is imputed by the sample mean of all the available values for that data type. Even though nearest neighbors imputation is a straightforward approach for dealing with missing values, it can be inefficient when the number of missing values is relatively high [5].

C. LOW-RANK MATRIX COMPLETION

A low-rank solution for matrix completion is a common technique for data imputation. The low-rank matrix has a decreased number of degrees of freedom and, therefore, it makes the estimation problem of missing values practical to solve [43]. The rank minimization problem can be addressed by using convex relaxation techniques utilizing the nuclear-norm [43], [44], which yields to the minimization of the following optimization problem:

$$\min_{\widetilde{\mathbf{X}}} \; ||\mathbf{p}\cdot\mathbf{x} - \mathbf{p}\cdot\tilde{\mathbf{x}}||_2^2 + \lambda||\widetilde{\mathbf{X}}||_*, \tag{12}$$

where $\widetilde{\mathbf{X}}$ is the completed data matrix which will be estimated, the data vectors $\mathbf{x}$ and $\tilde{\mathbf{x}}$ are flattened versions of the data matrices $\mathbf{X}$ and $\widetilde{\mathbf{X}}$, respectively, and $\cdot$ denotes the dot product. Moreover, the vector $\mathbf{p}$ is the flattened version of the projection matrix defined in (1), λ is the regularization parameter, and $||\cdot||_*$ denotes the nuclear norm. The optimization problem in (12) can be solved iteratively by using a soft-thresholding technique to obtain the updated data vector $\tilde{\mathbf{x}}$. The initial guess of $\tilde{\mathbf{x}}$ is given by the zero vector. Then, in the kth iteration, $\tilde{\mathbf{x}}$ is updated as follows:

$$\tilde{\mathbf{x}}_k = \tilde{\mathbf{x}}_{k-1} + (\mathbf{p}\cdot\mathbf{x} - \mathbf{p}\cdot\tilde{\mathbf{x}}_{k-1}). \tag{13}$$

After that, $\tilde{\mathbf{x}}_k$ is reshaped to a matrix form $\widetilde{\mathbf{X}}_k$, the singular value decomposition is applied to the reshaped matrix, the singular values are soft-thresholded to obtain the updated $\hat{\Sigma}_k$. The $\widetilde{\mathbf{X}}_k = \mathbf{U}\hat{\Sigma}_k\mathbf{V}^T$ is flattened to obtain the final $\tilde{\mathbf{x}}_k$ in the kth iteration. The λ is reduced by a cooling algorithm such that $\lambda_1 > \lambda_2 > \ldots > \lambda_\infty$. The final result is obtained when there is a sufficiently small relative change in the target function $||\mathbf{p}\cdot\mathbf{x} - \mathbf{p}\cdot\tilde{\mathbf{x}}||_2$ or when λ reaches the predefined tolerance.

D. TRANSFORMATION INTO SPHERICAL FORM

The prototype-based K-means and K-spatialmedians clustering methods are not intended to discover any shape clusters because the used location estimates (mean and spatial median) assume spherical symmetry. That is the difference from kernel-based methods; see, e.g., [45]. Such assumption is also inherent in the computation of Inter for cluster validation indices (see Table 2). However, the assumption that a data set contains clusters with spherical shapes can be unrealistic, making the clustering and cluster validation problems more challenging. In [32], a new approach for transforming and normalizing an arbitrarily shaped subset of data to an approximately spherical shape with a specified radius was introduced. The method is based on the notation of chains around high-density key points. The original method assumes a 2D data space. Thus, multidimensional scaling (MDS) [46] can be applied to project high-dimensional data sets into the 2D.

1) DEFINITION OF KEY POINT

The M points from $\mathbf{X}$ with relatively higher density and larger density-based distances are associated with the key points which can be determined by selecting M largest values based on the following equation:

$$\begin{aligned} p_i &= \rho_i r_i, \\ \rho_i &= (\sum_{k=1}^{4} d(\mathbf{x}_i, \mathbf{x}_{i,k}))^{-1}, \quad r_i = \min_{j:\rho_j>\rho_i} d(\mathbf{x}_i, \mathbf{x}_j), \end{aligned} \tag{14}$$

where ρ denotes the density of $\mathbf{x}_i$ and $\mathbf{x}_{i,1}\ldots\mathbf{x}_{i,k}$ are $k = 4$ nearest neighbors of $\mathbf{x}_i$, the minimum distance from $\mathbf{x}_i$ to other points with a higher density is denoted as r_i. The method connects points in the data set using density-based distance as the connection rule. Density-based connections are created until the key points are visited. In [32], the number of key points M was suggested to be selected as $|\sqrt{N}|$.

2) DEFINITION OF CHAIN

Points that are connected to a key point form a chain. Multiple chains to one key point are allowed. Let us assume c chains. Then the chain lengths can be defined as:

$$T_c = \sum_{i=1}^{n_c-1} d(\mathbf{x}_i^{(c)}, \mathbf{x}_{i+1}^{(c)}), \tag{15}$$

where n_c is the total number of points in chain c. In data set normalization, distances are transformed into a new one as follows:

$$d^*(\mathbf{x}_i^{(c)}, \mathbf{x}_{i+1}^{(c)}) = d(\mathbf{x}_i^{(c)}, \mathbf{x}_{i+1}^{(c)})/T_c. \tag{16}$$

After normalization, the lengths of the longer chains are shortened, whereas shorter chains are lengthened, i.e., longer chains move closer to the key points, and shorter chains move away from the key points. The normalized chains can be optionally scaled to a fixed size.

IV. CLUSTERING

A. BASIC ALGORITHMS

Prototype-based clustering methods consist of two main phases: selection of initial prototypes and iterative refinement until final convergence is reached, i.e., the cluster partition does not change (see Algorithm 4). In the classical

K-means [47], MacQueen's initialization phase is combined with Lloyd's search phase [48]. In general, the initialization phase is based on the random selection of initial prototypes, which most often causes the points to be selected from the same dense region yielding a poor performance [48]. Moreover, due to the initial point selections, it is known that the basic algorithm does not guarantee a unique solution to the global minimum of the error function [24]. Finding the global minimum is an NP-hard problem because there are Stirling number of the second kind different partitions for N observations into K groups [49]. In practice, the common way to perform clustering is to repeat the algorithm with multiple restarts and to use the smallest local clustering error as a selection criterion for the final prototypes [50].

The mean of the cluster points is the statistical estimate of the cluster prototype in K-means. The method assumes that data are spherical Gaussian distributed with normally distributed noise and equal variance in each cluster. The median and the spatial median, the latter also referred to as the Fermat-Weber or Weber point, are robust estimates of location [51], whose spherical symmetric distributions are uniform and Laplace distributions, respectively. The spatial median is a multivariate generalization of the univariate median. The median and the spatial median are robust prototypes of a data distribution since they can tolerate up to 50% of incorrect data values without being disturbed. The spatial median is rotation invariant so that robustness improves as the dimension of the continuous problem space grows [49].

Algorithm 4 The Main Phases of Prototype-Based Clustering

Input: Data set and the number of clusters K.
1. Select K observations as the initial prototypes.
until *the partition does not change* **do**
2. Assign each observation to the closest prototype.
3. Recompute the prototypes with the assigned observations.
Output: Partitions and prototypes corresponding K disjoint data subsets.

In the general case, the clustering error function can be written as follows:

$$\mathcal{J}_k = \sum_{\mathbf{x}_i \in C_k} d(\mathbf{x}_i, \mathbf{c}_k)_p^q, \quad \mathcal{J} = \sum_{k=1}^{K} \mathcal{J}_k, \tag{17}$$

where $d(\cdot)$ is the distance computation strategy in the l_p^q space, and $\{\mathbf{c}_k\}_{k=1}^K$ is the set of cluster prototypes that minimizes locally the error function (17) and partitions the data into K disjoint subsets. $\mathcal{J}_k$ is the within-cluster error in cluster C_k, and l_p-norm to the q-th power is the distance measure corresponding to the different location estimates of the error function (see [51], [52]). Specifically, the sample mean, median, and spatial median are obtained by choosing $(p = q = 2)$, $(p = q = 1)$, and $(p = 2, q = 1)$, respectively. The sample mean and the median are straightforward to compute, whereas the spatial median requires minimization of a non-smooth (i.e., nondifferentiable) optimization problem (see [52]) that requires more complex iterative methods to be computed [53]. For instance, the solution can be obtained efficiently by using the successive over-relaxation (SOR) method.

In the K-means++ initialization approach, the first prototype is selected as the centroid of the data set. Then, the following prototypes are selected iteratively from $\mathbf{X}$ based on the probability function obtained from previous prototype(s) $\min d(\mathbf{x}_i, \{\mathbf{c}\}_{k=1}^{K-1}) / \sum_{x_i \in C_k} \min d(\mathbf{x}_i, \{\mathbf{c}\}_{k=1}^{K-1})$. Thus, the initial prototypes are very probably selected separately. The selection procedure can also be performed incrementally [54]. It means that previously obtained $K - 1$ cluster prototypes are used as a fixed set of initial points where only one point is sampled according to the K-means++ principle. In high-dimensional problems, K-means++ may show deteriorating behavior which can be compensated by using dimension reduction techniques [33].

B. CLUSTERING BASED ON EXPECTED DISTANCES

Computing the expected distances rely on the assumption of normally distributed data. The central limit theorem suggests that the assumption is valid with many continuous data sets with appropriate sample sizes [37], [55]. However, the statistical parameters of data distribution are usually unknown, and missing values make estimating parameters more challenging. Usually, the EM algorithm can produce sufficiently accurate estimators of the unknown parameters, making the clustering task more approachable because the data characteristics are better known.

