Information Sciences 393 (2017) 29–46

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Using the stability of objects to determine the number of clusters in datasets

Etienne Lord[a,b], Matthieu Willems[a], François-Joseph Lapointe[b], Vladimir Makarenkov[a,*]

[a] Département d'informatique, Université du Québec à Montréal, C.P. 8888, Succ. Centre-Ville, Montréal (QC) H3C 3P8 Canada
[b] Département de sciences biologiques, Université de Montréal, C.P. 6128, Succ. Centre-Ville, Montréal (QC) H3C 3J7 Canada

ARTICLE INFO

Article history:
Received 21 February 2016
Revised 13 January 2017
Accepted 4 February 2017
Available online 6 February 2017

Keywords:
Cluster stability
Cluster validity indices
K-means
K-medoids
Number of clusters
Partitioning algorithms
Stability of individual objects in clustering

ABSTRACT

We introduce a novel method for assessing the robustness of clusters found by partitioning algorithms. First, we show how the stability of individual objects can be estimated based on repeated runs of the *K*-means and *K*-medoids algorithms. The quality of the resulting clusterings, expressed by the popular Calinski–Harabasz, Silhouette, Dunn and Davies–Bouldin cluster validity indices, is taken into account when computing the stability estimates of individual objects. Second, we explain how to assess the stability of individual clusters of objects and sets of clusters that are found by partitioning algorithms. Finally, we present a new and effective stability-based algorithm that improves the ability of traditional partitioning methods to determine the number of clusters in datasets. We compare our algorithm to some well-known cluster identification techniques, including *X*-means, Pvclust, Adegenet, Prediction Strength and Nselectboot. Our experiments with synthetic and benchmark data demonstrate the effectiveness of the proposed algorithm in different practical situations. The R package *ClusterStability* has been developed to provide applied researchers with new stability estimation tools presented in this paper. It is freely distributed through the Comprehensive R Archive Network (CRAN) and available at: https://cran.r-project.org/web/packages/ClusterStability.

1. Introduction

Clustering algorithms have been successively applied in many fields, including banking, bioinformatics, computer vision, marketing, and security, in order to extract the structure from a given dataset and to gain insight into its natural clusters [14,35]. There are two main clustering approaches that encompass hierarchical clustering and partitioning algorithms. In this article, we focus on the techniques for partitioning N objects into K clusters according to a specific similarity criterion. The total number of partitions of N objects into K non-empty and non-overlapping clusters is asymptotically equivalent to $K^N/K!$, as N tends to infinity [40]. Thus, heuristic algorithms such as *K*-means [29] and *K*-medoids [23] have been proposed to limit the number of possible solutions when searching for an optimal partition of objects. These heuristic algorithms are often preferred to more complex alternatives because of their simplicity and relatively good performances [41], as well as because of the availability of their parallel versions, which are scalable to large recognition problems [50]. Despite their popularity,

* Corresponding author.
E-mail addresses: lord.etienne@courrier.uqam.ca (E. Lord), willems.matthieu@courrier.uqam.ca (M. Willems), francois-joseph.lapointe@umontreal.ca (F.-J. Lapointe), makarenkov.vladimir@uqam.ca (V. Makarenkov).

http://dx.doi.org/10.1016/j.ins.2017.02.010
0020-0255/

K-means and *K*-medoids usually provide solutions that are only local optima [35,41]. Moreover, these algorithms highly depend on the number of random starts [41], and the choice of starting partitions is crucial for them [32,41,43]. In addition, the *K*-means algorithm is very sensitive to the presence of noisy features in the data [18,19]. Usually, several hundred starts of *K*-means with different input random partitions are required in order to select an appropriate clustering [41]. Finally, like many partitioning algorithms, *K*-means and *K*-medoids also suffer from the need to specify the desired number of clusters [29,35].

Recently, there has been a renewed interest in assessing the robustness of clustering solutions that are provided by partitioning algorithms [16,28,43]. Furthermore, alternative methods, such as model-based evaluation [10] or bootstrapping [18,19], have been proposed to assess the reproducibility of clusterings. Despite this increased attention, the intriguing and challenging problem of estimating the stability of individual objects in clustering has not been fully addressed in the literature [25,28].

In this paper, we define a novel measure for assessing the stability of individual objects (i.e., individual *ST*-index) in clustering solutions provided by partitioning algorithms, based on their repeated runs. We also propose a cluster stability index, which reflects the stability of clusters, and a global stability index (i.e., global *ST*-index), which characterizes the robustness of entire clusterings (i.e., resulting partitions or clustering solutions found by partitioning algorithms). These new indices can help practitioners decide which individual objects and clusters should be kept in the dataset and which of them should be removed from it in order to improve the stability of a given clustering. Moreover, the results of our simulation study indicate that the stability of a clustering estimated by our stability indices is directly related to its quality, and that the global *ST*-index can be effectively used to improve the ability of traditional clustering algorithms to determine the true number of clusters in datasets. Our R package *ClusterStability* provides researchers with the new stability estimation tools that we describe in this paper.

2. Background and related work

2.1. Cluster validity indices

A variety of cluster validity indices are available to determine the number of clusters in a given dataset [2,10,34]. They can be defined as measures of partitioning quality. Most of these indices take into consideration the compactness of the objects in the same cluster and their separation in the distinct clusters [2,34]. In our experiments with real and synthetic data, we will use the Calinski–Harabasz [11], Silhouette [39], Dunn [9] and Davies–Bouldin [13,24] measures, which have been among the most recommended cluster validity indices according to several simulation studies [2,10,34].

The Calinski–Harabasz index is a normalized ratio of the overall inter-cluster variance and the overall intra-cluster variance [11]. The Silhouette width [39] of an individual object i is defined using its average intra-cluster distance, $a(i)$, and its average nearest-cluster distance, $b(i)$. It is calculated as follows: $(b(i)-a(i))/\max(a(i), b(i))$. The global Silhouette width is defined as the average of the individual Silhouette widths of all the objects. The Dunn index is a ratio-type coefficient in which the cluster separation is expressed through the maximum cluster diameter, and the cluster cohesion is expressed through the nearest neighbor distance [9]. While there are various versions of the Dunn coefficient, the most reliable are the generalized Dunn's indices [9]. The Davies–Bouldin index is also based on a ratio of intra-cluster and inter-cluster distances [13]. For a pair of clusters (C_1, C_2), the pairwise cluster distance $db(C_1, C_2)$ is first calculated as the sum of the average distances between the objects and centroids in both clusters, which is then divided by the distance between the cluster centroids. The Davies–Bouldin index is defined as the average of the largest $db(C_k, C_l)$'s $(l \neq k)$, computed over all available clusters C_k. An improved variant of this coefficient proposed by Kim and Ramakrishna [24] provides very good cluster recovery performances according to a recent comparative study of cluster validity indices conducted by Arbelaitz et al. [2].

2.2. Stability of clustering solution

A number of theoretical and empirical studies have addressed the problem of solution stability in clustering [6,16,18,19,26,33,41-43]. Milligan and Cheng [33] were first to investigate how the addition and removal of objects influence the quality of the resulting clusterings. Ben-Hur et al. [6] proposed to use, as a measure of cluster stability, the distribution of pairwise similarities between partitions obtained from clustering sub-samples of a given dataset. In order to determine the true number of clusters in a dataset, the authors suggested examining the clusters in which a transition from a stable to an unstable clustering state can occur. Lange et al. [26] introduced a measure that quantifies the reproducibility of clustering solutions and defined a function that minimizes the risk of misclassification. Ben-David et al. [4] provided a formal definition of cluster stability and concluded that, for large datasets, cluster stability is closely related to the behavior of the objective function of a given clustering algorithm.

Hennig [18,19] discussed several strategies for assessing the support of individual clusters in a clustering solution. One of these strategies relies on the use of the Jaccard coefficient and resampling techniques such as bootstrapping, jittering, and subsetting [18]. Hennig [19] showed how to determine the dissolution point and the isolation robustness of a cluster by adding to it new objects and outliers. Fang and Wang [16] proposed a different variant of data bootstrapping that allows for selecting the number of clusters in a dataset by examining randomness in the samples. While several papers discuss the stability of clustering methods with respect to changes in a given dataset, the work of de Mulder [37] focuses on cluster

stability with respect to changes in the starting conditions of partitioning algorithms. de Mulder introduced the notions of instability and structure-preserving data elements and proved that the removal of a structure-preserving unstable data element from the dataset is a way for improving the robustness of a clustering solution.

Bertrand and Mufti [7] showed how the stability of a given cluster can be characterized using Loevinger's measures of rule quality. The authors defined the stability of a whole partition as a weighted mean of the stability scores of all clusters in the partition. Wang [49] proposed an efficient criterion for selecting the number of clusters in datasets that minimizes the algorithm's instability. He developed a new instability estimation scheme based on cross-validation and tested it in the framework of *K*-means clustering.

Several works addressing the problem of stability of *K*-means clustering have examined a set of solutions obtained for different starting partitions [22,43]. Steinley [43] presented a way of conducting stability analysis based on an object-by-object co-occurrence matrix, which was determined after several repeated runs of *K*-means. This matrix can be clustered and subsequently reordered to create a visual interpretation of the multidimensional cluster structure. Afterward, stability-based measures can be defined to determine the overall structure of a dataset and identify the number of its clusters [43]. Kuncheva and Vetrov [25] also assessed the stability of cluster ensembles with respect to random *K*-means initializations. They proposed a combined stability index, which was defined as the sum of the pairwise individual and ensemble stabilities. In order to define the pairwise and nonpairwise stability, Kuncheva and Vetrov used the average Adjusted Rand Index (ARI) between pairs of clusters in the ensemble and the entropy of the consensus matrix of the ensemble.

Despite the existing research efforts, the problem of stability of individual elements in clustering solutions has not been sufficiently addressed. In their recent work, Lord et al. [28] have proposed a simple way of defining the stability of an object in the context of clustering bioinformatics workflows. However, most of the work focuses on the analysis of the most stable and unstable groups of objects in datasets (where the unstable groups can be eventually associated with some kind of misclassified or noisy elements) [7,16,18,19,43]. In the next section, we will introduce a novel stability measure designed to characterize the robustness of individual objects, based on the random starts (i.e., random initializations) of a partitioning algorithm. There are two main properties that distinguish our measure from existing stability indices: (1) it takes into account the quality of each partition (i.e., expressed through the value of the selected cluster validity index) obtained after each random start of the selected partitioning algorithm, and (2) it includes a correction for chance co-occurrence (i.e., probability that two objects belong to the same cluster only by chance).

3. New indices for estimating the stability of objects, clusters and whole clustering solutions

In this section, we introduce a new stability index for individual objects in a clustering. It is defined by using partitions obtained after a series of random starts of a given partitioning algorithm. We will show that the elimination of the most unstable individual objects from the dataset increases the stability of clustering solution, therefore improving the clustering quality. The proposed individual stability index will be extended to calculate the robustness estimates for clusters (i.e., cluster stability score) and entire clustering solutions (i.e., global stability score). In many instances, clusters with low stability scores can be considered as clusters of noise which should be removed from the dataset. Moreover, we will present a new algorithm that can be used to determine the number of clusters in a dataset based on the value of the global stability score.