A clustering algorithm based on estimated distances was presented in [56]. The core steps of the method are shown in Algorithm 5. The algorithm skeleton is identical to the traditional clustering (see Algorithm 4) but consists of two additional steps (steps 2 and 3) that utilize distance estimation. Steps 4 and 5 are repeated with estimated distances until final convergence is reached. We noticed in the previous study that, on average (over 100 repetitions), clustering based on the distance estimation produced better initial prototypes than the clustering based on the ADS. However, giving the distance-estimated prototypes as the initial points to the K-spatialmedians based on ADS produced even more accurate solutions to the clustering tasks. Thus, step 6 was included in the developed method in Algorithm 5.

V. CLUSTER VALIDATION INDICES

Many clustering algorithms require the number of clusters as an input parameter. However, often this information is not available, and deciding the number can be challenging, especially in the case of multidimensional data, which humans cannot directly conceive. Even though there are many methods for illustrating multidimensional data, i.e., using different multidimensional visualization techniques [57] or dimension reduction techniques [58], [59], the data structure may not be obvious. Cluster validity provides a way to validate the quality of the clustering results by discovering the partition that best fits the nature of the data. Thus, because of the high diversity of data, cluster validation measures, e.g., CVIs, are

Algorithm 5 Clustering Based on EED-ADS Distance Computation

Input: Data set $\mathbf{X}$ with missing values and the number of clusters K.
1. Select the spatialmedian as the first prototype of the data set.
2. Iteratively select the initial prototypes by using previously selected $K-1$ prototypes and K-means++ initialization.
2. Compute the mean vector and covariance matrices of the data using the EM method.
3. Compute the expected distances between the observations and prototypes.
repeat
4. Assign individual observations to their closest prototypes.
5. Recompute the prototypes with the assigned observations.
until *The final convergence*
6. Repeat steps 4 and 5 without distance estimation.
Output: Partitions and prototypes corresponding K disjoint data subsets.

recommended, even essential, methods for determining the final number of clusters [31].

A. INTERNAL CLUSTER VALIDATION INDICES

Internal cluster validation indices are commonly based on two measures: 1) Compactness, also referred to as Intra, indicates how close the observations are to each other within the same cluster. A commonly used Intra is a clustering error itself, e.g., in the `Ray-Turi` index. 2) Separability, also known as Inter, indicates how distant a cluster is from the other clusters. Typically, Inter is computed as the minimum or maximum distance between all prototypes. Variability between prototypes around the centroid of the data is also used by many indices, e.g., in the `Calinski-Harabasz` index. In general, the purpose of CVIs is to minimize Intra and to maximize Inter, so that the argument minimum or maximum of division indicates the number of clusters.

Table 2 specifies the Inters and Intras of the best internal cluster validation indices according to [56]. Indices are presented in a general fashion for l_p^q-norm settings. Explanation of abbreviations are given in Table 3. The whole data prototype is denoted by $\mathbf{m}$, whereas n_k indicates the number of observations in the kth cluster. The special distance computation strategies given in Section II, denoted by $d(\cdot)$, are required if at least one data vector includes missing values. Note that the `WB-index`, `Calinski-Harabasz`, and `kCE-index` include penalization terms for a high number of clusters that were originally derived in the context of the squared formulas. Therefore, l_p^2-norms were used for these indices regardless of the clustering error criterion used. In the `Silhouette` index, Intra is the average dissimilarity of $\mathbf{x}_i$ to all other points in the same cluster, and Inter is the minimum average dissimilarity of $\mathbf{x}_i$ to all points in a different cluster:

$$\begin{aligned}\text{Intra}(\mathbf{x}_i) &= \frac{1}{n_k-1}\sum_{\mathbf{x}_j\in C_k} d(\mathbf{x}_i,\mathbf{x}_j),\\ \text{Inter}(\mathbf{x}_i) &= \min_{k\neq k'}\frac{1}{n_{k'}}\sum_{\mathbf{x}_j\in C_{k'}} d(\mathbf{x}_i,\mathbf{x}_j),\end{aligned} \tag{18}$$

where $\mathbf{x}_i$ belongs to cluster C_k.

TABLE 2. Internal cluster validation indices in general fashion.

Abbr	Intra	Inter	Formula
CH	$\mathcal{J}^2$	$\sum_{k=1}^{K} n_k\|\|\mathbf{c}_k-\mathbf{m}\|\|_p^2$	$\frac{K-1}{N-K}\times\frac{Intra}{Inter}$
DB	$\frac{\mathcal{J}_k}{n_k}+\frac{\mathcal{J}_{k'}}{n_{k'}}$	$\|\|\mathbf{c}_k-\mathbf{c}_{k^*}\|\|_p^q$	$\frac{1}{K}\sum_{k=1}^{K}\max_{k\neq k'}\frac{Intra(k,k')}{Inter(k,k')}$
DB*	$\frac{\mathcal{J}_k}{n_k}+\frac{\mathcal{J}_{k'}}{n_{k'}}$	$\|\|\mathbf{c}_k-\mathbf{c}_{k^*}\|\|_p^q$	$\frac{1}{K}\sum_{k=1}^{K}\frac{\max_{k\neq k'} Intra(k,k')}{\min_{k\neq k^*} Inter(k,k^*)}$
GD	$\max\frac{\mathcal{J}_k}{n_k}$	$\min_{k\neq k'}\|\|\mathbf{c}_k-\mathbf{c}_{k'}\|\|_p^q$	$\frac{2\times Intra}{Inter}$
KCE	$\mathcal{J}^2$	1	$K\times Intra$
PBM	$\mathcal{J}$	$\max_{k\neq k'}\|\|\mathbf{c}_k-\mathbf{c}_{k'}\|\|_p^q$	$(\frac{K\times Intra}{Inter})^2$
RT	$\frac{1}{N}\mathcal{J}$	$\min_{k\neq k'}\|\|\mathbf{c}_k-\mathbf{c}_{k'}\|\|_p^q$	$\frac{Intra}{Inter}$
SIL	See text	See text	$\frac{1}{N}\sum_{i=1}^{N}\frac{Inter(\mathbf{x}_i)-Intra(\mathbf{x}_i)}{\max(Intra(\mathbf{x}_i),Inter(\mathbf{x}_i))}$
WB	$\mathcal{J}^2$	$\sum_{k=1}^{K} n_k\|\|\mathbf{c}_k-\mathbf{m}\|\|_p^2$	$K\times\frac{Intra}{Inter}$
WG	$d(\mathbf{x}_i,\mathbf{c}_k)$	$\min_{k\neq k'} d(\mathbf{x}_i,\mathbf{c}_{k'})$	$\sum_{k=1}^{K}\sum_{\mathbf{x}_i\in C_k}\frac{Intra(\mathbf{x}_i)}{Inter(\mathbf{x}_i)}$

TABLE 3. Explanations of abbreviations.

Abbreviation	Name		
CH	DB	DB*	GD
Calinski-Harabasz	Davies-Bouldin	Davies-Bouldin*	Generalized Dunn
KCE	PBM	RT	SIL
kCE-index	Pakhira-Bandyopadhyay	Ray-Turi	Silhouette
WB	WG		
WB-index	Wemmert-Gançarski		

B. EXTERNAL CLUSTER VALIDATION INDICES

External cluster validation indices can validate the quality of the clustering result if the actual clustering labels are known. The simple external index is `Accuracy-index (ACC)` which computes the quotient of the correctly predicted data labels and the total number of the labels [60]. The `normalized mutual information index (NMI)` origins from information theory. The mutual information explains the reduction in the entropy between the real and the predicted cluster labels [61]. The normalization is used to scale the result to the range of [0, 1]. Many variants exist to normalize the mutual information, e.g., min, max, and square-root normalizations [61]. However, the arithmetic method is often used, which divides the mutual information by the average value of entropy terms as follows:

$$NMI = \frac{I(L_{real},L_{pred})}{(H(L_{real})+H(L_{pred}))/2}, \tag{19}$$

where $I(\cdot,\cdot)$ denotes the mutual information between the real and predicted clusters and $H(\cdot)$ denotes the entropy function. The `adjusted Rand index (ARI)` measures similarity between two clusterings of the same data using the permutation model [61]. The equation can be written as:

$$ARI = \frac{\Sigma_{ij}\binom{n_{ij}}{2} - [\Sigma_i\binom{a_i}{2}\Sigma_j\binom{b_j}{2}]/\binom{n}{2}}{\frac{1}{2}[\Sigma_i\binom{a_i}{2}+\Sigma_j\binom{b_j}{2}] - [\Sigma_i\binom{a_i}{2}\Sigma_j\binom{b_j}{2}]/\binom{n}{2}}, \tag{20}$$

where n_{ij} is an intersection table between the real and predicted cluster labels, the row and column sums of the intersection table are denoted by a_i and b_j, respectively.

VI. OVERVIEW OF THE TOOLBOX

The toolbox was implemented by using MATLAB (R2018b, 64-bit), and it is freely available with the MIT License on GitHub online[1]. The toolbox contains `benchmark_data`, `toolbox`, `test_macro`, and `results` folders.