Let N, $(N > 1)$, be the number of objects in a given dataset and K, $(1 < K \leq N)$, be the number of classes (i.e., clusters, groups) in the solution provided by the selected partitioning algorithm (here *K*-means or *K*-medoids). Let R be the number of random starts of this algorithm. Assume that at random start r, $(1 \leq r \leq R)$, the algorithm calculates the partition P_r of non-overlapping classes. A partition score, S_r, accounting for the partitioning quality, can be associated with each partition P_r $(1 \leq r \leq R)$. For instance, the popular Calinski–Harabasz and Silhouette cluster validity indices can be used as partition scores.

First, the pairwise support, *PS*, of two distinct objects i and j is defined as follows:

$$PS_{ij} = \frac{\sum_{r=1}^{R} S_{r,ij}}{\sum_{r=1}^{R} S_r}, \tag{1}$$

where S_r is the value of the selected cluster validity index associated with partition P_r; $S_{r,\,ij} = S_r$ if objects i and j belong to the same class in partition P_r, and $S_{r,\,ij} = 0$, otherwise. The values of PS_{ij} are located in the interval [0, 1]. For instance, $PS_{ij} = 1$ if objects i and j belong to the same class in all considered partitions P_r $(1 \leq r \leq R)$, and $PS_{ij} = 0$ if objects i and j belong to different classes in all of these partitions. The computation of PS_{ij} is straightforward in the case of the Calinski–Harabasz index. Since the Silhouette width varies from -1 to 1, a feature rescaling should be carried out to bring the values of S_r into a positive range (e.g., [0,1] interval). Clearly, the pairwise support PS_{ij} depends not only on the number of partitions in which objects i and j are located in the same class but also on the quality of these partitions.

The singleton support of object i, reflecting the probability that i is a single element in its class, is defined as follows:

$$PS_i = \frac{\sum_{r=1}^{R} S_{r,i}}{\sum_{r=1}^{R} S_r}, \tag{2}$$

where $S_{r,\,i} = S_r$ if object i belongs to a singleton class in partition P_r, and $S_{r,\,i} = 0$, otherwise. Thus, an object that is always classified as an element belonging to a singleton class will have its singleton support score equal to 1, whereas all pairwise support scores involving this object will be equal to 0. In the case of the Calinski–Harabasz (CH) index, the computation of all coefficients S_r, $(1 \le r \le R)$, takes $O(RNM)$ time, since the time complexity of the CH computation is $O(NM)$, where M is the number of variables. Thus, the computation of PS_{ij} for a given pair of objects (i,j) (or of PS_i for a given object i) takes $O(RNM)$ time, while the computation of all pairwise and singleton supports $(1 \le i, j \le N)$ takes $O(RN^2 + RNM)$ time.

By using a probabilistic approach, we can now define individual stability indices, or *ST*-indices, for objects. For all pairs of objects (i,j), we compare their pairwise score, PS_{ij} (Eq. 1), with the probability, $p(N, K)$, that two randomly chosen objects x and y are located in the same class in a randomly selected K-class partition P of N objects. The closer PS_{ij} to $p(N, K)$, the greater the probability that objects i and j have been assigned to the same class only by chance. Likewise, we need to compare the singleton support scores, PS_i (Eq. 2), with the probability, $p_s(N, K)$, that a randomly chosen object x is located in a singleton class in a randomly selected K-class partition P of N objects.

The quantities $p(N, K)$ and $p_s(N, K)$ can be computed as follows. Let Ω be a set of objects of cardinality N. Consider two distinct randomly chosen objects x and y in Ω, and a random partition, P, of objects of Ω into K non-empty classes. For any integer $K \ge 0$ and any set of objects E, we denote by $\mathrm{Par}(E, K)$ the set of all possible partitions of E into K non-empty classes. Note that $\mathrm{Par}(E, K) = \emptyset$ if $\mathrm{card}(E) < K$, and $\mathrm{Par}(E, 0) = \emptyset$ for any non-empty set E. If we denote by $\mathrm{Par}(\Omega, K, x, y) \subset \mathrm{Par}(\Omega, K)$ the set of all partitions P of Ω into K non-empty classes, such that x and y are located in the same class of P, we can define a map ∂: $\mathrm{Par}(\Omega, K, x, y) \to \mathrm{Par}(\Omega/\{x\}, K)$, such that $\partial(P)$ is the restriction of partition P of objects of Ω to $\Omega/\{x\}$. This map is well defined. Indeed, all classes of $\partial(P)$ are non-empty, since every class containing x also contains y. The defined map is clearly injective. It is also surjective because for all partitions P' of $\Omega/\{x\}$, partition $\partial^{-1}(P')$ is the partition obtained by adding x to the class containing y. Let $S(N, K)$ be the cardinality of the set $\mathrm{Par}(E, K)$ for any set E containing N objects. Thus, we obtain:

$$p(N,K) = \frac{\mathrm{Card}(\mathrm{Par}(\Omega, K, x, y))}{\mathrm{Card}(\mathrm{Par}(\Omega, K))} = \frac{\mathrm{Card}(\mathrm{Par}(\Omega/\{x\}, K))}{\mathrm{Card}(\mathrm{Par}(\Omega, K))} = \frac{S(N-1, K)}{S(N, K)}. \tag{3}$$

The quantities $S(N, K)$ are the Stirling numbers of the second kind. They can be computed by means of the following recurrence formula: $S(N, K) = K \times S(N-1, K) + S(N-1, K-1)$, with the following initial conditions: $S(0, 0) = 1$, $S(0, K) = S(N, 0) = 0$ and $S(N, 1) = 1$, for $K \ge 1$ and $N \ge 1$ [36]. Using these formulas, the Stirling numbers of the second kind can be calculated in $O(N^2)$. They can also be calculated using the following explicit formula [27]:

$$S(N,K) = \frac{1}{K!} \sum\nolimits_{0 \le l \le K} (-1)^{K-l} \binom{K}{l} l^N. \tag{4}$$

In order to avoid the stack overflow issue in our simulations (see Sections 4 and 5), we used an approximation of the Stirling numbers of the second kind. For instance, if we fix K and if $K < \frac{N}{\ln(N)}$, we have the following asymptotic approximation, as $N \to +\infty$ [40]:

$$S(N,K) = \frac{K^N}{K!} exp\left[\left(\frac{N}{K} - K\right) e^{-\frac{N}{K}}\right](1 + o(1)). \tag{5}$$

In particular, we have: $lim_{N \to +\infty} \frac{S(N,K)}{K^N/K!} = 1$.

Then, we obtain:

$$\lim_{N \to +\infty} p(N,K) = \lim_{N \to +\infty} \frac{K^{N-1}/K!}{K^N/K!} = \frac{1}{K}. \tag{6}$$

In the same way, it is easy to prove that we have the following formula for $p_s(N, K)$:

$$p_s(N,K) = \frac{\mathrm{Card}(\mathrm{Par}(\Omega/\{x\}, K-1))}{\mathrm{Card}(\mathrm{Par}(\Omega, K))} = \frac{S(N-1, K-1)}{S(N, K)}. \tag{7}$$

Thus, the following limit can be obtained:

$$\lim_{N \to +\infty} \frac{p_s(N,K)}{(K-1)^{N-1}/K^{N-1}} = 1. \tag{8}$$

Now we can define the individual stability index, $ST(i)$, characterizing object i as follows:

$$\begin{aligned} ST(i) = {} & \tfrac{1}{N} \sum_{j=1 (j \ne i)}^{N} max\left(\tfrac{1}{1-p(N,K)} \times \left(PS_{ij} - p(N,K)\right); \tfrac{1}{p(N,K)} \times \left(p(N,K) - PS_{ij}\right)\right) + \\ & \tfrac{1}{N} max\left(\tfrac{1}{1-p_s(N,K)} \times (PS_i - p_s(N,K)); \tfrac{1}{p_s(N,K)} \times (p_s(N,K) - PS_i)\right). \end{aligned} \tag{9}$$

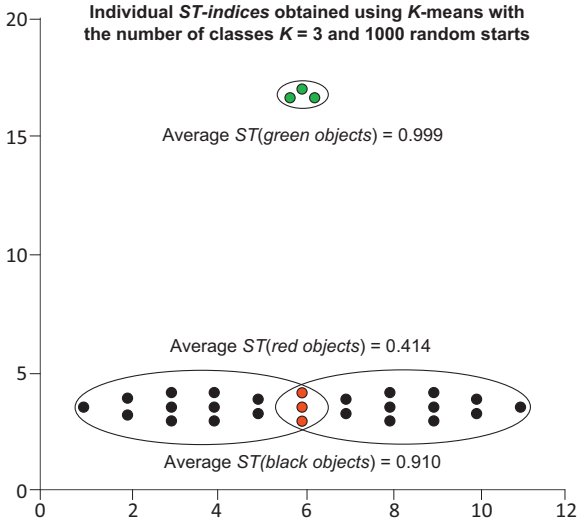

Fig. 1. A two-dimensional dataset, featuring 3 very stable green objects, 22 black objects, and 3 very unstable red objects, used to illustrate the impact of the proposed individual and global stability indices on the stability of clustering solutions. The exact coordinates of these objects are reported in Supplementary Table 1. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Using Formulas 6 and 8, we obtain the following approximation of $ST(i)$:

$$ST_{approx}(i) = \frac{1}{N}\sum_{j=1(j\neq i)}^{N} max\left(\frac{K}{K-1}\times\left(PS_{ij}-\frac{1}{K}\right);K\times\left(\frac{1}{K}-PS_{ij}\right)\right)+ \\ \frac{1}{N}max\left(\frac{K^{N-1}}{K^{N-1}-(K-1)^{N-1}}\times\left(PS_i-\frac{(K-1)^{N-1}}{K^{N-1}}\right);\frac{K^{N-1}}{(K-1)^{N-1}}\times\left(\frac{(K-1)^{N-1}}{K^{N-1}}-PS_i\right)\right). \quad (10)$$

This approximation should be used for large values of N, i.e., those for which the large values of $S(N, K)$ can cause overflow errors in the computer program. The following experimental limits for the exact computation of the individual stability index ST have been found by our R program (R version 3.2.1 was used) executed on an IBM PC computer equipped with an Intel i7 processor and 8Go of RAM: $N = 1020$ was the maximum possible value of N for this version of R and our computer configuration (with $K = 2$), and $N = 219$ was the greatest value of N for which we were able to calculate all values of $S(N, K)$.

In the case of the Calinski–Harabasz index, the computation of an individual stability index $ST(i)$, or $ST_{approx}(i)$, for a given object i, takes $O(RN+RNM+N^2)$ time, since the computation of $p(N, K)$ and $p_s(N, K)$ can be completed in $O(N^2)$. Therefore, the computation of all N indices $ST(i)$, or $ST_{approx}(i)$, $1 \le i \le N$, requires $O(RN^2 + RNM)$ time.

The stability score of cluster C, denoted by ST_C, can be defined as follows:

$$ST_C = \frac{1}{N_c}\sum_{i=1}^{N_c} ST(i), \quad (11)$$

where N_c is the number of objects in C.

Finally, the global stability score, ST_{global}, which characterizes the stability of the whole clustering solution, can be computed as follows:

$$ST_{global} = \frac{1}{N}\sum_{i=1}^{N} ST(i). \quad (12)$$

The first maximum appearing in Eqs. (9) and (10) accounts for the proportion of random starts where objects i and j belong (the first term of the maximum) or do not belong (the second term of the maximum) to the same class, while taking into consideration a correction for chance co-occurrence. Thus, two objects always, or never, belonging to the same class contribute the maximum value of 1 to the sums appearing in Eqs. (9) and (10). This corresponds to the maximum possible pairwise stability. The second maximum appearing in these equations accounts for the stability of the singleton elements. Both equations are normalized by the number of their terms, and the values of the three introduced stability indices (i.e., for individual objects - Eqs. (9) and (10), clusters - Eq. (11), and the whole clustering solution - Eq. (12)) vary from 0 to 1. The closer the global ST-index (Eq. 12) to 1, the higher the stability of the associated partitioning solution. Importantly, Eq. (12) can be also used to select the optimal number of classes, i.e., in the sense of robustness, in a dataset. This optimal number will correspond to the number of classes, K, providing the maximum value of ST_{global}.

The following simple example illustrates how the introduced individual and global stability scores reflect the contribution of individual objects to clustering (see Fig. 1). This figure presents a set of 28 objects in a two-dimensional space.

The depicted objects were clustered using traditional K-means with the following options: the number of classes $K = 3$, the cluster validity index = CH, and the number of random starts $R = 1000$. Fig. 1 shows 3 very stable green objects with the average individual stability index ST(green objects) = 0.999, 22 black objects with the average individual stability index ST(black objects) = 0.910, and 3 very unstable red objects with the average individual stability index ST(red objects) = 0.414. The 3 red objects constantly change their class location, and thus their class neighbours selected among the black objects, in 1000 clustering solutions resulting from 1000 random starts of K-means. The global stability score, ST_{global}, is 0.867 for this clustering solution. The individual stability indices help us identify both the most stable elements, which should be kept in the dataset, and the most unstable ones, which should be removed from it, in order to improve the stability and, as we will see in the next section, the quality of a given clustering solution. For instance, by removing the 3 red objects from the dataset presented in Fig. 1, we were able to considerably improve the stability of the original clustering. This is expressed through the increased value of the global stability score ($ST_{global} = 0.993$) for the modified dataset containing the green and the black objects only. In the next sections, we will show how the global stability score can be used to improve the recovery of the true number of classes in a dataset provided by traditional partitioning algorithms. The optimal number of classes for partitioning algorithms based on the introduced stability indices will correspond to the maximum value of ST_{global}.

The algorithm for both calculating the individual stability scores of objects and identifying the number of classes in a dataset is presented below.

Algorithm 1. Computation of ST-indices and identification of the number of classes, K_{opt}.

INPUT:
- the dataset Ω of N objects
- the partitioning method (e.g., K-means or K-medoids)
- the cluster validity index, CVI (e.g., Calinski–Harabasz or Silhouette)
- the number of random starts (R)
- the minimum and maximum numbers of classes (K_{min}, K_{max})

OUTPUT:
- the global ST-indices: $ST_{global} = \{ST_{global}(K_{min}), \ldots, ST_{global}(K_{max})\}$
- the optimal number of classes K_{opt}
- the optimal individual ST-indices: $ST = \{ST(1), \ldots, ST(N)\}$ corresponding to K_{opt}

PROCEDURE:

for $K = K_{min}$ **to** K_{max} **do**
  for $r = 1$ **to** R **do**
    Generate a random starting partition, RP_r, of objects of Ω into K non-empty classes
    Execute K-means or K-medoids using RP_r as input to get partition P_r
  for $i = 1$ **to** N **do**
    Compute the singleton support PS_i using the selected CVI and Eq. (2)
    for $j = i + 1$ **to** N **do**
      Compute the pairwise support PS_{ij} using the selected CVI and Eq. (1)
  for $i = 1$ **to** N **do**
  Compute the individual stability index $ST(i,K)$ or its approximate variant $ST_{approx}(i,K)$ using the support scores PS_i and PS_{ij}, and Eq. (9) and (10)
Compute the global stability index, $ST_{global}(K)$, for the case of K classes using Eq. (12)
end of the first loop for
Find the maximum value of $ST_{global}(K)$ over all tested values of K: $K_{min} \le K \le K_{max}$
The optimal number of classes, K_{opt}, will correspond to the maximum of $ST_{global}(K)$
for $i = 1$ **to** $i = N$ **do**
  $ST(i) = ST(i,K_{opt})$
end of the algorithm

Note that Algorithm 1 can be executed several times, with the most unstable objects removed from Ω after each new execution, according to the individual stability index $ST(i,K)$. Such repeated executions of the proposed algorithm allowed us to improve the clustering performance of our stability-based technique in several cases (see, for example, the analysis of the Iris dataset in Section 4.3).

Here the time complexity of our algorithm is determined in the case of the K-means partitioning method and the Calinski–Harabasz (CH) index. For a fixed number of classes K, the time complexity of K-means is $O(KNMR)$, where M is the number of variables, N is the number of objects and R is the number of random starts. The time complexity of K-means executed over the interval $[K_{min}, K_{max}]$ of values of K is $O((K_{max} - K_{min})(K_{max} + K_{min})RNM)$. This is equivalent to the total number of all K-means-related operations in our algorithm. Moreover, the complexity of the calculation of all stability indices $ST(i,K)$ is $O((K_{max} - K_{min})(RN^2 + RNM))$. This leads to the overall time complexity of $O((K_{max} - K_{min})RN((K_{max} + K_{min})M + N))$. After replacing $K_{max} - K_{min}$ and $K_{max} + K_{min}$ by K_{max}, we obtain a simpler running time estimate for our algorithm, which is $O(K_{max}RN(K_{max}M + N))$.

Table 1
Datasets' main characteristics and the numbers of classes obtained for the 21 selected benchmark datasets from the UCI repository [3] by classical *K*-means [29], *K*-means based on our stability index (*ST*-index) and Hennig's bootstrapping and jittering techniques applied to *K*-means [18]. These results were obtained using the Calinski–Harabasz and Silhouette cluster validity indices with the number of classes varying from 2 to 20. The last row reports the mean absolute differences between the obtained number of classes and the true numbers of classes as well as the corresponding standard deviation.

Dataset	Characteristics			Classical *K*-means		*K*-means *ST* [a]		Hennig's methods	
	Number of objects	Number of features	Number of classes	Calinski–Harabasz	Silhouette	Calinski–Harabasz	Silhouette	Bootstrap[b]	Jittering[c]
Breast tissue	106	9	6	6	2	4	4	4	3
B. Wisconsin	569	30	2	6	2	2	2	2	2
Ecoli	336	7	8	3	3	2	2	2	2
Glass	214	9	7	2	3	2	2	2	2
Haberman	306	3	2	4	2	4	4	4	4
Ionosphere	351	34	2	2	2	2	2	2	2
Iris	150	4	3	3	2	2	2	2	2
Move. libras	360	90	15	2	9	20	20	2	2
Musk	476	166	2	2	3	3	3	3	3
Parkinson	195	22	2	4	4	3	3	4	3
Segmentation	2310	19	7	5	11	4	3	3	3
Sonar all	208	60	2	2	2	2	2	3	3
Spectf. Heart	267	44	2	2	2	2	2	2	2
Transfusion	748	4	2	13	2	2	2	2	2
Vehicule	946	18	4	2	2	2	2	2	2
Vert. column	310	6	3	2	2	2	2	2	2
Vowel context	990	10	11	2	2	4	4	2	4
Wine	178	13	3	6	2	2	3	2	2
Wine qual. red	1599	11	6	3	2	5	5	2	4
Yeast	1484	8	10	2	2	2	2	2	2
Zoo	101	17	7	2	5	7	7	4	2
Mean absolute difference ± SD				3.57 ± 3.83	2.57 ± 2.71	**2.19 ± 2.50**	**2.19 ± 2.56**	3.10 ± 3.43	3.00 ± 3.33

[a] 1000 random starts of *K*-means.
[b] 1000 bootstrap replicates.
[c] 1000 random starts of *K*-means and jittering level of 0.75.

4. Experiments with benchmark data

In this section and the one to follow, we will use the introduced individual and global *ST*-indices to analyze real and simulated datasets that contain different numbers of objects, variables and clusters, and that encompass different types and levels of noise. First, we will evaluate the behavior of the individual and global *ST*-indices by considering 21 real datasets (see Table 1 or Supplementary Table 2 for their characteristics) from the popular UCI Machine Learning Repository [3]. Specifically, we selected the same 20 benchmark datasets that were examined by Arbelaitz et al. in their recent study, which compared different cluster validity indices in the framework of *K*-means clustering (see Table 1 here or Table 2 in [2]). We added to these data the well-known Zoo dataset [17]. In our simulations, we used the Euclidian distance and the classical MacQueen's implementation of *K*-means available in R. In the case of *K*-medoids, we carried out the R function *WeightedCluster* [44]. Both partitioning algorithms were executed with the limit of 100 iterations in the main algorithm's loop. The number of random starts for each dataset and both partitioning algorithms was set to 1000. To detail the advantages of the new method, we will more thoroughly describe the experimental results obtained for the Zoo, Blood transfusion and Iris datasets (Sections 4.1–4.3) before presenting the overall simulation results for real-world data (Section 4.4).

4.1. Analysis of Zoo dataset

The Zoo dataset ($N = 101$, $K = 7$) [17] includes measurements characterizing 101 species (i.e., objects) divided into 7 empirical classes: mammals (41), birds (20), reptiles (5), fishes (13), amphibians (4), insects (8), and invertebrates (10). Each species is described by a set of 16 binary variables (e.g., presence or absence of feathers) and one integer variable accounting for the number of legs. Zoo data are often used to demonstrate the application of partitioning and machine learning methods [31]. Similar to many empirical datasets (e.g., the Iris dataset presented in Section 4.3), the Zoo dataset cannot be clearly divided into 7 non-overlapping classes by traditional clustering algorithms.