Eight real-world classification data sets were selected from the University of California at Irvine (UCI)[2] Machine Learning Repository [62]. Seven were used in the first part of the experiments, and three were used in the second part. Further, eight synthetic data sets, including the four `S` sets[3] (15 centers and 5000 observations in each set), the `Sim5D2` set[4] (5 centers and 2970 observations), the `Sim2D2` set[4] (2 centers and 2000 observations), the `O200` set[4] (5 centers and 200 observations), and the `O2000` set[4] (5 centers and 2000 observations) were selected from a previous study [56] for the second part of the experiments. These synthetic data sets are two-dimensional. In addition, in total, 12 synthetic multidimensional data sets (10D, 50D, 100D) with 15 centers and 6000 observations in each set were created with the data set generator[5] [33] for the third part.

The toolbox includes routines for handling missing values, data preprocessing, clustering, and validating clusters. All the developed methods generalize to missing values in the data. The descriptions of the most vital MATLAB functions are given in Table 4. Notice that more detailed descriptions of functions are available using the `help` command in MATLAB (see the next section for the use case examples). Further, the `help` command shows the function calls, input and output parameters, and default values of the input parameters for each toolbox function. The toolbox supports computation strategies based on available data (ADS), partial distance (PDS), and expected distances (ESD, EED) that are used by clustering methods, cluster validation indices, and data preprocessing methods. In total, ten well-performing internal cluster validation indices depicted in Section V are supported. Further, as depicted in Sections II–III, the preprocessing functionality includes routines for data imputation, distance computation with a selected distance strategy, selecting key points, and transforming data sets into spherical forms.

A. GENERAL USE OF THE TOOLBOX

General use of the toolbox is demonstrated in the `toolboxdemo` macro (see the next section). The correct functionalities of the toolbox functions can be evaluated with test macros divided into three test case folders. The first test case folder includes the `Main` macro that performs comparisons of techniques for handling missing values. The macro selects the parameters used from the `params` file. The cluster validation process can be divided into three tasks in the second test case folder: data preparation, clustering, and cluster validation. The `Main` macro pipelines these tasks to one process and outputs an Excel file of the cluster validation results. Further, the toolbox offers missing values generation, clustering, and cluster validation as separate processes implemented in the `generatemissdata`, `clusterdata`, and `validateclustdata` macros, respectively. An optionally `visualizeresults` macro can be used to visualize the final results of the clustering and cluster validation.

The third test case folder includes the same `Main` macro functionalities as given in the second test case folder. However, the mechanism for generating missing values was modified to restore 0.5% of the original observations. It was required because the initialization of clustering uses the complete observations, and removing data values completely at random from high-dimensional data causes all observations to contain missing values.

[1] https://github.com/markoniem/nanclustering_toolbox
[2] https://archive.ics.uci.edu/ml/index.php
[3] http://cs.uef.fi/sipu/datasets/
[4] http://users.jyu.fi/ mapeniem/CVI/Data/
[5] https://github.com/jookriha/M_Spheres_Dataset_Generator

TABLE 4. Core functions.

Name	Description
normalizedata	Performs min-max or z-score normalization
genmissdata	Generates missing values to input data
knnimpute	Imputes missing values by using k-nearest neighbors
ICknnimpute	Utilizes Incomplete-Case k-nearest neighbors imputation
distancecalc	Computes distances of data with missing values
ecmnmlefunc	Obtains conditional means and variances of incomplete data
datasetmap	Transforms data approximately spherical symmetric
kcentroids	K-centroids clustering based on available data
kcentroids_partial	K-centroids clustering based on partial distances
kcentroids_expected	K-centroids clustering based on expected distances
scatter_results	Visualizes results of clustering
iterative_kcentroids	Clusters iteratively to the maximum number of clusters
keypointsComp	Computes selected number of key points
iterative_kcentroids_kp	Performs clustering iteratively based on selected key points
cluster_validation	Computes values of cluster validation indices
plot_indices	Plots curves of cluster validation indices

B. EXAMPLES OF BASIC USE

The basic use of the toolbox is given in the `toolboxdemo` file. It includes function calls for data preprocessing, clustering, and cluster validation. In the first example, 10% of missing values are generated for the input data. The result is min-max scaled to a range of [-1, 1], and the k-nearest neighbors imputation with five neighbors is performed. Then, the dimensionality of the imputed data set is reduced to 2D and transformed into a spherical form (Section III-D). Finally, the transformed data are visualized on a scatter plot.

```
load fisheriris;
X = meas;
addpath('../../../toolbox/preprocess');
Xm = genmissdata(X, 0.1);
Xnorm = normalizedata(Xm, 'min-max', [−1, 1]);
Ximp = knnimpute(Xnorm, 5);
Xmapped = datasetmap(Ximp);
scatter_data(Xmapped);
```

In the second example, clustering is performed based on available data in distance computation, i.e., using a

K-spatialmedians clustering method. The toolbox also supports clustering algorithms based on partial (kcentroids_partial) and expected (kcentroids_expected) distances. The clustered data set, the number of clusters, the number of replicates, the distance metric, the initialization criterion, and the initial values of the centroids are given as input parameters for the clustering function. The output parameters are cluster labels for each observation, the cluster centroids, and the within-cluster sums of points-to-centroid distances:

```
addpath('../../../toolbox/kcentroids');
[ L, C, sumd] = kcentroids(Xnorm, 5, 100, 'euc',
'kmeans++', [ ]);
```

In the final example, clustering is performed iteratively, with K ranging from 2 to 10. The default values of the parameters are used in clustering (see `help iterative_kcentroids`). The centroids and labels are used as input parameters for the cluster validation function. There are two ways to specify the indices (see the example). Finally, the results of the cluster validation indices are visualized:

```
addpath('../../../toolbox/cluster_indices');
% help iterative_kcentroids;
[ centers, labels] = iterative_kcentroids(Xnorm, 10);
% Select the cluster validation indices. The 'dist' parameter
% defines the selected distance metric used by inidices.
dist = 'euc';
indices = {@CalinskiHarabasz; @DaviesBouldin; @kCE;};
% An optional way to define indices. This overrides the 'dist' option.
indices = [ @CalinskiHarabasz, 'sqe'; @DaviesBouldin, 'euc'; @kCE,
'sqe'; ];
indices_values = cluster_validation(Xnorm, centers, labels, dist,
indices);
%
% In default, indices use available data strategy based computation.
% However, expected distances or partial distances are supported as well.
% indices_values =
%         cluster_validation(Xnorm, centers, labels, dist, indices, 'exp');
%
plot_indices(indices, indices_values);
```

VII. EXPERIMENTAL RESULTS

Experiments were divided into three parts which are discussed in the following sections.

A. VALIDATION OF DISTANCE ESTIMATION METHODS

In the first case, the experimental settings and the reference results were obtained from [5]. The real-world data sets were selected from the UCI repository. The experiments consisted of the z-score scaling of the data to the zero mean and unit variance. Then, the fixed probabilities (5, 15, 30, and 60%) of the data values were removed completely at random from each data set. The estimated distances were compared to the real distances, which were computed beforehand. The root mean square error (RMSE) between the real distances and the estimated distances was used. The RMSE included only the cases where estimations were needed, i.e., distances over complete observations were omitted. Further, in the cases of empty data vectors, the average distances over the data samples were used in error computing. The mean values and standard deviations of the results were recorded utilizing measurements over 250 repetitions.

We validated the functionalities of the implemented distance estimation algorithms against the reference methods given in [5]. An extension of the reference paper was the self-made implementation of the EM algorithm so that the `ecmnmle` function was not required (available only in MATLAB's commercial Financial Toolbox). Further, in addition to the ESD, PDS, and ICkNNI ($k = 5$) methods, the EED, ADS, kNNI ($k = 5$), and iterative soft-thresholding methods were added to the comparisons. Table 5 shows the results, which are in line with the reference results in the six cases over seven data sets. The exception is the `wine` data set, in which all distance computation mechanisms produced different results. In [5], a Monte Carlo simulation was used to remove data values in each repetition, whereas in our experiments, data values were removed completely at random. That may explain the differences in the results. In general, the results indicate that the EED is the best-performing algorithm. However, the ESD results are only slightly worse, and the method is computationally less expensive. Thus, the ESD method is highly recommended for computing pairwise distances.

B. PERFORMANCE EVALUATION OF CLUSTERING AND CLUSTER VALIDATION

In the second part, the data clustering and cluster validation indices methods were evaluated. The initial settings were selected from [56]. These settings included removing data values completely at random from data sets (see the toolbox overview section for detailed descriptions of the data sets), min-max scaling that results in a range of $[-1, 1]$, repeating the K-spatialmedians clustering with 100 replicates, and selecting the lowest local minima as the best clustering partition. The prototypes were initialized incrementally, benefiting the previous prototypes (see the last paragraph in Section IV-A). In [56], the clustering method based on the expected distances and giving the obtained prototypes as inputs in the K-spatialmedians with ADS algorithm was suggested. The clustering and cluster validation indices were revealed to be slightly more accurate based on the two-stage clustering approach. Thus, the same procedure was repeated in this study among the K-spatialmedians clustering.

The results given in [56] were reproduced to validate the functionality of the cluster validation. Note that the `results/params` folder includes the parameter files used in different experiments related to cluster validation. The experiments showed that the best cluster validation results are obtained using K-spatialmedians clustering based on EED-ADS distance estimation. The new approach improved the performance, especially when compared to the results, which were available using the real centers of the synthetic data sets as the initial points to K-spatialmedians based on the ADS (see `results` folder). The best-performing index was `Calinski-Harabasz (CH)` that always recommended

TABLE 5. The average RMSEs and standard deviations (over 250 repetitions) of distance computation algorithms in the direct estimation of pairwise distances with data sets consisting (5, 15, 30, and 60%) of missing values. The best results for each test set are underlined, and the results that are not statistically significantly different (two-tailed paired t-test, $\alpha = 0.01$) are in bold.

		EED	ESD	PDS	ADS	ICkNNI ($k = 5$)	kNNI ($k = 5$)	Soft-thresholding
Iris	5%	**0.240** (0.048)	**0.246** (0.045)	0.436 (0.049)	0.618 (0.062)	**0.244** (0.055)	0.261 (0.065)	0.294 (0.071)
($N = 150, n = 4$)	15%	**0.321** (0.042)	**0.329** (0.039)	0.592 (0.039)	0.875 (0.055)	0.335 (0.063)	0.493 (0.088)	0.401 (0.059)
	30%	**0.481** (0.043)	0.492 (0.039)	0.839 (0.030)	1.328 (0.045)	0.516 (0.064)	0.925 (0.060)	0.612 (0.057)
	60%	**0.920** (0.035)	0.931 (0.032)	1.208 (0.026)	2.262 (0.027)	1.175 (0.069)	1.420 (0.040)	1.247 (0.071)
Ecoli	5%	**0.450** (0.254)	**0.458** (0.250)	0.728 (0.180)	0.791 (0.193)	**0.439** (0.251)	**0.444** (0.247)	**0.469** (0.253)
($N = 336, n = 7$)	15%	**0.650** (0.238)	**0.661** (0.234)	1.057 (0.142)	1.170 (0.159)	**0.647** (0.246)	0.705 (0.235)	0.698 (0.241)
	30%	**0.960** (0.268)	**0.976** (0.259)	1.654 (0.121)	1.815 (0.142)	**0.990** (0.268)	1.142 (0.255)	1.081 (0.261)
	60%	**1.535** (0.275)	**1.563** (0.277)	2.431 (0.103)	3.004 (0.076)	1.710 (0.244)	1.774 (0.224)	1.884 (0.226)
Breast tissue	5%	**0.216** (0.099)	**0.217** (0.097)	0.428 (0.065)	0.568 (0.092)	0.244 (0.102)	**0.236** (0.103)	0.240 (0.110)
($N = 106, n = 9$)	15%	**0.347** (0.116)	**0.349** (0.114)	0.659 (0.067)	0.972 (0.106)	0.456 (0.137)	0.418 (0.139)	0.404 (0.128)
	30%	**0.584** (0.192)	**0.590** (0.190)	1.072 (0.070)	1.640 (0.110)	0.887 (0.149)	0.758 (0.164)	0.667 (0.142)
	60%	**1.183** (0.168)	**1.197** (0.165)	2.055 (0.112)	3.086 (0.101)	1.792 (0.272)	1.736 (0.204)	1.438 (0.198)
Glass	5%	**0.238** (0.086)	**0.242** (0.084)	0.539 (0.087)	0.650 (0.103)	0.361 (0.137)	0.363 (0.135)	0.328 (0.138)
($N = 214, n = 9$)	15%	**0.411** (0.082)	**0.420** (0.079)	0.809 (0.057)	1.055 (0.081)	0.569 (0.117)	0.564 (0.111)	0.546 (0.121)
	30%	**0.739** (0.097)	**0.755** (0.092)	1.330 (0.059)	1.758 (0.082)	0.972 (0.128)	0.962 (0.129)	0.934 (0.133)
	60%	**1.417** (0.079)	1.437 (0.073)	2.372 (0.070)	3.199 (0.057)	1.804 (0.150)	1.828 (0.128)	1.847 (0.134)
Wine	5%	**0.414** (0.227)	**0.416** (0.226)	0.613 (0.166)	0.701 (0.165)	**0.418** (0.210)	**0.421** (0.210)	**0.416** (0.212)
($N = 178, n = 13$)	15%	**0.702** (0.299)	**0.706** (0.296)	1.024 (0.161)	1.260 (0.166)	**0.738** (0.262)	**0.731** (0.262)	0.757 (0.263)
	30%	**1.048** (0.274)	**1.053** (0.269)	1.644 (0.104)	2.107 (0.135)	1.206 (0.268)	1.163 (0.277)	1.268 (0.268)
	60%	**1.512** (0.144)	**1.519** (0.133)	2.952 (0.152)	3.762 (0.064)	2.002 (0.230)	1.776 (0.227)	2.291 (0.195)
Parkinsons	5%	**0.181** (0.048)	**0.181** (0.047)	0.335 (0.031)	0.531 (0.039)	0.253 (0.049)	0.210 (0.048)	0.212 (0.050)
($N = 195, n = 22$)	15%	**0.318** (0.042)	**0.319** (0.042)	0.600 (0.029)	1.203 (0.050)	0.668 (0.062)	0.408 (0.053)	0.403 (0.051)
	30%	**0.511** (0.037)	**0.513** (0.036)	0.995 (0.035)	2.235 (0.067)	1.389 (0.231)	0.744 (0.075)	0.703 (0.055)
	60%	**0.970** (0.081)	**0.973** (0.081)	2.323 (0.076)	4.349 (0.076)	2.639 (0.123)	2.044 (0.180)	1.577 (0.114)
Sonar	5%	**0.191** (0.026)	**0.191** (0.026)	0.342 (0.023)	0.678 (0.031)	0.370 (0.038)	0.268 (0.038)	0.234 (0.036)
($N = 208, n = 60$)	15%	**0.465** (0.041)	**0.466** (0.041)	0.652 (0.031)	1.793 (0.041)	1.023 (0.053)	0.614 (0.056)	0.571 (0.053)
	30%	**0.786** (0.049)	**0.787** (0.049)	1.079 (0.034)	3.460 (0.049)	2.001 (0.063)	1.142 (0.070)	1.137 (0.069)
	60%	**1.435** (0.075)	**1.436** (0.075)	2.501 (0.056)	6.826 (0.049)	4.264 (0.082)	2.229 (0.103)	2.682 (0.105)

the correct numbers of clusters even if the data sets and degrees of missing values varied. The other well-performing indices were `kCE-index (KCE)`, `Silhouette`, and `Ray-Turi`.

Three external cluster validation indices were selected to measure the quality of the K-spatialmedians clustering results based on ADS and EED-ADS distance estimations strategies. The selected indices were: Accuracy (ACC), adjusted Rand index (ARI), and normalized mutual information (NMI). Table 6 shows the comparison results. Clearly, the EED-ADS-based estimation produces better solutions for the synthetic data sets. Especially, the better results were obtained with the `S2` data set and with the challenging `S4` and `Sim5D2` data sets. These results are in line with the results obtained by the internal indices, which especially recommended the better solutions with the EED-ADS estimation for the `Sim5D2` data set.

We applied the expected distance estimation to the actual cluster validation indices. It appeared that only `Silhouette`, `Wemmert-Gançarski (WG)`, and `Davies-Bouldin (DB)` benefited from the distance estimation, and the other indices decreased the performance for finding the correct number of clusters in the data sets. Compared to the other indices, which compute the pairwise distances between observations and complete centroids, `Silhouette` computes the pairwise distances between observations, which may be incomplete (see eq. (18)). Thus, it was expected that `Silhouette` performed better using the expected distances.

Key point selection (presented in Section III-D1) was used in the cluster validation. The number of key points can be fixed to $||\sqrt{N}||$, as recommended in [32]. However, we provided two modified versions of the original algorithm based on key point pruning, i.e., the algorithms started from the given maximum for the key points and then removed irrelevant points one by one. This was performed iteratively until the value of K of the chosen number of clusters was reached. The selected points were then used in the initialization of the selected clustering algorithm in each iteration. The experiments were performed for 2D data sets. For this purpose, `Ecoli`, `Iris`, and `Seeds` real-world data were transformed to 2D using multidimensional scaling. The selection assumed that the data sets were complete, therefore, the ICkNN ($k = 2$) imputation strategy was applied to the data sets with missing values. The figures for the key point selection result are given in `results/key_point_selection/img` folder in toolbox. The results for the cluster validation indices are given in Table 7. The reference results for the synthetic data sets were obtained from [56]. On average, the validation results for the key point selection were almost the same as the reference results, which were based on available data strategy and replicated clustering. The most challenging synthetic data set was `Sim5D2`. None of the indices was able to get all correct recommendations with the different degrees of

TABLE 6. The quality of clustering results determined with external cluster validation indices. The K-spatialmedians clusterings with available data strategy (ADS) and using both expected distance and available data computations (EED-ADS) were compared. The highest scores are bolded only if they differ between the two clustering methods.