The results provided by the conventional *K*-means and *K*-medoids algorithms used with the Calinski–Harabasz (CH) and Silhouette (SI) cluster validity indices exhibited similar trends, with slightly higher stability scores obtained for *K*-medoids. Thus, only the results provided by *K*-means are discussed here. As shown in Fig. 2a (the solution for $K = 7$ is presented here), the individual species support in clustering varies greatly, but the boundaries between different classes can be clearly identified by means of the individual *ST*-indices. We can observe that fruit bat (Fig. 2a: Object 1, $ST = 0.495$) and vampire bat (Fig. 2a: Object 5, $ST = 0.496$) have the lowest stability scores among all species. This implies that both species of bats were often assigned to different classes through the partitioning process. The positions of girl and gorilla (Fig. 2a: Objects

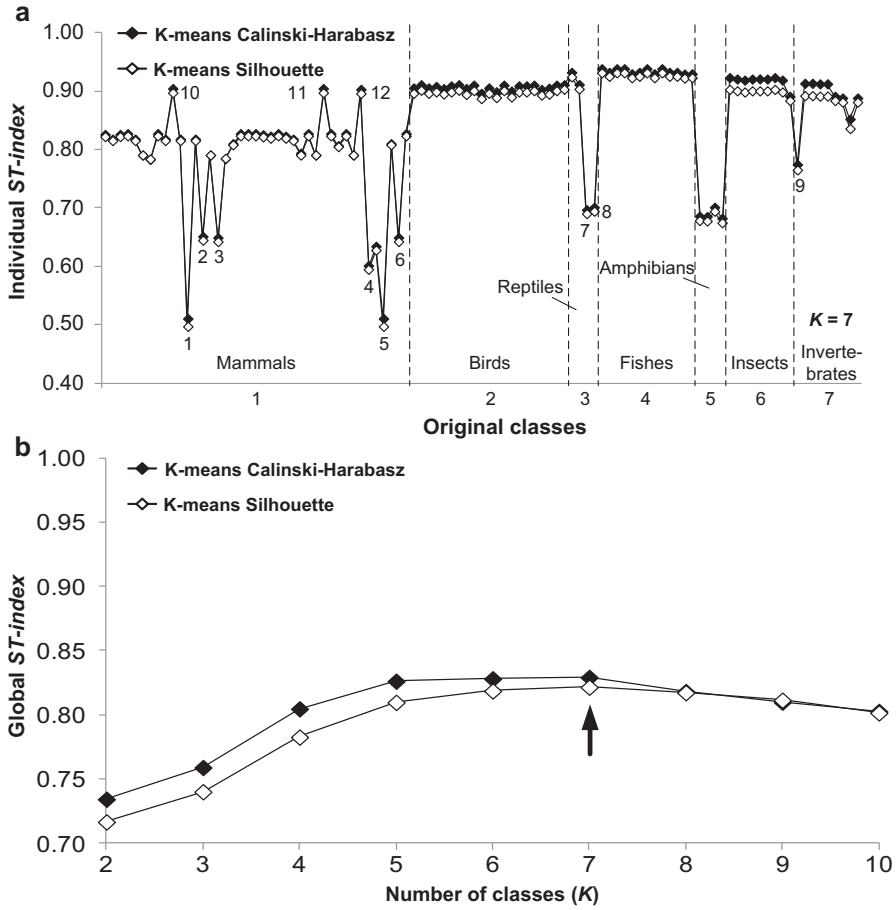

Fig. 2. Analysis of Zoo dataset: (a) Variation of the individual *ST*-scores for the species of Zoo dataset obtained using *K*-means partitioning and the CH and SI cluster validity indices. The annotated species (i.e., objects) are as follows: 1) fruit bat, 2) girl, 3) gorilla, 4) sea lion, 5) vampire bat, 6) wallaby, 7) tortoise, 8) tuatara, 9) crab, 10) porpoise, 11) dolphin, and 12) seal; (b) Variation of the global *ST*-scores with respect to the number of classes, *K*, shown for the CH and SI indices. The black arrow indicates the true number of classes for this dataset ($K = 7$).

2 and 3) were also very unstable (their individual *ST*-indices were equal to 0.643 and 0.641, respectively). On the other hand, and in accordance with previous results [31], porpoise and dolphin showed high stability within mammals (Fig. 2a: Objects 10 and 11). The groups of amphibians and reptiles were the most poorly supported, including the species of tortoise (Fig. 2a: Object 7) and tuatara (Fig. 2a: Object 8), which had low individual *ST*-indices. As noted by McKenzie and Forsyth [31], the neural network and *KNN* algorithms misclassify the reptiles as amphibians or fishes. Moreover, it is very likely that the poor stability observed for the reptiles also contributed to the low stability for the amphibians, since frequent inter-class pairings of individual objects decrease the stability of both involved classes. It is worth noting that the low values of some stability indices can be explained by the presence of binary features, which are often not as discriminative as continuous variables.

Finally, the global stability of the *K*-means partitioning solutions obtained on the range of 2 to 10 classes (Fig. 2b) using the CH and SI cluster validity indices was evaluated for the Zoo dataset. The maxima of the curves in Fig. 2b can be used as indicators of the correct number of classes. The maxima of both, CH and SI, curves shown in Fig. 2b allow us to find the true number of classes ($K = 7$) for Zoo data even though the solutions with 5, 6 or 8 classes also seem to be appropriate here. Note that the traditional *K*-means algorithm found that 2 and 5 would be the optimal numbers of classes for Zoo data, according to the CH and SI indices, respectively.

4.2. Analysis of Blood transfusion dataset

The Blood transfusion service dataset ($N = 748$, $K = 2$) contains blood donor data collected at Blood transfusion Hsin-Chu City Service Center in Taiwan [3,12,15]. Its four attributes describe respectively the number of months since last donation, the total number of blood donations, the total volume of donated blood, and the number of months since the first donation. The

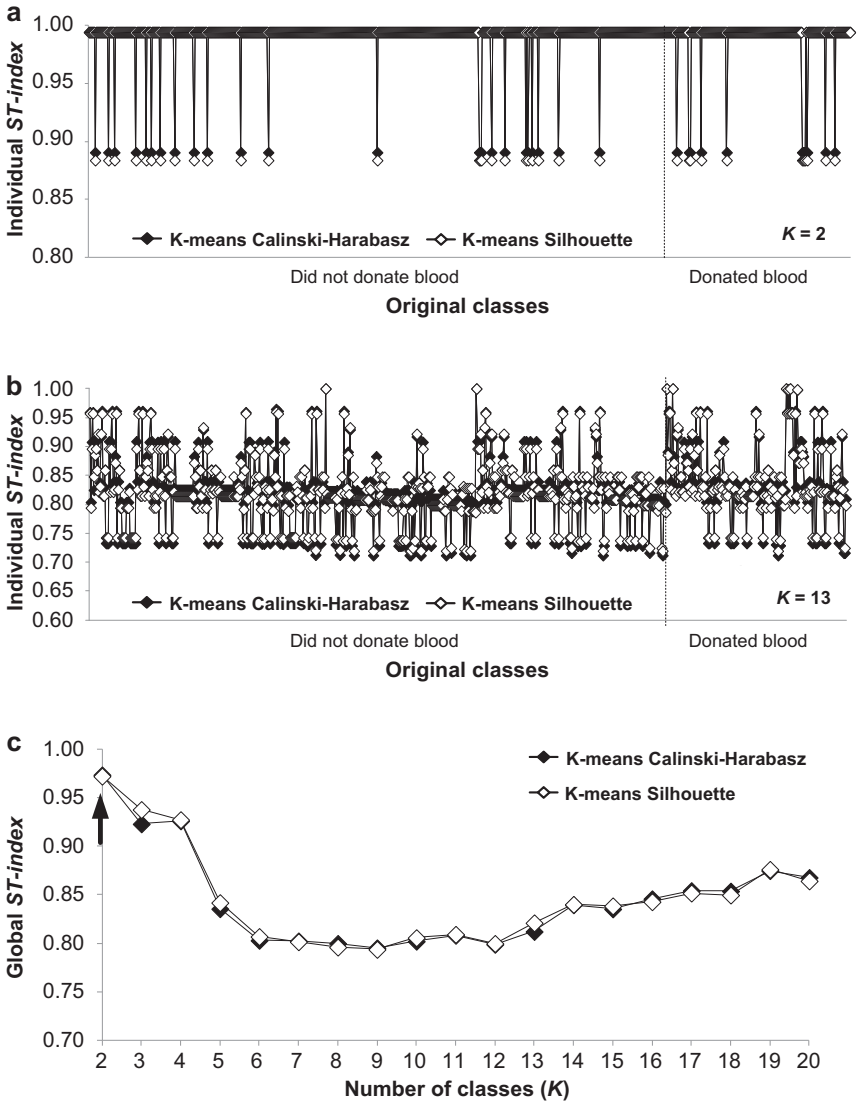

Fig. 3. Analysis of Blood transfusion dataset: (a) Variation of the individual *ST*-scores of objects of Blood transfusion dataset obtained using *K*-means partitioning and the CH and SI cluster validity indices for $K=2$; (b) Variation of the individual *ST*-scores for $K=13$; (c) Variation of the global *ST*-scores with respect to the number of classes, *K*, shown for the CH and SI indices; the black arrow indicates the true number of classes for this dataset ($K=2$).

dataset is divided into two classes. The first of them includes 570 individuals who did not donate blood in March 2007, while the second includes 178 individuals who were the blood donors during this period. The Blood transfusion dataset has been examined in a number of classification studies and has been complicated for clustering with traditional methods [12,15]. For example, Deus and Liao [15] found that 172 blood donors were misclassified by the traditional *K*-means algorithm. Moreover, Chaimontree et al. [12], who examined this dataset using *K*-means in conjunction with different cluster validity indices, observed that the number of classes found by *K*-means could vary from 2 to 15, depending on the selected cluster validity index. According to the same authors, the *KNN* clustering does not offer better results for this dataset, since the partitions with up to 18 classes were identified by different versions of *KNN* for Blood transfusion data.

The results of the traditional *K*-means analysis conducted on Blood transfusion data showed that the number of classes found using the CH index was 13, the SI index 2, the Dunn index 5, and the Davies–Bouldin (DB) index also 5 (see Table 1 and Supplementary Table 2). The application of our global *ST*-index allowed us to improve the results of traditional *K*-means: the highest global *ST* value was found for $K=2$ with all the four cluster validity indices tested here (see Supplementary Table 4 and Fig. 3). The CH and SI-based stability curves drawn for the cases $K=2$ (global $ST=0.973$ for CH and 0.972 for SI; see Fig. 3a) and $K=13$ (global $ST=0.812$ for CH and 0.821 for SI; see Fig. 3b) indicate that the latter clusterings have much higher variability of individual stability indices. It is worth noting that in the case of *K*-medoids, the use of the *ST*-index allowed us to reduce the difference between the estimated and true numbers of classes for all four

Table 2
Datasets' main characteristics and the numbers of classes obtained for the 21 selected benchmark datasets from the UCI repository [3] by classical *K*-medoids [23], *K*-medoids based on our stability index (*ST*-index) and Hennig's bootstrapping and jittering techniques applied to *K*-medoids [18]. These results were obtained using the Calinski–Harabasz and Silhouette cluster validity indices with the number of classes varying from 2 to 20. The last row reports the mean absolute differences between the obtained number of classes and the true numbers of classes as well as the corresponding standard deviation.