ADS	ACC				ARI				NMI			
EED-ADS	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%
S1	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0,974	1.000	1.000	1.000	1.000
	1.000	1.000	1.000	1.000	1.000	1.000	1.000	**1.000**	1.000	1.000	1.000	1.000
S2	1.000	0.994	0.990	0.947	1.000	0.987	0.981	0.969	1.000	0.990	0.999	0.970
	1.000	**1.000**	**1.000**	**1.000**	1.000	**1.000**	**1.000**	**0.999**	1.000	**1.000**	**1.000**	**0.999**
S3	1.000	**1.000**	0.979	0.929	1.000	**0.999**	0.943	0.909	**1.000**	0.998	**1.000**	0.931
	1.000	0.999	**0.999**	**1.000**	1.000	0.998	**0.999**	**0.999**	0.999	0.998	0.999	**0.999**
S4	0.998	0.999	0.999	0.999	0.997	0.998	0.994	0.785	0.996	0.998	0.994	0.963
	0.998	**1.000**	**1.000**	0.999	0.997	**1.000**	**0.999**	**0.998**	0.996	**1.000**	**0.998**	**1.000**
Sim2D2	1.000	0.999	1.000	1.000	1.000	0.996	1.000	1.000	1.000	0.990	1.000	1.000
	1.000	**1.000**	1.000	1.000	1.000	**1.000**	1.000	1.000	1.000	**1.000**	1.000	1.000
Sim5D2	0.999	1.000	1.000	0.796	0.999	1.000	1.000	0.748	0.997	1.000	**1.000**	0.777
	1.000	1.000	1.000	**1.000**	0.999	1.000	1.000	**1.000**	0.997	1.000	0.998	**1.000**
O200	0.990	**1.000**	**1.000**	0.979	0.976	**1.000**	**0.986**	0.972	0.969	**1.000**	**1.000**	0.947
	0.990	0.990	0.990	**0.995**	0.976	0.971	0.978	**0.985**	0.969	0.973	0.970	**0.983**
O2000	1.000	1.000	**1.000**	0.877	1.000	1.000	**1.000**	0.737	1.000	1.000	1.000	0.762
	1.000	1.000	0.999	**1.000**	1.000	1.000	0.999	**1.000**	1.000	1.000	1.000	**1.000**

missing values. The high-density clusters in `Sim5D2` caused many incorrect validation results. The sparse clusters were connected to higher-density ones after clustering, and therefore, many indices supported three as the correct number. Especially, the sparse clusters almost disappeared based on the ICkNN imputed data with 20% of missing values (see images from `results` folder). The validation results with real-world data were improved using the key point selection with all data sets. It appeared that `CH`, `KCE`, and `WB` indices recommended a very high number of clusters for `Ecoli` and `Iris` data sets without the key point selection.

C. CLUSTER VALIDATION WITH MULTIDIMENSIONAL DATA

In the third part of the experiments, the cluster validation indices were applied to multidimensional data sets that were created by the data set generator presented in [33]. The generator draws a random point on the M-dimensional sphere centered on $\mathbf{c}$ with radius d. The distance between centers is defined as $d_c = ||\mathbf{c}_i - \mathbf{c}_j||$, $\mathbf{c}_i, \mathbf{c}_j \in C$, where $i \neq j$, and C is a set of centers. The radius d is uniformly selected from the range of (0, 1] for each data point. It means the clusters do not overlap in the multidimensional space when the distance of the centers is $d_c \geq 2$, and the cluster overlap is approximately 50% if the distance is $d_c = 1$.

Table 8 shows the results of the cluster validation indices with the predefined number of missing values (0, 5, 10, and 20%) and different degrees of cluster overlap (d_c = [0.9, 0.8, 0.7, 0.6]). The best performing index was `WG` which recommended the correct number of clusters in almost all test cases (45/48 correct recommendations). Interestingly, the `CH`, `KCE`, and `WB-index`, which included the squared penalization term, always recommended the incorrect number of clusters. We also tested the non-squared penalizations but were not able to improve the results. The `KCE` uses only Intra which explains that the better separation in the multidimensional space depends on the quality of Inter. It supports the finding given in [33] that the difference between the clustering errors of good and bad clustering results in high-dimensional spaces is small. The curse of dimensionality can explain the findings, which causes relative differences between the distances to vanish in high dimensional spaces [63]. The other well-performing indices were `GD`, `RT`, and `DB*`, which recommended 37, 33, and 30 correct solutions, respectively. The highest overlapping clusters ($d_c = 0.6$) were challenging for the indices because only `WG` (11/12 times), `RT` (3/12 times), and `GD` (3/12 times) were able to find the correct numbers.

The experiments were also conducted with 2D-scaled `M-Sphere` data sets. However, the performance of all indices was poor in 2D data space (only a few correct recommendations), and therefore, these results were not reported. The generated clusters were compact and isolated in the high-dimensional space, which explains the far better validation results with these data sets in their original dimensions [63]. Further, the dimension reduction leads to a loss of information which also supports the findings. Nevertheless, the developed key point selection algorithms with ICkNN ($k = 2$) imputed data possess multidimensional functionality. The results of cluster validation indices with the key point selection and multidimensional `M-Sphere` data sets are given in the `results` folder in the toolbox. The indices can be concluded to perform better when the key point selection procedure was used to initialize the K-spatialmedians clustering with 0%, 5%, and 10% of missing values in the data sets. However, a decreased performance was observed with 20% of missing values in data.

VIII. DISCUSSION

The results indicate that the ESD distance estimation could be a better choice than EED in the general case due to the lower computational complexity. The overall best clustering models with synthetic 2D data sets seem to be obtained using expected distances in clustering and giving the prototypes as inputs to the K-spatialmedians originally based on the

TABLE 7. The number of clusters determined with internal cluster validation indices based on key point selection (every second row). The data sets consisted of predefined numbers of missing values, and experiments were performed using the K-spatialmedians clustering algorithm. Reference results were obtained using K-spatialmedians clustering with available data strategy.

REF	CH				DB				DB*				GD				KCE			
KP	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%
S1	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
	15	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
S2	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
	15	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
S3	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	4	**15**	**15**	**15**	**15**	**15**	**15**	**15**
	15	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	4	4	4	4	**15**	**15**	**15**	**15**
S4	**15**	**15**	**15**	**15**	17	17	17	**15**	13	13	13	13	4	3	3	4	**15**	**15**	**15**	**15**
	15	**15**	**15**	**15**	**15**	**15**	14	14	11	**15**	14	14	5	3	4	3	**15**	**15**	**15**	**15**
Sim2D2	**2**	**2**	**2**	**2**	**2**	**2**	**2**	20	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**
	2	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**
Sim5D2	**5**	**5**	**5**	4	3	3	3	3	3	3	3	3	3	3	3	3	**5**	**5**	**5**	4
	4	3	4	3	3	3	3	3	3	3	3	3	3	3	3	3	4	**5**	**5**	4
O200	**5**	**5**	**5**	**5**	**5**	**5**	**5**	18	**5**	**5**	**5**	18	4	4	**5**	**5**	**5**	**5**	20	19
	5	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	4	4	4	**5**	4	4	**5**	**5**	**5**	**5**
O2000	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	4	4	4	**5**	**5**	**5**	6	**5**
	5	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	4	4	4	**5**	**5**	**5**	**5**	**5**
Ecoli	10	8	9	20	**3**	19	20	20	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	10	8	9	20
	2	2	2	2	**3**	**3**	**3**	20	**3**	**3**	**3**	2	**3**	**3**	**3**	2	**3**	**3**	**3**	**3**
Iris	17	20	17	5	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	17	20	17	20
	3	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	15	**3**	12	**3**
Seeds	2	2	2	16	2	2	2	2	2	2	2	2	2	2	2	2	**3**	**3**	**3**	20
	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	**3**	**3**	**3**	**3**
Total	**8**	**8**	**8**	**7**	**8**	**7**	**7**	**6**	**8**	**8**	**8**	**7**	**5**	**6**	**7**	**8**	**9**	**9**	**7**	**6**
	8	**8**	**8**	**8**	**9**	**9**	**8**	**7**	**8**	**9**	**7**	**6**	**5**	**6**	**5**	**5**	**9**	**11**	**10**	**10**
	PBM				**RT**				**SIL**				**WB**				**WG**			
	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20	0%	5%	10%	20%
S1	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
	15	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
S2	**15**	**15**	**15**	**15**	**15**	**15**	**15**	13	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
	15	**15**	**15**	**15**	**15**	**15**	**15**	13	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
S3	4	4	4	4	**15**	**15**	**15**	**15**	**15**	**15**	**15**	2	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
	4	4	4	4	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
S4	5	5	4	4	**15**	**15**	**15**	**15**	**15**	14	**15**	14	**15**	**15**	**15**	**15**	17	16	17	**15**
	5	5	5	4	**15**	**15**	14	14	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
Sim2D2	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	12	12	9	16	**2**	**2**	**2**	**2**
	2	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	12	8	8	12	**2**	**2**	**2**	**2**
Sim5D2	**5**	**5**	**5**	4	3	3	3	3	3	3	3	3	**5**	**5**	**5**	4	3	3	3	3
	4	**5**	**5**	3	3	3	3	3	3	3	3	3	4	**5**	**5**	**5**	3	3	3	3
O200	3	3	4	4	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	19	19	20	18	**5**	**5**	20	20
	4	**5**	6	4	**5**	**5**	4	**5**	**5**	**5**	**5**	**5**	**5**	16	18	16	**5**	**5**	**5**	**5**
O2000	3	4	4	4	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**
	4	4	4	4	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	6	**5**	**5**	**5**	**5**
Ecoli	10	8	9	16	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	10	10	9	20	**3**	**3**	**3**	**3**
	3	**3**	**3**	2	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	17	**3**	**3**	**3**	**3**
Iris	**3**	**3**	**3**	**3**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	17	20	19	20	**2**	**2**	**2**	**2**
	3	**3**	**3**	**3**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	15	14	13	12	**2**	**2**	**2**	**2**
Seeds	**3**	**3**	10	**3**	2	2	2	2	2	2	2	2	**3**	**3**	19	20	2	2	2	2
	3	**3**	**3**	**3**	2	2	2	2	2	2	2	2	**3**	**3**	**3**	**3**	2	2	2	2
Total	**6**	**6**	**5**	**5**	**9**	**9**	**9**	**8**	**9**	**8**	**9**	**7**	**7**	**7**	**6**	**5**	**8**	**8**	**7**	**8**
	6	**8**	**7**	**5**	**9**	**9**	**7**	**7**	**9**	**9**	**9**	**9**	**8**	**8**	**8**	**6**	**9**	**9**	**9**	**9**

available data distance strategy. In the case of multidimensional data, we noticed that the quality of the clustering models highly depends on the form of the Inter term. In addition, significantly better validation results were achieved when the data sets resided in their original dimensions than in 2D presentation. The WG index clearly overperformed the other indices on the multidimensional sets.