Dataset	Characteristics			Classical *K*-medoids		*K*-medoids *ST* [a]		Hennig's methods	
	Number of objects	Number of features	Number of classes	Calinski–Harabasz	Silhouette	Calinski–Harabasz	Silhouette	Bootstrap[b]	Jittering[c]
Breast tissue	106	9	6	20	3	5	5	2	2
B. Wisconsin	569	30	2	2	2	2	2	2	2
Ecoli	336	7	8	3	3	3	3	3	4
Glass	214	9	7	3	3	4	4	2	2
Haberman	306	3	2	4	2	3	3	2	4
Ionosphere	351	34	2	2	3	3	3	2	2
Iris	150	4	3	3	2	2	2	2	2
Move. libras	360	90	15	5	10	6	6	6	6
Musk	476	166	2	2	3	3	3	2	2
Parkinson	195	22	2	13	8	4	4	2	3
Segmentation	2310	19	7	5	2	3	3	2	2
Sonar all	208	60	2	2	2	3	3	2	2
Spectf. Heart	267	44	2	3	2	5	5	2	3
Transfusion	748	4	2	7	10	3	3	3	9
Vehicule	946	18	4	2	2	4	4	2	3
Vert. column	310	6	3	2	2	4	4	2	2
Vowel context	990	10	11	2	2	4	4	4	4
Wine	178	13	3	20	2	4	4	2	3
Wine qual. red	1599	11	6	5	2	4	4	3	3
Yeast	1484	8	10	2	3	3	3	2	3
Zoo	101	17	7	4	4	8	8	4	3
Mean absolute difference ± SD				4.52 ± 5.05	3.14 ± 2.80	$\mathbf{2.48 \pm 2.52}$	$\mathbf{2.48 \pm 2.52}$	2.62 ± 2.89	2.95 ± 2.84

[a] 1000 random starts of *K*-medoids.
[b] 1000 bootstrap replicates.
[c] 1000 random starts of *K*-medoids and jittering level of 0.75.

cluster validity indices (from 7 to 3 for CH, from 10 to 3 for SI, from 7 to 3 for Dunn, and from 5 to 3 for DB; see Table 2 and Supplementary Table 3) even though the stability-based *K*-medoids procedure did not find the true number of classes in this instance.

4.3. Analysis of Iris dataset

The stability of individual objects and of entire clusterings was also evaluated for the well-known Iris dataset ($N = 150$, $K = 3$). This dataset consists of a series of 4 measurements (i.e., sepal length, sepal width, petal length and petal width) taken over 150 Iris plants [1] belonging to three species of Iris: *I. setosa* (50), *I. virginica* (50) and *I. versicolor* (50). It has the property that the classes *I. virginica* and *I. versicolor* cannot be clearly separated using traditional partitioning algorithms such as *K*-means and *K*-medoids ([45]; see also our PCA plot in Fig. 4a). The stability of the individual elements of Iris was assessed by carrying out the *K*-means and *K*-medoids algorithms with the CH and SI cluster validity indices (Fig. 4b; the solution for $K = 3$ is shown here). For both partitioning algorithms, the elements of the class *I. setosa* showed a strong individual stability: the mean (over CH and SI) *ST*-index = 0.938 for *K*-means and the mean (over CH and SI) *ST*-index = 1.0 for *K*-medoids. However, these scores were much lower for the individual elements of the two remaining classes, *I. virginica* and *I. versicolor*. The strongest drop in the individual *ST*-scores for both cluster validity indices was observed with *K*-medoids. For example, we found that for the 24 elements located in the grey area in Figs. 4a and b, the mean individual *ST*-score obtained using SI was equal to 0.621, whereas it was equal to 0.935 for the rest of the elements of *I. virginica* and *I. versicolor*.

Afterwards, we evaluated the behavior of the global *ST*-index (Fig. 4c). This measure represents the global stability of the obtained clustering solution for a given number of classes. The highest global stability support for both *K*-means and *K*-medoids and both CH and SI was attained with two classes ($ST_{global} = 1.0$).

The Iris dataset is an example of data in which stability measures do not lead directly to the recovery of the true number of classes because of the large overlap of the elements of *I. versicolor* and *I. virginica* [45]. Wang [49], who examined the global cluster stability of the Iris dataset using his cluster cross-validation procedure, also found that the highest cluster stability of *K*-means partitioning was obtained with two classes. Furthermore, de Mulder [Fig. 1a in 37] noticed in his analysis, which was conducted with different cluster instability functions and the *K*-means and fuzzy *C*-means [8] algorithms, that the lowest instability numbers for Iris were consistently obtained with two classes. On the other hand, Topchy et al. [48] found that at least four classes were necessary for minimizing the number of misclassified objects in the Iris dataset (see Fig. 4b in [48]). In comparison, our cluster stability algorithm allows for the identification of the most unstable ele-

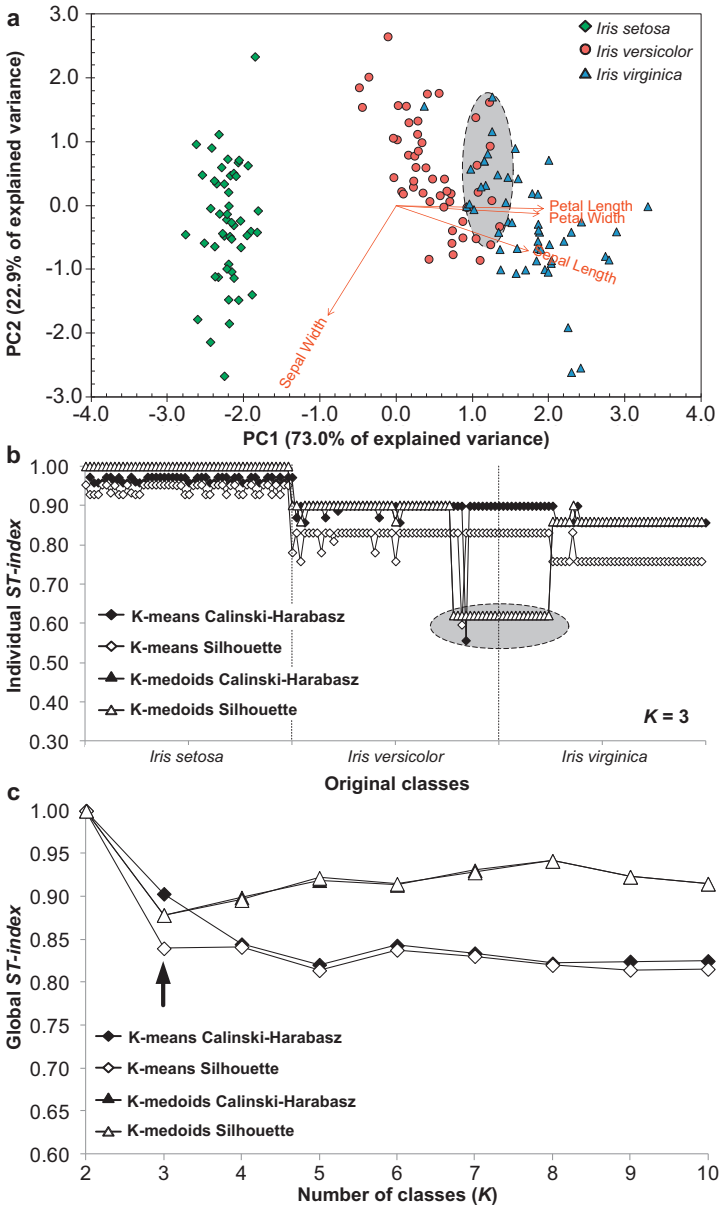

Fig. 4. Analysis of Iris dataset: (a) PCA plot of Iris data. The grey ellipse depicts the overlap of classes *I. virginica* and *I. versicolor*; (b) Variation of the individual *ST*-scores for the taxa of Iris dataset obtained using *K*-means and *K*-medoids and the CH and SI cluster validity indices; (c) Variation of the global *ST*-scores with respect to the number of classes, *K*, shown for both partitioning algorithms and both cluster validity indices. The black arrow indicates the true number of classes for this dataset ($K = 3$).

ments in the Iris dataset (i.e., the 24 elements located in the grey ellipse area on the PCA plot in Fig. 4a; these elements are well noticeable with the *K*-medoids stability analysis based on SI, see Fig. 4b). Moreover, after the elimination of these 24 most unstable elements identified on the first run of our algorithm, and then rerunning it with the remaining 126 elements, we obtained the highest possible value of the global stability index ($ST_{global} = 1.0$) using both *K*-means and *K*-medoids (with both CH and SI) for the solutions with 3 classes.

4.4. Analysis of 21 benchmark datasets using different clustering algorithms

In this section, we evaluate the ability of the *K*-means and *K*-medoids partitioning algorithms based on the maximum of the global *ST*-index to determine the true number of clusters in real datasets from the UCI Machine Learning Repository

Table 3
Datasets' main characteristics and the numbers of classes obtained for the 21 selected benchmark datasets from UCI [3] by *X*-means [38], Pvclust [46], Adegenet [21,22], Prediction strength [47] and Nselectboot [16]. The last row reports the mean absolute differences between the true and predicted numbers of classes as well as the corresponding standard deviation.

Dataset	Characteristics			Clustering methods				
	Number of objects	Number of features	Number of classes	*X*-means[a]	Adegenet[b]	Pvclust[c]	Prediction strength[d]	Nselectboot[e]
Breast tissue	106	9	6	3	6	2	2	13
B. Wisconsin	569	30	2	3	4	3	2	2
Ecoli	336	7	8	3	3	20	3	3
Glass	214	9	7	5	3	2	2	2
Haberman	306	3	2	2	3	3	1	20
Ionosphere	351	34	2	3	8	20	2	20
Iris	150	4	3	2	3	7	2	2
Move. libras	360	90	15	4	3	2	1	20
Musk	476	166	2	4	3	2	3	2
Parkinson	195	22	2	4	7	2	1	8
Segmentation	310	19	7	3	6	2	1	20
Sonar all	208	60	2	4	6	20	1	20
Spectf Heart	267	44	2	2	6	20	1	2
Transfusion	748	4	2	2	4	3	1	20
Vehicule	946	18	4	4	3	20	2	2
Vert. column	310	6	3	2	3	5	1	20
Vowel context	990	10	11	4	3	10	4	16
Wine	178	13	3	3	3	2	2	19
Wine qual. red	1599	11	6	2	3	20	3	4
Yeast	1484	8	10	2	4	20	2	20
Zoo	101	17	7	5	6	2	1	20
Mean absolute difference ± SD				2.66 ± 2.95	3.14 ± 3.10	7.10 ± 6.61	3.33 ± 3.45	8.52 ± 6.88

[a] 100 replicates.
[b] 100 replicates of the find.clusters function.
[c] 1000 bootstrap replicates using the Ward agglomerative method.
[d] 100 replicates using 100 divisions of the dataset with *K*-means.
[e] 100 replicates and 1000 resampling steps with *K*-means.

[3] and compare their results to those produced by several popular clustering algorithms. The main characteristics of the 20 UCI benchmark datasets recently analyzed by Arbelaitz et al. [2] and the above-discussed Zoo dataset [17] are presented in the first three columns of Tables 1–3. The *K*-means and *K*-medoids algorithms were carried out using the Calinski–Harabasz, Silhouette, Dunn and Davies–Bouldin cluster validity indices. Clustering solutions obtained for the range of 2 to 20 classes were examined.