The experiments to demonstrate the performance of the cluster validation indices were performed both on synthetic and real-world data sets. One challenge for testing indices with real-world data sets is that the correct number of clusters is not obvious. For instance, a clustering model may produce a useful presentation about the inherent structure of a data set while it does not necessarily agree with the given class distribution for the same data. In data mining, the goal of cluster analysis is, however, to discover new knowledge instead of training a prediction model in a supervised manner. In this scenario, one approach for validating a cluster model and estimating the number of clusters is to apply multiple indices that have previously performed well on several data sets.

We provided two modified versions of the original key point selection algorithm based on key point pruning. The developed algorithms included a mechanism for removing irrelevant key points. The algorithms resulted in good solutions for most of the data sets with a varying portion of missing values. However, there is still room for improvement in the heuristics to identify appropriate locations of the key points for diverse data sets. The development of clustering heuristics is not a trivial task because the notion of a cluster itself can be weakly defined [64]. It is also good to remember

TABLE 8. The number of clusters determined with internal cluster validation indices using K-spatialmedians clustering. The data sets consisted of predefined number of missing values, and different degrees of cluster overlap.

	CH				DB				DB*				GD				KCE			
	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%
M10-dc0.9	2	2	2	2	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	2	2	2	2
M10-dc0.8	2	2	2	2	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	2	2	2	2
M10-dc0.7	4	4	4	4	8	10	10	10	7	10	10	10	**15**	**15**	**15**	**15**	2	2	2	2
M10-dc0.6	2	2	2	2	9	9	9	9	14	14	14	9	3	3	3	3	2	2	2	2
M50-dc0.9	3	3	3	3	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	2	2	2	2
M50-dc0.8	2	2	2	2	13	13	13	13	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	2	2	2	2
M50-dc0.7	3	3	3	3	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	2	2	2	2
M50-dc0.6	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
M100-dc0.9	2	2	2	2	**15**	**15**	14	**15**	**15**	**15**	9	**15**	**15**	**15**	9	**15**	2	2	2	2
M100-dc0.8	2	2	2	2	**15**	**15**	14	**15**	**15**	**15**	14	**15**	**15**	**15**	12	**15**	2	2	2	2
M100-dc0.7	2	2	2	2	13	13	14	14	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	2	2	2	2
M100-dc0.6	3	3	3	3	13	11	10	11	6	3	3	3	**15**	**15**	3	**15**	2	2	2	2
Total	**0**	**0**	**0**	**0**	**6**	**6**	**4**	**6**	**8**	**8**	**6**	**8**	**10**	**10**	**7**	**10**	**0**	**0**	**0**	**0**
	PBM				**RT**				**SIL**				**WB**				**WG**			
	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20%	0%	5%	10%	20	0%	5%	10%	20%
M10-dc0.9	3	3	3	3	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	5	5	5	5	**15**	**15**	**15**	**15**
M10-dc0.8	3	2	2	2	**15**	**15**	**15**	**15**	8	8	8	**15**	7	7	7	8	**15**	**15**	**15**	**15**
M10-dc0.7	2	2	2	2	4	4	4	4	**15**	6	**15**	**15**	5	5	5	5	**15**	**15**	**15**	**15**
M10-dc0.6	2	2	2	2	9	9	9	13	3	3	3	3	3	3	3	3	**15**	**15**	**15**	**15**
M50-dc0.9	2	2	2	2	**15**	**15**	**15**	**15**	**15**	**15**	14	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**
M50-dc0.8	2	2	2	2	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	4	4	4	4	**15**	**15**	**15**	**15**
M50-dc0.7	2	2	2	2	**15**	**15**	**15**	**15**	**15**	**15**	14	12	5	5	5	5	**15**	**15**	**15**	**15**
M50-dc0.6	2	2	2	2	2	2	2	2	2	2	2	2	4	4	4	4	**15**	**15**	**15**	**15**
M100-dc0.9	2	2	2	2	**15**	**15**	14	**15**	**15**	**15**	16	**15**	5	5	5	5	**15**	**15**	14	**15**
M100-dc0.8	2	2	2	2	**15**	**15**	14	**15**	**15**	**15**	16	14	4	4	4	4	**15**	**15**	16	**15**
M100-dc0.7	2	2	2	2	**15**	**15**	**15**	**15**	**15**	**15**	**15**	**15**	3	3	4	3	**15**	**15**	**15**	**15**
M100-dc0.6	2	3	3	3	**15**	**15**	3	**15**	6	6	5	6	5	5	5	5	**15**	**15**	14	**15**
Total	**0**	**0**	**0**	**0**	**9**	**9**	**6**	**9**	**8**	**7**	**5**	**7**	**1**	**1**	**1**	**1**	**12**	**12**	**9**	**12**

that clustering is often in the eye of the beholder [65]. Before a clustering algorithm is applied to the data, one may also want to determine whether the data even has a clustering tendency [66]. The most central properties of clusters are density, variance, dimension, shape, and separation [67]. Further, what type of clustering model is the most useful always depends on the target application.

IX. CONCLUSION

Even though the basic idea behind cluster analysis is simple, the process presumes many decisions and choices with multiple options in different parts of the analysis. This study proposed a toolbox that enables researchers and practitioners to achieve reliable and consistent clustering results regardless of missing values in their data. The priorities of the present work were on data preprocessing, clustering, and cluster validation.

The toolbox supports missing values and enables its user to build automated data clustering pipelines from preprocessing to cluster analysis and model validation. The validity and performance of the algorithms were demonstrated using multiple test cases and several data sets. One should note that the aim of the presented experiments was not to perform a systematic method comparison since most of the underlying development work has already been accomplished in the previous studies cited in this paper.

We remind that some of the implemented functions can also be useful in other machine learning tasks. For instance, the distance computation methods for missing data cases provided in the preprocessing folder are readily applicable in supervised learning with the distance-based methods [68], [69].

The functionality of the toolbox was verified against the reference results from the previous publications. In the study, the two expected distances measuring metrics' performance were thoroughly demonstrated in handling missing values. Further, a recently published key point selection mechanism, which associates the data points with relatively higher density and larger density-based distances to the so-called key points, was applied to improve the cluster validation process. The cluster validation was experimented with challenging multidimensional data sets with various cluster overlap and numbers of missing values.

Even though the key point selection strategy seems to improve the performance of many cluster validation indices, further investigations are recommended, especially related to the key point selection procedure and the initialization of clustering algorithms. The initialization is an important part of the clustering process, and several studies are already available on the topic [33], [70]–[72]. The purpose of this toolbox is to facilitate and promote this research further. The UCI Repository provides a multitude of data sets, of which some are particularly proposed for clustering experiments. This toolbox enhances the testing of its methods with a wider range of sets.

APPENDIX A
EXPECTED SQUARED EUCLIDEAN DISTANCE

Let us assume the data are missing at random (MAR), i.e., missingness may depend on the value of available data:

$$P(M|\mathbf{x}_{avail}, \mathbf{x}_{miss}) = P(M|\mathbf{x}_{avail}). \quad (21)$$

The expected squared Euclidean distance between two data vectors can be partitioned into four parts depending on the missing and available values of each data vector:

$$\begin{aligned} E\left[||\mathbf{x}_i - \mathbf{x}_j||^2\right] &= \sum_{l \in A_i \cap A_j} ((\mathbf{x}_i)_l - (\mathbf{x}_j)_l)^2 + \sum_{l \in A_i \cap M_j} E[((\mathbf{x}_i)_l - (X_j)_l)^2] \\ &+ \sum_{l \in M_i \cap A_j} E[((X_i)_l - (\mathbf{x}_j)_l)^2] \\ &+ \sum_{l \in M_i \cap M_j} E[((X_i)_l - (X_j)_l)^2], \end{aligned} \tag{22}$$

where A_i and A_j denote the available values of data vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively, and M_i and M_j denote the missing values of the vectors. The first term ($l \in A_i \cap A_j$) represents pairwise known values of both vectors, and they can be computed directly. The rest of the sum contains terms where at least one part contains only missing values. The missing value can be replaced with a random value, i.e., $(\mathbf{x}_i)_l$ is denoted by $(X_i)_l$ for every $l \in M_i$. Thus, the equation can be expanded as follows:

$$\begin{aligned} E\left[||\mathbf{x}_i - \mathbf{x}_j||^2\right] &= \sum_{l \in A_i \cap A_j} \Big((\mathbf{x}_i)_l - (\mathbf{x}_j)_l\Big)^2 \\ &+ \sum_{l \in A_i \cap M_j} \Big(((\mathbf{x}_i)_l - E[(X_j)_l])^2 + Var[(X_j)_l]\Big) \\ &+ \sum_{l \in M_i \cap A_j} \Big((E[(X_i)_l] - (\mathbf{x}_j)_l)^2 + Var[(X_i)_l]\Big) \\ &+ \sum_{l \in M_i \cap M_j} \Big((E[(X_i)_l] - E[(X_j)_l])^2 \\ &+ Var[(X_i)_l] + Var[(X_j)_l]\Big). \end{aligned} \tag{23}$$

In more detail, the third summation ($l \in M_i \cap M_j$) can be written as:

$$\begin{aligned} &E[((X_i)_l - (X_j)_l)^2] \\ &= E[((X_i)_l)^2 - 2((X_i)_l)((X_j)_l) + ((X_j)_l)^2] \\ &= E[((X_i)_l)^2] - 2E[((X_i)_l)]E[((X_j)_l)] + E[((X_j)_l)^2] \\ &\quad + E[((X_i)_l)]^2 - E[((X_i)_l)]^2 + E[((X_j)_l)]^2 - E[((X_j)_l)]^2 \\ &= (E[((X_i)_l)] - E[((X_j)_l)])^2 + E[E[(X_i)_l^2] - (X_i)_l^2] \\ &\quad + E[E[(X_j)_l^2] - (X_j)_l^2] \\ &= (E[((X_i)_l)] - E[((X_j)_l)])^2 + Var((X_i)_l) + Var((X_j)_l). \end{aligned} \tag{24}$$

Thus, it is sufficient to compute the expected value and variance of each random value separately to obtain the final distance.

APPENDIX B
CONDITIONAL MEAN AND COVARIANCE

Let us assume multivariate normally distributed data which are partitioned as $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix}$ and define a linear combination $\mathbf{x} = \mathbf{x}_1 + \mathbf{A}\mathbf{x}_2$, where $\mathbf{A} = -\Sigma_{12}\Sigma_{22}^{-1}$. Now, we notice the following equality:

$$\begin{aligned} Cov[\mathbf{x}, \mathbf{x}_2] &= Cov[\mathbf{x}_1, \mathbf{x}_2] + Cov[\mathbf{A}\mathbf{x}_2, \mathbf{x}_2] \\ &= \Sigma_{12} + \mathbf{A}Var[\mathbf{x}_2] \\ &= \Sigma_{12} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{22} \\ &= \mathbf{0}. \end{aligned}$$

Thus, $\mathbf{x}$ and $\mathbf{x}_2$ are uncorrelated. In addition, they are jointly normally distributed, and therefore, independent. Following the initial assumptions, the conditional mean of $\mathbf{x}_1$ given $\mathbf{x}_2$ is obtained as follows:

$$\begin{aligned} E[\mathbf{x}_1|\mathbf{x}_2] &= E[\mathbf{x} - \mathbf{A}\mathbf{x}_2|\mathbf{x}_2] \\ &= E[\mathbf{x}|\mathbf{x}_2] - E[\mathbf{A}\mathbf{x}_2|\mathbf{x}_2] \\ &= E[\mathbf{x}] - \mathbf{A}\mathbf{x}_2 \\ &= \boldsymbol{\mu}_1 + \mathbf{A}(\boldsymbol{\mu}_2 - \mathbf{x}_2) \\ &= \boldsymbol{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2). \end{aligned}$$

Further, we find out the following equality:

$$\begin{aligned} Var[\mathbf{x}_1|\mathbf{x}_2] &= Var[\mathbf{x} - \mathbf{A}\mathbf{x}_2|\mathbf{x}_2] \\ &= Var[\mathbf{x}|\mathbf{x}_2] + Var[-\mathbf{A}\mathbf{x}_2|\mathbf{x}_2] \\ &\quad + Cov[\mathbf{x}, -\mathbf{A}\mathbf{x}_2] + Cov[-\mathbf{A}\mathbf{x}_2, \mathbf{x}] \\ &= Var[\mathbf{x}|\mathbf{x}_2] + \mathbf{A}Var[\mathbf{x}_2|\mathbf{x}_2]\mathbf{A}^T \\ &\quad - Cov[\mathbf{x}, \mathbf{x}_2]\mathbf{A}^T - \mathbf{A}Cov[\mathbf{x}_2, \mathbf{x}] \\ &= Var[\mathbf{x}]. \end{aligned}$$

Therefore, the conditional variance is defined as:

$$\begin{aligned} Var[\mathbf{x}_1|\mathbf{x}_2] &= Var[\mathbf{x}_1 + \mathbf{A}\mathbf{x}_2] \\ &= Var[\mathbf{x}_1] + \mathbf{A}Var[\mathbf{x}_2]\mathbf{A}^T \\ &\quad + Cov[\mathbf{x}_1, \mathbf{x}_2]\mathbf{A}^T + \mathbf{A}Cov[\mathbf{x}_2, \mathbf{x}_1] \\ &= \Sigma_{11} + \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{22}\Sigma_{22}^{-1}\Sigma_{21} \\ &\quad - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \\ &= \Sigma_{11} + \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} - 2\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \\ &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}. \end{aligned}$$

Note that the basic rules of matrix algebra are given in [73].

ACKNOWLEDGMENT

The authors acknowledge Joonas Hämäläinen, Ph.D., for his help in preparing the experiments reported in Table 8.

REFERENCES

[1] J. Kim, D. Tae, and J. Seok, "A survey of missing data imputation using generative adversarial networks," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2020, pp. 454–456.

[2] F. Biessmann, T. Rukat, P. Schmidt, P. Naidu, S. Schelter, A. Taptunov, D. Lange, and D. Salinas, "DataWig: Missing value imputation for tables," *J. Mach. Learn. Res.*, vol. 20, no. 175, pp. 1–6, 2019.