First, we compared the results of the stability-based *K*-means and *K*-medoids algorithms with those provided by conventional *K*-means [29] and *K*-medoids [23], as well as the bootstrapping and jittering techniques proposed by Hennig [18]. The jittering method was carried out with the jittering levels of 0.25, 0.50 and 0.75, and the best results, corresponding to the jittering level of 0.75, were reported. The bootstrapping and jittering methods were performed by the *clusterboot* function of the *fpc* R package [20] using the default parameters. The results of these simulations are presented in Tables 1 and 2, and Supplementary Tables 2 to 5. The last row of these tables reports the average absolute difference between the true number of classes in the datasets and the number of classes found by each method. Our results suggest that the proposed global stability index allows for improving the clustering performances of the traditional *K*-means and *K*-medoids algorithms in terms of determining the true number of classes in a dataset. The new method also outperforms both bootstrapping and jittering techniques. In particular, the lowest average absolute difference between the true and estimated numbers of classes and the lowest standard deviation were obtained using the *ST*-index with the CH (2.19 ± 2.50), SI (2.19 ± 2.56) and Dunn (2.19 ± 2.72) cluster validity indices for *K*-means (Supplementary Table 4), and with the DB index (2.38 ± 2.37) for *K*-medoids (Supplementary Table 5). The best overall performances of our stability-based algorithms (in comparison to the other methods) were obtained for the Blood transfusion, Wine quality red, and Zoo datasets. Overall, the results presented in Supplementary Tables 4 and 5 show that there was no significant difference in terms of determining the number of clusters by the global *ST*-index with respect to the cluster validity index (CH, SI, Dunn or DB). This was particularly evident for the *K*-medoids algorithm.

Second, we compared the stability-based *K*-means and *K*-medoids procedures with the popular *X*-means [38] and Pvclust [46] algorithms, which can automatically determine the number of clusters in a dataset. Moreover, our simulations also included the following well-known clustering methods: Discriminant analysis of principal components (from the *Adegenet* package) of Jombart et al. [21,22], Prediction Strength cluster validation method of Tibshirani and Walther [47], and Bootstrap stability procedure of Fang and Wang [16]. This comparison was also made using the 21 previously discussed benchmark datasets from the UCI Machine Learning Repository [3]. The obtained results are reported in Table 3.

The *X*-means algorithm is a version of *K*-means that efficiently searches the space of cluster locations in order to optimize either Bayesian Information Criterion (BIC) or Akaike's Information Criterion (AIC) [38]. In Table 3, we present the results

obtained by *X*-means using its C++ implementation by Pelleg and Moore [38]. The default parameters of Pelleg's program were used here (except the minimum and maximum numbers of classes that were set to 2 and 20, respectively). Our results indicate that *X*-means favors small numbers of classes, providing the mean absolute difference of 2.66 between the true and estimated numbers of classes (versus 2.19 yielded by our stability-based *K*-means).

The Adegenet algorithm implemented in the function *find.clusters* of R uses sequential *K*-means and discriminant analysis of principal components to compute a *BIC* score for each clustering [21,22]. Then, the clustering corresponding to the minimum *BIC* score is selected as optimal. As reported in Table 3, Adegenet was able to correctly identify the number of classes in some datasets, namely Breast tissue, Iris, Vertebral column, and Wine. However, it provided a higher mean absolute deviation (equal to 3.14) from the true number of classes compared to *X*-means and our stability-based *K*-means.

The popular Pvclust procedure of Suzuki and Shimodaira [46] relies on the calculation of *p*-values for each cluster using bootstrap resampling techniques applied to hierarchies. Two types of *p*-values are available in Pvclust: approximately unbiased *p*-values and bootstrap probability values. The Pvclust program was executed with the following options: 1000 bootstrap replicates for each random start of the program, the Ward agglomerative method, the significance threshold $\alpha = 0.95$ and 100 random starts for each dataset (the minimum number of significant clusters found over 100 random starts is shown in Table 3). Overall, the Pvclust procedure overestimated the true number of classes in our benchmark datas. Pvclust only identified the correct number of classes for the Musk and Parkinson datasets. The mean absolute deviation from the true number of classes was equal to 7.1 for this method.

The two remaining clustering methods tested in our simulations are implemented in Hennig's *fpc* package [20]. The first of them is the Prediction Strength algorithm of Tibshirani and Walther [47]. This method views clustering as a supervised classification problem in which it assesses the true class labels. The resulting *prediction strength* measure estimates how many classes can be predicted from the data, and how well it can be done. It is worth noting that the Prediction Strength algorithm can return solutions consisting of one class only (regardless of the minimum number of classes given as input to the program). The *K*-means algorithm was set as the clustering method. The parameter accounting for the maximum number of times that the dataset could be divided into two halves was set to 100, and the number of random starts was also set to 100 for each dataset. All other program parameters were the default parameters. The Prediction Strength algorithm found 9 one-cluster datasets among the 21 benchmark datasets tested in our simulations, while the mean absolute difference between the predicted and the true number of classes was equal to 3.33 for this method.

Finally, we also experimented with the Nselectboot stability-based clustering technique proposed by Fang and Wang [16]. Multiple paired bootstrap samples are randomly drawn in Nselectboot and then the clustering solution that minimizes the instability estimate calculated from these pairs is selected as optimal. The number of resampling steps was set to 1000 in the Nselectboot function of the *fpc* package, and the number of random starts was set to 100. All other parameters were the default parameters of the program. This bootstrap-based algorithm highly overestimated the true number of classes in several benchmark datasets, which resulted in the highest mean absolute difference between the predicted and the true number of classes (equal to 8.52) and the highest standard deviation (equal to 6.88) among all the methods tested in our simulations.

Fig. 5 summarizes the performances of the 17 clustering algorithms compared in our simulations with real data. Here we adopted graphical representation of Arbelaitz et al. [2]. In terms of the mean absolute difference between the true and predicted numbers of classes (Fig. 5a), the stability-based *K*-means and *K*-medoids algorithms outperformed its competitors regardless of the cluster validity index (CH or SI) used to select the number of classes. Their closest competitors here were traditional *K*-means [29] using SI, the *K*-medoid-based bootstrapping of Hennig [18], and the *X*-means algorithm of Pelleg and Moore [38]. In terms of the success rate (Fig. 5b), which is the percentage of correct guesses (i.e., when the number of classes was predicted correctly) made by each algorithm, the best overall results were achieved by our stability-based *K*-means using SI and the *K*-medoid-based bootstrapping. These methods were followed by traditional *K*-means and our stability-based *K*-means using CH. The *K*-medoids-based stability methods were not among the top performers in terms of the success rate.

Furthermore, we examined how many random starts of a partitioning algorithm are generally required for the convergence of the *ST*-index function. Supplementary Table 6 shows that 100 to 250 random starts were usually necessary for the *ST*-index convergence when running *K*-medoids, whereas 250 to 1000 random starts were generally required for the method's convergence with *K*-means. Steinley [42] observed a rapid decline in the number of local optima when performing 5000 random starts of *K*-means for both real and simulated datasets. He recommended using a combination of ARI and an approximation of the number of local optima to determine the required number of random starts. In the future, it would be interesting to investigate whether this approach could be applied for determining the number of random starts necessary for the convergence of the global *ST*-index.

5. Experiments with synthetic data

To further examine the properties of the individual and global *ST*-indices, we carried out a simulation with synthetic datasets that included various levels of noise and outliers. We first used the data generator of Milligan [32] to create 250 random datasets with 10 to 200 objects, 2 to 10 variables (dimensions or features), and 2 to 5 clusters; 250 datasets were generated for each parameter combination. Each dataset was represented by an object-by-variable matrix $\mathbf{Y}$, where y_{ij} was the value of the jth variable of the ith object.

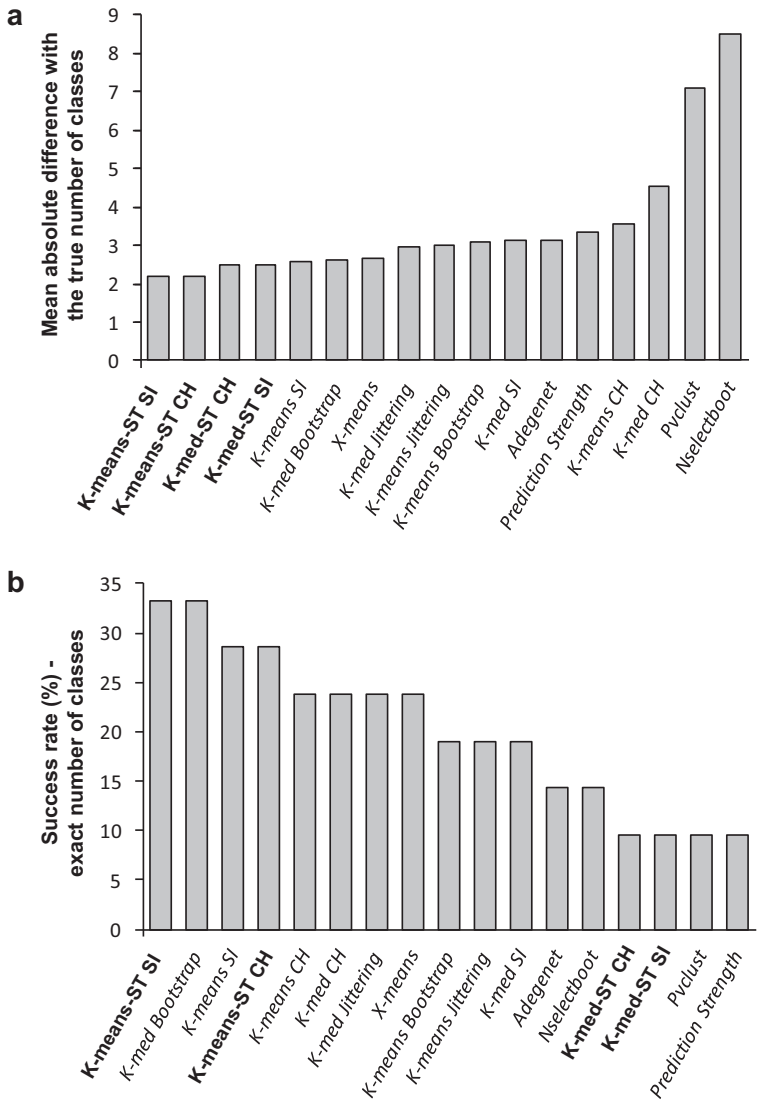

Fig. 5. Comparison of 17 clustering algorithms on the 21 benchmark dataset from the UCI repository presented in Tables 1–3. (a) Mean absolute difference between the true and predicted numbers of classes. (b) Success rate (given in %), accounting for the number of times the correct number of classes was found by each algorithm. The following clustering algorithms were compared: Traditional *K*-means and *K*-medoids using the Calinski–Harabasz (CH) and Silhouette (SI) cluster validity indices (*K-means CH, K-means SI, K-med CH* and *K-med SI*), *K*-means and *K*-medoids bootstrapping and jittering techniques of Hennig (*K-means Bootstrap, K-means Jittering, K-med Bootstrap* and *K-med Jittering*), *X*-means of Pelleg (*X-means*), Pvclust of Suzuki and Shimodaira (*Pvclust*), Adegenet of Jombart *et al.* (*Adegenet*), Prediction Strength of Tibshirani and Walther (*Prediction Strength*), Nselectboot of Fang and Wang (*Nselectboot*), and our stability-based *K*-means and *K*-medoids using the CH and SI indices (K-means-ST CH, K-means-ST SI, K-med-ST CH and K-med-ST SI – shown in bold in the figure). For the sake of clarity, the results of stability-based *K*-means and *K*-medoids obtained using the Dunn and Davies–Bouldin indices were not presented here (for more details, see Supplementary Tables 4 and 5).