[3] J. K. Dixon, "Pattern recognition with partly missing data," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 10, pp. 617–621, Oct. 1979.
[4] J. C. Gower, "A general coefficient of similarity and some of its properties," *Biometrics*, vol. 27, no. 4, pp. 857–871, 1971.
[5] E. Eirola, G. Doquire, M. Verleysen, and A. Lendasse, "Distance estimation in numerical data sets with missing values," *Inf. Sci.*, vol. 240, pp. 115–128, Aug. 2013.
[6] D. P. P. Mesquita, J. P. P. Gomes, A. H. S. Junior, and J. S. Nobre, "Euclidean distance estimation in incomplete datasets," *Neurocomputing*, vol. 248, pp. 11–18, Jul. 2017.
[7] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.
[8] S. Narayanan, R. J. Marks, J. L. Vian, J. J. Choi, M. A. El-Sharkawi, and B. B. Thompson, "Set constraint discovery: Missing sensor data restoration using autoassociative regression machines," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2002, pp. 2872–2877.
[9] M. Abdella and T. Marwala, "The use of genetic algorithms and neural networks to approximate missing data in database," in *Proc. IEEE 3rd Int. Conf. Comput. Cybern. (ICCC)*, Apr. 2005, pp. 207–212.
[10] I. B. Aydilek and A. Arslan, "A hybrid method for imputation of missing values using optimized fuzzy c-means with support vector regression and a genetic algorithm," *Inf. Sci.*, vol. 233, pp. 25–35, Jun. 2013.
[11] M. G. Rahman and M. Z. Islam, "Missing value imputation using decision trees and decision forests by splitting and merging records: Two novel techniques," *Knowl.-Based Syst.*, vol. 53, pp. 51–65, Nov. 2013.
[12] L. Tran, X. Liu, J. Zhou, and R. Jin, "Missing modalities imputation via cascaded residual autoencoder," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1405–1414.
[13] N. Abiri, B. Linse, P. Edén, and M. Ohlsson, "Establishing strong imputation performance of a denoising autoencoder in a wide range of missing data problems," *Neurocomputing*, vol. 365, pp. 137–146, Nov. 2019.
[14] J. Zhao, Y. Nie, S. Ni, and X. Sun, "Traffic data imputation and prediction: An efficient realization of deep learning," *IEEE Access*, vol. 8, pp. 46713–46722, 2020.
[15] L. Li, M. Franklin, M. Girguis, F. Lurmann, J. Wu, N. Pavlovic, C. Breton, F. Gilliland, and R. Habre, "Spatiotemporal imputation of MAIAC AOD using deep learning with downscaling," *Remote Sens. Environ.*, vol. 237, Feb. 2020, Art. no. 111584.
[16] M. Sangeetha and M. S. Kumaran, "Deep learning-based data imputation on time-variant data using recurrent neural network," *Soft Comput.*, vol. 24, no. 17, pp. 13369–13380, Sep. 2020.
[17] S. Ryu, M. Kim, and H. Kim, "Denoising autoencoder-based missing value imputation for smart meters," *IEEE Access*, vol. 8, pp. 40656–40666, 2020.
[18] Y. Lin, Y. Gou, Z. Liu, B. Li, J. Lv, and X. Peng, "COMPLETER: Incomplete multi-view clustering via contrastive prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 11174–11183.
[19] G. Dong, G. Liao, H. Liu, and G. Kuang, "A review of the autoencoder and its variants: A comparative perspective from target recognition in synthetic-aperture radar images," *IEEE Geosci. Remote Sens. Mag.*, vol. 6, no. 3, pp. 44–68, Sep. 2018.
[20] H.-P. Kriegel, P. Kröger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 1, no. 3, pp. 231–240, May/Jun. 2011.
[21] J. Tou and R. Gonzalez, *Pattern Recognition Principles*. Reading, MA, USA: Addison-Wesley, 1974.
[22] W. R. Dillon and M. Goldstein, *Multivariate Analysis: Methods and Applications*. Hoboken, NJ, USA: Wiley, 1984.
[23] M. R. Anderberg, *Cluster Analysis for Applications*. New York, NY, USA: Academic, 1973.
[24] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, Jun. 2010.
[25] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, 1988.
[26] C. K. Reddy and B. Vinzamuri, "A survey of partitional and hierarchical clustering algorithms," in *Data Clustering*. Boca Raton, FL, USA: CRC Press, 2018, pp. 87–110.
[27] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Hoboken, NJ, USA: Wiley, 1990.
[28] J. Han, M. Kamber, and A. Tung, *Spatial Clustering Methods in Data Mining: A Survey*. New York, NY, USA: Taylor and Francis, 2001, pp. 188–217.
[29] J. Hämäläinen, S. Jauhiainen, and T. Kärkkäinen, "Comparison of internal clustering validation indices for prototype-based clustering," *Algorithms*, vol. 10, no. 3, p. 105, Sep. 2017.
[30] M. J. Zaki and W. Meira, *Data Mining and Analysis: Fundamental Concepts*. Cambridge, U.K.: Cambridge Univ. Press, 2014.
[31] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, and J. M. Pérez, and I. Perona, "An extensive comparative study of cluster validity indices," *Pattern Recognit.*, vol. 46, no. 1, pp. 243–256, Jan. 2013.
[32] Q. Li, S. Yue, and M. Ding, "Volume and surface area-based cluster validity index," *IEEE Access*, vol. 8, pp. 24170–24181, 2020.
[33] J. Hämäläinen, T. Kärkkäinen, and T. Rossi, "Improving scalable K-means++," *Algorithms*, vol. 14, no. 1, p. 6, Dec. 2020.
[34] T. Kärkkäinen and J. Toivanen, "Building blocks for odd–even multigrid with applications to reduced systems," *J. Comput. Appl. Math.*, vol. 131, nos. 1–2, pp. 15–33, Jun. 2001.
[35] M. Saarela and T. Karkkainen, "Discovering gender-specific knowledge from Finnish basic education using Pisa scale indices," in *Proc. 7th Int. Conf. Educ. Data Mining*, 2014, pp. 60–67.
[36] M. Niemela, S. Ayramo, and T. Karkkainen, "Comparison of cluster validation indices with missing data," in *Proc. 26th Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn. (ESANN)*, 2018, pp. 461–466.
[37] H. Fischer, "A history of the central limit theorem," in *Sources and Studies in the History of Mathematics and Physical Sciences*. New York, NY, USA: Springer, 2011.
[38] T. Karkkainen and M. Saarela, "Robust principal component analysis of data with missing values," in *Machine Learning and Data Mining in Pattern Recognition*, P. Perner, Ed. Cham, Switzerland: Springer, 2015, pp. 140–154.
[39] E. Eirola, A. Lendasse, V. Vandewalle, and C. Biernacki, "Mixture of Gaussians for distance estimation with missing data," *Neurocomputing*, vol. 131, pp. 32–42, May 2014.
[40] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *J. Roy. Stat. Soc., Ser. B, Methodol.*, vol. 39, no. 1, pp. 1–22, 1977.
[41] F. Dellaert, "The expectation maximization algorithm," in *College of Computing*. Atlanta, GA, USA: Georgia Institute of Technology, 2003.
[42] J. Van Hulse and T. M. Khoshgoftaar, "Incomplete-case nearest neighbor imputation in software measurement data," *Inf. Sci.*, vol. 259, pp. 596–610, Feb. 2014.
[43] A. Majumdar and R. K. Ward, "Some empirical advances in matrix completion," *Signal Process.*, vol. 91, no. 5, pp. 1334–1338, 2011.
[44] R. Mazumder, T. Hastie, and R. Tibshirani, "Spectral regularization algorithms for learning large incomplete matrices," *J. Mach. Learn. Res.*, vol. 11, no. 3, pp. 2287–2322, Aug. 2019.
[45] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.
[46] A. Mead, "Review of the development of multidimensional scaling methods," *J. Roy. Stat. Soc., D Statistician*, vol. 41, no. 1, pp. 27–39, 1992.
[47] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. IT-28, no. 2, pp. 129–136, Mar. 1982.
[48] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Syst. Appl.*, vol. 40, no. 1, pp. 200–210, Jan. 2013.
[49] S. Ayramo and T. Karkkainen, "Introduction to partitioning-based clustering methods with a robust example," in *Reports of the Department of Mathematical Information Technology Series C. Software and Computational Engineering*. Jyväskylä, Finland: Univ. Jyvaskyla, 2006.
[50] R. Xu and D. C. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, Jun. 2005.
[51] S. Ayramo, *Knowledge Mining Using Robust Clustering. Jyvaskyla Studies in Computing*, vol. 63. Jyväskylä, Finland: Univ. Jyvaskyla, 2006.
[52] T. Karkkainen and E. Heikkola, "Robust formulations for training multilayer perceptrons," *Neural Comput.*, vol. 16, pp. 837–862, Apr. 2004.
[53] T. Karkkainen and S. Ayramo, "On computation of spatial median for robust data mining," in *Evolutionary and Deterministic Methods for Design Optimization and Control With Applications to Industrial and Societal Problems*. Munich, Germany: EU-ROGEN, 2005.
[54] E. Lughofer, "A dynamic split-and-merge approach for evolving cluster models," *Evolving Syst.*, vol. 3, no. 3, pp. 135–151, Sep. 2012.

[55] S. Shatskikh and L. E. Melkumova, "Normality assumption in statistical data analysis," in *Proc. CEUR Workshop*, vol. 1638, 2016, pp. 763–768.
[56] M. Niemelä and T. Kärkkäinen, "Improving clustering and cluster validation with missing data using distance estimation methods," in *Computational Sciences and Artificial Intelligence in Industry: New Digital Technologies for Solving Future Societal and Economical Challenges*, T. Tuovinen, J. Periaux, and P. Neittaanmäki, Eds. Springer, 2022, pp. 123–133.
[57] M. C. F. D. Oliveira and H. Levkowitz, "From visual data exploration to visual data mining: A survey," *IEEE Trans. Vis. Comput. Graphics*, vol. 9, no. 3, pp. 378–394, Jul. 2003.
[58] C. Boutsidis, A. Zouzias, and P. Drineas, "Random projections for k-means clustering," in *Proc. Adv. Neural Inf. Process. Syst., 24th Annu. Conf. Neural Inf. Process. Syst.*, 2010, pp. 298–306.
[59] I. Jolliffe, *Principal Component Analysis*, 2nd ed. Cham, Switzerland: Springer, 2002.
[60] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. Patel, A. Tiwari, M. Er, W. Ding, and C. T. Lin, "A review of clustering techniques and developments," *Neurocomputing*, vol. 267, pp. 664–681, Dec. 2017.
[61] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, Jan. 2010.
[62] D. Dua and C. Graff. UCI Machine Learning Repository. School of Information and Computer Sciences. University of California, Irvine. Accessed: 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[63] M. Verleysen and D. François, "The curse of dimensionality in data mining and time series prediction," in *Proc. Int. Work-Conf. Artif. Neural Netw.*, vol. 3512, 2005, pp. 758–770.
[64] P. N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Reading, MA, USA: Addison-Wesley, 2005.
[65] V. Estivill-Castro, "Why so many clustering algorithms: A position paper," *ACM SIGKDD Explor. Newslett.*, vol. 4, no. 1, pp. 65–75, Jun. 2002.
[66] S. P. Smith and A. K. Jain, "Testing for uniformity in multidimensional data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 1, pp. 73–81, Jan. 1984.
[67] M. S. Aldenderfer and R. K. Blashfield, *Cluster Analysis*. Newbury Park, CA, USA: Sage, 1984.
[68] T. Kärkkäinen, "Extreme minimal learning machine: Ridge regression with distance-based basis," *Neurocomputing*, vol. 342, pp. 33–48, May 2019.
[69] J. Hämäläinen, A. S. C. Alencar, T. Kärkkäinen, C. L. C. Mattos, A. H. S. Júnior, and J. P. P. Gomes, "Minimal learning machine: Theoretical results and clustering-based reference point selection," *J. Mach. Learn. Res.*, vol. 21, pp. 1–29, Oct. 2020.
[70] S. S. Khan and A. Ahmad, "Cluster center initialization algorithm for K-means clustering," *Pattern Recognit. Lett.*, vol. 25, no. 11, pp. 1293–1302, 2004.
[71] M. Marina and H. David, "An experimental comparison of several clustering and initialization methods," in *Proc. 14th Annu. Conf. Uncertainty Artif. Intell. (UAI)*. San Francisco, CA, USA: Morgan Kaufmann, 1998, pp. 386–395.
[72] S. Ayramo, T. Karkkainen, and K. Majava, "Robust refinement of initial prototypes for partitioning-based clustering algorithms," in *Recent Advances in Stochastic Modeling and Data Analysis*. Singapore: World Scientific, 2007, pp. 473–482.
[73] K. B. Petersen and M. S. Pedersen. The Matrix Cookbook. Technical University of Denmark. Accessed: 2012. [Online]. Available: https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf

MARKO NIEMELÄ received the M.Sc. (technology) degree in computer science and engineering from the University of Oulu, in 2013. He is currently working with the Faculty of Information Technology, University of Jyväskylä, Finland. His main research interests include machine learning, data mining, data analytics, and optimization.

SAMI ÄYRÄMÖ received the Ph.D. degree in mathematical information technology, in 2006. He is currently an Adjunct Professor of data analytics at the University of Jyväskylä. His research interests include machine learning and predictive modeling with a special focus on applications in sport, health, and medicine.

TOMMI KÄRKKÄINEN (Senior Member, IEEE) received the Ph.D. degree in mathematical information technology from the University of Jyväskylä (JYU), in 1995. Since 2002, he has been a Full Professor of mathematical information technology with the Faculty of Information Technology (FIT), JYU. He currently leads the Research Division and the Research Group on human and machine-based intelligence in learning. He has served in many administrative positions at FIT and JYU. He has led nearly 50 different research and development projects. He has been/is involved in supervising 57 Ph.D. students. He has published about 200 peer-reviewed articles. His current research interests include data mining, machine learning, learning analytics, and computing education research. He received the JYU Innovation Prize, in 2010.