Different types of noise were then added to **Y** (Supplementary Fig. 1). Following the simulation protocol used by Milligan [32] and by Makarenkov and Legendre [30], we considered the four following noise generation schemes:

(1) Error-free data: Original matrix **Y** without any noise or perturbation in the data (Supplementary Fig. 1a).
(2) Outliers: We added 40% of outliers to the original data. For each cluster, the outliers were randomly generated objects drawn from a distribution with a standard deviation three times higher than the distribution of the cluster. Moreover, all the dimensions of the outliers were not allowed to fall within the boundaries of any original cluster (Supplementary Fig. 1b).
(3) Error-perturbed data: This type of error involves the perturbation of **Y** by the addition of noise $\mathbf{Noise} = (noise_{ij})$ to all of its dimensions (Supplementary Fig. 1c). The noise, $noise_{ij}$, was randomly drawn from a standard normal distribution.

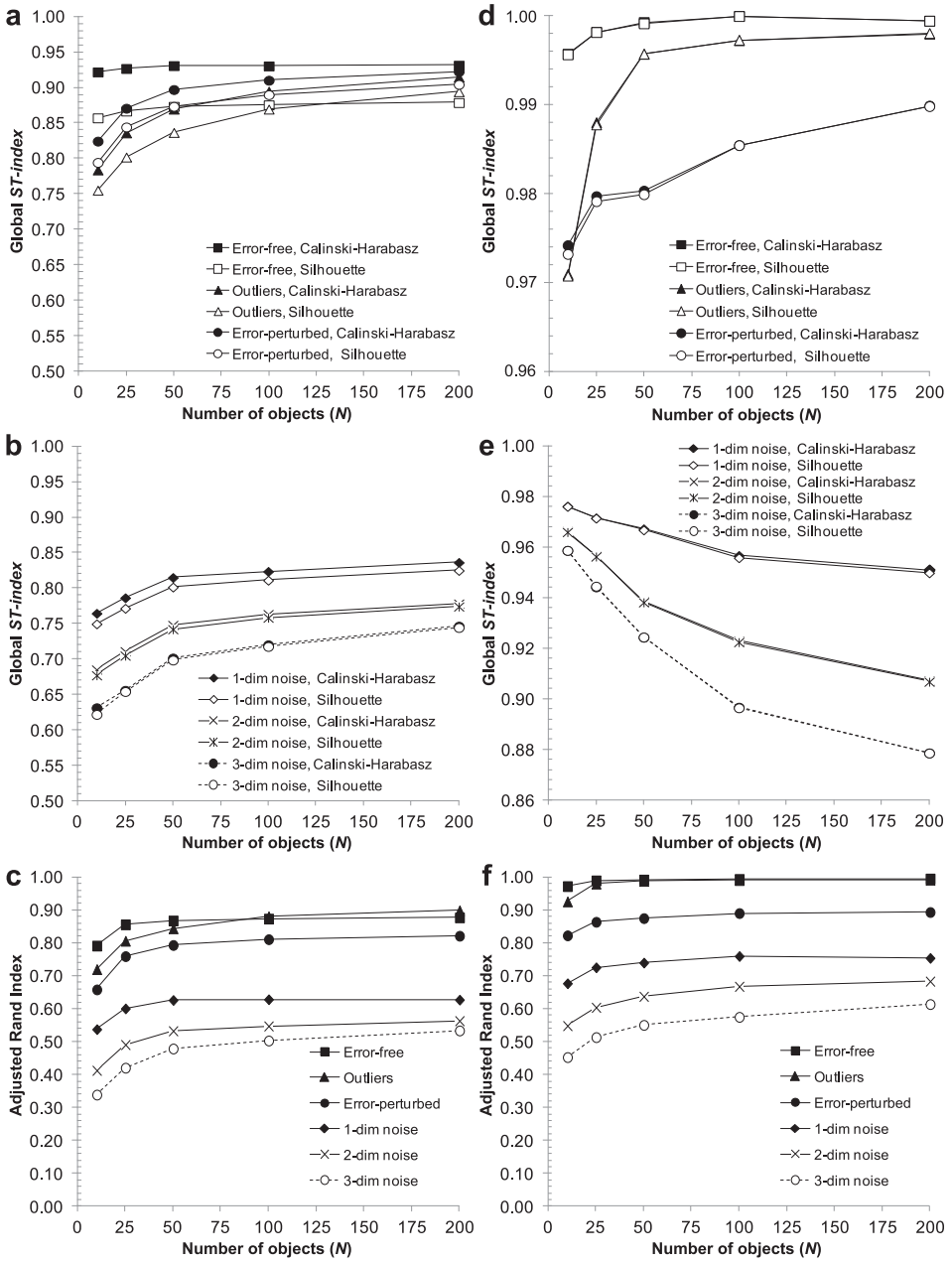

Fig. 6. Variation of the average global stability index (*ST*-index; panels a and d - error-free data, original data with outliers and error-perturbed data; panels b and e - original data with the addition of 1 to 3 noise features) and the Adjusted Rand Index (ARI; panels c and f) with respect to the noise conditions. The presented results are the averages obtained over 1000 replicates. The left column panels (a-c) depict the results obtained using *K*-means, and the right column panels (d-f) depict the results obtained using *K*-medoids. The results obtained with the CH and SI cluster validity indices are shown.

The resulting matrix $\mathbf{Y'} = (y'_{ij})$ was computed as follows:

$$y'_{ij} = y_{ij} + L \times noise_{ij}, \tag{12}$$

where the constant L was set to 2.0 as suggested in [38].

(4) 1-, 2-, 3-dimensional noise: In this case, 1, 2, or 3 noise variables were added to all objects of **Y**. These variables were generated in the same range as the first dimension of **Y**, for which the cluster overlap was not allowed. Again, these additional variables were drawn from a normal distribution with zero mean and the standard deviation equal to that of the first dimension of **Y** (Supplementary Fig. 1d).

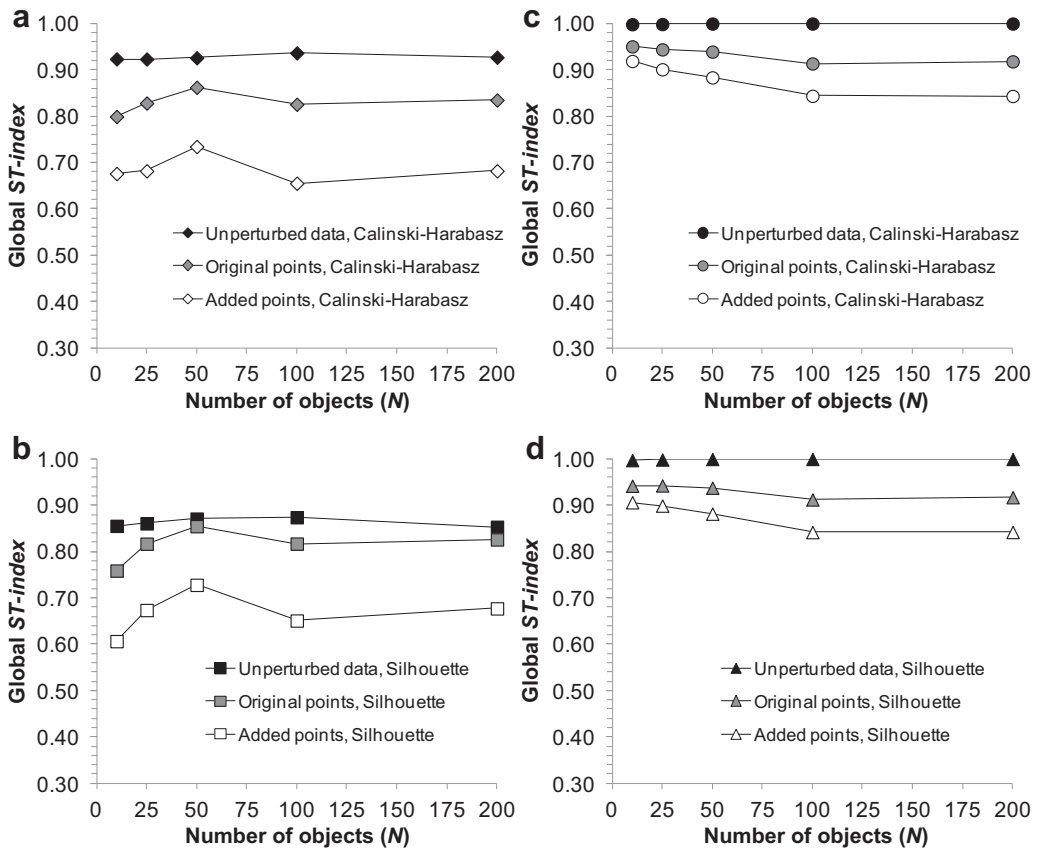

Fig. 7. Variation of the average individual stability indices for error-free data, original objects (after the addition of outliers) and outliers. The presented results are the averages obtained over 1000 replicates. The left column panels (a and b) depict the results obtained using *K*-means, and the right column panels (c and d) depict the results obtained using *K*-medoids. The results for the CH (a and c) and SI (b and d) cluster validity indices are shown.

Z-score normalization was performed for all the variables in **Y**. Using the *K*-means and *K*-medoids partitioning algorithms with the CH and SI cluster validity indices, we calculated the average values of the global *ST*-index and the Adjusted Rand Index (ARI) (Fig. 6); 1000 random starts of partitioning algorithms were performed for each synthetic dataset we examined. In the case of error-free data, the *K*-medoids partitioning (Fig. 6d) generally provided higher cluster stability than the *K*-means partitioning (Fig. 6a). In the framework of the *K*-means partitioning, the addition of noise and outliers to the original data caused a clear decline in stability (Fig. 6a), as well as in ARI (Fig. 6a), especially when the number of objects was lower than 100. Indeed, the *K*-means algorithm is known to be very sensitive to noise and outliers [18,19]. The stability of *K*-means (Fig. 6b) was also more affected by the addition of 1 to 3 noise features than the stability of *K*-medoids (Fig. 6e). Interestingly, the increase in the number of objects led to the increase in the values of both global *ST*-index and ARI in all cases (except the *K*-medoids partitioning with 1 to 3 noisy features) (Fig. 6e).

Moreover, we evaluated the stability of individual objects in the case when 40% of outliers were added to the error-free data. In particular, we separately calculated the average individual stability of the original objects and that of the outliers. The three curves depicted in Fig. 7 represent the averages obtained for error-free data (original data before adding the outliers), the original data only (after the outliers were added to the data), and the outliers only. For both partitioning methods (see panels a and b of Fig. 7 for *K*-means, and panels c and d for *K*-medoids) and both cluster validity indices (CH and SI) tested in these simulations, the original objects provided greater average stability scores than outliers. It is worth noting, however, that the individual stability of the outliers that were located far away from the clusters of objects were sometimes higher than the stability of the original objects in these clusters; but the individual stability of the outliers located between two well-defined clusters, and thus sharing similarities with both of them, were always much lower than the stability of the original objects (see Supplementary Fig. 1b).

Finally, we carried out simulations using the noise generation procedure recommended by Hennig [18]. The original (i.e., error-free) datasets were generated as described above (see also Milligan [32]). Hennig's procedure allows for the replacement of some randomly chosen original objects by objects drawn from a noise distribution. In our simulations, we used uniformly distributed noise data with a range of [−3,3] and the following proportions of replaced objects: 0%, 5%, 10%, 25%,

50%, 75%, and 100%. Alternatively, in the jittering procedure, a small amount of normally distributed noise was added to any single object [18]. The jittering level q, varying between 0 and 1, corresponds to the level of noise. Supplementary Fig. 2 illustrates the variation of the global *ST*-index (panels a and b) and ARI (panels c and d) in the framework of *K*-means partitioning. The results presented in this figure outline a very similar behavior for both considered indices. A gradual decrease of the values of both *ST*-index and ARI can be observed with the increase in the level of noise (Supplementary Figs. 2a and c), whereas a drastic drop in their values in jittering starts from the jittering level of 0.75 (Supplementary Figs. 2b and d).

6. Conclusion

In this paper, we have introduced a novel method for assessing the robustness of clustering solutions provided by partitioning algorithms. We have shown how the stability of individual objects, clusters, and entire clustering solutions can be estimated based on a series of repeated runs of a partitioning algorithm. To compute our stability indices, we first determine the pairwise support scores of the objects. These scores are defined taking into account the quality of the obtained clusterings expressed by the selected cluster validity index (CH, SI, Dunn and DB in our study). The stability indices are then defined based on the proportion of returned partitions in which pairs of objects belong to the same class, and then making a correction for chance co-occurrence.

We have demonstrated that the proposed stability indices can effectively identify the most stable and the most unstable elements in complex datasets. As we have seen with the example of the Iris data, the removal of the most unstable clustering elements leads to a better identification of the true number of clusters in datasets. Moreover, the results of our simulations indicate that the use of our stability indices allows for the improved recovery of the number of clusters by traditional clustering methods (the conventional *K*-means and *K*-medoids algorithms in this study). Our method also compares advantageously to the well-known bootstrapping and jittering techniques [18,19]. Finally, this new method makes it possible to recognize correct clustering patterns even in the presence of noise and outliers. Generally, our stability analysis can identify the outliers as the most unstable clustering elements, which are then removed from the data.

One possible extension of the presented method is the calculation of support scores based on triplets and quadruplets of objects. Additionally, our stability indices can be easily generalized and applied in the framework of fuzzy clustering algorithms, such as *C*-means [8] or OKM [5]. A new R package, called *ClusterStability*, which is freely distributed through the CRAN repository, includes the implementation of the discussed cluster stability indices and the related clustering methods.

Conflict of interest

None declared.

Acknowledgements

The authors thank Drs. Renato Cordeiro de Amorim and Witold Pedrycz for their helpful comments. This work was funded by Natural Sciences and Engineering Research Council of Canada [Grant Number 249644] and Le Fonds Québécois de la Recherche sur la Nature et les Technologies [Grant Number 173539].

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ins.2017.02.010.

References

[1] E. Anderson, The irises of the Gaspe Peninsula, Bull. Am. Iris Soc. 59 (1935) 2–5.
[2] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, An extensive comparative study of cluster validity indices, Pattern Recognit. 46 (2013) 243–256.
[3] K. Bache, M. Lichman, UCI Machine Learning Repository, University of California, School of Information and Computer Science, Irvine, CA, 2013 http://archive.ics.uci.edu/ml.
[4] S. Ben-David, D. Pál, H. Simon, Stability of k-means clustering, Lect. Notes Comput. Sci. 4539 (2007) 20–34.
[5] C.E. Ben N'Cir, G. Cleuziou, N. Essoussi, Generalization of c-means for identifying non-disjoint clusters with overlap regulation, Pattern Recognit. Lett. 45 (2014) 92–98.
[6] A. Ben-Hur, A. Elisseeff, I. Guyon, A stability based method for discovering structure in clustered data, Pac. Symp. Biocomput. 7 (2002) 6–17.
[7] P. Bertrand, G. Mufti, Loevinger's measures of rule quality for assessing cluster stability, Comput. Stat. Data Anal. 50 (2006) 992–1015.
[8] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, 1981 Plenum, NY.
[9] J.C. Bezdek, N.R. Pal, Some new indexes of cluster validity, IEEE Trans. Syst. Man Cybern. Part B 28 (1998) 301–315.
[10] M. Brun, C. Sima, J. Hua, J. Lowey, B. Carroll, Model-based evaluation of clustering validation measures, Pattern Recognit. 40 (2007) 807–824.
[11] T. Caliński, J. Harabasz, A dendrite method for cluster analysis, Commun. Stat. Theory Methods 3 (1974) 1–27.
[12] S. Chaimontree, K. Atkinson, F. Coenen, Best clustering configuration metrics: towards multiagent based clustering, in: L. Cao, Y. Feng, J. Zhong (Eds.), Advanced Data Mining and Applications, 6th International Conference, ADMA 2010, Springer Verlag, Berlin, Heidelberg, 2010, pp. 48–59.
[13] D.L. Davies, D.W. Bouldin, A cluster separation measure, IEEE Trans. Pattern Anal. Mach. Intell. 2 (1979) 224–227.
[14] R.C. de Amorim, V. Makarenkov, Applying subclustering and Lp distance in weighted K-means with distributed centroids, Neurocomputing 173 (2016) 700–707.
[15] C. Deus, Z. Liao, Statistical consensus method for cluster ensembles, in: Progress in Informatics and Computing (PIC), 2010 IEEE International Conference, Vol. 1, IEEE, 2010, pp. 185–189.
[16] Y. Fang, J. Wang, Selection of the number of clusters via the bootstrap method, Comput. Stat. Data Anal. 56 (2012) 468–477.

[17] R.S. Forsyth, Neural learning algorithms: Some empirical trials, in: Proceedings of the Third Conference on Neural Nets and their Applications, Nanterre, France, 1990, pp. 301–317.
[18] C. Hennig, Cluster-wise assessment of cluster stability, Comput. Stat. Data Anal. 52 (2007) 258–271.
[19] C. Hennig, Dissolution point and isolation robustness: robustness criteria for general cluster analysis methods, J. Multivariate Anal. 99 (2008) 1154–1176.
[20] C. Hennig, FPC: Flexible procedures for clustering, 2013 https://cran.r-project.org/web/packages/fpc/ *R* package version 2.1.
[21] T. Jombart, Adegenet: an R package for the multivariate analysis of genetic markers, Bioinformatics 24 (2008) 1403–1405.
[22] T. Jombart, S. Devillard, F. Balloux, Discriminant analysis of principal components: a new method for the analysis of genetically structured populations, BMC Genet. 11 (2010) 1.
[23] L.R. Kauffman, P. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, first ed., John Wiley &, Sons, Hoboken, New Jersey, 1990.
[24] M. Kim, R.S. Ramakrishna, New indices for cluster validity assessment, Pattern Recognit. Lett. 26 (2005) 2353–2363.
[25] L.I. Kuncheva, D.P. Vetrov, Evaluation of stability of k-means cluster ensembles with respect to random initialization, IEEE Trans. Pattern Anal. Mach. Intell. 28 (2006) 1798–1808.
[26] T. Lange, V. Roth, M.L. Braun, J.M. Buhmann, Stability-based validation of clustering solutions, Neural Comput. 16 (2004) 1299–1323.
[27] S.P. Lloyd, Least squares quantization, IEEE Trans. Inf. Theory 28 (1982) 129–136.
[28] E. Lord, A.B. Diallo, V. Makarenkov, Classification of bioinformatics workflows using weighted versions of partitioning and hierarchical clustering algorithms, BMC Bioinformatics 16 (2015) 68.
[29] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: L. Le Cam, J. Neyman (Eds.), Proceedings 5th Berkeley Symposium On Mathematical Statistics and Probability, University of California Press, Berkeley, 1967, pp. 281–297.
[30] V. Makarenkov, P. Legendre, Optimal variable weighting for ultrametric and additive trees and K-means partitioning: methods and software, J. Classification 18 (2001) 245–271.
[31] D.P. McKenzie, R.S. Forsyth, Classification by similarity: an overview of statistical methods of case-based reasoning, Comput. Hum. Behav. 11 (1995) 273–288.
[32] G.W. Milligan, A validation study of a variable weighting algorithm for luster analysis, J. Classification 6 (1989) 53–71.
[33] G.W. Milligan, R. Cheng, Measuring the influence of individual data points in a cluster analysis, J. Classification 13 (1996) 315–335.
[34] G.W. Milligan, M.C. Cooper, An examination of procedures for determining the number of clusters in a data set, Psychometrika 50 (1985) 159–179.
[35] B. Mirkin, Clustering: A Data Recovery Approach, second ed., Chapman and Hall/CRC Press, Boca Raton, Florida, 2012.
[36] V.H. Moll, Numbers and Functions: From a Classical-Experimental Mathematician's Point of View, Chapter 7, first ed., American Mathematical Society, Providence, Rhode Island, 2012.
[37] W. de Mulder, Instability and cluster stability variance for real clusterings, Inf. Sci. 260 (2014) 51–63.
[38] D. Pelleg, A.W. Moore, X-means: extending K-means with efficient estimation of the number of clusters, in: 17th International Conference on Machine Learning, 2000, pp. 727–734.
[39] P.J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, J. Comput. Appl. Math. 20 (1987) 53–65.
[40] V.N. Sachkov, Probabilistic Methods in Combinatorial analysis, First ed., Cambridge University Press, New York, 1997 Chapter 5.
[41] D. Steinley, Local optima in K-means clustering: what you don't know may hurt you, Psychol. Methods 8 (2003) 294–304.
[42] D. Steinley, Profiling local optima in K-means clustering: developing a diagnostic technique, Psychol. Methods 11 (2006) 178–192.
[43] D. Steinley, Stability analysis in K-means clustering, Br. J. Math. Stat. Psychol. 61 (2008) 255–273.
[44] M. Studer, in: WeightedCluster Library Manual: A Practical Guide to Creating Typologies of Trajectories in the Social Sciences with R, 24, 2013, pp. 1–32.
[45] C.A. Sugar, G.M. James, Finding the number of clusters in a dataset, J. Amer. Statist. Assoc. 98 (2003) 750–763.
[46] R. Suzuki, H. Shimodaira, Pvclust: an R package for assessing the uncertainty in hierarchical clustering, Bioinformatics. 22 (2006) 1540–1542.
[47] R. Tibshirani, G. Walther, Cluster validation by prediction strength, J. Comput. Graph. Statist. 14 (2005) 511–528.
[48] A. Topchy, A.K. Jain, W. Punch, Combining multiple weak clusterings, in: Third IEEE International Conference on Data Mining, IEEE, Melbourne, Florida, 2003, pp. 331–338.
[49] J. Wang, Consistent selection of the number of clusters via crossvalidation, Biometrika 97 (2010) 893–904.
[50] R. Wu, B. Zhang, M. Hsu, Clustering billions of data points using GPUs, in: Proceedings of the Combined Workshops on UnConventional High Performance Computing Workshop Plus Memory Access Workshop, ACM, New York, 2009, pp. 1–6.